Tightly Coupled Range Inertial Localization on a 3D Prior Map Based on Sliding Window Factor Graph Optimization

Kenji Koide[1], Shuji Oishi[1], Masashi Yokozuka[1], and Atsuhiko Banno[1]

***Abstract*— This paper presents a range inertial localization algorithm for a 3D prior map. The proposed algorithm tightly couples scan-to-scan and scan-to-map point cloud registration factors along with IMU factors on a sliding window factor graph. The tight coupling of the scan-to-scan and scan-to-map registration factors enables a smooth fusion of sensor ego-motion estimation and map-based trajectory correction that results in robust tracking of the sensor pose under severe point cloud degeneration and defective regions in a map. We also propose an initial sensor state estimation algorithm that robustly estimates the gravity direction and IMU state and helps perform global localization in 3- or 4-DoF for system initialization without prior position information. Experimental results show that the proposed method outperforms existing state-of-the-art methods in extremely severe situations where the point cloud data becomes degenerate, there are momentary sensor interruptions, or the sensor moves along the map boundary or into unmapped regions.**

I. Introduction

Map-based sensor localization is a crucial function for autonomous systems. Precise positional information enables the reliable navigation and recognition required for many applications, including service robots and autonomous driving. In particular, point-cloud-based localization algorithms have been one of the most popular approaches due to the recent emergence of precise and affordable range sensors, such as LiDARs and time-of-flight depth cameras.

The most naive approach to estimating a sensor pose on a 3D prior map is to iteratively apply fine point cloud registration (e.g., ICP [1] and NDT [2]) between scan point clouds and the map point cloud. Although this naive approach works in many environments, it often becomes unreliable under aggressive sensor motion because these fine registration methods require an accurate initial guess for convergence. To improve the robustness to quick sensor motion, these scan matching methods are often integrated with additional information sources (e.g., IMU [3] and wheel encoders [4]) to better predict the sensor pose and maintain registration results accurate. Furthermore, approaches that fuse point cloud registration errors and other sensor errors on a unified objective function (i.e., tight coupling) enable further robustness to sensor motion and partial degeneration of point clouds [5]. However, it is still challenging to deal with severe point cloud degeneration and interruptions that introduce ambiguity of sensor states. Furthermore, because many existing map-based localization methods perform scan-to-map registration decoupled from ego-motion estimation, they suffer from registration failures and become corrupted at map boundaries and in regions outside the map.

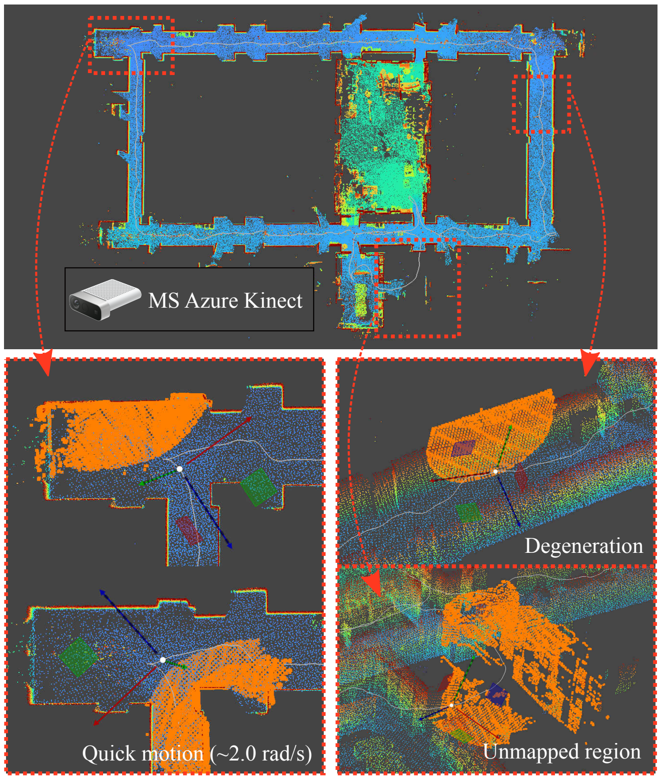


Fig. 1: Indoor localization experiment using a Microsoft Azure Kinect. The proposed method enables robust and accurate pose estimation in challenging situations including under quick sensor motions, complete degeneration of point clouds, and traveling across both mapped and unmapped regions. The color of the map point cloud indicates the height of each point.

In the present paper, we propose a tightly coupled range inertial localization method based on sliding window factor graph optimization by extending our previous odometry estimation algorithm [6] to a map-based localization scenario. Our formulation employs the same point cloud registration error function for scan-to-scan and scan-to-map registration constraints and jointly minimizes them along with IMU constraints in a unified objective function. This tight coupling of scan-to-scan and scan-to-map registration errors enables a smooth transition between mapped and unmapped regions.

*This work was supported in part by JSPS KAKENHI Grant Number 23K16979 and a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

[1]All the authors are with the Department of Information Technology and Human Factors, the National Institute of Advanced Industrial Science and Technology, Tsukuba, Ibaraki, Japan, `k.koide@aist.go.jp`

Unlike filtering-based methods, the proposed method keeps sensor states active while the states remain in a sliding window (e.g., 5 s). Furthermore, the proposed method is robust to cases where the point cloud has become degenerate and existing filtering-based methods suffer from state ambiguity. In addition, we propose a robust state initialization method that estimates the gravity direction of scan point clouds and helps perform 3- or 4-DoF global localization for initial pose estimation. Through experiments, we show that the proposed method enables robust sensor pose estimation in extremely severe situations including traveling across both mapped and unmapped regions and point clouds that have degenerated and have interruptions [1].

This work has three main contributions:

1) We propose a localization approach based on a tight coupling of scan-to-scan registration, scan-to-map registration, and IMU factors. This approach enables a smooth transition between mapped and unmapped regions and makes pose estimation robust to point cloud degeneration and interruptions.
2) We develop a simple and robust gravity direction estimation method based on the batch optimization of sensor poses and IMU measurements. This method allows us to perform global localization in a reduced DoF.
3) We release an evaluation dataset that can be used to evaluate the robustness of map-based localization algorithms in extremely severe situations.

II. Related Work

A. Iterative Scan Matching

Estimation of the position of a sensor on a map is essential for navigation systems. While several kinds of map representations are used, depending on the use scenario (e.g., high-definition vector maps [7] and wiki-based open geographic maps like OpenStreetMap [8], [9]), 3D point cloud maps are among the most popular representations owing to their simplicity and expressiveness. Because constructing a point cloud map is relatively easy with recent precise range sensors and mapping algorithms, point cloud maps are used for a wide range of applications, from indoor service robots to outdoor driving of autonomous vehicles.

For sensor localization on a point cloud map, local point cloud registration methods, such as ICP [1] and NDT [2], are often used. By iteratively applying point cloud registration between scan points and map points, we can easily and efficiently track a 6-DoF sensor pose. A key to obtaining fine registration results is to provide an accurate initial guess to ensure the convergence. For this reason, additional information sources, for example, IMU [3], wheel encoders [4], and leg joint angles of quadruped robots [10], are jointly used with range sensors to compensate for sensor motion and better predict the current sensor pose to be used as an initial guess for point cloud registration. A recent trend in sensor motion estimation is a tight coupling of multi-sensor constraints that fuses the cost functions of several sensors on a unified objective function. This makes it possible to keep the system well constrained when the data of a sensor become unreliable. In particular, the LiDAR-IMU tight coupling approach has been actively explored in recent studies [11], [12], [13]. Because the tight coupling approach significantly increases the computation cost, lightweight iterated Kalman filters are often used [5]. However, these filtering-based approaches suffer from severe degeneration of point cloud data because they immediately marginalize past frames when a new frame arrives and cannot accurately propagate the uncertainty of past observations. Furthermore, in many studies, scan-to-map registration was decoupled from the odometry estimation algorithm, which results in difficulty correcting sensor poses at map boundaries and maintaining system stability outside the map.

B. Monte Carlo Localization

Monte Carlo localization (MCL), which represents and estimates a state distribution with a finite set of state samples, is a popular approach to 2D map-based localization [14], [15]. Owing to its expressiveness for non-linear and non-Gaussian distributions, it is extremely robust to observation ambiguity and multi-hypothesis situations. Despite its success in 2D localization, MCL has not been commonly used in 3D localization problems because the required number of samples to fill a unit space grows exponentially as the number of dimensions increases, which becomes a computational burden. Many studies have performed 3D MCL in 3-DoF [16] or 4-DoF [17] with assumptions on sensor motion and environment structure. Although several studies have tackled 6-DoF MCL with a smaller number of samples using efficient state sampling [18], [19], [20], it is still difficult to perform a general full 6-DoF MCL without such assumptions and a prior knowledge of the sensor and the environment.

III. Methodology

A. Notation

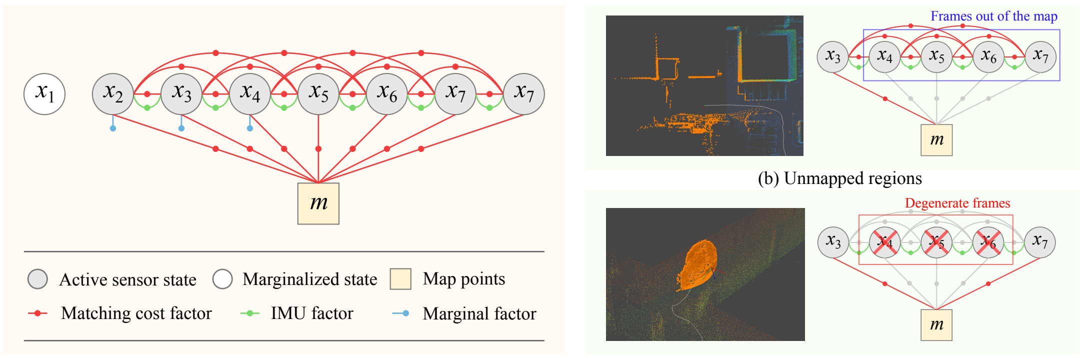
Given a map point cloud $\mathcal{M} = \{\boldsymbol{p}_k^M \in \mathbb{R}^3 \mid_{k=1,\dots,N^M}\}$, we estimate a time series of sensor states $\boldsymbol{x}_t$ in the map frame using measurements of point cloud scans $\mathcal{P}_t = \{\boldsymbol{p}_k^S \in \mathbb{R}^3 \mid_{k=1,\dots,N^P}\}$ and IMU data $\mathcal{I}_t = [\boldsymbol{a}_t, \boldsymbol{\omega}_t]$, where $\boldsymbol{a}_t \in \mathbb{R}^3$ and $\boldsymbol{\omega}_t \in \mathbb{R}^3$ are the linear acceleration and the angular velocity, respectively. The sensor state to be estimated is defined as:

$$\boldsymbol{x}_t = [\boldsymbol{T}_t, \boldsymbol{v}_t, \boldsymbol{b}_t], \quad (1)$$

where $\boldsymbol{T}_t = [\boldsymbol{R}_t \mid \boldsymbol{t}_t] \in SE(3)$ and $\boldsymbol{v} \in \mathbb{R}^3$ are respectively the sensor pose and velocity in the map frame, and $\boldsymbol{b}_t = [\boldsymbol{b}_t^a, \boldsymbol{b}_t^\omega] \in \mathbb{R}^6$ is the IMU acceleration and angular velocity bias. Note that we transform scan points into the IMU coordinate frame and consider them as if they are in the same coordinate frame for efficiency and simplicity.

In the following Sec. III-B and III-C, we describe the building blocks of the proposed framework, namely the point cloud registration factor and the preintegrated IMU factor,

[1]See the project page for supplementary videos: `https://staff.aist.go.jp/k.koide/projects/icra2024_gl/`



(a) Factor graph structure

(b) Unmapped regions

(c) Point cloud degeneration

Fig. 2: Proposed factor graph structure. (a) The proposed factor graph consists of tightly coupled scan-to-scan and scan-to-map registration factors along with IMU factors. Old frames that leave the optimization window are marginalized to bound the computation cost. The proposed method enables robust pose estimation in challenging situations including (b) the sensor travels over unmapped places, and (c) point cloud data becomes degenerated.

and then present the proposed factor graph structure in Sec. III-D.

B. Point Cloud Registration Factor

To constrain the relative pose between point clouds $\mathcal{P}_i$ and $\mathcal{P}_j$, we use the voxelized GICP (VGICP) registration error factor with GPU acceleration [6]. As with GICP [21], VGICP models each point as a Gaussian distribution $\boldsymbol{p}_k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{C}_k)$ representing the local geometrical shape and computes a sum of the distribution-to-distribution distances between corresponding points as follows:

$$e^{\mathrm{PC}}(\mathcal{P}_i, \mathcal{P}_j, \boldsymbol{T}_i, \boldsymbol{T}_j) = \sum_{\boldsymbol{p}_k \in \mathcal{P}_i} e^{\mathrm{D2D}}(\boldsymbol{p}_k, \boldsymbol{T}_i^{-1}\boldsymbol{T}_j), \tag{2}$$

$$e^{\mathrm{D2D}}(\boldsymbol{p}_k, \boldsymbol{T}_{ij}) = \boldsymbol{d}_k^\top \left(\boldsymbol{C}'_k + \boldsymbol{T}_{ij}\boldsymbol{C}_k\boldsymbol{T}_{ij}^\top \right)^{-1} \boldsymbol{d}_k, \tag{3}$$

where $\boldsymbol{T}_{ij} = \boldsymbol{T}_i^{-1}\boldsymbol{T}_j$ is the relative transformation between $\mathcal{P}_i$ and $\mathcal{P}_j$, $\boldsymbol{p}'_k \sim \mathcal{N}(\boldsymbol{\mu}'_k, \boldsymbol{C}'_k)$ is the point in $\mathcal{P}_j$ nearest to $\boldsymbol{p}_k$, and $\boldsymbol{d}_k = \boldsymbol{\mu}'_k - \boldsymbol{T}_{ij}\boldsymbol{\mu}_k$ is the residual between $\boldsymbol{\mu}'_k$ and $\boldsymbol{\mu}_k$. The covariance matrix $\boldsymbol{C}_k$ is computed from neighboring points of $\boldsymbol{p}_k$. To avoid a costly nearest neighbor search, VGICP voxelizes the target point cloud $\mathcal{P}_j$ at a specific resolution r and stores the average of means and covariance matrices of points in each voxel. During cost evaluation, it looks up a voxel corresponding to each input point and computes Eq. 3 using the voxel as $\boldsymbol{p}'_k$.

C. Preintegrated IMU Factor

To efficiently incorporate IMU measurements into the factor graph, we use the preintegration technique [22]. Given an IMU measurement ($\boldsymbol{a}_t$ and $\boldsymbol{\omega}_t$), the sensor states evolves over time as follows:

$$\boldsymbol{R}_{t+\Delta t} = \boldsymbol{R}_t \exp\left(\left(\boldsymbol{\omega}_t - \boldsymbol{b}_t^{\omega} - \boldsymbol{\eta}_k^{\omega} \right) \Delta t \right), \tag{4}$$

$$\boldsymbol{v}_{t+\Delta t} = \boldsymbol{v}_t + \boldsymbol{g}\Delta t + \boldsymbol{R}_t \left(\boldsymbol{a}_t - \boldsymbol{b}_t^{a} - \boldsymbol{\eta}_t^{a} \right) \Delta t, \tag{5}$$

$$\boldsymbol{t}_{t+\Delta t} = \boldsymbol{t}_t + \boldsymbol{v}_t\Delta t + \frac{1}{2}\boldsymbol{g}\Delta t^2 + \frac{1}{2}\boldsymbol{R}_t \left(\boldsymbol{a}_t - \boldsymbol{b}_t^{a} - \boldsymbol{\eta}_t^{a} \right) \Delta t^2, \tag{6}$$

where $\boldsymbol{g}$ is the gravity vector and $\boldsymbol{\eta}_t^a$ and $\boldsymbol{\eta}_t^{\omega}$ represent white noise in the IMU measurement.

The IMU preintegration factor integrates Eqs. 4 – 6 between times i and j to obtain the relative sensor motion $\Delta\boldsymbol{R}_{ij}$, $\Delta\boldsymbol{t}_{ij}$, and $\Delta\boldsymbol{v}_{ij}$ [22]. Then, the IMU prediction error is calculated as follows:

$$\begin{aligned} e^{IMU}(\boldsymbol{x}_i, \boldsymbol{x}_j) = & \,\| \log\left(\Delta\boldsymbol{R}_{ij}^\top \boldsymbol{R}_i^\top \boldsymbol{R}_j \right) \|^2 \\ & + \| \Delta\boldsymbol{t}_{ij} - \boldsymbol{R}_i^\top \left(\boldsymbol{t}_j - \boldsymbol{t}_i - \boldsymbol{v}\Delta t_{ij} - \frac{1}{2}\boldsymbol{g}\Delta t_{ij}^2 \right) \|^2 \\ & + \| \Delta\boldsymbol{v}_{ij} - \boldsymbol{R}_i^\top \left(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g}\Delta t_{ij} \right) \|^2. \end{aligned} \tag{7}$$

Because IMU measurements provide a constant amount of sensor motion information independent of the environment structure, they help keep the factor graph well-constrained in environments where point cloud registration factors can be degenerate (e.g., tunnels and long corridors).

D. Factor Graph Structure

Fig. 2 (a) illustrates the proposed factor graph structure, which consists of four major elements: scan-to-scan registration factors, scan-to-map registration factors, IMU factors, and marginal factors.

Ego-motion estimation: To estimate the sensor ego-motion, scan-to-scan registration factors are created between scan point clouds. To make the estimation robust to quick sensor motion, we create scan-to-scan registration factors between the latest frame and N^{pre} preceding frames (e.g., 3 frames), which results in a densely connected factor graph. IMU factors are created between consecutive frames to help predict the sensor pose and keep the factor graph well-constrained under point cloud degeneration.

Map-based drift correction: To suppress estimation drift, we create scan-to-map registration factors between every frame and the map point cloud. We treat the map point cloud as a static frame and create unary registration factors that constrain only the frame poses. The key idea here is to use the same error function for scan-to-scan and scan-to-map registration factors and jointly minimize them. This structure makes it possible to maintain sensor pose tracking on map boundaries and even in unmapped regions. As shown in Fig. 2 (b), even if a frame has only partial overlap with the map and the scan-to-map registration factor becomes degenerate, the latest frame is still well-constrained by the scan-to-scan registration factors, and we can safely incorporate the degenerate scan-to-map registration errors into the factor graph. Furthermore, when there is absolutely no overlap with the map, the factor graph simply falls back to a range inertial odometry estimation and maintains sensor pose tracking. Once the sensor comes back to the map, scan-to-map registration errors are re-activated and smoothly correct the estimation drift while retaining the consecutive frame matching consistency. The proposed graph structure also makes the estimation robust to degeneration of point cloud data. As in Fig. 2 (c), the tightly coupled scan-to-map registration factors enable deficient relative pose constraints (5-DoF with ambiguity in the corridor direction in the shown case) to be incorporated into the factor graph to reduce estimation errors.

Optimization: To bound the computation cost, we marginalize old frames that leave the optimization window (e.g., 5 s) and add constant linear factors to compensate for the marginalized factors at the last evaluation point. For factor graph optimization and marginalization, we use the iSAM2 optimizer [23] and its efficient Bayes tree elimination implemented in GTSAM [24].

In summary, the objective function of the proposed framework is defined as follows:

$$\begin{aligned} e(\mathcal{X}) = & \sum_{\boldsymbol{x}_i \in \mathcal{X}} \sum_{j=i-N^{\text{pre}}}^{i-1} e^{\text{PC}}(\mathcal{P}_i, \mathcal{P}_j, \boldsymbol{T}_i, \boldsymbol{T}_j) \\ & + \sum_{\boldsymbol{x}_i \in \mathcal{X}} e^{\text{PC}}(\mathcal{P}_i, \mathcal{M}, \boldsymbol{T}_i, \mathbb{I}^{4\times4}) \\ & + \sum_{\boldsymbol{x}_i \in \mathcal{X}} e^{\text{IMU}}(\boldsymbol{x}_{i-1}, \boldsymbol{x}_i) + \mathcal{C}, \end{aligned} \tag{8}$$

where $\mathcal{X}$ is a set of sensor states in the optimization window and $\mathcal{C}$ is a set of linear factors for marginalization.

E. Gravity Direction Estimation

In practice, the gravity direction is useful information that allows aligning the upward directions of scan and map point clouds and enables the search space to be narrowed down for global localization in 4-DoF [25] or 3-DoF with an additional sensor height assumption [26]. Although the gravity direction can easily be obtained using linear acceleration data when the sensor is stationary, the estimated gravity direction can be affected by sensor motion.

To aid global localization used for initial pose estimation, we developed a simple and robust gravity direction and IMU state estimation method based on batch optimization. We first estimate the sensor trajectory using only point cloud data. To this end, we use a combination of the continuous time ICP (CT-ICP) [27] and the voxel-based points container structure (linear iVox) [11]. Given the estimated sensor trajectory and IMU measurements in a certain optimization window (e.g., 2 s), we create a factor graph that consists of relative pose factors based on the estimated trajectory and preintegrated IMU factors between consecutive frames. The objective function is defined as follows:

$$e^{\text{Init}}(\mathcal{X}) = \sum_{\boldsymbol{x}_i \in \mathcal{X}} \left(e^{\text{RP}}(\boldsymbol{x}_{i-1}, \boldsymbol{x}_i) + e^{\text{IMU}}(\boldsymbol{x}_{i-1}, \boldsymbol{x}_i) \right), \tag{9}$$

where $e^{\text{RP}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \| \log(\widetilde{\boldsymbol{T}}_{ij}^{-1} \boldsymbol{T}_i^{-1} \boldsymbol{T}_j) \|^2$ is the relative pose constraint based on the relative pose measurement $\widetilde{\boldsymbol{T}}_{ij}$ computed from the estimated sensor trajectory. Because this objective function has ambiguity in the translation and yaw-axis rotation, we add a large constant to the diagonal elements of the information matrix of the first pose to fix the gauge freedom and maintain system positive definiteness. As an initial estimate of the very first frame, we compute the average linear acceleration vector and rotate the first frame such that the average linear acceleration vector is aligned with the world gravity direction. The initial rotation of the first frame is given by

$$\boldsymbol{\psi} = \boldsymbol{\mu}^a \times \boldsymbol{g}^w, \tag{10}$$

$$\boldsymbol{R}_0 = \mathbb{I}^{6\times6} + \widehat{\boldsymbol{\psi}} + \widehat{\boldsymbol{\psi}}^2 \frac{(1 - \boldsymbol{\mu}^a \cdot \boldsymbol{g}^w)}{|\boldsymbol{\psi}|^2}, \tag{11}$$

where $\boldsymbol{\mu}^a$ is the normalized average linear acceleration vector, $\boldsymbol{g}^w = [0, 0, 1]^T$ is the gravity direction in the world frame, and $\widehat{\cdot}$ is the hat operator to compute the skew symmetric matrix. We use the Levenberg-Marquardt optimizer to minimize Eq. 9. After optimization, we use the optimized sensor pose, velocity, and IMU bias of the last frame as the initial state of the localization system.

To demonstrate the usefulness of the gravity direction information for automatic system initialization, we implemented a 2D occupancy gridmap-based global localization algorithm using branch-and-bound search [28] [2]. In outdoor experiments shown in Sec. IV-B, we obtain a 2D slice of scan point clouds using the estimated gravity direction and feed it to the global localization algorithm to obtain an initial position estimate. We then initialize the proposed localization framework with the estimated initial position. Note that we designed the framework so that it can dynamically load a global localization module from a shared library. The framework is agnostic to the global localization algorithm and any 3- or 4-DoF global localization algorithm (e.g., [25]) can easily be incorporated into the proposed framework.

IV. Experiment

A. Indoor Experiment

Experimental setting: To demonstrate the robustness of the proposed method, we conducted localization experiments in the indoor environment shown in Fig. 1. Using a Microsoft Azure Kinect, we recorded two sequences (Easy01 and Easy02) of point cloud and IMU data without aggressive motion, and another one sequence (Hard) with quick sensor motion, point cloud degeneration, and traveling across both mapped and unmapped regions. The durations of the

[2]The implementation of the 2D global localization is available at: `https://github.com/koide3/hdl_global_localization`

TABLE I: Absolute trajectory errors for indoor sequences

Method	ATE [m] Easy01	Easy02	Hard
FAST_LIO (odom)	2.485 ± 1.216	7.101 ± 2.396	27.942 ± 23.022
FAST_LIO_LOC	0.068 ± 0.044	0.150 ± 0.118	27.265 ± 25.353
hdl_localization	0.210 ± 0.177	15.048 ± 10.506	25.507 ± 24.431
Proposed	**0.054 ± 0.008**	**0.041 ± 0.011**	**0.282 ± 0.253**

sequences Easy01, Easy02, and Hard were respectively 139, 136, and 164 s [3].

As a baseline, we ran two localization algorithms: FAST_LIO_LOCALIZATION (FAST_LIO_LOC) [4] and hdl_localization [29]. FAST_LIO_LOC uses FAST_LIO2, a tightly coupled LiDAR-IMU odometry estimation based on an iterated Kalman filter [5], to estimate the sensor ego-motion and periodically performs scan-to-map registration to correct estimation drift. For comparison, we also ran FAST_LIO2 [5] without map-based pose correction. hdl_localization [29] performs NDT-based scan-to-map registration and IMU fusion based on an unscented Kalman filter. For all the evaluated methods, we provided an initial pose manually.

To obtain reference sensor trajectories, we manually aligned scan point clouds with the map point cloud and performed batch optimization of the GICP scan-to-map registration errors and IMU motion errors. We evaluated the estimated trajectories with the absolute trajectory error (ATE) metric [30] using the *evo* toolkit [5].

Easy sequences: Table I summarizes the ATEs of the evaluated methods. We can see that FAST_LIO without map-based correction showed large trajectory estimation errors for the Easy01 and Easy02 sequences (2.485 and 7.101 m) because the estimation drift quickly accumulated due to the small feature-less environment and the narrow field of view of the sensor. FAST_LIO_LOC significantly improved the ATEs (0.068 and 0.150 m), and this result suggests the necessity of map-based correction to maintain the accuracy of sensor pose tracking. Although hdl_localization showed a decent ATE for the Easy01 sequence (0.210 m), its accuracy largely deteriorated for the Easy02 sequence (15.048 m) because of a scan matching failure caused by the small feature-less environment. The proposed method showed the best ATEs for the Easy01 and Easy02 sequences (0.054 and 0.041 m) thanks to its robustness to a feature-less environment.

Hard sequence: For the Hard sequence, both FAST_LIO_LOC and hdl_localization became corrupted. Fig. 3 shows the estimated trajectories. Because hdl_localization largely relied on scan-to-map registration to maintain sensor pose tracking, it immediately became corrupted when the sensor entered an unmapped region. FAST_LIO_LOC showed a slight estimation error in the unmapped region because of the decoupled scan-to-map registration that

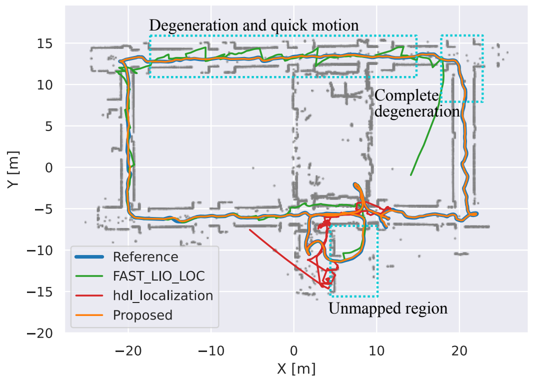


Fig. 3: Estimated trajectories for the Hard sequence. Existing methods suffered from unmapped regions and severe degeneration of point clouds. The proposed method successfully continued tracking the sensor pose under these severe conditions thanks to its tightly coupled registration factors and windowed state optimization.

became unreliable when the overlap with the map was small. We observed that FAST_LIO_LOC became unstable under point cloud degeneration because the underlying FAST_LIO2 suffered from pose ambiguity that made it difficult to accurately accumulate scan points into the model point cloud. As a consequence, it eventually became corrupted when a complete degeneration of point clouds occurred. The proposed method successfully continued tracking the sensor pose during the Hard sequence and showed the best ATE among the evaluated methods (0.282 m). Because of the tightly coupled scan-to-scan and scan-to-map registration factors, it showed smooth estimation over the unmapped region. Furthermore, its windowed optimization made it possible to deal with complete degeneration of point cloud data that resulted in a smooth trajectory estimation result.

B. Outdoor Experiment

Experimental setting: We conducted experiments in the outdoor environment shown in Fig. 4. We recorded two sequences of point cloud and IMU data using a Livox MID360. To evaluate the robustness of localization methods, we included the following challenging situations in each sequence:

A) The sensor was moved with quick translation and rotation (∼1.6 m/s and ∼1.1 rad/s).
B) The sensor view was occluded completely so that point cloud data were interrupted several times.
C) The sensor was strongly shaken in random directions (5.0 rad/s) over a long interval (5–10 s).
D) The sensor moved outside the map and traveled in unmapped regions for about 150–300 m.

Because the dataset constrains extremely difficult situations, when a localization method becomes corrupted, we restart that method and count the number of corruptions that occurred in each sequence.

For the proposed method, we did not provide an initial sensor pose but rather used a combination of the proposed

[3]The dataset is available at `https://staff.aist.go.jp/k.koide/projects/icra2024_gl/`
[4]`https://github.com/HViktorTsoi/FAST_LIO_LOCALIZATION`
[5]`https://github.com/MichaelGrupp/evo`

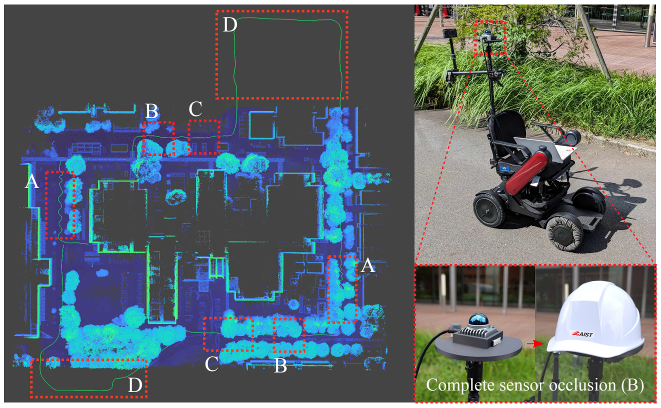
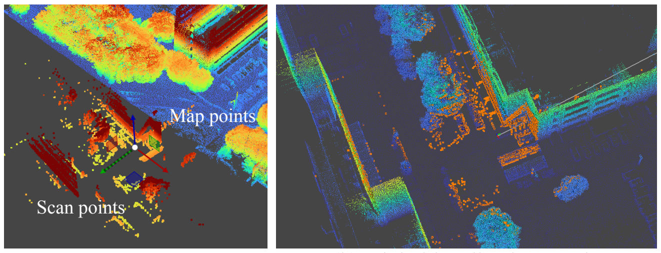


Fig. 4: Outdoor experimental environment (280 × 200 m^2). The evaluation sequences include four challenging situations: A) large translational and rotational movements, B) point cloud data interruptions, C) aggressive sensor rotations, and D) traveling over unmapped regions. In particular, in the B regions, the sensor was completely occluded several times, which created extreme challenges to localization methods.



(a) Gravity alignment result (b) Global localization result

Fig. 5: Automatic initialization result. (a) The gravity direction of the LiDAR scans was estimated, and (b) the 2D global localization successfully estimated the initial sensor position.

gravity direction estimation and the 2D global localization [28] for automatic system initialization.

Initialization: For both the Outdoor01 and Outdoor02 sequences, the initialization process properly estimated the gravity direction of scan points as shown in Fig. 5 (a) although the LiDAR was slightly tilted. Then, the 2D global localization was performed based on the gravity-aligned scan points, and it successfully estimated the sensor position, and the localization process was initiated. The gravity direction estimation and global localization took approximately 2.0 and 3.5 s, respectively. Consequently, the automatic initialization process took 5.5 s in total.

Estimation result: Table II summarizes the quantitative evaluation results, and Fig. 6 shows the estimated trajectories. Owing to IMU fusion, all the methods were able to continue tracking the sensor pose under quick translation

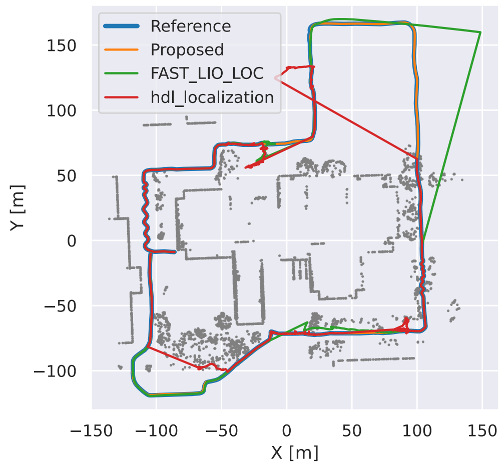


Fig. 6: Estimated trajectories for the Outdoor01 sequence.

and rotation in the region (A). Under data interruptions, both the existing methods became unstable, and hdl_localization immediately became corrupted in the region (B). Although FAST_LIO_LOC maintained pose tracking in the region (B), in the following region (C), it became corrupted due to very aggressive sensor rotation. Furthermore, both methods failed to maintain sensor pose tracking in unmapped regions (D). This result shows the weakness of decoupled scan-to-map registration, which can be unreliable in places where only a small overlap with the map is available.

The proposed method successfully continued tracking the sensor pose through the Outdoor01 and Outdoor02 sequences without estimation corruptions, whereas FAST_LIO_LOC and hdl_localization respectively became corrupted 4 and 5 times for each sequence. This result suggests the robustness of the proposed method owing to the sliding-window-based optimization and tightly coupled scan-to-map registration factors.

Processing time: For the proposed method, point cloud preprocessing and factor graph optimization respectively took 7.73 ± 1.89 and 9.48 ± 6.10 ms, and the total processing time per frame was 17.21 ms (58.1 FPS), which was much faster than the real-time requirement (10 FPS).

TABLE II: Absolute trajectory errors for outdoor sequences

Method	Outdoor01		Outdoor02	
	ATE [m]	Corruptions	ATE [m]	Corruptions
FAST_LIO_LOC	6.983 ± 6.693	4	4.849 ± 4.692	4
hdl_localization	5.592 ± 5.252	5	7.137 ± 6.833	5
Proposed	**0.959 ± 0.838**	**0**	**0.597 ± 0.401**	**0**

V. Conclusion

In this work, we proposed a map-based localization algorithm using the tight coupling of scan-to-scan and scan-to-map registration factors and IMU factors. Sensor states in a sliding window were continuously optimized. The proposed approach enabled dealing with severe point cloud degeneration and traveling across both mapped and unmapped regions. Through indoor and outdoor experiments, the proposed method was shown to successfully continue tracking the sensor pose in challenging situations with a bounded computation cost.

In future work, we plan to incorporate a more sophisticated 4-DoF global localization method [31] for reliable system initialization in complex 3D structured environments. We are also considering integrating tracking failure detection and re-localization mechanisms to deal with the kidnapping problem.

References

[1] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek, "The trimmed iterative closest point algorithm," in *Object recognition supported by user interaction for service robots*. IEEE, 2002, pp. 545–548.

[2] M. Magnusson, "The three-dimensional normal-distributions transform: an efficient representation for registration, surface analysis, and loop detection," Ph.D. dissertation, Örebro universitet, 2009.

[3] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[4] G. P. C. Junior, A. M. C. Rezende, V. R. F. Miranda, R. Fernandes, H. Azpurua, A. A. Neto, G. Pessin, and G. M. Freitas, "EKF-LOAM: An adaptive fusion of LiDAR SLAM with wheel odometry and inertial data for confined spaces with few geometric features," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1458–1471, July 2022.

[5] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.

[6] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Globally consistent and tightly coupled 3d LiDAR inertial mapping," in *IEEE International Conference on Robotics and Automation*. IEEE, May 2022.

[7] W.-C. Ma, R. Urtasun, I. Tartavull, I. A. Barsan, S. Wang, M. Bai, G. Mattyus, N. Homayounfar, S. K. Lakshmikanth, and A. Pokrovsky, "Exploiting sparse semantic HD maps for self-driving vehicle localization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Nov. 2019.

[8] F. Yan, O. Vysotska, and C. Stachniss, "Global localization on OpenStreetMap using 4-bit semantic descriptors," in *European Conference on Mobile Robots*. IEEE, Sept. 2019.

[9] Y. Cho, G. Kim, S. Lee, and J.-H. Ryu, "OpenStreetMap-based LiDAR global localization in urban environment without a prior LiDAR map," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4999–5006, Apr. 2022.

[10] D. Wisth, M. Camurri, and M. Fallon, "VILENS: Visual, inertial, lidar, and leg odometry for all-terrain legged robots," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 309–326, Feb. 2023.

[11] C. Bai, T. Xiao, Y. Chen, H. Wang, F. Zhang, and X. Gao, "Faster-LIO: Lightweight tightly coupled lidar-inertial odometry using parallel sparse incremental voxels," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4861–4868, Apr. 2022.

[12] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," in *IEEE International Conference on Robotics and Automation*. IEEE, May 2021.

[13] T.-M. Nguyen, M. Cao, S. Yuan, Y. Lyu, T. H. Nguyen, and L. Xie, "LIRO: Tightly coupled lidar-inertia-ranging odometry," in *IEEE International Conference on Robotics and Automation*. IEEE, May 2021.

[14] D. Fox, "Adapting the sample size in particle filters through KLD-sampling," *The International Journal of Robotics Research*, vol. 22, no. 12, pp. 985–1003, Dec. 2003.

[15] D. F. Sebastian Thrun, Wolfram Burgard, *Probabilistic Robotics*. The MIT Press, 2005.

[16] J. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, "Normal distributions transform monte-carlo localization (NDT-MCL)," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Nov. 2013.

[17] F. J. Perez-Grau, F. Caballero, A. Viguria, and A. Ollero, "Multi-sensor three-dimensional monte carlo localization for long-term aerial robot navigation," *International Journal of Advanced Robotic Systems*, vol. 14, no. 5, Sept. 2017.

[18] N. Akai, T. Hirayama, and H. Murase, "3d monte carlo localization with efficient distance field representation for automated driving in dynamic environments," in *IEEE Intelligent Vehicles Symposium*. IEEE, Oct. 2020.

[19] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "PoseRBPF: A rao–blackwellized particle filter for 6-d object pose tracking," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1328–1342, Oct. 2021.

[20] F. A. Maken, F. Ramos, and L. Ott, "Stein particle filter for nonlinear, non-gaussian state estimation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5421–5428, Apr. 2022.

[21] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." in *Robotics: science and systems*, vol. 2, no. 4. Seattle, WA, 2009, p. 435.

[22] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Robotics: Science and Systems*. Robotics: Science and Systems Foundation, July 2015.

[23] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, Dec. 2011.

[24] F. Dellaert and G. Contributors, "borglab/gtsam," May 2022. [Online]. Available: https://github.com/borglab/gtsam)

[25] H. Lim, S. Yeon, S. Ryu, Y. Lee, Y. Kim, J. Yun, E. Jung, D. Lee, and H. Myung, "A single correspondence is enough: Robust global registration to avoid degeneracy in urban environments," in *IEEE International Conference on Robotics and Automation*. IEEE, May 2022.

[26] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Oct. 2018, pp. 4802–4809.

[27] P. Dellenbach, J.-E. Deschaud, B. Jacquet, and F. Goulette, "CT-ICP: Real-time elastic LiDAR odometry with loop closure," in *IEEE International Conference on Robotics and Automation*. IEEE, May 2022, pp. 5580–5586.

[28] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d LIDAR SLAM," in *IEEE International Conference on Robotics and Automation*. IEEE, May 2016.

[29] K. Koide, J. Miura, and E. Menegatti, "A portable three-dimensional LIDAR-based system for long-term and wide-area people behavior measurement," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, Mar. 2019.

[30] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Oct. 2018, pp. 7244–7251.

[31] K. Aoki, K. Koide, S. Oishi, M. Yokozuka, A. Banno, and J. Meguro, "3d-bbs: Global localization for 3d point cloud scan matching using branch-and-bound algorithm," in *IEEE International Conference on Robotics and Automation*. IEEE, May 2024.