LIO-EKF: High Frequency LiDAR-Inertial Odometry using Extended Kalman Filters

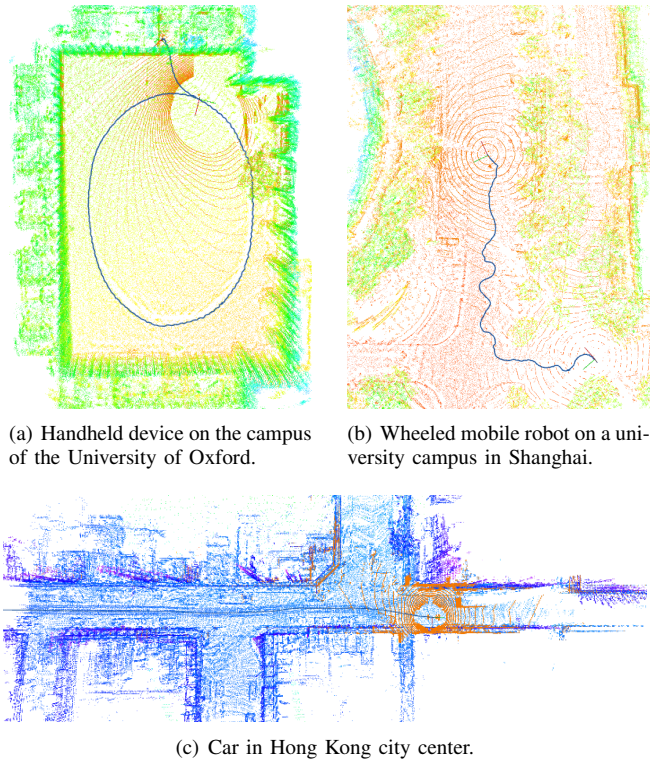
Yibin Wu Tiziano Guadagnino Louis Wiesmann Lasse Klingbeil Cyrill Stachniss Heiner Kuhlmann

***Abstract*— Odometry estimation is crucial for every autonomous system requiring navigation in an unknown environment. In modern mobile robots, 3D LiDAR-inertial systems are often used for this task. By fusing LiDAR scans and IMU measurements, these systems can reduce the accumulated drift caused by sequentially registering individual LiDAR scans and provide a robust pose estimate. Although effective, LiDAR-inertial odometry systems require proper parameter tuning to be deployed. In this paper, we propose LIO-EKF, a tightly-coupled LiDAR-inertial odometry system based on point-to-point registration and the classical extended Kalman filter scheme. We propose an adaptive data association that considers the relative pose uncertainty, the map discretization errors, and the LiDAR noise. In this way, we can substantially reduce the parameters to tune for a given type of environment. The experimental evaluation suggests that the proposed system performs on par with the state-of-the-art LiDAR-inertial odometry pipelines but is significantly faster in computing the odometry. The source code of our implementation is publicly available (https://github.com/YibinWu/LIO-EKF).**

I. Introduction

Ego-motion estimation is crucial for autonomous robot navigation. The knowledge of the robot's location enables higher-level tasks such as mapping, path planning, or obstacle avoidance. In the last decade, 3D LiDAR-inertial odometry (LIO) systems have attracted significant interest in the research community [1]–[7]. Combining LiDAR and IMU measurements provides a low-drift ego-motion estimation, which is robust to aggressive motion profiles, low-light conditions, and poorly structured environments. However, existing LiDAR-inertial systems require parameter tuning to provide an accurate pose estimate [2], [4], [5]. In particular, most of the tuning effort is focused on the point cloud registration module, where the feature extraction, number of iterations, and data association threshold must be properly set.

Recently, Vizzo et al. [8] proposed KISS-ICP, a LiDAR odometry system based on point-to-point iterative closest point (ICP) that can accurately estimate the ego-motion of the robot in a variety of scenarios basically without parameter tuning. Although effective, KISS-ICP has the same limitations as most other LiDAR odometry pipelines [9]–[12], for example, poor robustness to motion characterized by rapidly-changing acceleration. We tackle this issue by proposing an equivalent, easy-to-use system for LIO.



(a) Handheld device on the campus of the University of Oxford.

(b) Wheeled mobile robot on a university campus in Shanghai.

(c) Car in Hong Kong city center.

Fig. 1. Odometry estimation results of LIO-EKF with different sensing platforms in different environments.

The main contribution of this paper is a tightly-coupled LIO system based on point-to-point registration and the classical extended Kalman filter (EKF) scheme. Our system builds on top of the insight provided by KISS-ICP and provides a robust and effective ego-motion estimation by fusing LiDAR scans and IMU measurements. In particular, we show that, given the initial pose prediction from the IMU, an accurate pose can be computed by relying on a standard EKF scheme without the need for multiple iterations as in existing systems. Moreover, to reduce the number of parameters that need to be tuned, we design a novel adaptive threshold for data association that considers the uncertainty of the motion prediction coming from the IMU, the map discretization error, and the noise of the range measurements from the LiDAR. In this way, in our system, no tuning is required for feature extraction, data association, and the number of iterations in the correction step. Fig. 1 illustrates the localization and mapping results of our approach running on different challenging datasets.

All authors are with the Institute of Geodesy and Geoinformation, University of Bonn, Bonn, Germany. Cyrill Stachniss is additionally with the Department of Engineering Science at the University of Oxford, UK, and with the Lamarr Institute for Machine Learning and Artificial Intelligence, Germany. {firstname.lastname}@igg.uni-bonn.de.

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy-EXC 2070-390732324.

In sum, we make three key claims: LIO-EKF (i) is on par with the state-of-the-art LIO systems in terms of estimation accuracy, (ii) can provide an accurate pose estimate in different environments and vehicle motion profiles, (iii) can provide pose estimates at close-to-IMU frame rate. These claims are backed up by our experimental evaluation.

II. Related Works

A typical LIO system can be divided into two main components: front end and back end [13]. In the front end, the system performs data association to find corresponding points between the current LiDAR scan and the previous scan (or a map) [14]–[16]. In contrast, the back end utilizes various state estimation methods to fuse information from both LiDAR and IMU sensors to estimate the robot's pose.

In the front end of LIO systems, feature-based registration is the predominant paradigm. Feature-based registration, exemplified by LOAM [10] and its variants [9], [11], [17], involves extracting edges and planar patches from LiDAR scans and then finding corresponding features in the map to estimate the robot pose. LIO-SAM [5] builds on the front end of LOAM and optimizes the robot state estimates from LiDAR and IMU using a sliding window based on a factor graph. LIO-Mapping [6] uses a similar front end but includes a rotation-constrained refinement method to further enhance pose estimation and point-cloud maps. LINS [4] also relies on edges and planar features for data association. LiLi-OM [2] and FAST-LIO [1] tailor a similar feature extraction module for solid-state LiDARs. Although FAST-LIO2 [18] proposes to register the raw points without feature extraction, it relies on the point-to-plane metric, which also needs local plane approximation and normal vector computation. All of the above approaches require parameter tuning for feature extraction, depending on the specific LiDAR sensor in use and the structure of the environment. Conversely, our proposed method removes such a limitation and uses the classical point-to-point metric in the registration, following the insights of KISS-ICP [8].

In terms of the back end, most LiDAR-inertial odometry systems rely on either the iterated extended Kalman filter (IEKF) [1], [18] or factor graph optimization [19]–[21]. Although earlier methods fuse LiDAR and IMU measurements using factor graphs [2], [5], [6], recently the IEKF approaches [1], [4], [18] are gaining prominence. In an IEKF, the correction step of the filter is performed multiple times to reduce the linearization errors at the cost of increased computation. In our study, we suggest that, in most scenarios, the classical error-state Extended Kalman Filter (EKF) can be employed to fuse LiDAR scans and IMU readings effectively without additional iterations. This is because that we use a proper strapdown INS model [22], which can provide an accurate initial pose estimate between two successive scans.

Overall, our proposed LIO system, LIO-EKF, employs a point-to-point registration in the front end and uses classical EKF without iterations. Although composed of very straightforward components, our approach achieves comparable performance to state-of-the-art methods in pose estimation accuracy while being significantly faster.

III. Methodology

We aim to incrementally estimate the pose of a mobile robot by fusing sensor measurements obtained from a LiDAR scanner and an IMU in a tightly-coupled manner via EKF. We employ a conventional strapdown inertial navigation system (INS) for state prediction using the IMU readings. The predicted state is then corrected using the LiDAR scans in an observation model based on point-to-point residuals. To find reliable correspondences in the observation model, we propose an adaptive threshold that takes into account the predicted state uncertainty, map discretization errors, and LiDAR range noise.

In the following sections, we will first explain the error state model that our system uses. Then, we present the point-to-point based measurement model with associated preprocessing on the incoming LiDAR scan. Lastly, we introduce our novel adaptive thresholding module used for data association in the correction step of the EKF. In Fig. 2, we show an overview of our approach.

A. Error State Model

The robot state can be represented as

$$\boldsymbol{x} = \{\boldsymbol{t}\ ,\ \boldsymbol{v}\ ,\ \mathbf{R}\ ,\ \boldsymbol{b}_g\ ,\ \boldsymbol{b}_a\} \in \mathbb{R}^6 \times \mathrm{SO}(3) \times \mathbb{R}^6, \quad (1)$$

where $\boldsymbol{t}$, $\boldsymbol{v}$, and $\mathbf{R}$ are the position, velocity, and orientation of the robot in the global frame, and $\boldsymbol{b}_g$, $\boldsymbol{b}_a$ are the biases of the gyroscope and accelerometer respectively. As new gyroscope and accelerometer measurements are received, we can update the robot's state. We adopt an accurate INS mechanization model [22], [23] from the inertial navigation community, as it proves to be much more effective than a vanilla propagation scheme [5] especially when the vehicle undergoes significant motion.

Given an accelerometer measurement $\mathbf{a}_k$ and a gyroscope reading $\boldsymbol{\omega}_k$ in robot body frame at time k, we can compute the new state $\tilde{\boldsymbol{x}}_k$ through motion prediction. We indicate with $\tilde{\mathbf{y}}$ a variable $\mathbf{y}$ that has been updated with the IMU measurements, but not with the LiDAR scan. We can write the new state after INS prediction as

$$\begin{aligned}
\tilde{\boldsymbol{t}}_k &= \boldsymbol{t}_{k-1} + \frac{1}{2}(\boldsymbol{v}_{k-1} + \boldsymbol{v}_k)\, s_k, \\
\tilde{\boldsymbol{v}}_k &= \boldsymbol{v}_{k-1} + \boldsymbol{g}\, s_k + \mathbf{R}_{k-1}\left[\mathbf{a}_k\, s_k + \frac{1}{2}(\boldsymbol{\omega}_k \times \mathbf{a}_k)\, s_k^2\right. \\
&\quad \left. + \frac{1}{12}(\boldsymbol{\omega}_{k-1} \times \mathbf{a}_k - \boldsymbol{\omega}_k \times \mathbf{a}_{k-1})\, s_k^2\right], \\
\tilde{\mathbf{R}}_k &= \mathbf{R}_{k-1}\mathrm{Exp}(\boldsymbol{\omega}_k\, s_k + \frac{1}{12}(\boldsymbol{\omega}_{k-1} \times \boldsymbol{\omega}_k)\, s_k^2), \\
\tilde{\boldsymbol{b}}_{g,k} &= \boldsymbol{b}_{g,k-1}, \\
\tilde{\boldsymbol{b}}_{a,k} &= \boldsymbol{b}_{a,k-1},
\end{aligned} \quad (2)$$

where Exp is the exponential mapping to the $\mathrm{SO}(3)$ group, $\boldsymbol{g}$ is the gravity vector in the global frame, and s_k is the integration interval.

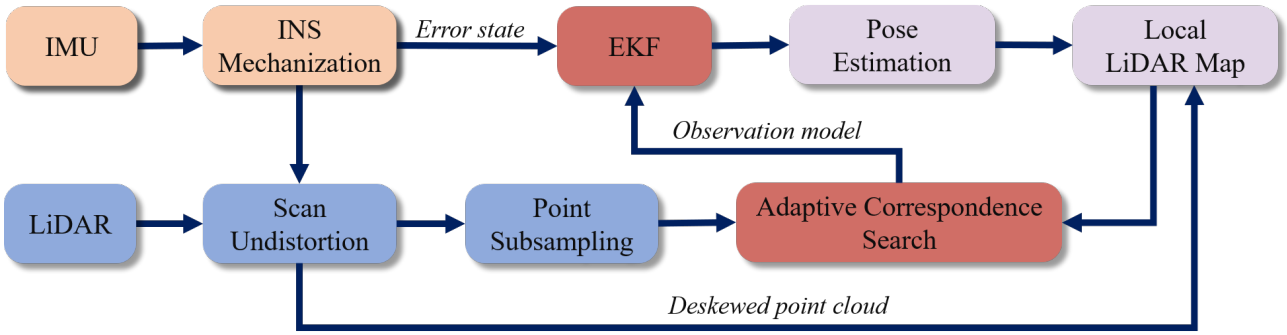


Fig. 2. Overview of LIO-EKF. We integrate the most basic front end (point-to-point association) and back end (EKF) to build a simple, small yet accurate, generic, and high frequency LIO system.

To better handle the non-linearities of the state dynamics in Eq. (2), we adopt an error-state formulation of the extended Kalman filter [24]. In this way, instead of directly estimating position, velocity, attitude, and IMU biases, we model the robot state $\boldsymbol{x}$ in terms of the error state vector

$$\Delta\boldsymbol{x} = \left[\Delta\boldsymbol{t}^\top \;,\; \Delta\boldsymbol{v}^\top \;,\; \Delta\boldsymbol{\phi}^\top \;,\; \Delta\boldsymbol{b}_g^\top \;,\; \Delta\boldsymbol{b}_a^\top\right]^\top \in \mathbb{R}^{15}, \quad (3)$$

where $\Delta\boldsymbol{t}$, $\Delta\boldsymbol{v}$ and $\Delta\boldsymbol{\phi}$ indicate the position, velocity, and attitude errors respectively, and $\Delta\boldsymbol{b}_g$ and $\Delta\boldsymbol{b}_a$ denote the bias errors of the accelerometer and gyroscope. There are several models in the literature describing the time-dependent behavior of the error state in an inertial system [22]. As we focus on robot ego-motion estimation with low-cost inertial sensors, we do not consider the earth rotation and the variation of the navigation frame. Therefore, a simplified Phi-angle error-state model [25] can be used. In particular, we use the discretized model

$$\mathbf{f}(\Delta\boldsymbol{x}_{k-1}, \mathbf{a}_k, \boldsymbol{\omega}_k) = \begin{bmatrix} \Delta\boldsymbol{t}_{k-1} + \Delta\boldsymbol{v}_{k-1} s_k \\ \Delta\boldsymbol{v}_{k-1} + (\mathbf{d}_k + \mathbf{R}_{k-1}\Delta\boldsymbol{b}_{a,k-1})\, s_k \\ \Delta\boldsymbol{\phi}_{k-1} - \mathbf{R}_{k-1}\,\Delta\boldsymbol{b}_{g,k-1} s_k \\ \Delta\boldsymbol{b}_{g,k-1} \\ \Delta\boldsymbol{b}_{a,k-1} \end{bmatrix},$$

$$\text{where} \quad \mathbf{d}_k = \mathbf{R}_{k-1}\mathbf{a}_k \times \Delta\boldsymbol{\phi}_{k-1}. \quad (4)$$

We can then write the prediction step of the filter as

$$\begin{aligned} \Delta\tilde{\boldsymbol{x}}_k &= \mathbf{f}(\Delta\boldsymbol{x}_{k-1}, \mathbf{a}_k, \boldsymbol{\omega}_k), \\ \tilde{\boldsymbol{\Sigma}}_k &= \mathbf{A}_k\,\boldsymbol{\Sigma}_{k-1}\,\mathbf{A}_k^\top + \mathbf{B}_k\,\boldsymbol{\Sigma}_{u,k}\,\mathbf{B}_k^\top + \boldsymbol{\Sigma}_{b,k}, \end{aligned} \quad (5)$$

where $\tilde{\boldsymbol{\Sigma}}_k \in \mathbb{R}^{15\times15}$ is the error state covariance, $\boldsymbol{\Sigma}_{u,k} \in \mathbb{R}^{6\times6}$ is the measurement noise of accelerometer and gyroscope, $\boldsymbol{\Sigma}_{b,k} \in \mathbb{R}^{15\times15}$ is the process uncertainty including IMU biases noise, and $\mathbf{A}_k \in \mathbb{R}^{15\times15}$ and $\mathbf{B}_k \in \mathbb{R}^{15\times6}$ are Jacobians defined as

$$\mathbf{A}_k = \frac{\partial\mathbf{f}(\Delta\boldsymbol{x}_{k-1}, \mathbf{a}_k, \boldsymbol{\omega}_k)}{\partial\Delta\boldsymbol{x}_{k-1}}, \quad \mathbf{B}_k = \frac{\partial\mathbf{f}(\Delta\boldsymbol{x}_{k-1}, \mathbf{a}_k, \boldsymbol{\omega}_k)}{\partial(\mathbf{a}_k, \boldsymbol{\omega}_k)}. \quad (6)$$

B. LiDAR Observation Model

Every time we process an incoming scan, we first deskew the point cloud of the scan using the IMU predicted pose. After scan deskewing, we employ the sub-sampling strategy proposed in KISS-ICP [8] to reduce the number of 3D points in the LiDAR point cloud and construct a local map using a voxel grid. We then transform the downsampled LiDAR frame to the robot body frame via the extrinsic calibration matrix between LiDAR and IMU. Please refer to Vizzo et al. [8] for the details of map maintenance and update. The correspondences between the current downsampled LiDAR frame and the local map are computed via the nearest neighbor search using voxel hashing, resulting in a correspondence set:

$$\mathcal{C} = \{\,(i,j) \;:\; \min_{i,j} \|\boldsymbol{p}_i - \boldsymbol{q}_j\|_2 < \tau\}, \quad (7)$$

where $\boldsymbol{p}_i$ is the i-th point in the downsampled LiDAR cloud, $\boldsymbol{q}_j$ is the j-th point in the local map, and τ is the maximum correspondence distance allowed for an inlier association.

The observation model for the i-th measured point $\boldsymbol{p}_i$ is:

$$\mathbf{h}_i(\Delta\tilde{\boldsymbol{x}}_k) = \mathrm{Exp}(\Delta\tilde{\boldsymbol{\phi}}_k)\,\tilde{\mathbf{R}}_k\,\boldsymbol{p}_i + \tilde{\boldsymbol{t}}_k + \Delta\tilde{\boldsymbol{t}}_k, \quad (8)$$

where $\tilde{\mathbf{R}}_k, \tilde{\boldsymbol{t}}_k$ represent the robot pose in the global frame after integrating the IMU readings between the previous and current LiDAR scan using Eq. (2). The corresponding innovation $\boldsymbol{e}_{ij}(\Delta\tilde{\boldsymbol{x}}_k)$ for $\boldsymbol{p}_i$ in the scan and corresponding point $\boldsymbol{q}_j$ in the map is:

$$\boldsymbol{e}_{ij}(\Delta\tilde{\boldsymbol{x}}_k) = \mathbf{h}_i(\Delta\tilde{\boldsymbol{x}}_k) - \boldsymbol{q}_j. \quad (9)$$

The Jacobian $\mathbf{J}_{ij}(\Delta\tilde{\boldsymbol{x}}_k)$ of $\boldsymbol{e}_{ij}(\Delta\tilde{\boldsymbol{x}}_k)$ is:

$$\mathbf{J}_{ij}(\Delta\tilde{\boldsymbol{x}}_k) = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \left[\tilde{\mathbf{R}}_k\,\boldsymbol{p}_i\right]_\times & \mathbf{0} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{3\times15}. \quad (10)$$

We can then update the error state vector and corresponding covariance using the alternative Kalman gain formulation proposed by Xu et al. [1]. In practice, to avoid explicitly forming the Kalman gain matrix, we rearrange the terms of the Kalman update equation as

$$\Delta\boldsymbol{x}_k = \Delta\tilde{\boldsymbol{x}}_k + \left(\mathbf{H}(\Delta\tilde{\boldsymbol{x}}_k) + \tilde{\boldsymbol{\Sigma}}_k^{-1}\right)^{-1} \mathbf{b}(\Delta\tilde{\boldsymbol{x}}_k), \quad (11)$$

$$\boldsymbol{\Sigma}_k = \left(\mathbf{I} - \left(\mathbf{H}(\Delta\tilde{\boldsymbol{x}}_k) + \tilde{\boldsymbol{\Sigma}}_k^{-1}\right)^{-1} \mathbf{H}(\Delta\tilde{\boldsymbol{x}}_k)\right) \tilde{\boldsymbol{\Sigma}}_k, \quad (12)$$

where

$$\mathbf{H}(\Delta\tilde{\boldsymbol{x}}_k) = \sum_{i,j\in\mathcal{C}} \mathbf{J}_{ij}^{\top}(\Delta\tilde{\boldsymbol{x}}_k)\, \boldsymbol{\Sigma}_p^{-1}\, \mathbf{J}_{ij}(\Delta\tilde{\boldsymbol{x}}_k), \tag{13}$$

$$\mathbf{b}(\Delta\tilde{\boldsymbol{x}}_k) = \sum_{i,j\in\mathcal{C}} \mathbf{J}_{ij}^{\top}(\Delta\tilde{\boldsymbol{x}}_k)\, \boldsymbol{\Sigma}_p^{-1}\, \boldsymbol{e}_{ij}(\Delta\tilde{\boldsymbol{x}}_k), \tag{14}$$

and $\boldsymbol{\Sigma}_p$ is the observation noise covariance. In this way, we avoid storing and multiplying the complete Jacobian $\mathbf{J}(\Delta\tilde{\boldsymbol{x}}_k) \in \mathbb{R}^{3n\times 15}$ of the n scan points.

Once the updated error state vector has been computed via Eq. (4), the position, velocity, attitude, and biases at the current timestamp k are computed via:

$$\begin{aligned}
\boldsymbol{t}_k &= \tilde{\boldsymbol{t}}_k + \Delta\boldsymbol{t}_k,\\
\boldsymbol{v}_k &= \tilde{\boldsymbol{v}}_k + \Delta\boldsymbol{v}_k,\\
\mathbf{R}_k &= \mathrm{Exp}(\Delta\boldsymbol{\phi}_k)\,\tilde{\mathbf{R}}_k,\\
\boldsymbol{b}_{a,k} &= \tilde{\boldsymbol{b}}_{a,k} + \Delta\boldsymbol{b}_{a,k},\\
\boldsymbol{b}_{g,k} &= \tilde{\boldsymbol{b}}_{g,k} + \Delta\boldsymbol{b}_{g,k}.
\end{aligned} \tag{15}$$

After that, the error state is set to zero and subsequently estimated when a new LiDAR scan is integrated.

C. Adaptive Data Association Threshold

In Eq. (7), we seek to establish associations between LiDAR points and local map points through the nearest neighbor search. To mitigate the effect of outlier correspondences, we introduce a threshold denoted as τ; it limits the maximum allowed distance between corresponding points. Instead of determining this threshold empirically, our approach aims to automatically estimate τ by considering the uncertainty of the initial pose prediction, map discretization errors, and sensor noise.

1) Pose Prediction: The most critical factor in determining good correspondences is the quality of the pose prediction, namely the distance between the initial estimate and the real pose of the robot. Since, in our case, we compute the pose prediction using IMU measurements, we can compute the uncertainty of the relative pose between two successive scans by integrating the noise from the accelerometer and gyroscope over the time interval between the two LiDAR scans. We follow the methodology of Forster et al. [26] but employ the INS mechanization detailed in Eq. (2). Consequently, we obtain a pose covariance matrix $\boldsymbol{\Sigma}_{r,k}$ that characterizes the uncertainty associated with the relative motion at time k. Our next step involves projecting this uncertainty into the space of point-to-point distances generated by the relative motion of the robot. These point-to-point distances can be approximated with an upper bound using the formula proposed by KISS-ICP [8]:

$$d(\mathbf{R},\boldsymbol{t}) = 2\; r_{\text{max}} \sin\left(\frac{1}{2}\boldsymbol{\Theta}(\mathbf{R})\right) + |\boldsymbol{t}|_2. \tag{16}$$

Here, $\boldsymbol{\Theta}(\mathbf{R})$ represents the angle corresponding to the rotation matrix $\mathbf{R}$, and r_{max} signifies the maximum range of the LiDAR. We can subsequently apply the unscented transform [27] in conjunction with Eq. (16) to obtain the variance σ^2_{p2p} of the point-to-point distances from $\boldsymbol{\Sigma}_{r,k}$.

2) Map Discretization Errors: Our pipeline employs a voxel grid as the map representation, with a predetermined maximum number of points m per voxel. This choice speeds up the nearest neighbor search but introduces a discretization error in the voxel grid. We account for this error by modeling a Gaussian over the distances between a scan point and m points uniformly distributed on a surface patch contained in a voxel of size v. The variance of this Gaussian can be then computed as:

$$\sigma^2_{\text{map}} = \frac{v^2}{m}. \tag{17}$$

3) Sensor Noise: We incorporate sensor noise that affects the range measurements from the LiDAR using the variance σ^2_{range}.

The threshold τ at time k can then be computed by assuming the three Gaussians to be independent and employing the three-sigma bound:

$$\tau = 3\,\sqrt{\sigma^2_{\text{p2p}} + \sigma^2_{\text{map}} + \sigma^2_{\text{range}}}. \tag{18}$$

IV. Experimental Results

This work presents LIO-EKF, a LIO system based on EKF, and a point-to-point metric that can provide an accurate pose at a high frequency in different environments without parameter tuning. This section presents real-world experimental results to support our key claims, namely, LIO-EKF (i) is on par with the state-of-the-art LIO systems in terms of estimation accuracy, (ii) can provide an accurate pose estimate in different environments and vehicle motion profiles, (iii) can provide pose estimates at close-to-IMU frame rate. After that, we provide ablation studies to analyze the key characteristics of LIO-EKF.

A. Experimental Setup

We compare LIO-EKF with two state-of-the-art LIO systems, FAST-LIO2 [18] and LIO-SAM [5] in three different datasets. We used the default parameters in the open-source code of FAST-LIO2 and LIO-SAM, respectively. We disabled the loop closure module in LIO-SAM for the sake of fairness. The three datasets that we selected are:

- *urbanNav* [28] an autonomous driving dataset composed of three sequences recorded in Hong Kong.
- *m2dgr* [29] a dataset composed of 7 sequences recorded with a wheeled mobile robot on a university campus in Shanghai.
- *newerCollege* [30] a dataset composed of two sequences recorded with a handheld device on the Oxford university campus.

The IMUs used in the datasets are consumer-grade microelectromechanical system (MEMS) IMUs. Please refer to the dateset websites for more details. To evaluate the different methods we use the Absolute Trajectory Error (ATE) [8] and the KITTI metric [31] for the relative error.

TABLE I

QUANTITATIVE RESULTS ON THE THREE DIFFERENT DATASETS

Sequence	Method	Avg. tra.	Avg. rot.	ATE. tra.	ATE. rot.
urbanNav 20210517	FAST-LIO2	4.11	1.68	**17.62**	4.45
	LIO-SAM	**3.18**	**1.40**	20.74	**4.20**
	LIO-EKF	3.20	1.45	24.73	5.05
urbanNav 20210518	FAST-LIO2	2.73	1.30	23.02	3.27
	LIO-SAM	2.52	1.31	**20.37**	**2.98**
	LIO-EKF	**2.20**	**1.14**	22.44	7.46
urbanNav 20210521	FAST-LIO2	3.56	1.63	47.29	5.18
	LIO-SAM	**2.94**	1.62	**30.98**	4.51
	LIO-EKF	2.96	**1.54**	34.97	**4.47**
m2dgr street_01-05	FAST-LIO2	**1.62**	**0.79**	**5.05**	1.79
	LIO-SAM	3.15	1.52	10.19	4.27
	LIO-EKF	1.68	0.83	5.33	**1.70**
m2dgr street_06	FAST-LIO2	3.41	**1.55**	**8.93**	2.29
	LIO-SAM	3.65	1.65	9.04	2.41
	LIO-EKF	**3.37**	1.56	9.05	**2.27**
m2dgr street_08	FAST-LIO2	**1.10**	**1.65**	**2.12**	**1.71**
	LIO-SAM	3.73	5.78	4.21	6.36
	LIO-EKF	1.28	1.85	2.22	1.88
newerCollege short_exp	FAST-LIO2	1.05	1.01	5.14	2.49
	LIO-EKF	**0.63**	**0.73**	**4.16**	**1.75**
newerCollege long_exp	FAST-LIO2	1.09	1.33	6.33	4.22
	LIO-EKF	**0.74**	**0.91**	**5.14**	**2.34**

Avg. tra. and Avg. rot. denote the relative translation error (%) and relative rotation error (deg/m) using KITTI [31] metrics, respectively.
ATE. tra. and ATE. rot. denote the absolute rotation error (deg) and absolute translation error (m) [8], respectively.
In bold the best performance.

B. System Parameters

For all the following experiments, we use the same system configuration. The list of tunable parameters of our pipeline are:

- Max. points per voxel m
- Voxel size v
- Initial state covariance $\mathbf{\Sigma}_0$
- Point observation covariance $\mathbf{\Sigma}_p$

Notice that we do not consider as parameters the noise covariance of the IMU measurements $\mathbf{\Sigma}_{u,k}$, the process covariance $\mathbf{\Sigma}_{b,k}$ related to the biases noise, and the range noise σ^2_{range} as those can be determined from the sensors datasheets.

C. Odometry Performance

In the first experiment, we analyze the odometry accuracy of our method and its generalizability to different environments and motion profiles. In Table I, we present the comparative results of LIO-EKF with the two baselines in all datasets. We average the results for the sequences "street_01" to "street_05" in m2dgr due to space limitation. Unfortunately, we were not able to run LIO-SAM on the newerCollege dataset because additional attitude estimation output from the IMU is needed to initialize the system.

The quantitative results show that the proposed approach performs on par with the state-of-the-art LIO systems. Moreover, it shows better generalization capabilities, as we can see by comparing the performances of LIO-EKF and FAST-LIO2 on the *urbanNav* and *newerCollege* sequences. This illustrates that, with a single configuration, LIO-EKF can handle different motion profiles of the platform and different structures of the environment. Moreover, these results show that a classical EKF scheme is sufficient to fuse IMU and LiDAR data to estimate an accurate robot pose without relying on more complex state estimation schemes such as factor graphs or IEKF. This evaluation supports the first two claims made in this paper.

TABLE II

COMPARISON OF THE AVERAGE PROCESSING TIME [MS]

	urbanNav	m2dgr	newerCollege
LIO-SAM	41.68	60.01	-
FAST-LIO2	24.03	29.96	9.42
LIO-EKF	**12.37**	**13.65**	**7.36**

Average processing time for the scan processing in all the selected datasets. The numbers are reported in milliseconds [ms].

TABLE III

COMPARISON OF LIO-EKF WITH DIFFERENT NUMBER OF ITERATIONS

Sequence	# iterations	Avg. tra.	Avg. rot.	Processing Time (ms)
m2dgr street_05	1	2.64	0.80	11.37
	10	2.61	0.77	35.10
	100	2.61	0.76	63.90
newerCollege short_exp	1	0.63	0.73	12.41
	10	0.60	0.62	26.92
	100	0.55	0.58	84.21

Avg. tra. and Avg. rot. denote the relative translation error (%) and relative rotation error (deg/m) using KITTI metrics, respectively.

D. Computation Efficiency

In this section, we analyze the computational speed of our method. The results support our third claim that LIO-EKF can compute the robot's ego motion at close-to-IMU frequency. We compute the average processing time of the correction step of the filter once a new LiDAR scan has been received. We chose this because the correction step is the most time-consuming part of the pipeline, including scan preprocessing, data association, and pose update. Furthermore, we also compare the time of both FAST-LIO2 and LIO-SAM in terms of the scan processing in all the selected datasets by averaging the values over the different sequences. All the methods have been run on a desktop machine with an Intel i7-10700 CPU at 2.90 GHz and 32 GB of RAM. The results are shown in Table II. As we can see from the results, LIO-EKF achieves the fastest processing time for the scan processing, which is close to a regular IMU stream rate (100 Hz). In particular, it is about two times faster than FAST-LIO2 and four times faster than LIO-SAM in most scenarios. This is because our method uses a classical EKF scheme, which is very efficient as the optimization does not require multiple iterations.

E. Ablation Studies

In this section, we perform two ablation studies to give better insights into the design choices of our pipeline. In particular, we show the impact of using the EKF scheme compared to an IEKF with different numbers of iterations. Furthermore, we showcase that our novel adaptive data association threshold significantly impacts the generalization capabilities of our proposed LIO pipeline.

1) Impact of multiple iterations of the odometry accuracy: We now investigate if multiple iterations can improve the odometry estimate of LIO-EKF. We replaced the used EKF scheme in our system with an IEKF, where we selected 10 and 100 max iterations for comparison. We use the "street_05" sequence of *m2dgr* and the "short_exp" sequence of *newerCollege* for this ablation study. We compare the relative error according to the KITTI metric and the average processing time of the scan. The results are shown in Table III.

As we can see from Table III, there is no significant improvement in the odometry accuracy with more iterations, with a maximum gap of 0.08% in relative translation error at 100 iterations in "short_exp". Conversely, the processing time is dramatically increased by a factor of 8. The reasons are two-fold. First, IMU has excellent short-term stability, and it can provide an accurate pose prediction for every LiDAR scan so that the estimation can converge to an accurate pose after one update. Second, our error state formulation better handles the nonlinearity of the system dynamics. This shows that our design decision to use a classical EKF effectively computes the robot pose without relying on more complex state estimation schemes.

2) Adaptive Threshold: In the second ablation study, we show the generalization capabilities and effectiveness of the proposed adaptive threshold model. We compare fixed threshold settings and the adaptive threshold scheme proposed in KISS-ICP [8]. For this comparative analysis, we use two *urbanNav* sequences and two *m2dgr* sequences. In this way, we aim to showcase that the proposed adaptive threshold can be generalized to different motion profiles. Table IV lists the results.

We can see in Table IV that the proposed adaptive threshold achieves the best generalization capability overall. To be specific, for *urbanNav*, $1\,\mathrm{m}$ seems to be as good as the proposed. However, it is worse in the *m2dgr*. That means a hand-crafted threshold cannot be guaranteed to work well with different vehicle motions in different environments. In addition, KISS-ICP uses historical registrations to estimate the average deviation between the constant velocity motion prediction and the corrected pose to indicate the threshold statistically. However, it is just an approximation of the motion prediction error, which cannot represent the exact uncertainty of the initial pose guess. In the proposed algorithm, we directly use the uncertainty of the IMU-predicted initial pose, which is more accurate than the statistical model proposed in KISS-ICP. Furthermore, we consider the range noise of the LiDAR sensor and the map discretization error, which better models the data association problem.

TABLE IV
COMPARISON OF LIO-EKF WITH DIFFERENT THRESHOLD

Sequence	**Threshold** (m)	**Avg. tra.**	**Avg. rot.**
urbanNav 20210518	0.3	2.23	1.25
	1	**2.18**	**1.12**
	KISS-ICP	2.24	1.14
	Ours	2.20	1.14
urbanNav 20210521	0.3	2.96	1.72
	1	2.95	1.55
	KISS-ICP	**2.95**	1.59
	Ours	2.96	**1.54**
m2dgr street_01	0.3	4.25	2.26
	1	1.46	0.86
	KISS-ICP	1.79	1.04
	Ours	**1.44**	**0.85**
m2dgr street_08	0.3	2.66	2.58
	1	1.33	1.93
	KISS-ICP	2.06	2.50
	Ours	**1.28**	**1.85**

Avg. tra. and Avg. rot. denote the relative translation error (%) and relative rotation error (deg/m) using KITTI metrics, respectively. In bold the best performing system configuration.

V. CONCLUSIONS

This paper presents LIO-EKF, a LiDAR-inertial odometry system based on a classical EKF scheme. Our approach exploits the classical point-to-point metric with an EKF to build a generic, tightly-coupled LIO system. Thanks to our error state formulation, we can better handle the non-linearities of the problem and converge to an accurate robot pose with a single correction step without relying on multiple iterations, as in IEKF schemes. Additionally, we propose a novel adaptive threshold model considering the pose uncertainty, map discretization error, and sensor noise for a more robust and effective data association.

Extensive real-world experiments conducted with different platforms in different environments show that LIO-EKF can achieve on par odometry performance compared to the more sophisticated state-of-the-art systems with a significantly more straightforward system design. Additionally, LIO-EKF can compute the pose at close to the IMU frame rate (100 Hz), which is much faster than other state-of-the-art methods.

REFERENCES

[1] W. Xu and F. Zhang, "FAST-LIO: A fast, robust lidar-inertial odometry package by tightly-coupled iterated Kalman filter," *IEEE Robotics and Automation Letters.*, vol. 6, DOI 10.1109/LRA.2021.3064227, no. 2, pp. 3317–3324, 2021.

[2] K. Li, M. Li, and U. D. Hanebeck, "Towards high-performance solid-state-lidar-inertial odometry and mapping," *IEEE Robotics and Automation Letters.*, vol. 6, DOI 10.1109/LRA.2021.3070251, no. 3, pp. 5167–5174, 2021.

[3] 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, DOI 10.1109/LRA.2022.3152830, no. 2, pp. 4861–4868, 2022.

[4] C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang, and M. Liu, "LINS: A lidar-inertial state estimator for robust and efficient navigation," in *Proc. of the IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 8899–8906, 2020.

[5] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *Proc. of the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, pp. 5135–5142, 2020.

[6] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3D lidar inertial odometry and mapping," in *Proc. of the IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 3144–3150, 2019.

[7] F. Huang, W. Wen, J. Zhang, and L.-T. Hsu, "Point wise or feature wise? a benchmark comparison of publicly available lidar odometry algorithms in urban canyons," *IEEE Intelligent Transportation Systems Magazine*, vol. 14, DOI 10.1109/MITS.2021.3092731, no. 6, pp. 155–173, 2022.

[8] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss, "KISS-ICP: In defense of point-to-point ICP – simple, accurate, and robust registration if done the right way," *IEEE Robotics and Automation Letters.*, vol. 8, DOI 10.1109/LRA.2023.3236571, no. 2, pp. 1029–1036, 2023.

[9] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-LOAM: Fast lidar odometry and mapping," in *Proc. of the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, DOI 10.1109/IROS51168.2021.9636655, p. 4390–4396, 2021.

[10] J. Zhang and S. Singh, "Low-drift and real-time lidar odometry and mapping," *Autonomous Robots*, vol. 41, pp. 401–416, 2017.

[11] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *Proc. of the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, pp. 4758–4765, 2018.

[12] P. Dellenbach, J.-E. Deschaud, B. Jacquet, and F. Goulette, "CT-ICP: Real-time elastic lidar odometry with loop closure," in *Proc. of the IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 5580–5586, 2022.

[13] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[14] T. Guadagnino, X. Chen, M. Sodano, J. Behley, G. Grisetti, and C. Stachniss, "Fast Sparse LiDAR Odometry Using Self-Supervised Feature Selection on Intensity Images," *IEEE Robotics and Automation Letters.*, vol. 7, DOI 10.1109/LRA.2022.3184454, no. 3, pp. 7597–7604, 2022.

[15] L. Di Giammarino, L. Brizi, T. Guadagnino, C. Stachniss, and G. Grisetti, "MD-SLAM: Multi-Cue Direct SLAM," in *Proc. of the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2022.

[16] B. Della Corte, I. Bogoslavskyi, C. Stachniss, and G. Grisetti, "A general framework for flexible multi-cue photometric point cloud registration," in *Proc. of the IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018.

[17] J. Lin and F. Zhang, "Loam livox: A fast, robust, high-precision lidar odometry and mapping package for lidars of small fov," in *Proc. of the IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 3126–3131, 2019.

[18] 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, 2022.

[19] 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, 2012.

[20] H.-A. Loeliger, "An introduction to factor graphs," *IEEE Signal Processing Magazine*, vol. 21, DOI 10.1109/MSP.2004.1267047, no. 1, pp. 28–41, 2004.

[21] C. Merfels and C. Stachniss, "Pose fusion with chain pose graphs for automated driving," in *Proc. of the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, pp. 3116–3123, 2016.

[22] E.-H. Shin, "Estimation techniques for low-cost inertial navigation," Ph.D. dissertation, Department of Geomatics Engineering, University of Calgary, Calgary, Canada, 2005.

[23] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. London, United Kingdom: Artech House, 2013.

[24] S. I. Roumeliotis, G. S. Sukhatme, and G. A. Bekey, "Circumventing dynamic modeling: Evaluation of the error-state Kalman filter applied to mobile robot localization," in *Proc. of the IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 2, pp. 1656–1663, 1999.

[25] X. Niu, Y. Wu, and J. Kuang, "Wheel-INS: A wheel-mounted MEMS IMU-based dead reckoning system," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 10, pp. 9814–9825, 2021.

[26] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration theory for fast and accurate visual-inertial navigation," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2015.

[27] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, 2000.

[28] L.-T. Hsu, F. Huang, H.-F. Ng, G. Zhang, Y. Zhong, X. Bai, and W. Wen, "Hong Kong UrbanNav: An open-source multisensory dataset for benchmarking urban navigation algorithms," *NAVIGATION: Journal of the Institute of Navigation*, vol. 70, DOI 10.33012/navi.602, no. 4, 2023.

[29] J. Yin, A. Li, T. Li, W. Yu, and D. Zou, "M2DGR: A multi-sensor and multi-scenario slam dataset for ground robots," *IEEE Robotics and Automation Letters.*, vol. 7, DOI 10.1109/LRA.2021.3138527, no. 2, pp. 2266–2273, 2021.

[30] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The newer college dataset: Handheld lidar, inertial and vision with ground truth," in *Proc. of the IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, DOI 10.1109/IROS45743.2020.9340849, pp. 4353–4360, 2020.

[31] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proc. of the IEEE Conf. Comput. Vis. Pattern. Recog. (CVPR)*, pp. 3354–3361, 2012.