2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Extrinsic Calibration of Multiple LiDARs of Small FoV in Targetless Environments

Xiyuan Liu[1] and Fu Zhang[1]

***Abstract*— The integration of multiple solid state LiDARs could achieve similar performance as a spinning LiDAR, by focusing on the dedicated area of interests. However, due to their small FoV settings, it is either required to rely on external sensors or form FoV overlaps to calibrate the extrinsic parameters between multiple LiDAR units. To overcome such limitations, we develop a targetless calibration method, which creates FoV overlaps (hence co-visible features) through movements and constructs a factor graph to resolve the constraints between LiDAR poses and extrinsic parameters. By solving the formulated problem with graph optimization, our proposed method could calibrate the extrinsic of LiDARs with few or even no overlapped FoVs, meanwhile, produce a globally consistent point cloud map. Experiments on different sensor setups and scenes have demonstrated the accuracy and robustness of our proposed approach.**

I. Introduction

The rapid development of solid state LiDARs based on such as micro-electromechanical systems (MEMS) or rotating prisms [1, 2] have made them a promising type of sensor for a variety of applications such as autonomous driving [3], mapping [4] and object tracking [5]. Compared with mechanical spinning LiDAR, e.g., Velodyne[1], solid state LiDARs enjoy a more compact size, higher resolution and is more cost efficient and space saving. One drawback of such LiDARs is their small Field-of-View (FoV). To overcome this issue, researchers usually integrate them together under precise rigid body transformations, i.e., extrinsic parameters. In this paper, our work deals with the extrinsic calibration of multiple LiDARs of small FoV.

Two methods are common to address the extrinsic calibration. One is the so-called Hand-Eye calibration, which exploits the fact that different LiDARs experience the same rigid motion [6]. This method requires sufficient excitation from the robot motion in all six degrees of freedom (6-DoF), which is usually not satisfied for constrained robots motion (e.g., not much elevation or pitching on flat ground) [7]. Moreover, the calibration accuracy is highly dependent on the trajectory estimation of each LiDAR that is even more challenging due to the limited FoV and strong motion distortion [4]. The other method is to calibrate two neighboring LiDARs based on their co-visible geometrical features in the FoV overlap (Fig. 1(c) and Fig. 1(d)) [8]–[10]. However, this FoV overlap may not exist or is very small (e.g., to maximize the FoV with the smallest number of LiDARs, see Fig. 1(e) and Fig. 1(f)), leading to large calibration error in the pairwise calibration. Such pairwise errors will further accumulate and cause considerable drift when looping the LiDAR chain (Fig. 1(d)).

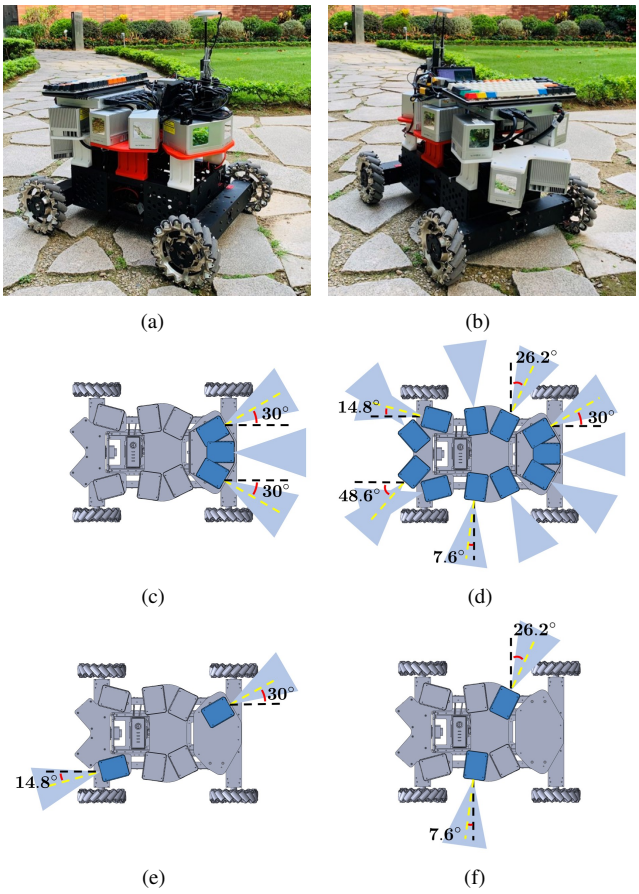


Fig. 1. Our customized multi-LiDAR vehicle platform with views from the front (a) and rear (b) side. Our method works for an arbitrary number of LiDARs with ((c) and (d)) or without ((e) and (f)) FoV overlap and produces globally consistent extrinsic estimation. The accompanying experiment video is available on https://youtu.be/SV9WfcQbq48.

To address the above challenges, in this paper we propose an automatic and targetless extrinsic calibration method for multiple LiDARs of small FoV. The proposed method does not require any external sensor (e.g., high-precision GPS/IMU) for LiDAR motion estimation and is scalable to any number of LiDARs of any FoV overlap configuration (see Fig. 1). In summary, our contributions are:

- We propose a full pipeline aimed for multi-LiDAR extrinsic calibration: data collection, graph construction,

*This project is funded by DJI (Project no. 200009538).

[1]Xiyuan Liu and Fu Zhang are with the Department of Mechanical Engineering, The University of Hong Kong, Hong Kong Special Administrative Region, People's Republic of China. {xliuaa@connect,fuzhang@}hku.hk

[1]https://velodynelidar.com/

and graph optimization. During the data collection, we artificially create FoV overlaps (and hence co-visible features) by introducing motions to the system: placing the system at multiple stationary poses to generate constraints for extrinsic parameters and the unknown robot poses. Then, we employ a KD-tree for feature correspondence searching and build a factor-graph based on such correspondences. Finally, we propose a multi-stage graph optimization method that can simultaneously estimate the extrinsic parameters and the robot poses.
- We present a metric to evaluate the extrinsic calibration results which could accurately estimate the extrinsic errors and the geometric consistency of plane and edge features in the reconstructed point cloud.
- We verify our method in different environments and on multiple LiDAR configurations, including three LiDARs with decent overlap, two LiDARs facing the opposite direction with no FoV overlap, and a chain of eleven LiDARs with few pairwise FoV overlap. Results show that the proposed method achieves high accuracy and produces a globally consistent map with very coarse initial values.

II. Related Works

Current multi-LiDAR extrinsic calibration methods are mainly divided into two categories: with and without external sensors. Pure LiDAR-based methods further break into targetless and target-based. Targetless methods proposed in [7, 11] calibrate the extrinsic parameters by utilizing planar structures (e.g., ground, walls) commonly seen by all LiDARs. These methods are verified on mechanical spinning LiDARs with full 360° FoV. As pointed out by the author, this method requires all LiDARs to have overlapping FoVs. In [12], 2D and 3D LiDARs are first locally pairwise calibrated with co-visible edge and plane features and then globally optimized with a hypergraph. Similar to [7, 11], the method in [12] requires overlapping FoV among all LiDARs. Target-based approaches have been studied in [8, 9] which utilize retro-reflective targets in the environment to assist the 3D feature (e.g., edge, plane, cone) extraction and relax the original non-linear problem into a second-order cone programming. Although target-based methods have reduced the computation complexity, setting up the targets requires additional work and time.

External sensors such as GPS/IMU and camera have also been employed to facilitate the LiDAR calibration. Methods in [13]–[15] utilize high resolution INS sensors to compensate the LiDAR motions, enabling each LiDAR to scan at a different time (hence pose) and register into a single aggregated scan. Then, the extrinsic parameters can be solved by aligning the aggregated scans of each LiDAR based on either Generalized ICP [16] and Rényi Quadratic Entropy [17]. The Rényi Quadratic Entropy-based map alignment has also been used to calibrate the extrinsic parameters of 2D-LiDARs in [18]. In this case, multiple 2D LiDARs are continuously rotated to create constraints among different LiDARs. Since the calibration utilizes the rotation angle measured by onboard encoders, this method is essentially a type of sensor-based approach. Apart from GPS/IMU, camera sensors provide dense and semantic information which could compensate for the sparsity of features in a point cloud from spinning LiDARs. In [19]–[21], authors separately calibrate each LiDAR w.r.t. the camera and jointly optimize the sensor pairs with graph optimization. Such a camera-assisted calibration is usually of high system complexity. The calibration error could also accumulate due to the lack of direct constraints among LiDARs.

Our method is targetless and does not rely on any external sensors. Compared with existing methods such as [7, 11, 12], our method does not require any overlapping FoV among the LiDARs to be calibrated. The key idea is to move the sensor suite to different poses, creating co-visible features for LiDARs at different poses. Compared with [13]–[15] which also create such co-visible features by motion, our method does not require any external sensor (e.g., GPS, IMU, encoders), but estimates the sensor's pose along with the extrinsic parameters by optimizing the map consistency. This general idea has been previously explored in the extrinsic calibration of camera and IMU [22], 3D LiDAR and GPS/INS [23], and 2D LiDAR and RGB-D camera [24]. Our work mainly focuses on the extrinsic of multiple solid-state LiDARs that have FoVs considerably smaller than spinning LiDARs used in previous work (e.g., 38.4° versus 360° FoV).

The rest of the paper is as follows, Section III elaborates our methodology in problem formulation and the development of factor graph solution. Section IV presents the detailed implementation and the performance evaluation of our proposed approach in the real-world environment. Conclusion and discussion are presented in Section V.

III. Methodology

A. Overview

Let us consider a robot platform rigidly mounted with multiple small FoV LiDARs and the goal of extrinsic calibration is to transform the point cloud from different sources into the same coordinate frame. We denote $\mathcal{L} = \{L_0, L_1, ..., L_{n-1}\}$ the set of n LiDARs, where L_0 is the base LiDAR used as reference and $L_1, ..., L_{n-1}$ are the LiDARs to be calibrated. The rigid transformation from coordinate frame a to frame b is denoted by ${}^b_a\mathbf{T} = ({}^b_a\mathbf{R}, {}^b_a\mathbf{t}) \in SE(3)$, where ${}^b_a\mathbf{R} \in SO(3)$ and ${}^b_a\mathbf{t} \in \mathbb{R}^3$ are respectively the rotation and translation. Denote $\mathcal{C} = \{{}^{L_0}_{L_1}\mathbf{T}, ..., {}^{L_0}_{L_n\ 1}\mathbf{T}\}$ the set of $n-1$ extrinsic parameters to be calibrated.

To create co-visible features among different LiDARs with small or even no overlap, we move (usually rotate) the sensor suite at m poses, respectively at time $\mathcal{T} = \{t_0, ..., t_{m-1}\}$. Denote $\mathcal{S} = \{{}^G_{L_0}\mathbf{T}_{t_1}, ..., {}^G_{L_0}\mathbf{T}_{t_m\ 1}\}$ the pose sequence of the base LiDAR L_0 relative to a global frame fixed at the first pose (i.e., ${}^G_{L_0}\mathbf{T}_{t_0} = \mathbf{I}_{4\times4}$). Hence, at time $t_j \in \mathcal{T}$, the i-th LiDAR has a pose ${}^G_{L_i}\mathbf{T}_{t_j} = {}^G_{L_0}\mathbf{T}_{t_j}\,{}^{L_0}_{L_i}\mathbf{T}$ and scans a local point cloud patch denoted as $\mathcal{P}_{L_i,t_j}$, $L_i \in \mathcal{L}, t_j \in \mathcal{T}$. This point cloud is in the LiDAR's local frame and can be transformed to the global frame as below

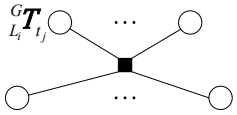
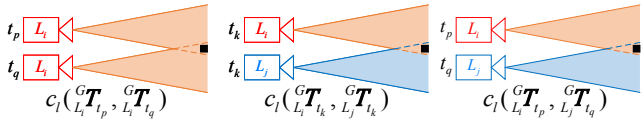


Fig. 2. The factor graph corresponding to the l-th feature, where the black square represents the factor item and the node represents the involved pose to be estimated. The l-th factor relates multiple nodes across different times, contributing to a cost item $c_l\left({}^{G}_{L_i}\mathbf{T}_{t_j}\right)$, where $L_i \in \mathcal{L}_l$, the set of LiDARs related to the l-th factor, and $t_j \in \mathcal{T}_{l,i}$, the set of times at which the i-th LiDAR is related to the l-th factor. It is obvious that $\mathcal{L}_l \subseteq \mathcal{L}$ and $\mathcal{T}_{l,i} \subseteq \mathcal{T}$.



(a) (b) (c)

Fig. 3. (a) A feature relating the same LiDAR across different times. This contributes to LiDAR localization. (b) A feature relating different LiDARs at the same time. This is the case where the two LiDARs have FoV overlaps and contributes to the extrinsic calibration. (c) A factor relating different LiDARs across different times. This creates a correlation between two LiDARs even without FoV overlap and contributes to extrinsic calibration.

$$
\begin{aligned}
{}^{G}\mathcal{P}_{L_i,t_j} &= {}^{G}_{L_i}\mathbf{T}_{t_j}\mathcal{P}_{L_i,t_j} \\
&\triangleq \{{}^{G}_{L_i}\mathbf{R}_{t_j}\mathbf{p}_{L_i,t_j} + {}^{G}_{L_i}\mathbf{t}_{t_j},\ \forall \mathbf{p}_{L_i,t_j} \in \mathcal{P}_{L_i,t_j}\}
\end{aligned} \tag{1}
$$

For a certain location (e.g., a feature) contained in the point cloud patch $\{{}^{G}\mathcal{P}_{L_i,t_j}\}$, a map consistency indicator $c_l(\cdots, {}^{G}\mathcal{P}_{L_i,t_j}, \cdots)$ is defined. This map consistency indicator creates a constraint for ${}^{G}_{L_i}\mathbf{T}_{t_j} = {}^{G}_{L_0}\mathbf{T}_{t_j}\,{}^{L_0}_{L_i}\mathbf{T}$ (see Fig. 2). Collecting all such consistency indicators at multiple locations creates a factor graph for the extrinsic parameters $\mathcal{C}$ and robot pose $\mathcal{S}$ (see Fig. 3).

Various methods could be used as the the map consistency indicator $c_l(\cdot)$, such as generalized ICP [16] or Rényi Quadratic Entropy [17]. In this paper, we use generalized ICP. Moreover, to save computation time, we only conduct generalized ICP at strong features, such as edge features and plane features extracted by the methods in [4]. In brief, the formulated problem is

$$
\begin{aligned}
&\min_{\mathcal{C},\mathcal{S}} c(\mathcal{C},\mathcal{S}) = \min_{\mathcal{C},\mathcal{S}} \sum_{l} c_l\left(\cdots, {}^{G}_{L_i}\mathbf{T}_{t_j}\mathcal{P}_{L_i,t_j}, \cdots\right) \\
&= \min_{\mathcal{C},\mathcal{S}} \left(\sum_{{}^{G}\mathbf{p}\in{}^{G}\mathcal{P}^{p}}^{N_p} \|r_{p2p}({}^{G}\mathbf{p})\|^2 + \sum_{{}^{G}\mathbf{p}\in{}^{G}\mathcal{P}^{e}}^{N_e} \|r_{p2e}({}^{G}\mathbf{p})\|^2 \right)
\end{aligned} \tag{2}
$$

where ${}^{G}\mathcal{P}^{p}, {}^{G}\mathcal{P}^{e}$ are the set of all plane and edge feature points for all LiDARs at all times, i.e.,

$$
\begin{aligned}
{}^{G}\mathcal{P}^{p} &= \{{}^{G}\mathcal{P}^{p}_{L_i,t_j}; L_i \in \mathcal{L}, t_j \in \mathcal{T}\} \\
{}^{G}\mathcal{P}^{e} &= \{{}^{G}\mathcal{P}^{e}_{L_i,t_j}; L_i \in \mathcal{L}, t_j \in \mathcal{T}\}
\end{aligned} \tag{3}
$$

and N_p, N_e are the total number of such feature points. r_{p2p} and r_{p2e} are the residuals contributed by these two types of features and they are detailed in Section III-B and Section III-C, respectively. The formulated problem in (2) is initialized by Section III-E.1 and then solved by a multi-stage optimization method in Section III-E.3.

B. Point-to-Plane Residual

Let ${}^{G}\mathbf{p}_{L_i,t_j} \in {}^{G}\mathcal{P}^{p}_{L_i,t_j}$ denote a plane feature point captured by LiDAR L_i at time t_j and ${}^{G}\mathcal{M}^{p}_{L_i,t_j}$ be the rest points, i.e., ${}^{G}\mathcal{M}^{p}_{L_i,t_j} = {}^{G}\mathcal{P}^{p} \backslash {}^{G}\mathcal{P}^{p}_{L_i,t_j}$, we calculate the *point-to-plane* projection error, i.e., the Euclidean distance between ${}^{G}\mathbf{p}_{L_i,t_j}$ and the plane defined by the nearest κ points in ${}^{G}\mathcal{M}^{p}_{L_i,t_j}$, i.e.

$$
r_{p2p}\left({}^{G}\mathbf{p}_{L_i,t_j}\right) = \mathbf{n}\mathbf{n}^{T}\left({}^{G}\mathbf{p}_{L_i,t_j} - {}^{G}\mathbf{p}_{c}\right) \tag{4}
$$

where ${}^{G}\mathbf{p}_{c}$ is the mean of all the κ nearest points and $\mathbf{n}$ is the direction of the eigenvector corresponding to the minimum eigenvalue of $\mathbf{S}$ matrix below:

$$
\begin{aligned}
{}^{G}\mathbf{p}_{c} &= \frac{1}{\kappa}\sum_{k=1}^{\kappa}\mathbf{p}_k;\ \mathbf{p}_k \in {}^{G}\mathcal{M}^{p}_{L_i,t_j} \\
\mathbf{S} &= \sum_{k=1}^{\kappa}\left(\mathbf{p}_k - {}^{G}\mathbf{p}_{c}\right)\left(\mathbf{p}_k - {}^{G}\mathbf{p}_{c}\right)^{T}
\end{aligned} \tag{5}
$$

Due to the measurement noise of the LiDAR, in practice, we obtain a threshold α to represent the smoothness of the surface. The factor is added only when $\lambda_3 \leq \alpha$, where λ_3 denotes the smallest eigenvalue of $\mathbf{S}$.

C. Point-to-Edge Residual

Similarly, let ${}^{G}\mathbf{p}_{L_i,t_j} \in {}^{G}\mathcal{P}^{e}_{L_i,t_j}$ denote an edge feature point captured by LiDAR L_i at time t_j and ${}^{G}\mathcal{M}^{e}_{L_i,t_j}$ be the rest points, i.e., ${}^{G}\mathcal{M}^{e}_{L_i,t_j} = {}^{G}\mathcal{P}^{e} \backslash {}^{G}\mathcal{P}^{e}_{L_i,t_j}$. The *point-to-edge* projection error, i.e., the Euclidean distance between ${}^{G}\mathbf{p}_{L_i,t_j}$ and the edge defined by the nearest κ points in ${}^{G}\mathcal{M}^{e}_{L_i,t_j}$ is

$$
r_{p2e}\left({}^{G}\mathbf{p}_{L_i,t_j}\right) = \left(\mathbf{I} - \mathbf{u}\mathbf{u}^{T}\right)\left({}^{G}\mathbf{p}_{L_i,t_j} - {}^{G}\mathbf{p}_{c}\right) \tag{6}
$$

where ${}^{G}\mathbf{p}_{c}$ is the mean of all the κ nearest points and $\mathbf{u}$ is the direction of the eigenvector corresponding to the maximum eigenvalue of matrix $\mathbf{S}$ in (5). We obtain a threshold β to represent the linearity of the edge that the factor is added only when $\lambda_2 \leq \beta$, where λ_2 denotes the second smallest eigenvalue of $\mathbf{S}$.

D. Graph Optimization

To solve the optimization problem in (2) with residuals defined in (4) and (6), we iterate between feature correspondence and nonlinear optimization until convergence (see Algorithm 1). At each iteration, for a certain type of feature point ${}^{G}\mathbf{p}_{L_i,t_j}$, a KD-tree is constructed from ${}^{G}\mathcal{M}_{L_i,t_j} = {}^{G}\mathcal{P} \backslash {}^{G}\mathcal{P}_{L_i,t_j}$ with current $\mathcal{S}$ and $\mathcal{C}$ to facilitate the search of the κ nearest points. Given the found κ feature points, the residual related to this feature type is added to the cost function (2).

For the given feature correspondences, the nonlinear optimization (2) is solved with the Levenberg-Marquardt (LM) method, which approximates the residuals by their first order approximations. Inspecting the residual in (4) and (6), we notice that all of $\mathbf{n}, \mathbf{u}$, ${}^{G}\mathbf{p}_{c}$ and ${}^{G}\mathbf{p}_{L_i,t_j}$ are dependent on poses $\mathcal{S}$ and extrinsic parameters $\mathcal{C}$ to be optimized. The full derivative of the residual would be very complicated to calculate due to the eigenvalue decomposition used to

Algorithm 1: Graph Optimization

Input : $\mathcal{S}, {}^{G}\mathcal{P}, \mathcal{C}$
Output: $\mathcal{S}^*, \mathcal{C}^*$

while *cost function* (2) *is not minimized* **do**
  for ${}^{G}\mathbf{p}_{L_i,t_j} \in {}^{G}\mathcal{P}_{L_i,t_j}$ **do**
    Construct ${}^{G}\mathcal{M}_{L_i,t_j} = {}^{G}\mathcal{P} \backslash {}^{G}\mathcal{P}_{L_i,t_j}$
    if κ *nearest points of* ${}^{G}\mathbf{p}_{L_i,t_j}$ *are found in* ${}^{G}\mathcal{M}_{L_i,t_j}$ **then**
      Add corresponding residual (4) or (6) to cost function
    end
  end
  Solve (2) with graph optimization and update $\mathcal{S}, \mathcal{C}$
end

obtain the vector $\mathbf{n}$ (or $\mathbf{u}$). In this paper, we simply ignore the dependence of $\mathbf{n}$ (or $\mathbf{u}$) and ${}^{G}\mathbf{p}_c$ on the optimization variables $\mathcal{C}$, $\mathcal{S}$ and only consider the derivative caused by ${}^{G}\mathbf{p}_{L_i,t_j}$. This simple approximation works well in practice (see Section IV-C).

To derive the derivative of the residual with respect to $\mathcal{C}$ and $\mathcal{S}$, which are both elements on $SE(3)$, we view them as $SO(3) \times \mathbb{R}^3$ and parameterize them in the tangent space of the current value:

$$\begin{aligned} {}^{G}_{L_i}\mathbf{R}_{t_j} &\leftarrow {}^{G}_{L_i}\mathbf{R}_{t_j} \mathrm{Exp}\left({}^{G}_{L_i}\boldsymbol{\varphi}_{t_j}\right) \\ {}^{G}_{L_i}\mathbf{t}_{t_j} &\leftarrow {}^{G}_{L_i}\mathbf{t}_{t_j} + {}^{G}_{L_i}\boldsymbol{\rho}_{t_j} \end{aligned} \tag{7}$$

Substituting (7) into (4) and (6) leads to

$$\begin{aligned} \frac{\partial r_{p2p}({}^{G}\mathbf{p}_{L_i,t_j})}{\partial \left({}^{G}_{L_i}\boldsymbol{\varphi}_{t_j}\right)} &= -\mathbf{n}\mathbf{n}^T \, {}^{G}_{L_i}\mathbf{R}_{t_j} \left(\mathbf{p}_{L_i,t_j}\right)^{\wedge} \\ \frac{\partial r_{p2p}({}^{G}\mathbf{p}_{L_i,t_j})}{\partial \left({}^{G}_{L_i}\boldsymbol{\rho}_{t_j}\right)} &= \mathbf{n}\mathbf{n}^T \\ \frac{\partial r_{p2e}({}^{G}\mathbf{p}_{L_i,t_j})}{\partial \left({}^{G}_{L_i}\boldsymbol{\varphi}_{t_j}\right)} &= \left(\mathbf{u}\mathbf{u}^T - \mathbf{I}\right) {}^{G}_{L_i}\mathbf{R}_{t_j} \left(\mathbf{p}_{L_i,t_j}\right)^{\wedge} \\ \frac{\partial r_{p2e}({}^{G}\mathbf{p}_{L_i,t_j})}{\partial \left({}^{G}_{L_i}\boldsymbol{\rho}_{t_j}\right)} &= \mathbf{I} - \mathbf{u}\mathbf{u}^T \end{aligned} \tag{8}$$

Since

$${}^{G}_{L_i}\mathbf{T}_{t_j} = {}^{G}_{L_0}\mathbf{T}_{t_j} \, {}^{L_0}_{L_i}\mathbf{T} \tag{9}$$

we have

$${}^{G}_{L_i}\mathbf{R}_{t_j} = {}^{G}_{L_0}\mathbf{R}_{t_j} \, {}^{L_0}_{L_i}\mathbf{R}; \;\; {}^{G}_{L_i}\mathbf{t}_{t_j} = {}^{G}_{L_0}\mathbf{R}_{t_j} \, {}^{L_0}_{L_i}\mathbf{t} + {}^{G}_{L_0}\mathbf{t}_{t_j} \tag{10}$$

and with perturbations

$$\begin{aligned} {}^{G}_{L_i}\mathbf{R}_{t_j}\mathrm{Exp}({}^{G}_{L_i}\boldsymbol{\varphi}_{t_j}) &= {}^{G}_{L_0}\mathbf{R}_{t_j}\mathrm{Exp}({}^{G}_{L_0}\boldsymbol{\varphi}_{t_j}) \, {}^{L_0}_{L_i}\mathbf{R} \; \mathrm{Exp}({}^{L_0}_{L_i}\boldsymbol{\varphi}); \\ {}^{G}_{L_i}\mathbf{t}_{t_j} + {}^{G}_{L_i}\boldsymbol{\rho}_{t_j} &= {}^{G}_{L_0}\mathbf{R}_{t_j}\mathrm{Exp}({}^{G}_{L_0}\boldsymbol{\varphi}_{t_j}) \left({}^{L_0}_{L_i}\mathbf{t} + {}^{L_0}_{L_i}\boldsymbol{\rho}\right) \\ &\quad + {}^{G}_{L_0}\mathbf{t}_{t_j} + {}^{G}_{L_0}\boldsymbol{\rho}_{t_j} \end{aligned} \tag{11}$$

The two equations (10) and (11) lead to

$$\begin{aligned} {}^{G}_{L_i}\boldsymbol{\varphi}_{t_j} &= {}^{L_0}_{L_i}\mathbf{R}^T \, {}^{G}_{L_0}\boldsymbol{\varphi}_{t_j} + {}^{L_0}_{L_i}\boldsymbol{\varphi} \\ {}^{G}_{L_i}\boldsymbol{\rho}_{t_j} &= -\,{}^{G}_{L_0}\mathbf{R}_{t_j} \left({}^{L_0}_{L_i}\mathbf{t}\right)^{\wedge} {}^{G}_{L_0}\boldsymbol{\varphi}_{t_j} + {}^{G}_{L_0}\boldsymbol{\rho}_{t_j} + {}^{G}_{L_0}\mathbf{R}_{t_j} \, {}^{L_0}_{L_i}\boldsymbol{\rho} \end{aligned} \tag{12}$$

Now collecting all the local parameters

$$\mathbf{x}^T = \big[\underbrace{\cdots \quad {}^{G}_{L_0}\boldsymbol{\varphi}_{t_j}^T \quad {}^{G}_{L_0}\boldsymbol{\rho}_{t_j}^T \quad \cdots}_{\mathcal{S} \text{ block, } \mathbf{x}_{\mathcal{S}}^T \in \mathbb{R}^{6(m\ \ 1)}} \quad \underbrace{\cdots \quad {}^{L_0}_{L_i}\boldsymbol{\varphi}^T \quad {}^{L_0}_{L_i}\boldsymbol{\rho}^T \quad \cdots}_{\mathcal{C} \text{ block, } \mathbf{x}_{\mathcal{C}}^T \in \mathbb{R}^{6(n\ \ 1)}}\big]$$

Therefore, according to the chain rule, we have

$$\mathbf{J}_p = \frac{\partial r_{p2p}({}^{G}\mathbf{p}_{L_i,t_j})}{\partial \mathbf{x}} = \begin{bmatrix} \mathbf{J}_p^{\mathcal{S}} & \mathbf{J}_p^{\mathcal{C}} \end{bmatrix} \in \mathbb{R}^{3\times 6(m+n-2)} \tag{13}$$

where

$$\mathbf{J}_p^{\mathcal{S}} = \begin{bmatrix} 0 & \cdots & 0 & \underbrace{-\mathbf{n}\mathbf{n}^T\mathbf{U}^{\mathcal{S}} \quad \mathbf{n}\mathbf{n}^T}_{j\text{-th block, dimension } 3\times 6} & 0 & \cdots & 0 \end{bmatrix}$$

$$\mathbf{U}^{\mathcal{S}} = {}^{G}_{L_i}\mathbf{R}_{t_j} \left(\mathbf{p}_{L_i,t_j}\right)^{\wedge} {}^{L_0}_{L_i}\mathbf{R}^T + {}^{G}_{L_0}\mathbf{R}_{t_j} \left({}^{L_0}_{L_i}\mathbf{t}\right)^{\wedge}$$

$$\mathbf{J}_p^{\mathcal{C}} = \begin{bmatrix} 0 & \cdots & 0 & \underbrace{-\mathbf{n}\mathbf{n}^T\mathbf{U}^{\mathcal{C}} \quad \mathbf{n}\mathbf{n}^T \, {}^{G}_{L_0}\mathbf{R}_{t_j}}_{i\text{-th block, dimension } 3\times 6} & 0 & \cdots & 0 \end{bmatrix}$$

$$\mathbf{U}^{\mathcal{C}} = {}^{G}_{L_i}\mathbf{R}_{t_j} \left(\mathbf{p}_{L_i,t_j}\right)^{\wedge}$$

and

$$\mathbf{J}_e^{\mathcal{S}} = \begin{bmatrix} 0 & \cdots & 0 & \underbrace{-\mathbf{V}\mathbf{U}^{\mathcal{S}} \quad \mathbf{V}}_{j\text{-th block, dimension } 3\times 6} & 0 & \cdots & 0 \end{bmatrix}$$

$$\mathbf{J}_e^{\mathcal{C}} = \begin{bmatrix} 0 & \cdots & 0 & \underbrace{-\mathbf{V}\mathbf{U}^{\mathcal{C}} \quad \mathbf{V} \, {}^{G}_{L_0}\mathbf{R}_{t_j}}_{i\text{-th block, dimension } 3\times 6} & 0 & \cdots & 0 \end{bmatrix}$$

$$\mathbf{V} = \mathbf{I} - \mathbf{u}\mathbf{u}^T$$

E. Calibration Pipeline

The workflow of our proposed approach is shown in Fig. 4, where each stage is detailed below:

1) Stage 1, online pose estimation and feature extraction during data collection: Since the edge and plane feature correspondences and residuals are all established in the global frame, the poses of all LiDARs need to be initialized well to ensure convergence. To do so, we run a LiDAR Odometry and Mapping (LOAM) [4] for each LiDAR. The LOAM package extracts all edge and plane feature points online by computing their local curvatures. Points of low curvature are viewed as edge features and those of high curvature are viewed as plane features [4]. Based on the extracted features, the LOAM further computes the LiDAR pose by aligning the feature points to edges and planes in the incrementally-built map, generating a pose trajectory of each LiDAR unit. The pose trajectory of the base LiDAR (i.e., $\mathcal{S}$) and the feature points of all LiDAR units are sent to stage 2 for further processing.

2) Stage 2, extrinsic refining: Given an initial extrinsic value (e.g., from Hand-Eye calibration), we perform the following two steps to refine the extrinsic:

Step 1. Refining base LiDAR poses $\mathcal{S}$: The base LiDAR poses $\mathcal{S}$ are further improved by optimizing (2) over $\mathcal{S}$ and considering factors due to feature points measured by L_0 (i.e., $\{{}^{G}\mathcal{P}^{p,e}_{L_0,t_j}, t_j \in \mathcal{T}\}$) only. This relates all feature points of L_0 across different times and creates a globally consistent map.

Step 2. Refining extrinsic parameters $\mathcal{C}$: With the refined pose trajectory $\mathcal{S}$ of L_0, we improve the extrinsic parameter (from the given initial value) by aligning the feature points

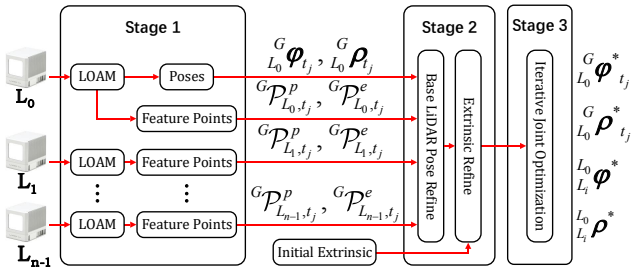
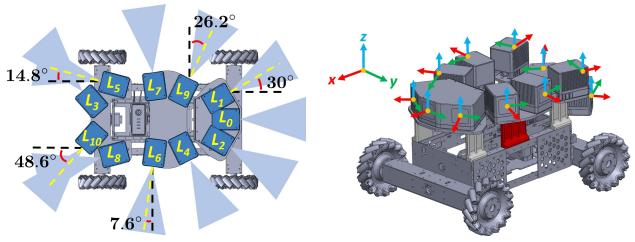
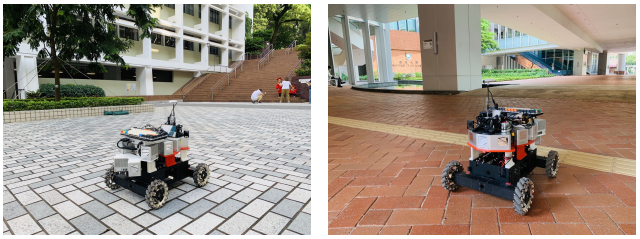


Fig. 4. The block diagram of our proposed multi-LiDAR calibration system. Stage 1: Base LiDAR poses initialization and feature extraction. Stage 2: refinement of base lidar pose and extrinsic parameter. Stage 3: Iterative and joint graph optimization.



(a) (b)

Fig. 5. Our customized multi-LiDAR vehicle platform. Left: the FoV coverage of each LiDAR with their designated numbers. Right: the orientation of each LiDAR denoted in the right-handed coordinate system.



(a) Scene-1 (b) Scene-2

Fig. 6. Our experiment test scenes.

of the LiDAR to the map from L_0. Repeating this map alignment for each LiDAR leads to a good initialization of extrinsic parameter $\mathcal{C}$.

3) Stage 3, Iterative and joint optimization: After the refinement in stage 2, the $\mathcal{S}$ and $\mathcal{C}$ are iteratively optimized from (2). That is, first holding $\mathcal{C}$ constant and optimize (2) over $\mathcal{S}$ only using Algorithm 1 in Section III-D; then optimizing $\mathcal{C}$ while holding $\mathcal{S}$ constant; these two steps are iterated until convergence. With the iterative optimization, both $\mathcal{S}$ and $\mathcal{C}$ are well optimized and suitable for joint optimization to further improve the map consistency, as shown in Algorithm 1.

IV. Experiments

A. Experiment Setup

Fig. 5 shows our customized vehicle platform, we use one MID-100 LiDAR and eight MID-40 LiDARs manufactured by Livox[2]. Each MID-40 LiDAR has $38.4°$ FoV and the MID-100 LiDAR consists of three MID-40 LiDARs (denoted as L_0, L_1 and L_2). The orientations of LiDAR $L_3 \sim L_{10}$ are placed such that every two adjacent LiDARs have around $4.6°$ FoV overlap. In all the following evaluations, the ground truth of extrinsic parameters of MID-100 LiDAR are obtained from the manufacturer, and the rest of which are calculated from the CAD model. The proposed algorithm has been implemented on a high performance PC with Intel i7-9700K processor, 32GB RAM, and Ceres Solver[3]. It is noted that the satellite receiver shown in this figure is not used.

We collect the LiDAR data in two different outdoor environments, i.e., the library front yard (referred as *Scene*-1, see Fig. 6(a)) and the entrance of campus garden (referred as *Scene*-2, see Fig. 6(b)). In *Scene*-1, our LiDAR platform locates in the middle of a large flat ground and is at far surrounded by library buildings, stairs, and trees, while in *Scene*-2, the platform stands at a challenging narrow crossroad with nearby vertical objects (e.g., pillars) and dynamic objects (e.g., pedestrians). We first conduct an extrinsic initialization (which is necessary for our calibration pipeline stage 2). To do so, we move the vehicle along a figure "8" path. During the movements, a LOAM [4] is running in real-time for each LiDAR unit and computes its pose trajectory. The computed pose trajectories are then used to initialize the LiDAR extrinsic with the standard Hand-Eye calibration [25] (by solving the equation $\mathbf{AX} = \mathbf{XB}$). Then, for the data collection of our proposed method, the whole vehicle is rotated around $360°$ and stays stationary for a given time period at multiple poses. Due to the unique non-repetitive scan pattern of Livox LiDARs [1], keeping the vehicle stationary enables us to capture high-resolution point cloud patches with dense feature points. Keeping the vehicle stationary at each pose also removes the need for high accuracy hardware synchronization among all LiDARs or motion distortion compensation. We manually segment the time $\mathcal{T}$ when the LiDAR is stationary, such that all the feature points and base LiDAR poses could be read for the use of our calibration pipeline.

B. Extrinsic Evaluation Metric

A metric η is defined to evaluate the performance of the proposed extrinsic calibration algorithm, by calculating the averaged r_{p2p} and r_{p2e} as

$$\eta = \frac{\sum_{\mathbf{p}\in\mathcal{P}^p}^{N_p} \|r_{p2p}(\mathbf{p})\| + \sum_{\mathbf{p}\in\mathcal{P}^e}^{N_e} \|r_{p2e}(\mathbf{p})\|}{N_p + N_e} \tag{14}$$

where the expression of $r_{p2p}(\mathbf{p})$ and $r_{p2e}(\mathbf{p})$ are the same as (4) and (6) with $\kappa = 13$. To show the correlation between η and extrinsic calibration error, we evaluate the value of η at different extrinsic errors. More specifically, we use a MID-100 LiDAR, where the three internal LiDAR units (L_0, L_1, and L_2) have known extrinsic ground truth calibrated by the manufacturer in the factory and we place it against a large wall area. Then we manually perturb the extrinsic translation of L_2 by {4cm,3cm,2cm,1cm,0cm} along the wall's normal vector and evaluate η on the resultant point cloud obtained

[2]https://www.livoxtech.com/mid-40-and-mid-100
[3]http://ceres-solver.org/

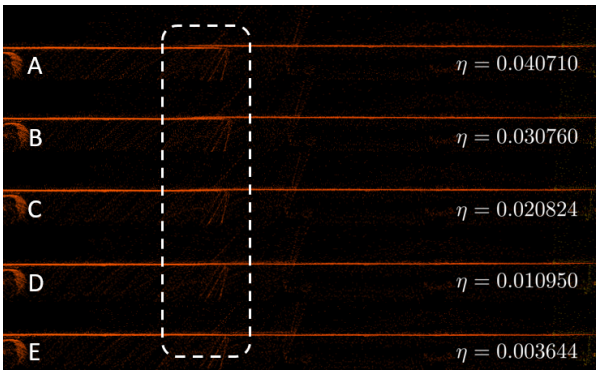


Fig. 7. Extrinsic evaluation metric η. From top to bottom: two planar surfaces (with left point cloud from L_0 and right from L_2) when manually perturbing the extrinsic translation ${}^{L_0}_{L_2}\mathbf{t}$ from its ground truth by 4cm, 3cm, 2cm, 1cm, and 0cm. The divergence appears in the white dashed rectangle. It is clear that η elaborates the extrinsic translation error w.r.t ground truth even in cases with no clear visual difference. The length of the planar surface in this figure is around 6m in the real world.

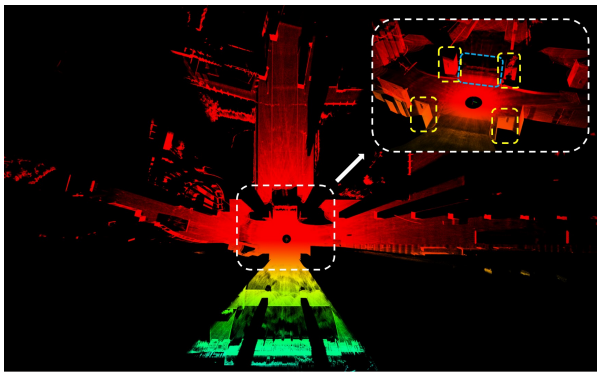


Fig. 9. The bird-eye's view of the reconstructed point cloud of eleven LiDARs in $Scene$-2. The white dashed rectangle detailed the challenges of extrinsic calibration in this scene: the discontinuity of planar surfaces on the corridor ceiling caused by four closely located pillars (yellow dashed rectangle) and the lifted ground floor (blue dashed rectangle).

by L_0 and L_2. The values of η at different values of the added extrinsic translation (i.e., extrinsic error) are shown in Fig. 7, where we can see that the η matches well with the extrinsic error both qualitatively and quantitatively. When no extrinsic error is added, the metric η = 3mm caused by the LiDAR point accuracy or the residual extrinsic error during the manufacturer in-factory calibration.

C. Real World Calibration Results

1) Evaluation on the number of poses: In this section, we use a MID-100 LiDAR to evaluate the effect of the number of poses (i.e., m) on the calibration accuracy. We rotate the LiDAR platform at $m = 35$ stationary poses in $Scene$-1 with each two consecutive stationary poses owning around 26° of FoV overlap. The middle LiDAR of MID-100 is chosen as the base LiDAR and the extrinsic parameters are initialized identically in each test, by adding 0.17rad to the ground truth of each Euler angle $(roll, pitch, yaw)$ of ${}^{L_0}_{L_i}\mathbf{R}$ and 0.1m to each axis (x, y, z) of ${}^{L_0}_{L_i}\mathbf{t}$. Half (with around 13° FoV overlap between two poses), two-thirds (with around 20° FoV overlap), and all stationary poses are used and their results are summarized in Table I.

It is seen that the algorithm is very robust, able to achieve accurate and consistent calibration results (rotation error around 0.004rad and translation error around 5mm) at the different number of poses and with a very poor initial extrinsic estimate. The more stationary poses are used, the more co-visible features (i.e., constraints) are obtained hence the smaller calibration error is; but this improvement is quite minimal after $m > 26$. In the following experiments, to save data collection and processing time, we have rotated the LiDAR platform w.r.t the robot center around with $m_{Scene_1} = 23$ and $m_{Scene_2} = 22$ stationary poses respectively.

2) Sensitivity on Initial Extrinsic Parameters: We investigate the robustness of our calibration pipeline with respect to the initial extrinsic estimate. To do so, we intentionally perturb the Hand-Eye calibration results supplied to stage 2 of the calibration pipeline (see Fig. 4). We use a MID-100 LiDAR sensor and the same data collected in Section. IV-C.1. For each LiDAR, the supplied initial extrinsic is perturbed eight times, each time the rotation is perturbed by a random angle $\theta_{rand} \in [0.15\text{rad}, 0.5\text{rad}]$ and the translation is perturbed by a random value $t_{rand} \in [0.08\text{m}, 0.30\text{m}]$.

We evaluate our algorithm in both $Scene$-1 and $Scene$-2. Their results are shown in Fig. 8. The large existence of planar ground surfaces in $Scene$-1 has provided strong feature correspondences and dragged the extrinsic parameters to the desired values, making them invulnerable to the poor initialization (with rotation and translation errors up to 0.5rad and 0.3m). In $Scene$-2 (see Fig. 9), when the rotational disturbance is greater than 0.3rad, due to the discontinuous and uneven ground surfaces, wrong feature correspondences have trapped the extrinsic parameters to local minimums. By adjusting the tuning parameters, correct feature correspondences are established and the proposed method could yield fine results as shown in Fig. 8.

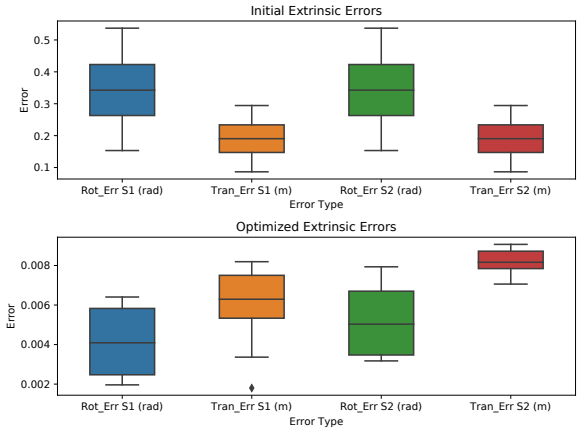


Fig. 8. Extrinsic calibration results of MID-100 LiDAR with randomly generated initial values. The initial extrinsic parameter for each LiDAR to be calibrated is set the same in both test scenes (S1 for $Scene$-1 and S2 for $Scene$-2). Each box in the plot is the statistical results from 16 setups, eight for each LiDAR. The error value (both Rot_Err and Tran_Err) is the norm of the corresponding error vector.

3) No FoV Overlap Scenario: In this section, we show the proposed method is capable of calibrating two LiDARs

TABLE I
CALIBRATION RESULTS WITH MULTIPLE POSES OF DATA

No. of Poses Used	FoV Overlap between Poses	Rotation Error (rad)		Translation Error (m)		Metric η (m)	
		Initial	Optimized	Initial	Optimized	Initial	Optimized
18 Poses	13 Degrees	0.3105966	0.0040898	0.1732051	0.0057419	0.2409940	0.0058518
26 Poses	20 Degrees	0.3105966	0.0042098	0.1732051	0.0048270	0.1554720	0.0054672
35 Poses	26 Degrees	0.3105966	**0.0039664**	0.1732051	**0.0046011**	0.1421230	**0.0054168**

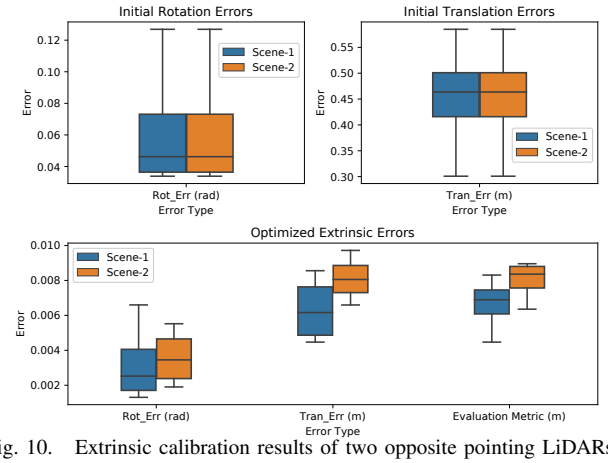


Fig. 10. Extrinsic calibration results of two opposite pointing LiDARs in two test scenes.Each box in this plot consists of four values respectively from four pairs of LiDARs, where each pair uses the same set of initial extrinsic parameters for the two scenes. The mean and standard deviation of the initial extrinsic errors are 0.0633rad and 0.0434rad for rotation, and 0.4532m and 0.1168m for translation, respectively.

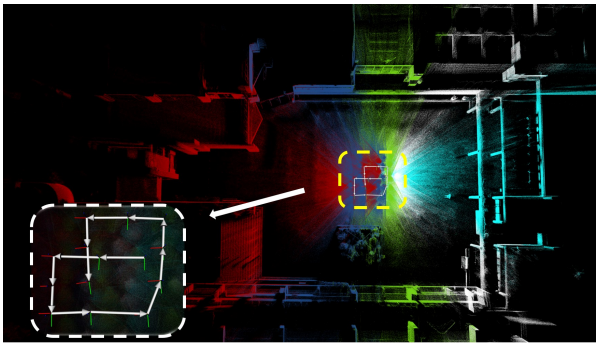


Fig. 11. Reconstructed point cloud with the proposed method using the same colorization style as Fig. 14. The white arrows indicate the robot trajectory and the axes markers display the global stationary poses. The mean and standard deviation of the extrinsic errors are 0.0232m and 0.0140m for translation and 0.0101rad and 0.0063rad for rotation, respectively.

with no view overlap. We collect the data in the way similar to Section. IV-C.1 and use four sets of LiDARs respectively, i.e., $\{L_1, L_8\}$, $\{L_2, L_5\}$, $\{L_6, L_9\}$ and $\{L_4, L_7\}$ as depicted in Fig. 1(e) and Fig. 1(f). In each set, two LiDARs are pointing in roughly opposite directions. The calibration results summarized in Fig. 10 demonstrate that our proposed method is capable of calibrating extrinsic parameters with extreme LiDAR configurations.

4) Few FoV Overlap Scenario: In this section, we demonstrate the capability of our proposed method in calibrating all eleven LiDARs with few FoV overlap as depicted in Fig. 1(d). We test two sensor movement profiles: pure

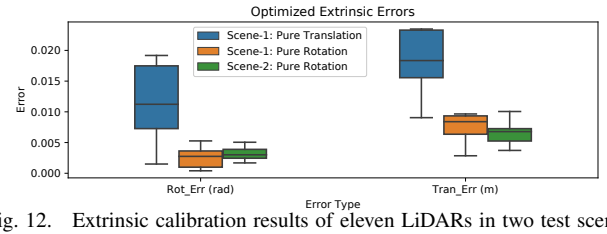


Fig. 12. Extrinsic calibration results of eleven LiDARs in two test scenes. Each box in this plot consists of ten values. The metric η in $Scene$-1 is η = 8mm and $Scene$-2 is η = 7.5mm for pure rotation profile.

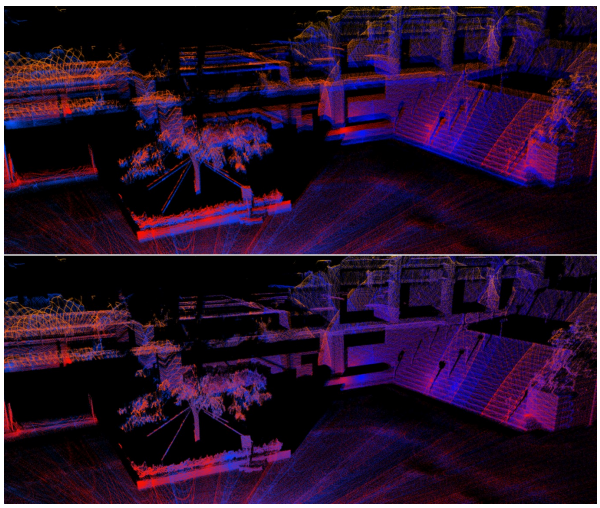


Fig. 13. Comparison between pairwise calibration and our method. The red point cloud is from L_0 whereas the blue one is from L_1. Top: the extrinsic parameter is calculated by pairwise calibration (with rotation error up to 0.016rad, translation error up to 0.022m). Bottom: the extrinsic parameter is calculated by our proposed method.

rotation (see Section. IV-C.1) in both $Scene$-1 and $Scene$-2 and pure translation in $Scene$-1. The detailed illustration of the reconstructed point cloud in $Scene$-1 (see Fig. 11 and Fig. 14) and the calibration results (see Fig. 12) show that our method achieves both acceptable accuracy and produces globally consistent point clouds in both movement profiles. In particular, the rotation movement achieves higher accuracy due to the large FoV overlap created by the motion. We further show the proposed pose graph based method outperforms the pairwise calibration method. As illustrated in Fig. 13, though the error in each pairwise calibration is negligible, the accumulated errors through ten pairs lead to an inaccurate result.

V. Conclusion

In this paper, we have presented a targetless method for multiple small FoV LiDARs simultaneously calibration

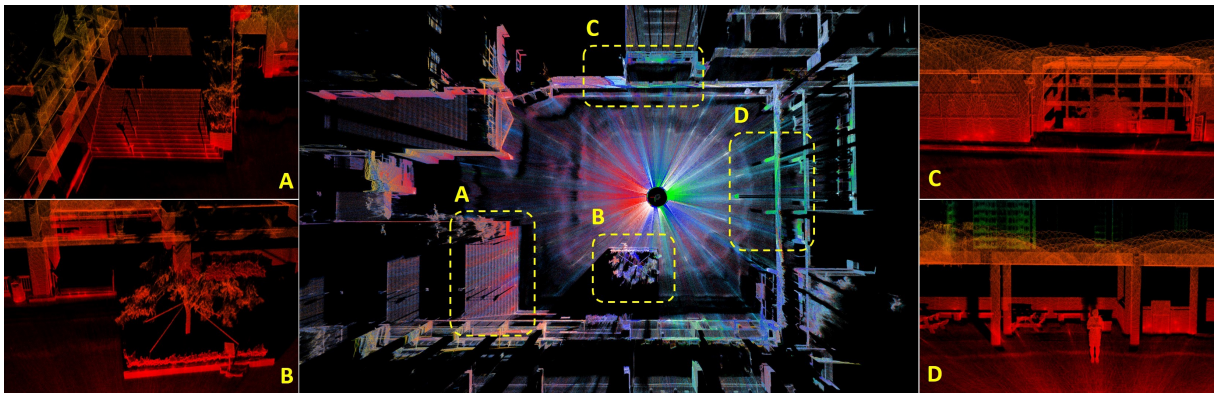


Fig. 14. The reconstructed point cloud map of eleven LiDARs in *Scene*-1. The point cloud of L_0, L_1, L_2 are colored red, L_4, L_9 are colored deep blue, L_6, L_7 are colored green, L_5, L_8 are colored cyan and L_3, L_{10} are colored white. Details of the point cloud can be zoomed in with (A) stairs, (B) tree, (C) window and (D) pillars and the author. The final evaluation metric η is 8mm.

and mapping. With initial extrinsic parameters obtained via Hand-Eye calibration, our approach does not rely on any external sensor nor assume FoV overlaps between LiDARs. By creating co-visible features through LiDARs' movement, we have verified the calibration accuracy and mapping consistency of our proposed method including data collection, graph construction, and graph optimization in two arbitrary test scenes with multiple LiDAR configurations.

Since only strong feature correspondences are considered as constraints for extrinsic optimization, the proposed method might produce unreliable results in feature-less scenes due to poor initialization and wrongly established correspondences. With a proper initialization and sufficient features in the scene, our proposed method can yield accurate extrinsic calibration results.

REFERENCES

[1] Z. Liu, F. Zhang, and X. Hong. Low-cost retina-like robotic lidars based on incommensurable scanning. *IEEE/ASME Transactions on Mechatronics*, to be published. doi:10.1109/TMECH.2021.3058173.

[2] F. Negin, A. Aijazi, L. Malaterre, L. Trassoudaine, and P. Checchin. Systematic Evaluation And Characterization Of 3D Solid State Lidar Sensors For Autonomous Ground Vehicles. In *XXIV ISPRS Congress*, volume XLIII-B1-2020, pages 199–203, Nice, France, August 2020.

[3] H. W. Yoo, N. Druml, D. Brunner, C. Schwarzl, T. Thurner, M. Hennecke, and G. Schitter. Mems-based lidar for autonomous driving. *e & i Elektrotechnik und Informationstechnik*, 135(6):408–415, 2018.

[4] 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 International Conference in Robotics and Automation (ICRA)*, 2020.

[5] A. Gunter, S. Boker, M. Konig, and M. Hoffmann. Privacy-preserving people detection enabled by solid state lidar. *2020 16th International Conference on Intelligent Environments (IE)*, 2020.

[6] J. Lin, X. Liu, and F. Zhang. A decentralized framework for simultaneous calibration, localization and mapping with multiple lidars. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4870–4877, 2020.

[7] J. Jiao, Y. Yu, Q. Liao, H. Ye, R. Fan, and M. Liu. Automatic calibration of multiple 3d lidars in urban environments. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 15–20, 2019.

[8] C. Gao and J. R. Spletzer. On-line calibration of multiple LIDARs on a mobile vehicle platform. In *2010 IEEE International Conference on Robotics and Automation*. IEEE, May 2010.

[9] B. Xue, J. Jiao, Y. Zhu, L. Zhen, D. Han, M. Liu, and R. Fan. Automatic calibration of dual-LiDARs using two poles stickered with retro-reflective tape. In *2019 IEEE International Conference on Imaging Systems and Techniques (IST)*. IEEE, December 2019.

[10] A. Dhall, Kunal Chelani, Vishnu Radhakrishnan, and K. M. Krishna. Lidar-camera calibration using 3d-3d point correspondences, 2017. arXiv:1705.09785.

[11] J. Jiao, Q. Liao, Y. Zhu, T. Liu, Y. Yu, R. Fan, L. Wang, and M. Liu. A novel dual-lidar calibration algorithm using planar surfaces. *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019.

[12] J. L. Owens, P. R. Osteen, and K. Daniilidis. Msg-cal: Multi-sensor graph-based calibration. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[13] J. Levinson and S. Thrun. Unsupervised calibration for multi-beam lasers. *Experimental Robotics Springer Tracts in Advanced Robotics*, 79:179–193, 2014.

[14] W. Maddern, A. Harrison, and P. Newman. Lost in translation (and rotation): Rapid extrinsic calibration for 2d and 3d lidars. *2012 IEEE International Conference on Robotics and Automation*, 2012.

[15] M. Billah and J. A. Farrell. Calibration of multi-lidar systems: Application to bucket wheel reclaimers. *IEEE Transactions on Control Systems Technology*, page 1–12, 2019.

[16] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[17] A. Rényi. On measures of entropy and information. *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, page 547–561, 1961.

[18] M. Sheehan, A. Harrison, and P. Newman. Self-calibration for a 3d laser. *The International Journal of Robotics Research*, 31(5):675–687, 2011.

[19] H. Chen and S. Schwertfeger. Heterogeneous multi-sensor calibration based on graph optimization. *2019 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, 2019.

[20] C. Park, P. Moghadam, S. Kim, S. Sridharan, and C. Fookes. Spatiotemporal camera-lidar calibration: A targetless and structureless approach. *IEEE Robotics and Automation Letters*, 5(2):1556–1563, 2020.

[21] J. Quenzel, N. Papenberg, and S. Behnke. Robust extrinsic calibration of multiple stationary laser range finders. *2016 IEEE International Conference on Automation Science and Engineering (CASE)*, 2016.

[22] J. Rehder, R. Siegwart, and P. Furgale. A general approach to spatiotemporal calibration in multisensor systems. *IEEE Transactions on Robotics*, 32(2):383–398, 2016.

[23] Z. Taylor and J. Nieto. Motion-based calibration of multimodal sensor extrinsics and timing offset estimation. *IEEE Transactions on Robotics*, 32(5):1215–1229, 2016.

[24] B. Della C., H. Andreasson, T. Stoyanov, and G. Grisetti. Unified motion-based calibration of mobile multi-sensor platforms with time delay estimation. *IEEE Robotics and Automation Letters*, 4(2):902–909, 2019.

[25] H. Radu and D. Fadi. Hand-eye calibration. *The International Journal of Robotics Research*, 14(3):195–210, June 1995.