Fast and Accurate Extrinsic Calibration for Multiple LiDARs and Cameras

Xiyuan Liu[1], Chongjian Yuan[1], and Fu Zhang[1]

***Abstract*—In this letter, we propose a fast, accurate, and targetless extrinsic calibration method for multiple LiDARs and cameras based on adaptive voxelization. On the theory level, we incorporate the LiDAR extrinsic calibration with the bundle adjustment method. We derive the second-order derivatives of the cost function w.r.t. the extrinsic parameter to accelerate the optimization. On the implementation level, we apply the adaptive voxelization to dynamically segment the LiDAR point cloud into voxels with non-identical sizes, and reduce the computation time in the process of feature correspondence matching. The robustness and accuracy of our proposed method have been verified with experiments in outdoor test scenes under multiple LiDAR-camera configurations.**

***Index Terms*—Calibration and Identification, Sensor Fusion, Mapping.**

I. Introduction

Multiple LiDARs and cameras have been increasingly used on mobile robots for missions such as autonomous navigation [1] and mapping [2]–[4]. This is due to the superior characteristics of the LiDAR in three-dimensional range detection and point cloud density, and the rich color information from the camera. The integration of multiple sensors could facilitate the state estimation of the robot [5], meanwhile produce a dense and colorized map (see Fig. 1). To better perceive the surrounding environment, it is worthwhile to transform the perceptions from multiple sensors into the same coordinate frame, i.e., to know the rigid transformation between each pair of senors. In this letter, our work deals with the extrinsic calibration between multiple LiDARs and cameras.

Several challenges reside in the multi-sensor extrinsic calibration: (1) limited field-of-view (FoV) overlap among the sensors. Current methods usually require the existence of common FoV between each pair of sensors [6]–[9], such that each feature is viewed by all sensors. However, this FoV overlap might be very small or not even exist (e.g., to only focus on a dedicated area of interest), making these methods less practical. (2) computation time demands. For general ICP-based LiDAR extrinsic calibration approaches [4, 10], the extrinsic is optimized by aligning the point cloud from all LiDARs and maximizing the point cloud's consistency. The increase in the number of LiDARs implies that the feature point correspondence searching will be more time-consuming. This is due to the reason that each feature point needs to search for and match with nearby feature points using a k-d tree which contains the whole point cloud. In the LiDAR-camera extrinsic calibration, a larger amount of LiDAR points will also lead to more computation time in the LiDAR feature extraction.

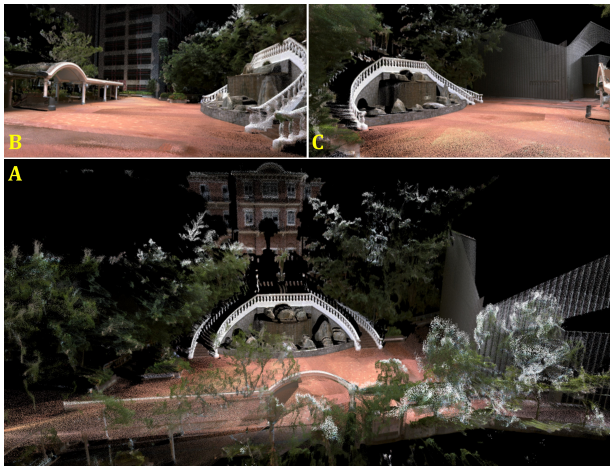


Fig. 1: A) The dense colorized point cloud with the LiDAR poses and extrinsic parameters optimized by our proposed method. The views from other perspectives are exhibited in B) left side and C) right side. Our experiment video is available at https://youtu.be/PaiYgAXl9iY.

To address the above challenges, we propose a fast and targetless extrinsic calibration method for multiple LiDARs and cameras. To create co-visible features among all sensors, we introduce motions to the sensor platform such that each sensor will scan the same area (hence features) at different times. We first calibrate the extrinsic among LiDARs by registering their point cloud using an efficient bundle adjustment (BA) method. To produce multi-view correspondence, we implement the adaptive voxelization to dynamically segment the point cloud into multiple voxels with non-identical sizes. This process greatly reduces the time consumption during the feature correspondence matching as only one plane feature exists in each voxel. We then calibrate the extrinsic between the cameras and LiDARs by matching the co-visible features between the images and the above-reconstructed point cloud. To further accelerate the feature correspondence matching, we inherit the above adaptive voxel map to extract LiDAR edge features. Moreover, we utilize depth-continuous edges from the point cloud to avoid foreground inflation and bleeding points issues. In summary, our contributions are listed as follows:

- We formulate the multi-LiDAR extrinsic calibration into a BA problem and derive the Jacobian and Hessian matrix of the cost function w.r.t. the LiDAR extrinsic to accelerate the optimization.

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

- We implement the adaptive voxelization to accelerate the process of feature correspondence matching and LiDAR depth-continuous edge extraction.
- We propose a fast, reliable, and targetless extrinsic calibration method for multiple LiDARs and cameras. Our method could handle the configurations when there is little or even no FoV overlap among the sensors. The precision and robustness of our proposed method are comparable to target-based methods and have been validated in outdoor test scenes.
- We open source our implementation on GitHub[1] to benefit the community.

II. Related Works

A. LiDAR-LiDAR Extrinsic Calibration

The extrinsic calibration methods between multiple LiDARs could be divided into two categories, motion-based and motionless approaches. In motion-based approaches, the sensor suite is usually required to move along such a trajectory that the onboard sensors could percept the same region of interests [4, 5, 11, 12]. In this manner, more constraints will be considered in the optimization problem, including those between the base and auxiliary LiDARs and those between different poses of the base LiADR during the motion. In [13]–[15], authors also introduce external inertial navigation sensors to ease the motion estimation of LiDARs. Then the extrinsic parameter could be calibrated by optimizing the consistency of the point cloud map with ICP-based [4, 10] or entropy-based [14] indicators. The issue within these methods is that they generally consider the correlation between each pair of features, which will be computationally expensive when the number of LiDAR increases. In [5], authors have maintained a state vector containing the extrinsic parameter of each LiDAR w.r.t. the robot center and updated it with EKF whenever a new measurement is available. This approach relies highly on the accuracy of the LiDAR odometry result that its calibration precision might be unreliable.

Motionless methods have been discussed in [6, 7] where authors attach the retro-reflective tapes to the surface of calibration targets to create and facilitate the feature extraction among multiple LiDARs. These methods require prior preparation work and FoV overlap between LiDARs, which have limited their works from wide implementation.

B. LiDAR-Camera Extrinsic Calibration

The extrinsic calibration between LiDAR and camera could be mainly divided into target-based and targetless methods. In target-based approaches, the geometric features, e.g., edges and surfaces, are extracted from natural geometric solids [16]–[18] or chessboard [19, 20] using intensity and color information. These features are matched either automatically or manually and are solved with general non-linear optimization tools. Since extra calibration targets and manual work are needed, these methods are less practical compared with targetless solutions.

[1]https://github.com/hku-mars/mlcc

The targetless methods could be further divided into motion-based and motionless approaches. In motion-based methods, the extrinsic parameter is usually initialized by the motion information and refined by the appearance information. In [21], authors reconstruct the point cloud from images using the structure from motion (SfM) to determine the initial extrinsic parameter and refine it by back-projecting LiDAR points onto the image plane. In [12], authors initialize the extrinsic parameter by Hand-eye calibration [22] and optimize it by minimizing the re-projection error between each image and adjacent two LiDAR scans. Though the introduction of motion addresses extra constraints between sensors, these methods require the sensor suite to move along a sufficiently excited trajectory. In motionless approaches, only the edge features that co-exist in both sensors' FoV are extracted and matched. Then the extrinsic parameter is optimized by minimizing the re-projected edge-to-edge distances [8, 23]–[25] or by maximizing the mutual information between the back-projected LiDAR points and the images [9].

Our proposed work is targetless and requires no FoV overlap between any sensor pairs to be calibrated. We create co-visible features by moving the sensor suite to multiple poses to eliminate the requirement of FoV overlap. Unlike [4, 12] which also optimize the LiDAR poses and extrinsic by minimizing the summed point-to-plane distances, our work directly operates on the raw point cloud without feature extraction and is more reliable in terms of both time consumption and precision. Compared with [9, 23] which also optimizes the LiDAR-camera extrinsic by minimizing the re-projection errors, our work could also handle the configuration when the LiDAR and camera have no common FoV and is more time-consuming.

III. Methodology

A. Overview

Let ${}^B_A\mathbf{T} = ({}^B_A\mathbf{R}, {}^B_A\mathbf{t}) \in SE(3)$ represent the rigid transformation from frame A to frame B, where ${}^B_A\mathbf{R} \in SO(3)$ and ${}^B_A\mathbf{t} \in \mathbb{R}^3$ are the rotation and translation. We denote $\mathcal{L} = \{L_0, L_1, \cdots, L_{n-1}\}$ the set of n LiDARs, where L_0 represents the base LiDAR for reference, $\mathcal{C} = \{C_0, C_1, \cdots, C_h\}$ the set of h cameras, $\mathcal{E}_L = \{{}^{L_0}_{L_1}\mathbf{T}, {}^{L_0}_{L_2}\mathbf{T}, \cdots, {}^{L_0}_{L_{n-1}}\mathbf{T}\}$ the set of LiDAR extrinsic parameters and $\mathcal{E}_C = \{{}^{L_0}_{C_0}\mathbf{T}, {}^{L_0}_{C_1}\mathbf{T}, \cdots, {}^{L_0}_{C_h}\mathbf{T}\}$ the set of LiDAR-camera extrinsic parameters. To create co-visible features between multiple LiDARs and cameras that may share no FoV overlap, we rotate the robot platform to m poses such that the same region of interest is scanned by all sensors (see Fig. 2). Denote $\mathcal{T} = \{t_0, t_1, \cdots, t_{m-1}\}$ the time for each of the m poses and the pose of the base LiDAR at the initial time as the global frame, i.e., ${}^G_{L_0}\mathbf{T}_{t_0} = \mathbf{I}_{4\times4}$. Denote $\mathcal{S} = \{{}^G_{L_0}\mathbf{T}_{t_1}, {}^G_{L_0}\mathbf{T}_{t_2}, \cdots, {}^G_{L_0}\mathbf{T}_{t_{m-1}}\}$ the set of the base LiDAR poses in global frame. The point cloud patch scanned by LiDAR $L_i \in \mathcal{L}$ at time $t_j \in \mathcal{T}$ is denoted by $\mathcal{P}_{L_i,t_j}$, which is in L_i's local frame. This point cloud patch could be transformed to global frame by

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

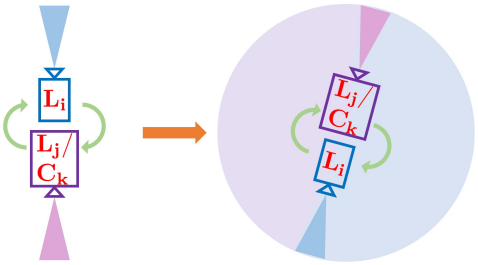


Fig. 2: FoV overlap created by rotation between two opposite pointing sensors. The original setup of two sensors L_i and L_j/C_k share no FoV overlap. With the introduction of rotational motion, the same region is scanned by all sensors across different times.

In our proposed approach of multi-sensor calibration, we sequentially calibrate the $\mathcal{E}_L$ and $\mathcal{E}_C$. In the first step, we simultaneously estimate the LiDAR extrinsic $\mathcal{E}_L$ and the base lidar pose trajectory $\mathcal{S}$ based on an efficient multi-view registration (see Sec. III-C). In the second step, we calibrate the $\mathcal{E}_C$ by matching the depth-continuous edges extracted from images and the above-reconstructed point cloud (see Sec. III-D). Lying in the center of both LiDAR and camera extrinsic calibration is an adaptive map, which finds correspondence among LiDAR and camera measurements efficiently (Sec. III-B).

B. Adaptive Voxelization

To find the correspondences among different LiDAR scans, we assume the initial base LiDAR trajectory $\mathcal{S}$, LiDAR extrinsic $\mathcal{E}_L$, and camera extrinsic $\mathcal{E}_C$ are available. The initial base LiDAR trajectory $\mathcal{S}$ could be obtained by an online LiDAR SLAM (e.g., [2]) and the initial extrinsic could be obtained from the CAD design or a rough Hand-eye calibration [22]. Our previous work [4] extracts edge and plane feature points from each LiDAR scan and matches them to the nearby edge and plane points in the map by a k-nearest neighbor search (kNN). This would repeatedly build a k-d tree of the global map at each iteration. In this letter, we use a more efficient voxel map proposed in [3] to create correspondences among all LiDAR scans.

The voxel map is built by cutting the point cloud (registered using the current $\mathcal{S}$ and $\mathcal{E}_\mathcal{L}$) into small voxels such that all points in a voxel roughly lie on a plane (with some adjustable tolerance). The main problem of the fixed-resolution voxel map is that if the resolution is high, the segmentation would be too time-consuming, while if the resolution is too low, multiple small planes in the environments falling into the same voxel would not be segmented. To best adapt to the environment, we implement an adaptive voxelization process. More specifically, the entire map is first cut into voxels with a pre-set size (usually large, e.g., 4m). Then for each voxel, if the contained points from all LiDAR scans roughly form a plane (by checking the ratio between eigenvalues), it is treated as a planar voxel; otherwise, they will be divided into eight octants, where each will be examined again until the contained points roughly form a plane or the voxel size reaches the pre-set minimum lower bound. Moreover, the adaptive voxelization is performed directly on the LiDAR raw points, so no feature points extraction is needed as in [4].

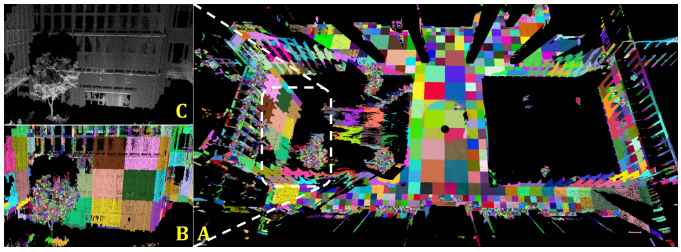


Fig. 3: A) LiDAR point cloud segmented with the adaptive voxelization. Points within the same voxel are colored identically. The detailed adaptive voxelization of points in the dashed white rectangle could be viewed in B) colored points and C) original points. The default size for the initial voxelization is 4m, and the minimum voxel size is 0.25m.

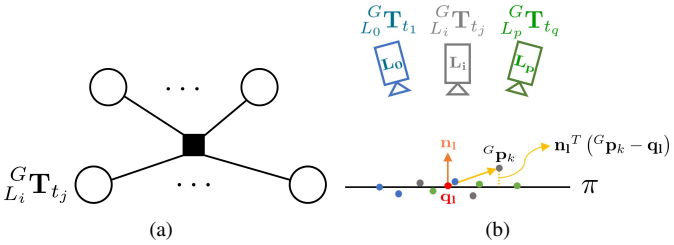


Fig. 4: (a) The l-th factor item relating to $\mathcal{S}$ and $\mathcal{E}_L$ with $L_i \in \mathcal{L}$ and $t_j \in \mathcal{T}$. (b) The distance from the point ${}^G\mathbf{p}_k$ to the plane $\boldsymbol{\pi}$.

Fig. 3 shows a typical result of the adaptive voxelization process in a complicated campus environment. As can be seen, this process is able to segment planes of different sizes, including large planes on the ground, medium planes on the building walls, and tiny planes on tree crowns.

C. Multi-LiDAR Extrinsic Calibration

With the adaptive voxelization, we can obtain a set of voxels of different sizes, and each voxel contains points that are roughly on a plane and creates a planar constraint for all LiDAR poses that have points in this voxel. More specifically, considering the l-th voxel consisting of a group of points $\mathcal{P}_l = \{{}^G\mathbf{p}_{L_i,t_j}\}$ scanned by $L_i \in \mathcal{L}$ at times $t_j \in \mathcal{T}$. We define a point cloud consistency indicator $c_l\left({}^G_{L_i}\mathbf{T}_{t_j}\right)$ which forms a factor on $\mathcal{S}$ and $\mathcal{E}_L$ as shown in Fig. 4(a). Then, the base LiDAR trajectory and extrinsic are estimated by optimizing the factor graph. A natural choice for the consistency indicator $c_l(\cdot)$ would be the summed Euclidean distance between each ${}^G\mathbf{p}_{L_i,t_j}$ to the plane to be estimated (see Fig. 4(b)). Taking account of all such indicators within the voxel map, we could formulate the problem as

$$\arg\min_{\mathcal{S},\mathcal{E}_L,\mathbf{n}_l,\mathbf{q}_l} \sum_l \underbrace{\left(\frac{1}{N_l}\sum_{k=1}^{N_l}\left(\mathbf{n}_l^T\left({}^G\mathbf{p}_k - \mathbf{q}_l\right)\right)^2\right)}_{l\text{-th factor}}, \tag{2}$$

where ${}^G\mathbf{p}_k \in \mathcal{P}_l$, N_l is the total number of points in $\mathcal{P}_l$, $\mathbf{n}_l$ is the normal vector of the plane and $\mathbf{q}_l$ is a point on this plane. The optimization dimension in (2) is too high due to the dependence on the planar parameters $\boldsymbol{\pi} = (\mathbf{n}_l, \mathbf{q}_l)$.

Fortunately, since one plane parameter is independent from another, we can optimize over $(\mathbf{n}_l, \mathbf{q}_l)$ first, i.e.,

$$\arg\min_{\mathcal{S},\mathcal{E}_L} \sum_l \left(\min_{\mathbf{n}_l,\mathbf{q}_l} \frac{1}{N_l} \sum_{k=1}^{N_l} \left(\mathbf{n}_l^T \left({}^G\mathbf{p}_k - \mathbf{q}_l \right) \right)^2 \right). \quad (3)$$

The inner optimization over $(\mathbf{n}_l, \mathbf{q}_l)$ in (3) could be further performed on $\mathbf{q}_l$ first and on $\mathbf{n}_l$ then, i.e.,

$$\arg\min_{\mathbf{n}_l} \left(\min_{\mathbf{q}_l} \frac{1}{N_l} \sum_{k=1}^{N_l} \left(\mathbf{n}_l^T \left({}^G\mathbf{p}_k - \mathbf{q}_l \right) \right)^2 \right). \quad (4)$$

As can be seen, the cost function in (4) is quadratic w.r.t. $\mathbf{q}_l$. Hence the inner optimization can be solved analytically by setting the derivatives to zeros, i.e.,

$$\mathbf{n}_l \mathbf{n}_l^T \left(\frac{1}{N_l} \sum_{k=1}^{N_l} \left({}^G\mathbf{p}_k - \mathbf{q}_l \right) \right) = \mathbf{0}. \quad (5)$$

It is seen that the solution to (5) is not unique as long as $\sum_{k=1}^{N_l} \left({}^G\mathbf{p}_k - \mathbf{q}_l \right)$ is perpendicular to $\mathbf{n}_l$, which allows $\mathbf{q}_l$ to move freely along any direction perpendicular to $\mathbf{n}_l$. Since this free movement of $\mathbf{q}_l$ does not change the plane parameterized by it, nor affect the cost function in (4), any solution of $\mathbf{q}_l$ satisfying (5) would be an optimal solution to the inner optimization problem of (4). One such solution could be

$$\mathbf{q}_l^* = \frac{1}{N_l} \sum_{k=1}^{N_l} {}^G\mathbf{p}_k. \quad (6)$$

Substituting the optimal solution of $\mathbf{q}_l$ in (6) back to (4) leads to

$$\arg\min_{\|\mathbf{n}_l\|=1} \mathbf{n}_l^T \underbrace{\left(\frac{1}{N_l} \sum_{k=1}^{N_l} {}^G\mathbf{p}_k \, {}^G\mathbf{p}_k^T - \mathbf{q}_l^* \mathbf{q}_l^{*T} \right)}_{\mathbf{A}_l} \mathbf{n}_l. \quad (7)$$

Again, this optimization problem has the well-known analytical optimal solution $\mathbf{n}_l^*$, which is the eigenvector corresponding to the smallest eigenvalue λ_3 of the matrix $\mathbf{A}_l$. As a result, substituting the optimal $\mathbf{n}_l^*$ back to (3) leads to

$$\mathcal{S}^*, \mathcal{E}_L^* = \arg\min_{\mathcal{S},\mathcal{E}_L} \sum_l \lambda_3 \left(\mathbf{A}_l \right). \quad (8)$$

As can be seen, the optimization variables $(\mathbf{n}_l, \mathbf{q}_l)$ are analytically solved before the optimization, which significantly reduces the optimization dimension. The resultant optimization in (8) is over the LiDAR pose ${}^G_{L_i}\mathbf{T}_{t_j}$ (hence the base LiDAR trajectory $\mathcal{S}$ and extrinsic $\mathcal{E}_L$) only. To see this, we note that $\mathbf{A}_l$ depends on ${}^G\mathbf{p}_k$ (directly or via $\mathbf{q}_l^*$ in (6)), which is observed locally by pose ${}^G_{L_i}\mathbf{T}_{t_j}$.

The optimization in (8) is nonlinear and solved iteratively. In each iteration, the cost function is approximated to the second order. More specifically, we view λ_3 as a function of all the contained points ${}^G\mathbf{p}$ which is the column vector containing each ${}^G\mathbf{p}_k \in \mathcal{P}_l$:

$${}^G\mathbf{p} = [{}^G\mathbf{p}_1^T \, {}^G\mathbf{p}_2^T \cdots {}^G\mathbf{p}_{N_l}^T]^T \in \mathbb{R}^{3N_l}.$$

The $\lambda_3({}^G\mathbf{p})$ in (8) could be approximated by

$$\lambda_3 \left({}^G\mathbf{p} + \delta^G\mathbf{p} \right) \approx \lambda_3 \left({}^G\mathbf{p} \right) + \mathbf{J} \cdot \delta^G\mathbf{p} + \frac{1}{2} \delta^G\mathbf{p}^T \cdot \mathbf{H} \cdot \delta^G\mathbf{p}, \quad (9)$$

where $\mathbf{J}$ and $\mathbf{H}$ are the first and second derivatives of $\lambda_3({}^G\mathbf{p})$ w.r.t. ${}^G\mathbf{p}$. The detailed derivation of $\mathbf{J}$ and $\mathbf{H}$ could be found in [3] and is omitted here due to limited space.

Suppose the k-th point ${}^G\mathbf{p}_k$ in ${}^G\mathbf{p}$ is scanned by LiDAR L_i at time t_j, then

$$\begin{aligned} {}^G\mathbf{p}_k &= {}^G_{L_i}\mathbf{T}_{t_j} \mathbf{p}_k = {}^G_{L_0}\mathbf{T}_{t_j} \cdot {}^{L_0}_{L_i}\mathbf{T} \cdot \mathbf{p}_k \\ &= {}^G_{L_0}\mathbf{R}_{t_j} \left({}^{L_0}_{L_i}\mathbf{R} \cdot \mathbf{p}_k + {}^{L_0}_{L_i}\mathbf{t} \right) + {}^G_{L_0}\mathbf{t}_{t_j}, \end{aligned} \quad (10)$$

which implies ${}^G\mathbf{p}_k$ is dependent on $\mathcal{S}$ and $\mathcal{E}_L$. To perturb ${}^G\mathbf{p}_k$, we perturb a pose $\mathbf{T}$ in its tangent plane $\delta\mathbf{T} = [\boldsymbol{\phi}^T \ \delta\mathbf{t}^T]^T \in \mathbb{R}^6$ with the $\boxplus$ as defined in [26], i.e.,

$$\begin{aligned} \mathbf{T} &= (\mathbf{R}, \mathbf{t}) \\ \mathbf{T} \boxplus \delta\mathbf{T} &= \left(\mathbf{R} \exp\left(\boldsymbol{\phi}^\wedge \right), \mathbf{t} + \delta\mathbf{t} \right). \end{aligned} \quad (11)$$

Based on the error parameterization in (11) for both ${}^G_{L_0}\mathbf{T}_{t_j}$ and extrinsic ${}^{L_0}_{L_i}\mathbf{T}$, the perturbed point location in (10) is

$$\begin{aligned} {}^G\mathbf{p}_k + \delta^G\mathbf{p}_k = {}^G_{L_0}\mathbf{R}_{t_j} \exp\left({}^G_{L_0}\boldsymbol{\phi}^\wedge_{t_j} \right) \Big({}^{L_0}_{L_i}\mathbf{R} \exp\left({}^{L_0}_{L_i}\boldsymbol{\phi}^\wedge \right) \mathbf{p}_k \\ + {}^{L_0}_{L_i}\mathbf{t} + \delta^{L_0}_{L_i}\mathbf{t} \Big) + {}^G_{L_0}\mathbf{t}_{t_j} + \delta^G_{L_0}\mathbf{t}_{t_j}. \end{aligned} \quad (12)$$

Then, subtracting (10) from (12), we obtain

$$\begin{aligned} \delta^G\mathbf{p}_k \approx {}^G_{L_0}\mathbf{R}_{t_j} \left({}^{L_0}_{L_i}\mathbf{R}\mathbf{p}_k + {}^{L_0}_{L_i}\mathbf{t} \right)^\wedge {}^G_{L_0}\boldsymbol{\phi}_{t_j} + \delta^G_{L_0}\mathbf{t}_{t_j} + \\ {}^G_{L_i}\mathbf{R}_{t_j} \left(\mathbf{p}_k \right)^\wedge {}^{L_0}_{L_i}\boldsymbol{\phi} + {}^G_{L_0}\mathbf{R}_{t_j} \delta^{L_0}_{L_i}\mathbf{t} \end{aligned} \quad (13)$$

and

$$\delta^G\mathbf{p} = \mathbf{D} \cdot \delta\mathbf{x}, \quad (14)$$

where

$$\delta\mathbf{x} = [\cdots {}^G_{L_0}\boldsymbol{\phi}^T_{t_j} \ \delta^G_{L_0}\mathbf{t}^T_{t_j} \cdots {}^{L_0}_{L_i}\boldsymbol{\phi}^T \ \delta^{L_0}_{L_i}\mathbf{t}^T \cdots]^T \in \mathbb{R}^{6(m+n-2)}$$

is a small perturbation of the entire optimization vector $\mathbf{x}$

$$\mathbf{x} = [\cdots {}^G_{L_0}\mathbf{R}_{t_j} \ \ {}^G_{L_0}\mathbf{t}_{t_j} \cdots {}^{L_0}_{L_i}\mathbf{R} \ \ {}^{L_0}_{L_i}\mathbf{t} \cdots],$$

and

$$\mathbf{D} = \begin{bmatrix} \vdots & \vdots \\ \cdots \ \mathbf{D}^{\mathcal{S}}_{k,p} & \cdots \ \mathbf{D}^{\mathcal{E}_L}_{k,q} \ \cdots \\ \vdots & \vdots \end{bmatrix} \in \mathbb{R}^{3N_l \times 6(m+n-2)}$$

$$\mathbf{D}^{\mathcal{S}}_{k,p} = \begin{cases} \Big[- {}^G_{L_0}\mathbf{R}_{t_j} \left({}^{L_0}_{L_i}\mathbf{R}\mathbf{p}_k + {}^{L_0}_{L_i}\mathbf{t} \right)^\wedge \mathbf{I} \Big], & \text{if } p = j \\ \mathbf{0}_{3\times 6}, & \text{else} \end{cases}$$

$$\mathbf{D}^{\mathcal{E}_L}_{k,q} = \begin{cases} \Big[- {}^G_{L_0}\mathbf{R}_{t_j} \, {}^{L_0}_{L_i}\mathbf{R} \left(\mathbf{p}_k \right)^\wedge \ \ {}^G_{L_0}\mathbf{R}_{t_j} \Big], & \text{if } q = i \\ \mathbf{0}_{3\times 6}, & \text{else.} \end{cases} \quad (15)$$

Substituting (14) to (9) leads to

$$\begin{aligned} \lambda_3(\mathbf{x} \boxplus \delta\mathbf{x}) &\approx \lambda_3(\mathbf{x}) + \mathbf{J}\mathbf{D}\delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T\mathbf{D}^T\mathbf{H}\mathbf{D}\delta\mathbf{x} \\ &= \lambda_3(\mathbf{x}) + \bar{\mathbf{J}}\delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T\bar{\mathbf{H}}\delta\mathbf{x}. \end{aligned} \quad (16)$$

Then the optimal $\mathbf{x}^*$ could be determined by iteratively solving the (17) with the LM method and updating the $\delta\mathbf{x}$ to $\mathbf{x}$.

$$\left(\bar{\mathbf{H}} + \mu\mathbf{I} \right) \delta\mathbf{x} = -\bar{\mathbf{J}}^T \quad (17)$$

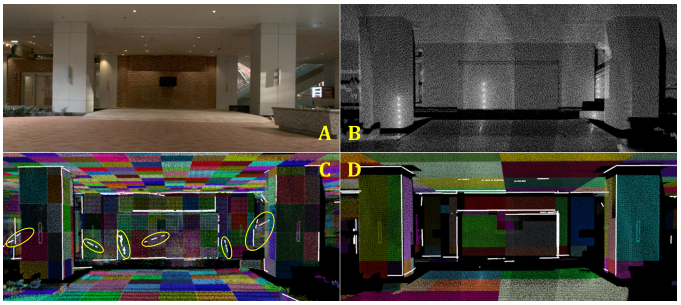


Fig. 5: Depth-continuous edge extraction comparison. A) real-world image. B) raw point cloud of this scene. C) edges extracted using method in [23] where the yellow circles indicate the false estimations. D) edges extracted with adaptive voxelization. The time consumption of edge extraction in this scene is 38s for [23] versus 5s of our proposed method.

D. LiDAR-Camera Extrinsic Calibration

With the calibrated $\mathcal{E}_L^*$ and $\mathcal{S}^*$, we could obtain a dense point cloud in the global frame. This global point cloud could then be used to find the optimal extrinsic $\mathcal{E}_C$ by matching edge features from the point cloud and the image. Two types of edges could be extracted from the point cloud and images. One is depth-discontinuous edges between foreground and background objects, and the other is the depth-continuous edge between two neighboring non-parallel planes. As explained in [23], depth-discontinuous edges suffer from foreground inflation and bleeding points phenomenon, we hence use depth-continuous edges to match point cloud and images.

In [23], the point cloud is segmented into uniform size voxels and the planes inside each voxel are estimated by the RANSAC algorithm. In contrast, our method uses the same adaptive voxel map obtained in Sec. III-B. Then for every two adjacent voxels, we calculate the angle between their containing planes. If this angle exceeds a threshold, the intersection line of these two planes is extracted as the depth-continuous edge. As shown in Fig. 5, our method could effectively remove the false estimations and saves computation time.

For image edge extraction, we use the Canny algorithm. To further accelerate the calibration process and avoid miss-matching, we conduct an FoV check that only the depth-continuous edges within the current camera's FoV are used. The correspondence between the LiDAR edge and camera edge is built by projecting the LiDAR edge onto the image plane with the current extrinsic estimate. Then we optimize the extrinsic parameters by minimizing the residuals of point-to-edge distances on the image plane similar to [23].

E. Calibration Pipeline

The workflow of our proposed multi-sensor calibration is illustrated in Fig. 6. At the beginning of the calibration, the base LiDAR's raw point cloud is processed by a LOAM [2] to obtain the initial base LiDAR trajectory $\mathcal{S}$. Then, the raw point cloud of all LiDARs are segmented by time into point cloud patches, i.e., $\mathcal{P}_{L_i,t_j}, L_i \in \mathcal{L}, t_j \in \mathcal{T}$ that is collected under the pose ${}^{G}_{L_i}\mathbf{T}_{t_j}$.

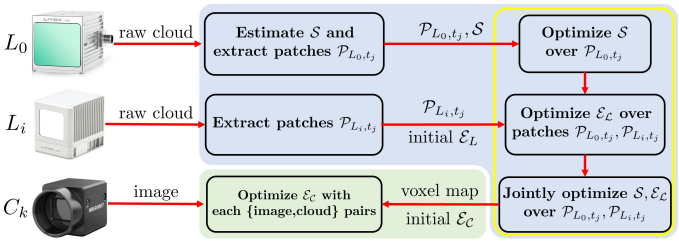


Fig. 6: The workflow of our proposed method: multi-LiDAR extrinsic calibration (light blue region) and LiDAR-camera extrinsic calibration (light green region). The adaptive voxelization takes effect in the steps surrounded by the yellow lines.

In multi-LiDAR extrinsic calibration, the base LiDAR poses $\mathcal{S}$ are first optimized using the base LiDAR's point cloud patches $\mathcal{P}_{L_0,t_j}$. Noticed that only $\mathcal{S}$ is involved and optimized in (3). Then the extrinsic $\mathcal{E}_L$ are calibrated by aligning the point cloud from the LiDAR to be calibrated with those from the base LiDAR. In this stage's problem formulation (3), $\mathcal{S}$ is fixed at the optimized values from the previous stage, and only $\mathcal{E}_L$ is optimized. Finally, both $\mathcal{S}$ and $\mathcal{E}_L$ are jointly optimized using the entire point cloud patches. In each iteration of the optimization (over $\mathcal{S}$, $\mathcal{E}_L$, or both), the adaptive voxelization (as described in Sec. III-B) is performed with the current value of $\mathcal{S}$ and $\mathcal{E}_L$.

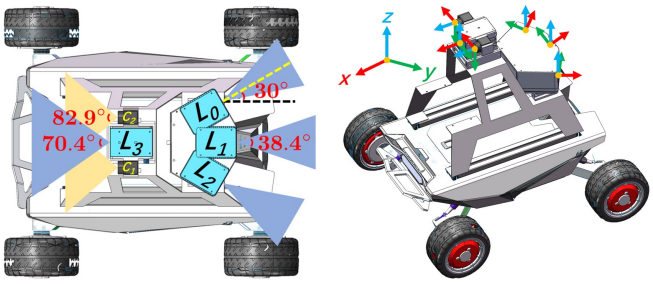
In multi-LiDAR-camera extrinsic calibration, the adaptive voxel map obtained with the $\mathcal{S}^*$ and $\mathcal{E}_L^*$ in the previous step is used to extract 3D depth-continuous edges (Sec. III-D). Then those 3D edges are back-projected onto each image using the extrinsic parameter $\mathcal{E}_C$ and are matched with 2D Canny edges extracted from the image. By minimizing the residuals defined by these two edges, we iteratively solve for the optimal $\mathcal{E}_C^*$ with the Ceres Solver[2].

IV. Experiments and Results

To test the proposed algorithm, we customized a remotely operated vehicle platform[3] with one Livox AVIA LiDAR[4] (with 70.4° FoV), one Livox MID-100 LiDAR[5] and two MV-CA013-21UC[6] cameras (with 82.9° FoV each), as illustrated in Fig. 7. The MID-100 LiDAR consists of three MID-40 LiDAR units (with 38.4° FoV each), of which the extrinsic parameters are calibrated by the manufacturer and could be used as the ground truth for the calibration evaluation. Note that the two types of LiDAR units (e.g., AIVIA and MID-40) have different scanning patterns, densities, and FoVs.

We have verified our proposed algorithm with the data collected in two random test scenes in our campus as shown in Fig. 8. Scene-1 is a square in front of the library with moving pedestrians and scene-2 is an area near a garden. In Sec. IV-A, the data collected in our previous work [4] have also been used for comparison with the previous method. All

[2]http://ceres-solver.org/
[3]https://www.agilex.ai/product/3?lang=en-us
[4]https://www.livoxtech.com/avia
[5]https://www.livoxtech.com/mid-40-and-mid-100
[6]https://www.rmaelectronics.com/hikrobot-mv-ca013-21uc/



(a) (b)

Fig. 7: Our customized multi-sensor vehicle platform. Left: the FoV coverage of each sensor with their FoV specs. Right: the orientation of each sensor is denoted in the right-handed coordinate system.

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

Fig. 8: Our experiment test scenes.

experiments are conducted on a high-performance PC with an i7-9700K processor and 32GB RAM.

For our proposed multi-LiDAR extrinsic calibration, we first conduct a standard Hand-eye calibration [22] with an '8'-figure path to initialize the extrinsic $\mathcal{E}_{\mathcal{L}}$. Then we rotate our multi-sensor platform by 360° and keep the robot platform still every few degrees, such that we could acquire dense enough point cloud from each LiDAR at each pose. Keeping the robot platform still during data collection also eliminates the problem caused by motion distortion and time synchronization. The timestamps $\mathcal{T}$ are manually selected that only the point cloud and image data when the robot platform is still, are selected.

A. Convergence and Computation Time Comparison

In this section, we demonstrate that the proposed algorithm converges faster than our previous work [4] in terms of both iteration times and computation time while remains accurate. We use the dataset collected in [4] on MID-100 and choose the middle MID-40 as the base LiDAR to calibrate the adjacent two LiDARs. We perform 10 independent trials that in each trial the initial extrinsic $\mathcal{E}_{\mathcal{L}}$ is initialized by randomly perturbing each Euler angle of ${}^{L_1}_{L_i}\mathbf{R}$ by $\theta_{roll,pitch,yaw} \in [-3°, 3°]$ and each axis's offset ${}^{L_1}_{L_i}\mathbf{t}$ by $t_{x,y,z} \in [-0.1m, 0.1m]$ from the manufacturer's calibrated values.

The extrinsic rotation and translation errors of both methods versus iteration time are plotted in Fig. 9 and the averaged time cost of each iteration of both methods is summarized in Table. I. It is shown in Fig. 9 that the proposed work makes both the extrinsic translation and rotation errors quickly converge to the appropriate values. This is due to the second-order optimization we used in Sec. III-C, where the Jacobian

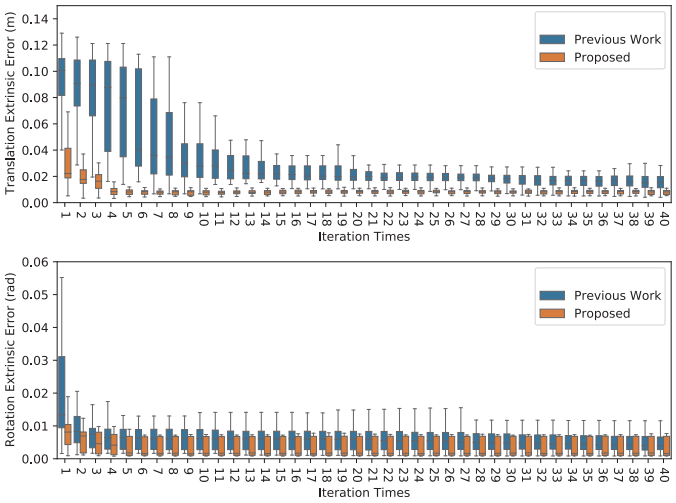


Fig. 9: Convergence comparison of the proposed method and previous work [4]. Each box-plot consists of 40 values from 10 trials, two test scenes and two LiDAR pairs, i.e., $\{L_1, L_0\}, \{L_1, L_2\}$. The mean and standard deviation of the initial extrinsic errors are 0.0929m and 0.0262m for translation and 0.0553rad and 0.0257rad for rotation, respectively.

and Hessian matrix with respect to the optimization variables ($\mathcal{S}$ and $\mathcal{E}_L$) are exactly derived. In contrast, in the previous work [4], only the Jacobian of the residual w.r.t. one LiDAR is considered, causing inaccurate Jacobian computation. The calibration results of both methods in the above 10 trials are plotted in Fig. 10 which indicates that the increase in speed of our proposed method does not result in the loss of accuracy. Considering the computation time cost and the convergence rate, our proposed algorithm could save the total calibration time to at least one-tenth of the previous work.

TABLE I: COMPUTATION TIME COMPARISON

	previous work [4]	proposed
time cost per iteration (s)	7.6372	**1.4516**

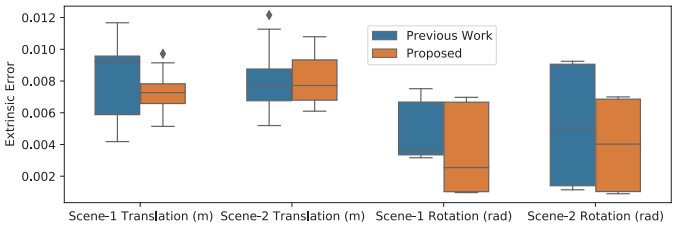


Fig. 10: Extrinsic calibration results of three MID-40 LiDARs. Each box plot consists of 20 values respectively from 10 trials and two pairs of LiDARs.

B. Multi-LiDAR Calibration

1) MID-100 LiDAR Self Calibration: In this section, we compare our algorithm with the motion-based method [12] using the MID-100 LiDAR and the data collected in both test scenes. The middle MID-40 is chosen as the base LiDAR to calibrate the extrinsic $\mathcal{E}_{\mathcal{L}}$ of other MID-40s, i.e., ${}^{L_1}_{L_0}\mathbf{T}$, ${}^{L_1}_{L_2}\mathbf{T}$. For

both methods, the extrinsic $\mathcal{E}_{\mathcal{L}}$ are initialized by the Hand-eye calibration and the results are summarized in Table. II. Since the MID-40 LiDAR is of small FoV and the vehicle's movements in both test scenes are limited to planar motions, the pure motion-based method is less comparable to our proposed method.

TABLE II: EXTRINSIC CALIBRATION RESULTS OF LIDARS INSIDE MID-100 IN TWO TEST SCENES

Methods	Rotation Error		Translation Error	
	mean	sd	mean	sd
motion based [12]	2.7223°	2.4137°	0.3955m	0.1267m
proposed	**0.2173°**	**0.1699°**	**0.0075 m**	**0.0016m**

2) AVIA and MID-100 LiDAR: In this section, we demonstrate that our method works well given two types of LiDARs with different FoVs and point cloud densities and we compare the results with those from motion-based method [12]. The AVIA is chosen as the base LiDAR to calibrate the extrinsic $\mathcal{E}_{\mathcal{L}}$ between AVIA and each MID-40s, i.e., ${}^{L_3}_{L_0}\mathbf{T}$, ${}^{L_3}_{L_1}\mathbf{T}$ and ${}^{L_3}_{L_2}\mathbf{T}$. Then we calculate the ${}^{L_1}_{L_0}\mathbf{T}$, ${}^{L_1}_{L_2}\mathbf{T}$ from the above results and compare them with the known values obtained from manufacturer. For both methods, the extrinsic $\mathcal{E}_{\mathcal{L}}$ are initialized by Hand-eye calibration and the results from both test scenes are summarized in Table. III. It is shown that the proposed method's performance is less affected by the distinct characteristics introduced from different types of LiDARs.

TABLE III: EXTRINSIC CALIBRATION RESULTS BETWEEN AVIA AND MID-100 IN TWO TEST SCENES

Methods	Rotation Error		Translation Error	
	mean	sd	mean	sd
motion based [12]	5.0876°	4.3721°	0.9945m	0.5701m
proposed	**0.2510°**	**0.2184°**	**0.0084m**	**0.0023m**

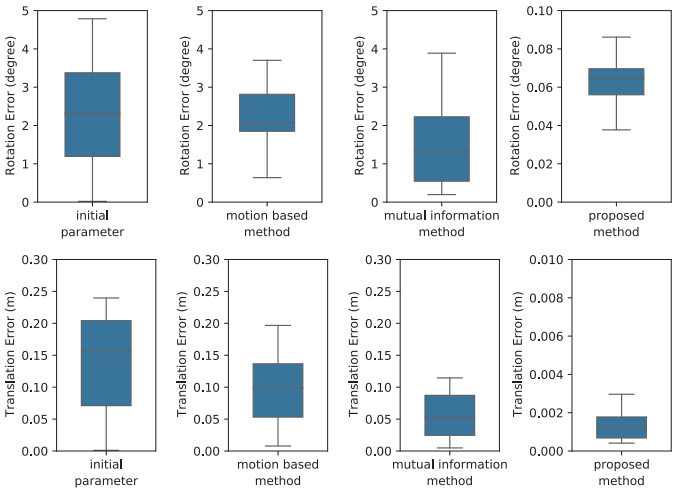


Fig. 11: Extrinsic calibration results of [9, 12] and the proposed method. Each box-plot illustrates the results of 20 trials using the data collected in scene-2. The average and standard deviation of the initial rotation error are 2.5390° and 1.4203°. The average and standard deviation of the initial translation error are 0.1389m and 0.0778m, respectively.

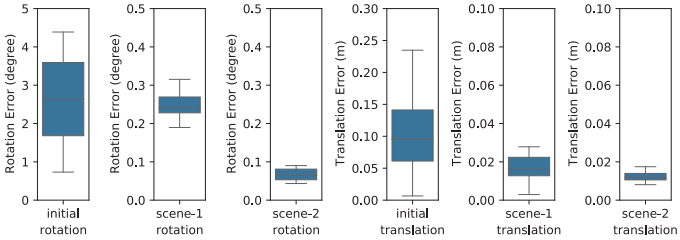


Fig. 12: Extrinsic calibration results of MID-100 and opposite pointing cameras in two test scenes. Each box-plot illustrates the results of 20 trials. The average and standard deviation of the initial rotation error are 2.5913° and 1.2771°. The average and standard deviation of the initial translation error are 0.1007m and 0.0588m, respectively.

C. Multiple LiDAR Camera Calibration

1) Among AVIA, MID-100 and Cameras: In this section, we compare our proposed LiDAR-camera extrinsic calibration method with the motion-based [12] and the mutual information based [9] methods. Both [9, 12] require the sensors to share the common FoV, utilize the intensity information of the LiDAR point cloud and match it with the edge features extracted from the image. Here, we select the AVIA as the base LiDAR. The initial extrinsic $\mathcal{E}_{\mathcal{C}}$ are calculated by adding disturbance to the values measured from the CAD model. We perform 20 independent trials with the data collected in scene-2, that in each trial we randomly perturb each Euler angle of ${}^{L_3}_{C_k}\mathbf{R}$ by $\theta_{\{roll,pitch,yaw\}} \in [-2^\circ, 2^\circ]$ and each axis's offset of ${}^{L_3}_{C_k}\mathbf{t}$ by $t_{\{x,y,z\}} \in [-0.1m, 0.1m]$ from the CAD model's measurements. We calibrate the extrinsic of each camera individually (i.e., ${}^{L_3}_{C_1}\mathbf{T}$ and ${}^{L_3}_{C_2}\mathbf{T}$), then we calculate the ${}^{C_1}_{C_2}\mathbf{T}$ and compare it with that obtained by the standard chessboard method serving the ground-truth. The calibration results are illustrated in Fig. 11. It is shown that our proposed method outperforms [9, 12] both quantitatively and qualitatively.

2) MID-100 and Cameras: In this section, we demonstrate that the proposed method could calibrate the extrinsic $\mathcal{E}_{\mathcal{C}}$ between LiDAR and cameras without FoV overlap. We choose the middle MID-40 of the MID-100 as the base LiDAR and calibrate the extrinsic of each LiDAR-camera pairs (i.e., ${}^{L_1}_{C_1}\mathbf{T}$, ${}^{L_1}_{C_2}\mathbf{T}$). The initial extrinsic $\mathcal{E}_{\mathcal{C}}$ are calculated by adding disturbance to the values measured from the CAD model. We perform 20 independent trials with the data collected in both scenes, that in each trial we randomly perturb each Euler angle of ${}^{L_1}_{C_k}\mathbf{R}$ by $\theta_{\{roll,pitch,yaw\}} \in [-2^\circ, 2^\circ]$ and each axis's offset of ${}^{L_1}_{C_k}\mathbf{t}$ by $t_{\{x,y,z\}} \in [-0.1m, 0.1m]$ from the CAD's measurements. Then we calculate the ${}^{C_1}_{C_2}\mathbf{T}$ and compare it with that obtained by the standard chessboard method. The calibration results and the corresponding colorized point cloud are illustrated in Fig. 12 and Fig. 13.

It is seen that the general extrinsic calibration performance between MID-40 and cameras is less competitive than that between AVIA and cameras. This is due to the fact that AVIA has larger FoV coverage (70.4° versus 38.4°) and thus point cloud density (6 laser beams versus 1 laser beam) than MID-40, which will provide more edge correspondences in all

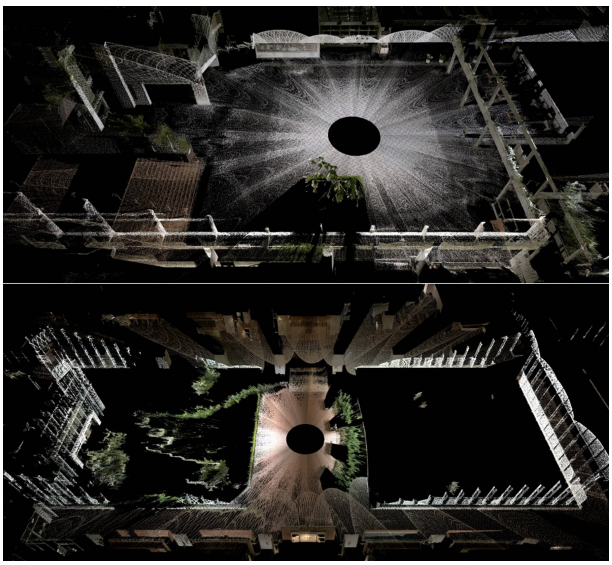


Fig. 13: Colorized point cloud using MID-100 LiDAR and opposite pointing cameras. The left camera's images color the point clouds in both test scenes. The brightness of the window is due to the reflection of the sunlight. Top: scene-1; Bottom: scene-2.

directions. The performance of MID-40 and cameras extrinsic calibration in scene-2 is also more robust and acceptable than scene-1. This is probably due to the reason that the extracted LiDAR edges mismatch with and trapped into the image edges largely existed on the ground of scene-1.

V. Conclusion

In this letter, we propose a fast and targetless extrinsic calibration method for multiple LiDARs and cameras. We analytically derive the derivatives of the cost function w.r.t. the LiDAR extrinsic, and implement adaptive voxelization which has greatly shortened the total calibration time. Experiment results under multiple LiDAR-camera configurations in outdoor test scenes demonstrate the robustness and reliability of our proposed method, even when no FoV overlap exists between the sensor pairs.

Acknowledgment

The authors thank Livox Technology and AgileX Robotics for their product support.

References

[1] F. Kong, W. Xu, Y. Cai, and F. Zhang. Avoiding dynamic small obstacles with onboard sensing and computation on aerial robots. *IEEE Robotics and Automation Letters*, 6(4):7869–7876, 2021.

[2] 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.

[3] Z. Liu and F. Zhang. Balm: Bundle adjustment for lidar mapping. *IEEE Robotics and Automation Letters*, 6(2):3184–3191, 2021.

[4] X. Liu and F. Zhang. Extrinsic calibration of multiple lidars of small fov in targetless environments. *IEEE Robotics and Automation Letters*, 6(2):2036–2043, 2021.

[5] 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.

[6] 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*, pages 279–284, 2010.

[7] 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)*, pages 1–6, 2019.

[8] J. Levinson and S. Thrun. Automatic online calibration of cameras and lasers. In *Robotics: Science and Systems*, volume 2, page 7. Citeseer, 2013.

[9] G. Pandey, J. R. McBride, S. Savarese, and R. Eustice. Automatic extrinsic calibration of vision and lidar by maximizing mutual information. *J. Field Robotics*, 32:696–722, 2015.

[10] J. Jiao, H. Ye, Y. Zhu, and M. Liu. Robust odometry and mapping for multi-lidar systems with online extrinsic calibration. *IEEE Transactions on Robotics*, pages 1–10, 2021.

[11] L. Heng. Automatic targetless extrinsic calibration of multiple 3d lidars and radars. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10669–10675, 2020.

[12] 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.

[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. In *2012 IEEE International Conference on Robotics and Automation*, pages 3096–3102, 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] J. Kummerle and T. Kuhner. Unified intrinsic and extrinsic camera and lidar calibration under uncertainties. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[17] X. Gong, Y. Lin, and J. Liu. 3d lidar-camera extrinsic calibration using an arbitrary trihedron. *Sensors*, 13(2):1902–1918, 2013.

[18] Y. Park, S. Yun, C. Won, K. Cho, K. Um, and S. Sim. Calibration between color camera and 3d lidar instruments with a polygonal planar board. *Sensors*, 14(3):5333–5353, 2014.

[19] G. Koo, J. Kang, B. Jang, and N. Doh. Analytic plane covariances construction for precise planarity-based extrinsic calibration of camera and lidar. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[20] L. Zhou, Z. Li, and M. Kaess. Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[21] B. Nagy, L. Kovács, and C. Benedek. Online targetless end-to-end camera-lidar self-calibration. In *2019 16th International Conference on Machine Vision Applications (MVA)*, pages 1–6, 2019.

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

[23] C. Yuan, X. Liu, X. Hong, and F. Zhang. Pixel-level extrinsic self calibration of high resolution lidar and camera in targetless environments. *IEEE Robotics and Automation Letters*, 6(4):7517–7524, 2021.

[24] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong. Camvox: A low-cost and accurate lidar-assisted visual slam system. *arXiv preprint arXiv:2011.11357*, 2020.

[25] D. Scaramuzza, A. Harati, and R. Siegwart. Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4164–4169. IEEE, 2007.

[26] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder. Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds. *Information Fusion*, 14(1):57–77, 2013.