Pixel-level Extrinsic Self Calibration of High Resolution LiDAR and Camera in Targetless Environments

Chongjian Yuan, Xiyuan Liu, Xiaoping Hong, and Fu Zhang

***Abstract*— In this letter, we present a novel method for automatic extrinsic calibration of high-resolution LiDARs and RGB cameras in targetless environments. Our approach does not require checkerboards but can achieve pixel-level accuracy by aligning natural edge features in the two sensors. On the theory level, we analyze the constraints imposed by edge features and the sensitivity of calibration accuracy with respect to edge distribution in the scene. On the implementation level, we carefully investigate the physical measuring principles of LiDARs and propose an efficient and accurate LiDAR edge extraction method based on point cloud voxel cutting and plane fitting. Due to the edges' richness in natural scenes, we have carried out experiments in many indoor and outdoor scenes. The results show that this method has high robustness, accuracy, and consistency. It can promote the research and application of the fusion between LiDAR and camera. We have open sourced our code on GitHub[1] to benefit the community.**

I. Introduction

Light detection and ranging (LiDAR) and camera sensors are commonly combined in developing autonomous driving vehicles. LiDAR sensor, owing to its direct 3D measurement capability, has been extensively applied to obstacle detection [1], tracking [2], and mapping [3] applications. The integrated onboard camera could also provide rich color information and facilitate the various LiDAR applications. With the recent rapid growing resolutions of LiDAR sensors, the demand for accurate extrinsic parameters becomes essential, especially for applications such as dense point cloud mapping, colorization, and accurate and automated 3D surveying. In this letter, our work deals with the accurate extrinsic calibration of high-resolution LiDAR and camera sensors.

Current extrinsic calibration methods rely heavily on external targets, such as checkerboard [4]–[6] or specific image pattern [7]. By detecting, extracting, and matching feature points from both image and point cloud, the original problem is transformed into and solved with least-square equations. Due to its repetitive scanning pattern and the inevitable vibration of mechanical spinning LiDAR, e.g., Velodyne[2], the reflected point cloud tends to be sparse and of large noise. This characteristic may mislead the cost function to the unstable results. Solid-state LiDAR, e.g., Livox [8], could compensate for this drawback with its dense point cloud. However, since these calibration targets are typically placed close to the sensor suite, the extrinsic error might be enlarged in the long-range scenario, e.g., large-scale point cloud colorization. In addition, it is not always practical to calibrate the sensors with targets being prepared at the start of each mission.

Fig. 1: Point cloud of three aligned LiDAR scans colored using the proposed method. The accompanying experiment video has been uploaded to https://youtu.be/e6Vkkasc4JI.

To address the above challenges, in this letter, we propose an automatic pixel-level extrinsic calibration method in targetless environments. This system operates by extracting natural edge features from the image and point cloud and minimizing the reprojection error. The proposed method does not rely on external targets, e.g., checkerboard, and is capable of functioning in both indoor and outdoor scenes. Such a simple and convenient calibration allows one to calibrate the extrinsic parameters before or in the middle of each data collection or detect any misalignment of the sensors during their online operation that is usually not feasible with the target-based methods. Specifically, our contributions are as follows:

- We carefully study the underlying LiDAR measuring principle, which reveals that the commonly used depth-discontinuous edge features are not accurate nor reliable for calibration. We propose a novel and reliable depth-continuous edge extraction algorithm that leads to more accurate calibration parameters.
- We evaluate the robustness, consistency, and accuracy of our methods and implementation in various indoor and outdoor environments and compare our methods with other state-of-the-art. Results show that our method is robust to initial conditions, consistent to calibration scenes, and achieves pixel-level calibration accuracy in natural environments. Our method has an accuracy that is

C. Yuan, X. Liu 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. {ycj1,xliuaa}@connect.hku.hk, fuzhang@hku.hk

X. Hong is with the School of System Design and Intelligent Manufacturing, Southern University of Science and Technology, Shenzhen, People's Republic of China. hongxp@sustech.edu.cn

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

[2]https://velodynelidar.com/

on par to (and sometimes even better than) target-based methods and is applicable to both emerging solid-state and conventional spinning LiDARs.
- Based on the analysis, we develop a practical calibration software and open source it on GitHub to benefit the community.

II. Related Works

Extrinsic calibration is a well-studied problem in robotics and is mainly divided into two categories: target-based and targetless. The primary distinction between them is how they define and extract features from both sensors. Geometric solids [9]–[11] and checkerboards [4]–[6] have been widely applied in target-based methods, due to its explicit constraints on plane normals and simplicity in problem formulation. As they require extra preparation, they are not practical, especially when they need to operate in a dynamically changing environment.

Targetless methods do not detect explicit geometric shapes from known targets. Instead, they use the more general plane and edge features that existed in nature. In [12], the LiDAR points are first projected onto the image plane and colored by depth and reflectivity values. Then 2D edges are extracted from this colormap and matched with those obtained from the image. Similarly, authors in [13] optimize the extrinsic calibration by maximizing the mutual information between the colormap and the image. In [14, 15], both authors detect and extract 3D edges from the point cloud by laser beam depth discontinuity. Then the 3D edges are back-projected onto the 2D image plane to calculate the residuals. The accuracy of edge estimation limits this method as the laser points do not strictly fall on the depth discontinuity margin. Motion-based methods have also been introduced in [16, 17] that the extrinsic is estimated from sensors' motion and refined by appearance information. This motion-based calibration typically requires the sensor to move along a sufficient excited trajectory [17].

Our proposed method is a targetless method. Compared to [12, 13], we directly extract 3D edge features in the point cloud, which suffer from no occlusion problem. Compared to [14, 15], we use depth-continuous edges, which proved to be more accurate and reliable. Our method works for a single pair LiDAR scan and achieves calibration accuracy comparable to target-based methods [4, 6].

III. Methodology

A. Overview

Fig. 2 defines the coordinate frames involved in this paper: the LiDAR frame L, the camera frame C, and the 2D coordinate frame in the image plane. Denote ${}^C_L\mathbf{T} = ({}^C_L\mathbf{R}, {}^C_L\mathbf{t}) \in SE(3)$ the extrinsic between LiDAR and camera to be calibrated. Due to the wide availability of edge features in natural indoor and outdoor scenes, our method aligns these edge features observed by both LiDAR and camera sensors.

Fig. 2 further illustrates the number of constraints imposed by a single edge to the extrinsic. As can be seen, the following degree of freedom (DoF) of the LiDAR pose relative to the camera cannot be distinguished: (1) translation along the edge

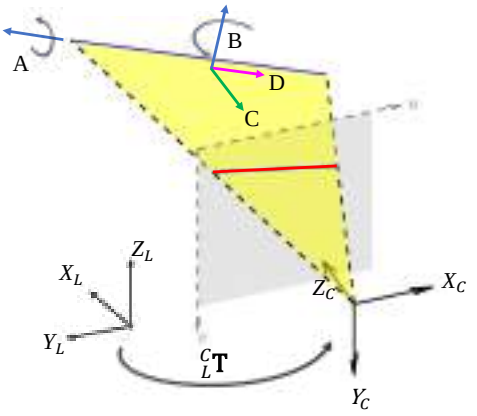


Fig. 2: Constraints imposed by an edge feature. The blue line represents the 3D edge, its projection to the image plane (the gray plane) produces the camera measurement (the red line). The edge after a translation along the axis C or axis D, or a rotation about the axis A (i.e., the edge itself) or axis B (except when the edge passes through the camera origin), remains on the same yellow plane and hence has the same projection on the image plane. That means, these four pose transformations on the edge (or equivalently on ${}^C_L\mathbf{T}$) are not distinguishable.

(the red arrow D, Fig. 2), (2) translation perpendicular to the edge (the green arrow C, Fig. 2), (3) rotation about the normal vector of the plane formed by the edge and the camera focal point (the blue arrow B, Fig. 2), and (4) rotation about the edge itself (the purple arrow A, Fig. 2). As a result, a single edge feature constitutes two effective constraints to the extrinsic ${}^C_L\mathbf{T}$. To obtain sufficient constraints to the extrinsic, we extract edge features of different orientations and locations, as detailed in the following section.

B. Edge Extraction and Matching

1) Edge Extraction: Some existing works project the point cloud to the image plane and extract features from the projected point cloud, such as edge extraction [12] and the mutual information correlation [13]. A major problem of the feature extraction after points projection is the multi-valued and zero-valued mapping caused by occlusion. As illustrated in Fig. 3 (a), if the camera is above the LiDAR, region A is observed by the camera but not LiDAR due to occlusion, resulting in no points after projection in this region (zero-valued mapping, the gap in region A, Fig. 3 (b)). On the other hand, region B is observed by the LiDAR but not the camera, points in this region (the red dots in Fig. 3 (a)) after projection will intervene the projection of points on its foreground (the black dots in Fig. 3 (a)). As a result, points of the foreground and background will correspond to the same image regions (multi-valued mapping, region B, Fig. 3 (b)). These phenomenon may not be significant for LiDARs of low-resolution [13], but is evident in as the LiDAR resolution increases (see Fig. 3 (b)). Extracting features on the projected point clouds and match them to the image features, such as [13], would suffer from these fundamental problems and cause significant errors in edge extraction and calibration.

To avoid the zero-valued and multi-valued mapping problem caused by projection, we extract edge features directly on the LiDAR point cloud. There are two types of edges: depth-

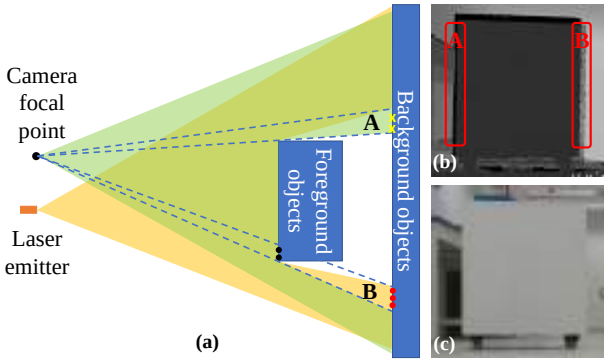


Fig. 3: Multi-valued mapping and zero-valued mapping.

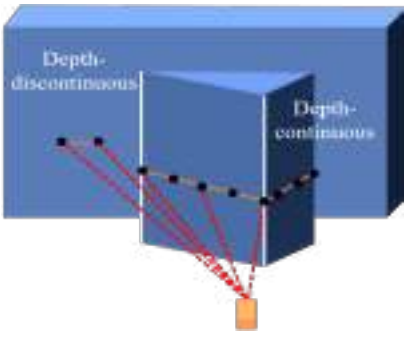


Fig. 4: Depth-discontinuous edge and depth-continuous edge.

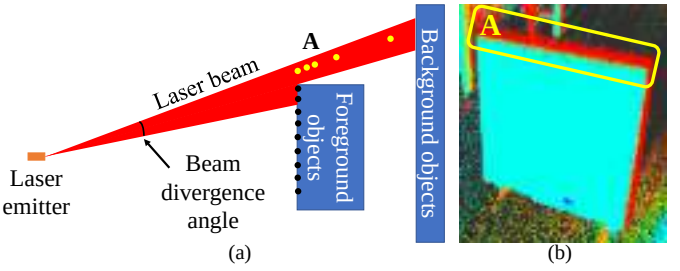


Fig. 5: Foreground objects inflation and bleeding points caused by laser beam divergence angle.

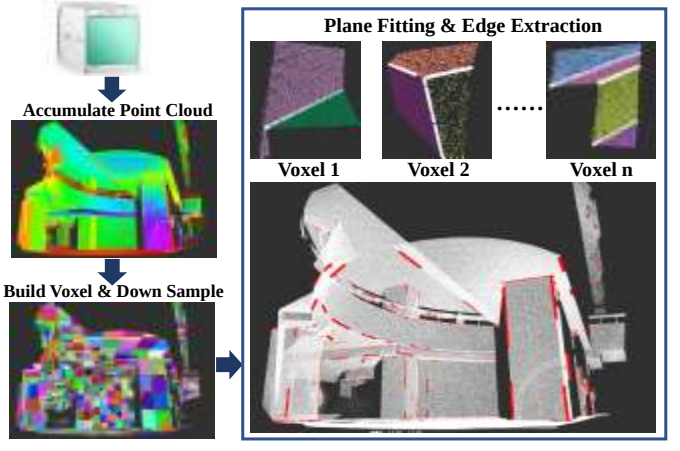


Fig. 6: Depth-continuous edge extraction. In each voxel grid, different colors represent different voxels. Within the voxel, different colors represent different planes, and the white lines represent the intersections between planes.

discontinuous and depth-continuous. As shown in Fig. 4 (a), depth-discontinuous edges refer to edges between foreground objects and background objects where the depth jumps. In contrast, depth-continuous edges refer to the plane joining lines where the depth varies continuously. Many existing methods [14, 15] use the depth-discontinuous edges as they can be easily extracted by examining the point depth. However, carefully investigating the LiDAR measuring principle, we found that depth-discontinuous edges are not reliable nor accurate for high accuracy calibration. As shown in Fig. 5, a practical laser pulse is not an ideal point but a beam with a certain divergence angle (i.e., the beam divergence angle). When scanning from a foreground object to a background one, part of the laser pulse is reflected by the foreground object while the remaining reflected by the background, producing two reflected pulses to the laser receiver. In the case of the high reflectivity of the foreground object, signals caused by the first pulse will dominate, even when the beam centerline is off the foreground object, this will cause fake points of the foreground object beyond the actual edge (the leftmost yellow point in Fig. 5 (a)). In the case the foreground object is close to the background, signals caused by the two pulses will join, and the lumped signal will lead to a set of points connecting the foreground and the background (called the *bleeding points*, the yellow points in Fig. 5 (a)). The two phenomena will mistakenly inflate the foreground object and cause significant errors in the edge extraction (Fig. 5 (b)) and calibration.

To avoid the foreground inflation and bleeding points caused by depth-discontinuous edges, we propose to extracting depth-continuous edges. The overall procedure is summarized in Fig. 6: we first divide the point cloud into small voxels of given sizes (e.g., $1m$ for outdoor scenes and $0.5m$ for indoor scenes). For each voxel, we repeatedly use RANSAC to fit and extract planes contained in the voxel. Then, we retain plane pairs that are connected and form an angle within a certain range (e.g., $[30°, 150°]$) and solve for the plane intersection lines (i.e., the depth-continuous edge). As shown in Fig. 6, our method is able to extract multiple intersection lines that are perpendicular or parallel to each other within a voxel. Moreover, by properly choosing the voxel size, we can even extract curved edges.

Fig. 7 shows a comparison between the extracted depth-continuous and depth-discontinuous edges when overlaid with the image using correct extrinsic value. The depth-discontinuous edge is extracted based on the local curvature as in [18]. It can be seen that the depth-continuous edges are more accurate and contain less noise.

For image edge extraction, we use the Canny algorithm [19]. The extracted edge pixels are saved in a k-D tree ($k = 2$) for the correspondence matching.

Fig. 7: Comparison between extracted depth-continuous edges and depth-discontinuous edges.

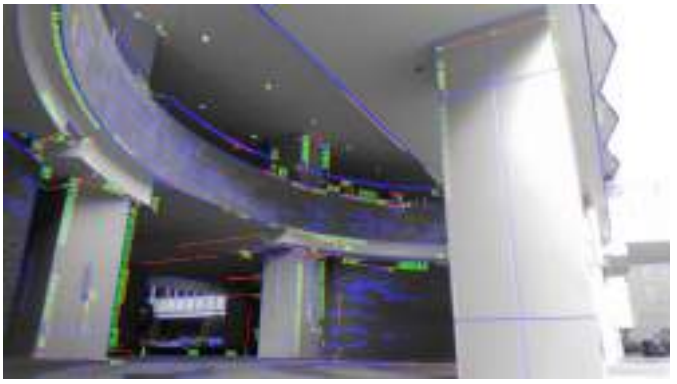


Fig. 8: LiDAR edges (red lines), image edge pixels (blue lines) and their correspondences (green lines).

2) Matching: The extracted LiDAR edges need to be matched to their corresponding edges in the image. For each extracted LiDAR edge, we sample multiple points on the edge. Each sampled point ${}^{L}\mathbf{P}_i \in \mathbb{R}^3$ is transformed into the camera frame (using the current extrinsic estimate ${}^{C}_{L}\bar{\mathbf{T}} = ({}^{C}_{L}\bar{\mathbf{R}}, {}^{C}_{L}\bar{\mathbf{t}}) \in SE(3)$)

$$ {}^{C}\mathbf{P}_i = {}^{C}_{L}\bar{\mathbf{T}}({}^{L}\mathbf{P}_i) \in \mathbb{R}^3, \tag{1} $$

where ${}^{C}_{L}\bar{\mathbf{T}}({}^{L}\mathbf{P}_i) = {}^{C}_{L}\bar{\mathbf{R}} \cdot {}^{L}\mathbf{P}_i + {}^{C}_{L}\bar{\mathbf{t}}$ denotes applying the rigid transformation ${}^{C}_{L}\mathbf{T}$ to point ${}^{L}\mathbf{P}_i$. The transformed point ${}^{C}\mathbf{P}_i$ is then projected onto the camera image plane to produce an expected location ${}^{C}\mathbf{p}_i \in \mathbb{R}^2$

$$ {}^{C}\mathbf{p}_i = \boldsymbol{\pi}({}^{C}\mathbf{P}_i) \tag{2} $$

where $\boldsymbol{\pi}(\mathbf{P})$ is the pin-hole projection model. Since the actual camera is subject to distortions, the actual location of the projected point $\mathbf{p}_i = (u_i, v_i)$ on the image plane is

$$ \mathbf{p}_i = \mathbf{f}({}^{C}\mathbf{p}_i) \tag{3} $$

where $\mathbf{f}(\mathbf{p})$ is the camera distortion model.

We search the κ nearest neighbor of $\mathbf{p}_i$ in the k-D tree built from the image edge pixels. Denotes $\mathbf{Q}_i = \{\mathbf{q}_i^j; j = 1, \cdots, \kappa\}$ the κ nearest neighbours and

$$ \mathbf{q}_i = \frac{1}{\kappa}\sum_{j=1}^{\kappa}\mathbf{q}_i^j; \quad \mathbf{S}_i = \sum_{j=1}^{\kappa}(\mathbf{q}_i^j - \mathbf{q}_i)(\mathbf{q}_i^j - \mathbf{q}_i)^T. \tag{4} $$

Then, the line formed by $\mathbf{Q}_i$ is parameterized by point $\mathbf{q}_i$ lying on the line and the normal vector $\mathbf{n}_i$, which is the eigenvector associated to the minimal eigenvalue of $\mathbf{S}_i$.

Besides projecting the point ${}^{L}\mathbf{P}_i$ sampled on the extracted LiDAR edge, we also project the edge direction to the image plane and validate its orthogonality with respect to $\mathbf{n}_i$. This can effectively remove error matches when two non-parallel lines are near to each other in the image plane. Fig. 8 shows an example of the extracted LiDAR edges (red lines), image edge pixels (blue lines) and the correspondences (green lines).

C. Extrinsic Calibration

1) Measurement noises: The extracted LiDAR edge points ${}^{L}\mathbf{P}_i$ and the corresponding edge feature $(\mathbf{n}_i, \mathbf{q}_i)$ in the image are subject to measurement noises. Let ${}^{I}\mathbf{w}_i \in \mathcal{N}(\mathbf{0}, {}^{I}\boldsymbol{\Sigma}_i)$ be the noise associated with $\mathbf{q}_i$ during the image edge extraction, its covariance is ${}^{I}\boldsymbol{\Sigma}_i = \sigma_I^2\mathbf{I}_{2\times 2}$, where $\sigma_I = 1.5$ indicating the one-pixel noise due to pixel discretization.

For the LiDAR point ${}^{L}\mathbf{P}_i$, let ${}^{L}\mathbf{w}_i$ be its measurement noise. In practice, LiDAR measures the bearing direction by

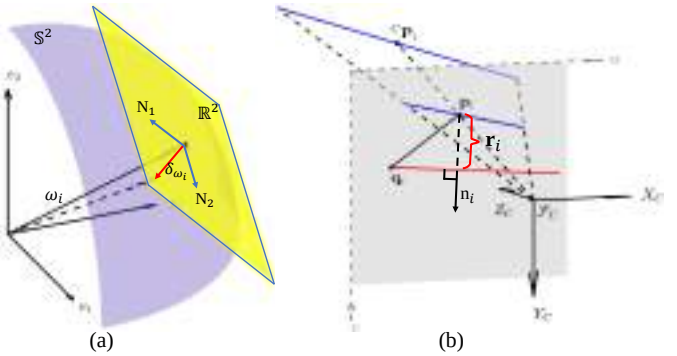


Fig. 9: (a) Perturbation to a bearing vector on $\mathbb{S}^2$; (b) Projection of a LiDAR edge point, expressed in the camera frame ${}^{C}\mathbf{P}_i$, to the image plane $\mathbf{p}_i$, and the computation of residual $\mathbf{r}_i$.

encoders of the scanning motor and the depth by computing the laser time of flight. Let $\boldsymbol{\omega}_i \in \mathbb{S}^2$ be the measured bearing direction and $\boldsymbol{\delta}_{\boldsymbol{\omega}_i} \sim \mathcal{N}(\mathbf{0}_{2\times 1}, \boldsymbol{\Sigma}_{\boldsymbol{\omega}_i})$ be the measurement noise in the tangent plane of $\boldsymbol{\omega}_i$ (see Fig. 9). Then using the $\boxplus$-operation encapsulated in $\mathbb{S}^2$ [20], we obtain the relation between the true bearing direction $\boldsymbol{\omega}_i^{\text{gt}}$ and its measurement $\boldsymbol{\omega}_i$ as below:

$$ \boldsymbol{\omega}_i^{\text{gt}} = \boldsymbol{\omega}_i \boxplus_{\mathbb{S}^2} \boldsymbol{\delta}_{\boldsymbol{\omega}_i} \triangleq e^{\lfloor \mathbf{N}(\boldsymbol{\omega}_i)\boldsymbol{\delta}_{\boldsymbol{\omega}_i} \times \rfloor}\boldsymbol{\omega}_i \tag{5} $$

where $\mathbf{N}(\boldsymbol{\omega}_i) = \begin{bmatrix}\mathbf{N}_1 & \mathbf{N}_2\end{bmatrix} \in \mathbb{R}^{3\times 2}$ is an orthonormal basis of the tangent plane at $\boldsymbol{\omega}_i$ (see Fig. 9 (a)), and $\lfloor \times \rfloor$ denotes the skew-symmetric matrix mapping the cross product. The $\boxplus_{\mathbb{S}^2}$-operation essentially rotates the unit vector $\boldsymbol{\omega}_i$ about the axis $\boldsymbol{\delta}_{\boldsymbol{\omega}_i}$ in the tangent plane at $\boldsymbol{\omega}_i$, the result is still a unit vector (i.e., remain on $\mathbb{S}^2$).

Similarly, let d_i be the depth measurement and $\delta_{d_i} \sim \mathcal{N}(0, \Sigma_{d_i})$ be the ranging error, then the ground-truth depth d_i^{gt} is

$$ d_i^{\text{gt}} = d_i + \delta_{d_i} \tag{6} $$

Combining (5) and (6), we obtain the relation between the ground-truth point location ${}^{L}\mathbf{P}_i^{\text{gt}}$ and its measurement ${}^{L}\mathbf{P}_i$:

$$ \begin{aligned} {}^{L}\mathbf{P}_i^{\text{gt}} &= d_i^{\text{gt}}\boldsymbol{\omega}_i^{\text{gt}} = (d_i + \delta_{d_i})\,(\boldsymbol{\omega}_i \boxplus_{\mathbb{S}^2} \boldsymbol{\delta}_{\boldsymbol{\omega}_i}) \\ &\approx \underbrace{d_i\boldsymbol{\omega}_i}_{{}^{L}\mathbf{P}_i} + \underbrace{\boldsymbol{\omega}_i\delta_{d_i} - d_i\lfloor\boldsymbol{\omega}_i\times\rfloor\mathbf{N}(\boldsymbol{\omega}_i)\boldsymbol{\delta}_{\boldsymbol{\omega}_i}}_{{}^{L}\mathbf{w}_i} \end{aligned} \tag{7} $$

Therefore,

$$ \begin{aligned} {}^{L}\mathbf{w}_i &= \underbrace{\begin{bmatrix}\boldsymbol{\omega}_i & -d_i\lfloor\boldsymbol{\omega}_i\times\rfloor\mathbf{N}(\boldsymbol{\omega}_i)\end{bmatrix}}_{\mathbf{A}_i}\begin{bmatrix}\delta_{d_i}\\ \boldsymbol{\delta}_{\boldsymbol{\omega}_i}\end{bmatrix} \sim \mathcal{N}(\mathbf{0}, {}^{L}\boldsymbol{\Sigma}_i), \\ {}^{L}\boldsymbol{\Sigma}_i &= \mathbf{A}_i\begin{bmatrix}\Sigma_{d_i} & \mathbf{0}_{1\times 2}\\ \mathbf{0}_{2\times 1} & \boldsymbol{\Sigma}_{\boldsymbol{\omega}_i}\end{bmatrix}\mathbf{A}_i^T. \end{aligned} \tag{8} $$

This noise model will be used to produce a consistent extrinsic calibration as detailed follow.

2) Calibration Formulation and Optimization: Let ${}^{L}\mathbf{P}_i$ be an edge point extracted from the LiDAR point cloud and its corresponding edge in the image is represented by its normal vector $\mathbf{n}_i \in \mathbb{S}^1$ and a point $\mathbf{q}_i \in \mathbb{R}^2$ lying on the edge (Section III-B.2). Compensating the noise in ${}^{L}\mathbf{P}_i$ and projecting it to the image plane using the ground-truth extrinsic should lie exactly on the edge $(\mathbf{n}_i, \mathbf{q}_i)$ extracted from the image (see (1 – 3) and Fig. 9 (b)):

$$ 0 = \mathbf{n}_i^T\left(\mathbf{f}\left(\boldsymbol{\pi}\left({}^{C}_{L}\mathbf{T}\left({}^{L}\mathbf{P}_i + {}^{L}\mathbf{w}_i\right)\right)\right) - \left(\mathbf{q}_i + {}^{I}\mathbf{w}_i\right)\right) \tag{9} $$

where ${}^{L}\mathbf{w}_i \in \mathcal{N}(\mathbf{0}, {}^{L}\mathbf{\Sigma}_i)$ and ${}^{I}\mathbf{w}_i \in \mathcal{N}(\mathbf{0}, {}^{I}\mathbf{\Sigma}_i)$ are detailed in the previous section.

Equation (9) implies that one LiDAR edge point imposes one constraint to the extrinsic, which is in agreement with Section III-A that an edge feature imposes two constraints to the extrinsic as an edge consists of two independent points. Moreover, (9) imposes a nonlinear equation for the extrinsic ${}^{C}_{L}\mathbf{T}$ in terms of the measurements ${}^{L}\mathbf{P}_i, \mathbf{n}_i, \mathbf{q}_i$ and unknown noise ${}^{L}\mathbf{w}_i, {}^{I}\mathbf{w}_i$. This nonlinear equation can be solved in an iterative way: let ${}^{C}_{L}\bar{\mathbf{T}}$ be the current extrinsic estimate and parameterize ${}^{C}_{L}\mathbf{T}$ in the tangent space of ${}^{C}_{L}\bar{\mathbf{T}}$ using the $\boxplus$-operation encapsulated in $SE(3)$ [20, 21]:

$$ {}^{C}_{L}\mathbf{T} = {}^{C}_{L}\bar{\mathbf{T}} \boxplus_{SE(3)} \delta\mathbf{T} \triangleq \mathrm{Exp}(\delta\mathbf{T}) \cdot {}^{C}_{L}\bar{\mathbf{T}} \tag{10} $$

where

$$ \delta\mathbf{T} = \begin{bmatrix} \delta\boldsymbol{\theta} \\ \delta\mathbf{t} \end{bmatrix} \in \mathbb{R}^6; \ \mathrm{Exp}(\delta\mathbf{T}) = \begin{bmatrix} e^{\lfloor \delta\boldsymbol{\theta} \times \rfloor} & \delta\mathbf{t} \\ 0 & 1 \end{bmatrix} \in SE(3). $$

Substituting (10) into (9) and approximating the resultant equation with first order terms lead to

$$ \begin{aligned} 0 &= \mathbf{n}_i^T \left(\mathbf{f}\left(\boldsymbol{\pi}\left({}^{C}_{L}\mathbf{T} \left({}^{L}\mathbf{P}_i + {}^{L}\mathbf{w}_i \right) \right) \right) - \left(\mathbf{q}_i + {}^{I}\mathbf{w}_i \right) \right) \\ &\approx \mathbf{r}_i + \mathbf{J}_{\mathbf{T}_i} \delta\mathbf{T} + \mathbf{J}_{\mathbf{w}_i} \mathbf{w}_i \end{aligned} \tag{11} $$

where

$$ \begin{aligned} \mathbf{r}_i &= \mathbf{n}_i^T \left(\mathbf{f}\left(\boldsymbol{\pi}\left({}^{C}_{L}\bar{\mathbf{T}}({}^{L}\mathbf{P}_i) \right) \right) - \mathbf{q}_i \right) \in \mathbb{R} \\ \mathbf{J}_{\mathbf{T}_i} &= \mathbf{n}_i^T \frac{\partial \mathbf{f}(\mathbf{p})}{\partial \mathbf{p}} \frac{\partial \boldsymbol{\pi}(\mathbf{P})}{\partial \mathbf{P}} \begin{bmatrix} -\lfloor ({}^{C}_{L}\bar{\mathbf{T}}({}^{L}\mathbf{P}_i)) \times \rfloor & \mathbf{I} \end{bmatrix} \in \mathbb{R}^{1\times 6} \\ \mathbf{J}_{\mathbf{w}_i} &= \begin{bmatrix} \mathbf{n}_i^T \frac{\partial \mathbf{f}(\mathbf{p})}{\partial \mathbf{p}} \frac{\partial \boldsymbol{\pi}(\mathbf{P})}{\partial \mathbf{P}} {}^{C}_{L}\bar{\mathbf{R}} & -\mathbf{n}_i^T \end{bmatrix} \in \mathbb{R}^{1\times 5} \\ \mathbf{w}_i &= \begin{bmatrix} {}^{L}\mathbf{w}_i \\ {}^{I}\mathbf{w}_i \end{bmatrix} \in \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_i), \mathbf{\Sigma}_i = \begin{bmatrix} {}^{L}\mathbf{\Sigma}_i & \mathbf{0} \\ \mathbf{0} & {}^{I}\mathbf{\Sigma}_i \end{bmatrix} \in \mathbb{R}^{5\times 5} \end{aligned} \tag{12} $$

The calculation of $\mathbf{r}_i$ is illustrated in Fig. 9 (b). Equation (11) defines the constraint from one edge correspondence, stacking all N such edge correspondences leads to

$$ \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}}_{\mathbf{0}} \approx \underbrace{\begin{bmatrix} \mathbf{r}_1 \\ \vdots \\ \mathbf{r}_N \end{bmatrix}}_{\mathbf{r}} + \underbrace{\begin{bmatrix} \mathbf{J}_{\mathbf{T}_1} \\ \vdots \\ \mathbf{J}_{\mathbf{T}_N} \end{bmatrix}}_{\mathbf{J}_\mathbf{T}} \delta\mathbf{T} + \underbrace{\begin{bmatrix} \mathbf{J}_{\mathbf{w}_1} & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{J}_{\mathbf{w}_N} \end{bmatrix}}_{\mathbf{J}_\mathbf{w}} \underbrace{\begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_N \end{bmatrix}}_{\mathbf{w}} \tag{13} $$

where

$$ \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}), \ \mathbf{\Sigma} = \mathrm{diag}(\mathbf{\Sigma}_1, \cdots, \mathbf{\Sigma}_N) $$

Equation (13) implies

$$ \mathbf{v} \triangleq -\mathbf{J}_\mathbf{w}\mathbf{w} = \mathbf{r} + \mathbf{J}_\mathbf{T}\delta\mathbf{T} \sim \mathcal{N}(\mathbf{0}, \mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T). \tag{14} $$

Based on (14), we propose our maximal likelihood (and meanwhile the minimum variance) extrinsic estimation:

$$ \begin{aligned} &\max_{\delta\mathbf{T}} \log p(\mathbf{v}; \delta\mathbf{T}) = \max_{\delta\mathbf{T}} \log \frac{e^{-\frac{1}{2}\mathbf{v}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1}\mathbf{v}}}{\sqrt{(2\pi)^N \det\left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)}} \\ &= \min_{\delta\mathbf{T}} (\mathbf{r} + \mathbf{J}_\mathbf{T}\delta\mathbf{T})^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1} (\mathbf{r} + \mathbf{J}_\mathbf{T}\delta\mathbf{T}) \end{aligned} \tag{15} $$

The optimal solution is

$$ \delta\mathbf{T}^* = -\left(\mathbf{J}_\mathbf{T}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1} \mathbf{J}_\mathbf{T} \right)^{-1} \mathbf{J}_\mathbf{T}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1} \mathbf{r} \tag{16} $$

This solution is updated to ${}^{C}_{L}\bar{\mathbf{T}}$

$$ {}^{C}_{L}\bar{\mathbf{T}} \leftarrow {}^{C}_{L}\bar{\mathbf{T}} \boxplus_{SE(3)} \delta\mathbf{T}^*. \tag{17} $$

The above process ((16 and (17)) iterates until convergence (i.e., $\|\delta\mathbf{T}^*\| < \varepsilon$) and the converged ${}^{C}_{L}\bar{\mathbf{T}}$ is the calibrated extrinsic.

3) Calibration Uncertainty: Besides the extrinsic calibration, it is also useful to estimate the calibration uncertainty, which can be characterized by the covariance of the error between the ground-truth extrinsic and the calibrated one. To do so, we multiply both sides of (13) by $\mathbf{J}_\mathbf{T}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1}$ and solve for $\delta\mathbf{T}$:

$$ \begin{aligned} \delta\mathbf{T} \approx & \underbrace{-\left(\mathbf{J}_\mathbf{T}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1} \mathbf{J}_\mathbf{T} \right)^{-1} \mathbf{J}_\mathbf{T}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1} \mathbf{r}}_{\delta\mathbf{T}^*} \\ & - \left(\mathbf{J}_\mathbf{T}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1} \mathbf{J}_\mathbf{T} \right)^{-1} \mathbf{J}_\mathbf{T}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1} \mathbf{J}_\mathbf{w}\mathbf{w} \\ & \sim \mathcal{N}\left(\delta\mathbf{T}^*, \left(\mathbf{J}_\mathbf{T}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1} \mathbf{J}_\mathbf{T} \right)^{-1} \right) \end{aligned} \tag{18} $$

which means that the ground truth $\delta\mathbf{T}$, the error between the ground truth extrinsic ${}^{C}_{L}\mathbf{T}$ and the estimated one ${}^{C}_{L}\bar{\mathbf{T}}$ and parameterized in the tangent space of ${}^{C}_{L}\bar{\mathbf{T}}$, is subject to a Gaussian distribution that has a mean $\delta\mathbf{T}^*$ and covariance equal to the inverse of the Hessian matrix of (15). At convergence, the $\delta\mathbf{T}^*$ is near to zero, and the covariance is

$$ \mathbf{\Sigma}_\mathbf{T} = \left(\mathbf{J}_\mathbf{T}^T \left(\mathbf{J}_\mathbf{w}\mathbf{\Sigma}\mathbf{J}_\mathbf{w}^T\right)^{-1} \mathbf{J}_\mathbf{T} \right)^{-1} \tag{19} $$

We use this covariance matrix to characterize our extrinsic calibration uncertainty.

D. Analysis of Edge Distribution on Calibration Result

The Jacobian $\mathbf{J}_{\mathbf{T}_i}$ in (12) denotes the sensitivity of residual with respect to the extrinsic variation. In case of very few or poorly distributed edge features, $\mathbf{J}_{\mathbf{T}_i}$ could be very small, leading to large estimation uncertainty (covariance) as shown by (19). In this sense, the data quality is automatically and quantitatively encoded by the covariance matrix in (19). In practice, it is usually useful to have a quick and rough assessment of the calibration scene before the data collection. This can be achieved by analytically deriving the Jacobian $\mathbf{J}_{\mathbf{T}_i}$. Ignoring the distortion model and substituting the pinhole projection model $\boldsymbol{\pi}(\cdot)$ to (12), we obtain

$$ \mathbf{J}_{T_i} = \mathbf{n}_i^T \begin{bmatrix} \frac{-f_x X_i Y_i}{Z_i^2} & f_x + \frac{f_x X_i^2}{Z_i^2} & \frac{-f_x Y_i}{Z_i} & \frac{f_x}{Z_i} & 0 & -\frac{f_x X_i}{Z_i^2} \\ -f_y - \frac{f_y Y_i^2}{Z_i^2} & \frac{f_y X_i Y_i}{Z_i^2} & \frac{f_y X_i}{Z_i} & 0 & \frac{f_y}{Z_i} & \frac{-f_y Y_i}{Z_i^2} \end{bmatrix} \tag{20} $$

where ${}^{C}\mathbf{P}_i = \begin{bmatrix} X_i & Y_i & Z_i \end{bmatrix}^T$ is the LiDAR edge point ${}^{L}\mathbf{P}_i$ represented in the camera frame (see (1)). It is seen that points near to the center of the image after projection (i.e., small $X_i/Z_i, Y_i/Z_i$) lead to small Jacobian. Therefore it is beneficial to have edge features equally distributed in the image. Moreover, since LiDAR noises increase with distance as in (8), the calibration scene should have moderate depth.

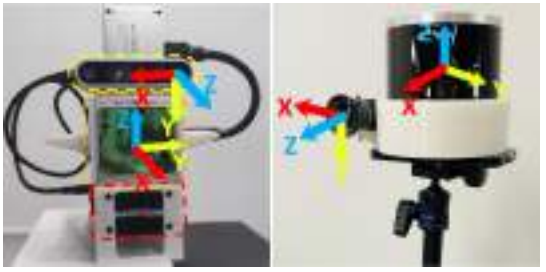


Fig. 10: Our sensor suite. Left is Livox Avia[1] LiDAR and Intel Realsense-D435i[2] camera, which is used for the majority of the experiments (Section IV.A and B). Right is spinning LiDAR (OS2-64[3])and industry camera (MV-CA013-21UC[4]), which is used for verification on fixed resolution LiDAR (Section IV.C) . Each sensor suite has a nominal extrinsic, e.g., for Livox AVIA, it is $(0, -\frac{\pi}{2}, \frac{\pi}{2})$. for rotation (ZYX Euler Angle) and zeros for translation.

E. Initialization and Rough Calibration

The presented optimization-based extrinsic calibration method aims for high-accuracy calibration but requires a good initial estimate of the extrinsic parameters that may not always be available. To widen its convergence basin, we further integrate an initialization phase into our calibration pipeline where the extrinsic value is roughly calibrated by maximizing the percent of edge correspondence ($P.C.$) defined below:

$$P.C. = \frac{N_{match}}{N_{sum}} \tag{21}$$

where N_{sum} is the total number of LiDAR edge points and N_{match} is the number of matched LiDAR edge points. The matching is based on the distance and direction of a LiDAR edge point (after projected to the image plane) to its nearest edge in the image (see Section III-B.2). The rough calibration is performed by an alternative grid search on rotation (grid size $0.5°$) and translation (grid size 2cm) over a given range.

IV. Experiments and Results

In this section, we validate our proposed methods in a variety of real-world experiments. We use a solid-state LiDAR called Livox AVIA, which achieves high-resolution point-cloud measurements at stationary due to its non-repetitive scanning [8], and an Intel Realsense-D435i camera (see Fig. 10). The camera intrinsic, including distortion models, has been calibrated beforehand. During data acquisition, we fix the LiDAR and camera in a stable position and collect point cloud and image simultaneously. The data acquisition time is 20 seconds for the Avia LiDAR to accumulate sufficient points.

A. Calibration Results in Outdoor and Indoor Scenes

We test our methods in a variety of indoor and outdoor scenes shown in Fig. 11, and validate the calibration performance as follows.

1) Robustness and Convergence Validation: To verify the robustness of the full pipeline, we test it on each of the 6 scenes individually and also on all scenes together. For each scene-setting, 20 test runs are conducted, each with a random initial value uniformly drawn from a neighborhood

[1]https://www.livoxtech.com/avia

[2]https://www.intelrealsense.com/depth-camera-d435i

[3]https://ouster.com/products/os2-lidar-sensor

[4]https://www.rmaelectronics.com/hikrobot-mv-ca013-21uc

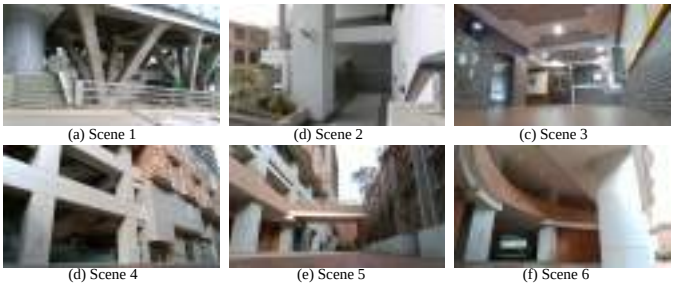


Fig. 11: Calibration scenes.

($\pm 5°$ for rotation and ± 10cm for translation) of the value obtained from the CAD model. Fig. 12 shows the percent of edge correspondence before and after the rough calibration and the convergence of the normalized optimization cost $\frac{1}{N_{match}} \mathbf{r}^T \left(\mathbf{J_w} \mathbf{\Sigma} \mathbf{J_w}\right)^{-1} \mathbf{r}$ during the fine calibration. It is seen that, in all 7 scene settings and 20 test runs of each, the pipeline converges for both rough and fine calibration.

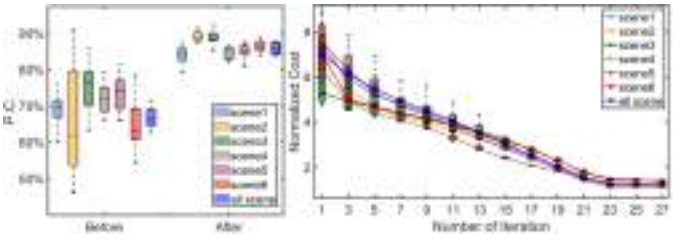


Fig. 12: Percent of edge correspondence before and after the rough calibration (left) and cost during the optimization (right).

Fig. 13 shows the distribution of converged extrinsic values for all scene settings. It is seen that in each case, the extrinsic value converges to almost the same value regardless of the large range of initial value distribution. A visual example illustrating the difference before and after our calibration pipeline is shown in Fig. 14. Our entire pipeline, including feature extraction, matching, rough calibration and fine calibration, takes less than 60 seconds

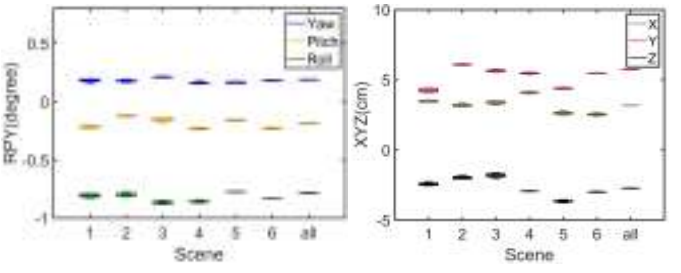


Fig. 13: Distribution of converged extrinsic values for all scene settings. The displayed extrinsic has its nominal part removed.

2) Consistency Evaluation: To evaluate the consistency of the extrinsic calibration in different scenes, we compute the

Fig. 14: LiDAR projection image overlaid on the camera image with an initial (left) and calibrated extrinsic (right). The LiDAR projection image is colored by Jet mapping on the measured point intensity.

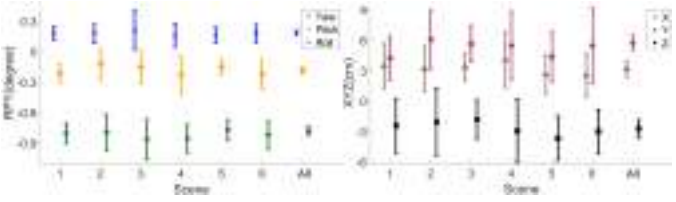


Fig. 15: Calibrated extrinsic with 3σ bound in all 7 scene settings. The displayed extrinsic has its nominal part removed.

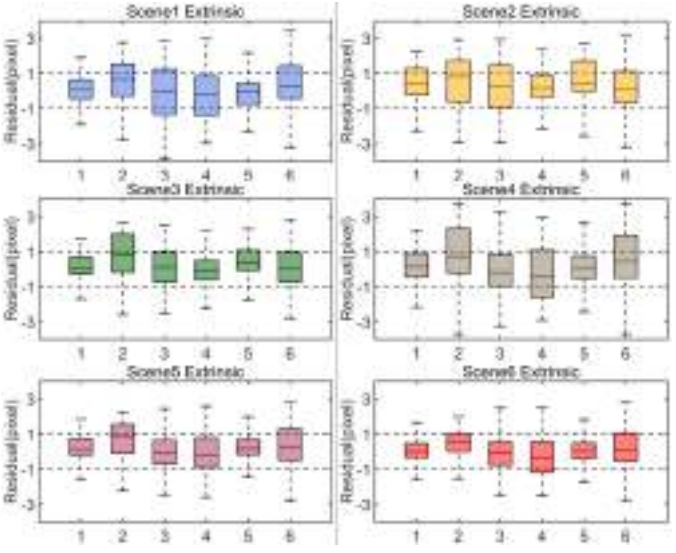


Fig. 16: Cross validation results. The 6 box plots in i-th subfigure summarizes the statistical information (from up to down: maximum, third quartile, median, first quartile, and minimum) of residuals of the six scenes applied with extrinsic calibrated from scene i.

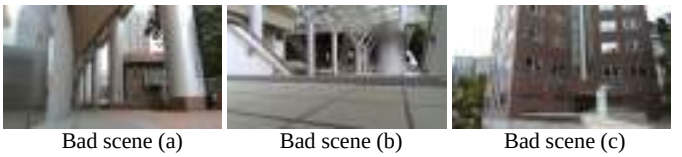
standard deviation of each degree of freedom of the extrinsic:

$$\sigma_k = \sqrt{(\mathbf{\Sigma_T})_{k,k}}, \quad k \in \{1, 2, ..., 6\} \tag{22}$$

where $\Sigma_\mathbf{T}$ is computed from (19). As shown in Fig. 13, the converged extrinsic in a scene is almost identical regardless of its initial value, we can therefore use this common extrinsic to evaluate the standard deviation of the extrinsic calibrated in each scene. The results are summarized in Fig. 15. It is seen that the 3σ of all scenes share overlaps and as expected, the calibration result based on all scenes have much smaller uncertainty and the corresponding extrinsic lies in the 3σ confidence level of the other 6 scenes. These results suggest that our estimated extrinsic and covariance are consistent.

3) Cross Validation: We evaluate the accuracy of our calibration via cross validation among the six individual scenes. To be more specific, we calibrate the extrinsic in one scene and apply the calibrated extrinsic to the calibration scene and all the rest five scenes. We obtain the residual vector $\mathbf{r}$ whose statistical information (e.g., mean, median) reveal the accuracy quantitatively. The results are summarized in Fig. 16, where the 20% largest residuals are considered as outliers and removed from the figure. It is seen that in all calibration and validation scenes (36 cases in total), around 50% of residuals, including the mean and median, are within one pixel. This validates the robustness and accuracy of our methods.

4) Bad Scenes: As analyzed in Section III-D, our method requires a sufficient number of edge features distributed properly. This does put certain requirements to the calibration scene. We summarize the scenarios in which our algorithm does not work well in Fig.17. The first is when the scene contains all cylindrical objects. Because the edge extraction is based on plane fitting, round objects will lead to inaccurate edge extraction. Besides, cylindrical objects will also cause parallax issues, which will reduce calibration accuracy. The second is the uneven distribution of edges in the scene. For example, most of the edges are distributed in the upper part of the image, which forms poor constraints that are easily affected by measurement noises. The third is when the scene contains edges only along one direction (e.g., vertical), in which the constraints are not sufficient to uniquely determine the extrinsic parameters.



Bad scene (a) Bad scene (b) Bad scene (c)

Fig. 17: Examples of bad scenarios.

B. Comparison Experiments

We compare our methods with [4, 6], which both use checkerboard as a calibration target. The first one ACSC [4] uses point reflectivity measured by the LiDAR to estimate the 3D position of each grid corner on the checkerboard and computes the extrinsic by solving a 3D-2D PnP problem. Authors of [4] open sourced their codes and data collected on a Livox LiDAR similar to ours, so we apply our method to their data. Since each of their point cloud data only has 4 seconds data, it compromises the accuracy of the edge extraction in our method. To compensate for this, we use three (versus 12 used for ACSC [4]) sets of point clouds in the calibration to increase the number of edge features. We compute the residuals in (12) using the two calibrated extrinsic, the quantitative result is shown in Fig. 18 (a).

The second method [6] estimates the checkerboard pose from the image by utilizing the known checkerboard pattern size. Then, the extrinsic is calibrated by minimizing the distance from LiDAR points (measured on the checkerboard) to the checkerboard plane estimated from the image. The method is designed for multi-line spinning LiDARs that have much lower resolution than ours, so we adopt this method to our LiDAR and test its effectiveness. The method in [6] also considers depth-discontinuous edge points, which are less accurate and unreliable on our LiDAR. So to make a fair comparison, we only use the plane points in the calibration when using [6]. To compensate for the reduced number of constraints, we place the checkerboard at more than 36 different locations and poses. In contrast, our method uses only one pair of data. Fig. 19 shows the comparison results on one of the calibration scenes with the quantitative result supplied in Fig. 18 (b). It can be seen that our method achieves similar calibration accuracy with [6] although using no checkerboard information (e.g., pattern size). We also notice certain board inflation (the blue points in the zoomed image of Fig. 19), which is caused by laser beam divergence explained in Section

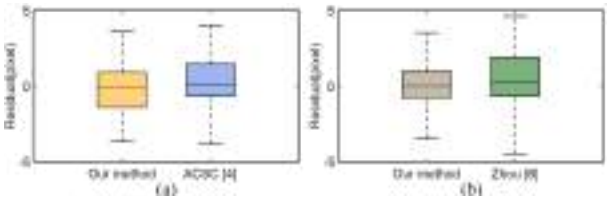
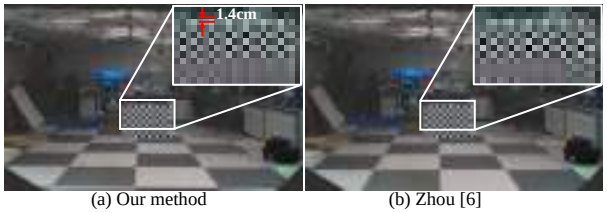


Fig. 18: Comparison of residual distribution.



(a) Our method (b) Zhou [6]

Fig. 19: Colored point cloud: (a) our method; (b) Zhou [6].

III-B. The inflated points are 1.4cm wide and at a distance of 6m, leading to an angle of $0.014/6 = 0.13°$, which agrees well with half of the vertical beam divergence angle (0.28°).

C. Applicability to Other Types of LiDARs

Besides Livox Avia, our method could also be applied to conventional multi-line spinning LiDARs, which possess lower resolution at stationary due to the repetitive scanning. To boost the scan resolution, the LiDAR could be moved slightly (e.g., a small pitching movement) so that the gaps between scanning lines can be filled. A LiDAR (-inertial) odometry [22] could be used to track the LiDAR motion and register all points to its initial pose, leading to a much higher resolution scan enabling the use of our method. To verify this, We test our methods on another sensor platform (see Fig. 10) consisting of a spinning LiDAR (Ouster LiDAR OS2-64) and an industrial camera (MV-CA013-21UC). The point cloud registration result and detailed quantitative calibration results are presented in the supplementary material (https://github.com/ChongjianYUAN/SupplementaryMaterials) due to the space limit. The results show that our method is able to converge to the same extrinsic for 20 initial values that are randomly sampled in the neighborhood ($\pm 3°$ in rotation and $\pm$5cm) of the value obtained from the CAD model.

V. Conclusion

This paper proposed a novel extrinsic calibration method for high resolution LiDAR and camera in targetless environments. We analyzed the reliability of different types of edges and edge extraction methods from the underlying LiDAR measuring principle. Based on this, we proposed an algorithm that can extract accurate and reliable LiDAR edges based on voxel cutting and plane fitting. Moreover, we theoretically analyzed the edge constraint and the effect of edge distribution on extrinsic calibration. Then a high-accuracy, consistent, automatic, and targetless calibration method is developed by incorporating accurate LiDAR noise models. Various outdoor and indoor experiments show that our algorithm can achieve pixel-level accuracy comparable to target-based methods. It also exhibits high robustness and consistency in a variety of natural scenes.

References

[1] L. Xiao, R. Wang, B. Dai, Y. Fang, D. Liu, and T. Wu. Hybrid conditional random field based camera-lidar fusion for road detection. *Information Sciences*, 432:543–558, 2018.

[2] M. Dimitrievski, P. Veelaert, and W. Philips. Behavioral pedestrian tracking using a camera and lidar sensors on a moving vehicle. *Sensors*, 19(2):391, 2019.

[3] J. Lin and F. Zhang. Loam livox: A fast, robust, high-precision lidar odometry and mapping package for lidars of small fov. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3126–3131, 2020.

[4] J. Cui, J. Niu, Z. Ouyang, Y. He, and D. Liu. Acsc: Automatic calibration for non-repetitive scanning solid-state lidar and camera systems, 2020.

[5] 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.

[6] 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.

[7] S. Chen, J. Liu, X. Liang, S. Zhang, J. Hyyppa, and R. Chen. A novel calibration method between a camera and a 3d lidar with infrared images. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[8] Z. Liu, F. Zhang, and X. Hong. Low-cost retina-like robotic lidars based on incommensurable scanning. *IEEE/ASME Transactions on Mechatronics*, page 1–1, 2021.

[9] 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.

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

[11] 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.

[12] 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.

[13] G. Pandey, J. McBride, S. Savarese, and R. Eustice. Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, 2012.

[14] 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.

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

[16] 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.

[17] 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.

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

[19] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[20] D. He, W. Xu, and F. Zhang. Embedding manifold structures into kalman filters. *arXiv preprint arXiv:2102.03804*, 2021.

[21] 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.

[22] W. Xu and F. Zhang. Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter. *IEEE Robotics and Automation Letters*, 6(2):3317–3324, 2021.