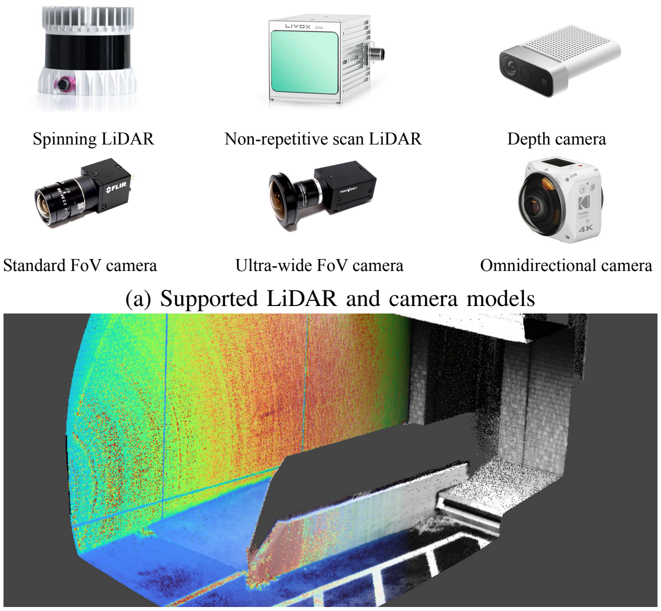
General, Single-shot, Target-less, and Automatic LiDAR-Camera Extrinsic Calibration Toolbox

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

Abstract— **This paper presents an open source LiDAR-camera calibration toolbox that is general to LiDAR and camera projection models, requires only one pairing of LiDAR and camera data without a calibration target, and is fully automatic. For automatic initial guess estimation, we employ the SuperGlue image matching pipeline to find 2D-3D correspondences between LiDAR and camera data and estimate the LiDAR-camera transformation via RANSAC. Given the initial guess, we refine the transformation estimate with direct LiDAR-camera registration based on the normalized information distance, a mutual information-based cross-modal distance metric. For a handy calibration process, we also present several assistance capabilities (e.g., dynamic LiDAR data integration and user interface for making 2D-3D correspondence manually). The experimental results show that the proposed toolbox enables calibration of any combination of spinning and non-repetitive scan LiDARs and pinhole and omnidirectional cameras, and shows better calibration accuracy and robustness than those of the state-of-the-art edge-alignment-based calibration method.**



(a) Supported LiDAR and camera models

(b) Calibration result (left to right: LiDAR to camera intensity)

Fig. 1: We present a complete LiDAR-camera calibration framework that can handle various LiDAR and camera models and calibrate the transformation between them from only a single pairing of a LiDAR point cloud and a camera image. The pixel-level direct alignment algorithm enables high-quality LiDAR-camera data fusion.

I. Introduction

LiDAR-camera extrinsic calibration is the task of estimating the transformation between the coordinate frames of a LiDAR and a camera. It is necessary for LiDAR-camera sensor fusion and is required for many applications, including autonomous vehicle localization, environmental mapping, and surrounding-object recognition.

Although LiDAR-camera calibration has been actively studied over the last decade, the robotics community still lacks a handy and complete LiDAR-camera calibration toolbox. The existing LiDAR-camera calibration frameworks require preparing a calibration target that is sometimes difficult to create [1], taking many shots of LiDAR-camera data that results in a large amount of effort [2], or choosing a geometry-rich environment carefully [3]. Furthermore, they rarely support various LiDAR and camera projection models, such as spinning and non-repetitive scan LiDARs and ultra-wide FoV and omnidirectional cameras. We believe that the lack of an easy-to-use LiDAR-camera calibration method has been a long-standing barrier to the development of LiDAR-camera sensor fusion systems.

As a benefit to the robotics community, herein we present a new complete LiDAR-camera calibration toolbox that has the following features:

- **Generalizable:** The proposed toolbox is sensor-model-independent and can handle various LiDAR and camera projection models, including spinning and non-repetitive scan LiDARs, and pinhole, fisheye, and omnidirectional projection cameras, as shown in Fig. 1 (a).
- **Target-less:** The proposed calibration algorithm does not require a calibration target but uses the environment structure and texture for calibration.
- **Single-shot:** At a minimum, only one pairing of a LiDAR point cloud and a camera image is required for calibration. Optionally, multiple LiDAR-camera data pairs can be used to further improve the calibration accuracy.
- **Automatic:** To make the calibration process fully automatic, the system includes an initial guess estimation algorithm with cross-modal 2D-3D correspondence matching based on SuperGlue [4].
- **Accurate and robust:** A pixel-level direct LiDAR-camera registration algorithm is employed to robustly and accurately perform LiDAR-camera calibration in environments without rich geometrical features, where existing edge alignment-based algorithms [3] would fail,

*This work was supported in part by a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

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

as long as there exists mutual information between LiDAR and camera data.

To our knowledge, there is no open implementation that has all the above features, and we believe the release of this toolbox will be beneficial to the robotics community.

The main contributions of this paper are as follows:

- We present a robust initial guess estimation algorithm based 2D-3D correspondence estimation. To take advantage of the recent graph neural network-based image matching [4], we generate a LiDAR intensity image with a virtual camera and find correspondences between the LiDAR intensity image and the camera image. An estimate of the LiDAR-camera transformation is then given by RANSAC and reprojection error minimization.
- For robust and accurate calibration, we combined a direct LiDAR-camera fine registration algorithm based on the normalized information distance (NID), a mutual-information (MI)-based cross-modal distance metric, with a view-based hidden points removal algorithm that filters out points that are occluded and should not be visible from the viewpoint of the camera.
- The entire system was carefully designed to be general to LiDAR and camera projection models so that it can be applied to various sensor models.
- We released the code of the developed method as open source to benefit the community [1].

II. Related Work

LiDAR-camera extrinsic calibration methods are categorized into three approaches: 1) target-based, 2) motion-based, and 3) scene-based.

A. Target-based calibration

As in the well-known camera intrinsic calibration process, the target-based approach is the most natural way for LiDAR-camera extrinsic calibration. Once we obtain the 3D coordinates of points on a calibration target and their corresponding 2D coordinates projected in the image, we can easily estimate the LiDAR-camera transformation by solving the perspective-n-point problem. The challenge here is that it is often difficult to design and create a calibration target that can robustly and accurately be detected by both the LiDAR and the camera. Several studies used a 3D structured calibration target that is not as easy to create as the well-known chessboard pattern [5], [6]. Although several other works used a planar pattern that is easy to create, they required manual annotation of LiDAR data [7] and multiple data acquisitions, resulting in a large amount of effort [8].

B. Motion-based calibration

As in the hand-eye calibration problem, we can estimate the transformation between two frames on a rigid body based on their motions [9]. This approach can easily handle heterogeneous sensors and does not need an overlap between sensors [10]. However, it requires careful time synchronization, which is not always possible when, for example, we use an affordable web camera. Furthermore, we need to estimate the per-sensor motion as accurately as possible for better calibration results.

C. Scene-based calibration

Scene-based methods estimate the LiDAR-camera transformation by considering the consistency between pairs of LiDAR point clouds and camera images. LiDAR points are projected in the image space, and then their consistency is measured with pixel values based on some metrics.

Similar to the visual odometry estimation problem, there are two major approaches for LiDAR-camera data consistency evaluation: indirect and direct.

The indirect approach first extracts feature points (e.g., edge points) from both the camera and LiDAR data, and computes the reprojection error between 2D-3D corresponding points [3], [11], [12]. While it exhibits good convergence due to the discriminative features and robust correspondence creation, it requires the environment to be geometry-rich to extract sufficient feature points. Because a majority of texture information for surfaces in the environment is discarded in this approach, it is less accurate compared with the direct approach.

The latter approach compares the pixel and point intensities directly. Because LiDAR and cameras usually exhibit very different intensity distributions, simply taking the difference between them does not work in practice. To overcome the difference in modalities, MI-based metrics have been used for LiDAR-camera data comparison [13], [14]. Because they do not use the difference between LiDAR and image intensities but instead consider their co-occurrence, they can robustly measure the consistency between LiDAR and camera intensity values. Following [15], [16], we used the NID metric derived from MI such that it satisfies the metric space axioms and becomes more robust than MI [15].

III. Methodology

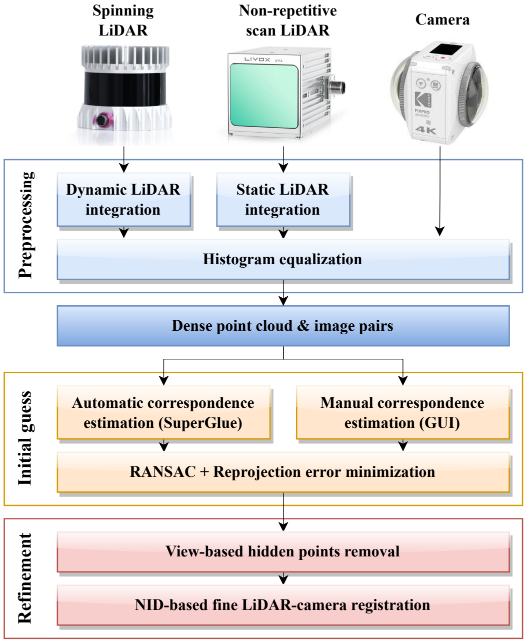
A. Overview

Fig. 2 shows an overview of the proposed LiDAR-camera calibration toolbox.

To handle various LiDAR models with a unified processing pipeline, we first create a dense point cloud by merging multiple LiDAR frames. For non-repetitive scan LiDARs (e.g., Livox Avia), we simply accumulate points to densify the point cloud. For spinning LiDARs (e.g., Velodyne and Ouster LiDARs), we use a dynamic LiDAR point integration technique based on continuous-time ICP [17].

From a paired densified point cloud and camera image, we then obtain a rough estimate of the LiDAR-camera transformation based on 2D-3D correspondences estimated with SuperGlue [4]. We also provide an intuitive user interface for creating manual correspondence for fail-safe. Given the 2D-3D correspondences, we perform RANSAC and reprojection error minimization to obtain an initial estimate of the LiDAR-camera transformation.

[1] The code is available at https://github.com/koide3/direct_visual_lidar_calibration



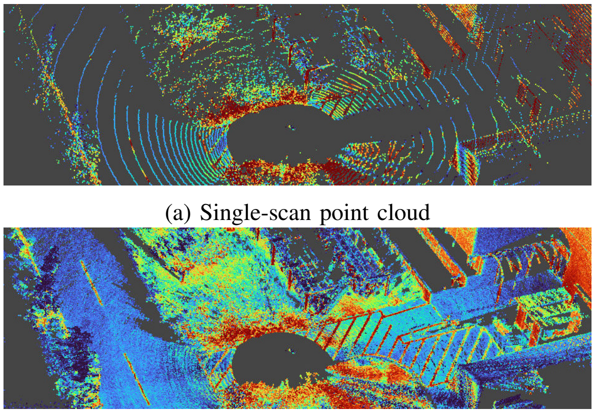
Spinning LiDAR
Non-repetitive scan LiDAR
Camera
Preprocessing
Dynamic LiDAR integration
Static LiDAR integration
Histogram equalization
Dense point cloud & image pairs
Initial guess
Automatic correspondence estimation (SuperGlue)
Manual correspondence estimation (GUI)
RANSAC + Reprojection error minimization
Refinement
View-based hidden points removal
NID-based fine LiDAR-camera registration

Fig. 2: Overview of proposed LiDAR-camera calibration process. Input point clouds are merged to create dense point clouds using static and dynamic LiDAR point integrators. Given the densified point cloud and camera image, we find 2D-3D correspondences using the SuperGlue pipeline. We also provide an easy-to-use manual correspondence estimation tool. Given the 2D-3D correspondences, a rough estimate of the LiDAR-camera transformation is obtained via RANSAC and reprojection error minimization. Finally, we perform fine LiDAR-camera registration based on NID minimization.

Given the initial estimate of the LiDAR-camera transformation, we apply view-based hidden points removal to remove LiDAR points that should not be visible from the viewpoint of the camera. We then refine the LiDAR-camera transformation estimate via fine LiDAR-camera registration based on NID minimization.

B. Notation

Our goal is to estimate the transformation between LiDAR and camera coordinate frames ${}^C\boldsymbol{T}_L$ from pairings of LiDAR point clouds $\mathcal{P}_i = [{}^L\boldsymbol{p}_1, \cdots, {}^L\boldsymbol{p}_N]$ with point intensities $\mathcal{L}_i = [l_1, \cdots, l_N]$ and camera images $\mathcal{I}_i(\boldsymbol{x}_j) = y_j$, where $\boldsymbol{x}_j \in \mathbb{R}^2$ are the pixel coordinates and y_j is the pixel intensity. A point in the LiDAR frame ${}^L\boldsymbol{p}_j$ is transformed into the camera coordinate frame as ${}^C\boldsymbol{p}_j = {}^C\boldsymbol{T}_L{}^L\boldsymbol{p}_j$ and projected into the image space using a projection function π; $\boldsymbol{x}_j = \pi\left({}^C\boldsymbol{p}_j\right)$. In this paper, we mainly use the conventional pinhole camera model with plumb-bob lens distortion and the omnidirectional equirectangular camera model [18] as the projection function. Note that other major camera models, including ATAN [19], fisheye [20], and unified omnidirectional camera models [21], have been implemented and are supported in the developed toolbox.



(a) Single-scan point cloud

(b) Densified point cloud

Fig. 3: Point cloud densification for spinning LiDARs. LiDAR integration enables the creation of a dense point cloud from a few seconds of dynamic LiDAR data. The densified point cloud exhibits rich geometrical and surface texture information.

C. Preprocessing

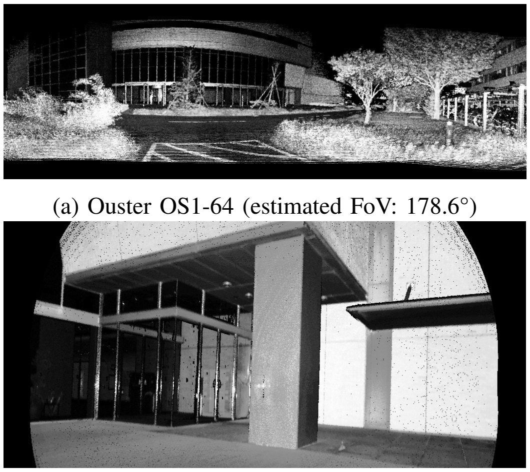
Spinning LiDARs (e.g., Ouster OS1-64) exhibit a sparse and repetitive scan pattern, and it is difficult to extract meaningful geometrical and texture information from only a single scan (see Fig. 3 (a)). For such a LiDAR, we move the LiDAR in the up-down direction for a few seconds and accumulate points while compensating for the viewpoint change and point cloud distortion. To estimate the LiDAR motion, we use the CT-ICP algorithm, which jointly optimizes the LiDAR poses at the scan beginning and end by minimizing the distance between the current LiDAR scan and a model point cloud with the interpolated LiDAR pose. To efficiently create the target point cloud from past observations, we use the linear iVox [22] structure, which simply keeps points in a linear container for each voxel. Based on the estimated LiDAR scan beginning and end poses, we correct the motion distortion on the input point cloud and create a dense point cloud by accumulating all points in the coordinate frame of the first scan. Fig. 3 (b) shows a densified point cloud after application of this dynamic LiDAR point integration process. We can see that the densified point cloud exhibits rich geometrical and texture information that was difficult to see in the single-scan point cloud.

For LiDARs with a non-repetitive scan mechanism (e.g., Livox Avia), we simply accumulate all scans into one frame, resulting in a dense point cloud, as shown in Fig. 1 (b).

For the densified point cloud and the camera image, we apply histogram equalization because the NID metric used in the fine registration step works best with uniform intensity distributions.

D. Initial Guess Estimation

To obtain a rough estimate of the LiDAR-camera transformation, we first obtain 2D-3D correspondences between the input images and point clouds and then estimate the LiDAR-camera transformation via RANSAC and reprojection error minimization.



(a) Ouster OS1-64 (estimated FoV: 178.6°)

(b) Livox Avia (estimated FoV: 76.2°)

Fig. 4: LiDAR intensity images rendered with virtual cameras (images are cropped due to space limitations). Either the pinhole or equirectangular projection model is selected depending on the FoV of the LiDAR.

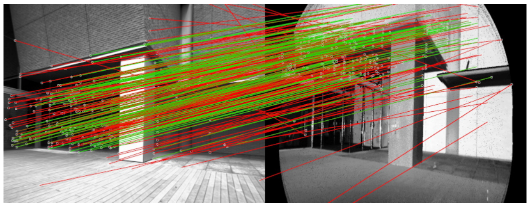


Fig. 5: SuperGlue can find correspondences between LiDAR and camera images in different modalities with a very low sensitive matching threshold setting. The result, however, contains many false correspondences that need to be filtered out before pose estimation (green: inliers, red: outliers).

To take advantage of the graph neural network-based image matching pipeline [4], we generate LiDAR intensity images from dense point clouds with a virtual camera model. To select the best projection model for rendering the entire point cloud, we first estimate the FoV of the LiDAR. We extract the convex hull of the input point cloud using the quickhull algorithm [23] and then find the point pair with the maximum angle distance in the convex hull using a brute-force search. If the estimated FoV of the LiDAR is smaller than 150°, we create a virtual camera with the pinhole projection model. Otherwise, we create a virtual camera with the equirectangular projection model. With the virtual camera, we render the point cloud with intensity values to obtain LiDAR intensity images, as shown in Fig. 4. Along with the intensity images, we also generate point index maps to efficiently look up pixel-wise 3D coordinates in the subsequent pose estimation step. Note that while we simply render each point without interpolation and gap filling, rendering results exhibit good appearance quality thanks to the densely accumulated point clouds.

To find correspondences between the LiDAR and camera intensity images, we use the SuperGlue pipeline [4]. It first detects keypoints on images using SuperPoint [24] and then finds correspondences between the keypoints using a graph neural network. The weights pretrained on the MegaDepth dataset is used in this work. While SuperGlue can find correspondences between images in different modalities, we found that the matching threshold needs to be set to a very small value (e.g., 0.05) to obtain a sufficient number of correspondences. However, with this setting, we observed that many false correspondences are created, as shown in Fig. 5.

Algorithm 1 Rotation-only RANSAC

```
Input: 2D keypoints $\mathcal{K} = [\boldsymbol{x}_1^K, \cdots, \boldsymbol{x}_M^K]$ and corresponding
    3D coordinates $\mathcal{D} = [{}^L\boldsymbol{p}_1^K, \cdots, {}^L\boldsymbol{p}_M^K]$
function EstimateRotationRANSAC($\mathcal{K}, \mathcal{D}$)
    ${}^C\boldsymbol{d}_j \leftarrow \frac{\pi^{-1}(\boldsymbol{x}_j^K)}{\|\pi^{-1}(\boldsymbol{x}_j^K)\|}$        ▷ Bearing vectors of $\boldsymbol{x}_j^K$
    ${}^L\boldsymbol{d}_j \leftarrow \frac{{}^L\boldsymbol{p}_j^K}{\|{}^L\boldsymbol{p}_j^K\|}$        ▷ Bearing vectors of ${}^L\boldsymbol{p}_j^K$
    for $i \in [1, \cdots, N^{\text{iteration}}]$ do
        Randomly sample two correspondences $j_0$ and $j_1$
        ${}^C\boldsymbol{R}_L \leftarrow$ FindRotationLSQ($j_0$, $j_1$)
        $N \leftarrow$ count of $\boldsymbol{x}_j^i$ s.t. $|\pi\left({}^C\boldsymbol{R}_L {}^L\boldsymbol{p}_j^K\right) - \boldsymbol{x}_j^K| < \alpha$
        if $i = 1$ or $N > N^*$ then
            $N^* \leftarrow N$
            ${}^C\boldsymbol{R}_L^* \leftarrow {}^C\boldsymbol{R}_L$
    return ${}^C\boldsymbol{R}_L^*$
function EstimateRotationLSQ($j_0$, $j_1$) [25]
    $\boldsymbol{A} \leftarrow [{}^C\boldsymbol{d}_{j_0} {}^C\boldsymbol{d}_{j_1}]$, $\boldsymbol{B} \leftarrow [{}^L\boldsymbol{d}_{j_0} {}^L\boldsymbol{d}_{j_1}]$
    $\boldsymbol{U\Sigma V}^* = \boldsymbol{AB}$        ▷ SVD
    $s \leftarrow 1$ if $|\boldsymbol{U}||\boldsymbol{V}| >= 0$ else $-1$
    return $\boldsymbol{U}\text{diag}([1, 1, s])\boldsymbol{V}^*$
```

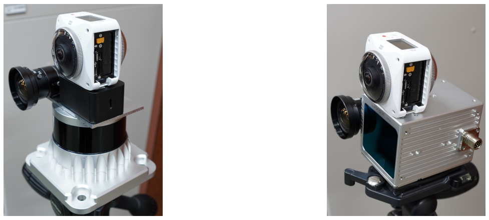
Given 2D keypoint coordinates $\mathcal{K} = [\boldsymbol{x}_1^K, \cdots, \boldsymbol{x}_M^K]$ and corresponding 3D coordinates in the LiDAR frame $\mathcal{D} = [{}^L\boldsymbol{p}_1^K, \cdots, {}^L\boldsymbol{p}_M^K]$, we first perform the rotation-only RANSAC described in Alg. 1 to robustly deal with outlier correspondences. Given the estimated rotation as an initial guess, we then obtain the 6 DoF LiDAR-camera transformation ${}^C\tilde{\boldsymbol{T}}_L$ by minimizing the reprojection errors of all keypoints using the Levenberg-Marquardt optimizer:

$${}^C\tilde{\boldsymbol{T}}_L = \underset{{}^C\boldsymbol{T}_L}{\arg\min} \sum_{j=1}^{M} \rho\left(\|\pi\left({}^C\boldsymbol{T}_L {}^L\boldsymbol{p}_j^K\right) - \boldsymbol{x}_j^K\|^2\right), \tag{1}$$

where ρ is the Cauchy robust kernel.

E. NID-based Direct LiDAR-Camera Registration

Some points in the LiDAR point cloud can be occluded and not visible from the camera due to the viewpoint difference. If we simply project all the LiDAR points, such points can cause false correspondences and affect the calibration result, as discussed in [3]. To avoid this problem, we apply efficient view-based hidden point removal to filter out LiDAR points that should not be visible from the viewpoint of the camera. With the current estimate of the LiDAR-camera transformation, we project the LiDAR points in the image and keep only the point with the minimum distance for each pixel (i.e., depth buffer testing). From the projected image



(a) Ouster OS1-64 (b) Livox Avia

Fig. 6: Sensor configurations for LiDAR-camera calibration experiments.

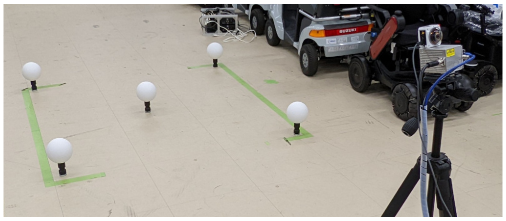


Fig. 7: Reference LiDAR-camera transformations were measured using high-reflectivity sphere targets; 2D and 3D positions of the targets were manually annotated and the transformation was estimated by minimizing their reprojection errors.

that retains only points visible from the camera, we obtain a 3D point cloud and use it for fine registration.

We then perform direct LiDAR-camera registration based on the NID metric [15]. To compute the NID, we transform LiDAR points ${}^{L}\boldsymbol{p}_j \in \mathcal{P}_i$ in the camera frame and project them into the image space $\boldsymbol{x}_j = \pi\left({}^{C}\boldsymbol{T}_L{}^{L}\boldsymbol{p}_j\right)$. From the LiDAR point intensities l_j and corresponding pixel intensities $\mathcal{I}_i\left(\boldsymbol{x}_j\right)$, we create $\mathrm{P}(\mathcal{L}_i)$, $\mathrm{P}(\mathcal{I}_i)$, and $\mathrm{P}(\mathcal{L}_i, \mathcal{I}_i)$, which are marginal and joint histograms of LiDAR and pixel intensities, and calculate their entropies $\mathrm{H}(\mathcal{L}_i)$, $\mathrm{H}(\mathcal{I}_i)$, and $\mathrm{H}(\mathcal{L}_i, \mathcal{I}_i)$ as follows:

$$\mathrm{H}\left(X\right) = -\sum_{x \in X} p(x) \log p(x), \tag{2}$$

where x is each bin in the histogram. The NID between $\mathcal{L}_i$ and $\mathcal{I}_i$ is then defined as follows:

$$\mathrm{NID}\left(\mathcal{L}_i, \mathcal{I}_i\right) = \frac{\mathrm{H}\left(\mathcal{L}_i, \mathcal{I}_i\right) - \mathrm{MI}\left(\mathcal{L}_i; \mathcal{I}_i\right)}{\mathrm{H}\left(\mathcal{L}_i, \mathcal{I}_i\right)}, \tag{3}$$

$$\mathrm{MI}\left(\mathcal{L}_i; \mathcal{I}_i\right) = \mathrm{H}\left(\mathcal{L}_i\right) + \mathrm{H}\left(\mathcal{I}_i\right) - \mathrm{H}\left(\mathcal{L}_i, \mathcal{I}_i\right), \tag{4}$$

where $\mathrm{MI}\left(\mathcal{L}_i; \mathcal{I}_i\right)$ is the mutual information between $\mathcal{L}_i$ and $\mathcal{I}_i$. By using the Nelder-Mead optimizer, we find the LiDAR-camera transformation that minimizes Eq. 3.

Because LiDAR points visible from the camera can change as the LiDAR-camera transformation is updated, we iterate the view-based hidden points removal and NID-based LiDAR-camera registration until the displacement of the transformation update converges.

IV. Experiment

We evaluated the proposed calibration toolbox with all four combinations of the spinning and non-repetitive scan LiDARs (Ouster OS1-64 and Livox Avia) and pinhole and omnidirectional cameras (Omron Sentech STC-MBS202POE and Kodak PixPro 4KVR360) shown in Fig. 6. For each combination, we recorded 15 pairs of LiDAR point clouds and camera images in indoor and outdoor environments, and we ran the proposed calibration process for each pair (i.e., single-shot calibration).

TABLE I: Calibration errors for Ouster OS1-64 and pinhole camera

		Proposed		Edge-based [3]	
Data	Init. guess	Trans. [m]	Rot. [°]	Trans. [m]	Rot. [°]
00	✓	0.019	0.688	0.043	0.135
01	✗	0.028	0.348	0.029	0.685
02	✗	0.014	0.180	0.035	0.490
03	✓	0.006	0.663	0.056	0.377
04	✗	0.029	0.400	0.158	1.499
05	✓	0.023	0.450	0.170	2.334
06	✓	0.033	0.613	0.136	1.034
07	✓	0.017	0.392	0.033	0.154
08	✓	0.048	0.386	0.520	1.458
09	✓	0.007	0.251	0.317	1.254
10	✓	0.064	0.210	0.191	1.463
11	✓	0.011	0.146	0.384	0.768
12	✓	0.325	0.530	0.226	0.524
13	✓	0.010	0.137	1.211	7.344
14	✓	0.015	0.217	0.034	0.421
Avg.	12 / 15	0.043	0.374	0.236	1.329

✓and ✗respectively represent the success and failure of the initial guess.

TABLE II: Calibration errors for Livox Avia and pinhole camera

		Proposed		Edge-based [3]	
Data	Init. guess	Trans. [m]	Rot. [°]	Trans. [m]	Rot. [°]
00	✓	0.047	0.478	1.054	6.964
01	✓	0.162	0.885	0.152	1.336
02	✓	0.028	0.356	1.587	14.278
03	✓	0.098	0.757	0.200	3.480
04	✓	0.124	1.430	0.065	0.933
05	✓	0.027	0.466	0.105	3.852
06	✓	0.032	0.410	0.128	1.577
07	✓	0.026	0.273	0.216	3.609
08	✓	0.031	0.270	0.358	4.450
09	✓	0.054	0.665	0.117	1.522
10	✓	0.071	0.887	0.214	1.590
11	✓	0.029	0.412	0.085	0.651
12	✓	0.046	0.297	0.181	2.133
13	✓	0.080	0.645	0.170	2.152
14	✓	0.032	0.452	0.210	3.934
Avg.	15 / 15	0.059	0.579	0.323	3.497

✓and ✗respectively represent the success and failure of the initial guess.

As a reference, we estimated the LiDAR-camera transformation using survey-grade high-reflectivity sphere targets, as shown in Fig.7. We manually annotated the 2D and 3D positions of the targets to find the LiDAR-camera transformations that minimized the target reprojection errors. From visual inspection, we confirmed that the estimated transformations closely describe the projection of the cameras. We used the estimated transformations as the “pseudo” ground truth.

Table I summarizes the calibration results for the combination of the Ouster LiDAR and pinhole camera. For the initial guess estimation, if the translation and rotation errors are smaller than 0.5 m and 1.0°, respectively, we consider the initial guess estimation as successful. As shown in Table I, the proposed algorithm provided a reasonable initial guess for 12 out of 15 data correlations. Fig. 8 shows the correspondences of a failure case (Dataset 02). We can see that the estimated

TABLE III: Calibration errors for Ouster OS1-64, Livox Avia, and omnidirectional camera

	Ouster OS1-64 & Omnidirectional			Livox Avia & Omnidirectional		
Data	Init.	Trans. [m]	Rot. [°]	Init.	Trans. [m]	Rot. [°]
00	✓	0.123	1.018	✓	0.029	0.515
01	✓	0.111	0.355	✗	0.059	0.902
02	✓	0.085	0.747	✓	0.071	0.518
03	✓	0.045	0.744	✓	0.037	0.958
04	✓	0.061	0.664	✗	0.067	1.265
05	✓	0.033	0.632	✗	0.113	1.447
06	✓	0.057	0.540	✗	0.070	0.312
07	✓	0.038	0.410	✓	0.056	1.466
08	✓	0.035	0.356	✓	0.079	0.635
09	✓	0.087	0.349	✓	0.407	1.164
10	✓	0.087	0.428	✗	0.161	0.753
11	✓	0.062	0.769	✓	0.098	0.681
12	✓	0.079	0.497	✓	0.212	0.936
13	✓	0.095	2.939	✓	0.033	0.409
14	✓	0.044	0.415	✓	0.028	0.149
Avg.	15 / 15	0.069	0.724	10 / 15	0.101	0.807

✓and ✗respectively represent the success and failure of the initial guess.

TABLE IV: Multi-data calibration errors

	Pinhole		Omnidirectional	
	Trans [m]	Rot [°]	Trans [m]	Rot [°]
Ouster	0.034	0.414	0.082	0.425
Livox	0.010	0.132	0.011	0.400

correspondences contain many false correspondences due to the flat and repetitive environment structure. Note that this kind of data pairing, where correct correspondences are difficult to find, can automatically be filtered out by RANSAC in multiple-data calibration and does not affect the estimation result. For data that resulted in a failed initial guess, we manually annotated 2D-3D correspondences and re-estimated the transformation for evaluation of the fine registration algorithm.

The proposed fine registration algorithm worked well for all the data and achieved 0.043 m and 0.374° calibration errors on average. As a baseline, we also applied the state-of-the-art edge alignment-based calibration method [3]. For this method, we used the reference LiDAR-camera transformation as the initial transformation, and thus it was evaluated with an almost ideal initial guess. However, as can be seen in Table I, it showed large calibration errors for several data correlations, resulting in worse average calibration errors (0.236 m and 1.329°). This was because both image and point cloud edge extraction in [3] were very sensitive to environment changes and could not properly extract edge points in several environments.

TABLE V: Processing time

LiDAR	Camera	Preprocessing	Init. guess	Calibration	Total
Ouster	Pinhole	56.6 s	10.3 s	28.8 s	95.7 s
Ouster	Omnidir.	56.3 s	11.2 s	181.5 s	249.0 s
Livox	Pinhole	9.3 s	10.6 s	54.5 s	74.4 s
Livox	Omnidir.	9.2 s	9.7 s	84.3 s	103.2 s

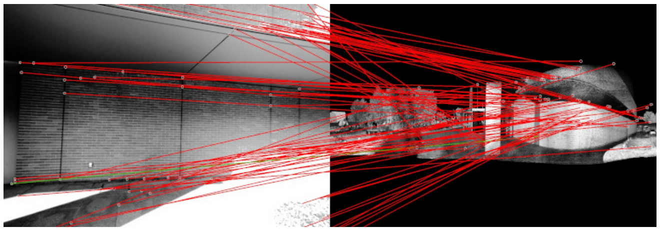


Fig. 8: Failure case for initial guess estimation (Ouster 02). (green: inliers, red: outliers)

As shown in Table II, for the combination of the Livox LiDAR and pinhole camera, the initial guess estimation successfully provided good initial LiDAR-camera transformations for all data pairings, and the fine registration algorithm achieved average calibration errors of 0.069 m and 0.724°, which were better than those for the edge-based calibration [3] (0.323 m and 3.497°).

Table III summarizes the calibration results for the omnidirectional camera. While it showed a good initial guess success rate (15 / 15) and low calibration errors for the Ouster LiDAR (0.069 m and 0.724°), both the initial guess estimation and the fine LiDAR-camera registration tended be degraded for the combination of the Livox LiDAR and omnidirectional camera (10 / 15 success rate, 0.101 m and 0.807°). This is because only a small portion of the omnidirectional camera images was used for calibration due to the very different FoVs of the LiDAR and camera, and the resolution of the images was not sufficient to represent fine environmental details with this limited FoV. We think that the calibration accuracy can be improved by using higher-resolution or multiple images.

Finally, we performed multi-data calibration with all data pairings and evaluated the calibration errors for all the LiDAR-camera combinations. Table IV summarizes the calibration errors. We can see that good calibration results were obtained even for the combination of the Livox and omnidirectional camera (0.011 m and 0.400°). For the combination of the Ouster LiDAR and pinhole camera, we observed the best calibration accuracy (0.010 m and 0.132°). This is because in this combination, the LiDAR and camera have similar FoVs and most of input data were used for calibration while some points and image regions are not used in other combinations due to the difference of FoV.

Table V shows the processing times for each calibration step. Depending on the combination of the LiDAR and camera models, it took from 74 to 249 s to calibrate the LiDAR-camera transformation from 15 data pairings. Although the combination of the Ouster LiDAR and omnidirectional camera took a longer time because both the sensors have 360° FoVs and need projection of most of the points, we consider it is in a reasonable level for offline calibration.

V. Conclusion

We developed a general LiDAR-camera calibration toolbox. For a fully automatic calibration process, we used image matching-based initial guess estimation. The initial estimate was then refined by a NID-based direct LiDAR-camera registration algorithm. The experimental results showed that the toolbox can accurately calibrate the transformation between spinning and non-repetitive scan LiDARs and pinhole and omnidirectional cameras.

References

[1] Y. Xie, L. Deng, T. Sun, Y. Fu, J. Li, X. Cui, H. Yin, S. Deng, J. Xiao, and B. Chen, "A4lidartag: Depth-based fiducial marker for extrinsic calibration of solid-state lidar and camera," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6487–6494, July 2022.

[2] D. Tsai, S. Worrall, M. Shan, A. Lohr, and E. Nebot, "Optimising the selection of samples for robust lidar camera calibration," in *IEEE International Intelligent Transportation Systems Conference*. IEEE, Sept. 2021, pp. 2631–2638.

[3] 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*, vol. 6, no. 4, pp. 7517–7524, Oct. 2021.

[4] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning feature matching with graph neural networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, June 2020, pp. 4938–4947.

[5] J. Beltran, C. Guindel, A. de Escalera la, and F. Garcia, "Automatic extrinsic calibration method for LiDAR and camera sensor setups," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, Mar. 2022.

[6] C. Fang, S. Ding, Z. Dong, H. Li, S. Zhu, and P. Tan, "Single-shot is enough: Panoramic infrastructure based calibration of multiple cameras and 3d LiDARs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Sept. 2021, pp. 8890–8897.

[7] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Sept. 2004, pp. 2301–2306.

[8] L. Zhou, Z. Li, and M. Kaess, "Automatic extrinsic calibration of a camera and a 3d LiDAR using line and plane correspondences," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Oct. 2018, pp. 5562–5569.

[9] R. Tsai and R. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand/eye calibration," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 345–358, June 1989.

[10] R. Ishikawa, T. Oishi, and K. Ikeuchi, "LiDAR and camera calibration using motions estimated by sensor fusion odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Oct. 2018, pp. 7342–7349.

[11] X. Zhang, S. Zhu, S. Guo, J. Li, and H. Liu, "Line-based automatic extrinsic calibration of LiDAR and camera," in *IEEE International Conference on Robotics and Automation*. IEEE, May 2021, pp. 9347–9353.

[12] X. Liu, C. Yuan, and F. Zhang, "Targetless extrinsic calibration of multiple small FoV LiDARs and cameras using adaptive voxelization," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, May 2022.

[13] G. Pandey, J. McBride, S. Savarese, and R. Eustice, "Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information," *AAAI Conference on Artificial Intelligence*, vol. 26, no. 1, pp. 2053–2059, Sept. 2021.

[14] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, "Automatic extrinsic calibration of vision and lidar by maximizing mutual information," *Journal of Field Robotics*, vol. 32, no. 5, pp. 696–722, Sept. 2014.

[15] A. Stewart, "Localisation using the appearance of prior structure," Ph.D. dissertation, Oxford University, UK, 2016.

[16] J. Jeong, Y. Cho, and A. Kim, "The road is enough! extrinsic calibration of non-overlapping stereo camera and LiDAR using road information," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2831–2838, July 2019.

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

[18] A. Torii, M. Havlena, and T. Pajdla, "From google street view to 3d city models," in *IEEE International Conference on Computer Vision Workshops*. IEEE, Sept. 2009, pp. 2188–2195.

[19] F. Devernay and O. Faugeras, "Straight lines have to be straight," *Machine Vision and Applications*, vol. 13, no. 1, pp. 14–24, Aug. 2001.

[20] J. Kannala and S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 8, pp. 1335–1340, Aug. 2006.

[21] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easily calibrating omnidirectional cameras," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Oct. 2006, pp. 5695–5701.

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

[23] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469–483, Dec. 1996.

[24] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: Self-supervised interest point detection and description," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, June 2018, pp. 224–236.

[25] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 4, pp. 376–380, Apr. 1991.