

Automatic Calibration of Multiple 3D LiDARs in Urban Environments

Jianhao Jiao¹, Yang Yu¹, Qinghai Liao¹, Haoyang Ye¹, Ming Liu¹

Abstract—Multiple LiDARs have progressively emerged on autonomous vehicles for rendering a wide field of view and dense measurements. However, the lack of precise calibration negatively affects their potential applications in localization and perception systems. In this paper, we propose a novel system that enables automatic multi-LiDAR calibration without any calibration target, prior environmental information, and initial values of the extrinsic parameters. Our approach starts with a hand-eye calibration for automatic initialization by aligning the estimated motions of each sensor. The resulting parameters are then refined with an appearance-based method by minimizing a cost function constructed from point-plane correspondences. Experimental results on simulated and real-world data sets demonstrate the reliability and accuracy of our calibration approach. The proposed approach can calibrate a multi-LiDAR system with the rotation and translation errors less than 0.04 [rad] and 0.1 [m] respectively for a mobile platform.

I. INTRODUCTION

A. Motivation

Accurate extrinsic calibration has become increasingly essential for the broad applications of multiple sensors. Numerous research work has been studied on [1]–[3]. Over the past decade, Light Detection and Ranging (LiDAR) sensors have appeared as a dominant sensor in mobile robotics for their active nature of providing accurate and stable distance measurements. They have been widely applied in 3D reconstruction [4], localization [5], and object detection [6]. However, the common drawbacks of LiDARs are the low spatial resolution on measurements and the sensibility to occlusion. These limit its potential utilization in robotic systems. Fig. 1 (bottom) presents two examples, which shows a point cloud captured by the top LiDAR. In the block A, the pedestrians and vehicles are scanned with a few points, making the detection of these objects challenging. About the block B, points are gathering together since they are occluded by the vehicle’s body. To solve these problems, the development of a multi-LiDAR configuration is necessary.

Traditional calibration techniques are realized by either placing markers in scenes or hand-labeled correspondences. But these approaches suffer from impracticality and limited scalability to the multi-LiDAR configuration. There are surprisingly few discussions on calibrating multiple 3D LiDARs. Moreover, the majority of current approaches involve one or more of the following assumptions: prior knowledge about the structure of environments [7], usage of additional sensors [8], and user-provided initial values of the extrinsic

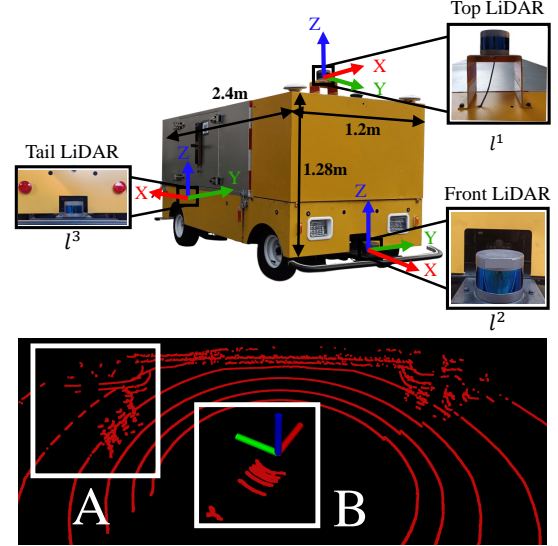


Fig. 1. (Top) Our mobile platform consists of a multi-LiDAR system with unknown extrinsic parameters. (Bottom) A point cloud captured by l_1 . The white boxes indicate two drawbacks presented in a single LiDAR configuration: (A) measurement sparsity and (B) occlusion.

parameters [9]. Inspired by the progress on the hand-eye calibration, we find that the extrinsic parameters can be directly recovered from individual motions provided by each LiDAR. In addition, the geometric features in environments also form constraints to solve the calibration. Therefore, we conclude that the complementary usage of these approaches is a prospective solution to the multi-LiDAR calibration.

B. Contribution

In this paper, we proposed a novel system which allows automatic calibration of multiple LiDARs in urban environments. This system consists of three components: **motion estimation** of each sensor, **motion-based initialization**, and appearance-based **refinement**. We show a variety of experiments to demonstrate the reliability and accuracy of our proposed approach. The contributions of this paper are summarized as following:

- A pipeline to automatically calibrate the extrinsic parameters of multiple LiDARs which releases the assumptions of calibration targets, prior knowledge about surroundings, and initial values given by users.
- A complementary usage of motion-based method for initialization and appearance-based method for refining the estimates.
- Extensive evaluation experiments on simulated and real-world data sets.

¹ Jianhao Jiao, Yang Yu, Qinghai Liao, Haoyang Ye and Ming Liu are with the Robotics and Multi-Perception Laboratory, Robotics Institute, The Hong Kong University of Science and Technology, Hong Kong SAR, China. {jjiao, yang.yu, qinghai.liao, hy.ye, eelium}@ust.hk

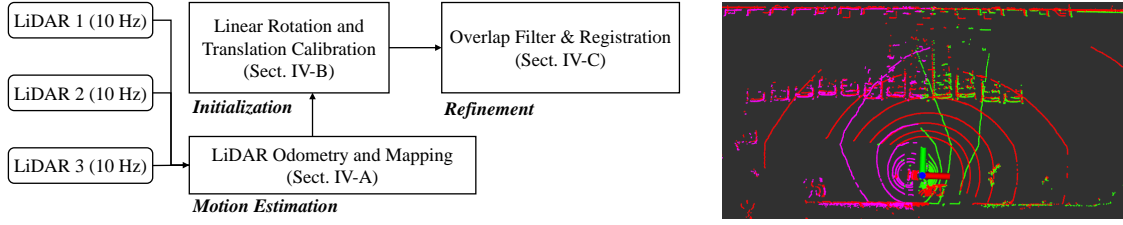


Fig. 2. This figure illustrates the full pipeline of the proposed approach (left) and the fused point clouds after calibration (right). Note that the red, green, and purple point clouds are captured by the top LiDAR, front LiDAR, and tail LiDAR respectively.

C. Organization

The rest of the paper is organized into the following sections. In Sect. II, the relevant literature is discussed. An overview of the complete system pipeline is given in Sect. III. The methodology of our approach which includes motion estimation, automatic initialization, and refinement is introduced in Sect. IV, followed by experimental results presented in Sect. V. Finally, Sect. VI summarizes the paper and discusses possible future work.

II. RELATED WORK

Besides the multi-LiDAR calibration, there are extensive discussions related to our work on the calibration among LiDARs, cameras, and IMUs. In this section, we categorize them as appearance-based or motion-based methods.

A. Appearance-based approaches

Appearance-based approaches that recover the spatial offset using appearance cues in surroundings are considered as a category of registration problem. The key challenge is searching correspondences among data. Artificial markers that are observable to sensors have been prevalently used to acquire correspondences. Gao et al. [8] proposed a multi-LiDAR calibration method using point constraints from the retro-reflective targets placed in scenes. Choi et al. [9] determined spatial offset of dual 2D LiDARs by employing the appearance of two orthogonal planes. For calibrating a LiDAR with a camera, Zhou et al. [3] demonstrated a technique to form line and plane constraints between the two sensors in the presence of a chessboard, while Liao et al. [10] published a toolkit using an arbitrary polygon, which is more general than the previous approach. All these methods require fixed markers, which are time-consuming and labor intensive. Our approach only depends on the common features such as edges and planes, which is more general and effective.

The automatic markerless calibration in an arbitrary scene has led the trend recently. He et al. [11] extracted geometric features among scan points to achieve robust registration, and their work was extended to a challenging scenario [12]. Levinson [13] first put forward an online calibration of a camera-LiDAR system. This is accomplished by aligning edge points with image contours and minimizing a cost function. Other metrics for matching sensor data are proposed including Renyi Quadratic Entropy [14], Mutual Information [15], and Gradient Orientation Measure [16]. However,

the success of these approaches highly relies on the prior knowledge of initial values. Compared with their methods, our approach can recover the initial extrinsic parameters from sensor's motions, which enables calibration in poor human intervention.

B. Motion-based approaches

The motion-based approaches treat calibration as a well-researched hand-eye calibration problem [17], where the extrinsic parameters are computed by combining the motions of all available sensors. The hand-eye calibration problem is usually referred to solve \mathbf{X} in $\mathbf{AX} = \mathbf{XB}$, where \mathbf{A} and \mathbf{B} are the motions the two sensors undergo, and \mathbf{X} is the transformation between them. As described in [18], [19], this problem has been addressed since 1980s. The ongoing research has extended this motion-based approaches to calibrate multiple sensors in outdoor environments. Heng et al. [1] proposed CamOdoCal, a versatile algorithm with a bundle adjustment to calibrate four cameras. For more general sensor configurations, Taylor et al. [20] provided a solution to calibrate sensors in three different modes. As presented in [2], [21], these approaches can also be utilized to estimate temporal offset between sensors. Additionally, several state-of-the-art visual-inertial navigation systems adopted the motion-based approaches to achieve online calibration of camera-IMU transformation [22]. Although the motion-based calibration has been extensively developed, the accuracy of the results is easily affected by the accumulated drifts of the estimated motions. In contrast, our method takes advantages of extracted appearance features to refine the motion-based calibration of multiple LiDARs.

III. OVERVIEW

The notations are defined as following. We denote $[0, K]$ the time interval during calibration, and define $\{l_k^i\}$ as the sensor coordinate system of the i^{th} LiDAR l^i at timestamp $k \in [0, K]$. The x -, y - and z - axes of these coordinate systems are pointing forward, left and upward respectively. We denote I the number of LiDARs to be calibrated, and l^1 the reference LiDAR among them. The transformation, rotation, and translation from $\{a\}$ to $\{b\}$ are denoted by \mathbf{T}_b^a , \mathbf{R}_b^a , and \mathbf{t}_b^a respectively. The unknown transformations from $\{l^1\}$ to $\{l^i\}$ are thus represented as $\mathbf{T}_{l^i}^{l^1}$ or $(\mathbf{R}_{l^i}^{l^1}, \mathbf{t}_{l^i}^{l^1})$. Additionally, the point cloud perceived by l^i at k is denoted by $\mathcal{P}^{l_k^i}$, and the extracted ground points are denoted by $\mathcal{G}^{l_k^i}$. We assume that LiDARs are synchronized, where point clouds are captured at the same time. We also assume that

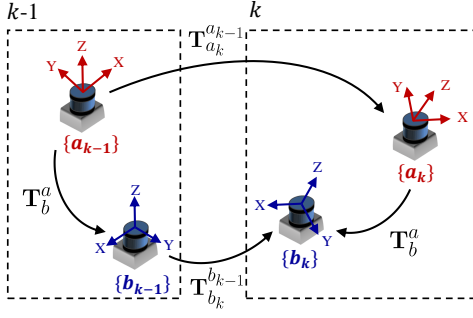


Fig. 3. The transformations between different LiDARs at $[k-1, k]$.

the vehicle is able to perform sufficient motions over a planar surface. With the pre-defined notations and assumptions, the calibration problem can be defined as:

Problem: Given a sequence of point clouds ($\mathcal{P}^{l_i}, i \leq I, k \leq K$) during the calibration, compute the extrinsic parameters of a multi-LiDAR system by combining the estimated motions with surrounding appearance.

The pipeline of our proposed calibration system consists of three phases, as illustrated in Fig. 2. The first phase takes point clouds as input, and results in the incremental motions of each LiDAR at within a time interval $[k-1, k]$ (Sect. IV-A). The second phase initializes $\mathbf{T}_{l_i}^{l^1}$ using a least-squares solution (Sect. IV-B). Finally, the third phase utilizes the appearance cues in surroundings to register difference LiDARs for refinement (Sect. IV-C).

IV. METHODOLOGY

A. Motion Estimation

To calculate a set of incremental motions between consecutive frames of each LiDARs, the LeGO-LOAM algorithm [5] is used. This method makes use of geometric features in environments to estimate the ego-motion. In our implementation, the individual transformations of all the sensors and the extracted ground point clouds are further used to provide constraints to the extrinsic parameters, which will be discussed in Sect. IV-B and Sect. IV-C.

B. Initialization

With l^1 as the reference, we present a method of calibrating l^i with l^1 pairwise. To simplify the notations, we replace $\{l^1\}, \{l^i\}$ with $\{a\}, \{b\}$ to indicate the coordinate system of reference sensor and target sensor respectively. The constant transformation of two LiDARs can be initialized by aligning their estimated motions. Fig. 3 depicts the relationship between the motions of two LiDARs and their relative transformation. As the vehicle moves, the extrinsic parameters can be recovered using these motions for any k :

$$\mathbf{T}_{a_k}^{a_{k-1}} \mathbf{T}_b^a = \mathbf{T}_b^a \mathbf{T}_{b_k}^{b_{k-1}}, \quad (1)$$

where (1) can be decomposed in terms of its rotation and translation components with the following two equations:

$$\mathbf{R}_{a_k}^{a_{k-1}} \mathbf{R}_b^a = \mathbf{R}_b^a \mathbf{R}_{b_k}^{b_{k-1}}, \quad (2)$$

$$(\mathbf{R}_{a_k}^{a_{k-1}} - \mathbf{I}_3) \mathbf{t}_b^a = \mathbf{R}_b^a \mathbf{t}_{b_k}^{b_{k-1}} - \mathbf{t}_{a_k}^{a_{k-1}}. \quad (3)$$

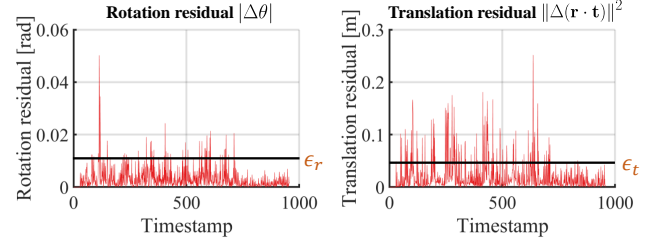


Fig. 4. The screw motion residuals in rotation and translation of a set of estimated motions. These motions are visualized as the blue line in Fig. 9 (left). ϵ_r, ϵ_t are the thresholds to filter the outliers.

The method described in [1] is used to solve these two equations. Based on (2), the pitch-roll rotation can be calculated directly using the estimated rotations, while the yaw rotation, the translation can be computed using (3).

1) *Outlier Filter:* The motions of dual sensors have two constraints that are independent from the extrinsic parameters, which were proposed as the screw motion in [23]:

$$\theta_{a_k}^{a_{k-1}} = \theta_{b_k}^{b_{k-1}} \quad (4)$$

$$\mathbf{r}_{a_k}^{a_{k-1}} \cdot \mathbf{t}_{a_k}^{a_{k-1}} = \mathbf{r}_{b_k}^{b_{k-1}} \cdot \mathbf{t}_{b_k}^{b_{k-1}}, \quad (5)$$

where θ denotes the angle of a rotation matrix \mathbf{R} , and \mathbf{r} is the corresponding rotation axis¹. The screw motion residuals include rotation and translation residuals, which are calculated as: $|\theta_a - \theta_b|, \|\mathbf{r}_a \cdot \mathbf{t}_a - \mathbf{r}_b \cdot \mathbf{t}_b\|^2$.

We adopt the screw motion residuals to evaluate the performance of the previous motions estimation phase. Fig. 4 shows the screw motion residuals in a real-world example, where we find the estimated motions are very noisy. Hence, the outliers are filtered if both their rotation and translation residuals are larger than the thresholds: ϵ_r, ϵ_t . The number of the filtered motions are denoted by N .

2) *Pitch-roll rotation computation:* It is challenging to use rotation matrix to solve (2) since the orthogonal constraint should be considered. For this reason, we employ the quaternion ($\mathbf{q} = [q_w, q_x, q_y, q_z]^T$) following Hamilton notation to represent rotation. (2) can be thus rewritten by substituting $\mathbf{q}_b^a = (\mathbf{q}_b^a)_z (\mathbf{q}_b^a)_{yx}$ as below:

$$\begin{aligned} \mathbf{q}_{a_n}^{a_{n-1}} \otimes (\mathbf{q}_b^a)_{yx} &= (\mathbf{q}_b^a)_{yx} \otimes \mathbf{q}_{b_n}^{b_{n-1}} \\ \Rightarrow [\mathbf{Q}_1(\mathbf{q}_{a_n}^{a_{n-1}}) - \mathbf{Q}_2(\mathbf{q}_{b_n}^{b_{n-1}})] \cdot (\mathbf{q}_b^a)_{yx} \\ \Rightarrow \mathbf{Q}_n^{n-1} \cdot (\mathbf{q}_b^a)_{yx} &= \mathbf{0}, \end{aligned} \quad (6)$$

where

$$\begin{aligned} \mathbf{Q}_1(\mathbf{q}) &= \begin{bmatrix} q_w \mathbf{I}_3 + [\mathbf{q}_{xyz}]_{\times} & \mathbf{q}_{xyz} \\ -\mathbf{q}_{xyz}^T & q_w \end{bmatrix} \\ \mathbf{Q}_2(\mathbf{q}) &= \begin{bmatrix} q_w \mathbf{I}_3 - [\mathbf{q}_{xyz}]_{\times} & \mathbf{q}_{xyz} \\ -\mathbf{q}_{xyz}^T & q_w \end{bmatrix} \end{aligned} \quad (7)$$

are matrix representations for left and right quaternion multiplication, $[\mathbf{q}_{xyz}]_{\times}$ is the skew-symmetric matrix of $\mathbf{q}_{xyz} = [q_x, q_y, q_z]^T$, and \otimes is the quaternion multiplication operator.

¹The rotation angle and axis are calculated using the $\log(\cdot)^\vee$ operator such that $\phi = \log(\mathbf{R})^\vee, \phi = \theta \mathbf{r}$.

With N pairs of filtered rotations, we are able to formulate an over-constrained linear system as follows:

$$\begin{bmatrix} \mathbf{Q}_1^0 \\ \mathbf{Q}_2^0 \\ \vdots \\ \mathbf{Q}_N^{N-1} \end{bmatrix} \cdot (\mathbf{q}_b^a)_{yx} = \mathbf{Q}_N \cdot (\mathbf{q}_b^a)_{yx} = \mathbf{0}, \quad (8)$$

where \mathbf{Q}_N is a $4N \times 4$ matrix. Using the singular value decomposition (SVD) $\mathbf{Q}_N = \mathbf{U}\mathbf{S}\mathbf{V}^\top$, $(\mathbf{q}_b^a)_{yx}$ is computed by the weighted sum of \mathbf{v}_3 and \mathbf{v}_4 :

$$(\mathbf{q}_b^a)_{yx} = \lambda_1 \mathbf{v}_3 + \lambda_2 \mathbf{v}_4, \quad (9)$$

where \mathbf{v}_3 and \mathbf{v}_4 are the last two column vectors of \mathbf{V} , and λ_1, λ_2 are two scalars. Therefore, $(\mathbf{q}_b^a)_{yx}$ can be obtained solving the following equations:

$$\begin{aligned} x(\mathbf{q}_b^a)_{yx} y(\mathbf{q}_b^a)_{yx} &= -z(\mathbf{q}_b^a)_{yx} w(\mathbf{q}_b^a)_{yx} \\ \|(\mathbf{q}_b^a)_{yx}\| &= 1, \end{aligned} \quad (10)$$

where $x_{\mathbf{q}}, y_{\mathbf{q}}, z_{\mathbf{q}}, w_{\mathbf{q}}$ are the elements of a quaternion.

3) *Yaw rotation and translation computation:* Due to our planar motion assumption, the translation offset on z -axis is unobservable. Consequently, We set $t_z = 0$ and rewrite (3) by removing the third row as follows:

$$\mathbf{R}_1 \begin{bmatrix} t_x \\ t_y \end{bmatrix} - \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) \\ \sin(\gamma) & \cos(\gamma) \end{bmatrix} \mathbf{t}_1 = -\mathbf{t}_2, \quad (11)$$

where t_x and t_y are unknown translations along x - and y -axes, and γ is the unknown rotation angle around z -axis. \mathbf{R}_1 is a 2×2 upper-left submatrix of $(\mathbf{R}_{a_n}^{a_{n-1}} - \mathbf{I}_3)$, $\mathbf{t}_1 = [t_{11}, t_{12}]^\top$ are the first two elements of $\mathbf{R}_b^a \mathbf{t}_{b_n}^{b_{n-1}}$, and \mathbf{t}_2 denote the first two elements of $\mathbf{t}_{a_n}^{a_{n-1}}$. We can rewrite (12) as a matrix vector equation:

$$\underbrace{\begin{bmatrix} \mathbf{R}_1 & \mathbf{J} \end{bmatrix}}_{\mathbf{G}_{2 \times 4}} \begin{bmatrix} t_x \\ t_y \\ -\cos(\gamma) \\ -\sin(\gamma) \end{bmatrix}, \quad (12)$$

where $\mathbf{J} = \begin{bmatrix} t_{11} & -t_{12} \\ t_{12} & t_{11} \end{bmatrix}$.

We can also construct a linear system from (12) with the filtered motions:

$$\underbrace{\begin{bmatrix} \mathbf{G}_1^0 \\ \mathbf{G}_2^0 \\ \vdots \\ \mathbf{G}_N^{N-1} \end{bmatrix}}_{\mathbf{A}_{2N \times 4}} \underbrace{\begin{bmatrix} t_x \\ t_y \\ -\cos(\gamma) \\ -\sin(\gamma) \end{bmatrix}}_{\mathbf{x}_{4 \times 1}} = - \underbrace{\begin{bmatrix} (\mathbf{t}_2)_1^0 \\ (\mathbf{t}_2)_2^0 \\ \vdots \\ (\mathbf{t}_2)_N^{N-1} \end{bmatrix}}_{\mathbf{b}_{2N \times 1}}, \quad (13)$$

where \mathbf{x} is obtained by applying the least-squares approach.

C. Refinement

In this section, we combine the coarse initialization results with the sensor measurements to refine the extrinsic parameters. Firstly, to recover the unknown t_z , the ground points are utilized. And then we estimate a set of transformations from $\{a\}$ to $\{b\}$ by registering all the available point clouds. To improve the registration accuracy, we apply an overlap filter to retain the points that lie in the overlapping regions.

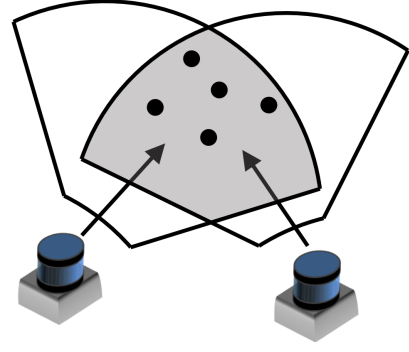


Fig. 5. The FOV of each LiDAR is visualized partially, where the gray area is the overlapping region.

1) *Ground planes alignment:* In the motion estimation phase, we have extracted K pairs of ground points $\mathcal{G}^{a_k}, \mathcal{G}^{b_k}$ of the reference and target LiDARs. Since the segmented point clouds are noisy, we implement the random sample consensus (RANSAC) plane fitting algorithm [24] to reject several outliers. Denoting $\mathbf{c}_k^a, \mathbf{c}_k^b$ as the centroids of $\mathcal{G}^{a_k}, \mathcal{G}^{b_k}$ after filtering, we use the mean value of $(\mathbf{c}_k^a - \mathbf{R}_b^a \mathbf{c}_k^b)_z$ at each timestamp to determine t_z .

2) *Overlap Filter:* The registration between these LiDARs is challenging since their overlapping field of view (FOV) is both limited and unknown. To tackle this issue, we propose an overlap filter to retains the points that lie within the counterpart LiDARs' FOV, as the gray area depicted in Fig. 5. Denoting $\mathcal{P}^{a_k}, \mathcal{P}^{b_k}$ the point clouds captured by a and b at k respectively, a point $\mathbf{p} \in \mathcal{P}^{b_k}$ can be transformed from $\{b\}$ to $\{a\}$ using the initial \mathbf{T}_b^a such that:

$$\tilde{\mathbf{p}} = \mathbf{T}_b^a \mathbf{p}. \quad (14)$$

Denoting $\mathcal{S}^{a_k}, \mathcal{S}^{b_k}$ the sets of points living in the volume of intersection between $\mathcal{P}^{a_k}, \mathcal{P}^{b_k}$. After transformation, we adopt the KD-Tree searching method [25] to construct them:

$$\begin{aligned} \mathcal{S}^{a_k} &= \left\{ \forall \mathbf{p}_1 \in \mathcal{P}^{a_k} : d(\mathbf{p}_1, \tilde{\mathbf{p}}_2) < r, \exists \mathbf{p}_2 \in \mathcal{P}^{b_k} \right\} \\ \mathcal{S}^{b_k} &= \left\{ \forall \mathbf{p}_2 \in \mathcal{P}^{b_k} : d(\tilde{\mathbf{p}}_2, \mathbf{p}_1) < r, \exists \mathbf{p}_1 \in \mathcal{P}^{a_k} \right\}, \end{aligned} \quad (15)$$

where \mathbf{p}_1 and \mathbf{p}_2 represent an individual point from each point cloud, $d(\cdot, \cdot)$ is the Euclidean distance between two points, and r is a threshold. We define the overlap factor as:

$$\Omega_k = \frac{|\mathcal{S}^{a_k}|}{|\mathcal{P}^{a_k}|} \cdot \frac{|\mathcal{S}^{b_k}|}{|\mathcal{P}^{b_k}|}, \quad (16)$$

where $|\cdot|$ is the size of a set.

The point clouds $\mathcal{S}^{a_k}, \mathcal{S}^{b_k}$ with $\Omega_k > 0.8$ are selected as the inputs for the following registration step.

3) *Registration:* To calculate the relative transformation between two point clouds, the point-to-plane Iterative Closest Point (ICP) is used. After registering all the point cloud pairs captured at a different time, we obtain a set of transformations. For each result, its registration error and calibration error compared with ground truth are computed. An example of these errors over timestamp is depicted in Fig. 6, where we

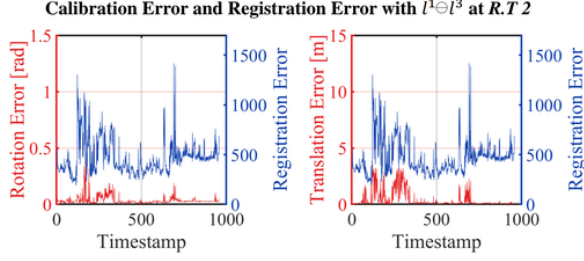


Fig. 6. This figure depicts the curves calibration error and registration error with $l^1 \ominus l^3$ at *R.T 2*.

find that the curves of registration error and calibration error have a similar trends, especially the positions of the peak. For this reason, we only select a series of transformations with the minimum registration error as candidates and acquire the optimal refinement results by computing their mean values.

V. EXPERIMENT

In this section, we divide the evaluation into two separate steps. Firstly, the initial calibration experiments are presented with simulated data and real sensor sets. Then we test the refined calibration on the real sensor data, and demonstrate that the initial results can be improved using appearance cues.

A. Implementation Details

We use the Ceres Solver [26] to solve the initialization problem and adopt the ICP library [27] to process point clouds. With empirically setting parameters: $\epsilon_r = 0.01$, $\epsilon_t = 0.01$, $r = 10$, our method can obtain promising results. The success of motion-based calibration highly depends on the quality of the motions which a vehicle undergo. For this reason, we design several paths with different rotations and scales to test our proposed algorithm. These paths include three simulated trajectories (*S.T 1-S.T 3*) and two real trajectories (*R.T 1-R.T 2*) on simulation and real sensor sets respectively. In the experiments, two platforms with different sensor setups are used for data collection.

1) *Simulation*: The simulated car platform² is a well known publicly available software. For testing, we manually mount two sensors at different positions on the platform, as shown in Fig. 7 (left). The rotation offset between them is approximately $[0, 3.14, 1.57]$ rads in roll, pitch, and yaw respectively. The corresponding translation are approximately $[-2.5, 1.5, 0]$ meters along x -, y -, and z - axes respectively. We can thus acquire the positions of these sensors in the form of ground truth at 5 Hz. The refinement is not tested in simulation because this platform does not provide stable point clouds without time distortion.

2) *Real Sensor*: While our approach is not limited to a particular number of target sensors, we are specifically interested in calibrating between the reference LiDAR and two target LiDARs based on our platform. As shown in Fig. 1, three 16-beam RS-LiDARs³ are rigidly mounted

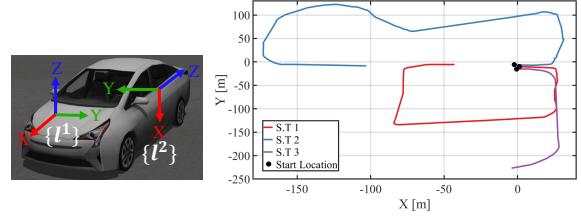


Fig. 7. (Left) The simulated platform and (right) the three trajectories which the platform follows. The rotation offset is about $[0, 3.14, 1.57]$ rads in roll, pitch, and yaw respectively. The corresponding translation are about $[-2.5, 1.5, 0]$ meters along x -, y -, and z - axes respectively.

TABLE I
THE INITIAL CALIBRATION RESULTS WITH SIMULATED DATA.

σ^2	Trajectory	Rotation Error [rad]		Translation Error [m]	
		Kabsch	Proposed	Kabsch	Proposed
σ_1^2	<i>S.T 1</i>	0.10	0.01	0.22	0.28
	<i>S.T 2</i>	0.80	0.00	0.66	0.28
	<i>S.T 3</i>	0.08	0.01	0.80	0.48
σ_2^2	<i>S.T 1</i>	1.37	0.07	2.13	1.25
	<i>S.T 2</i>	0.94	0.04	1.80	1.20
	<i>S.T 3</i>	1.17	0.02	1.66	1.44

on the vehicle. The setup of this multi-LiDAR system has significant transformation among sensors. Especially, l^3 is mounted with approximately 180 degree rotation offset in yaw. In later sections, we use $l^1 \ominus l^i$ to represent the configuration between l^1 and l^i . Since we do not know the precise extrinsic parameters of the multi-LiDAR system, we use the parameters (shown in Table II) provided by the manufacturer to evaluate our proposed algorithm.

With respect to the accuracy calculation, the error in rotation is measured according to the angle difference between the ground truth \mathbf{R}_{gt} and the resulting rotation $\mathbf{R}_{resulting}$, which is calculated as $e_r = \|\log(\mathbf{R}_{gt} \mathbf{R}_{resulting}^{-1})\|_2$. Similarly, the difference in translation is computed using vector subtraction as $e_t = \|\mathbf{t}_{gt} - \mathbf{t}_{resulting}\|_2$. The translation error on z - axis will not be counted of the initialization results because of the planar movement assumption.

B. Performance of Initialization

We take the modified Kabsch algorithm [19], [20] that operates at matrix representation for comparison.

1) *Simulation*: The simulated trajectories (*S.T 1-S.T 3*) are visualized in Fig. 7 (right), where the third one is considered as the most challenging one since it has fewer rotations. To verify the performance of our proposed algorithm, the sensor's motions are added with zero-mean Gaussian noise $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2)$. σ^2 is set to two values: $\sigma_1^2 = 0.0001$ and $\sigma_2^2 = 0.001$ for evaluation. For each value, all simulated motions are tested. The calibration results are shown in Table I. The proposed algorithm can successfully initialize the rotation offset with low error, and outperform the Kabsch algorithm in most of cases. We also note that the translation error with σ_2^2 is larger than 1m, meaning that our initialization approach is not robust in noisy data. But the usage of appearance-based refinement can solve this problem.

²<https://www.osrfoundation.org/simulated-car-demo/>

³<https://www.robosense.ai/rslidar/rs-lidar-16>

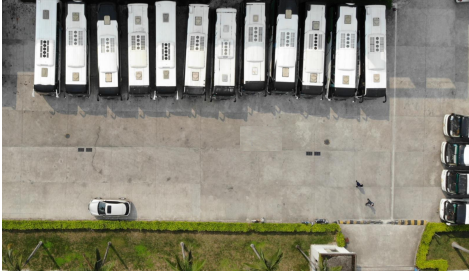


Fig. 8. The testing environment for our calibration experiments.

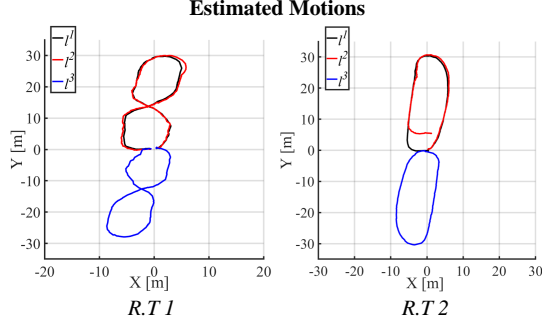


Fig. 9. The black, red, and blue lines indicate the estimated motions of l_1, l_2, l_3 respectively at R.T 1-R.T 2.

2) *Real Sensor*: We carry out a real sensor experiment to validate the proposed method. Two trajectories (R.T 1-R.T 2) are designed in an urban environment (shown in 8), and we drive the vehicle to follow these trajectories. After estimating the individual motions of each LiDAR, we can use these results to initial the calibration. The estimated motions are depicted in Fig. 9. We observe that the calculated trajectory of l^2 at R.T 2 drifts. The initialization results are presented in Table III. Our method performs well in recovering the rotation offset ($< 0.15\text{rad}$), but fail in calculating the translation offset ($> 0.5\text{m}$) in all cases. With the above results in simulation and real-world environments, we can conclude that the initialization phase can provide coarse estimates to the extrinsic parameters. For precise results, an additional refinement step is required.

C. Performance of Refinement

In our experiments, we select 10 resulting transformations as the candidates for the optimal refinement results. We plot their calibration errors in each case in Fig. 10. The detailed calibration results are shown in Table IV, where all the rotation and translation errors are less than 0.04 [rad] and 0.1 [m] respectively. Compared with the result in Table III, the refinement phase can improve the estimated parameters.

D. Discussion

Since our proposed method achieves accurate calibration results, but its performance will be influenced by the below conditions. Firstly, the initialization relies on the accuracy of estimated motions, especially when the translation offset is imprecise. Furthermore, we assume that the LiDARs' view should have overlapping regions. Finally, the registration between point clouds will fail in several feature-less scenes.

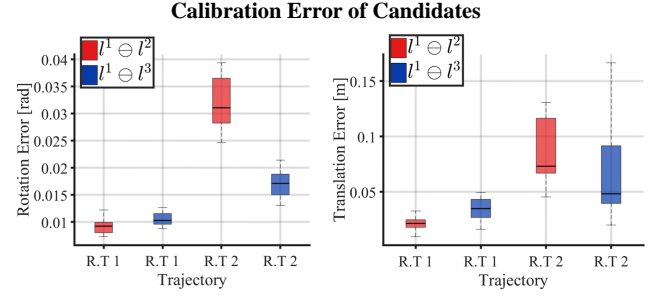


Fig. 10. The calibration errors in each case of the candidate. The means of them are small at R.T 1, but large at R.T 2.

TABLE II
THE CALIBRATION GROUND TRUTH OF OUR MULTI-LIDAR SYSTEM.

Conf.	Rotation [rad]			Translation [m]		
	x	y	z	x	y	z
$l^1 \ominus l^2$	0.01	0.08	0.03	0.42	0.00	-1.26
$l^1 \ominus l^3$	-0.02	0.01	-3.11	-2.11	0.06	-1.18

TABLE III
THE INITIALIZATION RESULTS.

Conf.	Traj.	Rotation Error [rad]		Translation Error [m]	
		Kabsch	Proposed	Kabsch	Proposed
$l^1 \ominus l^2$	R.T 1	0.94	0.06	1.60	0.47
	R.T 2	0.73	0.03	4.42	1.95
$l^1 \ominus l^3$	R.T 1	1.29	0.14	1.23	1.26
	R.T 2	0.60	0.08	1.36	2.04

TABLE IV
THE REFINEMENT RESULTS.

Conf.	Traj.	$l^1 \ominus l^2$		$l^1 \ominus l^3$	
		R.T 1	R.T 2	R.T 1	R.T 2
Rotation [rad]	x	0.01	0.01	-0.02	-0.02
	y	0.08	0.07	0.02	0.00
	z	0.04	0.04	-3.14	-3.13
	error	0.01	0.03	0.01	0.02
Translation [m]	x	0.43	0.46	-2.13	-2.07
	y	0.00	-0.01	0.09	0.08
	z	-1.26	-1.27	-1.18	-1.20
	error	0.01	0.08	0.03	0.04

VI. CONCLUSION AND FUTURE WORK

In this paper, we presented a novel method for automatically calibrating a multi-LiDAR system without any extra sensors, calibration target, or prior knowledge about surroundings. Our approach makes use of the complementary strengths of the motion-based and appearance-based calibration methods, which consists of three phases. The individual motions of each LiDAR are estimated by a LiDAR odometry algorithm. The calculated motions are then utilized to initialize the extrinsic parameters. Finally the results are refined by exploiting appearance cues in sensors' overlap. The performance of our method is demonstrated through a series of simulated and real-world experiments with reliable and accurate calibration results. There are several possible extensions to this work, (1) online multi-LiDAR calibration, (2) calibration without the overlapping requirement, (3) applications based on a multi-LiDAR system.

REFERENCES

- [1] L. Heng, B. Li, and M. Pollefeys, "Camodocal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 1793–1800.
- [2] Z. Taylor and J. Nieto, "Motion-based calibration of multimodal sensor extrinsics and timing offset estimation," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1215–1229, 2016.
- [3] L. Zhou, Z. Li, and M. Kaess, "Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences," in *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on*. IEEE, 2018.
- [4] J. Behley and C. Stachniss, "Efficient surfel-based slam using 3d laser range data in urban environments," in *Robotics: Science and Systems (RSS)*, 2018.
- [5] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," 2018.
- [6] X. Du, M. H. Ang, S. Karaman, and D. Rus, "A general pipeline for 3d detection of vehicles," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3194–3200.
- [7] Y. Xie, R. Shao, P. Guli, B. Li, and L. Wang, "Infrastructure based calibration of a multi-camera and multi-lidar system using apriltags," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 605–610.
- [8] C. Gao and J. R. Spletzer, "On-line calibration of multiple lidars on a mobile vehicle platform," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 279–284.
- [9] D.-G. Choi, Y. Bok, J.-S. Kim, and I. S. Kweon, "Extrinsic calibration of 2-d lidars using two orthogonal planes," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 83–98, 2016.
- [10] Q. Liao, Z. Chen, Y. Liu, Z. Wang, and M. Liu, "Extrinsic calibration of lidar and camera with polygon," in *IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2019.
- [11] M. He, H. Zhao, F. Davoine, J. Cui, and H. Zha, "Pairwise lidar calibration using multi-type 3d geometric features in natural scene," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 1828–1835.
- [12] M. He, H. Zhao, J. Cui, and H. Zha, "Calibration method for multiple 2d lidars system," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3034–3041.
- [13] J. Levinson and S. Thrun, "Automatic online calibration of cameras and lasers," in *Robotics: Science and Systems*, vol. 2, 2013.
- [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*. IEEE, 2012, pp. 3096–3102.
- [15] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, "Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information," in *AAAI*, 2012.
- [16] Z. Taylor, "A mutual information approach to automatic calibration of camera and lidar in natural environments," in *Australian Conference on Robotics and Automation*, 2012, pp. 3–5.
- [17] R. Horaud and F. Dornaika, "Hand-eye calibration," *The international journal of robotics research*, vol. 14, no. 3, pp. 195–210, 1995.
- [18] K. Daniilidis, "Hand-eye calibration using dual quaternions," *The International Journal of Robotics Research*, vol. 18, no. 3, pp. 286–298, 1999.
- [19] W. Kabsch, "A discussion of the solution for the best rotation to relate two sets of vectors," *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, vol. 34, no. 5, pp. 827–828, 1978.
- [20] Z. Taylor and J. Nieto, "Motion-based calibration of multimodal sensor arrays," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4843–4850.
- [21] T. Qin and S. Shen, "Online temporal calibration for monocular visual-inertial systems," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3662–3669.
- [22] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.
- [23] H. H. Chen, "A screw motion approach to uniqueness analysis of head-eye geometry," in *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 1991, pp. 145–151.
- [24] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.
- [25] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.
- [26] S. Agarwal, K. Mierle *et al.*, "Ceres solver," 2012.
- [27] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing icp variants on real-world data sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.