3D LiDAR Intrinsic Calibration and Automatic System for LiDAR to Camera Calibration

Jiunn-Kai Huang, Chenxi Feng, Madhav Achar, Maani Ghaffari, and Jessy W. Grizzle

***Abstract*—Periodic intrinsic and extrinsic (re-)calibrations are essential for modern perception and navigation systems deployed on autonomous robots. To date, intrinsic calibration models for LiDARs have been based on hypothesized physical mechanisms for how a spinning LiDAR functions, resulting in anywhere from three to ten parameters to be estimated from data. Instead we propose to abstract away from the physics of a LiDAR type (spinning vs solid state, for example) and focus on the spatial geometry of the point cloud generated by the sensor. This leads to a unifying view of calibration. In experimental data, we show that it outperforms physics-based models for a spinning LiDAR. In simulation, we show how this perspective can be applied to a solid state LiDAR. We complete the paper by reporting on an open-source automatic system for target-based extrinsic calibration from a LiDAR to a camera.**

I. Introduction and Contributions

Camera and LIght Detection And Ranging (LiDAR) are common sensors for current robotics and autonomous systems. For real-world operations, the performance of such systems highly depends on the quality of intrinsic and extrinsic calibration parameters. Moreover, without an automated system for multi-sensor calibration, the scalability of autonomous systems remains low.

Intrinsic calibration of a sensor is the process of ensuring that obtained measurements are meaningful and valid. Intrinsic calibration of a camera is relatively mature and open-sourced packages are available [1]–[3]. In this work, we are concerned with the problem of multi-beam LiDAR intrinsic calibration. We focus on narrow-pulsed time of flight (ToF) method: spinning LiDARs and solid-state LiDARs as shown in Fig. 1. Additionally, Target-based LiDAR-camera extrinsic calibration methods often suffer from manual target extraction and feature association. We develop a fully automatic pipeline for both intrinsic and extrinsic calibration tasks.

A. Spinning LiDAR

A spinning LiDAR as used in this paper consists of K laser emitter-receiver pairs mounted at various positions and elevation angles on the rotating head as shown in Fig. 1. The lasers are pulsed at a fixed frequency. Each revolution of the rotating head sweeps out a cone in space, called a LiDAR scan, composed of discrete return points traced out by each pulsed emitter-receiver. The points associated with a single laser beam are also called a ring.

The measurements are typically viewed as being made in a spherical coordinate system. The range measurement is made by estimating ToF through a clocking system on a circuit board, the azimuth is estimated from the position of the rotating head, and the elevation angle of a beam is determined by its mounting angle on the rotating head.

The authors are with the Robotics Institute, University of Michigan, Ann Arbor, MI 48109, USA. {bjhuang, chenxif, achar, maanigj, grizzle}@umich.edu.

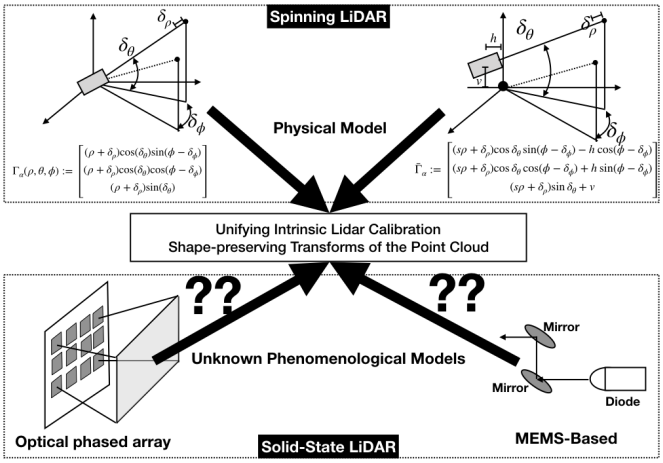

Fig. 1: This figure motivates the main contribution of the paper, a means of calibrating a LiDAR without modeling the physical mechanism itself. Top left: shows a simple physics-based calibration model for a spinning LiDAR. Top right: shows a similar model with more parameters, also for a spinning LiDAR. In the literature, the number of parameters can vary from three to 10. Bottom left and bottom right show, respectively, an optical phased array solid-state LiDAR and a MEMS-based solid-state LiDAR. There are many design options and many manufacturing steps to fabricate these LiDARs . Is it necessary, or even possible, to parameterize all of them? This paper moves the attention to the geometry of the LiDAR's point cloud and proposes that calibration can be achieved via shape-preserving transformations. In experiments, the proposed method outperforms the methods showed in top figures for spinning LiDARs, and via simulation, we support that it may be used to calibrate solid-state LiDARs as well.

Existing error models for LiDARs are phenomenological in nature, that is, they are tied to physical explanations of a sensor's design and/or operation [4]–[11]. Broadly speaking, two sources of uncertainty are typically considered in spinning LiDAR models: (1) errors in the clocking systems, (one for ToF and one for pulsing the lasers) and (2) mechanical errors associated with positioning of the beams on the spinning LiDAR. Some authors overlook that the measurements are actually made in a local coordinate frame for each emitter-receiver pair, and are then transformed to a single "global" frame for the LiDAR unit, resulting in models with three to six parameters [5], [6], [10], while others take this into account and use as many as ten parameters [11].

B. Solid-State LiDAR

In comparison to spinning LiDARs, solid-state LiDARs are smaller, quieter, lower-cost, and produce less vibration. Several

types of solid-state LiDAR are coming on the market, including those based on (1) Optical Phased Array (OPA) [12]–[14], and (2) Micro-Electro-Mechanical System (MEMS) mirrors[1] [15]–[18].

OPA-based solid-state LiDARs function similarly to a phased array in antenna theory. A single OPA contains a significant number of emitters patterned on the chip. By varying the phase shift (i.e., time delay) of each emitter, the direction of the resulting wave-front or the spread angle can be adjusted. In particular, the sensor is able to zoom in or out of an object on command. On the other hand, MEMS-based solid-state LiDARs use only one or a few emitters, with the beams guided by a mirror system regulated by a sophisticated controller. The controller seeks to steer the beam (or beams) in a scanning pattern, such as a zig-zag [15].

To perfectly align OPA emitters or MEMS mirrors is challenging. It is also hard to maintain high accuracy under different temperatures and weather conditions. Thus, how to properly calibrate a solid-state LiDAR is a critical problem.

The variety of solid-state LiDARs raises the question: is it necessary, or even possible, to design/create parameterized models for every type of solid-state LiDAR? This paper seeks to present a unifying view of LiDAR calibration by abstracting away from the physics of a LiDAR type (spinning vs solid state, for example) and focusing instead on the spatial geometry of the point cloud generated by the sensor. This leads to a unifying view of calibration.

C. Our Contributions

1) A new error model for intrinsic LiDAR calibration is proposed based on shape-preserving transformations[2], that is a rigid body transformation plus a scale factor. Currently, intrinsic calibration models only exist for spinning LiDARs. The calibration models are tied to postulated physical mechanisms for how the LiDAR functions, resulting in anywhere from three to ten parameters to be estimated from data. Instead, we *abstract away from the physics of a LiDAR type* (spinning head vs solid state, for example) and focus on the spatial geometry of the point cloud generated by the sensor. *This leads to a unifying view of calibration* and results in an error model that can be represented as the action of the seven-dimensional matrix Lie group, $\mathrm{Sim}(3)$, on subsets of the measurements returned by the LiDAR.
2) We show in simulation that our method can calibrate equally well a spinning LiDAR and a solid-state LiDAR. In experiments on a spinning LiDAR, we show that it works as well or better than two physics-based models.
3) We provide a MATLAB-based simulator for intrinsic LiDAR calibration and validation that allows one to study the effects of target placement and the effectiveness of various calibration methods as a function of "uncertainty models" for a LiDAR.
4) A system for target-based automatic LiDAR to camera extrinsic calibration is given. Specifically, using the back end from [19], a front end is developed that automatically takes in LiDAR and camera data for diamond-shaped planar targets, synchronizes them, extracts the LiDAR payloads and AprilTag [20] corners for the camera images, and then passes the data to the back end that produces the rigid body transformation between the camera and LiDAR.
5) Open-source release of all of the software for intrinsic and extrinsic calibration, including implementations of the baseline methods and the simulator; see [21]–[27].

[1]Some authors argue MEMS-based solid-state LiDARs are not truly solid-state devices due to the movable mirrors; the mirrors, however, are tiny compared to a spinning LiDAR.

[2]Called *similarity transformations* in projective geometry.

II. Proposed LiDAR Intrinsic Calibration Method

Let $\mathcal{PC} = \{\mathbf{x}_i | i = 1, \ldots, M \text{ and } \mathbf{x}_i \in \mathbb{R}^3\}$ denote a LiDAR point cloud associated with a given target. We model the calibration parameter as an element of the similarity transformation Lie group $\mathrm{Sim}(3)$. Such a parameter has seven degrees of freedom and takes into account 3D translation and orientation as well as scaling (a rigid body transformation together with scaling). An element of this group in matrix form is given by

$$\mathbf{H} = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathrm{Sim}(3), \tag{1}$$

where $\mathbf{R} \in \mathrm{SO}(3)$ is the 3D rotation matrix, $\mathbf{t} \in \mathbb{R}^3$ is the translation vector, and $s \in \mathbb{R}^+$ is the scale parameter. Furthermore, the action of $\mathrm{Sim}(3)$ on $\mathbb{R}^3$ is $\mathbf{H} \cdot \mathbf{x} = s\mathbf{R}\mathbf{x} + \mathbf{t}$. We note that this is the most general form of transformation in 3D space while preserving the shape; and therefore, it is suitable for intrinsic calibration of the LiDAR.

The cost associated with a collection of LiDAR returns from a given *planar target* is computed by the point-to-plane (P2P) distance [28], [29]. Let $\mathbf{n}_t$ be a unit normal vector of the target and let $\mathbf{p}_{0,t}$ be a fixed point on the target. The cost is defined as

$$J := \sum_{i=1}^{M} |\mathbf{n}_t^\mathsf{T} \left(\mathbf{x}_i - \mathbf{p}_{0,t}\right)|, \tag{2}$$

where $\mathbf{n}_t^\mathsf{T} \left(\mathbf{x}_i - \mathbf{p}_{0,t}\right)$ is the orthogonal projection of the measurement onto the normal vector and $|\cdot|$ is the absolute value. Then the calibration problem can be formulated as follows.

Problem 1. *For a given collection of points $\mathcal{PC}$, possibly from multiple targets $t \in \{1, \ldots, T\}$, we seek a similarity transformation $\boldsymbol{H}^\star$ that solves*

$$\min_{\boldsymbol{H} \in \mathrm{Sim}(3)} \sum_{t=1}^{T} \sum_{i=1}^{M_t} |\boldsymbol{n}_t^\mathsf{T} \left(\boldsymbol{H} \cdot \boldsymbol{x}_i - \boldsymbol{p}_{0,t}\right)|. \tag{3}$$

Remark 1. In practice each target's normal vector and a point on the target must be estimated from data; see Sec. IV-C.

The specific method for parsing a target's point cloud and the number of calibration transformations depends on the nature of the sensor. For a spinning LiDAR, points on a

target are typically separated by beam numbers so that each beam has its own calibration parameters. For example, for a LiDAR with 32 beams, the set of calibration parameters is $\{\mathbf{H}_k | k = 1, \ldots, 32$ and $\mathbf{H}_k \in \mathrm{Sim}(3)\}$. For an OPA-based solid-state LiDAR, we form an $m \times n$ grid over the planar target and then parse the point cloud based on each point's projection onto the target along the normal of each grid. This way, the required number of calibration transformations becomes $n \cdot m$.

Remark 2. For a single planar target, the $\mathrm{Sim}(3)$ model is over parameterized for a spinning LiDAR when data is parsed by beam number. In fact, the rotation about the target normal $\mathbf{n}_t$, two components of the translation vector $\mathbf{t}$ (translation in the plane of the target) and the scale s are unconstrained. For two planar targets, the translation along the line orthogonal to the targets' normal vectors and the scale are unconstrained.

The following provides a guideline for target placement for calibrating a LiDAR .

Proposition 1. *Consider three targets with normals $\boldsymbol{n}_i$, points on target $\boldsymbol{p}_{0,i}$, and point clouds $\mathcal{PC}_i$. Denote the orthogonal complement of $\boldsymbol{n}_i$ by $\boldsymbol{n}_i^{\perp}$. Assume the set of normal vectors, $\{\boldsymbol{n}_1, \boldsymbol{n}_2, \boldsymbol{n}_3\}$, is linearly independent.*

a) The plane defined by the i-th target is $\mathcal{V}_i := \boldsymbol{p}_{0,i} + \boldsymbol{n}_i^{\perp}$ and there exists a unique point $\boldsymbol{q}_0 \in \mathbb{R}^3$ defined by

$$\boldsymbol{q}_0 := \bigcap_{i=1}^{3} \mathcal{V}_i. \tag{4}$$

b) For $\boldsymbol{H} \in \mathrm{SE}(3)$ there exists a unique solution.

c) For $\boldsymbol{H} \in \mathrm{Sim}(3)$ the scale s is unconstrained.

d) Suppose $\mathcal{L}_i$ is a line formed by a beam on the i-th target. If all of the lines $\mathcal{L}_i$, $1 \leq i \leq 3$, pass through the point $\boldsymbol{q}_0$, then the solution is not unique.

If the condition in parts (d) holds, then the scale factor, s, is not unique; see Fig. 2.

Corollary 2. *Consider a scene with four targets. Problem 1 is non-degenerate if for any combination of three targets, the corresponding normals are linearly independent.*

III. Baseline Intrinsic Calibration Methods

There are two standard calibration models for spinning LiDARs of the type. They are based on spherical coordinates (ρ, θ, ϕ), referred to as range, elevation and azimuth,

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = f(\rho, \theta, \phi) = \begin{bmatrix} \rho \cos\theta \sin\phi \\ \rho \cos\theta \cos\phi \\ \rho \sin\theta \end{bmatrix}, \tag{5}$$

and the inverse function,

$$\begin{bmatrix} \rho \\ \theta \\ \phi \end{bmatrix} = f^{-1}(x, y, z) = \begin{bmatrix} \sqrt{x^2 + y^2 + z^2} \\ \sin^{-1}\left(\frac{z}{\sqrt{x^2+y^2+z^2}}\right) \\ \mathrm{atan2}(x, y) \end{bmatrix}. \tag{6}$$

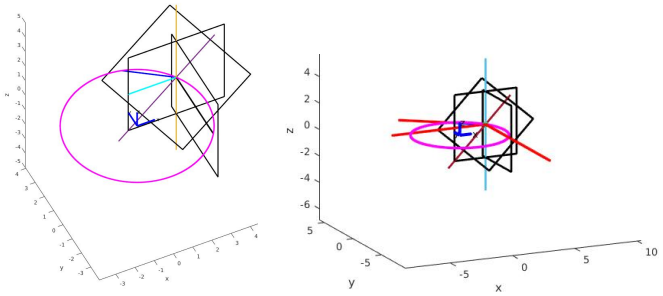

Fig. 2: This figure illustrates a degenerate case. The black squares are different planes with the normal vectors being linearly independent. The magenta circle shows LiDAR ring passing through q_0 in (4). The blue, red, black lines are the intersections of the ring the planes. On the right figure shows after applying a transformation $\mathbf{H} = \begin{bmatrix} s\mathbf{I} & (1-s)\mathbf{q}_0 \\ \mathbf{0} & 1 \end{bmatrix}$, the cost remains zero because the transformed lines (in red) overwrite the original blue, red, and black lines.

A. 3-Parameter Model

The most basic model [5] assumes the LiDAR measurements are made in spherical coordinates, (ρ, θ, ϕ). Corrections to a measurement are given by a collection offsets $\alpha := (\delta_\rho, \delta_\theta, \delta_\phi)$. Expressing the calibrated measurement in Cartesian coordinates gives

$$\Gamma_\alpha(\rho, \theta, \phi) := \begin{bmatrix} (\rho + \delta_\rho)\cos(\delta_\theta)\sin(\phi - \delta_\phi) \\ (\rho + \delta_\rho)\cos(\delta_\theta)\cos(\phi - \delta_\phi) \\ (\rho + \delta_\rho)\sin(\delta_\theta) \end{bmatrix} \tag{7}$$

Remark 3. **(a)** The nominal elevation for each ring is taken as zero;, i.e., $\theta = 0$. **(b)** While it is not necessarily a drawback, most LiDAR interfaces only return a Cartesian representation of a measured point. Hence, the user must assure the transformation to spherical coordinates, make the measurement correction coming from the calibration, and then transform back to Cartesian coordinates.

B. 6-Parameter Model

This model [6], [7] also works in spherical coordinates. In addition to the three offsets above, it includes s, a scale factor of ρ, h, a horizontal offset of the origin, and v, a vertical offset for the origin. The correction model becomes

$$\bar{\Gamma}_\alpha := \begin{bmatrix} (s\rho + \delta_\rho)\cos\delta_\theta \sin(\phi - \delta_\phi) - h\cos(\phi - \delta_\phi) \\ (s\rho + \delta_\rho)\cos\delta_\theta \cos(\phi - \delta_\phi) + h\sin(\phi - \delta_\phi) \\ (s\rho + \delta_\rho)\sin\delta_\theta + v \end{bmatrix}. \tag{8}$$

and $\alpha := (\delta_\rho, \delta_\theta, \delta_\phi, s, h, v)$.

C. Can these be expressed as transformations by similar matrices?

The short answer is no. When viewed in Cartesian coordinates, the calibration model (8) is fundamentally nonlinear and can be expressed as $\mathbf{R}_1(\mathbf{R}_2\mathbf{t}_1 + \mathbf{t}_2)$, where $\mathbf{R}_1, \mathbf{R}_2, \mathbf{t}_1$ and $\mathbf{t}_2$ depend on the measured point,

$$\mathbf{x}^{\mathsf{T}} = [x, y, z] \xrightarrow{f^{-1}} (\rho, \theta, \phi),$$

$$\mathbf{R}_1 = \begin{bmatrix} \sin(\phi - \delta_\phi) & -\cos(\phi - \delta_\phi) & 0 \\ \cos(\phi - \delta_\phi) & \sin(\phi - \delta_\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{9}$$

$$\mathbf{R}_2\mathbf{t}_1 + \mathbf{t}_2 = \begin{bmatrix} \cos(\delta_\theta) & 0 & -\sin(\delta_\theta) \\ 0 & 1 & 0 \\ \sin(\delta_\theta) & 0 & \cos(\delta_\theta) \end{bmatrix} \begin{bmatrix} s\rho + \delta_\rho \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ h \\ v \end{bmatrix}.$$

The calibration first applies a corrected elevation angle to a range value that has been offset and scaled. Next, the (y, z) position of the origin is offset, and lastly, a corrected rotation about the z-axis is applied.

D. Cost function for baseline models

For a given collection of points $\mathcal{PC}$, possibly from multiple targets $t \in \{1, \ldots, T\}$, we seek calibration parameters that solve

$$\min_{\alpha} \sum_{t=1}^{T} \sum_{i=1}^{M_t} |\mathbf{n}_t^{\mathsf{T}} \left(F(\mathbf{x}_i, \alpha) - \mathbf{p}_{0,t} \right)|, \tag{10}$$

where

1) for baseline model $\mathbf{BL}_1$ [5] in (7), $\alpha = (\delta_\rho, \delta_\theta, \delta_\phi)$ and

$$F(\mathbf{x}, \alpha) := \Gamma_\alpha \circ f^{-1}(\mathbf{x});$$

2) for baseline model $\mathbf{BL}_2$ [6], [7] in (8), $\alpha = (\delta_\rho, \delta_\theta, \delta_\phi, s, h, v)$ and

$$F(\mathbf{x}, \alpha) := \bar{\Gamma}_\alpha \circ f^{-1}(\mathbf{x})$$

Remark 4. For the 3-parameter model $\mathbf{BL}_1$, a single planar target is sufficient to uniquely determine a calibration. For the 6-parameter model $\mathbf{BL}_2$, non-uniqueness can occur as outlined in Proposition 1 and Fig. 2.

IV. Automatic Calibration System

Target-based LiDAR-camera extrinsic calibration methods often suffer from manual target extraction and feature association. The only manual process in our previous software release [19], determining the image corners, has now been made automatic. Because experimentally comparing and validating the intrinsic calibration methods of Sec. II and III would require significant data collection, we decided to create a fully automatic pipeline for both intrinsic and extrinsic calibration, as shown in Fig 3. We demonstrate the capability of this pipeline in the experimental section by collecting a significant amount of data allowing us to check the quality of the calibration of our sensor suite (a LiDAR-camera pair) at different distances; see Sec. VI-B.

A. Front-End

Fiducial markers are widely used in vision-based perception tasks because they can be easily identified and automatically extracted from the background. Inspired by AprilTags [20], [31], [32], we introduced a similar concept for LiDAR point clouds, called a LiDARTag, consisting of a payload and a pattern that makes each marker uniquely distinguishable and identifiable in real-time by both a LiDAR and a camera [26]. An open-source package for processing LiDARTags is available [23], while AprilTag3 is available as a ROS package [24].

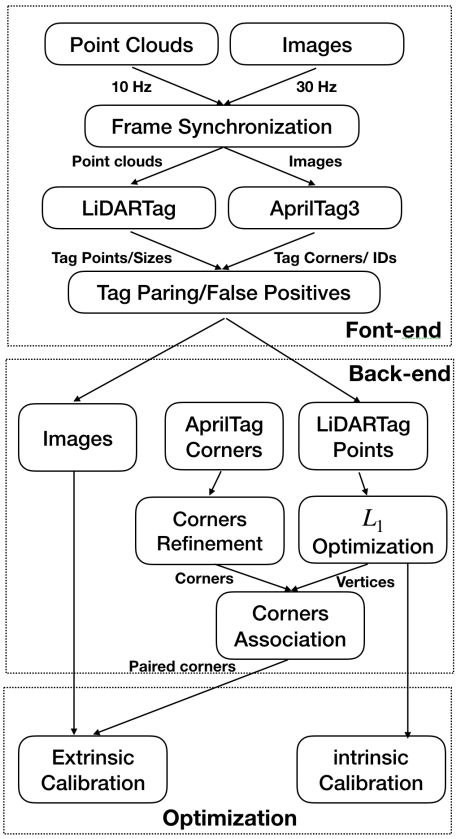

Fig. 3: A system diagram for automatic intrinsic and extrinsic calibration. The top shows the front-end of the pipeline. Its input is raw camera and LiDAR data, which are subsequently synchronized. The AprilTag and LiDARTag packages are used to extract the target information, which is then examined to remove outliers and ensure that targets are still properly synchronized. Each run of the front-end saves all related information into a ROS bagfile, which can then be sent to the back-end for further processing. The back-end takes in (possibly many) ROS bagfiles and does the following: (i) refines the image corners; and (ii) extracts vertices of the LiDAR targets. The correspondences are established between corners and vertices and an extrinsic transformation is determined PnP as in [30]. For intrinsic calibration, the resulting vertices of the LiDAR targets are used to extract normal vectors and a point on the plane. The calibration parameters are determined to minimize the P2P distance from the plane to the target points provided by the LiDARTag package.

As LiDAR and camera data streams arrive, they are synchronized, examined for fiducial markers, and checked for false positives. For LiDAR-camera extrinsic calibration, we use target vertices and image corners as features. The relevant data (images, AprilTag corners, points on LiDARTags) are saved as ROS bagfiles for processing in MATLAB. Technical details and software are available at [25].

B. Back-End

The back-end begins by estimating the LiDARTag vertices, which, due to sparsity of the LiDAR point cloud, are not directly observable. We use the method developed in [19, Sec. II] to determine the vertices of a diamond-shaped target. A rigid-body transformation is determined that "best fits" the

target's LiDAR points to the target's known geometry. Best fit is defined in an L_1 sense.

From the vertices, the target normal $\mathbf{n}_t$ and point on the target, $\mathbf{p}_{0,t}$, can be easily determined for evaluating the P2P cost function (3) for intrinsic calibration, while the vertices in combination with the camera corners are used to determine the extrinsic calibration via a PnP problem.

The corners obtained from the AprilTag package are automatically refined using a process detailed in [19, Sec.V-C]. Given the camera corners, denoted ($\{Y_i\}_{i=1}^4$), and LiDAR vertices, denoted $\{X_i\}_{i=1}^4$, we use the vertical and horizontal positions in their own coordinates to sort them and establish the correspondences.

C. Intrinsic and Extrinsic Calibration

For extrinsic calibration, the rigid body transformation between LiDAR to camera is found by solving a Perspective-n-Point (PnP) problem as in [30], namely

$$(\mathbf{R}_L^{C*}, \mathbf{t}_L^{C*}) = \arg\min_{(\mathbf{R},\mathbf{t})} \text{dist}(P(\mathbf{R}X_i + \mathbf{t}), Y_i), \qquad (11)$$

where X_i are coordinates of the LiDAR vertices, Y_i are the coordinates of the camera corners, and P is the standard 3D$\rightarrow$2D camera "projection map", based in its intrinsic parameters. Furthermore, $\mathbf{R}_L^C, \mathbf{t}_L^C$ is the LiDAR-frame to camera-frame rigid body transformation, and dist is a distance or error measure (typically L_2). For example results, see [19].

With the method in [19], the target vertices are automatically co-planar and hence uniquely define a unit normal vector and a point on the target. The P2P distance is minimized for each of the three calibration models, **BL**1, **BL**2, and $\text{Sim}(3)$, for a collection of targets, yielding α^* (for **BL**1 and **BL**2) and $\mathbf{H}_*$ for $\text{Sim}(3)$.

For experimental data, the process of intrinsic calibration involves iterating the estimation of the target vertices and the parameters in the LiDAR model until a given stopping criterion is met. Here, we stopped when the maximum change δ_m over the set of target vertices was less than $1e-5$.

while $\delta_m > 10^{-5}$ **do**

Step 1: Re-estimate the target vertices $\{X_i\}_{i=1}^4$ after applying the current intrinsic calibration parameters to the point cloud. This gives updated values for the target normals and points on the targets.

Step 2: Using the updated target vertices, re-estimate the intrinsic calibration parameters for the three calibration models.

end while

For simulation data, no iteration is required as the target information is known from "ground truth".

V. Simulation Results

This section presents a simulation study of the Intrinsic Calibration Problem that will show the importance of properly constructed scenes for eliminating over parameterization in two of the three calibration models. Results here will be used to inform the experimental work in the next section.

A. Simulated LiDAR Environment

Due to lack of ground truth, we build a LiDAR simulator from scratch to compare our calibration method against two baselines and to illustrate the role of target positioning in the construction of a calibration environment. As an example of the latter, Fig. 2 illustrates a scene where a unique solution does not exist for two of the three calibration models, namely, (3) and (8). Our simulator can model LiDAR sensors of different working principles (spinning vs solid-state). With our simulator, it is easy to control sources of uncertainty, including both mechanical model uncertainty and measurement noise. Targets are assumed to be planar and polygonal.

To simulate 3D points on a planar target, we generate rays from the LiDAR sensor, and define a "target" point as the point at which the ray intersects the target. The simulator has an option to account for shadowing or not. After locating the exact LiDAR returns on the target, based on the LiDAR type, we then add two types of uncertainty to the returns and report them as measured data.

Remark 5. The simulator first finds all intersection points with the (infinite) plane defined by the target. To determine if a point on the plane is within the boundary of the target polygon is a famous problem in computer graphics, called the Point-In-Polygon (PIP) problem. The Winding Number algorithm [33] is implemented in this simulator. If the winding number of a point is not zero, the point lies inside the boundary; otherwise it is outside.

B. Calibration Scenes

To illustrate Prop. 1 and the role of target selection and placement in the case of a spinning LiDAR, we define five different scenes, where four will be used for "training" and one will be used for "validation". Because the calibration is being done for a spinning LiDAR, the points are parsed "per ring" as discussed in Sec. II. The target sizes vary with distance to ensure a minimum of ten points per ring on each target.

- Scene 1 : Three targets are placed along the same side of x-axis, parallel to each other and to the z-axis
- Scene 2: Two targets are placed at different distance and opposite sides along the x-axis of the LiDAR, parallel to one another and to the z-axis
- Scene 3: Four targets are placed at right angles. This models the use the common use of vertical walls for calibration.
- Scene 4 : a complex scene for validation. eight targets are placed at various distance and orientations. Four of them are close the LiDAR and the other four are far from LiDAR.
- Scene 5 : We place 12 targets at different distances with various angles.
- Scene 6 : We place 24 targets at different distances with various angles. To show that our method can still get a good result in this complex scene

Remark 6. The placement of only a few targets in a scene is done to demonstrate the "minimum" requirements to calibrate

TABLE I: Induce 3-parameter perturbation model ($\mathbf{N}_1$), 6-parameter perturbation model ($\mathbf{N}_2$), and $\mathrm{Sim}(3)$ noise model ($\mathbf{N}_3$) for a spinning LiDAR in the simulator.

Noise Model	Methods	scene 1	scene 2	scene 3	scene 4	scene 5	scene 6
N1	Baseline1	0.036998	0.037076	0.034804	0.032165	0.017921	0.028471
	Baseline2	0.16108	0.15412	0.15745	0.035424	0.023221	0.023275
	Our Method	**0.020421**	**0.028072**	**0.013583**	**0.010428**	**0.0016329**	**0.0010792**
N2	Baseline1	0.024689	0.025205	0.027227	0.021967	0.010316	0.012955
	Baseline2	0.14897	0.14993	0.14192	0.033905	0.012305	0.0086185
	Our Method	**0.015302**	**0.018722**	**0.012405**	**0.01152**	**0.003471**	**0.00076376**
N3	Baseline1	0.037758	0.037428	0.038231	0.033412	0.023682	0.031802
	Baseline2	0.16082	0.15571	0.15756	0.04164	0.022272	0.024862
	Our Method	**0.02297**	**0.025774**	**0.012842**	**0.013986**	**0.0053697**	**0.00010772**

a spinning LiDAR. One can place as many targets as desired in the simulator.

C. LiDAR Intrinsic Calibration of a Spinning LiDAR

For each scene, we generate deterministic measurements by applying the following perturbations to the data on a per-ring basis (i.e, values vary by ring number): (i) simple model $(\delta_\rho, \delta_\theta, \delta_\phi)$, (ii) complex model $(\delta_\rho, \delta_\theta, \delta_\phi, s, h, v)$, and (iii) $\mathrm{Sim}(3)$ model $(\mathbf{R}, \mathbf{t}, s)$, denoted as $\mathbf{N}_1, \mathbf{N}_2, \mathbf{N}_3$, respectively. The exact values used are in a `.mat` file uploaded to GitHub [21]. For each target, its true normal vector $\mathbf{n}$ and point $\mathbf{p}_0$ on the target are assumed known so that the P2P cost functions can be evaluated. For each of the three cost functions (3), (7), (8), the `fmincon` function in MATLAB is used to determine calibration parameters achieving a local minimum for each Scene. The initializations for `fmincon` are selected to correspond to zero perturbation error, that is the parameters are zero, one or the identify matrix as the case requires. The optimized calibration parameters are then applied to the data of Scene 5 and the P2P cost of (3) is evaluated and reported as validation data in Table I.

The simulated results confirm that a single target results in degeneracy when calibrating with the $\mathbf{BL}_2$ and $\mathrm{Sim}(3)$ models, and while $\mathbf{BL}_1$ works fine with a single target, it drops in performance as the complexity of the calibration errors is increased. The well-posedness of the $\mathbf{BL}_2$ and $\mathrm{Sim}(3)$ calibration models was confirmed when the conditions of Prop. 1 are respected.

D. LiDAR Intrinsic Calibration of a Solid State LiDAR

An OPA solid-state LiDAR is fabricated on a planar wafer with a large number of emitters, see Sec. I. We induce geometric uncertainty into the system by assuming the plane of the wafer is slightly warped. In the simulator, the OPA solid-state LiDAR has 20×20 emitters with 160, 40 horizontal and vertical field of view, respectively. We place two targets of the same size. As proposed in Sec. II, we parse the LiDAR returns according to an 8×8 uniform grid over the planar targets (indicated with different colors), for a total of 64 calibration models. For each grid, the cost function (3) is optimized using the `fmincon` function in MATLAB to determine calibration parameters achieving a local minimum. The calibrated sensor is then validated on the other three scenes that have different angles and distances as shown in Fig. 4; the mean P2P cost of (3) is evaluated and reported as validation data in Table II.

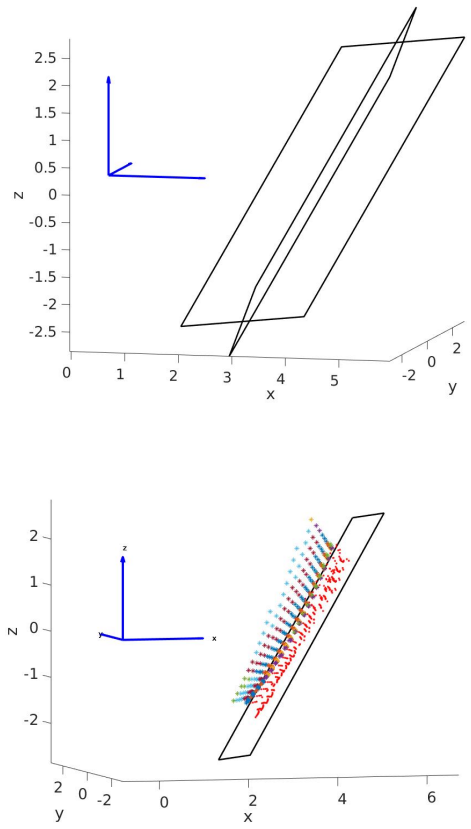

Fig. 4: Left figure shows the placement of the validation scene. On the right shows one of the three validation results for OPA solid-state LiDAR. The red dots indicates the calibrated solid-state LiDAR returns and the "warped" curvature is the noisy measurement.

VI. Experimental Intrinsic and Extrinsic Calibration for a Spinning LiDAR

This section reports on experimental data from a *32-Beam Velodyne ULTRA Puck LiDAR* and an *Intel RealSense Depth Camera D435*, mounted on an in-house designed torso for a Cassie-series bipedal robot [34]. We use the automatic system Fig. 3 for both intrinsic calibration and extrinsic calibration.

TABLE II: Geometric uncertainty of a solid-state LiDAR by assuming the plane of the wafer is slightly warped.

Solid-State LiDAR	Before Calibration	After Calibration	Improvement
Training results	0.3315	0.0029	99.12 %
Validation Scene1	0.2659	0.0197	92.59 %
Validation Scene2	0.3193	0.0323	89.89 %
Validation Scene3	0.2958	0.0202	93.16 %

TABLE III: Experiments data collected using different numbers of targets in the calibration scene.

No.Tags	4	8	12	24
Baseline1	**0.0064**	**0.0066**	0.0066	0.0068
Baseline2	0.0146	0.0103	0.0083	0.0082
Our Method	0.0201	0.0084	**0.0062**	**0.0052**

A. Automatic Intrinsic Calibration

Informed by the simulation study, a set of LiDAR data was collected using the LiDARTag package [23] with targets arranged to avoid the degeneracies indicated in Prop. 1. All together, 39 targets were collected as potential training data. On a separate day, an additional 12 targets were collected for validation. Table III reports validation data for the three calibration methods on various subsets of the targets based on number of targets: five, seven, 10 of them.

It is seen that the trends observed in the simulation data for a spinning LiDAR continue to hold in experiments. We conclude that the proposed idea of reformulating the calibration problem in terms of shape-preserving transformations on the observed point cloud has been supported (i.e., not invalidated). With the open-source release of our code and data, it will be straightforward for other groups to confirm or deny this finding.

B. Automatic Extrinsic Calibration System

We collected 16 different ROS bagfiles of extrinsic calibration data. Each scene includes two targets, one approximately 120 cm square and the other approximately 80.5 cm square, with the smaller target placed closer to the camera-LiDAR pair. The front-end of the system Fig. 3 is used to collect and pre-process 10 s of data for each scene, resulting in an approximate total of 1600 pairs of LiDAR scans and camera images. All of the relevant information is saved in ROS bagfiles. The back-end of the system takes in the bagfiles, and post-processes the data, resulting in roughly 1200 pairs of LiDAR and camera corners. The rigid-body transformation from LiDAR to camera is "trained" on the eight of the sixteen scenes from both targets and then "validated" on the remaining scene. Fig. 5 shows the root mean square error [19, Sec.V-E] of the validation results as a function of distance to a target.

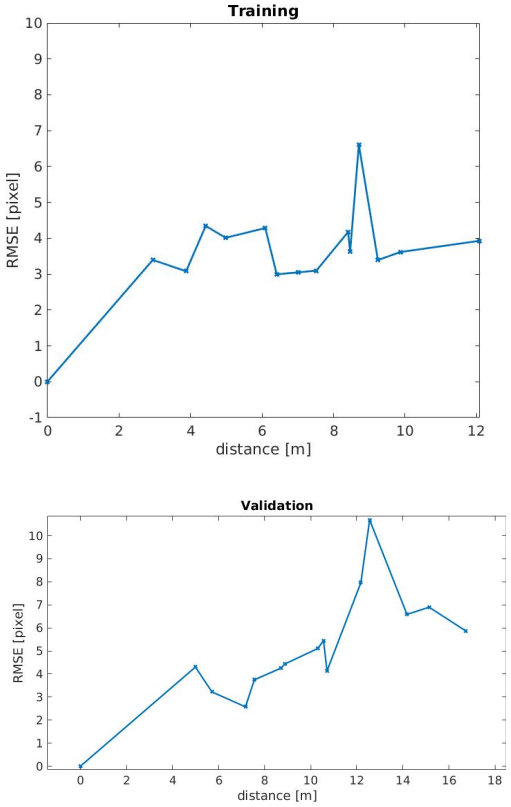

Fig. 5: A plot of RMS error [19, Sec.V-E] of the validation results as a function of distance. The extrinsic parameters are determined from roughly 600 pairs of LiDAR vertices and camera corners, and validated on another set of 600 pairs of LiDAR and camera correspondences.

VII. Conclusions

We developed a novel LiDAR intrinsic calibration method that uses the spatial geometry of the point cloud generated by the sensor. The calibration parameter was modeled using $\mathrm{Sim}(3)$ Lie group and nicely generalized previous parametrizations into a unifying problem. In experiments showed that the proposed method can serve as a generic model for intrinsic calibration of both spinning and solid-state LiDARs. We also provided an analysis that showed, when solving for a transformation in $\mathrm{Sim}(3)$, a minimum of four targets are required in the scene.

Yang et al. [35] solves a related minimization problem for 3D registration with outliers. The work reformulate the 3D point cloud registration problem using a truncated least squares and solves it via semi-definite relaxation. In the future, it is hoped that something similar can be done for the problem discussed in this work. As an alternative future research direction, Clark et al. [36] solves the registration problem using continuous functions in a reproducing kernel Hilbert space. This approach is data association free and inherently robust to the input noise. In this framework, while the solutions are local, it is possible to make the problem convex by continuously modifying the kernel length-scale [36, See Fig. 9].

Acknowledgment

Funding for this work was provided by the Toyota Research Institute (TRI) under award number N021515. Funding for J. Grizzle was in part provided by TRI and in part by NSF Award No. 1808051. The first author thanks Wonhui Kim for useful conversations.

REFERENCES

[1] J. Maye, P. Furgale, and R. Siegwart, "Self-supervised calibration for robotic systems," in *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, pp. 473–480.

[2] Itseez, "Open source computer vision library," https://github.com/itseez/opencv, 2015.

[3] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, 2009.

[4] F. M. Mirzaei, D. G. Kottas, and S. I. Roumeliotis, "3d lidar–camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization," *The International Journal of Robotics Research*, vol. 31, no. 4, pp. 452–467, 2012.

[5] G. Pandey, J. McBride, S. Savarese, and R. Eustice, "Extrinsic calibration of a 3d laser scanner and an omnidirectional camera," *IFAC Proceedings Volumes*, vol. 43, no. 16, pp. 336–341, 2010.

[6] C. Glennie and D. D. Lichti, "Static calibration and analysis of the velodyne hdl-64e s2 for high accuracy mobile scanning," *Remote sensing*, vol. 2, no. 6, pp. 1610–1624, 2010.

[7] H. Nouira, J.-E. Deschaud, and F. Goulette, "Point cloud refinement with a target-free intrinsic calibration of a mobile multi-beam lidar system," 2016.

[8] T. Chan, D. D. Lichti, and D. Belton, "Temporal analysis and automatic calibration of the velodyne hdl-32e lidar system," *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci*, vol. 2, pp. 61–66, 2013.

[9] G. Atanacio-Jiménez, J.-J. González-Barbosa, J. B. Hurtado-Ramos, F. J. Ornelas-Rodríguez, H. Jiménez-Hernández, T. García-Ramirez, and R. González-Barbosa, "Lidar velodyne hdl-64e calibration using pattern planes," *International Journal of Advanced Robotic Systems*, vol. 8, no. 5, p. 59, 2011.

[10] N. Muhammad and S. Lacroix, "Calibration of a rotating multi-beam LIDAR," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 5648–5653.

[11] R. Bergelt, O. Khan, and W. Hardt, "Improving the intrinsic calibration of a velodyne lidar sensor," in *2017 IEEE SENSORS*. IEEE, 2017, pp. 1–3.

[12] A. Frederiksen and S. Hartmann, "Lidar device including at least one diffuser element," Feb. 6 2020, uS Patent App. 16/525,010.

[13] C. V. Poulton, M. J. Byrd, P. Russo, E. Timurdogan, M. Khandaker, D. Vermeulen, and M. R. Watts, "Long-range lidar and free-space data communication with high-performance optical phased arrays," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 25, no. 5, pp. 1–8, 2019.

[14] C. V. Poulton, P. Russo, E. Timurdogan, M. Whitson, M. J. Byrd, E. Hosseini, B. Moss, Z. Su, D. Vermeulen, and M. R. Watts, "High-performance integrated optical phased arrays for chip-scale beam steering and lidar," in *CLEO: Applications and Technology*. Optical Society of America, 2018, pp. ATu3R–2.

[15] H. W. Yoo, N. Druml, D. Brunner, C. Schwarzl, T. Thurner, M. Hennecke, and G. Schitter, "Mems-based lidar for autonomous driving," *e & i Elektrotechnik und Informationstechnik*, vol. 135, no. 6, pp. 408–415, 2018.

[16] G. Pelz and N. Elbel, "Matrix light source and detector device for solid-state lidar," Feb. 6 2020, uS Patent App. 16/052,862.

[17] N. Druml, I. Maksymova, T. Thurner, D. van Lierop, M. Hennecke, and A. Foroutan, "1d mems micro-scanning lidar," in *Conference on Sensor Device Technologies and Applications (SENSORDEVICES)*, vol. 9, 2018.

[18] X. Lee and C. Wang, "Optical design for uniform scanning in mems-based 3d imaging lidar," *Applied optics*, vol. 54, no. 9, pp. 2219–2223, 2015.

[19] J.-K. Huang and J. W. Grizzle, "Improvements to target-based 3d lidar to camera calibration," *arXiv preprint arXiv:1910.03126*, 2019.

[20] J. Wang and E. Olson, "Apriltag 2: Efficient and robust fiducial detection," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4193–4198.

[21] J.K. Huang, C Feng, M. Achar, M. Ghaffari and Jessy W. Grizzle, "Intrinsic LiDAR Calibration," 2019. [Online]. Available: https://github.com/UMich-BipedLab/LiDAR_intrinsic_calibration

[22] J.K. Huang, C Feng and Jessy W. Grizzle, "LiDAR Simulator Package," 2020. [Online]. Available: https://github.com/UMich-BipedLab/lidar_simulator

[23] J.K. Huang, C Feng, M. Achar, M. Ghaffari and Jessy W. Grizzle, "LiDARTag ROS Package," 2020. [Online]. Available: https://github.com/UMich-BipedLab/LiDARTag

[24] J.K. Huang, C Feng and Jessy W. Grizzle, "AprilTag ROS Package," 2020. [Online]. Available: https://github.com/UMich-BipedLab/AprilTag_ROS

[25] J.K. Huang, C Feng, M. Achar, M. Ghaffari and Jessy W. Grizzle, "Automatic Extrinsic LiDAR Camera Calibration," 2020. [Online]. Available: https://github.com/UMich-BipedLab/automatic_lidar_camera_calibration

[26] J.-K. Huang, M. Ghaffari, R. Hartley, L. Gan, R. M. Eustice, and J. W. Grizzle, "Lidartag: A real-time fiducial tag using point clouds," *arXiv preprint arXiv:1908.10349*, 2019.

[27] J.K. Huang and Jessy W. Grizzle, "Extrinsic LiDAR Camera Calibration," 2019. [Online]. Available: https://github.com/UMich-BipedLab/extrinsic_lidar_camera_calibration

[28] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 1991, pp. 2724–2729.

[29] P. Georgiades, "Signed distance from point to plane," in *Graphics Gems III (IBM Version)*. Elsevier, 1992, pp. 223–224.

[30] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, p. 155, 2009.

[31] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3400–3407.

[32] M. Krogius, A. Haggenmiller, and E. Olson, "Flexible layouts for fiducial tags," 2018.

[33] D. Alciatore and R. Miranda, "A winding number and point-in-polygon algorithm," *Glaxo Virtual Anatomy Project Research Report, Department of Mechanical Engineering, Colorado State University*, 1995.

[34] J. Huang. (2019) Cassie Blue walks Around the Wavefield. https://youtu.be/LhFC45jweFMc.

[35] H. Yang, J. Shi, and L. Carlone, "TEASER: Fast and certifiable point cloud registration," 2020. [Online]. Available: https://github.com/MIT-SPARK/TEASER-plusplus

[36] W. Clark, M. Ghaffari, and A. Bloch, "Nonparametric continuous sensor registration," *arXiv preprint arXiv:2001.04286*, 2020. [Online]. Available: https://github.com/MaaniGhaffari/c-sensor-registration