

Single Frame Lidar-Camera Calibration Using Registration of 3D Planes

Ashutosh Singandhupe, Hung Manh La *IEEE Senior Member*, Quang Phuc Ha *IEEE Senior Member*

Abstract—This work focuses on finding the extrinsic parameters (rotation and translation) between Lidar and an RGB camera sensor. We use a planar checkerboard and place it inside the Field-of-View (FOV) of both sensors, where we extract the 3D plane information of the checkerboard acquired from the sensor's data. The plane coefficients extracted from the sensor's data are used to construct a well-structured set of 3D points. These 3D points are then 'aligned,' which gives the relative transformation between the two sensors. We use our proposed Correntropy Similarity Matrix Iterative Closest Point (CoSM-ICP) Algorithm to estimate the relative transformation. This work uses a single frame of the point cloud data acquired from the Lidar sensor and a single frame from the calibrated camera data to perform this operation. From the camera image, we use the projection of the calibration target's corner points to compute the 3D points, and along the process, we calculate the 3D plane equation using the corner points. We evaluate our approach on a simulated dataset with complex environment settings, making use of the freedom to assess under multiple configurations. Through the obtained results, we verify our method under various configurations.

I. INTRODUCTION

Multi-sensor autonomous systems are generally designed on a predefined setup where the sensors are placed at known locations relative to each other [1]. Since the sensors' relative translation and rotation components are already known, it does not necessarily involve any calibration. However, modern autonomous systems with unique designs and sensors placed at different configurations demand automatic and efficient calibration methods for multi-sensor setups [2], [3]. Calibration in a multi-sensor system is essentially to find the relative transformation between the sensors. With our focus on autonomous navigation, lidar and camera configurations are explicitly designed to the requirement of the system and need to be calibrated (finding the relative rotation and translation between the two sensors) for efficient autonomous navigation [4]–[9]. This task is particularly important in many field robotics autonomous systems, e.g. in defense [10], [11], robotic manipulation [12]–[15], intelligent transportation systems [16], [17], object tracking [18], [19], civil infrastructure inspection [20]–[24], construction and mining [25]–[27] sectors.

This work is supported by the Vingroup Joint Stock Company and supported by Vingroup Innovation Foundation (VINIF) under project code VINIF.2020.NCUD.DA094. The views, opinions, findings, and conclusions reflected in this publication are solely those of the authors and do not represent the official policy or position of the VINIF.

Ashutosh Singandhupe and Dr. Hung La are with the Advanced Robotics and Automation (ARA) Lab, Department of Computer Science and Engineering, University of Nevada, Reno, NV 89557, USA. Dr. Quang Ha is with the School of Electrical and Data Engineering, University of Technology Sydney, Australia. Corresponding author: Hung La, email: hla@unr.edu.

The source code of our implementation is available at the ARA lab's github: <https://github.com/aratlab-unr/Lidar-Cam-Calib>

Lidar and camera calibration is a well-studied problem, and most of the previous approaches require calibration cards or some well-defined calibration objects [4], [28]–[37]. Figure 1 shows a simple lidar camera configuration setup. A calibration card is placed in the Field-of-View (FOV) of both the sensors. Extracting features from a checkerboard is convenient, making a planar checkerboard one of the most widely used calibration objects. Most of these approaches have emphasized placing the calibration objects in the FOV of both the sensors. Finding the correspondences between the lidar and the camera data points is crucial for efficient lidar and camera calibration [38]. The correspondences are calculated either manually or automatically using feature extraction algorithms. The accurate estimation of the correspondences is essential for efficient calibration.

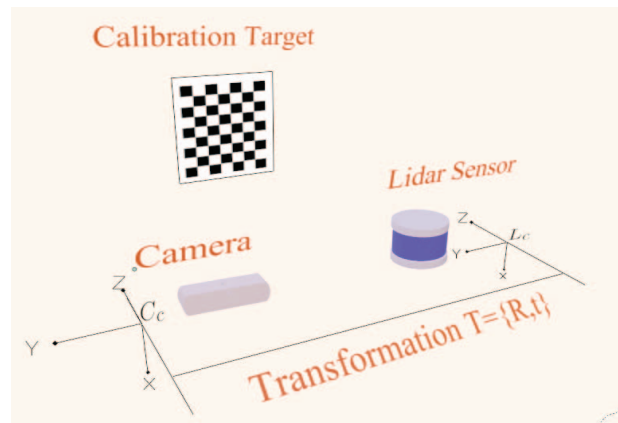


Fig. 1: Sample lidar and camera configuration setup, a stereo camera setup, with only the images from the left camera being used for our work: C_c is the camera coordinate frame (here we consider the left camera center of the stereo camera's coordinate frame), and L_c is the lidar coordinate frame. The objective is to compute the transformation T between L_c and C_c .

The earlier method proposed by Zhang and Pless [28] aimed to calibrate a 2D laser and a monocular camera, and they used a planar checkerboard pattern to achieve that. Their approach involved placing the checkerboard in front of the 2D laser and the camera with different poses. The relative transformation was calculated by minimizing the re-projection error of features of the checkerboard as viewed by the 2D lidar and the camera. Their algorithm required at least five poses, whereas, in theory, the extrinsic calibration could be achieved in 3 poses. Unnikrishnan and Hebert [39] extended this method to a 3D laser, in

which they estimated the plane parameters from the 3D lidar and the camera data. In their approach, the initial rotation and translation were calculated using plane-plane correspondence and later refined by minimizing the point to plane distance. Another work proposed by Geiger et al. [40], placed several checkerboards on the scene where their proposed algorithm detected the checkerboards and matched them with the planes from the laser data. Since there were several checkerboards in the scene, one shot of the scene could be enough. Most of these methods involve the use of plane information of the checkerboard and do not consider the boundary of the checkerboard. Work by [41] requires fewer poses of the calibration target and has shown accurate results. Therein, the checkerboard was placed closer to the sensor, allowing the laser point to approximate the boundary more accurately, and hence it showed high accuracy. The authors introduced a similarity transformation between the lidar and the camera, which showed better results than using rigid transformation.

Apart from the planar checkerboard, researchers also used several other calibration objects for the calibration method [42]–[45]. In [31], with a polygonal board, the calibration parameters were estimated using the vertices of the board. However, in the camera frame, it involved manual labeling of the correspondences. These correspondences were used to estimate the calibration parameters with a genetic algorithm-based approach. Other approaches [32]–[34] have used planar boards with rectangular holes, planar boards with circular holes, and other 3D objects. One of the drawbacks of the above methods is that they are all laborious and time-consuming.

In addition to the approaches mentioned above, several target-less or no object-based calibration techniques have been proposed as well [46]–[48]. Initial work by Scaramuzza [49] has introduced the use of natural scenes when calibrating a 3D laser and an omnidirectional camera. The features were extracted automatically from the sensor data based on their algorithm, and they manually selected the correspondence between the features. Another work by Moghadam [50] exploited the linear features present in the indoor environment. It used the 3D line features from the point cloud and the corresponding 2D line segment from the camera to estimate the rigid body transformation between the two. Boughorbal [51] proposed to maximize the correlation between the sensor data using a chi-square test. This technique was based on the statistical dependence of the data acquired from the two sensors. Levinson and Thrun [52] used a series of corresponding laser scans and camera images to estimate the calibration parameters. In [53], similar mutual information metrics between the two sensors' data were adopted by measuring the statistical dependence between the data [54].

Pandey [53] proposed a theoretical derivation that estimates the kernel density of the probability distribution of the sensor data. Scott [54] suggested an approach for automatic calibra-

tion using diligently selected natural scenes. This algorithm could search over the chosen locations to extract models and yield better results. This approach has advantages in that it requires no knowledge of the physical world and continuously finds scenes to constrain the optimization parameters. Notably, Jeong et al. [55] exploited known features, such as road markings, captured by both the lidar and the stereo sensors and used a multiple cost function for robust optimization even with rough initial values. Their work considered the road as a reference for computing the transformation. However, it should be done in a controlled environment, is time-consuming and requires significant effort to obtain the transformation parameters.

In this paper, we propose an easy and efficient procedure to perform the lidar and camera calibration using a checkerboard calibration target. Similarly to [56], in our approach, we first compute the planes' coefficients by using the data from both the lidar and the camera data. Next, these coefficients are used to build a well-structured set of 3D points residing in that plane. But here, our recently-developed Correntropy Similarity Matrix (CoSM ICP) approach is used to align the constructed set of 3D points and then, to compute the relative transformation between the sensors. In essence, the significant contributions of our work are outlined as follows:

- A new algorithm developed to find planes of the checkerboard pattern acquired from both the lidar and the camera sensor's data.
- We propose to compute the plane coefficients separately from the data of both sensors.
- From the coefficients obtained, we determine the plane's location and populate it with structured data points.
- Our proposed algorithm only needs to populate a limited number of points for the plane to plane matching.
- Our CoSM ICP approach is leveraged to perform the alignment and derive the relative transformation.

The remaining paper is organized as follows: Section II describes the proposed methodology and its implementation. The results and evaluation analysis are included in Section III. The discussion is given in Section IV. A conclusion is drawn in Section V.

II. PROPOSED METHODOLOGY

In this section, we provide detail of our lidar-camera calibration method and describe the steps involved, correspondingly to the data acquired from the 3D lidar sensor and the camera data. The overall procedure in our work is shown in Fig. 2. As can be seen therein from left to right, Fig. 2 initially shows the simulated scene setup that includes a lidar and a camera separated by a certain transformation. We acquire only a single frame of the lidar's 3D data and the camera's 2D image data, which contains the calibration target in its view. For the 3D lidar sensor, we manually select the region containing the 3D points corresponding to the calibration target (checkerboard). For the camera's 2D data, we assume

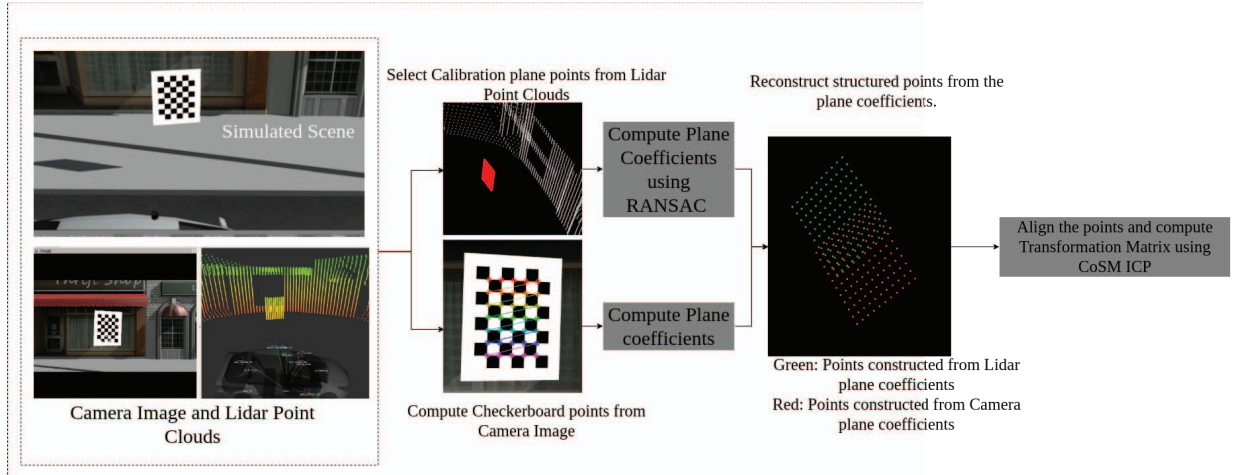


Fig. 2: Flowchart of the proposed approach.

that the camera has been calibrated, which means that the camera's intrinsic parameters are already known. We project the 2D corner points to 3D points in the world frame. We discuss this process in more detail in subsection II-B. Here, it is convenient to use the same calibration target for performing the camera calibration with the key point being to determine the plane coefficients acquired from the lidar's 3D data and the camera's projected 3D points. We compute the plane coefficients for the manually selected region of the lidar data using Random Sample Consensus (RANSAC). Using the camera's data, we compute the corner points of the checkerboard. We then project the corners points in 3D space in the world frame using the intrinsic and extrinsic parameters to compute the plane coefficients of these 3D projected corner points. We construct a well-structured set of points using the plane coefficients, computed independently from the lidar and the camera data. Our CoSM ICP approach is applied to compute the transformation between these constructed point sets. The transformation computed by the CoSM ICP eventually returns the relative transformation between the lidar and the camera sensors.

A. Lidar data processing

Using a single calibration target is relatively easy and intuitive for extrinsic calibration between multiple sensors. As mentioned earlier, the first step for our procedure is to capture the 3D points of the lidar data. From the 3D data, we manually pick the region containing the points corresponding to the calibration target and ignore the rest. The selected data contain points of the calibration target, which is effectively a plane. Other automatic approaches, like distance filtering, can be used for this process. The user can deliberately pick the region for complete control. Now, from this 'selected' point set, one can get the corresponding plane coefficients. For this, RANSAC is our choice, which determines the best-fit plane of 3D points using inliers. The following steps are adopted for our work:

- Randomly selects 3 points from the 'selected' 3D lidar

points.

- Compute the plane equation using these 3 points.
- Compute inliers by using the computed plane with all other 3D points.
- Repeat the process with the highest inlier ratio.

For this setup, we set the maximum iterations for our RANSAC algorithm as 1000. Any 3D points that are within 10mm from the plane are considered inliers. The inlier ratio, which crosses 90% or more, is considered as a best-fit plane. The plane equation computed from the lidar points is obtained as $a_l x + b_l y + c_l z + d_l = 0$, where a_l, b_l, c_l and d_l represent the coefficients of the plane.

B. Camera data processing

This section details the process of computing the 3D plane equations of the calibration target using the 2D image data acquired from the camera. In this work, we assume that the camera is already calibrated. We know the intrinsic parameters $\alpha_x, \alpha_y, x_0, y_0$, where α_x, α_y represents the camera's focal length in terms of pixel dimensions and x_0, y_0 represent the principal point in the pixel dimensions. The parameters are described in the matrix form denoted as K . We ignore the skew, radial, and tangential distortion since we operate in a simulated environment. The checkerboard corners are computed using OpenCV *findChessboardCorners* function, based on the algorithm by Duda and Frese [57]. After the corner point detection, we use the well-known PnP algorithm to compute the pose of the calibration object from 3D-2D point correspondences. The result of PnP gives us the rotation and the translation components that transform a 3D point expressed in the object coordinate frame to the camera coordinate frame. To this end, we use an iterative method based on the Levenberg-Marquardt optimization method to minimize the reprojection error. From the set of 3D computed corner points of the checkerboard pattern, we select three random points and transform them from the object frame to the camera frame using the rotation and the translation components computed from the PnP method. Let the 3 randomly

selected points be $p_1 = \{x_1, y_1, z_1\}$, $p_2 = \{x_2, y_2, z_2\}$ and $p_3 = \{x_3, y_3, z_3\}$. We compute the normal vector \vec{n} as,

$$\vec{n} = (p_2 - p_1) \times (p_3 - p_1), \quad (1)$$

where \times is the cross product in Equ.(1). The coefficients of the plane $a_c x + b_c y + c_c z + d_c = 0$ computed from the camera points are then obtained as,

$$\begin{aligned} a_c &= n.x, \\ b_c &= n.y, \\ c_c &= n.z, \\ d_c &= -(a * p_1.x + b * p_1.y + c * p_1.z), \end{aligned} \quad (2)$$

where $*$ denotes multiplication and $.$ represents the individual components in a vector.

C. Transformation estimation

We compute the transformation between these sensors from the plane equations calculated for both the lidar and the camera data. In this step, we use the calculated plane equations to populate the planes with a fixed number of points separated by a known distance (e.g., 100 points). The point sets generated from this process can be 'aligned.' Figure 3 shows typically the structured set of points. The computed transformation is between the lidar and the camera sensor. The primary reason for this process is that the number of points in the lidar point set is different from the number of points in the computed 3D points from the camera. An equal number of points is required for alignment. We describe our CoSM ICP approach to compute the transformation between the lidar and the camera data points in the following.

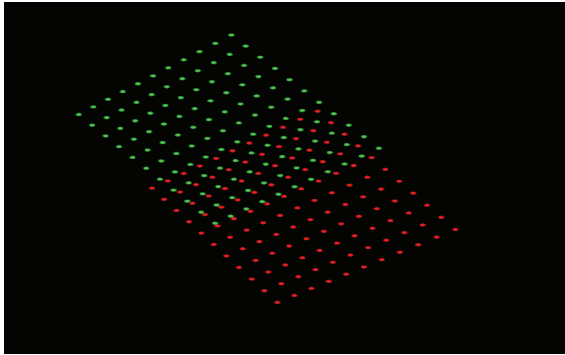


Fig. 3: Structured set of points: Green is the *Source* point cloud \mathbf{P}_s (as computed from lidar points). \mathbf{P}_s contains N points \mathbf{p}_j ($j = 1, \dots, N$), each of which is a 3D representation x_j, y_j, z_j . White is the *Target* point cloud \mathbf{P}_t (as computed from camera data). \mathbf{P}_t contains N points \mathbf{q}_k ($k = 1, \dots, N$), each of which is a 3D representation x_k, y_k, z_k .

D. Correntropy similarity matrix with iterative closest point algorithm

The idea of Correntropy coming from our previous work is applied to the well-known ICP algorithm [58]. Traditional ICP [59] describes the alignment of point clouds such that the Mean Square Error (MSE) between point sets is

minimized. The MSE is the key criterion of convergence in ICP and its variants. The ICP revolves around the problem of aligning point sets \mathbf{P}_s and \mathbf{P}_t , defined as $\mathbf{P}_s = \{\{\mathbf{p}_j\}_{j=1}^N\}$ and $\mathbf{P}_t = \{\{\mathbf{q}_k\}_{k=1}^N\}$, respectively. Our goal is to best align *Source* point set \mathbf{P}_s to *Target* point set (or a model point set) \mathbf{P}_t . This leads to finding the appropriate rotation and translation between the two point sets, such that the Root Mean Square Error (RMSE) error between the two point sets is minimized (or is bounded within a defined threshold). This process is commonly known as registration between the *Source* \mathbf{P}_s and the *Target* \mathbf{P}_t . Now, the problem can be formulated as to find the best rotation \mathbf{R} and translation \mathbf{tr} such that the MSE, ε^2 , between two point clouds (\mathbf{P}_s and \mathbf{P}_t) is minimized, i.e. mathematically,

$$\varepsilon^2 = \sum_{j,k=1}^N \|\mathbf{p}_j - (s\mathbf{R}\mathbf{q}_k + \mathbf{tr})\|^2, \quad (3)$$

where s is the scaling component. $\mathbf{p}_j \in \mathbf{P}_s$ is a point in the *Source* point set, and $\mathbf{q}_k \in \mathbf{P}_t$ is a point in the *Target* point set ($j, k = 1, \dots, N$). It is well known from Arun et al. [59] that initially the corresponding points between *Source* and *Target* point sets are computed as the shortest point from each point in the *Source* dataset to the *Target* dataset. This can be efficiently done using the $k-d$ tree data structure. Here, we introduce an additional parameter using the Correntropy criterion to find the similarity metric between points. Now, we say that for a *Source* point index i the corresponding point index in the *Target* point set is $\mathbf{c}(i)$, where \mathbf{c} is the vector of corresponding points from *Source* to *Target*. (Note: we do not perform reciprocal correspondence, i.e., from *Target* to *Source*). We compute the similarity from the difference \mathbf{d} between the two corresponding points \mathbf{P}_s and \mathbf{P}_t as

$$\begin{aligned} \mathbf{d} &= \mathbf{P}_s(i) - \mathbf{P}_t(\mathbf{c}(i)), \\ G_\sigma(\mathbf{P}_s(i) - \mathbf{P}_t(\mathbf{c}(i))) &= \frac{1}{(2\pi\sigma)^{\frac{D}{2}}} \exp\left(-\frac{\mathbf{d}^T \mathbf{d}}{2\sigma^2}\right). \end{aligned} \quad (4)$$

Here, D is the dimension with $D = 3$ in this case [60]. Now, we can create a similarity matrix \mathbf{SM} between the *Source* and the *Target* points based on the similarity metric (4). Intuitively, the size of the similarity matrix is $N \times N$. In every iteration we initialize the value of this similarity matrix at zeros and update them in every iteration based on the similarity metric (4). Its elements are defined as,

$$\begin{aligned} \mathbf{SM}(i, \mathbf{c}(i)) &= G_\sigma(\mathbf{P}_s(i) - \mathbf{P}_t(\mathbf{c}(i))), \\ \mathbf{SM}(\mathbf{c}(i), i) &= G_\sigma(\mathbf{P}_s(i) - \mathbf{P}_t(\mathbf{c}(i))). \end{aligned} \quad (5)$$

The similarity matrix \mathbf{SM} is used in the computation of the rotation component. The centroids of both point clouds, the *Source* and *Target*, are computed as,

$$\begin{aligned} \mathbf{s}_{\text{cen}} &= \frac{1}{N} \sum_{i=1}^N \mathbf{p}_i, \\ \mathbf{t}_{\text{cen}} &= \frac{1}{N} \sum_{i=1}^N \mathbf{q}_i, \end{aligned} \quad (6)$$

where \mathbf{s}_{cen} and \mathbf{t}_{cen} are the computed centroid of the *Source* and *Target* data set, respectively. The difference is computed between the computed centroid \mathbf{s}_{cen} and the individual *Source* points. The same procedure is followed for the *Target* points. Those differences are thus written as,

$$\begin{aligned} \mathbf{p}'_i &= \mathbf{p}_i - \mathbf{s}_{\text{cen}}, \\ \mathbf{q}'_j &= \mathbf{q}_j - \mathbf{t}_{\text{cen}}. \end{aligned} \quad (7)$$

The Singular Value Decomposition (SVD) for finding the rotation component is based first on finding the following matrix:

$$\mathbf{H} = \sum_{i=1}^N \mathbf{p}'_i \mathbf{SM} \mathbf{q}'_i{}^T, \quad (8)$$

where the similarity matrix \mathbf{SM} is of size $N \times N$, and the size of \mathbf{p}' , \mathbf{q}' is $4 \times N$ (under homogeneous transformation), resulting in the dimension 4×4 for matrix \mathbf{H} . The rest of the algorithm is to proceed with the SVD of matrix \mathbf{H} determined conventionally by

$$\mathbf{H} = \mathbf{U} \mathbf{\Lambda} \mathbf{V}^T. \quad (9)$$

Now the Rotation component is computed as,

$$\mathbf{R} = \mathbf{V} \mathbf{U}^T. \quad (10)$$

The determinant of matrix \mathbf{R} must be a positive integer. The translation component \mathbf{tr} is simply computed as the difference of the centroids, i.e.

$$\mathbf{tr} = \mathbf{s}_{\text{cen}} - \mathbf{t}_{\text{cen}}. \quad (11)$$

Algorithm 1 describes the entire procedure of our proposed method. Algorithm 2 describes how the shot noise or impulse noise is generated.

III. RESULTS

We performed our initial evaluation in a simulated environment provided by Open Robotics [61]. To demonstrate our results, we start with a basic dataset containing simulated lidar data and camera setup (mounted on a simulated Prius model car) established in a virtual environment in Gazebo, which is shown in Fig.4 (a) and (b) [62]. The primary reason for selecting this simulation setup is to compare our results since we know the ground truth. This setup contains other environmental complexities like buildings and cars (apart from the calibration card) and simulated lighting conditions.

Our experiments test the results with a linear transformation ranging from $0.05m$ to $2.5m$ (along x, y, z axes). We set up a simple test case of lidar-camera, where the stereo camera faces the calibration target and is placed near the lidar sensor under various configurations (e.g., $5cm$ along the y axis of the lidar sensor or $\mathbf{t} = [0, 0.05, 0.0]$). Here, we intend to calculate the transformation between the camera (C_c) with respect to the lidar's frame L_c . The camera has a resolution of 1280×720 , and since we use a simulated setup, we ignore the radial and tangential distortion (both were set to

Algorithm 1 CoSM Algorithm for Lidar-Stereo Camera Calibration

```

1: function READDATASETS( $\mathbf{P}_s, \mathbf{P}_t$ ) ▷ Read the Source
   and the Target datasets.
2:   while not converged do
3:      $\mathbf{c} \leftarrow$  ComputeCorrespondence( $\mathbf{P}_s, \mathbf{P}_t$ ). ▷
   Compute Correspondence
4:      $\mathbf{SM} = \text{zeros}(N, N)$  ▷ Initialize  $N \times N$ 
   Similarity Matrix to all zeros.
5:     for  $i \leftarrow 1$  to  $N$  do
6:        $\mathbf{d} = \mathbf{P}_s(i) - \mathbf{P}_t(\mathbf{c}(i))$ .
7:       Compute  $G_\sigma$  as shown in Equ.4.
8:        $\mathbf{SM}(i, \mathbf{c}(i)) = G_\sigma(\mathbf{P}_s(i) - \mathbf{P}_t(\mathbf{c}(i)))$ .
9:        $\mathbf{SM}(\mathbf{c}(i), i) = G_\sigma(\mathbf{P}_s(i) - \mathbf{P}_t(\mathbf{c}(i)))$ .
10:    end for
11:    Compute centroids as given in Equ.(6).
12:    Compute the differences as given in Equ. (7).
13:    Compute  $\mathbf{H}$  Matrix as given in Equ. (8).
14:    Compute SVD of  $\mathbf{H}$  as given in Equ. (9).
15:    Compute  $\mathbf{R}$  as given in Equ. (10).
16:    Compute  $\mathbf{tr}$  as given in Equ. (11).
17:  end while
18: end function

```

Algorithm 2 Algorithm to generate shot noise/impulse noise.

```

1: function GENERATEOUTLIERS( $min, max$ ) ▷ Function
   to Generate outliers.
2:    $N=100$ 
3:    $meanidx = \text{GenerateRandomNumers}(min, max, N)$ 
4:    $varianceidx = \text{GenerateRandomNumers}(0.01, 100, N)$ 
5:   for  $i \leftarrow 1$  to  $N$  do
6:      $vals = \text{GenerateNumbers}(meanidx[i], varianceidx[i])$ 
7:     SaveFile(m1)
8:   end for
9:    $noisevals = \text{PickRandomVals}(vals)$ 

```

0). In this scenario, we can evaluate our approach on multiple configurations given the known ground truth, as can be seen in Fig. 4. In this case, our problem statement is defined as finding the transformation between the Lidar (L_c) and the camera (C_c) setup by using our proposed methodology. This setup provides a basic test case to verify our algorithm, which can be extended to complex real-time test cases.

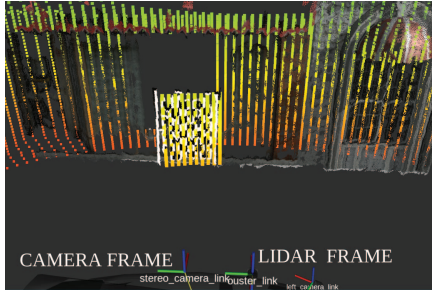
A. Evaluation on Simulated data

The various configurations for our experiments are shown in Table I. It shows the original ground truth transformations between the lidar and the camera sensor. We place the calibration target at an average distance of 2-3 m from the calibration target throughout our experiments. This distance is in accordance with the lidar coordinate frame (L_c).

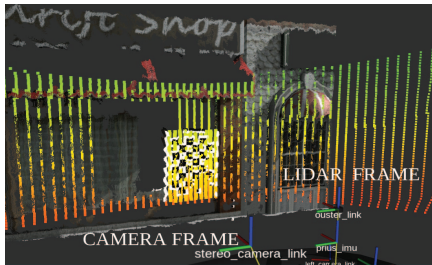
For further evaluation, we show the average error in multiple configurations as obtained in the experiments. Figure 5 depicts the average error of the individual components of the rotation and translation components under various



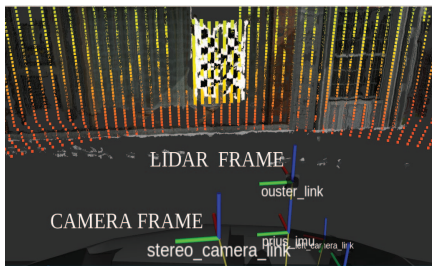
(a)



(b)



(c)



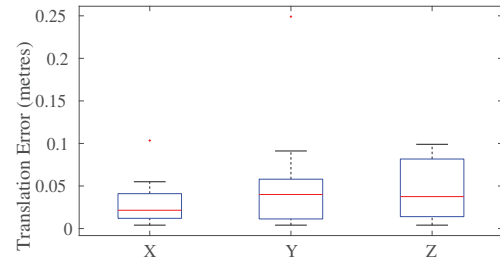
(d)

Fig. 4: Simulation setup for evaluating lidar-camera calibration under multiple configurations: (a) denotes a gazebo simulation of Prius car model with lidar and stereo camera; (b) denotes the TF-frames of the lidar and camera. The link *ouster_link* is the reference frame for lidar; (c) and (d) denote TF-frames of multiple configurations of lidar-camera setup. $\mathbf{t} = [t_x, t_y, t_z]$ denotes the translation component along x, y, z . It essentially denotes how the camera is transformed with respect to the lidar sensor along x, y, z . For (c) the translation between the lidar and the stereo sensor is $\mathbf{t} = [0, 0.5, 0.0]$, and for (d) it is $\mathbf{t} = [-0.5, 0.5, 0]$.

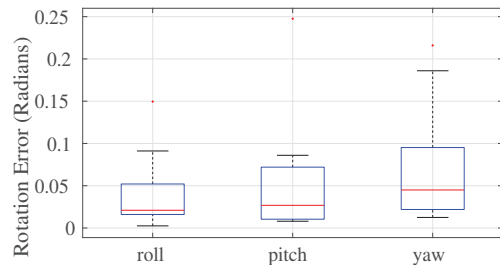
TABLE I: Configuration setups used in our experiments (or Ground Truth).

| Setting | $t_x(m)$ | $t_y(m)$ | $t_z(m)$ | roll (rad) | pitch(rad) | yaw(rad) |
|---------|----------|----------|----------|------------|------------|-----------|
| 1 | 0 | 0.4 | 0.0 | 0.0 | 0.0 | 0.0 |
| 2 | 0 | 0.1 | 0.0 | 0.0 | 0.0 | 0.0 |
| 3 | 0 | 0.05 | 0.0 | 0.0 | 0.0 | 0.0 |
| 4 | 0 | 0.6 | 0.0 | 0.0 | 0.0 | 0.0 |
| 5 | 0 | 1.2 | 0.0 | 0.0 | 0.0 | 0.0 |
| 6 | 0 | 2.2 | 0.0 | 0.0 | 0.0 | 0.0 |
| 7 | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 | 0.0 |
| 8 | -0.5 | 1.5 | 0.3 | 0.0 | 0.0 | 0.0 |
| 9 | -0.5 | 0.5 | 0 | 0.0 | 0.0 | -0.0 |
| 10 | 0.5 | 0.5 | 0 | 0 | 0 | -0.523599 |
| 11 | 0.5 | 0.5 | 0.3 | 0 | 0.349066 | -0.523599 |
| 12 | 0.3 | 0.6 | 0.4 | 0 | 0.349066 | -0.523599 |
| 13 | 0.2 | 0.3 | 0.2 | 0.261799 | 0 | -0.523599 |
| 14 | 0.7 | 0.2 | 0.9 | 0 | 0.349066 | 0 |

configurations. From the experiments performed with the results Fig. 5, we can see that under simple translation along x, y, z axes, our algorithm displays good performance with individual RMSE's ~ 0.01 for all the individual components. Even under small rotations, our approach returns relatively close values (RMSE ~ 0.02) compared to the ground truth. However, when the transformation between C_c and L_c is significant (>60 degrees or 1.0472 radians), the overall RMSE increases and the computed transformation appears to deflect from the ground truth values (RMSE >6.891).



(a)



(b)

Fig. 5: Translation and Rotation errors of individual components (x, y, z and roll, pitch and yaw) under various configurations compared to ground truth.

One of the critical observations here was to accurately estimate 3D points computed from the camera setup. Throughout the experiments, datasets were collected for all the individual configurations as presented in Table I. Here, we deliberately selected each dataset having a good collection of 3D point

sets with less visible outliers. It is expected that with further advancements in 3D depth estimation, we can get more accurate and robust results.

As with a simulated setup, it could be simple to have a lidar sensor, a camera setup, and a calibration target. Since our primary goal was to have an effective calibration technique under various environments, we plan to evaluate our approach in real-time. Testing a lidar and a camera setup under different configurations is however time-consuming. As such, in our simulated environment, we added environmental complexities and verified the merits of our approach. A user can select the region in the 3D point sets corresponding to the calibration plane. Since it is essential for the algorithm to have accurate plane coefficients, a manual selection can provide complete control.

IV. CONCLUSIONS

This work proposes a novel algorithm for efficient calibration of Lidar-Camera sensors by using a single frame from the lidar data and 3D points estimated from the camera data. Our approach can avoid using multiple poses of the calibration target since it only needs one frame for data of both the lidar and the camera. The later steps involve manually selecting 3D lidar points and estimating the plane coefficients from the lidar data, as well as automatically detecting the planes via the checkerboard corners from the camera data. Once the computed plane coefficients have been obtained (from both the sensor's data), we construct a well-spaced 3D point structure. In the procedure, we propose to use our recently-developed methodology, the CoSM ICP, to compute the transformation between the 'structured' points, thereby accomplishing the calibration purpose. Experiments on a standard simulated environment have indicated that CoSM ICP is robust to rotations and translations, which could aid in computing the transformation between the lidar and the camera frame. The results have shown a positive approach towards a single frame lidar-camera calibration with promising results in the transformation computation between the lidar and the camera sensor. It would benefit the community to validate our approach for their own lidar-camera setup in real-time environments.

REFERENCES

- [1] R. S. Lim, H. M. La, and W. Sheng, "A robotic crack inspection and mapping system for bridge deck maintenance," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 2, pp. 367–378, 2014.
- [2] Q. P. Ha, H. M. La, S. Wang, and C. Balaguer, "Special issue on recent advances in field and service robotics: handling harsh environments and cooperation," *Robotica*, p. 1–3, 2022.
- [3] A. Singandhupe and H. M. La, "Single frame lidar and stereo camera calibration using registration of 3d planes," in *2021 Fifth IEEE International Conference on Robotic Computing (IRC)*, 2021, pp. 115–118.
- [4] L. Huang and M. Barth, "A novel multi-planar lidar and computer vision calibration procedure using 2d patterns for automated navigation," in *2009 IEEE Intelligent Vehicles Symposium*. IEEE, 2009, pp. 117–122.
- [5] A. Sehgal, A. Singandhupe, H. La, A. Tavakkoli, and S. Louis, "Lidar-monocular visual odometry with genetic algorithm for parameter optimization," in *The 14th International Symposium on Visual Computing (ISVC), Oct. 7-9, Lake Tahoe, NV, USA*, 10 2019, pp. 358–370.
- [6] A. Singandhupe and H. La, "A review of slam techniques and security in autonomous driving," in *Proceedings of the 3rd IEEE International Conference on Robotic Computing (IRC), February 25-27, Naples, Italy*, 02 2019, pp. 602–607.
- [7] J. Beltrán, C. Guindel, and F. García, "Automatic extrinsic calibration method for lidar and camera sensor setups," *CoRR*, vol. abs/2101.04431, 2021. [Online]. Available: <https://arxiv.org/abs/2101.04431>
- [8] P. An, T. Ma, K. Yu, B. Fang, J. Zhang, W. Fu, and J. Ma, "Geometric calibration for lidar-camera system fusing 3d-2d and 3d-3d point correspondences," *Opt. Express*, vol. 28, no. 2, pp. 2122–2141, Jan 2020. [Online]. Available: <http://www.opticsexpress.org/abstract.cfm?URI=oe-28-2-2122>
- [9] J. Shi, Z. Zhu, J. Zhang, R. Liu, Z. Wang, S. Chen, and H. Liu, "Calibrating camera and lidar by recurrent convolutional neural network and geometric constraints," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10 197–10 202, 2020.
- [10] Q. Ha, L. Yen, and C. Balaguer, "Robotic autonomous systems for earthmoving in military applications," *Automation in Construction*, vol. 107, p. 102934, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0926580518309932>
- [11] Y. Alghamdi, A. Munir, and H. M. La, "Architecture, classification, and applications of contemporary unmanned aerial vehicles," *IEEE Consumer Electronics Magazine*, vol. 10, no. 6, pp. 9–20, 2021.
- [12] A. Sehgal, N. Ward, H. La, and S. Louis, "Automatic parameter optimization using genetic algorithm in deep reinforcement learning for robotic manipulation tasks," 2022. [Online]. Available: <https://arxiv.org/abs/2204.03656>
- [13] A. Sehgal, M. Sehgal, and H. M. La, "Aacher: Assorted actor-critic deep reinforcement learning with hindsight experience replay," 2022. [Online]. Available: <https://arxiv.org/abs/2210.12892>
- [14] P. Acharya, K.-D. Nguyen, H. M. La, D. Liu, and I.-M. Chen, "Nonprehensile manipulation: a trajectory-planning perspective," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 1, pp. 527–538, 2021.
- [15] H. Nguyen and H. La, "Review of deep reinforcement learning for robot manipulation," in *2019 Third IEEE International Conference on Robotic Computing (IRC)*, 2019, pp. 590–595.
- [16] P. Nguyen, H. Nguyen, D. Nguyen, T. N. Dinh, H. M. La, and T. Vu, "Parksense: Automatic parking positioning by leveraging in-vehicle magnetic field variation," *IEEE Access*, vol. 5, pp. 25 021–25 033, 2017.
- [17] H. M. La, R. S. Lim, J. Du, S. Zhang, G. Yan, and W. Sheng, "Development of a small-scale research platform for intelligent transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1753–1762, 2012.
- [18] H. Bozorgi, X. T. Truong, H. M. La, and T. D. Ngo, "2d laser and 3d camera data integration and filtering for human trajectory tracking," in *2021 IEEE/SICE International Symposium on System Integration (SII)*, 2021, pp. 634–639.
- [19] L. Van Nguyen and H. M. La, "Real-time human foot motion localization algorithm with dynamic speed," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 6, pp. 822–833, 2016.
- [20] A. Q. Pham, A. T. La, E. Chang, and H. M. La, "Flying-climbing mobile robot for steel bridge inspection," in *2021 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2021, pp. 230–235.
- [21] S. T. Nguyen and H. M. La, "Development of a steel bridge climbing robot," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 1912–1917.
- [22] H. M. La, N. Gucunski, K. Dana, and S.-H. Kee, "Development of an autonomous bridge deck inspection robotic system," *Journal of Field Robotics*, vol. 34, no. 8, pp. 1489–1504, 2017. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21725>

- [23] S. Gibb, T. Le, H. M. La, R. Schmid, and T. Berendsen, "A multi-functional inspection robot for civil infrastructure evaluation and maintenance," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2672–2677.
- [24] H. M. La, N. Gucunski, S.-H. Kee, J. Yi, T. Senlet, and L. Nguyen, "Autonomous robotic system for bridge deck data collection and analysis," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 1950–1955.
- [25] H. A. Nguyen and Q. P. Ha, "Robotic autonomous systems for earthmoving equipment operating in volatile conditions and teaming capacity: a survey," *Robotica*, p. 1–25, 2022.
- [26] R. Chen, K. T. Tran, H. M. La, T. Rawlinson, and K. Dinh, "Detection of delamination and rebar debonding in concrete structures with ultrasonic sh-waveform tomography," *Automation in Construction*, vol. 133, p. 104004, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0926580521004556>
- [27] H. Ahmed, H. M. La, and K. Tran, "Rebar detection and localization for bridge deck inspection and evaluation using deep residual networks," *Automation in Construction*, vol. 120, p. 103393, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0926580520309730>
- [28] Qilong Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, 2004, pp. 2301–2306 vol.3.
- [29] O. Naroditsky, A. Patterson, and K. Daniilidis, "Automatic alignment of a camera with a line scan lidar system," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 3429–3434.
- [30] S. Verma, J. S. Berrio, S. Worrall, and E. Nebot, "Automatic extrinsic calibration between a camera and a 3d lidar using 3d point and plane correspondences," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 3906–3912.
- [31] A. Dhall, K. Chelani, V. Radhakrishnan, and K. M. Krishna, "Lidar-camera calibration using 3d-3d point correspondences," *arXiv preprint arXiv:1705.09785*, 2017.
- [32] C. Guindel, J. Beltrán, D. Martín, and F. García, "Automatic extrinsic calibration for lidar-stereo vehicle sensor setups," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017, pp. 1–6.
- [33] Y. Park, S. Yun, C. Won, K. Cho, K. Um, and S. Sim, "Calibration between color camera and 3d lidar instruments with a polygonal planar board," *Sensors (Basel, Switzerland)*, vol. 14, pp. 5333–5353, 03 2014.
- [34] S. Debattisti, L. Mazzei, and M. Panciroli, "Automated extrinsic laser and camera inter-calibration using triangular targets," in *2013 IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 696–701.
- [35] X. Xu, L. Zhang, J. Yang, C. Liu, Y. Xiong, M. Luo, Z. Tan, and B. Liu, "Lidar-camera calibration method based on ranging statistical characteristics and improved ransac algorithm," *Robotics and Autonomous Systems*, vol. 141, p. 103776, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0921889021000610>
- [36] Y. Zhu, C. Li, and Y. Zhang, "Online camera-lidar calibration with sensor semantic information," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4970–4976.
- [37] Y. Su, Y. Ding, J. Yang, and H. Kong, "A two-step approach to lidar-camera calibration," in *2020 25th International Conference on Pattern Recognition (ICPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jan 2021, pp. 6834–6841. [Online]. Available: <https://doi.ieeecomputersociety.org/10.1109/ICPR48806.2021.9412085>
- [38] F. M. Mirzaei, D. G. Kottas, and S. Roumeliotis, "3d lidar-camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization," *The International Journal of Robotics Research*, vol. 31, pp. 452 – 467, 2012.
- [39] R. Unnikrishnan and M. Hebert, "Fast extrinsic calibration of a laser rangefinder to a camera," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-05-09*, 01 2005.
- [40] A. Geiger, F. Moosmann, O. Car, and B. Schuster, "Automatic camera and range sensor calibration using a single shot," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 3936–3943.
- [41] L. Zhou, Z. Li, and M. Kaess, "Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 5562–5569.
- [42] Z. Pusztai and L. Hajder, "Accurate calibration of lidar-camera systems using ordinary boxes," in *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017, pp. 394–402.
- [43] T. Tóth, Z. Pusztai, and L. Hajder, "Automatic lidar-camera calibration of extrinsic parameters using a spherical target," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 8580–8586.
- [44] Z. Bai, G. Jiang, and A. Xu, "Lidar-camera calibration using line correspondences," *Sensors*, vol. 20, p. 6319, 11 2020.
- [45] M. Velas, M. Spänel, Z. Materna, and A. Herout, "Calibration of rgb camera with velodyne lidar," 2014.
- [46] L. Heng, M. Bürki, G. Lee, P. Furgale, R. Siegwart, and M. Pollefeys, "Infrastructure-based calibration of a multi-camera rig," 05 2014, pp. 4912–4919.
- [47] C. Häne, L. Heng, G. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys, "3d visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection," *Image and Vision Computing*, 08 2017.
- [48] K. Irie, M. Sugiyama, and M. Tomono, "Target-less camera-lidar extrinsic calibration using a bagged dependence estimator," in *2016 IEEE International Conference on Automation Science and Engineering (CASE)*, 2016, pp. 1340–1347.
- [49] D. Scaramuzza, A. Harati, and R. Siegwart, "Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 4164–4169.
- [50] P. Moghadam, M. Bosse, and R. Zlot, "Line-based extrinsic calibration of range and image sensors," vol. 2, 05 2013.
- [51] F. Boughorbel, D. Page, C. Dumont, and M. Abidi, "Registration and integration of multisensor data for photorealistic scene reconstruction," *Proc SPIE*, 03 2001.
- [52] J. Levinson and S. Thrun, *Unsupervised Calibration for Multi-beam Lasers*, 01 2014, pp. 179–193.
- [53] G. Pandey, J. McBride, S. Savarese, and R. Eustice, "Automatic extrinsic calibration of vision and lidar by maximizing mutual information," *Journal of Field Robotics*, vol. 32, 09 2014.
- [54] T. Scott, A. A. Morye, P. Piniés, L. M. Paz, I. Posner, and P. Newman, "Choosing a time and place for calibration of lidar-camera systems," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4349–4356.
- [55] 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, 2019.
- [56] E.-s. Kim and S.-Y. Park, "Extrinsic calibration between camera and lidar sensors by matching multiple 3d planes," *Sensors*, vol. 20, no. 1, 2020. [Online]. Available: <https://www.mdpi.com/1424-8220/20/1/52>
- [57] A. Duda and U. Frese, "Accurate detection and localization of checkerboard corners for calibration," 09 2018.
- [58] A. Singandhupe, H. M. La, T. D. Ngo, and V. A. Ho, "Registration of 3d point sets using correntropy similarity matrix," *CoRR*, vol. abs/2107.09725, 2021. [Online]. Available: <https://arxiv.org/abs/2107.09725>
- [59] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 5, p. 698–700, May 1987. [Online]. Available: <https://doi.org/10.1109/TPAMI.1987.4767965>
- [60] J. Principe and J. Iii, "Information-theoretic learning," *Advances in unsupervised adaptive filtering*, 09 2000.
- [61] M. Quigley, K. Conley, B. P. 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.
- [62] O. S. R. Foundation, *Vehicle and city Simulation*, 2017-10-26. [Online]. Available: http://gazebosim.org/blog/car_sim