# Automatic extrinsic calibration between a camera and a 3D Lidar using 3D point and plane correspondences

Surabhi Verma[1], Julie Stephany Berrio[1], Stewart Worrall[1], Eduardo Nebot[1]

*Abstract*— **This paper proposes an automated method to obtain the extrinsic calibration parameters between a camera and a 3D lidar with as low as $16$ beams. We use a checkerboard as a reference to obtain features of interest in both sensor frames. The calibration board centre point and normal vector are automatically extracted from the lidar point cloud by exploiting the geometry of the board. The corresponding features in the camera image are obtained from the camera's extrinsic matrix. We explain the reasons behind selecting these features, and why they are more robust compared to other possibilities. To obtain the optimal extrinsic parameters, we choose a genetic algorithm to address the highly non-linear state space. The process is automated after defining the bounds of the 3D experimental region relative to the lidar, and the true board dimensions. In addition, the camera is assumed to be intrinsically calibrated. Our method requires a minimum of $3$ checkerboard poses, and the calibration accuracy is demonstrated by evaluating our algorithm using real world and simulated features.**

## I. INTRODUCTION

Autonomous systems, using a multitude of sensors including 3D lidar and cameras, are being increasingly used for research and industrial applications. To enable higher levels of autonomy however, we need to extend the capabilities of such robots in order for them to construct accurate and complete models of their environment. One way to achieve this is by combining complementary information provided by the camera and lidar. Cameras, while being capable of supplying dense information in terms of color, texture, and shape of objects, are limited in their ability to provide high quality depth information at longer ranges. Lidars, on the other hand, capture accurate depth information with the drawback of lower vertical spatial resolution and a lack of colour/texture information. Furthermore, lidars are more suited to environments with variation in illumination and weather conditions. The fusion of camera and lidar makes it possible to overcome the limitations of each individual sensor. The main challenge in fusing these two different sensor modalities is the requirement for a precise calibration of the camera's intrinsic parameters, and the geometrical extrinsic parameters which includes the 3D transformation between the two sensors [1].

Given accurate extrinsic parameters, it is possible to transform the lidar point cloud to the camera frame and then project these points on an image based on the intrinsic parameters for the camera. Such a projection can be used for various purposes, for example in [2], segmented point clouds are used as the ground truth to validate the performance of a CNN for a specific label. These parameters also enable camera to laser projection, where the corresponding image pixel data (RGB values, labels, etc.) is used to augment the point cloud fields. For example, authors in [3] translate the semantic information from images to point clouds to generate 3D semantic maps. Applications such as these can improve the overall perception of the robot. We therefore propose an approach to obtain the extrinsic parameters, describing the relative 3D rotation and translation, between a camera and a 3D lidar. This is a particularly challenging problem as the object features are obtained from different sensors with different modalities and noise patterns. Noisy features reduce the accuracy of calibration. Furthermore, not all lidars/cameras have similar behavior and measurement errors making it difficult to generalize an approach. We address these issues by selecting features that are less susceptible to noise from sensor measurements, and by using a robust optimization strategy. It is challenging to calibrate a lidar sensor with a small number of beams as this results in a reduced vertical resolution. Our approach is designed to provide an accurate calibration for lidars with as low as 16-beams, though it is equally applicable to lidars with more beams.

Previously, several approaches have been proposed to solve the camera-lidar calibration problem. These approaches can be broadly classified into two categories: target-based and target-less methods. Targets such as a checkerboard, fiducial marker [2] [3] or custom-made target [4] [5] have been used to find correspondences between features perceived in both the sensor frames. On the other hand, a target-less method uses features from natural scenes. Some of these methods [6] involve feature correspondences in the image and point cloud that are input manually. These approaches tend to require a significant number of features in the environment to reach a satisfactory calibration accuracy. Our proposed strategy uses a planar checkerboard target to compute the calibration parameters. With this approach, the inner corners of a checkerboard can be detected by the camera with sub-pixel accuracy and a planar model can be fit to the set of lidar points corresponding to the board. We will show that the automated extraction of lidar features leads to an accurate estimate of the board's geometric features. Such a method is appealing for automotive applications as the information required can be automatically obtained with minimal infrastructure within the vehicle production line.

When it comes to choosing features to form geometric

*This work was not supported by any organization

[1]S. Verma, J. Berrio, S. Worrall, E. Nebot are with the Australian Centre for Field Robotics (ACFR) at the University of Sydney (NSW, Australia). E-mails: {s.verma, j.berrio, s.worrall, e.nebot}@acfr.usyd.edu.au

constraints, several possibilities arise. In our case, with a 3D lidar, we can extract the board normal, its distance from the lidar sensor, 3D lines (edges and lines on the board plane), and points (corner and centre). These features can also be found in the image. We specifically choose the checkerboard's normal and centre point to determine its orientation and 3D location respectively. Although lines and points on the board edges as described in [7] could generate sufficient constraints to solve the calibration problem with even a single checkerboard pose, we avoided using these features. This is because estimating board edges from the lidar data incorporates two sources of error: fitting a plane to the board points, and fitting a line to the edge points. As a result, the edges tend to have a higher error when compared to the board normal, which incorporates only a single source of error from the plane fitting. In addition, the edge points also suffer from beam divergence, a characteristic inherent to the lidar **[8]**. As a result, the edge lengths, found after fitting lines to the edge points, deviate from the ground truth by a few centimeters. Issues such as these make them less suitable for calibration. The centre point on the other hand, close to the centroid of the point cluster, is less prone to errors as points near the edges vary.

Once the feature correspondences are determined, a transformation matrix can be estimated between the camera and the lidar. For this, it is crucial to use an appropriate optimization algorithm as the geometric constraints used in the cost function tends to make it non-linear. Unlike other approaches that use gradient based algorithms [7] [9] [10] [11] [12] which are susceptible to a local minima, we use a Genetic Algorithm (GA) which is a gradient-free method, similar to [13].

In the next section, we present prior work relating to target based calibration with a checkerboard. In section III, we explain the details of automatically extracting the features from a planar checkerboard target and the optimization strategy. The experiments and outcomes are presented in section IV.

## II. RELATED WORK

A checkerboard was first used by Zhang and Pless [9] to find the extrinsic parameters between a camera and a 2D Laser Rangefinder (LRF). In their method, for a number of checkerboard poses the checkerboard plane parameters are found relative to the camera. They then optimize for the transformation by minimizing the euclidean distance error between the laser points (after transformation) and the checkerboard plane (points on plane constraint). A similar approach was adopted by Unnikrishnan et al. [14] to calibrate a 3D laser scanner and a camera. They manually choose the 2D region of interest in the laser range image to find plane correspondences in both sensor frames. A two stage optimization process is used which involves estimating the rotation and translation independently and then jointly optimizing the two sets of parameters. Pandey et al. [10] calibrate an omni-directional camera and a 3D laser using the same geometric constraint as [9]. The issue with this approach is that the mere inclusion of the plane parameters, i.e. the distance of the checkerboard plane from the camera and its normal vector, to form a single constraint does not affix the location of the checkerboard on the plane. They may therefore require a higher number of checkerboard poses to converge to an optimal solution. In contrast, Zhou [15] exploits two geometric constraints from plane-line correspondences to calibrate a 2D LRF and a camera. In [12], the authors use a V-shaped calibration target formed by two triangular boards with a checkerboard on each triangle. As their target is non-planar, they are able to exploit the geometry of the setup to formulate a well-constrained cost function, minimizing point to plane distances. Our method uses a planar checkerboard, which can be used for the calibration of a camera as well. To overcome the need of obtaining several checkerboard poses for a satisfactory calibration, Geiger et al. [16] attach multiple checkerboards in different scene locations. They are thus able to get their results with a single scene shot. In practice, it is not feasible to attach several checkerboards with accuracy every time the sensors are required to be calibrated. Zhou et al. [11] estimate the rotation and translation separately. Additionally, they introduce a weighting factor for each checkerboard pose depending on the measurement uncertainty from images. In order to decouple rotation and translation, features close to the ground truth are required, which is very difficult to obtain considering the noise in the sensors, particularly the lidar.

More recently in [7], points corresponding to the edges of the checkerboard were used to fit lines. The corresponding edge lines were obtained from the camera and constraints were formed using plane-line correspondence. As explained in the previous section, these features are more prone to errors and hence we avoid using them.

## III. METHODOLOGY

Our method uses a checkerboard as a reference to obtain features of interest in the image and point cloud. For a given $i^{th}$ sample among $N$ camera$_c$-lidar$_l$ scan pairs of the checkerboard, we extract the centre point $(\mathbf{o}^i_{\mathbf{c}/\mathbf{l}})$ and the normal vector $(\mathbf{n}^i_{\mathbf{c}/\mathbf{l}})$ of the board. While the normal vector helps us determine the board plane's orientation, the centre point affixes the board's 3D location in each sensor's reference frame. Once the features are extracted, we exploit the correspondences, $\mathbf{o}^i_{\mathbf{c}} \leftrightarrow \mathbf{o}^i_{\mathbf{l}}$ and $\mathbf{n}^i_{\mathbf{c}} \leftrightarrow \mathbf{n}^i_{\mathbf{l}}$, to form constraints for rotation $(\mathbf{R}^{\mathbf{c}}_{\mathbf{l}})$ and translation $(\mathbf{t}^{\mathbf{c}}_{\mathbf{l}})$ of the lidar frame with respect to the camera frame.

The parameters required before starting the calibration process are list below:

- $(l, w)$: length and width of the rectangular board in metres on which the checkerboard is attached.
- $(m, n)$: grid size of the checkerboard, i.e. the number of internal corners in each row and column.
- $(s)$: side length of each square within the checkerboard in metres.
- $(bb_{x_{mn,mx}}, bb_{y_{mn,mx}}, bb_{z_{mn,mx}})$: minimum and maximum bounds of the 3D experimental region along the lidar's $x, y, z$ axis.

- $(f_{xy}, c_{xy}, d_{1,2,3,4,5})$: camera intrinsic parameters including the focal length, principal point and 4/5 distortion coefficients corresponding to a fisheye/pinhole camera model.

## A. *Data collection setup*

To collect the samples required for calibration, we firmly attach a checkerboard on a rigid, opaque, and rectangular board such that both their centres align and their edges remain parallel to one another. In every sample, we ensure that the entire board is visible to both sensors and the 3D experimental region is free from any other objects apart from the board and its stand. The stand is chosen such that it does not hold the board with significant protruding elements close to the board boundaries or corners. This is necessary to conveniently filter out the points corresponding to the board.

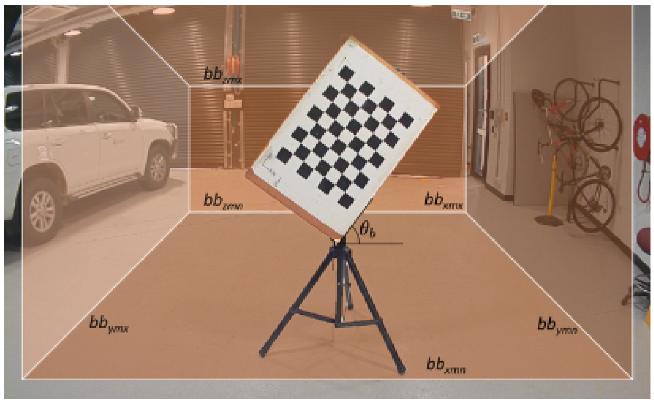

Fig. 1. Data collection setup. A virtual bounding box shows the experimental region inside which the board is located to obtain the samples.

The board is kept tilted at an angle $\theta_b$ of around 45 to 60 degrees with respect to the ground plane. Such a configuration is adopted to offset the low vertical angular resolution of the lidar. Furthermore, we ensure that the checkerboard pattern is detectable by the camera and a minimum of 2 lidar scan beams pass though each of the board's edges to allow the interpolation of points lying on the edges. We note that scan lines represent the board more accurately when the board is kept facing the sensor pair. Lastly, a diverse sample set is collected in terms of the board location in the experimental region. The data collection setup is shown in Fig. 1.

## B. *Feature Extraction*

*1) From the lidar:* The minimum and maximum bounds $(bb_{x_{mn,mx}}, bb_{y_{mn,mx}}, bb_{z_{mn,mx}})$ allow us to separate the experimental region, consisting of the board and it's stand, from the environment point cloud. Since the stand has no protruding elements through which the board is held, we can remove the points corresponding to the legs of the stand in order to obtain the board point cloud. For this, the point with the maximum $z$-coordinate value is found in the experimental region. As this point lies close the board's top most corner in the tilted configuration, we reset $bb_{z_{mx}}$ to this $z$-coordinate value and accordingly redefine $bb_{z_{mn}}$ to extract the board cloud.

$$bb_{z_{mn}} = bb_{z_{mx}} - \sqrt{l_b^2 + w_b^2} \tag{1}$$

(1) was formulated assuming that the difference in angle between the board plane and the $y-z$ plane of the lidar frame is small enough to obtain the lowermost point of the board by subtracting the newly defined $bb_{z_{mx}}$ with the board diagonal.

Next, we fit a plane to the filtered cluster of board points using 3D RANSAC [17] to get a robust estimate of the board plane. The obtained plane equation allows us to calculate $\mathbf{n}_\mathbf{l}^i$. In order to determine the centre point, we project the board point cloud to the estimated plane and determine the edge points by finding the points with the minimum and maximum $y$-coordinate value in each horizontal scan beam. The corner points of the board are then obtained by fitting 4 lines through these edge points and calculating their intersection $\overline{\mathbf{o}}_\mathbf{lk}^\mathbf{i}$; $k = 1, 2, 3, 4$. The line joining opposite corner points, $\overline{\mathbf{o}}_\mathbf{l1}^i$ and $\overline{\mathbf{o}}_\mathbf{l3}^i$, correspond to one of the board diagonals. The midpoint of this diagonal is $\mathbf{o}_\mathbf{l}^i$. Fig. 2 depicts the sensor frames and the features extracted from the board point cloud. One may note that using the board edges and corner points, claimed to be noisy in the previous section, to find $\mathbf{o}_\mathbf{l}^i$ would make it less precise as well. However, the centre point remains immune to changes in the orientation of board due to lidar measurement errors and therefore to errors in the edges and corner points.

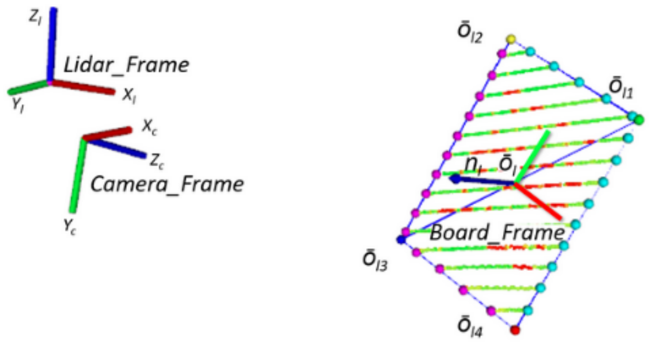

Fig. 2. Sensor frames and features extracted from the point cloud.

*2) From the camera:* The checkerboard pattern in the image is detected using the OpenCV function cv::findChessboardCorners [18]. This information allows us to compute the pose of the board reference frame relative to the camera with the Perspective-n-Point (PnP) algorithm [19]. In case of a pinhole camera, we can directly use the chessboard corners in the image frame and the board frame along with the camera intrinsic parameters as an input to PnP. For a fisheye camera model, we first undistort the raw image and obtain the corresponding checkerboard corners. This is then fed to the PnP algorithm along with zero distortion coefficients. The output of PnP is the pose, rotation

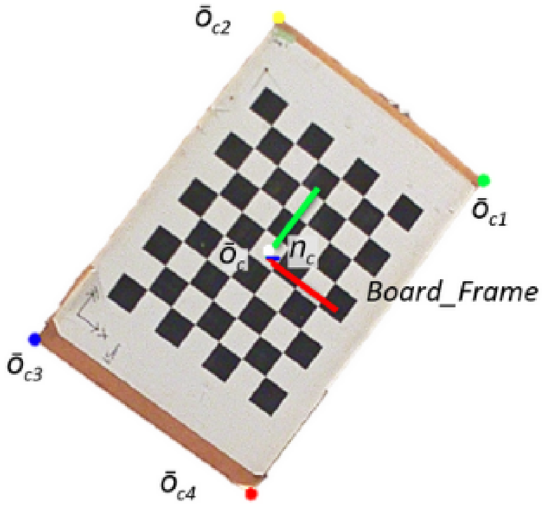

Fig. 3. Image extracted corner and centre points features.

and translation, of the board reference frame (placed at the centre of the checkerboard) relative to the camera frame. We thereby obtain $\mathbf{o^i_c}$. The $3^{rd}$ column of the rotation matrix, parametrized by the Euler angles, represents the z-axis of the board reference. This coincides with $\mathbf{n^i_c}$ given that the $x, y$-axis of the board frame lies on the board plane and the $z$-axis, normal to it. Furthermore, we can find the corner points relative to the board centre knowing $l$ and $w$. This information along with the camera extrinsics, allow us to compute $\mathbf{\bar{o}^i_{c1,2,3,4}}$ in the camera frame. The board features, as seen by the camera in its image is shown in Fig. 3.

## C. Optimization Strategy

After obtaining the required features, we optimize for the transformation from the lidar to camera frame. For this, we choose a Genetic Algorithm (GA) [20] as the preferred optimizer. Due to it's stochastic nature, GA "evolves" towards better solutions without being susceptible to pitfalls of convergence to a local minima in our highly non-convex state space. Our optimization variables are the Euler angles $\boldsymbol{\theta^c_l} = [\theta_x, \theta_y, \theta_z]$ in the $xyz$-convention, and the translation $\mathbf{t^c_l} = [x,\ y,\ z]$ from the lidar to camera frame. To restrict the search space for these variables, we obtain an initial estimate $(\mathbf{\tilde{R}^c_l}(\boldsymbol{\tilde{\theta}^c_l})\ , \mathbf{\tilde{t}^c_l})$ of the required rotation and translation as follows.

$$\begin{aligned}
\mathbf{\tilde{R}^c_l N_l} &= \mathbf{N_c} \\
\mathbf{N_l}^T(\mathbf{\tilde{R}^c_l})^T &= \mathbf{N_c}^T \\
\mathbf{N_l N_l}^T(\mathbf{\tilde{R}^c_l})^T &= \mathbf{N_l N_c}^T \\
(\mathbf{\tilde{R}^c_l})^T &= (\mathbf{N_l N_l}^T)^{-1}(\mathbf{N_l N_c}^T) \\
\mathbf{\tilde{R}^c_l} &= ((\mathbf{N_l N_l}^T)^{-1}(\mathbf{N_l N_c}^T))^T
\end{aligned} \tag{2}$$

The notations we use are, $\mathbf{N_l}, \mathbf{N_c}, \mathbf{O_l}$ and $\mathbf{O_c}$, representing 3x$N$ matrices comprising of the column vectors $\mathbf{n^i_l}, \mathbf{n^i_c}, \mathbf{o^i_l}$, and $\mathbf{o^i_c}$, for $i = 1, 2, ..., N$ samples, respectively. To refine $\mathbf{\tilde{R}^c_l}(\boldsymbol{\tilde{\theta}^c_l})$, we first optimize for rotation before jointly optimizing for $\mathbf{R^c_l}(\boldsymbol{\theta^c_l})$ and $\mathbf{t^c_l}$. The components used to form the fitness function for the rotation optimization are:

1) The dot product average between the vector $(\mathbf{o^i_c}$ - $\mathbf{\bar{o}^i_{c1}})$ and $\mathbf{n^i_{l,c}}$ (this value should ideally be 0 if the transformed normal $\mathbf{n^i_{l,c}}$ is perpendicular to the vector $(\mathbf{o^i_c}$ - $\mathbf{\bar{o}^i_{c1}})$ lying on the plane)

$$e_d = \frac{1}{N}\left\{\sum_{i=1}^{N}\left(\left(\mathbf{o^i_c} - \mathbf{\bar{o}^i_{c1}}\right)\cdot \mathbf{n^i_{l,c}}\right)^2\right\} \tag{3}$$

where, $\mathbf{n^i_{l,c}}$ (=$\mathbf{R^c_l n^i_l}$) is $\mathbf{n^i_l}$ in the camera frame.

2) The alignment between $\mathbf{n^i_{l,c}}$ and $\mathbf{n^i_c}$

$$e_r = \frac{1}{N}\left\{\sum_{i=1}^{N}\left(\sqrt{\sum\left(\mathbf{n^i_{l,c}} - \mathbf{n^i_c}\right)^2}\right)\right\} \tag{4}$$

The fitness function combining the above factors is:

$$\mathbf{\tilde{R}^c_l}(\boldsymbol{\tilde{\theta}^c_l}) = \underset{\mathbf{R^c_l}(\boldsymbol{\theta^c_l})}{\operatorname{argmin}}\, e_d + e_r \tag{5a}$$

Once we get a good estimate of $\mathbf{\tilde{R}^c_l}$ in terms of lower values of $e_r$ and $e_d$, we obtain $\mathbf{\tilde{t}^c_l}$; the translation estimate,

$$\mathbf{\tilde{t}^c_l} = mean(\mathbf{O_c} - \mathbf{\tilde{R}^c_l O_l}), \tag{5b}$$

where $mean$ represents the average operation performed row-wise. Knowing, $(\mathbf{\tilde{R}^c_l}, \mathbf{\tilde{t}^c_l})$, we constrict the bounds of the joint search space for $\mathbf{R^c_l}$ and $\mathbf{t^c_l}$. We form the corresponding fitness function by introducing the following factors in addition to $e_d$ and $e_r$.

1) The euclidean distance average between $\mathbf{o^i_{l,c}}$ and $\mathbf{o^i_c}$,

$$e_t = \frac{1}{N}\left\{\sum_{i=1}^{N}\left(\sqrt{\sum\left(\mathbf{o^i_c} - \mathbf{o^i_{l,c}}\right)^2}\right)\right\}, \tag{6}$$

where, $\mathbf{o^i_{l,c}}$ (=$\mathbf{R^c_l o^i_l} + \mathbf{t^c_l}$) is $\mathbf{o^i_l}$ measured in the camera frame.

2) The variance in the euclidean distance between $\mathbf{o^i_{l,c}}$ and $\mathbf{o^i_c}$, in all the samples $N$.

$$v_t = \frac{1}{N}\left\{\sum_{i=1}^{N}\left(\sqrt{\sum\left(\mathbf{o^i_c} - \mathbf{o^i_{l,c}}\right)^2} - e_t\right)^2\right\} \tag{7}$$

The variance component is included to avoid any bias in the euclidean distance for a particular sample. Note: The units of each of the above components are in metres, same as that of all the prerequisite measurements in the calibration process. The range of distance errors in metres are similar to that of rotation errors in radians, eliminating the need for normalizing any variable.

The factors defined in (6) and (7), consider the errors and variance in 3D Cartesian space. To incorporate the errors in the 2D image plane, we minimize the maximum value of the re-projection error between the centre points, $\mathbf{o^i_{l,c}}$ and $\mathbf{o^i_c}$, amongst all the samples:

$$e_{t,I} = max\left\{\sqrt{\sum\left(\mathbf{o^i_{c,I}} - \mathbf{o^i_{l,c,I}}\right)^2}\right\} \tag{8}$$

such that, $i = 1, 2, .., N$. Since the error in a 2D image is in pixels, we convert it into metres in order to be able to add them with the errors obtained previously. This is done by finding the number of pixels $p_l$ lying on the edge of the square located in the middle of the checkerboard. Knowing $s$, we can then find the metre length corresponding to 1 pixel for a particular distance of the checkerboard in the $i^{th}$ sample. We assume this metre correspondence of a pixel to be constant for all the pixels lying close to the board centre. This conversion is hence applied to $e_{t,I}$. The combined fitness function is defined below.

$$(\hat{\mathbf{R}}^{\mathbf{c}}_{\mathbf{l}}(\hat{\boldsymbol{\theta}}^{\mathbf{c}}_{l}), \hat{\mathbf{t}}^{\mathbf{c}}_{\mathbf{l}}) = \underset{\mathbf{R}^{\mathbf{c}}_{\mathbf{l}}(\boldsymbol{\theta}^{\mathbf{c}}_{l}), \mathbf{t}^{\mathbf{c}}_{\mathbf{l}}}{\operatorname{argmin}} \; e_t + v_t + e_d + e_r + k e_{t,I}, \quad (9)$$

subject to:

$$\boldsymbol{\theta}^{\mathbf{c}}_{l} \in \tilde{\boldsymbol{\theta}}^{\mathbf{c}}_{l} \pm \pi/18$$
$$\mathbf{t}^{\mathbf{c}}_{\mathbf{l}} \in \tilde{\mathbf{t}}^{\mathbf{c}}_{\mathbf{l}} \pm 0.05$$

In (9), $k (= \frac{s}{p_l})$ is the error conversion from pixel to metre and $(\boldsymbol{\theta}^{\mathbf{c}}_{l}, \mathbf{t}^{\mathbf{c}}_{\mathbf{l}})$ is bound to vary within $\pm\ \pi/18$ radians or 10 degrees from $\tilde{\boldsymbol{\theta}}^{\mathbf{c}}_{l}$ and 0.05 metres from $\tilde{\mathbf{t}}^{\mathbf{c}}_{\mathbf{l}}$ respectively. As $(\tilde{\boldsymbol{\theta}}^{\mathbf{c}}_{l}, \tilde{\mathbf{t}}^{\mathbf{c}}_{\mathbf{l}})$ have been refined in (5a) and (5b), we can set a small bound.

## IV. EXPERIMENTAL RESULTS

To test the robustness of our algorithm, we conduct experiments with simulated and real data. Furthermore, we verify our results by visually inspecting the projection of the point cloud on the image with two different sensor configurations.

For all the experiments, we initiate GA with a population size of 200 lying within the variable bounds of (9). Due to the non-deterministic nature of GA, we run the process 10 times for a given sample set, $N$, and take the average of the obtained extrinsic parameters. The average value is considered as the final extrinsics obtained from that particular $N$. Our algorithm took between 2-10 minutes for $N$ ranging from 3-30.

### *A. Simulated Data*

In order to perform the calibration in a simulated scenario, we generate the sensor frames with an arbitrary transform and place the board (board centre) in different possible locations in 3D space. This is done making sure that the board is visible to both sensors. Next, we vary the orientation of the board around a randomly chosen axis. Since we start with a known transform of both sensor frames relative to the world frame, we can find the features of interest as measured by both sensors. However, sensor measurements (especially the lidar) have errors involved in them. Due to this, the lidar data points from a static scene differ in each time step. By observing this variation, we note that the range of the board normal and board centre, results to around $4^\circ$ and 1 cm respectively. This forms the basis for generating the noise in the simulated features as obtained from the lidar. We add three different noise levels to the board normal, $\pm\ 1.5^\circ$, $\pm\ 2^\circ$, and $\pm\ 2.5^\circ$. The normal vectors have a Gaussian distribution with the ground truth normal as the mean, deviating up to the noise level. The centre point is displaced within a sphere of 1 cm diameter, with the ground truth as the sphere centre. This displacement has a Gaussian distribution across samples. We do not add any noise to the features obtained by the camera as we just require noisy data, be it from the camera or the lidar.

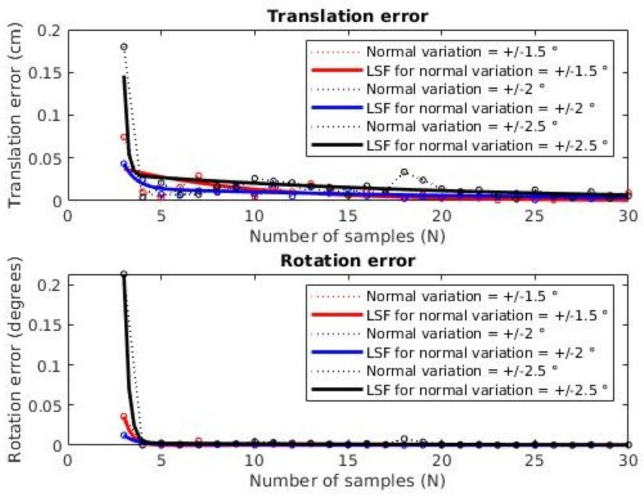

Fig. 4. Translation and rotation error for simulated data.

We run the optimizer starting with an input of 3 random samples, the minimum number required by our algorithm. As the ground truth $(\mathbf{R}^{\mathbf{c}}_{\mathbf{l}}, \mathbf{t}^{\mathbf{c}}_{\mathbf{l}})$ and estimated transform $(\hat{\mathbf{R}}^{\mathbf{c}}_{\mathbf{l}}, \hat{\mathbf{t}}^{\mathbf{c}}_{\mathbf{l}})$ are known, we can calculate the translation and rotation error [21] as follows:

$$e_{translation} = \|\mathbf{t}^{\mathbf{c}}_{\mathbf{l}} - \hat{\mathbf{t}}^{\mathbf{c}}_{\mathbf{l}}\| \quad (10)$$
$$e_{rotation} = \|\mathbf{I} - (\mathbf{R}^{\mathbf{c}}_{\mathbf{l}})^{-1}\hat{\mathbf{R}}^{\mathbf{c}}_{\mathbf{l}}\|_F \quad (11)$$

where, $\|.\|$ denotes the norm operation and $\|.\|_F$ is the Frobenius norm. Fig. 4 shows the errors in rotation and translation as more samples get introduced into the optimizer. We note that some samples can be relatively more noisy when compared to the rest. This is due to the way the board is placed relative to the lidar, the number of scan lines passing through it and the corresponding plane model estimation errors. So depending upon the quality of the incoming samples, the rotation and translation errors might increase or decrease relative to the error from the previous sample set. However, the overall tendency of the error is to decay exponentially as more samples are added. This is represented by taking the Least Square Fit (LSF) through the data points. It can be noticed in Fig. 4 that the red colored LSF curve, representing a lower noise of $\pm\ 1.5^\circ$ when compared to $\pm\ 2^\circ$ by the blue colored LSF curve, eventually surpasses the blue LSF curve and leads to a lower transformation error in comparison. The green LSF curve, corresponding to a $\pm\ 2.5^\circ$ noise shows a higher convergence error in comparison. Nonetheless, the error converges to less than 0.5 cm for translation and close to 0 degree error for rotation with a high sample number. After $N \approx 9$, the error tends to become constant across samples. Therefore, we do our analysis upto 9 samples with the real data.

## B. Real Data

We obtain real data, i.e. the point cloud corresponding to the board and the image of the checkerboard, from Velodyne's VLP-16 lidar and NVIDIA's 2Mega SF3322 automotive GMSL camera. The sensor setup for data collection is shown in Fig. 5 and we specifically calibrate the camera located in the centre and the lidar. VLP-16 is a 16 beam lidar with a $360°$ horizontal field of view (FOV) and $\pm 15°$ vertical FOV. The range accuracy is $\pm 3cm$. The GMSL camera lens has a $100°$ horizontal FOV and $60°$ vertical FOV. Images can be captured at 30 frame per second (fps) with the resolution of $1928 \times 1208$ (2.3M pixel).

Fig. 5. Sensor setup.

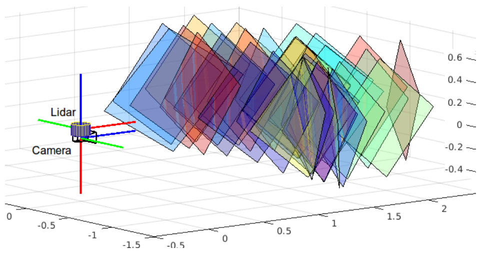

Fig. 6. Real samples used to evaluate our calibration algorithm

To evaluate our approach with real data, we collect a set of 30 input samples as shown in Fig. 6. We begin by randomly choosing 100 sets out of the different possible combinations of 3 samples and feed them into the optimizer. Thereafter, we increment the samples for which the combinations are obtained. In Fig. 7, we show the distribution of 100 randomly chosen extrinsic parameters obtained from the combinations of 3, 4 and 9 samples. The width of each bin for translation and rotation is 0.5 centimetre and $0.5°$ respectively. It can be observed that as N increases, the spread in the extrinsic parameters decrease. This is indicative of the robustness of our algorithm across samples. The outliers in the histogram can be attributed to an improper/noisy sample set, which decreases as N is incremented.

For a qualitative analysis, we project the point cloud on the image as shown in Fig. 8. In this image, the color of the points vary relative to the distance between the obstacle and the lidar. Assuming that the intrinsics are correct, an accurate extrinsic calibration would imply a projection such that there is a visually evident correspondence between the boundaries of objects in the point cloud and the edges of objects in the image. We specifically chose to visualize the projection of the laser points to the entire image and not just the board. This is because the highly non-linear fitness function can lead to an accurate projection of points on the board, i.e. near the sample point used in the optimizer, but not elsewhere in the image due to over-fitting. We can see this happening in Fig. 9, where the board projection is visually accurate when $N = 3, 4$, and 9, but the overall projection improves as $N$ increases. Also, the trend of the graph observed in

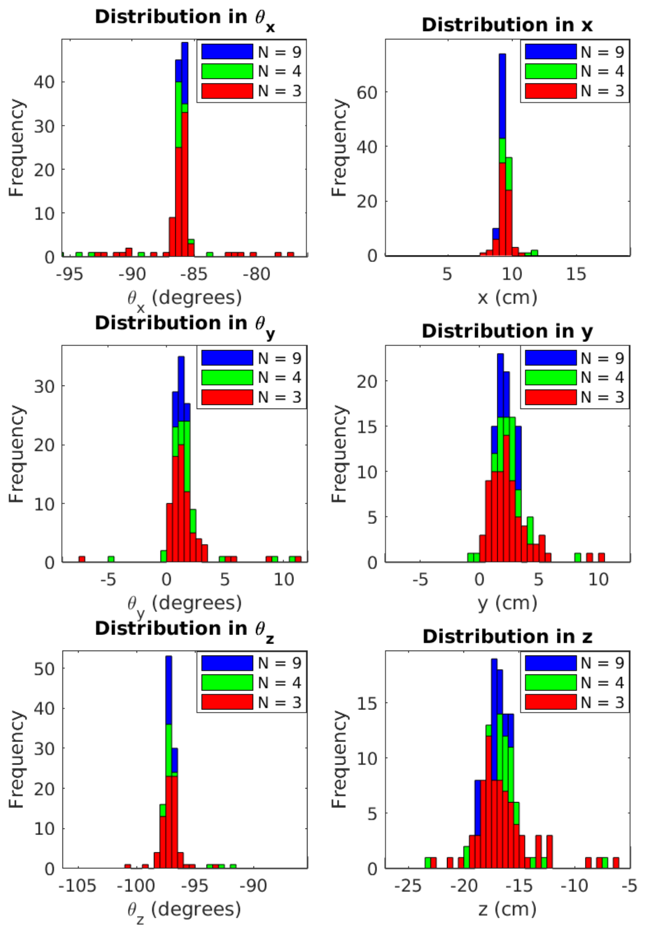

Fig. 7. Histogram plots implying a low spread in the calibration parameters.

Fig. 4 is evident in the projection where there is significant improvement from $N = 3$ to $N = 4$ when compared $N = 4$ to $N = 9$.

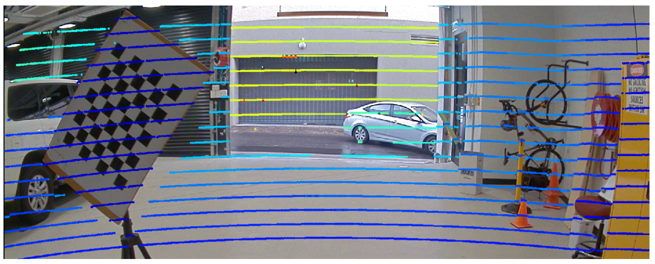

Fig. 8. Point cloud projection on an image for 9 samples.

When closely observing the calibration boards in Fig. 9, we notice that the board lidar projection extends slightly beyond the board edges in each image. This can be attributed to the spot size of the laser points. According to the VLP-16 data sheet, the laser spot size at around 2 metres, the distance from the lidar at which the board is kept, is 18.2 millimetres in the horizontal direction. Assuming that the spot centre falls at the board edge, this is an error of around

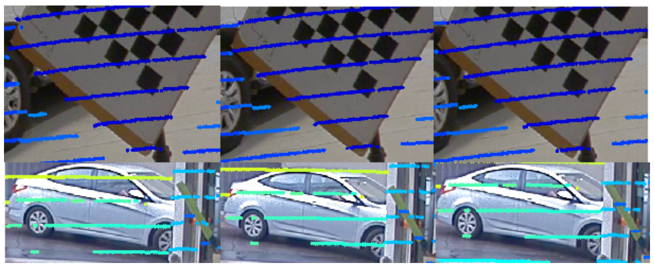

Fig. 9. Projection for N = 3, N = 4, N = 9.

9 millimetres, equivalent to $6-7$ pixels, in the horizontal direction. We note that as this error is equal for each edge of the board, it does not affect the position of the board centre. This reinforces the appropriateness of choosing the centre point in comparison to other more noisy features such as the board corners or its edges.

## V. CONCLUSIONS AND FUTURE WORK

In this paper we presented a robust and automated approach to estimate the extrinsic calibration parameters between a pinhole/fisheye camera and 3D lidar using a planar checkerboard. For this, we chose the most stable features based on the errors in the lidar measurements (compared to other features), to obtain the 3D point and plane correspondences. Our method automatically extracted these features for calibration and GA was used to obtain a globally optimal calibration result.

We demonstrated experimentally that our method is able to obtain consistent results which improve as more samples are added into the optimizer. Occasional outliers can be attributed to the measurement error and quality of the sample. An analysis with the simulated data and the image projection made us conclude that a globally optimal solution can be achieved by adding additional samples (N = 9 or 10), beyond the minimum requirement of 3, into the optimizer.

In future, we plan to synchronize the sensor data and apply motion correction to the lidar scans so that the mobile platform can be moved around the calibration target to collect different samples and obtain the extrinsic parameters at run-time. This could be very useful in the production line for autonomous vehicles or other robots.

## ACKNOWLEDGMENT

This work has been funded by the ACFR and the Australian Research Council Discovery Grant DP160104081 and University of Michigan / Ford Motors Company Contract “Next generation Vehicles”.

## REFERENCES

[1] J. S. Berrio, W. Zhou, J. Ward, S. Worrall, and E. Nebot, “Octree map based on sparse point cloud and heuristic probability distribution for labeled images,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 3174–3181.

[2] F. M. Mirzaei, D. G. Kottas, and S. I. Roumeliotis, “3d lidarcamera 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.

[3] A. Dhall, K. Chelani, V. Radhakrishnan, and K. M. Krishna, “LiDAR-Camera Calibration using 3D-3D Point correspondences,” *ArXiv e-prints*, May 2017.

[4] S. A. R. Florez, V. Frémont, and P. Bonnifait, “Extrinsic calibration between a multi-layer lidar and a camera,” in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, MFI 2008, Seoul, South Korea, August 20-22, 2008*, 2008, pp. 214–219. [Online]. Available: https://doi.org/10.1109/MFI.2008.4648067

[5] X. Gong, Y. Lin, and J. Liu, “3d lidar-camera extrinsic calibration using an arbitrary trihedron,” *Sensors (Basel, Switzerland)*, vol. 13, pp. 1902–18, 02 2013.

[6] D. Scaramuzza, A. Harati, and R. Siegwart, “Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes,” *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4164–4169, 2007.

[7] L. Zhou, Z. Li, and M. Kaess, “Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences,” in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS*, October 2018.

[8] Velodyne, “Vlp-16 user manual,” https://usermanual.wiki/Pdf/VLP16Manual.1719942037.pdf, San Jose, California, USA, 2018, accessed: 2019-04-10.

[9] Q. Zhang and R. Pless, “Extrinsic calibration of a camera and laser range finder (improves camera calibration),” vol. 3, 01 2004, pp. 2301 – 2306 vol.3.

[10] G. Pandey, J. McBride, S. Savarese, and R. Eustice, “Extrinsic calibration of a 3d laser scanner and an omnidirectional camera,” 01 2010.

[11] L. Zhou and Z. Deng, “Extrinsic calibration of a camera and a lidar based on decoupling the rotation from the translation,” in *2012 IEEE Intelligent Vehicles Symposium*. IEEE, 2012, pp. 642–648.

[12] W. Dong and V. Isler, “A novel method for extrinsic calibration of a 2-d laser-rangefinder and a camera,” *CoRR*, vol. abs/1603.04132, 2016. [Online]. Available: http://arxiv.org/abs/1603.04132

[13] Z. Taylor and J. M. G. Nieto, “Automatic calibration of lidar and camera images using normalized mutual information,” 2012.

[14] R. Unnikrishnan and M. Hebert, “Fast extrinsic calibration of a laser rangefinder to a camera,” 01 2005.

[15] L. Zhou, “A new minimal solution for the extrinsic calibration of a 2d lidar and a camera using three plane-line correspondences,” *IEEE Sensors Journal*, vol. 14, no. 2, pp. 442–454, Feb 2014.

[16] A. Geiger, F. Moosmann, mer Car, and B. Schuster, “Automatic camera and range sensor calibration using a single shot,” in *International Conference on Robotics and Automation (ICRA)*, St. Paul, USA, May 2012.

[17] M. A. Fischler and R. C. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981. [Online]. Available: http://doi.acm.org/10.1145/358669.358692

[18] Itseez, “Open source computer vision library,” https://github.com/itseez/opencv, 2015.

[19] Y. Zheng, Y. Kuang, S. Sugimoto, K. strm, and M. Okutomi, “Revisiting the pnp problem: A fast, general and optimal solution,” in *2013 IEEE International Conference on Computer Vision*, Dec 2013, pp. 2344–2351.

[20] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1998.

[21] D. Q. Huynh, “Metrics for 3d rotations: Comparison and analysis,” *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, Oct 2009. [Online]. Available: https://doi.org/10.1007/s10851-009-0161-2