COUPLED: Calibration of a LiDAR and Camera Rig Using Automatic Plane Detection

Omar Montoya[1], Octavio Icasio[2], and Joaquín Salas[1](✉)

[1] CICATA Querétaro, Instituto Politécnico Nacional, Cerro Blanco 141, Cimatario,
76030 Querétaro, México
jsalasr@ipn.mx

[2] Centro Nacional de Metrología, Querétaro, México

Abstract. LiDARs and cameras are two widely used sensors in robotics and computer vision, particularly for navigation and recognition in 3D scenarios. Systems combining both may benefit from the precise depth of the former and the high-density information of the latter, but a calibration process is necessary to relate them spatially. In this paper, we introduce COUPLED, a method that finds the extrinsic parameters to relate information between them. The method implies the use of a setup consisting of three planes with charuco patterns to find the planes in both systems. We obtain corresponding points in both systems through geometric relations between the planes. Afterward, we use these points and the Kabsch algorithm to compute the transformation that merges the planes between both systems. Compared to recent single plane algorithms, we obtain more accurate parameters, and only one pose is required. In the process, we develop a method to automatically find the calibration target using a plane detector instead of manually selecting the target in the LiDAR frame.

1 Introduction

In the field of mobile robotics, sensors to understand the world around them are required. Two commonly used sensors are cameras and LiDARs (*Light Detection and Ranging*). Cameras allow proper feature extraction, but it is challenging to obtain depth information from a single camera. In contrast, LiDARs provide precise spatial details but at a much lower density, which makes feature extraction difficult. In our research, we study systems that combine both technologies to take advantage of their strengths. Nonetheless, one of the main challenges is that while each sensor outputs spatial data in its reference frame, typically we process their information in a common one, which requires a calibration process.

To calibrate a camera with a LiDAR, one usually finds corresponding points in both systems, which we use to calculate a transformation that relates them. Often, one uses the corner points of a checkerboard on a rectangular board. The LiDAR can capture the boards pose by finding its edges, and the camera can

K. M. Figueroa Mora et al. (Eds.): MCPR 2020, LNCS 12088, pp. 209–218, 2020.
https://doi.org/10.1007/978-3-030-49076-8_20

estimate the pose of the checkerboard. These methods typically suffer from noise in using edge points to find the corners in the LiDAR [3,15]. Also, even though finding the target in the camera is easy, finding the target in the LiDAR frame is relatively more complicated. LiDARs typically use an array of angled lasers rotating over an axis to generate a point cloud that is denser in the horizontal direction than in the vertical one. This configuration makes it difficult to find the target with conventional plane detection methods. Thus, most calibration methods either manually find the target and isolate it [3,10], which can be time-consuming when making acquisitions to reduce calibration error. Alternatively, other methods physically isolate the target from other objects to facilitate plane detection [15], which is not ideal because we would need to reserve a dedicated area exclusively for calibration.

In this paper, we introduce COUPLED, a method for calibrating a LiDAR-camera rig that uses three planes. In the process, we develop an approach to automatically find the target in the LiDAR reference frame using a plane detector. Combining both, we develop a system for automatic calibration, no longer requiring a dedicated area or manually selecting the target for the LiDAR frame. We divide the rest of the paper into a review of related literature, followed by our methodology, then our experiments, and finally, a conclusion summarizing the relevant findings.

2 Related Literature

We review the literature on two fronts: Methods for the automatic detection of planar surfaces and the calibration of a LiDAR-camera rig.

Plane Detection with LiDAR. Finding planes within 3D point clouds is a fundamental step for complex algorithms. The introduction of LiDAR and other light-based distance sensors has made this activity vital. Usually, when working with dense homogenous point clouds, Hough transform and RANSAC are tried and true approaches [1,16]. However, the mechanical workings of certain LiDAR and MLS (*Mobile Laser Scanning*) sensors do not output point clouds favorable for these techniques, so novel methods are required.

Commonly, LiDARs have an array of angled lasers rotating on an axis from point clouds that are sparse on its vertical axis relative to its horizontal axis. The path traced by any of the lasers forms a cone, and when a plane intersects the cone, we get a conic section. Grant *et al.* [7] segments the LiDAR lines into conic sections and then uses a Hough transform where they accumulate the planes that can be formed by each line, and the planes with enough votes are taken. Another sensor that presents problems with classic plane detection methods is MLS since it usually has its laser head perpendicular to the trajectory of a carrier vehicle. Nguyen *et al.* [13] uses a similar approach segmenting scan points into straight lines, then parallel lines are grouped, and the singular vectors for each group are obtained to determine whether the grouped lines form a plane.

LiDAR-Camera Rig Calibration. In mobile robotics, combining sensors such as multiple cameras or cameras and LiDAR has become common [2,5,8]. Many techniques have been developed to accomplish this task. Single checkerboards are the most usually involved [10,15] but also researchers have developed techniques using different kinds of targets [3,11] or no targets at all [9,12].

Practitioners have created variations using a single checkerboard. For instance, Verma *et al.* [15] detect the centroid of the target, in at least three poses, and its normal, then they use genetic algorithms to obtain the parameters. Kim *et al.* [10] proposed another technique that requires three poses. In their case, they employ the normals and use an energy function to calculate the rotation and the translation that minimizes the distance to the camera and the LiDAR's plane. Kümmerle *et al.* [11] use a spherical target because its center can be more accurately extracted relative to checkerboards. Chai *et al.* [3] use a cube with aruco markers printed on its sides. The LiDAR detects the three sides, and the aruco markers are used to detect the poses of the sides in the camera.

Researchers have developed techniques that require no target at all to allow calibration when no target is available. For instance, Kang *et al.* [9] use edge detection in both the camera and LiDAR to find the transformation between both sets of edges. Similarly, Nagy *et al.* [12] use structure from motion to create a 3D point cloud for the camera reference frame. Then, they detect objects in both systems by grouping near points.

3 Method

Calibrating our LiDAR-camera rig means finding the parameters of a rotation matrix $\boldsymbol{R}_c^L \in SO(3)$ and translation vector $\boldsymbol{t}_c^L \in \mathbb{R}^3$ that transforms a 3D point set $\boldsymbol{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_m\}$ in the camera reference frame into $\boldsymbol{Q} = \{\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_m\}$, its corresponding in the LiDAR reference frame, as $\mathbf{R}_c^L \mathbf{p}_i + \mathbf{t}_c^L = \mathbf{q}_i, \quad \forall\, i = 1, \ldots, m$. Because of noise in the sensors, a perfect solution normally does not exist, so we instead minimize the sum of squares as

$$(\mathbf{R}_c^L, \mathbf{t}_c^L) \leftarrow \arg\min_{\boldsymbol{R}, \mathbf{t}} \sum_{i=1}^{m} \|\mathbf{R}\boldsymbol{p}_i + \mathbf{t} - \boldsymbol{q}_i\|^2. \tag{1}$$

3.1 Calibration Pattern

First, we need a target for which we can find corresponding points in the LiDAR and the camera. We propose to use a pattern that consists of three planes with charuco boards [6]. We use OpenCV to find the planes by detecting the charucos for the camera, and plane fitting to find the three planes in the LiDAR. However, to find precise point to point correspondences, we use geometric relations such as the vertex common to the three planes, and the lines formed by the intersection of a pair of planes, as there are three pair combinations we obtain three lines. With this, we can obtain four point correspondences and proceed to obtain the parameters minimizing (1).

To compute the pose of our target planes in the camera reference frame, we use charuco markers printed on our pattern. Our design combines aruco markers and checkerboards into charuco [6]. It uses the aruco markers to interpolate the corners of the checkerboards, and since each aruco has a unique ID, each checkerboard corner can also be identified even if part of the checkerboard is occluded. Then, we refine the position of the checkerboard corners with subpixel accuracy. Finally, we compute the pose of the board, as with regular checkerboards, using PnP (Fig. 1).

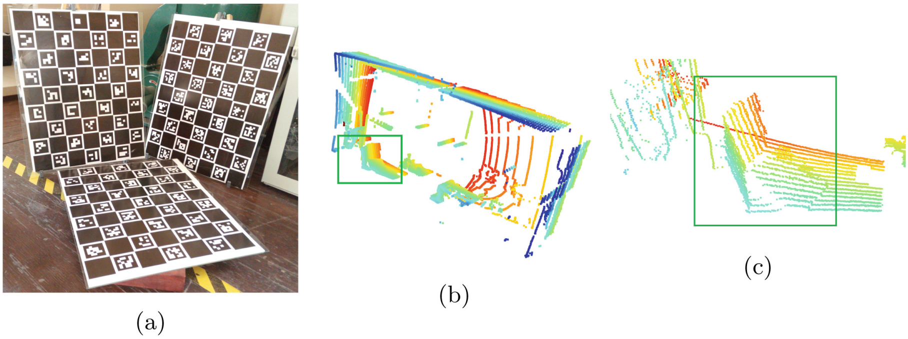


Fig. 1. Calibration pattern. (a) Each plane has a different set of charuco markers for the camera to detect. (b) The full point cloud: The calibration target is enclosed in the green rectangle. (c) Zooming in on the target in the previous point cloud: The calibration pattern is enclosed in the green rectangle.

3.2 Automatic Target Detection for LiDAR

To find the target in the LiDAR reference frame, we developed an algorithm to detect the planes in the point cloud sensed by the LiDAR and segment the three planes belonging to our target. There exist many methods to find planes within a point cloud [1,16], but for some LiDARs, with few lasers, the problem is particularly challenging. Take, for instance, the VLP-16 sensor, which consists of 16 laser range finders spinning along an axis, making the density of points much higher horizontally than vertically. This arrangement causes problems with RANSAC and Hough transform-based plane finders, when the best fits for a plane erroneously end up being horizontal planes that contain all the points of a single beam. Note that rotating laser LiDAR beams form cones as they spin. When a cone is intersected by a plane, a conic section is formed. This observation is the basis of our algorithm, which consists in finding the curves, grouping them into planes and segmenting the target from the planes found.

Curve Finding. The first step for our plane detection is finding the conic sections. Because fitting such a high number of points to a conic section equation is computationally intensive, we use an approximation and instead look for smooth curves. We start by sampling each of the lasers separately.

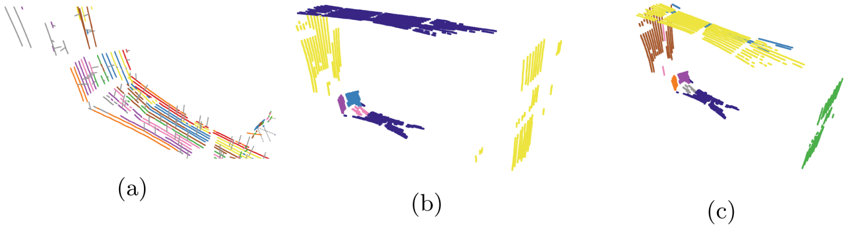


Fig. 2. Illustration of the steps of the plane detection process. (a) The point cloud is segmented into smooth curves with a normal vector assigned to them. (b) We group the lines by similar normals. We end up with groups containing one or more parallel planes. (c) We use RANSAC plane fitting to segment the points to their appropriate planes.

First, we take a single beam and treat its points as a 1D signal ordered by the azimuth given by the LiDAR, which starts at positive Y and runs along the Z axis clockwise. Then, we apply a spatial Gaussian filter with a standard deviation σ to reduce the noise in the point cloud. Now, to segment parts of the 1D signal that have a smooth trajectory, we apply a custom kernel $\mathbf{k}$ that was made empirically to simulate a Laplacian to help detect rapid changes in the signal. We run the kernel through the signal and threshold to highlight where we separate the curves. We also discard segments that have less than a certain number of points p to avoid short curves. Finally, we combine curves by verifying whether their direction is similar, and they are close enough together.

Grouping Plane Proposals. Once we have curves, we need to find a way to group lines belonging to the same plane. To do this, we take a random line, and the nearest line to it. We calculate the principal components for the group of points of the two curves via SVD (*Singular Value Decomposition*). Then, we test whether the ratio between the first and second singular values is below a certain threshold r_{12} to avoid planes formed from very narrow lines, which can be noisy. Next, we test wether the ratio r_{23} between the second and third components is large to determine that the points form a valid flat plane. If both tests are passed, the normal vector of the plane is saved in an accumulator. We repeat this process for the lines within a certain maximum distance $l_{\max}$ from our original line. Finally, we assign the normal vector with the most votes. We do this for each line, so the lines end up with a normal vector assigned. We then group lines by normal vectors in the same direction using DBSCAN [4] (see Fig. 2). After this process, we have groups of lines likely belonging to one or more planes with the same normal as the lines. We can now easily segment applying RANSAC on each group.

Finding the Target. We have now extracted the planes in our point cloud, but for our application, we need the planes that correspond to our target. To isolate these planes, we use OpenCV to calculate the pose of our three target planes in the camera frame. We also calculate the center of the target planes. So now, we can calculate the angles between the normals and the distances between centroids of any two planes of the target. With this, we end up with three angles and three distances that we use as a descriptor for the target. Finally, we use brute force matching to find the combination of three planes found by our detector that best matches the descriptor of our target.

3.3 Camera and LiDAR Calibration

Once we have our target in the LiDAR and the cameras, we get the poses in the camera reference frame using the charucos. OpenCV gives us a translation and rotation from a reference frame with its origin on one of the outer most corners of the board, and its X and Y axis aligned with the sides of the board to the camera frame [6]. We obtain a vector normal to the board in the camera reference frame using $\mathbf{n}_i^c = \mathbf{R}_r^c \mathbf{e}_z$, where $\mathbf{n}_{ci}$ is the vector normal to plane i in the camera frame, $\mathbf{R}_r^c$ is the rotation from the charuco i frame to the camera frame, and $\mathbf{e}_z^T = (0, 0, 1)$. With this rotation, we obtain the normals for the planes containing the targets. For the LiDAR, we get the principal components of the planes obtained in the previous section.

Next, we calculate the vertex between the three planes and the three intersections from two planes. The intersections can be defined as vectors of unitary magnitude. Now, we have four corresponding points to obtain the extrinsic parameters, P points in the camera frame, and Q points in the LiDAR frame. We use Kabsch algorithm [14] to obtain the rotation and translation from the camera to the LiDAR that minimizes the mean square error expressed in (1).

We call COUPLED to our method to calibrate the LiDAR-camera rig using automatic plane detection.

4 Experimental Results

To test our algorithm, we took 40 readings with our system, moving the LiDAR-camera rig to different positions, making sure that the target remains within the fields of view of both sensors. To show the relevance of COUPLED, we compare our results with a method recently introduced.

4.1 Experimental Setup

For our experiments, we used a Velodyne VLP/16 LiDAR. It consists of 16 lasers operating at a wavelength of 903 nm and turning at a programmable rotational speed between 5 and 20 Hz. With a range of 100 m, it has a range accuracy of ±3 cm and a vertical field of view of ±15°. Also, we employed a MicaSense Parrot Sequoia multispectral camera, which weighs about 72 g (the additional sunshine sensor weights 35 g). It produces images with spectral response peaking at 550 nm (green), 660 nm (red), 735 nm (red edge), and 790 nm (near infrared). Each of these images has a spatial resolution of 1,280 (horizontal) × 960 (vertical) pixels.

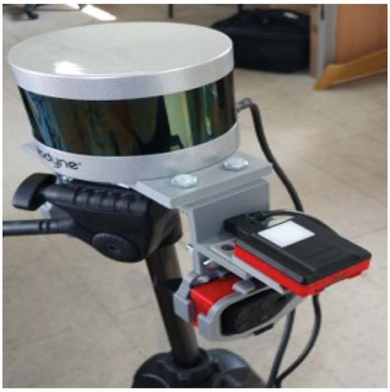


Fig. 3. Our LiDAR-camera rig, consisting of a VLP-16 LiDAR and a Sequoia Multi-spectral camera. (Color figure online)

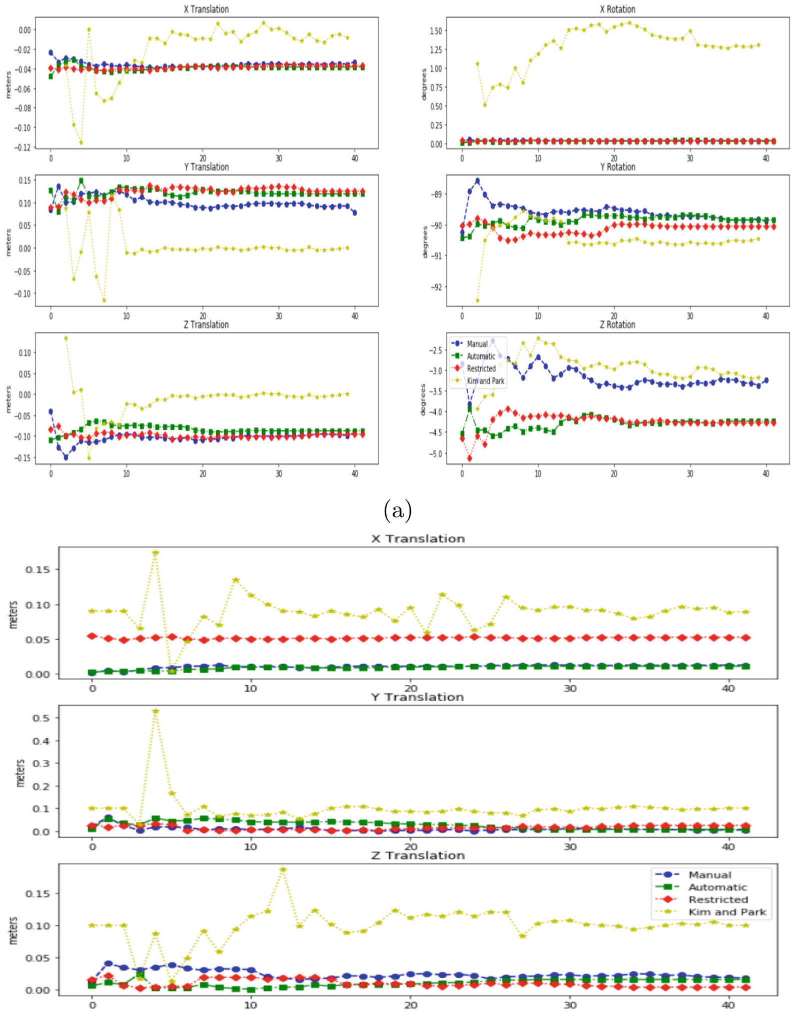
Also, the Sequoia includes an RGB color camera with a resolution of 4,068 (h) × 3,456 (v) pixels resolution. We print the charucos targets on 3 mm thick mirror surfaces measuring 0.5 m per side each (Fig. 3).

In our experiments, the Gaussian filter has a spread of $\sigma = 0.08$. We discard segments with fewer than $p = 5$ points. Also, we set the distance between end-points $d = 10$ cm. In addition the angle between them has to be $\alpha < 5°$. To filter out elongated segments, we set $r_{12} = 1/80$ and $r_{23} = 1/200$. Finally, we define the maximum distance between lines at 40 cm.

4.2 LiDAR-Camera Rig Calibration

With the data, we used three procedures to find the calibration target. In the first one, we manually selected the three planes of our target from the LiDAR point cloud. In the second one, we automatically found the target by introducing the entire point cloud to our plane detector and finding the best three planes (the method introduced in this paper). In the third procedure, we limited the point cloud to an azimuth between 230° and 330° and automatically find the planes within the limited point cloud (we call this procedure *restricted*). We know the target is in this azimuth because this direction is required in our rig for the camera to see the target.

We calculated the extrinsic parameters for the 40 views using the three procedures described. Moreover, to verify the effect of using multiple views, we calculate the accumulated extrinsic parameters in which we use the Kabsch algorithm on the points from the concatenation of the current frame points and the previous frames. Although our target detection algorithm finds the correct target for most of the frames, there are still outliers. We eliminated the outliers by using the DBSCAN clustering algorithm on the translation vectors, knowing that the correct translation will form the largest cluster. After the clustering step, 10 out of the 40 frames were discarded in the fully automatic procedure, while only 4 out of 40 were discarded in the restricted procedure.



(a)

(b)

Fig. 4. Results of 40 calibrations (best seen in color). The blue, green, red and yellow lines show when we manually select the LiDAR planes, give the entire point cloud to the target detector, provided the target detector a reduced azimuth point cloud knowing that the target is within that cloud, and for Kim *et al.* [10] method for comparison. (a) Calibration results for accumulated acquisitions to demonstrate convergence. (b) Plotted error from the accumulated calibration.

We compared our method against Kim *et al.* [10] who uses a checkerboard and fits a normal vector to his target, but only uses one plane. In Fig. (4a), we plot the results of accumulated acquisitions. We can see that the estimation of rotation is similar, deviating by a maximum of three degrees. At the same time, the translation differs by as much as 12 cm and does not converge after 40 acquisitions. In Fig. (4b), we plot the error between the calculated translation and the one we measured, we can see that our method allows us to obtain extrinsic parameters with a single acquisition compared with the three acquisitions required by Kim *et al.*'s method and is more accurate. Both methods run in a couple of seconds.

5 Conclusion

In this paper, we present COUPLED, a method to obtain the extrinsic parameters of a LiDAR-camera rig that uses three planes with charuco boards printed on them. We use the charuco boards to find the pose of the planes and a plane detection algorithm to find the planes in the LiDAR frame. Our experimental results show its benefits when compared with the manual selection of the target or the region of interest.

By using multiple frames, we can refine the extrinsic parameter calibration. Here, using the automatic target detector can be very beneficial because it allows us to quickly find our target in multi-capture settings instead of manually selecting it. However, notice that using the automatic detector with a restricted azimuth is an excellent middle ground as it only requires the input of the azimuth, and it has a better performance than the plane detector on the full point cloud. We also presented a new method for plane detection in point clouds generated by LiDARs. Rotational LiDARs have a much higher horizontal density compared to vertical density, which makes it challenging to use traditional methods such as RANSAC and Hough Transform. Our system combines both methods, allowing for automatic calibration by using the plane detector to find the target in the LiDAR frame as opposed to manually selecting it. This permits smooth refinement since we can quickly capture many frames and compute the extrinsic parameters using the points from the different frames.

References

1. Borrmann, D., Elseberg, J., Lingemann, K., Nüchter, A.: The 3D Hough Transform for plane detection in point clouds: a review and a new accumulator design. 3D Res. **2**(2), 3 (2011). https://doi.org/10.1007/3DRes.02(2011)3
2. Caltagirone, L., Bellone, M., Svensson, L., Wahde, M.: LiDAR-camera fusion for road detection using fully convolutional neural networks. Robot. Auton. Syst. **111**, 125–131 (2019)
3. Chai, Z., Sun, Y., Xiong, Z.: A novel method for LiDAR camera calibration by plane fitting. In: International Conference on Advanced Intelligent Mechatronics, pp. 286–291 (2018)

4. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, vol. 96, pp. 226–231 (1996)
5. Gao, H., Cheng, B., Wang, J., Li, K., Zhao, J., Li, D.: Object classification using CNN-based fusion of vision and LiDAR in autonomous vehicle environment. IEEE Trans. Ind. Inform. **14**(9), 4224–4231 (2018)
6. Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., Marín-Jiménez, M.: Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recogn. **47**(6), 2280–2292 (2014)
7. Grant, S., Voorhies, R., Itti, L.: Finding planes in LiDAR point clouds for real-time registration. In: RSJ International Conference on Intelligent Robots and Systems, pp. 4347–4354 (2013)
8. Hwang, S., Kim, N., Choi, Y., Lee, S., Kweon, I.: Fast multiple objects detection and tracking fusing color camera and 3D LiDAR for intelligent vehicles. In: International Conference on Ubiquitous Robots and Ambient Intelligence, pp. 234–239. IEEE (2016)
9. Kang, J., Doh, N.: Automatic Targetless camera-LiDAR calibration by aligning edge with Gaussian mixture model. J. Field Robot. **37**(1), 158–179 (2020)
10. Kim, E.S., Park, S.Y.: Extrinsic calibration of a camera-LiDAR multi sensor system using a planar chessboard. In: International Conference on Ubiquitous and Future Networks, pp. 89–91 (2019)
11. Kümmerle, J., Kühner, T., Lauer, M.: Automatic calibration of multiple cameras and depth sensors with a spherical target. In: International Conference on Intelligent Robots and Systems, pp. 1–8 (2018)
12. Nagy, B., Kovács, L., Benedek, C.: Online targetless end-to-end camera-LiDAR self-calibration. In: International Conference on Machine Vision Applications, pp. 1–6 (2019)
13. Nguyen, H., Belton, D., Helmholz, P.: Planar surface detection for sparse and heterogeneous mobile laser scanning point clouds. J. Photogramm. Remote Sens. **151**, 141–161 (2019)
14. Sorkine-Hornung, O., Rabinovich, M.: Least-squares rigid motion using SVD. Computing **1**(1), 1–5 (2017)
15. Verma, S., Berrio, J.S., Worrall, S., Nebot, E.: Automatic extrinsic calibration between a camera and a 3D Lidar using 3D point and plane correspondences. In: Intelligent Transportation Systems Conference (ITSC), pp. 3906–3912. IEEE (2019)
16. Zeineldin, R.A., El-Fishawy, N.A.: A survey of RANSAC enhancements for plane detection in 3D point clouds. Menoufia J. Electron. Eng. Res. **26**(2), 519–537 (2017)