2017 IEEE International Conference on Computer Vision Workshops

Accurate Calibration of LiDAR-Camera Systems using Ordinary Boxes

Zoltan Pusztai
Geometric Computer Vision Group
Machine Perception Laboratory
MTA SZTAKI, Budapest, Hungary
zoltan.pusztai@sztaki.mta.hu

Levente Hajder
Geometric Computer Vision Group
Machine Perception Laboratory
MTA SZTAKI, Budapest, Hungary
levente.hajder@sztaki.mta.hu

Abstract

This paper deals with the calibration of a visual system, consisting of RGB cameras and 3D Light Detection And Ranging (LiDAR) sensors. Registering two separate point clouds coming from different modalities is always challenging. We propose a novel and accurate calibration method using simple cardboard boxes with known sizes. Our approach is principally based on the detection of box planes in LiDAR point clouds, thus it can calibrate different LiDAR equipments. Moreover, camera-LiDAR calibration is also possible with minimal manual intervention. The proposed algorithm is validated and compared to state-of-the-art techniques both on synthesized data and real-world measurements taken by a visual system consisting of LiDAR sensors and RGB cameras.

1. Introduction

Nowadays it is more and more important to acquire information of our environment. This is a significant task especially in the case of autonomous cars and robots which have to control themselves without any human interaction. 3D machine perception can be done in several ways, e.g. with the help of cameras, microphones, radars, scanners, just to mention a few possibilities.

One of the most favorite technology of today is the 3D Light Detection And Ranging (LIDAR) which can measure our surroundings by obtaining a sparse point cloud based on the distances measured by light beams. The main benefit of 3D LiDAR technology is the active illumination which works independent of ambient light. It can be used in any lighting conditions and LiDAR sensors can accurately map the 3D world even at a long range. However, these devices are still expensive, their resolution is limited, *e.g.* the popular Velodyne-64 LiDAR can only measure 64 channels and has low refresh rate.

On the other hand, RGB cameras produce high resolution, color images, but they have to deal with the lighting conditions, problems occur especially at night, moreover, occlusion and shadow can yield problems for image processing. Fortunately, most of the downsides of LiDAR sensors can be compensated by cameras and vice versa, thus the 3D LiDARs and cameras are often used together to detect objects [5, 25, 26], reconstruct scenes [20, 31, 34] or solve navigation tasks [21].

By 3D vision, one can make point clouds denser, however, registering two separate point clouds, obtained by different modalities, is a very challenging task. The extrinsic calibration is necessary for these sensors to work together, which means that their relative position and orientation need to be known a priori. There exist some matching algorithms, but their accuracy is not satisfying.

In case of robots and cars, the devices are fixed and their location and orientation do not change during time. Therefore the rigid body transformation between two instruments can be calculated offline, before the application of the instruments. We show here that the calibration can be done using an ordinary box.

Extrinsic Calibration. Several methods have been proposed to calibrate a camera-LiDAR sensor pair. The early work concentrates on 2D LiDAR devices like the one overviewed in [35]. The list of methods for 3D LiDAR calibration can be divided into three groups. Some of them [9, 22] use planar chessboards, other algorithms [24, 32] apply different form of planar surfaces and there are methods [8, 23] using no calibration objects at all, however, the accuracy of latter ones is far from the desired level. The main problem with planar board based methods is that accurate detection of its edges is hard in the LiDAR point cloud. Especially if low resolution devices are used, *e.g.* the very popular Velodyne-16, which has only 16 vertical laser beams. More problems occur when traditional chessboard patterns are used on planar boards. This pattern heavily pollutes LiDAR point clouds due to its black and white colors [27, 24]. In most of the cases, the camera needs to be pre-calibrated, in other words, its intrinsic parameters are known. The proposed method uses a spatial calibration ob-

2473-9944/17 $31.00 © 2017 IEEE
DOI 10.1109/ICCVW.2017.53

ject and assumes pre-calibrated cameras as well.

In [27], Rodriguez *et al*. use a black circle-based planar board to avoid the large noise caused by chessboard patterns. Their algorithm searches for the 3D coordinates of the circle center and normal vector of the plane. At least 6 positions of the calibration object are needed, the initial guess of the LiDAR-camera rigid transformation is refined by the well-known Levenberg-Marquardt [17, 19] (LM) algorithm.

Alismail *et al*. [2] published an automatic calibration method which uses planar calibration object with a black circular region on it and marked center. The center and normal of the circle is computed from a single view image and a Random Sample Consensus [7] (RANSAC) based method is applied for plane extraction. Finally, point-plane ICP [4] is used with nonlinear optimization by LM to refine the extrinsic parameters.

The method introduced by Park *et al*. [24] uses a white, homogeneous, planar triangle or diamond shaped board for calibration. They need to take more than one image of the board from several positions or use at least three boards at the same time. Another downside of their algorithm is that the spatial coordinates of the planar board are estimated and not measured. This fact influences the accuracy of the calibration. The details of the method can be found in Section 3.

Gong *et al*. published a method in [10] that needs at least two scans of the same trihedron object measured by both instruments for calibration. This produces significantly more data to process. In their work it takes 20 seconds to calibrate using 9 observations. The main disadvantage of their method is that the manufacturing of a trihedron calibrating object is not easy. Moreover, the calibration needs a lot of human intervention, and the separation of the trihedron points and selection of the related planes in the images needs to be done manually.

The method proposed by Velas *et al*. in [32] uses an uncommon calibration object. They assume a white background and a planar object, containing four circular holes inside, their method is based on the work of Levison and Thrun [18]. The holes in both the 3D LiDAR point cloud and the acquired image are detected automatically. However, we did not managed to reproduce this method with our Velodyne HDL-16 LiDAR, because denser point cloud is required for this algorithm.

Geiger *et al*. [9] introduced a method to calibrate a LiDAR-camera pair taking only one measurement by the LiDAR and a single image by the camera. The method is fully automatic, however, it needs multiple chessboards and at least 2 cameras to calibrate. The algorithm will be briefly introduced in Section 3, where it will be compared against the proposed method.

The goal of the paper is twofold: (i) to propose a semi-automatic calibration process and (ii) to achieve accurate camera-to-LiDAR calibration in real-world applications. *The main contribution of this paper is that our algorithm is accurate, and the calibration process requires only a simple cardboard box. Another benefit of our approach is that the extrinsic parameters of arbitrary number of cameras and LiDAR sensors can be calibrated.*

2. Proposed Calibration Method

As it was mentioned in the introduction, the goal of our method is to accurately calibrate the extrinsic parameters of a camera-LiDAR system. Ordinary boxes were selected to archive this goal. The main benefit of these boxes is that they are frequently used in weekdays, and no extraordinary manufacturing or printing is required. Another reason for boxes to be used is the fact that their sides are perpendicular to each other, therefore, the intersections of these planes can be precisely calculated from the point cloud acquired even from a low resolution LiDAR. Other methods which use planar calibration objects are heavily affected by the inaccurate measurement of the plane edges. The proposed method uses only the location of the points and no other information from the LiDAR such as intensity, thus it can be used with any type of LiDAR device.

The calibration process is demonstrated for one calibration object, however, the method can be easily extended for the use of more boxes. This ability is important, because, *e.g*. in autonomous driving, it is a common case to arrange the cameras and LiDARs in a ring structure, thus some of them may not have a joint view and more objects may be needed for the calibration. Since the cameras are assumed to be pre-calibrated – their intrinsic parameters are known a priory –, the placement of the calibration box is almost arbitrary, image distortion has no effect on the accuracy of the calibration. The only requirement is that 3 sides of the calibration box need to be clearly visible in the point cloud and in the image as well.
The inputs for the method are as follows:

1. point cloud(s), acquired by the LiDAR device(s);
2. one image per camera;
3. intrinsic parameters for the cameras; and
4. the measured lengths of the box edges.

2.1. Point Cloud Clustering: Plane Detection

The main idea behind our calibration algorithm is that, if seven corners – along the intersections of three planes – of the calibration box are known, and the projections of these corners are also known in the image, then the problem is reduced to a simple Perspective-n-Point (PnP) problem which can be effectively solved nowadays [12, 16, 30, 36].

Thus the first goal is to accurately calculate these points in the point cloud.

First of all, the rough area of the calibration box needs to be cropped. It is done manually, however, it does not need to be precise. As demonstrated later, the algorithm can automatically separate the sides of the box from other objects falling into this area. Then the algorithm searches for planes in the point cloud. Because of the low number of points located inside the area, a simple sequential RANSAC algorithm [15, 33] is applied. It selects a plane which has the most number of inliers, based on the Euclidean distance, in each iteration. The points related to the most dominant plane is removed from the set, and the robust fitting is repeated until planes can be detected. Then the visible sides of the calibration box are detected. The planes, which are perpendicular to each other, are selected. We apply the following error to be minimized for selecting the nearly perpendicular sides:

$$E(\mathbf{n_1}, \mathbf{n_2}, \mathbf{n_3}) = |\mathbf{n_1}^T \mathbf{n_2}| + |\mathbf{n_1}^T \mathbf{n_3}| + |\mathbf{n_2}^T \mathbf{n_3}|, \quad (1)$$

where $\mathbf{n_k}$ is the normal of the k-th plane ($k \in \{1, 2, 3\}$). Since the number of planes is usually low, the exhaustive search among the plane candidates does not require a lot of computation time. An example for the results of this step can be seen in Fig. 1 left, where the box was placed on a chair and the points located outside a radius were excluded. Sequential RANSAC found 5 planes, then the red, green, and yellow ones were selected as the 3 sides of the box, see the right image of Fig. 1.

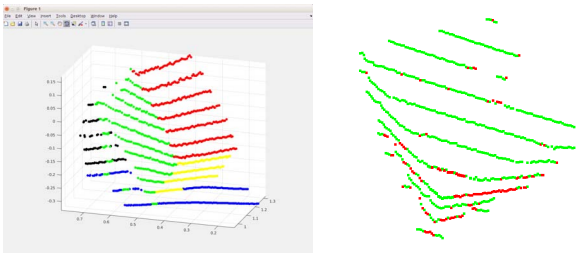

Figure 1. Left: five planes found by the sequential RANSAC. Each plane is differently colored. Right: points clustered based on our box model as inliers (green) and outliers (red).

2.2. Box Fitting and Outlier Removal

The following task of the proposed algorithm is the outlier removal. This is an important step, because LiDAR point clouds can be heavily affected by noise, like range-reflectivity bias caused by texture, as it is discussed in [24]. In this case, the outliers are the points, which belong to the calibration box, but are under heavy noise effect. These points are excluded from the remaining steps. Let us denote the points of the planes, inherited from the previous step, by L_1, L_2 and L_3, the ordering does not matter.

Our algorithm uses RANSAC [7] again to determine the outliers, but this time the model to be fit represents three perpendicular planes. Note that in the previous step those planes were selected which yield the lowest error by Eq. 1 and their orthogonality was not required. The model fitting goes as follows: Three points are selected from L_1 first, these points determine a plane, then two more points are selected from L_2, these two points determine a plane which is perpendicular to the first one and finally, one more point is selected from L_3 determining the third plane which is perpendicular to the first two. The model with the most number of inliers is selected, while the outliers are excluded.

The double outlier filtering may seem to be redundant, but it is not. The aim of the first one is the separation of the calibration box from other objects, while this one determines the noisy box points.

2.3. Iterative Box Refinement

After the outliers are dropped from the point set, an iterative algorithm consisting of two steps refines the planes of the box to the points. This refinement is responsible to accurately fit three perpendicular planes to the point sets. The first step rotates the box model, while the second one translates that.

2.3.1 Rotation Step

In this step, two planes are selected and rotated along their intersection line – the edge of the box. The rotation minimizes the sum of the square errors between the points and the planes.

Let $\mathbf{p_j^1}, j \in \{1, 2, ...m_1\}$, $\mathbf{p_j^2}, j \in \{1, 2, ...m_2\}$, and $\mathbf{p_j^3}, j \in \{1, 2, ...m_3\}$ denote the points in $L1$, $L2$ and $L3$ respectively. Let $\mathbf{q^i}$ denote a point lying on the i-th plane, which does not need to be in the set $\mathbf{p_j^i}$. Let $\mathbf{n}^i$ be the normals of the planes.

The fitting problem is equivalent to minimize the sum of squared distances of the points with respect to the corresponding plane. The least squares cost function is the sum of all point-plane distances. It is defined as follows:

$$C = \sum_{i=1}^{3} \sum_{j=1}^{m_i} \left| \left(\mathbf{p_j^i} - \mathbf{q^i}\right)^T \mathbf{n^i} \right|^2. \quad (2)$$

Without loss of generality, the coordinate system of the world is fixed to the intersection of box planes, the principal directions of the world equal to the edges of the cuboid. Let us consider the rotation around the third axis by angle γ. Then the rotation matrix is as follows:

$$\mathbf{R}_Z^T = \begin{bmatrix} c & -s & 0 \\ s & c & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (3)$$

where $c = \cos\gamma$ and $s = \sin\gamma$. The normal of the three planes are $[1 \quad 0 \quad 0]^T$, $[0 \quad 1 \quad 0]^T$, and $[0 \quad 0 \quad 1]^T$. The rotation does not influence the fitting error of the points of the third plane, therefore the minimization problem becomes

$$C'' = \sum_{i=1,2} \sum_{j=1}^{m_i} \left| \left(\begin{bmatrix} c & -s & 0 \\ s & c & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p_j^i} - \mathbf{q^i} \right)^T \mathbf{n^i} \right|^2 . \quad (4)$$

After elementary modifications, the cost function is transformed to the minimization of the norm $|\mathbf{Ax}|$ subject to $\mathbf{x}^T\mathbf{x} = 1$, where

$$\mathbf{A} = \begin{bmatrix} x_1^1 & -y_1^1 \\ \vdots & \vdots \\ x_{m_1}^1 & -y_{m_1}^1 \\ y_1^2 & x_1^2 \\ \vdots & \vdots \\ y_{m_2}^2 & x_{m_2}^2 \end{bmatrix}, \mathbf{x} = \begin{bmatrix} c \\ s \end{bmatrix}. \quad (5)$$

The constants m_1 and m_2 denote the point numbers belonging to the first and second plane, respectively. The optimal solution for $\mathbf{x}$ is retrieved from the eigenvector of matrix $\mathbf{A}^T\mathbf{A}$ corresponding to the smaller eigenvalue[1]. The angle γ is calculated as $\gamma = atan2(s, c)$.

The rotations around axes $\mathbf{X}$ and $\mathbf{Z}$ are similarly obtained.

2.3.2 Translation Step

The translation of the box is run along the three normals of the planes one by one. The advantage of the translation is that only the point fitting error of the selected plane is being changed if the translation of the box is parallel to its normal.

The cost function defined in Eq. 2 is modified as follows:

$$C''' = \sum_{j=1}^{m_i} \left| \left(\mathbf{p_j^i} - \mathbf{q^i} - \alpha\mathbf{n} \right)^T \mathbf{n^i} \right|^2 , \quad (6)$$

where α is the length of the translation. The translations have to be calculated for the three perpendicular directions one by one. However, the problem can be solved as well by a single step. For this case, let us consider the translation problem as minimization of the following cost function:

$$C'''' = \sum_{i=1}^{3} \sum_{j=1}^{m_i} \left| \left(\mathbf{p_j^i} - \mathbf{q^i} - \mathbf{t} \right)^T \mathbf{n^i} \right|^2 , \quad (7)$$

where $\mathbf{t}$ is the optimal translation vector.

[1] Matrix $\mathbf{A}^T\mathbf{A}$ has always two non-negative real eigenvalues.

This problem can be written as a homogeneous linear system of equations as $\mathbf{B}t = \mathbf{c}$ as follows:

$$\mathbf{B} = \begin{bmatrix} \mathbf{n^{1T}} \\ \vdots \\ \mathbf{n^{1T}} \\ \mathbf{n^{2T}} \\ \vdots \\ \mathbf{n^{2T}} \\ \mathbf{n^{3T}} \\ \vdots \\ \mathbf{n^{3T}} \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} \mathbf{n^{1T}} \left(\mathbf{p_1^1} - \mathbf{q^1} \right) \\ \vdots \\ \mathbf{n^{1T}} \left(\mathbf{p_{m_1}^1} - \mathbf{q^1} \right) \\ \mathbf{n^{2T}} \left(\mathbf{p_1^2} - \mathbf{q^2} \right) \\ \vdots \\ \mathbf{n^{2T}} \left(\mathbf{p_{m_2}^2} - \mathbf{q^2} \right) \\ \mathbf{n^{3T}} \left(\mathbf{p_1^3} - \mathbf{q^3} \right) \\ \vdots \\ \mathbf{n^{3T}} \left(\mathbf{p_{m_3}^3} - \mathbf{q^3} \right) \end{bmatrix}, \quad (8)$$

where row vector $\mathbf{n^{jT}}$ denotes the transpose of column vector $\mathbf{n^j}$. Each term of the cost function gives an equation to the system. The solution is given by the well known formula $\mathbf{t} = \left(\mathbf{B^T B}\right)^{-1} \mathbf{B^T c}$. As the vectors $\mathbf{n^1}$, $\mathbf{n^2}$, and $\mathbf{n^3}$ are perpendicular to each other, matrix $\mathbf{B}$ is always non-singular, thus the matrix inversion can be calculated.

2.4. Convergence

The rotation and translation steps explained above are repeated until convergence. We experienced that less than 30 iterations are enough. The convergence of the proposed method is tested so that the origin of the coordinate system, where the planes were aligned with the axis, gave the initial translation/rotation values and it was found that the iteration always converged to an acceptable result.

2.5. Calculation of Extrinsic Parameters

After the iterative refinement of the box sides is done, the corner points of the calibration box can be calculated, because the sizes of the box are known. The colored points in Fig. 3 indicate the corners of the box, the whole neighborhood of the boxes was the input for the algorithm, after it was cut from the original point clouds.

If the extrinsic parameters of a LiDAR-LiDAR pair need to be known, the algorithm can find the corner points in the LiDAR point clouds separately. Then the transformation between the two point clouds can be calculated from the point correspondences of box corners by point registration [3, 13, 14].

The calibration method can be also used to calibrate LiDAR-Camera systems. In this case the projections of the calibration box corners need to be selected in the processed image. Then the selected points are refined by Harris [11] corner detector. The problem of finding the extrinsic parameters is then equal to a PnP problem as 3D-2D point-point correspondences are known. We apply the Effective-PnP (EPnP) algorithm [16] in our approach. Since the cameras are already calibrated, the intrinsic parameters are taken into

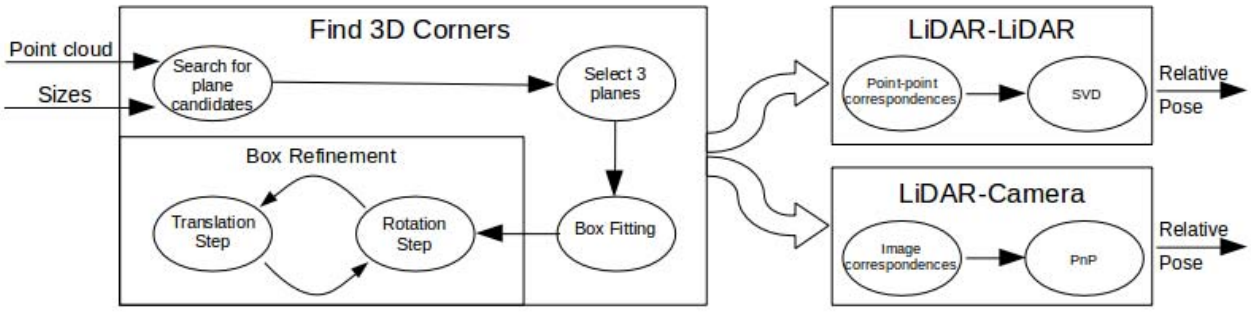

Figure 2. The block diagram for the calibration process. The box corners have to be detected in the spatial point cloud. The LiDAR-LIDAR calibration is based on point registration, while the essence of LiDAR-camera calibration is the application of a PnP algorithm.

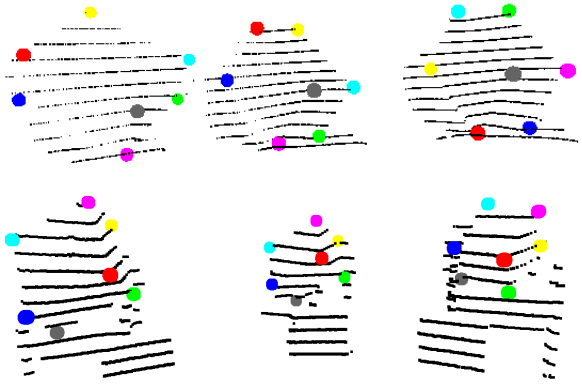

Figure 3. The algorithm finds corners even within the sparse point cloud generated by VELODYNE-16.

consideration during the PnP, radial and tangential distortion have no effect on the accuracy of the calibration.

3. Tests

The method is validated on both synthetic and real-world test data. For the synthetic tests, Blender Sensor Simulator (Blensor) [1] is used which is an open source simulation package for the widely-used 3D modeling and rendering software[2]. The real-world results consist of colored point clouds and point cloud fusions coming from both high and low resolution LiDAR devices.

3.1. Synthetic Tests

Blender is a free and open source software for 3D simulation, rendering, animation and modeling. It supports the use of multiple cameras with a variety of parameters to set, even photo-realistic images can be saved. Chessboards and a calibration box is modeled inside the synthetic environment of Blender for simulating realistic test scenarios. Blensor [1] is a sensor simulation package, which extends the capability of Blender with different kind of LiDAR and time-of-light devices. Velodyne-64 is used for the synthetic tests with the default settings, however, it is possible to set the scanning range, level of noise, rotation speed. In our test, only the level of noise is varied. Gaussian noise is added to the ground truth (GT) distance from the sensor with zero mean and varying variance between 0 and 0.14. This means that the noisy points are located on the rays casted by the range sensor.

An example of the virtual scene is visualized in Fig. 5, which is created by Blensor [1] simulation package. It was kept in mind during the construction of the scene that three rival algorithms have to use the same camera-LiDAR system setup, and every single one uses different object(s) for the calibration.

The rival methods are as follows:

- The first algorithm introduced by Geiger *et al*. in [9] uses chessboards,
- the second algorithm by Park *et al*. [24] uses polygonal boards,
- while the proposed algorithm applies a virtual box in order to determine the relative pose of the equipments to be calibrated.

The algorithm by Geiger *et al*., labeled as *'kitti'* in charts, needs at least two camera images of many chessboards to calibrate. The fully automatic method reconstructs the scene of the chessboards processing the camera images, and then tries to merge the point cloud with the one acquired by the LiDAR sensor. The last step can be problematic and highly depends on the environment of the calibration. We use the online demo[3] at the website of the authors to get the calibration data.

[2] www.blender.org

[3] www.cvlibs.net/datasets/kitti

Figure 4. Re-projected points of 3 out of 6 clouds visualized in Fig. 3. Parameters of camera pose estimated by method of Geiger *et al*. (left) and proposed one (right).

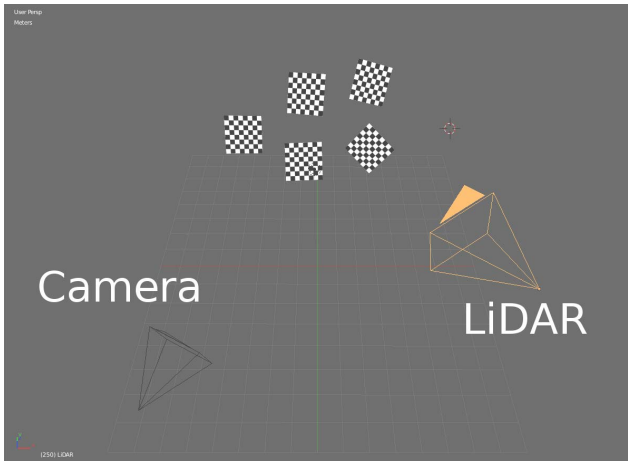

Figure 5. An example of the scenery created by Blensor [1]. The left and right pyramids show the location of the camera and the LiDAR, respectively.

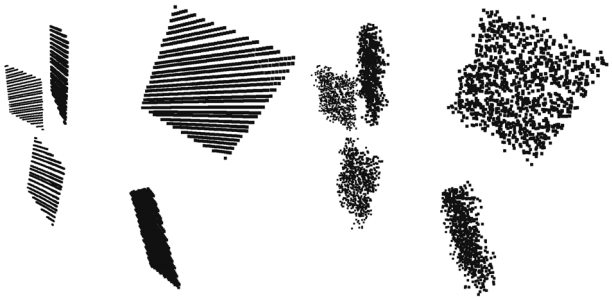

Figure 6. Point clouds of the scene taken by Blensor. Scene model visualized in Fig. 5. Left: without noise. Right: Gaussian noise, $\sigma = 0.14$.

The algorithm, labeled as *'polygonal'*, is introduced by Park *et al*. in [24]. In our tests the method is used with five diamond-shaped planar boards, as the authors suggested. The points of these boards need to be selected one by one in the LiDAR point cloud manually. Then the planes of the boards are estimated by RANSAC and *virtual points* are calculated using the LiDAR scan lines. The *virtual points* are applied to estimate the edges of each board and the intersections of these edges result the corners of the board. The projections of the intersections are selected from FAST [28, 29] feature points. Finally, singular value decomposition (SVD) and the Levenberg-Marquardt algorithm [17, 19] are used to get the extrinsic parameters.

In our real life experiments, it is found that the noise level in the LiDAR spatial point cloud depends not only on the texture and material of the object, but on the type of the LiDAR sensor as well. Even LiDARs from the same manufacturer can be different – the sparse point cloud of the Velodyne-16 is more accurate than the denser point cloud coming from its big model, the Velodyne-64.

The algorithms are tested w.r.t. Gaussian noise effecting the LiDAR point cloud. The standard deviation (σ) is varied between 0 to 0.14. In the synthetic tests, the LiDAR sensor is placed at $(4.72945, -5.24017, 8.76321)$, while the main camera at $(-5.5727, -6.98041, 3.55163)$, the distances are measured in meters. Two more cameras and five chessboards are used for the method of Geiger *et al*., five square board for the method by Park *et al*. [24], and only one calibration box was placed for the proposed algorithm. The size of the chessboards and planar boards is 2x2 meters, that of the calibration box is 1x2x3 meters.

For the evaluation of the methods, we compare the extrinsic parameters (rotation and translation) and the ground

truth (GT) data. The translation error in the top plot of Fig. 7 is the Euclidean distance between the GT translation vector and the estimated vector obtained by the algorithms.

The calculation of the rotation error is not trivial as a rotation can be represented in several ways: using three angles, an orthonormal matrix, an axis with an angle, etc. We have compared several error metrics, the characteristics of those for our test cases were the same. Due to its simplicity, we have calculated the rotation angle of matrix $\mathbf{R}_{GT}^{T}\mathbf{R}$ representing the error by the well-known formula [6] as follows:

$$\alpha = cos^{-1}\left(trace\left(\mathbf{R}_{GT}^{T}\mathbf{R}\right) - 1\right)/2), \qquad (9)$$

where $\mathbf{R}_{GT}$ is the GT rotation matrix, retrieved from Blensor data, and $\mathbf{R}$ is the rotation matrix obtained by the tested algorithms.

Fig. 7 shows the translation and rotation errors of the algorithms. It seems to be clear that the method of Park *et al.*, labeled as *'polygonal'* in the figures, and ours, labeled as *'proposed'*, can calculate the rotation matrices very accurately, their error do not exceed 1.5 degrees even in the presence of high standard deviation Gaussian noise. This is a good result, because even low error in the rotation matrices can heavily effect the result of the colored point cloud, especially if the object are located at a long distance. The translation error of the algorithms is varying between 5 centimeters to 0.5 meters, see the top plot of Fig. 7. It can be seen that the proposed method is significantly more accurate than the others, The differences are between 5 to 10 centimeters. Both rotation and translation errors are approximately linear w.r.t. Gaussian noise.

3.2. Real World Tests

The testing scene consists of 3 calibration boxes and 4 chessboards. For the sake of comparison, we use the reprojection of certain points of the LiDAR point cloud to the camera image. The test is completed as follows: Three cardboard boxes are put on the top of chairs first to be visible for the Velodyne-16 LiDAR. Five chessboards are installed in order to run the method of Geiger *et al.* [9]. An image is taken at the original position of the camera, and another one at a different location, because the method needs at least two images for the calibration.

In Fig. 4 the re-projected points of the partial point clouds are visualized, the processed scene is seen in Fig. 3. On the left image, the extrinsic parameters resulted by Geiger *et al.*, on the right the extrinsic parameters of the proposed algorithm are utilized for the reprojection. Remark that the projected locations of the chessboard corners, the top of the calibration boxes, and the points of the leg of the green chair are incorrectly drifted. The drift represents the inaccuracy of the method. On the right, these reprojected points align the corresponding image regions,

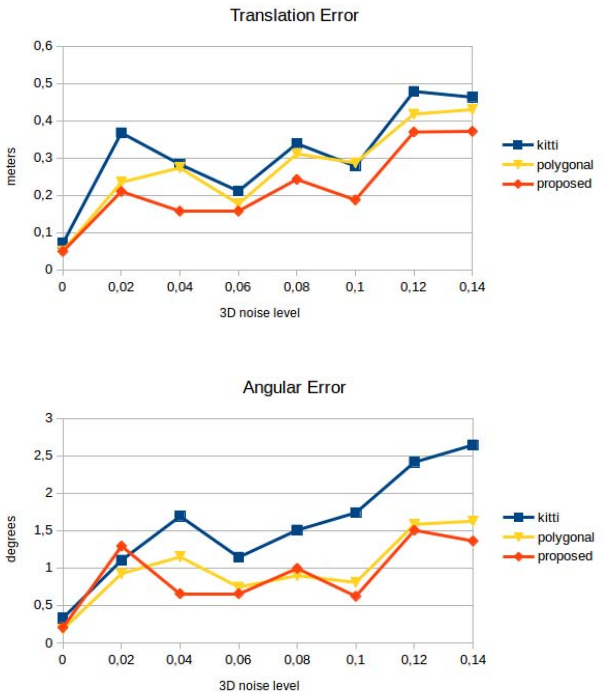

Figure 7. Translation (top) and rotation (bottom) error against Gaussian noise.

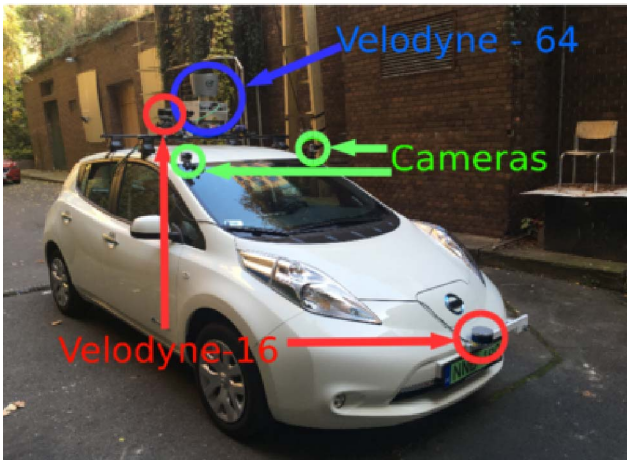

Figure 8. A Velodyne-64, 2 Velodyne-16 and 2 cameras are mounted on our autonomous car.

suggesting that the quality of the proposed method is significantly higher.

The method is tested on real life data obtained by the visual system of our autonomous car. One high resolution and two low resolution LiDAR sensors, and two cameras are mounted on the car, see Fig. 8. The calibration is done in a backyard. Fig. 9 shows the result of the calibration which is a point cloud fusion of the high and two low resolution LiDAR devices, and colored using intensity values

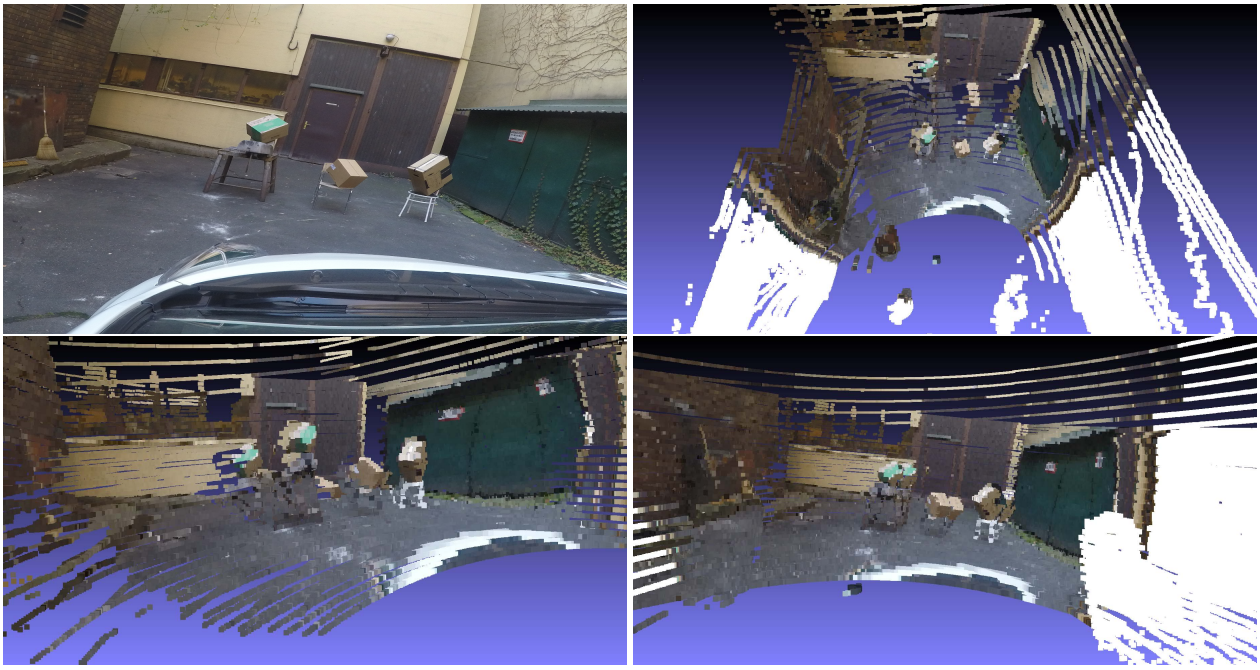

Figure 9. Point cloud fusion of 3 LiDARs and colored by the left camera of the autonomous car. The Top-left and right images show the camera image and the point cloud from a bird-view. The images at the bottom are taken from the viewpoint of the camera and from the Velodyne-64 LiDAR, respectively. The occlusion caused by different LiDAR views was not considered during the point cloud coloring process, white points are located outside of the camera view.

of the RGB cameras: locations in image space are obtained by projecting the spatial points to the cameras using the obtained extrinsic camera parameters by the proposed algorithm.

4. Conclusion

In this paper, an offline algorithm for extrinsic calibration of LiDAR-camera system was introduced. The proposed method requires only an ordinary calibration box with known sizes and a pre-calibrated camera. The method is capable to find the calibration object robustly, even in a low resolution LiDAR point cloud. The core of the algorithm is to locate accurately the box corners in the point cloud. This is carried out by robust search across plane candidates, then outlier extraction and a two-step iteration method are executed which refines the box to the point cloud. The algorithm can be used to calibrate camera-LiDAR and LiDAR-LiDAR pairs as well. The proposed method is compared with state-of-the-art algorithms in a synthetic test environment. Another test was carried out: real-world data obtained by complex visual system of our autonomous car was applied to demonstrate the accuracy of the proposed method.

References

[1] Blender sensor simulation. http://www.blensor.org. 5, 6

[2] H. S. Alismail, L. D. Baker, and B. Browning. Automatic calibration of a range sensor and camera system. In *2012 Second Joint 3DIM/3DPVT Conference: 3D Imaging, Modeling, Processing, Visualization & Transmission (3DIMPVT 2012)*, Pittsburgh, PA, October 2012. IEEE Computer Society. 2

[3] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-D point sets. *PAMI*, 9(5):698–700, 1987. 4

[4] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image Vision Comput.*, 10(3):145–155, Apr. 1992. 2

[5] H. Cho, Y.-W. Seo, B. V. Kumar, and R. R. Rajkumar. A multi-sensor fusion system for moving object detection and tracking in urban driving environments. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1836–1843. IEEE, 2014. 1

[6] David Eberly. Rotation Representations and Performance Issues. January 2002, Magic Software, Inc. (Online; accessed 21 June 2017). 7

[7] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2, 3

[8] R. Frohlich, Z. Kato, A. Trémeau, L. Tamas, S. Shabo, and Y. Waksman. Region based fusion of 3d and 2d visual data

for cultural heritage objects. In *23rd International Conference on Pattern Recognition, ICPR 2016, Cancún, Mexico, December 4-8, 2016*, pages 2404–2409, 2016. 1

[9] A. Geiger, F. Moosmann, O. Car, and B. Schuster. Automatic camera and range sensor calibration using a single shot. In *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*, pages 3936–3943, 2012. 1, 2, 5, 7

[10] X. Gong, Y. Lin, and J. Liu. 3d lidar-camera extrinsic calibration using an arbitrary trihedron. *Sensors*, 13(2):1902, 2013. 2

[11] C. Harris and M. Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988. 4

[12] J. A. Hesch and S. I. Roumeliotis. A direct least-squares (dls) method for pnp. In *International Conference on Computer Vision*, pages 383–390. IEEE, 2011. 2

[13] B. Horn. Closed-form Solution of Absolute Orientation using Unit Quaternions. *Journal of the Optical Society of America*, 4:629–642, 1987. 4

[14] B. Horn, H. Hilden, and S. Negahdaripourt. Closed-form Solution of Absolute Orientation Using Orthonormal Matrices. *Journal of the Optical Society of America*, 5(7):1127–1135, 1988. 4

[15] Y. Kanazawa and H. Kawakami. Detection of planar regions with uncalibrated stereo using distributions of feature points. In *BMVC*, 2004. 3

[16] V. Lepetit, F.Moreno-Noguer, and P.Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal Computer Vision*, 81(2), 2009. 2, 4

[17] K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944. 2, 6

[18] J. Levinson and S. Thrun. Automatic online calibration of cameras and lasers. In *Robotics: Science and Systems*, 2013. 2

[19] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math.*, 11:431–441, 1963. 2, 6

[20] F. M. Mirzaei, D. G. Kottas, and S. I. Roumeliotis. 3d lidar–camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization. *The International Journal of Robotics Research*, 31(4):452–467, 2012. 1

[21] P. Moghadam, W. S. Wijesoma, and D. J. Feng. Improving path planning and mapping based on stereo vision and lidar. In *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on*, pages 384–389. IEEE, 2008. 1

[22] G. Pandey, J. McBride, S. Savarese, and R. Eustice. Extrinsic calibration of a 3d laser scanner and an omnidirectional camera. In *7th IFAC Symposium on Intelligent Autonomous Vehicles*, volume 7, Leece, Italy, September 2010. 1

[23] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice. Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 2053–2059, Toronto, Canada, July 2012. 1

[24] Y. Park, S. Yun, C. S. Won, K. Cho, K. Um, and S. Sim. Calibration between color camera and 3d lidar instruments with a polygonal planar board. *Sensors*, 14(3):5333–5353, 2014. 1, 2, 3, 5, 6

[25] C. Premebida, O. Ludwig, and U. Nunes. Lidar and vision-based pedestrian detection system. *Journal of Field Robotics*, 26(9):696–711, 2009. 1

[26] C. Premebida, G. Monteiro, U. Nunes, and P. Peixoto. A lidar and vision-based approach for pedestrian and vehicle detection and tracking. In *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pages 1044–1049. IEEE, 2007. 1

[27] S. Rodriguez F, V. Fremont, and P. Bonnifait. Extrinsic calibration between a multi-layer lidar and a camera. 1, 2

[28] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *Proceedings of the 9th European Conference on Computer Vision - Volume Part I*, ECCV'06, pages 430–443, Berlin, Heidelberg, 2006. Springer-Verlag. 6

[29] E. Rosten, R. Porter, and T. Drummond. Faster and better: A machine learning approach to corner detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(1):105–119, Jan. 2010. 6

[30] G. Schweighofer and A. Pinz. Globally optimal o(n) solution to the pnp problem for general camera models. In *BMVC*, 2008. 2

[31] L. Smadja, J. Ninot, and T. Gavrilovic. Road extraction and environment interpretation from lidar sensors. *IAPRS*, 38:281–286, 2010. 1

[32] M. Veĺas, M. Španěl, Z. Materna, and A. Herout. Calibration of rgb camera with velodyne lidar. In *WSCG 2014 Communication Papers Proceedings*, volume 2014, pages 135–144. Union Agency, 2014. 1, 2

[33] E. Vincent and R. Laganiére. Detecting planar homographies in an image pair. In *ISISPA*, 2001. 3

[34] J. Zhang and S. Singh. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 2174–2181. IEEE, 2015. 1

[35] Q. 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, Sendai, Japan, September 28 - October 2, 2004*, pages 2301–2306, 2004. 1

[36] Y. Zheng, Y. Kuang, S. Sugimoto, K. Åström, and M. Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *ICCV*, pages 2344–2351, 2013. 2