remote sensing

MDPI

Article

Reflectance Intensity Assisted Automatic and Accurate Extrinsic Calibration of 3D LiDAR and Panoramic Camera Using a Printed Chessboard

Weimin Wang [1,*], Ken Sakurada [1] and Nobuo Kawaguchi [1,2]

[1] Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan; sakurada@nagoya-u.jp (K.S.); kawaguti@nagoya-u.jp (N.K.)

[2] Institute of Innovation for Future Society, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan

* Correspondence: weimin@ucl.nuee.nagoya-u.ac.jp; Tel.: +81-452-789-4544

Received: 27 April 2017; Accepted: 13 August 2017; Published: 16 August 2017

Abstract: This paper presents a novel method for fully automatic and convenient extrinsic calibration of a 3D LiDAR and a panoramic camera with a normally printed chessboard. The proposed method is based on the 3D corner estimation of the chessboard from the sparse point cloud generated by one frame scan of the LiDAR. To estimate the corners, we formulate a full-scale model of the chessboard and fit it to the segmented 3D points of the chessboard. The model is fitted by optimizing the cost function under constraints of correlation between the reflectance intensity of laser and the color of the chessboard's patterns. Powell's method is introduced for resolving the discontinuity problem in optimization. The corners of the fitted model are considered as the 3D corners of the chessboard. Once the corners of the chessboard in the 3D point cloud are estimated, the extrinsic calibration of the two sensors is converted to a 3D-2D matching problem. The corresponding 3D-2D points are used to calculate the absolute pose of the two sensors with Unified Perspective-n-Point (UPnP). Further, the calculated parameters are regarded as initial values and are refined using the Levenberg-Marquardt method. The performance of the proposed corner detection method from the 3D point cloud is evaluated using simulations. The results of experiments, conducted on a Velodyne HDL-32e LiDAR and a Ladybug3 camera under the proposed re-projection error metric, qualitatively and quantitatively demonstrate the accuracy and stability of the final extrinsic calibration parameters.

Keywords: 3D LiDAR; panoramic camera; reflectance intensity; automatic calibration

1. Introduction

A combination of the Light Detection And Ranging (LiDAR) sensor and the panoramic camera has been widely utilized for deriving the benefits of color as well as depth information. Two typical examples in which the combination is used are 3D mapping and model generation, which use the LiDAR sensor and the color and the texture information of images [1,2], and the improvement of the pedestrian detection accuracy in images using the distance information obtained from the LiDAR [3–5]. The first and critical step for fusing multi-modal data from the two devices is the accurate and convenient extrinsic calibration.

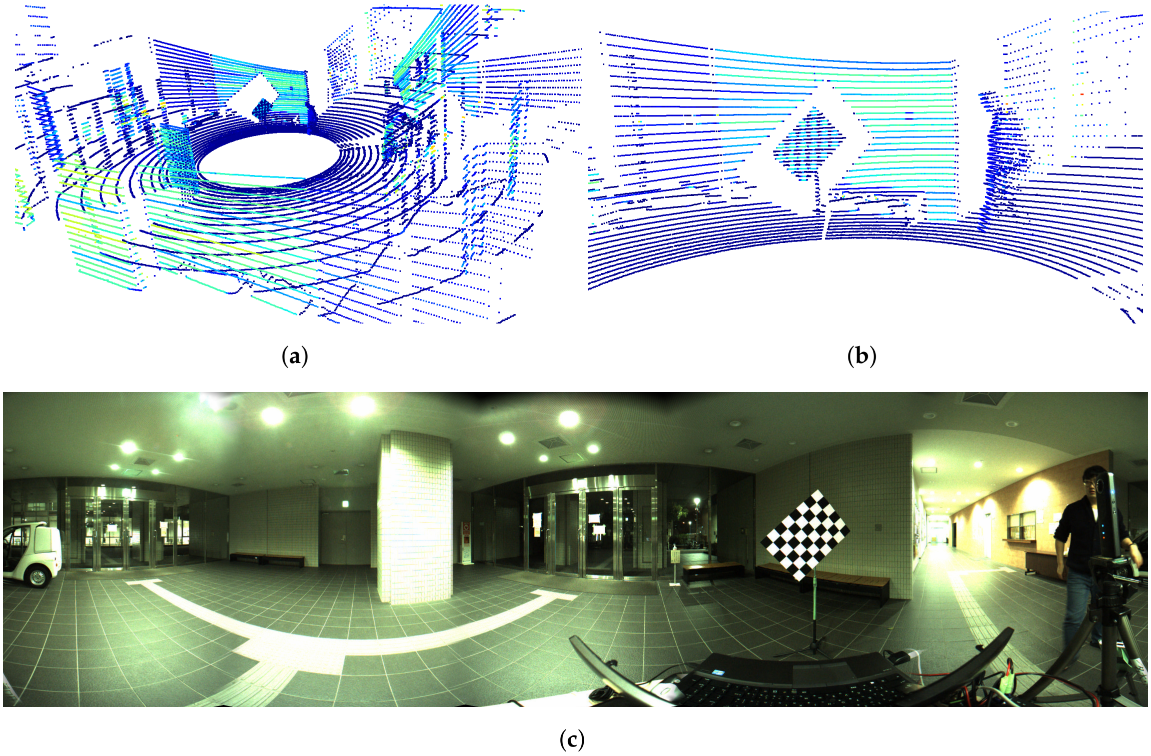
The process of the extrinsic calibration between the LiDAR and the camera involves the calculation of a proper transformation matrix to align the coordinate systems of the two sensors. This process has been studied for many years in the fields of both robotics and computer vision. Extrinsic calibration methods can be classified into target-based methods and non-target methods. The focus of the target-based methods is to find corresponding features of the common target from multi-modal data. Non-target methods estimate the transformation matrix by maximizing the correlation of mutual information in multi-modal data, such as edges in images and discontinuity of the scanline in the point cloud [6,7], as well as luminance of images and reflectance of the point cloud [8]. In this work, we focus on the target-based extrinsic calibration.

For target-based calibration, the conventional method involves finding the vertices of a polygonal board, which can be a chessboard or a triangular board, both in the point cloud obtained by the LiDAR and the image captured by the camera either manually or automatically [9–11]. The vertices are estimated by constructing the convex hull of the extracted board's point cloud. However, the vertices only include the geometric features of the target board's outer contour, which are formed by the discontinuity of the scanlines, while the information inside the counter is not used. To make full use of the information acquired by the LiDAR, the "inner texture" of the point cloud, i.e., the reflectance intensity, which is also derived from the LiDAR sensor, is used. Unlike the approach in [8], we utilize the reflectance intensity to estimate the corners of the chessboard from the 3D point cloud. If the corners of the 3D point cloud are identified, the extrinsic calibration is converted to a 3D-2D matching problem.

The idea of the estimation of corners from the point cloud is based on the fact that the intensity of white patterns differs from that of black patterns, as seen in Figure 1. Figure 1a shows a point cloud; colors are representatives of the intensity. Figure 1b shows the zoomed-in part of the chessboard shown in Figure 1c, in which the change in reflectance intensity between the different patterns can be observed. However, the noise and sparsity of the point cloud present a challenge. To address this challenge, we propose a novel method that fits a chessboard model to the chessboard's point cloud, and we also introduce Powell's method [12] for optimizing a defined cost function to obtain the fitting solution. After the corners of the chessboard in the 3D point cloud are estimated, the initial value of the transformation matrix is calculated using the Unified Perspective-n-Point (UPnP) method [13]. The initial value is then applied to refine the result by using the nonlinear least squares optimization with the Levenberg-Marquardt (LM) method [14,15]. We evaluate the proposed method by using data obtained from a Velodyne HDL-32e LiDAR sensor and a FLIR Ladybug3 panoramic camera under the proposed reflectance intensity-based re-projection error metrics.

Three main contributions of this work are as follows. First, we propose a method that utilizes the reflectance intensity of the point cloud to estimate the corners of the chessboard's point cloud. Further, we introduce Powell's method to optimize the corner detection problem. To the best of our knowledge, this is the first published work that utilizes the intensity information to estimate the corners from the point cloud. Second, the quantitative error evaluation of the proposed method is conducted using simulations. In addition, we analyze the relationship between the estimation error and other conditions such the measurement error and the distance of the chessboard. Finally, we provide the Python implementation of the proposed method, which can be downloaded from https://github.com/mfxox/ILCC.

The rest of this paper is structured as follows. Related works are revisited in Section 2. The overview of the proposed method and notations used in this work are described in Section 3. In Section 4, detailed procedures from automatic detection of the chessboard to the corner estimation and optimization are explained. The corner detection on the panoramic image and the correspondence grouping of the detected corners in modal data for the two devices is described in Section 5. In addition, the process of the final extrinsic calibration after the correspondence grouping of the corners is described in this section. Simulation results for corners estimation in the point cloud and experimental results are described in Section 6. Discussions about this work and some tips for better extrinsic calibration with the proposed method are made in Section 7. Finally, we present our conclusions and scope for future work in Section 8.


(**a**) (**b**)

(**c**)

Figure 1. Data from an identical scene captured by the LiDAR sensor and the panoramic camera. (**a**) The points are colored by the reflectance intensity (blue indicates low intensity, red indicates high intensity); (**b**) The zoomed chessboard. We can see the changes in reflection intensity of the point cloud between the white and black patterns; (**c**) The panoramic image of the same scene.

2. Related Works

Extrinsic calibration of two sensors in the fields of computer vision as well as robotics is usually achieved by solving an optimization problem under certain constraints. These constraints can be geometric relationships, the correspondence of common features, or the correlation of mutual information between two sensor modalities. Related works we revisited during the course of this study are classified on the basis of these types of constraints.

2.1. Multiple Views on a Planar Checkerboard

Zhang (2004) first published the work about the calibration for a system comprising a 2D LiDAR and a monocular camera [16]. He extrinsically calibrated the two sensors under the constraint that in the camera coordinate system, the scalar projection of a vector between the origin and a point on the plane onto the normal of the plane equals the distance between the origin and the plane. A system for 3D LiDAR sensors with constraints similar to those in Zhang's method was proposed and implemented as a MATLAB toolkit in [17]. This system also made several assumptions, key among those being that the point cloud is dense enough, which is challenging when using single frame data obtained by mobile LiDAR sensors such as the Velodyne HDL-32e. Pandey et al. extended the method to a system of Velodyne HDL-64E LiDAR and Ladybug3 panoramic camera, which is similar to the set we used for this work [18]. This method was applied to the Ford Campus Dataset [19]. Mirzaei et al. (2012) subsequently proposed a method for both intrinsic and extrinsic calibration in [9].

2.2. Multiple Geometry Elements

The extrinsic calibration can also be performed if common geometry features like points, line segments, or planes are extracted from the data obtained by the two sensors and the correspondence of features from the two modalities are built. Scaramuzza et al. proposed a method based on the point correspondences [20]. Once the point correspondences are known, the transformation matrix can be calculated using the methods for PnP (Perspective n Points) problems. However, it is difficult to manually identify the corresponding 3D point features accurately in point cloud, especially for a sparse point cloud. To overcome this drawback, Moghadam et al. proposed a method based on automatically extracted 2D and 3D lines [21]. However, this approach requires the user to manually determine the correspondence of lines. Gong et al. proposed a method under the plane-to-plane constraints using a trihedral object, which also requires human intervention for plane selection [22]. There are also some methods that automatically extract feature points, such as vertices of a polygonal planar checkerboard, from the LiDAR data. Nevertheless, these approaches require either manual operation for feature points selection from the image [10] or customized checkerboard for feature points generation in the point cloud [11]. Geiger et al. proposed an automatic method for extrinsic calibration with one shot of multiple checkerboards in [23]. This approach recovers the 3D structure from the detected corners in images. Then the calibration is performed under the constraints that planes of the chessboards recovered from the images should coincide with the planes detected from the LiDAR data. This method was applied for extrinsic calibration between the cameras and the LiDAR sensor in KITTI Dataset [24]. However, to recover the 3D structure from corners of different chessboards, the cameras require stereo configuration for sufficient common field of view of the chessboards. This is challenging for panoramic camera like Ladybug3 we use in this work. Geiger's method for corner detection in images showed robustness from the experimental results, and hence we also apply it for corners detection from the panoramic images in this work.

2.3. Correlation of Mutual Information

The abovementioned methods require either artificial observation objects (chessboards or triangular boards) or user intervention (features or correspondence selection). This is inconvenient for applications that need frequent extrinsic calibration like autonomous driving cars, in which case the relative pose may drift due to the vibration. To solve this problem, several online calibration methods are proposed. These methods are generally based on the maximization of the mutual information (MI), such as edges in images and discontinuity of the scanline in the point cloud or luminance of images and reflectance intensity of the point cloud [6–8]. However, online calibration methods remain difficult to apply with poor initializations. This means that the pre-knowledge of a roughly accurate initial guess, which maybe estimate using the off-line calibration method or manual measurement, is necessary.

2.4. Our Approach

In our proposed method for extrinsic calibration, we also applied constraints by obtaining corresponding feature points on a printed chessboard. This method belongs to the constraints on multiple geometry elements. However, unlike the approaches mentioned above, we use the corners instead of the vertices of the chessboard's point cloud. To estimate the corners of the point cloud, we formulate a cost function under the constraint that the 3D points with high reflectance intensity must lie on the white patterns of the chessboard and vice versa. Once the corners are detected in both multi-modal data, the correspondences are further made in a predefined order. Initial parameters can be obtained using the UPnP method [13] on the corresponding corners and further refined using the Levenberg-Marquardt method [14,15]. Our method is fully automatic and thus dose not require user intervention in the whole process from the detection of the chessboard, corner detection, the correspondence of the corners, to the final optimization.

A combination of a Velodyne HDL-32e LiDAR sensor and a Ladybug3 panoramic camera is used for experiments in this work. There are several studies on the intrinsic calibration of the 3D LiDAR sensor [9,25] and the panoramic camera [26,27]. In the remainder of this paper, we assume that before the process of the extrinsic calibration, both the LiDAR and the panoramic cameras have been intrinsically calibrated from the factory setting.

3. Overview and Notations

3.1. Overview

The overview of the proposed method is illustrated in Figure 2. First, the point cloud obtained from the LiDAR is segmented into multiple parts. The point cloud of the chessboard is identified from within the segments based on the characters of the segment. The corners of the chessboard in the point cloud are estimated by minimizing a defined cost function. On the other hand, corners of the chessboard in the image are detected using an existing method. Correspondence of the corners is built based on the predefined counting order. The corresponding pairs are then used to estimate an initial value of the transformation matrix by solving an absolute pose problem. Finally, the value is refined by optimizing a proposed nonlinear cost function.

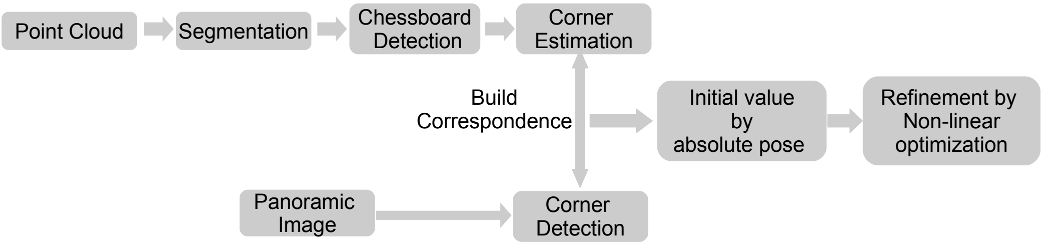


Figure 2. Overview of the proposed method.

3.2. Notations

For the convenience of explanation, the following notations are used in this paper.

- $\boldsymbol{p}_i = (x_i, y_i, z_i)^T$: coordinates of a 3D point.
- $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \dots, \boldsymbol{p}_n\}$: set of n 3D points.
- $\boldsymbol{\theta} = (\theta_x, \theta_y, \theta_z)^T$: rotation angle vector whose element corresponds to the rotation angle along x-, y-, z-axis respectively.
- $\boldsymbol{t} = (t_x, t_y, t_z)^T$: the translation vector.
- $\mathbf{R}(\boldsymbol{\theta}) = \mathbf{R}_z(\theta_z)\mathbf{R}_y(\theta_y)\mathbf{R}_x(\theta_x)$: rotation matrix.
- $T_r(\boldsymbol{\theta}, \boldsymbol{t}, \boldsymbol{p_i}) = \mathbf{R}(\boldsymbol{\theta})\boldsymbol{p_i} + \boldsymbol{t}$: function that transforms the 3D point $\boldsymbol{p}_i$ with the angle vector $\boldsymbol{\theta}$ and translation vector $\boldsymbol{t}$.
- $\hat{\boldsymbol{p}}_i = T_r(\boldsymbol{\theta}, \boldsymbol{t}, \boldsymbol{p}_i)$: transformed point of $\boldsymbol{p}_i$.
- $P^c = \{\boldsymbol{p}_1^c, \boldsymbol{p}_2^c, \dots, \boldsymbol{p}_N^c\}$: set of estimated 3D corner points of the chessboard from the point cloud. N is the number of the corners in the chessboard.
- $\mathrm{x}_i = (u_i, v_i)^T$: coordinates of 2D pixel.
- $\mathrm{X}^c = \{\mathrm{x}_1^c, \mathrm{x}_2^c, \dots, \mathrm{x}_N^c\}$: set of detected 2D corner pixels of the chessboard from the image.

4. Corner Estimation from the Point Cloud

This section explains the detailed process of corner estimation from one frame of the point cloud data obtained by the LiDAR sensor. All concerned coordinates are located in the LiDAR coordinate system in this section.

4.1. Automatic Detection of the Chessboard

This subsection describes the procedure of automatic extraction of the points that are reflected from the chessboard. The points discussed in this subsection refer to all points in one frame of the point cloud.

4.1.1. Segmentation of the Point Cloud

Region growing [28] is often used to segment the point cloud. Region growing estimates the curvature value and the normal vector of each point based on the plane constructed by its kNN points. Then the method clusters the points according to the Euclidean distance and the angle of normal vectors of points. RANdom Sample Consensus (RANSAC) [29] is also used for the shape extraction. For example, the RANSAC of the planar model is applied for the plane fitting and extraction from the point cloud. However, both methods encounter challenges while processing the sparse and non-uniformly distributed point cloud, which may be generated by a single frame scanning of the mobile LiDAR sensor like Velodyne HDL-32e in this work.

The scanline-based segmentation methods are suitable for processing this kind of point cloud, such as the method in [30]. This method first cluster a single frame of the point cloud into scanline segments according to change of distance and direction between successive points along the scan direction. Then, scanline segments are agglomerated into object segments based on their similarity. This method showed stable segmentation result from the experimental results and thus we apply it for segmentation in this work.

4.1.2. Finding the Chessboard from the Segments

After the segmentation of the point cloud, the segment of the chessboard needs to be correctly identified. We use characteristics such as the planarity, bounds, and points distribution of segments as the conditions for filtering the segments automatically.

To reduce the computational cost, we first filter out some improbable segments based on the theoretical number n_{theo} of points of a segment defined in Equation (1). n_{theo} represents the theoretically maximum number of points and calculated from the vertical and horizontal angle of the LiDAR when the chessboard is parallel to the rotation axis of Velodyne, as shown in Figure 3. Segments in which the number of points fall within the interval $[\epsilon_{theo} n_{theo}, n_{theo}]$ are further processed, where ϵ_{theo} is a coefficient and empirically set to 0.5 for this study.

$$n_{theo} \approx \lfloor \frac{d_W}{2r\sin(\frac{\Delta h}{2})} \rfloor \lfloor \frac{d_H}{2r\sin(\frac{\Delta v}{2})} \rfloor \tag{1}$$

where d_W and d_H are the width and height of the chessboard, r is the Euclidean distance from the segment's centroid to the LiDAR sensor, Δh and Δv represent horizontal and vertical angular resolution, which are 0.16° and 1.33° for Velodyne HDL-32e [31] respectively, $\lfloor \rfloor$ represents the ceiling truncation of a real number.

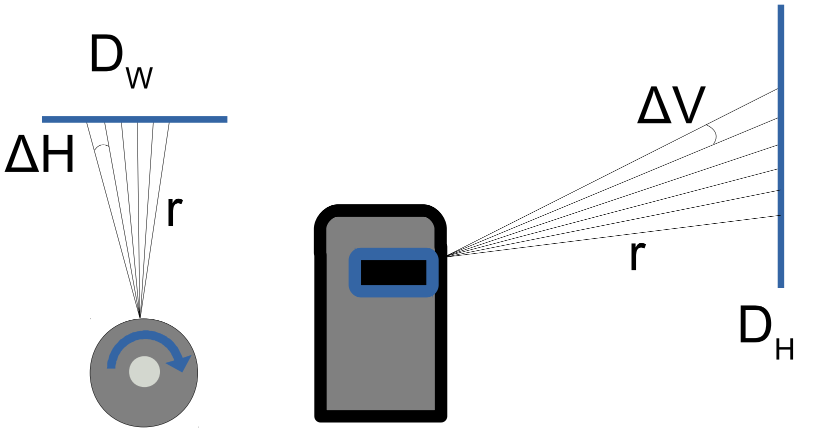


Figure 3. Angular resolution of the used LiDAR in this work. The left figure is the top view and the right one is the side view of the LiDAR and the chessboard.

The planarity of the segment is verified using the Principle Components Analysis (PCA) [32] method. The matrix $M_{n\times 3}$ consisting of all points in the segment is decomposed along three basis vectors $M_b = (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\mu}_3)^T$ with components ratios λ_1, λ_2, λ_3 on each basis vector. The segment whose least ratio λ_3 is less than 0.01 is considered as a planar object.

For a planar segment, all points in this segment are projected to the estimated plane formed by the RANSAC method [29] for further verification. Fitted points are denoted as $M_{n\times 3,f}$. As the final step, the range of the bounding box and the uniformity of the distribution are checked. For manipulation convenience, we rotate coordinates of all points to the chessboard plane and align λ_1, λ_2 to the x-axis and y-axis with the Equation (2). Then the centroid of the chessboard's points $M_{n\times 3,fr}$ is translated to the origin using of Equation (3).

$$M_{n\times 3,fr} = (M_b \cdot M^{\mathsf{T}}_{n\times 3,f})^{\mathsf{T}} \tag{2}$$

$$M_{n\times 3,frt} = M_{n\times 3,fr} - mean(M_{n\times 3,fr}) \tag{3}$$

After the transformation by Equations (2) and (3), the planar segment is transformed to the XOY plane and the centroid of the segment coincides with the origin point. The segment with the bounding box within $[0.8d_W, 1.6d_W]$ and $[0.8d_H, 1.6d_H]$ is considered as a potential chessboard. Uniformity of the point distribution is determined by the difference between the points distribution in four equally divided regions, which is illustrated in Figure 4. Two sample segments that fall in the range of the threshold bounding box are shown in Figure 4. The dashed lines divide the theoretical region of the chessboard into four equal parts. The difference of the points' number in each part is used for uniformity check. For example, as the difference of points' number in each part in Figure 4a is smaller then that in Figure 4b, Figure 4a has better uniformity than Figure 4b. Let us assume that the maximum number of points in the regions is n_{max}, and minimum is n_{min}. The uniformity of the distribution is calculated as $\epsilon_{norm} = 1 - \frac{n_{max} - n_{min}}{n_{all}} \in [0, 1)$, where n_{all} is the total number of points in the segment. A large ϵ_{norm} value indicates that the points are distributed normally. The threshold for uniformity ϵ_{norm} is set to 0.85.

If more than one segment satisfies the above conditions, the segment with greater uniformity is selected. The set of points in the detected chessboard's segment is denoted as $P^M = \{\boldsymbol{p}_1^M, \boldsymbol{p}_2^M, \ldots, \boldsymbol{p}_n^M\}$ (The superscript "M" refers to the chessboard Marker).

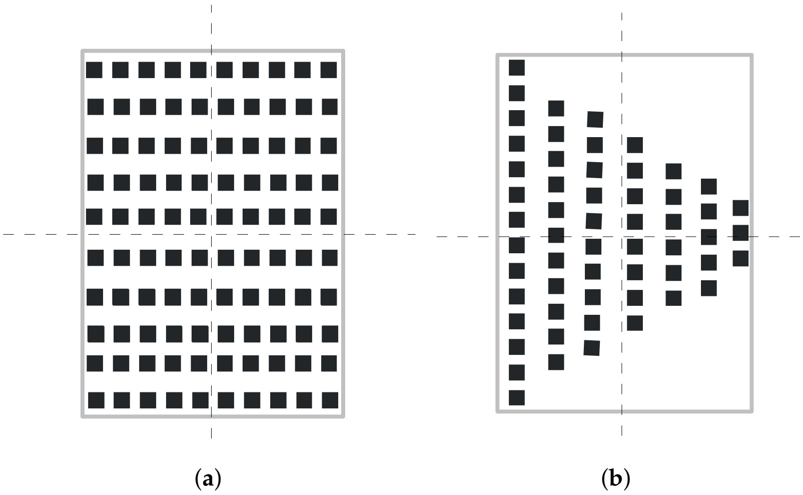


Figure 4. Uniformity of the points distribution. (**a**) shows an example of the point cloud with better uniformity than (**b**).

4.2. Corner Estimation

This subsection explains the principle and detailed process for estimation of the corners from the point cloud of the chessboard. The points discussed in this subsection refers to points of the chessboard only.

4.2.1. Model Formulation

After the chessboard points P^M are automatically extracted, we utilize the intensity of points to estimate the corners. Without loss of generality, we use a model chessboard with a 2 × 3 pattern as shown in Figure 5a. Figure 5b illustrates theoretical laser-reflected points from the chessboard by lasers. The blue points indicate low intensity reflected from black patterns while red points indicate high intensity reflected from white patterns. If we find a pose of the model that make the reflected points best fit the model (Figure 5c), we can use the corners (green points in Figure 5d) of the model to represent the corners of the points.

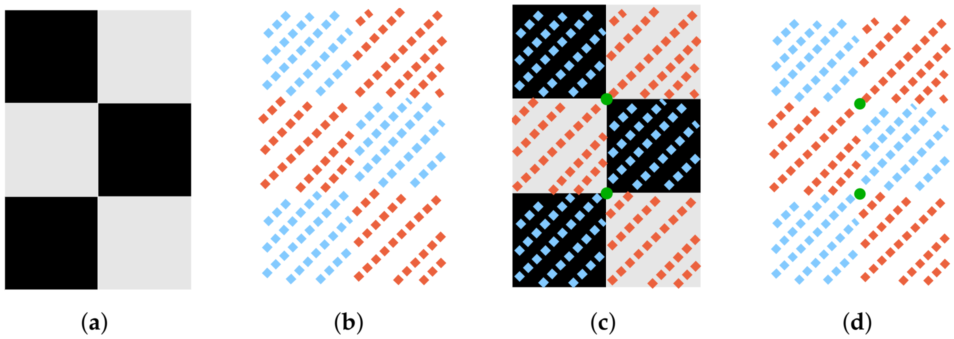


Figure 5. The principle used to estimate corners in the points. (**a**) The chessboard model; (**b**) The scanned point cloud of the chessboard. Colors indicate the intensity (blue for low and red for high reflectance intensity); (**c**) Find a matrix that translates the most 3D points on the corresponding patterns. Green points are estimated corners; (**d**) Consider the corners of the chessboard model as the corners of the point cloud.

In addition, similar to the process in Section 4.1.2, the coordinate system of the chessboard is transformed to the chessboard plane by rotating with the matrix consisting of three PCA vectors and subtracting the mean of the rotated points, as shown in Figure 6. Then, we can treat the points as illustrated in Figure 5. However, for the consistency of further correspondence with corners from images, the directions of principle vectors μ_1, μ_2, μ_3 are deliberately determined. The conditions for the basis vectors are illustrated in Figure 6:

- directions of μ_1, μ_2, μ_3 are defined to obey to the right hand rule.
- direction of μ_3 (the normal of the chessboard) is defined to point to the side of origin of the LiDAR coordinate system.
- angle between μ_1 and x axis of the LiDAR coordinate system is not more than 90°

The rotation matrix $M^M_{XOY^P}$ is defined as $(\mu_1, \mu_2, \mu_3)^T$, where directions of μ_1, μ_2, μ_3 satisfy the aforementioned conditions. After the translation $t^M_{XOY^P}$ by subtracting the mean of the rotated points, μ_1, μ_2, μ_3 and the center of the original point cloud are transformed to x^P-, y^P-, z^P-axis and the origin plane coordinates respectively.

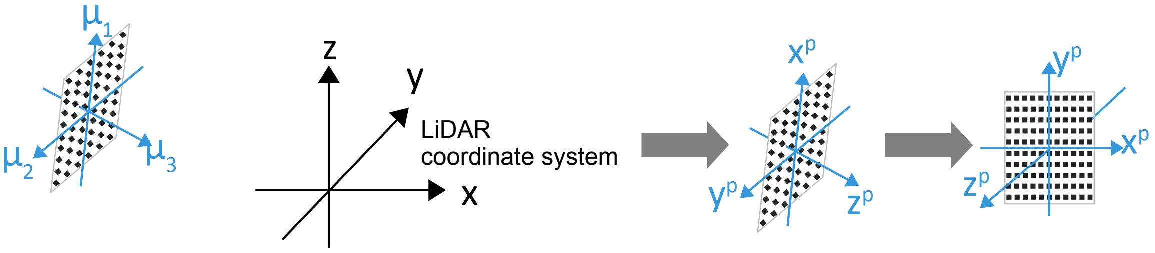


Figure 6. Directions of the basis vectors relative to the LiDAR coordinate system. Blue arrow lines in the left figure represent the basis vectors decomposed by PCA. After transformation with the basis vectors, chessboard's points are mapped to the $X^P OY^P$ (chessboard plane). Then we can apply the model described in Figure 5.

4.2.2. Correspondence of Intensity and Color

Another problem to solve before the formulation of the cost function is the correspondence between the reflectance intensity and the color of the pattern. There are only two colors in the pattern, black and white. However, the reflectance intensity values distribute discretely, mainly due to the divergence of the laser beam. Figure 7a shows the scatter plot of the intensity values of a real chessboard's point cloud. Further, the absolute value of the reflectance intensity is also affected by the distance.

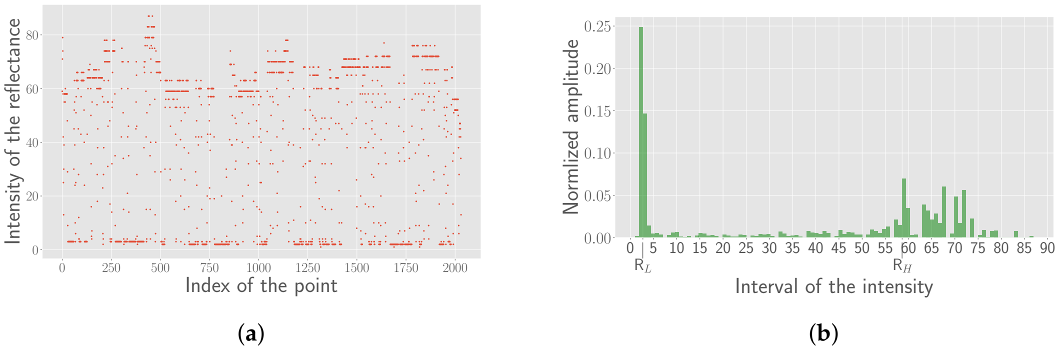


Figure 7. Estimated parameters for each frame. (**a**) Scatter diagram of the intensity for all points in the chessboard; (**b**) The histogram of the intensity. R_L, R_H can be found at the peaks of the two sides.

To process the intensity data adaptively, we define a range called the gray zone denoted as $[\tau_l, \tau_h]$. The points with the intensity less than τ_l are considered as reflected from black patterns and those greater than τ_h are considered as reflected from white patterns. To evaluate the values of τ_l and τ_h, the histogram is created and the bins (R_L, R_H) of peaks in both sides of the mean intensity are detected automatically, as shown in Figure 7b for illustration. The gray zone $[\tau_l, \tau_h]$ is then defined as follows,

$$\begin{cases} \tau_l = & ((\epsilon_g - 1)R_L + R_H)/\epsilon_g \\ \tau_h = & (R_L + (\epsilon_g - 1)R_H)/\epsilon_g \end{cases} \tag{4}$$

where $\epsilon_g \geq 2$ is a constant. ϵ_g is set to 2 for corner estimation with enough points (gray zone will be zero and all points are used) and 4 for error evaluation with enough confidence of the pattern color from the reflectance intensity in this study.

4.2.3. Cost Function and Optimization

After completing the processes explained above, we formulate the cost function for corner estimation. The cost function is formulated based on the constraints of the correspondence between the intensity and color, and defined in Equation (5).

$$C_m = \sum_{i}^{n} f_g(r_i)\{f_{\text{in}}(\hat{\boldsymbol{p}}_i, \boldsymbol{G})\,|c_i - \hat{c}_i|\, f_d(\hat{\boldsymbol{p}}_i, \boldsymbol{V_i}) + [1 - f_{\text{in}}(\hat{\boldsymbol{p}}_i, \boldsymbol{G})] f_d(\hat{\boldsymbol{p}}_i, \boldsymbol{G})\}, \quad \hat{\boldsymbol{p}}_i \in \hat{P}^M \tag{5}$$

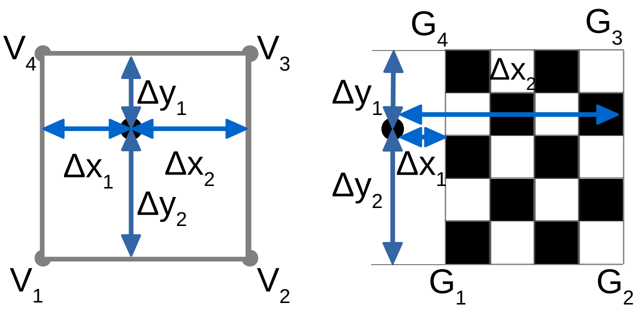
where $\hat{P}^M = \{T_r(\boldsymbol{\theta}^M, \boldsymbol{t}^M, \boldsymbol{p}_i{}^M),\ \boldsymbol{p}_i{}^M \in P^M\}$ is the set of points that are transformed to the XOY plane using the aforementioned processes, and z coordinates of all points in P^M is 0. Namely 3D points are degenerated to 2D after rotation by the matrix of three PCA vectors. Thus, the transformation parameters along the plane are $\boldsymbol{\theta}^M = [0, 0, \theta_z^M]^T$ and $\boldsymbol{t}^M = [t_x^M, t_y^M, 0]^T$. r_i is the reflectance intensity of i-th point. $f_g(r_i)$ is used to determine whether a point falls into the gray zone and is defined in Equation (6). $\boldsymbol{G}$ represents the four corners of the chessboard and $\boldsymbol{V}_i$ represents the four vertices of the grid corresponding to the i-th point.

$$f_g(r_i) = \begin{cases} 1: & r_i \notin [\tau_l, \tau_h] \\ 0: & \text{else} \end{cases} \tag{6}$$

$$f_{in}(\hat{\boldsymbol{p}}_i, \boldsymbol{V}_i) = \begin{cases} 1: & \text{if } \hat{\boldsymbol{p}}_i \text{ in the polygon with vertices } \boldsymbol{G} \\ 0: & \text{else} \end{cases} \tag{7}$$

$$f_d = \min(\Delta x_1, \Delta x_2) + \min(\Delta y_1, \Delta y_2) \tag{8}$$

$f_{\text{in}}(\hat{\boldsymbol{p}}_i, \boldsymbol{V_i})$ indicates whether the polygon with the vertices $\boldsymbol{V_i}$ contains the point $\hat{\boldsymbol{p}}_i$. c_i is the estimated color from reflectance intensity r_i. We define $c_i = 0$ if $r_i < \tau_l$ and $c_i = 1$ if $r_i > \tau_h$. $\hat{c}_i$ represents the color of the pattern the $\hat{p}_i$ falls in. It is defined as 0 for black and 1 for white. We experimentally use $L1$ distance to calculate the cost for points that fall in the chessboard or out the chessboard as shown in Figure 8, and the function f_d is defined in Equation (8). Figure 8a shows the situation that a point $\hat{p}_i$ falls in the wrong pattern constructed by $\boldsymbol{V}_i$. $\Delta x_1, \Delta x_2$ represent the distances from the point $\hat{p}_i$ to the two sides of the pattern and $\Delta y_1, \Delta y_2$ represent the distances to the other two sides. Figure 8b shows the situation that a point falls out of the chessboard's region. Similarly, $\Delta x_1, \Delta x_2$ represent the distances from the point $\hat{p}_i$ to the two sides of the chessboard and $\Delta y_1, \Delta y_2$ represent the distances to the other two sides of the chessboard in Figure 8b.



(**a**) (**b**)

Figure 8. Cost definition for corner detection in the point cloud. (**a**) An example of the point falling into the wrong pattern. The square represents a white pattern of the chessboard; (**b**) An example of the point falling out of the chessboard model; (**a**) describes the first term and (**b**) describes the second term of the cost function (Equation (5)).

Since the cost function C_m is discontinuous, it is impractical to derive for optimization. Thus, we utilize the Powell optimization method [12] that needs no derivative. We used the implementation by SciPy [33] in this work. As mentioned before, the set of chessboard's points are almost fitted to the chessboard model after the transformation. Therefore, all three parameters θ_z^M, t_x^M, t_y^M are initialized with zero when the Powell's method is applied.

5. Extrinsic Calibration Estimation

5.1. Corner Estimation from the Image

There exist many methods for corner detection in perspective images or even in distorted or blurred images [23,34]. In this work, we utilize the method in [23] to detect the corners from the panoramic images directly.

5.2. Correspondence of the 3D-2D Corners

After the corners on the images are detected, the detected corners both on the image and in the point cloud are corresponded by defining the common counting order that starts from left-down side of the chessboard.

5.3. Initial Value by PnP

With these corresponding 3D-2D pair points, an initial value for nonlinear optimization can be obtained by estimating the central absolute pose with the UPnP (Unified Perspective-n-Point) method [13]. We used the implementation by OpenGV [35].

5.4. Refinement with Nonlinear Optimization

We use the difference of inclination angle and azimuth angle in the spherical coordinate system as the error metric for optimization to be independent of the panoramic image projection models. For the i-th 3D-2D pair, the residual is calculated as $E_{err} = \sqrt{f_{p2a}^2(\hat{\boldsymbol{p}}_i) - f_{\mathrm{x}2a}^2(\mathrm{x}_i)}$, where f_{p2a} and $f_{\mathrm{x}2a}$ convert a 3D point and a pixel to the inclination angle and azimuth angle, respectively, in the spherical coordinate system.

6. Experimental Results and Error Evaluation

The setup of the sensors and the measurement environment in this work are introduced in Section 6.1. To evaluate the accuracy of 3D corner detection with the proposed method, simulation results under different conditions are presented in Section 6.2. Then the method is applied to the real data for 3D corner detection of the chessboard's point cloud. Some results of detected 3D corners as well as 2D corners of the panoramic images are shown in Section 6.3. Section 6.4 presents the extrinsic parameters estimated with the 3D-2D corner correspondences. Finally, quantitative and qualitative evaluations are performed in Sections 6.5 and 6.6 respectively.

6.1. Setup

We setup the system with a Velodyne HDL-32e LiDAR sensor mounted atop a Ladybug3 camera as shown in Figure 9a. The chessboard used in this work is constructed by 6 × 8 grids with a side length of 7.5 cm as shown in Figure 9b. Velodyne rotates at 600 rpm and the resolution of the panoramic image output by Ladybug3 is set to 8000 × 6000.

In total, we captured 20 frames of the chessboard. For each horizontal camera of the Ladybug3, the chessboard is placed at 4 different places. The top and side views of the 20 frames are shown in Figure 10.

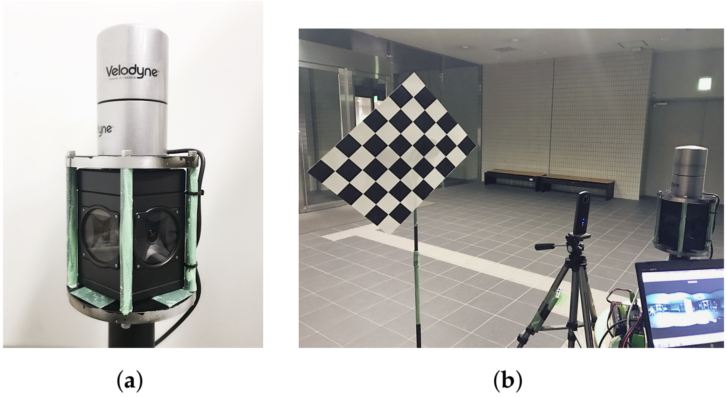


Figure 9. Setup. (**a**) the image of the setup for two sensors; (**b**) a scene for the data acquisition.

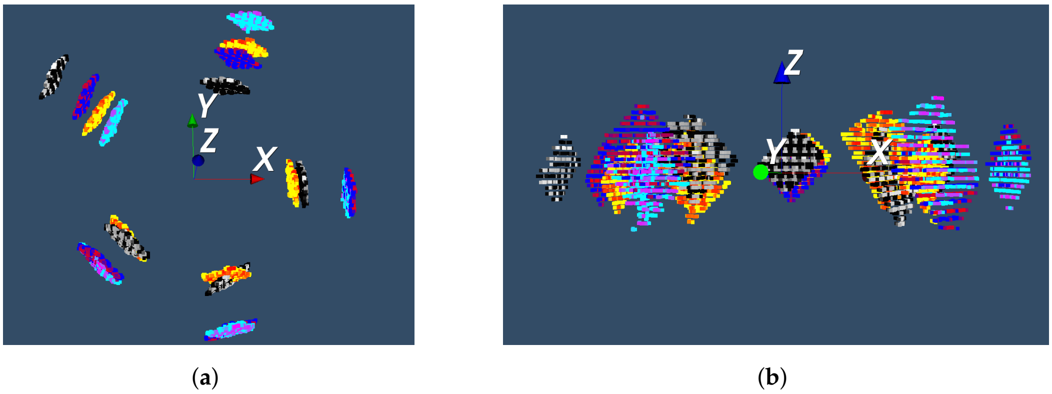


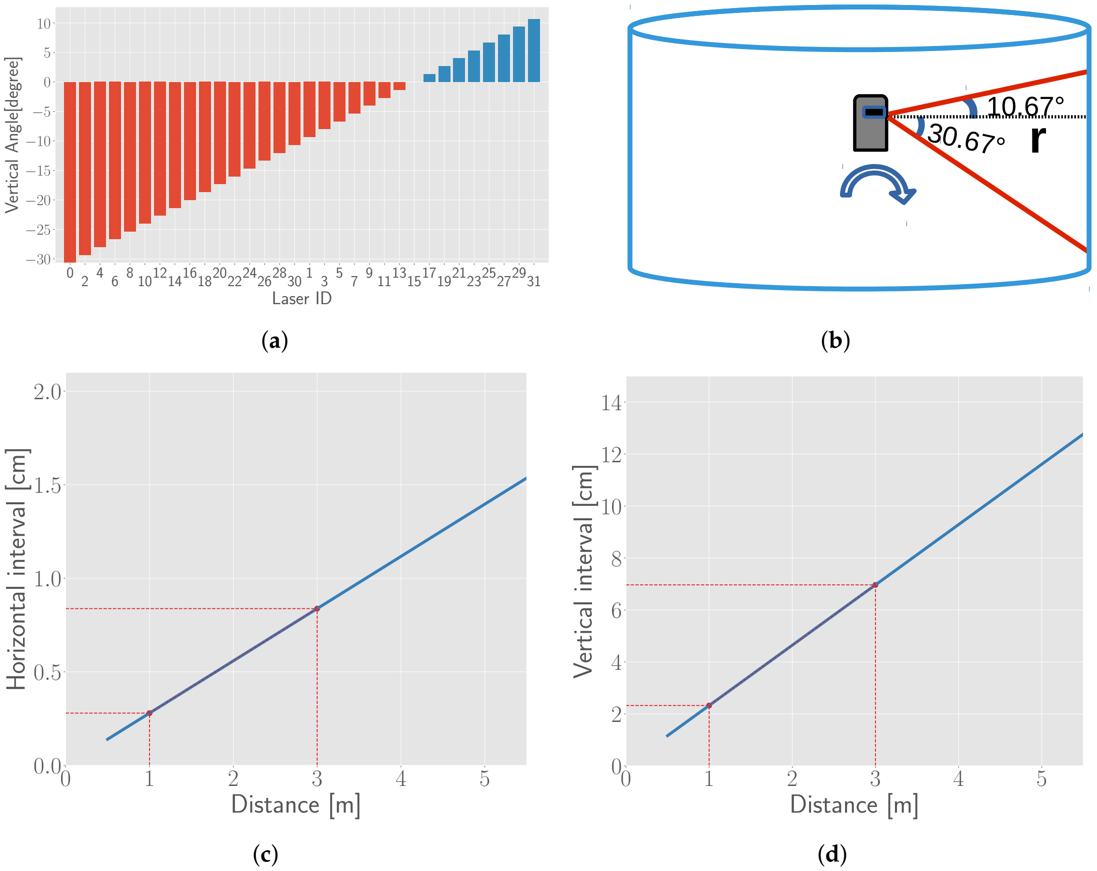
Figure 10. Distribution of the 20 chessboard positions. The chessboard is captured by the LiDAR from different heights and angles. The length of the coordinate axis is 1 m. Four groups of colors represent four positions of the chessboard for each horizontal camera. (**a**) Top view of the point clouds of the chessboard; (**b**) Side view of the point clouds of the chessboard.

6.2. Simulation for Corner Detection Error in the Point Cloud

To evaluate the error of estimated corners from the point cloud, the ground truth is indispensable. However, it is difficult to obtain ground truth from the real data. Thus, we simulate the scanned point cloud of a generated chessboard model and record the corners' coordinates of the chessboard as the ground truth for evaluation. Subsequently, the error is evaluated by comparing the estimated results with the ground truth.

6.2.1. Simulation of the Point Cloud

There are many aspects that affect the measured distance and intensity of the points reflected from the chessboard, such as distance, noise, divergence of the laser beam and the pose of the chessboard. The theoretical interval between two laser beams and two successive points along the scanning are calculated based on the distance and the LiDAR laser distribution, as shown in Figure 11. The noise is added to the points along x-, y-, z-axis independently. The probability model of the noise is considered as a Gaussian distribution. The base line of the deviation σ for noise is empirically set to 0.0016, 0.0016 and 0.01 m with the mean $\mu = 0$ for each axis. Simulated points are transformed with a rotation matrix and translation vector.



(**a**)

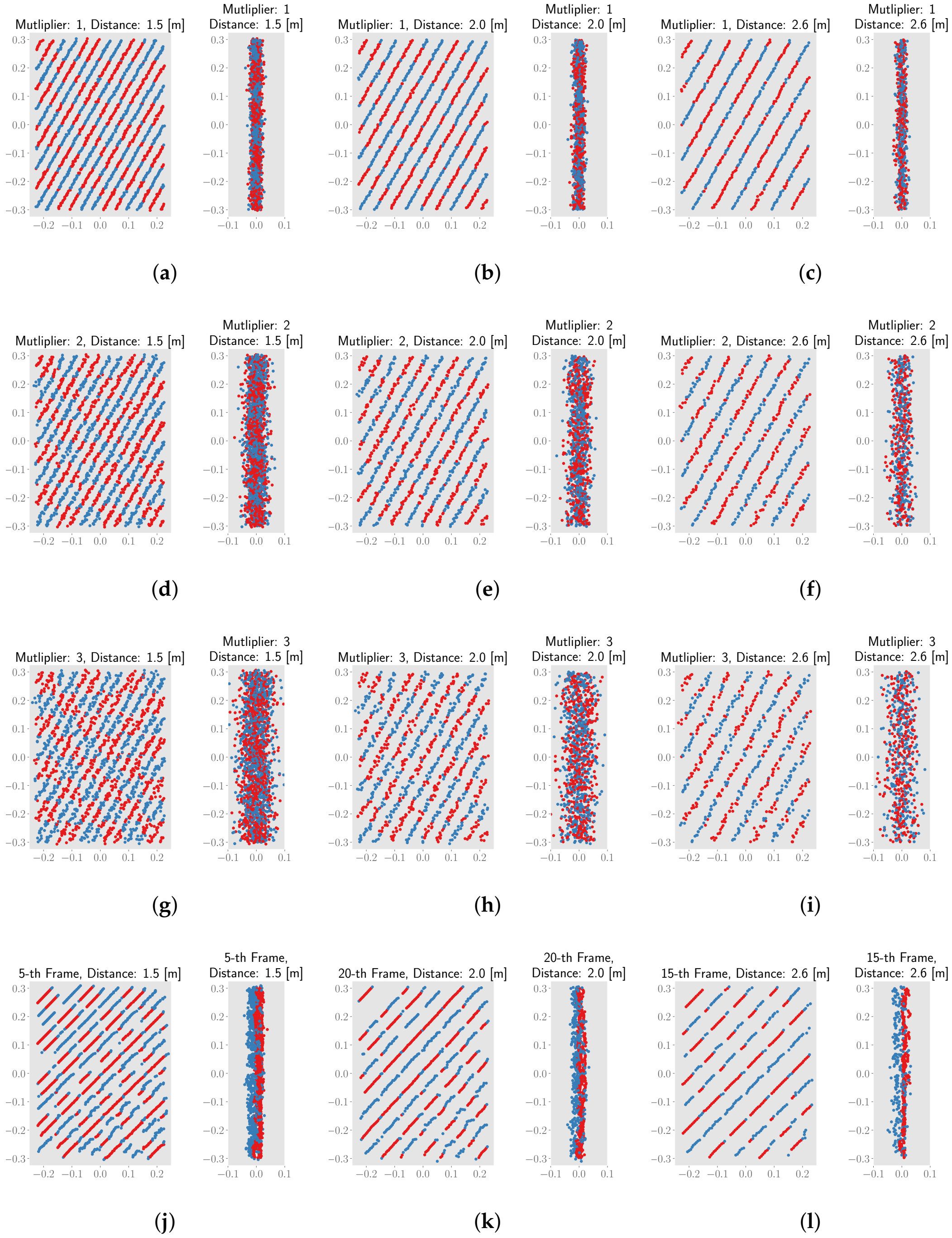
(**b**)

(**c**)

(**d**)

Figure 11. Vertical field view of Velodyne HDL-32e and the relationship between interval and distance. (**a**) Vertical angles of Velodyne HDL-32e; (**b**) Vertical field of view; (**c**) Relationship between the horizontal interval of two adjacent lasers and noise of the point cloud; (**d**) Relationship between the interval of two successive points of the scanline and the distance of the chessboard. Red lines in (**c**,**d**) show the range of chessboard's distance we place in this work.

Figure 12 shows some simulated point clouds for different noise and distance conditions. The last row shows the points from real data for comparison. By comparing the front view of the points clouds between the first row and the last row in Figure 12, we can see that the points with baseline noise are similar to the real data. As for the side view of real data, we see that blue points and red points are generally distributed separately. This indicates that the noise of distance differs for different intensity for real measurements. However, as explained in Section 4.2.3, since the proposed corners estimation method project all points onto the XOY plane, the noise along z-axis can be ignored. Thus, the difference between the simulated data and real data along z-axis will not affect the error evaluation of the proposed corner estimation method.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

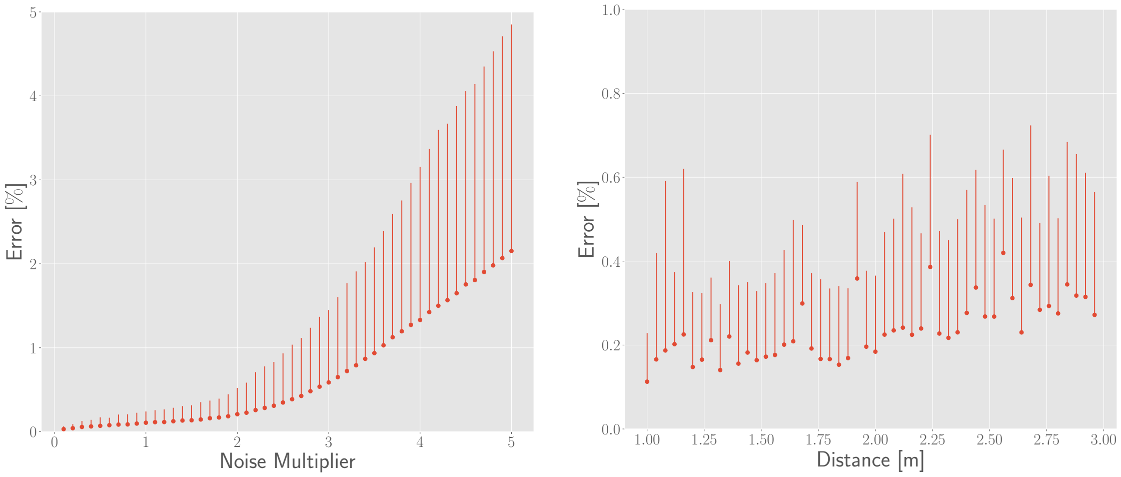
(**g**) (**h**) (**i**)

(**j**) (**k**) (**l**)

Figure 12. Front view and side view of chessboard's point clouds from simulation results and real data. (**a**–**c**) Simulated point clouds with the multiplier 1 at different distances; (**d**–**f**) Simulated point clouds with the multiplier 2 at different distances; (**g**–**i**) Simulated point clouds with the multiplier 3 at different distances; (**j**–**l**) Real point clouds at different distances.

6.2.2. Error Results from the Simulation

We applied the proposed method to simulated points with varying noise and distance conditions to estimate the coordinates of the point cloud. The distance error is calculated $\frac{\sqrt{\sum_{i=1}^{n}|\hat{p}_i-p_i|^2}}{n}$, where $\hat{p}_i$ depicts coordinates of the estimated corner and p_i is the ground truth. Then the relative error is calculated with the unit of percentage by dividing the side length (7.5 cm in this work) of a pattern in the chessboard. For each noise and distance condition, we repeat the simulation with different random seeds 100 times and calculate the average and standard deviation. Error results are shown in Figure 13. The horizontal axes represent the simulation conditions and the vertical axes represent the relative error. From Figure 13a, we can see that the error and uncertainty of the estimation increases drastically as the noise of measurement increases. The influence of distance on the error is relatively lower than that of noise, as shown in Figure 13b.



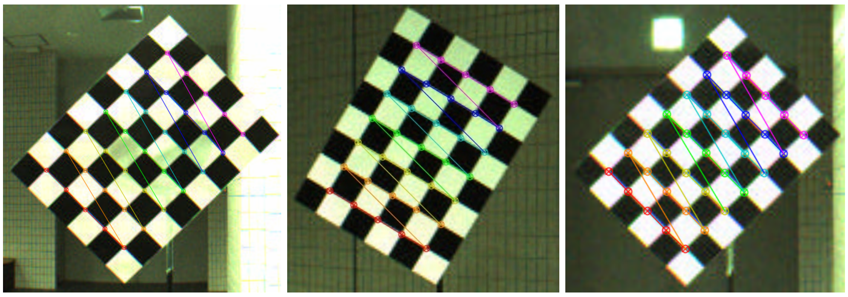
(**a**) (**b**)

Figure 13. Corner detection error by simulation. The horizontal axes represent different simulation conditions and the vertical axes represent the relative error. Red points represent the mean value and the vertical lines represent the 3σ range of the results simulated by 100 times at each simulation condition. (**a**) Relationship between the error and noise of the point cloud at 1 m. The x axis represent the multiplier for the noise baseline; (**b**) Relationship between the error and the distance of the chessboard with the baseline noise.

6.3. Detected Corners

6.3.1. From the Image

We use the method in [23] to detect the corners in the panoramic image. Some example results are shown in Figure 14. We can see that all corners are robustly detected. Compared to conventional vertex-based methods, it is difficult to detect vertices accurately and automatically when the background color is similar to the color of the planar board (lower left of the chessboard in Figure 14c). However, it can be accurately and automatically detected if we use the corners as feature points. For the correspondence with the corners from the point cloud, corners are counted starting from the lower left board.



(**a**) (**b**) (**c**)

Figure 14. Detected corners from the panoramic images. (**a**–**c**) Example results of detected corners from images with different poses and distances.

6.3.2. From the Point Cloud

We applied the proposed corner detection method from the point cloud to real measured data. An example of the estimated corners is shown in Figure 15. Figure 15a shows the position of the chessboard's point cloud and the estimated chessboard model in the LiDAR coordinate system. The zoomed-in image of the chessboard is shown in Figure 15b. The corners of the chessboard are the estimated corners, which can be calculated with M_{XOY}^{M}, t_{XOY}^{M} in Section 4.2.1 and $T_r(\boldsymbol{\theta}^M, \boldsymbol{t}^M)$ in Section 4.2.3. The counting order also starts from the lower left, which is the same as that in the corners from the correspondence with corners from the image. From Figure 15b, we can see that most blue points (low intensity) are mapped on the black patterns, and red points (high intensity) are mapped on the white patterns. This means that the chessboard fits the points well.

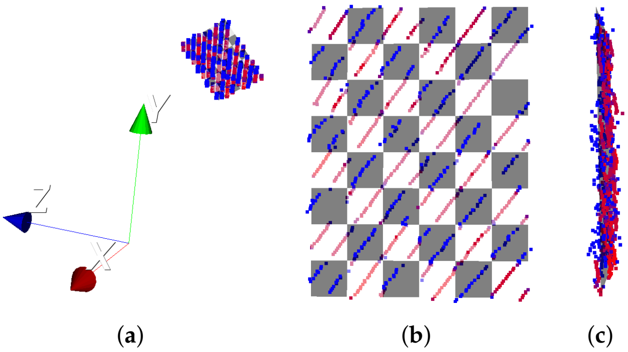


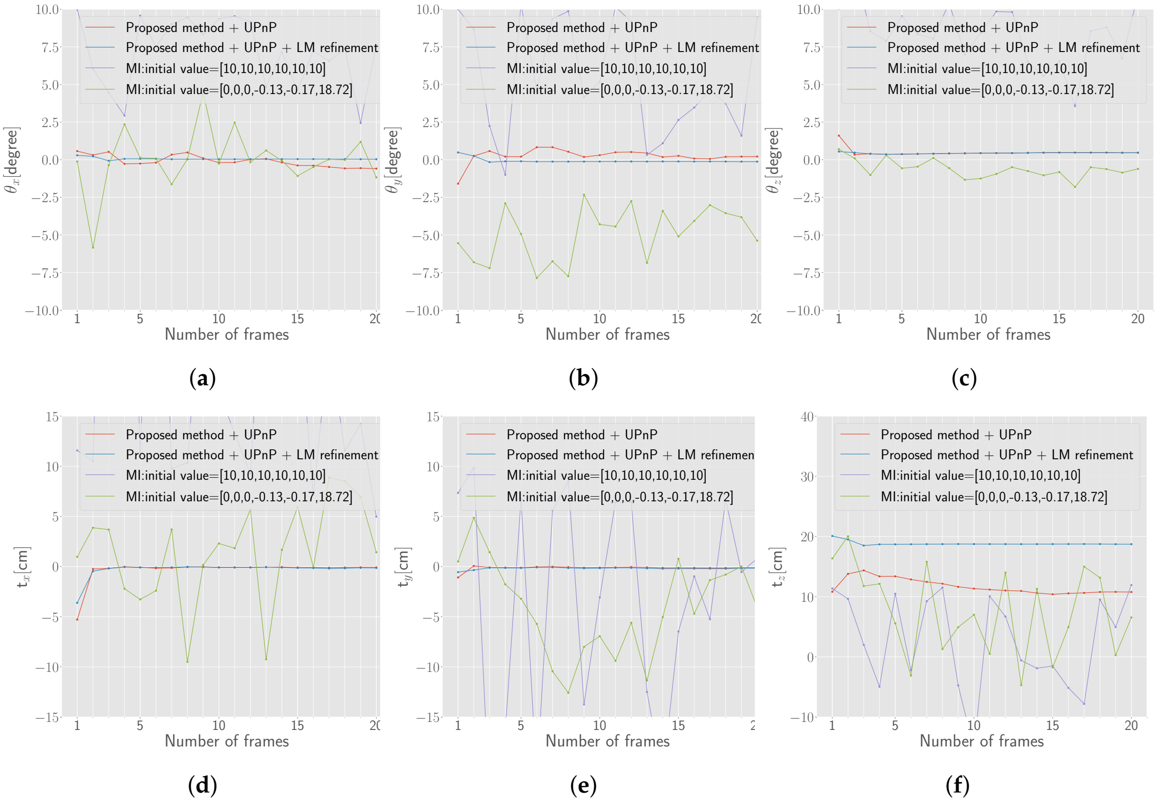
Figure 15. Estimated corners of chessboard. (**a**) The fitted chessboard model of the point cloud in the real Velobug LiDAR coordinate system; (**b**) The front view of the zoom checkerboard; (**c**) The side view of the zoom checkerboard.

6.4. Estimated Extrinsic Parameters

The initial and refined transformation parameters with different numbers of frames are shown in Figure 16. As mentioned above, 20 frames are used for this evaluation. The x-axis represents the number of frames used for the estimation. For example, 5 indicates that the first five frames frame#1 $\sim$ frame#5 are used for the extrinsic calibration. The frames are indexed as in Table 1 to make the variation of the chessboard's point cloud increase in the horizontal direction as the number of frame increase for global optimization. The index of each camera is defined in [36].

Figure 16 depicts the state from when the parameters start to become stable after the nonlinear refinement with more than four frames, while some parameters estimated by UPnP are erratic as the number of frames changes. Generally, there is no significant difference between the results obtained by UPnP and by nonlinear refinement. The translation value along the z-axis estimated by UPnP is 11.7 cm, which differs from 18.7 cm obtained using nonlinear refinement. We manually measure the offset between the LiDAR and the panoramic camera of the set. It is difficult to accurately find the center of each sensor, the range of the offset is supposed to be in the range between 17 and 20 cm. Thus, the refined result is more accurate.

As a comparison, we also applied Pandey's online method [8], which also uses laser's reflectance intensity, to the data in this work. The results under different initial guesses are shown in Figure 16. One initial guess is [10°, 10°, 10°, 10 cm, 10 cm, 10 cm] and the other is [0°, 0°, 0°, −0.13 cm, −0.17 cm, 18.72 cm] which are almost the accurate parameters. From the comparison of the two methods, we can conclude that the proposed method is more stable and accurate in terms of both rotation and translation. The main reason why Pandey's method performed not well with the data in the work is considered that the surrounding environment is almost static and the increment of the frames does not increase the variance of the data for the optimization. However, as stated before, the original problem settings of two methods are different. Online calibration methods like Pandey's are very convenient and can estimate an approximate results if a good initial value is provided while the proposed methods is target-based, but it can provide accurate and stable extrinsic calibration result.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 16. Estimated parameters by the proposed method and Pandey's Mutual Information (MI) method [8] with different initial values as the numbers of frame increases. (**a**–**c**) Rotation angle along each axis; (**d**–**f**) Translation along each axis.

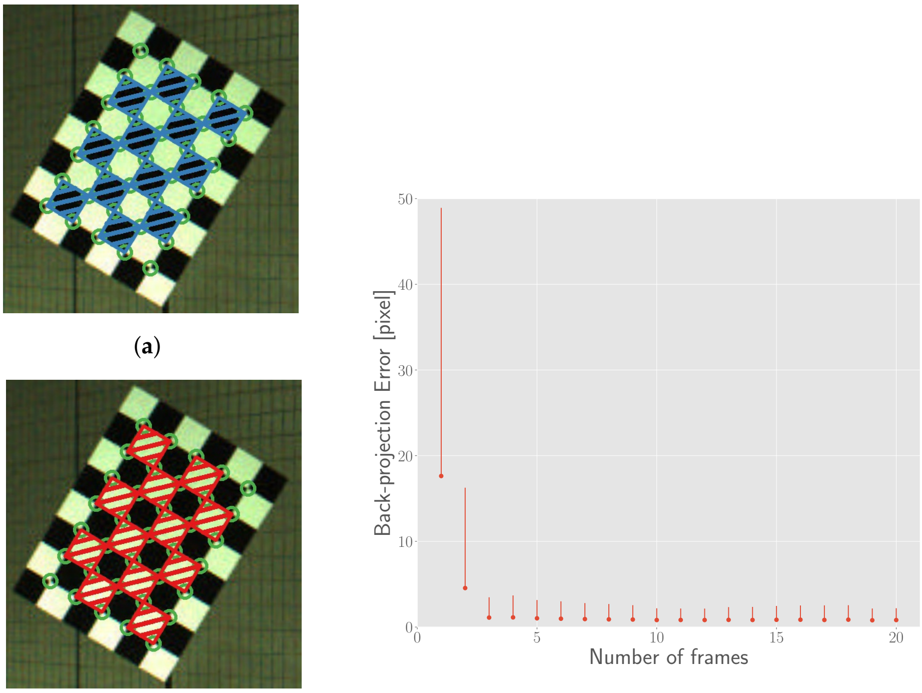
Table 1. Relationship between the index of the frame and the index of the camera.

Camera Index	**0**	**1**	**2**	**3**	**4**
Frame Index	1, 6, 11, 16	2, 7, 12, 17	3, 8, 13, 18	4, 9, 14, 19	5, 10, 15, 20

6.5. Re-Projection Error

To quantitatively evaluate two groups of estimated parameters, we define a re-projection error metric based on intensity, which is similar to Equation (5) except that the error metric is defined on the image plane. To increase the confidence, only the points that are contained within the polygon of the detected corners on the images are counted. Moreover, the gray zone of intensity is increased for the same consideration and ϵ_g in Equation (4) is set to 4.

For the error calculation, we first transform the point cloud of the chessboard with estimated extrinsic parameters. After mapping all 3D points of the transformed chessboard point cloud at the panoramic image based on an equirectangular projection model, we calculate the error on the chessboard plane of the panoramic image. Only the point that is mapped within the quadrilateral region constructed by the four detected corners on the image is counted, as shown in Figure 17a,b. As the gray zone is identified, we can classify the mapped points into white or black colors. If the estimated color of a mapped point differs from the color of the quadrilateral it falls in, the error of this point is calculated using Equation (8).



(**a**)

(**b**)

(**c**)

Figure 17. Re-projection error calculation and results. (**a**,**b**) Shaded quadrilaterals show the regions of black and white patterns respectively. Points mapped into these regions are counted for error calculation; (**c**) The errors for parameters estimated by different numbers of frames. The point and vertical line represent the mean and 3σ range of all errors calculated by applying the estimated parameters to the different number of frames.

In the measurement, the farther the chessboard is, the smaller the occupied pixels of chessboard on the image will be. Because the re-projection error is measured with absolute pixel, the error of farther chessboard will be relatively smaller. To ensure that the error metric is unaffected by distance, we normalize the error by multiplying the value of the distance with the consideration that the size of an object on the image is generally proportional to the reciprocal of the distance. Moreover, the "attendance rate" of the point, which indicates the rate at which points are counted for error calculation, is also taken into account for the overall re-projection performance. The final formula is defined in Equation (9)

$$e = \frac{C_m}{N_c} \cdot r^M \cdot \frac{P_c N_a}{P_a N_c} \tag{9}$$

Here, C_m is the cost function defined in Equation (5), r_M is the Euclidean distance of the chessboard, P_a is the total number of the patterns in the chessboard, P_c is the number of the quadrilaterals constructed by detected corners for calculating the re-projection error, N_a is the points number of chessboard's point cloud, N_c is the number of points that fall into the quadrilaterals constructed by detected corners, r^M is a factor of the error caused by the distance and $\frac{P_c N_a}{P_a N_c}$ is the penalty factor by dividing the "attendance rate". The "attendance rate" is derived from $\frac{N_c}{P_c} \div \frac{N_a}{P_a}$, which is the ratio of the average number of point in each pattern of the region (Figure 17a,b) only counted for error calculation and the average number of point in each pattern of the whole chessboard.

The re-projection error with the metric defined above is shown in Figure 17c. The point and vertical line represent the mean and 3σ range of all errors calculated by applying the estimated parameters to 20 frames. The length of one grid of the chessboard in the 4000 × 8000 panoramic image at 1 m is approximately 100 pixels. The back projection error is 0.8 pixel when the optimization begins to converge after 4 frames, and the relative error to the side length is 0.8%.

6.6. Re-Projection Results

For qualitative evaluation, we apply the final extrinsic transformation matrix estimated by the proposed method to rotate point cloud and project it to the image. The result of re-projected corners is shown in Figure 18. The large green circles and cyan lines indicate the detected corners. The small pink circles in large green circles and lines indicate re-projected corners estimated from the point cloud. The large blue and red circles represent the start and end for counting the corners. We can see that the re-projected corners estimated from the point cloud almost coincide with the corners detected in the image.

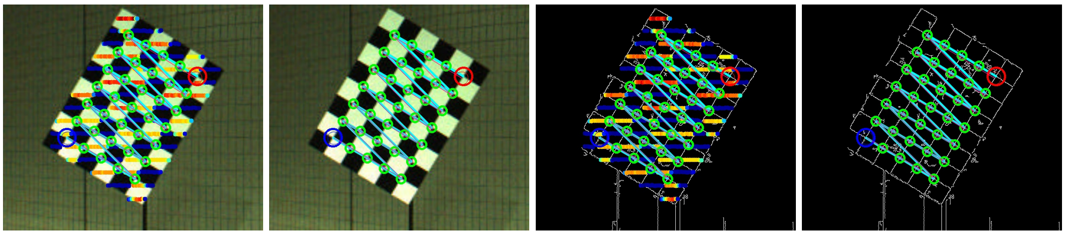
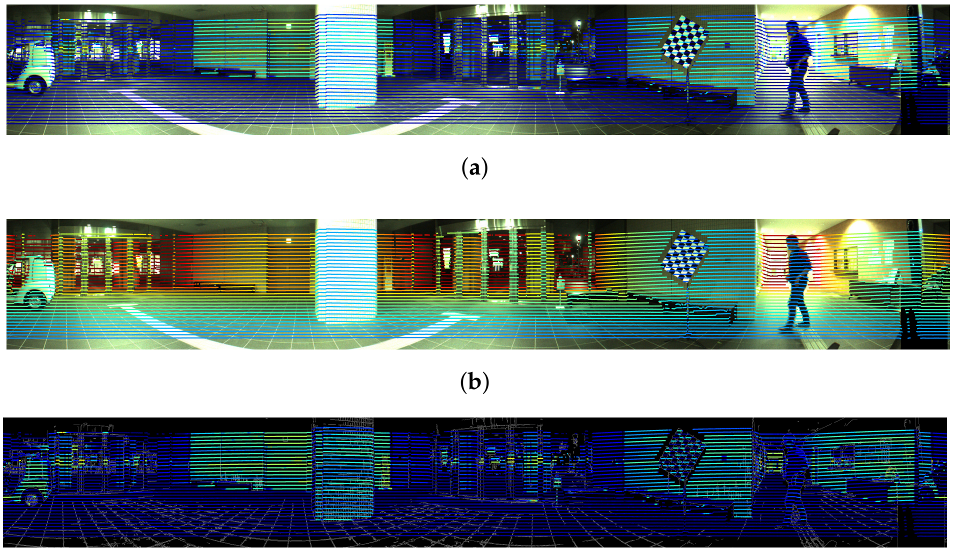


Figure 18. Re-projected corners and points of the chessboard (best viewed when zoomed in). Big green circles and cyan lines indicate the detected corners. Small pink circles in big green circles and pink lines indicate re-projected corners estimated from the point cloud. Big blue and red circles represent the start and end for counting the corners. Blue points indicate low reflectance intensity and red points indicate high reflectance intensity.

Re-projection results of all points within one frame are as listed in Figures 19–21. For better visualization, points colored according to the intensity and distance are mapped to the original RGB images and the edges-extracted images, respectively. Figure 19 shows the overall results on the panoramic image. Details of individual objects are shown in Figure 20. We can see that the bounds generated by the change of re-projected 3D points with different intensities and distance fit the edges of the image exactly, for example the fourth images in Figure 20a,b. Some inconsistencies occur due to the occlusion, for example the lower left and upper right parts of the chessboard in Figure 20a and the upper part of the human in Figure 20b. The final colored point cloud is shown in Figure 21. Red points indicate the region occluded by the chessboard.

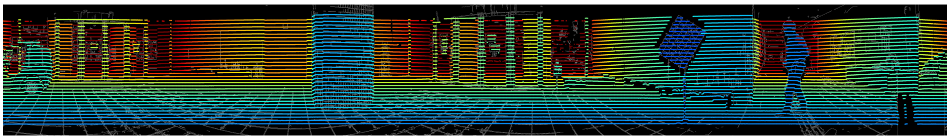


(**a**)

(**b**)

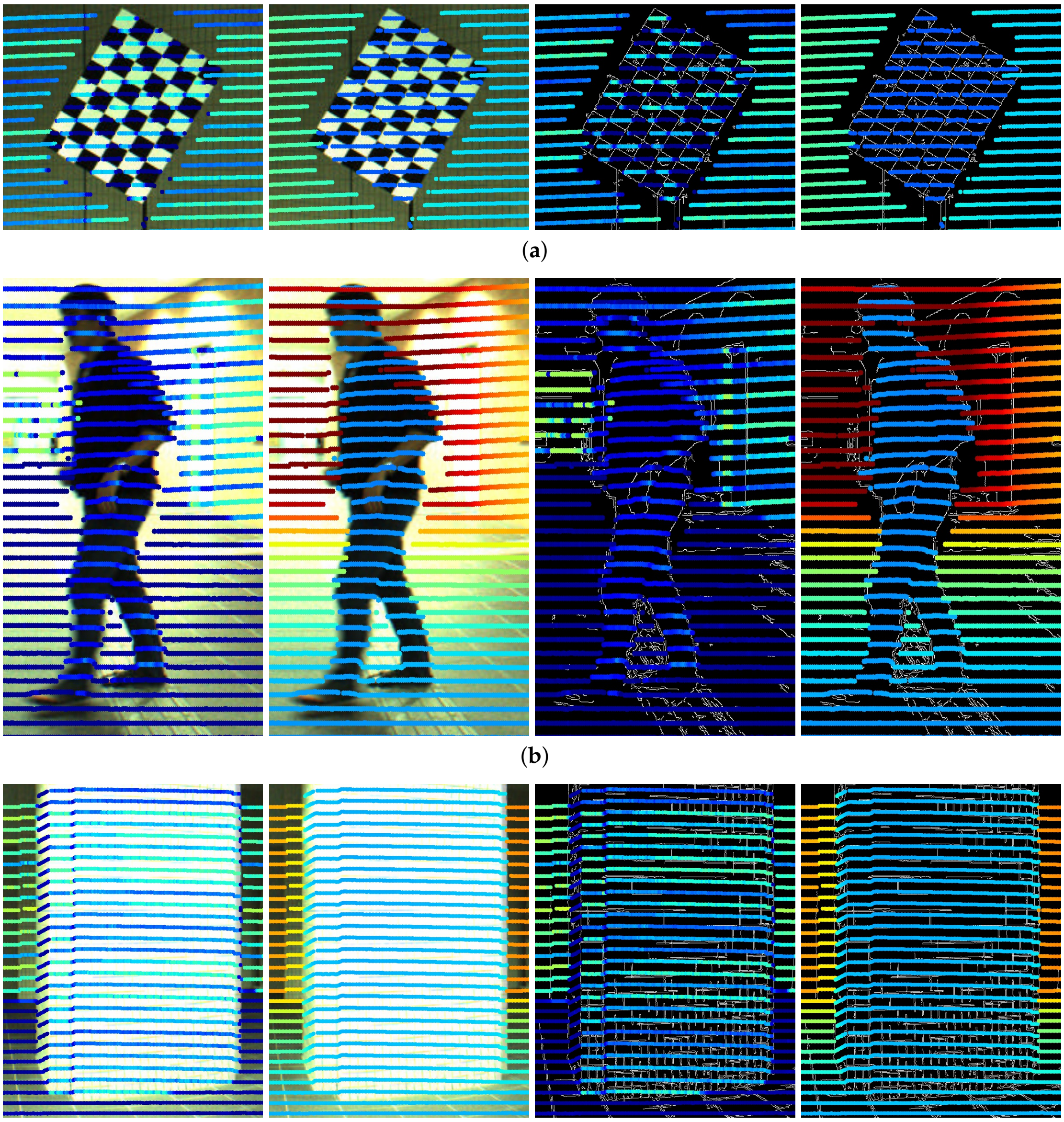
(**c**)

Figure 19. *Cont.*



(**d**)

Figure 19. Re-projection results. (**a**) All re-projected points on the color panoramic image and colored by intensity; (**b**) All re-projected points on the color panoramic image and colored by distance; (**c**) Re-projected result on edge extracted image of all points colored by intensity; (**d**) Re-projected result on edge extracted image of all points colored by distance.

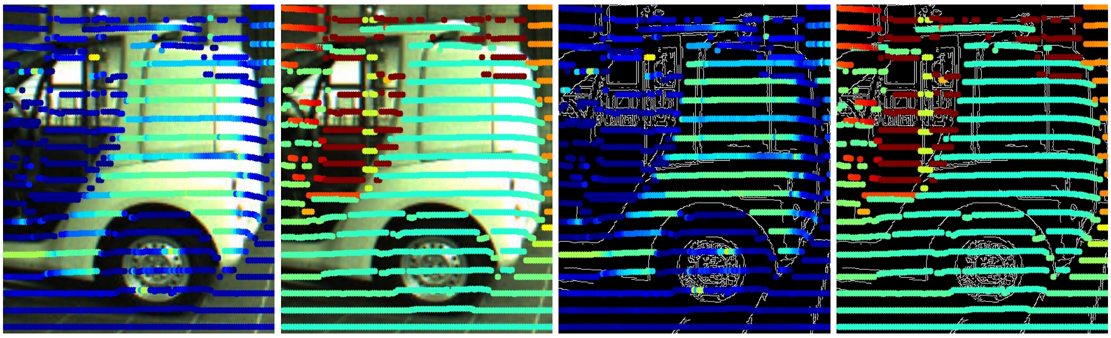


(**a**)

(**b**)

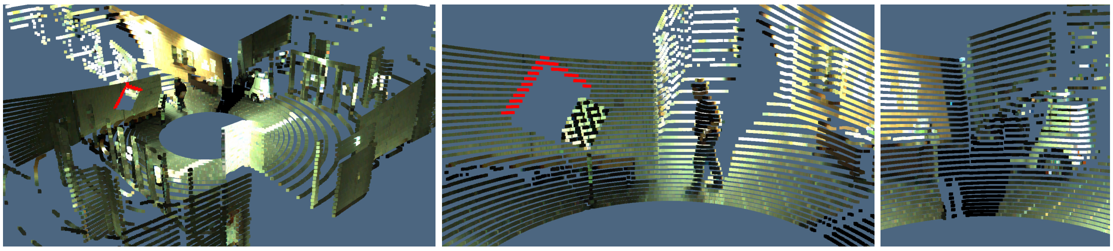
(**c**)

Figure 20. *Cont.*



(**d**)

Figure 20. Zoomed details of re-projected points. (**a**–**d**) Re-projected results on chessboard, human, pillar and car respectively. Each column represents re-projected points colored by intensity and distance on original RGB images and edges-extracted images. Blue indicates low value and red indicates high value.



(**a**) (**b**) (**c**)

Figure 21. Projection of the RGB information from the image to the point cloud with the estimated extrinsic parameters. (**a**) An example of the colored point cloud; (**b**) The zoomed view of the chessboard in (**a**). The red points in (**a**,**b**) are occluded region caused by the chessboard; (**c**) The zoomed view of the car in (**a**).

7. Discussions

- *Automatic segmentation.* As the first step of the proposed method, automatic segmentation is performed. The current segmentation method is only based on the distance information, which needs the chessboard to be spatially separated from the surrounding objects. Nevertheless, slight under-segmentation caused by the stand of the chessboard or over-segmentation caused by the measurement noise may still occur. The degree of mis-segmentation generated by the segmentation method used in this work is experimentally shown to be negligible for the corners estimation with the overall optimization of the proposed method.
- *Simulation.* To evaluate the performance for the corner estimation with the proposed method, we approximately simulated the points by considering the probability model of the distance as Gaussian distribution. However, the probability model for the noise of reflectance intensity, which is an aspect for corners estimation, is not considered. Under the influence of reflectance intensity, the real error of corner estimation is supposed to be higher than the simulated results in this work. This is one of the reasons why the relative error for corners estimation is about 0.2%, as shown in Figure 13b, and the final re-projection error increased to 0.8% in Section 6.5. For a more precise simulation, the probability model of the reflectance value related to the incidence angle, the distance and the divergence of laser beam needs to be formulated.
- *Chessboard.* As shown in Figure 11, both the horizontal and vertical intervals increase as the distance increases. To gather enough information for corner estimation, the side length of one grid in the chessboard is suggested to be greater than 1.5 times of the theoretical vertical interval at the farthest place. In addition, the intersection angle between the diagonal line of the chessboard

and the z-axis of the LiDAR is suggested to be less than 15° to enable the scanning of as many patterns as possible.

We use the panoramic image for calibration, therefore, to remain unaffected by the stitching error, it is better to place the chessboard in the center of the field of view for each camera.

- *Correspondence of 3D and 2D corners.* In this work, a chessboard with 6∼8 patterns is used and the counting order is defined as starting from the of the chessboard for automatic correspondence. To make the "lower left" identified correctly, the chessboard should be captured to make the "lower left" of the real chessboard be same with that of chessboard in the image during the data acquisition. Also, the direction of z-axis of the two sensors should be almost consistent shown as in Figure 9b. However, these restrictions can be released with the introduction of asymmetrical patterns in practical use.

8. Conclusions and Future Works

In this work, we proposed a novel and fully automatic reflectance intensity based calibration method for LiDAR and panoramic camera system. Compared to existing methods, we make use of the corners information of the chessboard's point cloud instead of the edges information. Corners of the sparse and noisy chessboard's point cloud are estimated by solving the optimization problem with the intensity information. After the correspondence of corners both on the image and in the point cloud, the extrinsic transformation matrix is generated by solving the nonlinear optimization problem. To evaluate the performance of the corner estimation from the point cloud, we simulated the points and compared the estimated corners with the ground truth. We applied the proposed method to an equipment set consisting of a Velodyne LiDAR sensor and a Ladybug3 panoramic camera. Quantitative evaluation of the accuracy of the proposed method was performed with a proposed intensity-based re-projection error metric. The re-projected results were verified by qualitative evaluation. The Python implementation of the proposed method can be downloaded from https://github.com/mfxox/ILCC.

In this work we evaluate the proposed method only with the Velodyne LiDAR HDL-32e sensor, we will evaluate on more types of sensor in the future. In practical 3D color mapping applications with the fusion of two sensor modalities , occlusion due to the difference in views must be addressed in the future. In addition, we aim to implement an online calibration without any information about the correct matrix based on the segmentation of point cloud according to both scanline discontinuity and reflectance intensity.

Acknowledgments: This research was partially supported by JST COI STREAM.

Author Contributions: Weimin Wang and Ken Sakurada proposed the method. Weimin Wang implemented the algorithm, collected the data and performed the experiment, and wrote the manuscript. Weimin Wang and Ken Sakurada contributed to the improvement of the algorithm and the discussion of the results. Ken Sakurada and Nobuo Kawaguchi supervised the entire process of the research and reviewed the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mastin, A.; Kepner, J.; Fisher, J. Automatic registration of LIDAR and optical images of urban scenes. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Miami, FL, USA, 20–25 June 2009.
2. Paparoditis, N.; Papelard, J.P.; Cannelle, B.; Devaux, A.; Soheilian, B.; David, N.; Houzay, E. Stereopolis II: A multi-purpose and multi-sensor 3d mobile mapping system for street visualisation and 3d metrology. *Rev. Fr. Photogramm. Télédétec.* **2012**, *200*, 69–79.
3. Szarvas, M.; Sakai, U.; Ogata, J. Real-time pedestrian detection using LIDAR and convolutional neural networks. In Proceedings of the 2006 IEEE Intelligent Vehicles Symposium (IV), Tokyo, Japan, 13–15 June 2006.
4. Premebida, C.; Nunes, U. Fusing LIDAR, camera and semantic information: A context-based approach for pedestrian detection. *Int. J. Robot. Res. (IJRR)* **2013**, *32*, 371–384.

5. Schlosser, J.; Chow, C.K.; Kira, Z. Fusing LIDAR and images for pedestrian detection using convolutional neural networks. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016.
6. Levinson, J.; Thrun, S. Automatic Online Calibration of Cameras and Lasers. Available online: http://roboticsproceedings.org/rss09/p29.pdf (accessed on 27 April 2017).
7. Taylor, Z.; Nieto, J.; Johnson, D. Multi-modal sensor calibration using a gradient orientation measure. *J. Field Robot. (JFR)* **2014**, *32*, 675–695.
8. Pandey, G.; McBride, J.R.; Savarese, S.; Eustice, R.M. Automatic extrinsic calibration of vision and lidar by maximizing mutual information. *J. Field Robot. (JFR)* **2014**, *32*, 696–722.
9. Mirzaei, F.M.; Kottas, D.G.; Roumeliotis, S.I. 3D LIDAR–camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization. *Int. J. Robot. Res. (IJRR)* **2012**, *31*, 452–467.
10. Park, Y.; Yun, S.; Won, C.; Cho, K.; Um, K.; Sim, S. Calibration between color camera and 3D LIDAR instruments with a polygonal planar board. *Sensors* **2014**, *14*, 5333–5353.
11. García-Moreno, A.I.; Hernandez-García, D.E.; Gonzalez-Barbosa, J.J.; Ramírez-Pedraza, A.; Hurtado-Ramos, J.B.; Ornelas-Rodriguez, F.J. Error propagation and uncertainty analysis between 3D laser scanner and camera. *Robot. Auton. Syst.* **2014**, *62*, 782–793.
12. Powell, M.J.D. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Comput. J.* **1964**, *7*, 155–162.
13. Kneip, L.; Li, H.; Seo, Y. UPnP: An optimal O(n) solution to the absolute pose problem with universal applicability. In Proceedings of the European Conference on Computer Vision (ECCV), Zurich, Switzerland, 6–12 September 2014; Springer: Berlin, Germany, 2014; pp. 127–142.
14. Levenberg, K. A method for the solution of certain non-linear problems in least squares. *Q. Appl. Math.* **1944**, *2*, 164–168.
15. Marquardt, D.W. An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Ind. Appl. Math.* **1963**, *11*, 431–441.
16. Zhang, Q.; Pless, R. Extrinsic calibration of a camera and laser range finder (improves camera calibration). In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sendai, Japan, 28 September–2 October 2004.
17. Unnikrishnan, R.; Hebert, M. Fast Extrinsic Calibration of a Laser Rangefinder to a Camera. Available online: http://repository.cmu.edu/robotics/339/ (accessed on 27 April 2017)
18. Pandey, G.; McBride, J.; Savarese, S.; Eustice, R. Extrinsic calibration of a 3D laser scanner and an omnidirectional camera. *IFAC Proc. Vol.* **2010**, *43*, 336–341.
19. Pandey, G.; McBride, J.R.; Eustice, R.M. Ford Campus vision and lidar data set. *Int. J. Robot. Res. (IJRR)* **2011**, *30*, 1543–1552.
20. Scaramuzza, D.; Harati, A.; Siegwart, R. Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes. In Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Diego, CA, USA, 29 October–2 November 2007.
21. Moghadam, P.; Bosse, M.; Zlot, R. Line-based extrinsic calibration of range and image sensors. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013.
22. Gong, X.; Lin, Y.; Liu, J. 3D LIDAR-camera extrinsic calibration using an arbitrary trihedron. *Sensors* **2013**, *13*, 1902–1918.
23. Geiger, A.; Moosmann, F.; Car, O.; Schuster, B. Automatic camera and range sensor calibration using a single shot. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Saint Paul, MN, USA, 14–18 May 2012.
24. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res. (IJRR)* **2013**, *32*, 1231–1237.
25. Atanacio-Jiménez, G.; González-Barbosa, J.J.; Hurtado-Ramos, J.B.; Ornelas-Rodríguez, F.J.; Jiménez-Hernández, H.; García-Ramirez, T.; González-Barbosa, R. LIDAR velodyne HDL-64E calibration using pattern planes. *Int. J. Adv. Robot. Syst.* **2011**, *8*, 59.
26. Zhang, Z. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* **2000**, *22*, 1330–1334.

27. Scaramuzza, D.; Martinelli, A.; Siegwart, R. A toolbox for easily calibrating omnidirectional cameras. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Beijing, China, 9–15 October 2006.
28. Rabbani, T.; Van Den Heuvel, F.; Vosselmann, G. Segmentation of point clouds using smoothness constraint. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. (ISPRS)* **2006**, *36*, 248–253.
29. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395.
30. Wang, W.; Sakurada, K.; Kawaguchi, N. Incremental and enhanced scanline-based segmentation method for surface reconstruction of sparse LiDAR data. *Remote Sens.* **2016**, *8*, 967.
31. Velodyne LiDAR, Inc. *HDL-32E User's Manual*; Velodyne LiDAR, Inc.: San Jose, CA, USA, 2012.
32. Jolliffe, I. *Principal Component Analysis*; Wiley Online Library: Hoboken, NJ, USA, 2002.
33. Jones, E.; Oliphant, T.; Peterson, P. SciPy: Open Source Scientific Tools for Python. Available online: http://www.scipy.org (accessed on 27 April 2017).
34. Rufli, M.; Scaramuzza, D.; Siegwart, R. Automatic detection of checkerboards on blurred and distorted images. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Nice, France, 22–26 September 2008.
35. Kneip, L.; Furgale, P. OpenGV: A unified and generalized approach to real-time calibrated geometric vision. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 1–8.
36. Point Grey Research, Inc. *Technical Application Note (TAN2012009): Geometric Vision Using Ladybug® Cameras*; Point Grey Research, Inc.: Richmond, BC, Canada, 2012.