actuators

MDPI

Article

3D Object Recognition and Localization with a Dense LiDAR Scanner

Hao Geng, Zhiyuan Gao, Guorun Fang and Yangmin Xie *

Shanghai Key Laboratory of Intelligent Manufacturing and Robotics, School of Mechatronics Engineering and Automation, Shanghai University, Shanghai 200444, China; gheric@shu.edu.cn (H.G.); gaoyy2021@shu.edu.cn (Z.G.); guorunfang@shu.edu.cn (G.F.)
* Correspondence: xieym@shu.edu.cn

Abstract: Dense scanning is an effective solution for refined geometrical modeling applications. The previous studies in dense environment modeling mostly focused on data acquisition techniques without emphasizing autonomous target recognition and accurate 3D localization. Therefore, they lacked the capability to output semantic information in the scenes. This article aims to make complementation in this aspect. The critical problems we solved are mainly in two aspects: (1) system calibration to ensure detail-fidelity for the 3D objects with fine structures, (2) fast outlier exclusion to improve 3D boxing accuracy. A lightweight fuzzy neural network is proposed to remove most background outliers, which was proven in experiments to be effective for various objects in different situations. With precise and clean data ensured by the two abovementioned techniques, our system can extract target objects from the original point clouds, and more importantly, accurately estimate their center locations and orientations.

Keywords: 3D Lidar scanning; Lidar calibration; 3D object localization; fuzzy neural network

Citation: Geng, H.; Gao, Z.; Fang, G.; Xie, Y. 3D Object Recognition and Localization with a Dense LiDAR Scanner. *Actuators* **2022**, *11*, 13. https://doi.org/10.3390/act11010013

Academic Editor: Ioan Ursu

Received: 11 November 2021
Accepted: 29 December 2021
Published: 5 January 2022

1. Introduction

With the rapid development of LiDAR (light detection and ranging) technology, it has become the primary environment modeling tool to obtain the 3D geometry of a large-scale space in the form of point clouds. Though with much higher accuracy in range measurement, general commercial scanning system has inferior performance in resolution compared to camera systems. Researchers developed various dense scanning systems with additional DOFs (degree of freedom) of motion and applied them in geological exploration [1], building reconstruction [2], virtual city modeling [3], landslide monitoring [4], and quality inspection for constructions [5]. However, most of the abovementioned work only focused on gaining geometric information, and little has been done to interpret the semantic elements in the space. For some applications, such as the hazards auto-detection in landslide monitoring or the autonomous searching and resecuring with UGVs (unmanned ground vehicle), the recognition and localization for critical targets are beneficial, if not necessary. This paper provides a systematic solution for discovering and localizing the vital targets in the surrounding space by dense laser scanning.

The mechanical structure of our system is similar to the previous studies [1,6–9]. A node mechanism is added, which changes an original 16-line laser scanner to a system with hundreds of lines, which provides a resolution comparable to camera systems. As pointed out [10,11], the accuracy of the intrinsic and extrinsic parameters of the scanner in this kind of system determines its 3D measurement accuracy. The intrinsic parameter error is the deviation of the lines in the laser scanner from the ideal homocentric assumption, and the extrinsic parameter errors are generally caused by the manufacturing error in the dense scanning system. Together they could induce observable misalignments among scanning lines, which leads to shape deformation of the objects. Therefore, we designed a calibration method to diminish both intrinsic and extrinsic parameter errors all at once.

The 3D object recognition and localization methods can be categorized into three kinds based on how the point cloud data is processed. The first kind is the volumetric-based method. The space is divided into 3D grids, and usually, a 3D convolution neural network is used for shape analysis [12]. Early studies used voxel grids to store the occupancy map of the point cloud [13], and some of the later works adopted octree to save memory and computational cost [14,15]. The occupancy map failed to capture detailed local geometric features, which are replaced by a learned feature map [16]. Despite the abovementioned efforts, volumetric-based methods still suffer from their inherent confliction between high resolution and low computation efficiency.

The second kind is the point-based methods. They break the grid pattern representation and focus on feature extraction at each point. The most representative work is the Pointnet [17], which used several multilayer perceptron (MLP) layers to extract global and local features for each point. Many works have been done to further improve the PointNet on its sampling techniques and feature optimization methods [18]. Another way to address the irregular point distribution is to use the 3D convolution methods directly on points [19,20] with redefined kernels on local neighbor points. They output the classification results of each point. The calculation cost of point-based methods is much higher, especially for dense point clouds, so they are generally used on a small dataset for object classification, instead of a large dataset for instance segmentation.

The third kind is to project the point cloud into a virtual plane, which results in a 2D image. In this way, the traditional 2D image object detection and instance segmentation methods can be utilized, and the running time is much shorter than the previous two kinds. The point cloud was projected in a bird view [21] or a range view [22,23]. The distance of the object with respect to the sensor has less impact on recognition accuracy in bird view than in range view, but the latter usually is faster due to its smaller image size. The object recognition accuracy of this kind of method is limited with sparse data. Previous researchers added camera images to assist in object recognition to solve this problem [24,25]. They used a high-resolution camera image for the object detection and segmentation, then found the corresponding points in the point cloud using the camera-to-laser geometric transformation. As a result, the recognition accuracy is much improved with additional sensing resources.

Though many approaches have been developed, they mainly focused on sparse LiDAR sensing, such as on the 64-line KITTI dataset [26]. In the case of dense scanning, as we deal with in this paper, there are two aspects to be addressed compared to previous research. The first one is fast object recognition. As analyzed above, the 2D image projection method is a proper choice for dense data. With the dense scanning system, we can obtain image resolution comparable to a camera, so a similar recognition accuracy using only one sensor compared to the camera and Lidar combination can be expected. The second aspect we need to address is the complicated outlier situations in dense scanning. Previous research divided radius sections on the sparse frustum data and removed the sections which are less likely to have the target object using deep learning methods [27,28]. Outlier exclusion is already challenging in sparse data [25,29], and the problems become more severe for the dense scanning case if the same deep learning method is applied. First, the background distribution is very complicated, and full coverage of all possible situations in the training set is difficult, if not impossible. As a result, there is no guarantee of deep learning results. Second, the exclusion process can be very time-consuming. In this paper, we proposed to use a lightweight fuzzy logic neural network to assist in data truncation, which achieved fast and desirable outlier exclusion results. The refined data helped accurate 3D localization of the target, as testified in different scenes. The statistical results show that the system provides a stable and precise estimation of the heading directions and 3D locations for various test subjects. The contributions of the paper are summarized as follows:

- A low-cost dense Lidar scanning system is designed, and a new calibration approach is proposed to correct the intrinsic and external parameters of the system simultaneously.

The method ensures constraints in all 6 DOFs, which fully considers all effects in the 3D space induced by parameter drifts.

- A lightweight fuzzy network is designed to assist in outlier exclusion, achieving much faster computation speed than a deep learning network. The refined data helped accurate 3D localization of the target, as testified in various scenes.
- A new criterion is proposed to estimate the location and orientation accuracy of the detected target, which is more suitable for mobile robot applications than the rough estimation by IoU (intersection over union). Under the new criterion, the algorithms above are tested in experiments and proven to provide precise target information for mobile robot operations.

The pipeline of the paper is shown in Figure 1. The dense scanning system and its calibration method are introduced in Section 2. The image generation and corresponding 2D instance segmentation are presented in Section 3, which results in a direct reconstruction of the 3D point cloud of the object. We proposed a fuzzy neural network to remove the background outliers in Section 4, which helps generate an accurate 3D box for the object. The experimental results are presented and discussed in Section 5, and the paper concludes in Section 6.

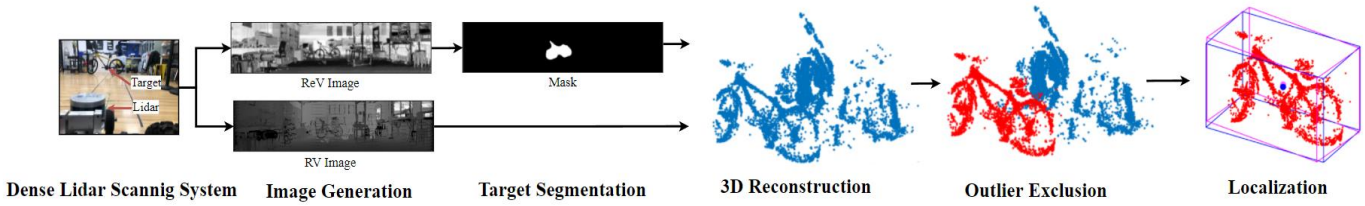
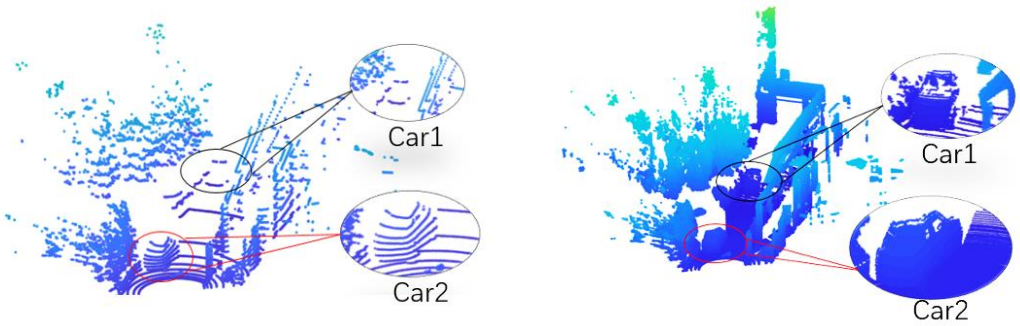


Figure 1. The pipeline of the paper.

2. Dense LiDAR Scanning System

General commercial LiDAR sensors can only obtain sparse point clouds, which leads to tremendous difficulties in identifying objects at a long distance. The custom-made dense scanning system in this paper aims to provide more abundant geometric information comparable to the resolution of an RGB camera.

Figure 2 shows the comparison of the point clouds for the same scene by the 16-line laser sensor and our dense scanning system, respectively. With dense scanning, the geometric details of distant objects can be obtained. In the following content, the dense scanning system design and calibration are introduced, which ensures sufficient and accurate data resources as the fundament for the rest work in this paper.



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

Figure 2. Comparison between point clouds with and without dense scanning system: (**a**) Original LiDAR point cloud from 16-line 3D LiDAR; (**b**) Point cloud with the dense scanning system.

2.1. System Introduction

The mechanical structure of the dense scan system is shown in Figure 3. The system consists of a laser sensor, a motion generation subsystem, a power transmission subsystem, and a support frame. The laser sensor used is a VLP-16 LiDAR with a vertical field of view of 30° and a horizontal field of view of 360°, attached to a rotational axle with a holder. The motion generation subsystem is driven by a stepping motor connected to a gearbox to generate high-precision controllable rotary motion output. Two synchronous belt systems connect the power transmission subsystem to transfer the rotation motion from the gearbox axle to the sensor bearing axle with a 1:1 ratio.

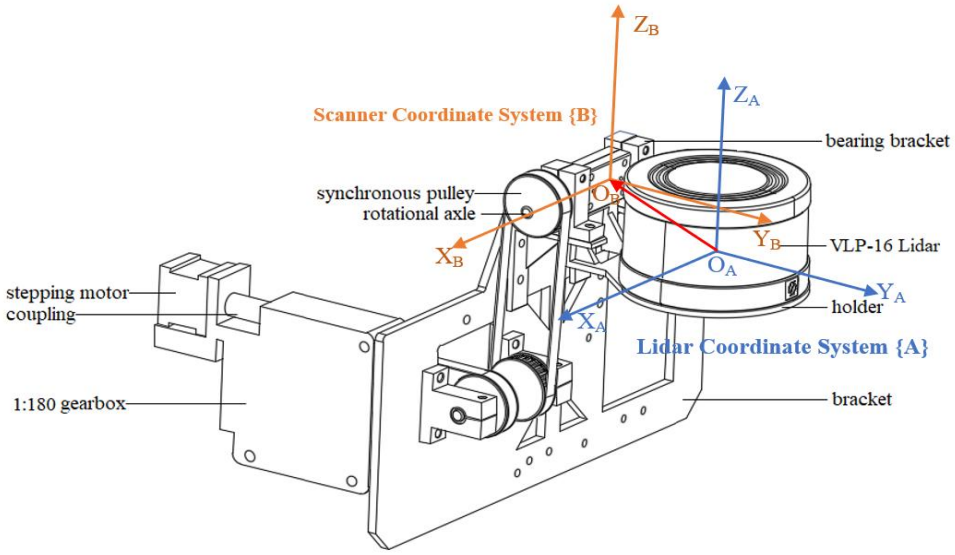


Figure 3. Mechanical design of the dense scan system.

The laser beams from the VLP-16 have a fixed gap of 2° between each other from −15° to 15° in the vertical direction. The additional head node rotation from the mechanical system helps to fill the gap with a resolution of 0.1°. The original data from the VLP-16 at each nod position needs to be transformed and combined in a static coordinate system with respect to the scanning system since the pose of the LiDAR coordinate system varies with the nod angle.

The coordinate systems are defined in Figure 3. The origin of the LiDAR coordinate system {**A**} is located at the optical center of the sensor, the *Y*-axis points forward, and the *Z*-axis points upward with respect to the sensor. The origin of the scanner coordinate system is set as the center of the rotation axle, the *X*-axis is the center axis of the axle, and the *Z*-axis is vertically upward.

The measurements of each point include horizontal angle (α), vertical angle (β), range (d), and reflectivity (ref). Through the first three values, we can get the three-dimensional coordinates of the scanned object in {$\boldsymbol{A}$}. As illustrated by Figure 4, the coordinates (x_A, y_A, z_A) of a scanned point can be calculated with Equation (1).

$$\begin{cases} x_A = d\cos\beta\cos\alpha \\ y_A = d\cos\beta\sin\alpha \\ z_A = d\sin\beta \end{cases} \tag{1}$$

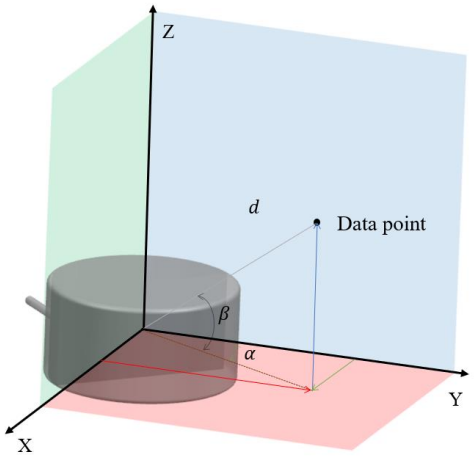


Figure 4. The point in coordinate system {**A**}.

For a point $\boldsymbol{P}$ in the space, ${}^{A}\boldsymbol{P}$, ${}^{B}\boldsymbol{P}$ are its coordinates in {$\boldsymbol{A}$} and {$\boldsymbol{B}$}, respectively. When the stepping motor rotates the laser sensor, the origin of {$\boldsymbol{A}$} rotates around the X-axis of {$\boldsymbol{B}$} by an angle of $\boldsymbol{\theta}$. The transformation of the point coordinates between {$\boldsymbol{A}$} and {$\boldsymbol{B}$} can be written as

$$ {}^{B}\boldsymbol{P} = {}^{B}\boldsymbol{R}_{x}(\boldsymbol{\theta})\left({}^{B}_{A}\boldsymbol{R}\,{}^{A}\boldsymbol{P} + {}^{B}_{A}\boldsymbol{T}\right) \tag{2} $$

where ${}^{B}_{A}\boldsymbol{R}$ and ${}^{B}_{A}\boldsymbol{T}$ are the rotation matrix and translation vector between the coordinate {$\boldsymbol{A}$} and {$\boldsymbol{B}$}, respectively, which are determined by the mechanical design of the system.

2.2. Intrinsic and External Parameters Calibration

Each beam is a ray emanating from a laser sensor in a multi-beam scanner. Ideally, they lie in a vertical plane and intersect at an origin. However, each sensor deviates from its ideal pose due to intrinsic parameter errors [10]. Such error could cause shape distortion in 3D models of the environments. An example is shown in Figure 5, where the measurement of a plane wall by the dense scan system clearly splits into discontinuous blocks. The inconsistency in surface modeling is mainly caused by inaccurate alignments of the point clouds from different beams. The shape deviation could be more damaging due to the destruction of their surface shapes. In addition, the extrinsic parameter error, which is due to the manufacturing error by our customized nodding mechanism, causes extra location and orientation displacements of the measured points [11].

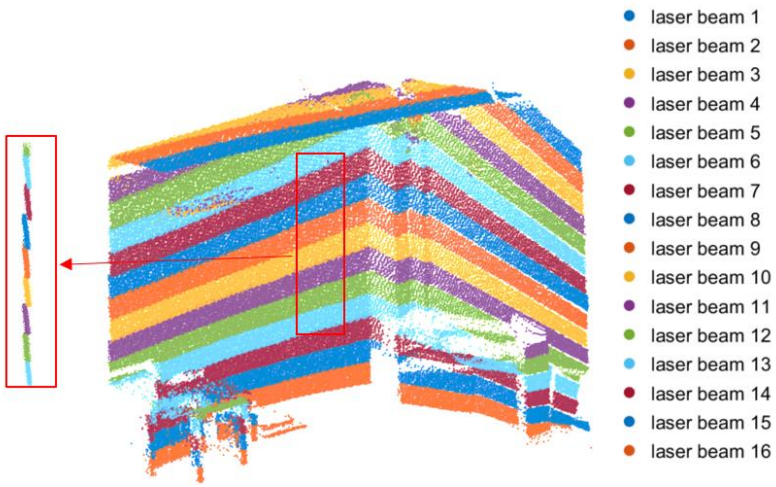


Figure 5. Uncalibrated dense point cloud.

To diminish the effect of such parameter inaccuracy, we propose a full-constrained calibration method. When parameter errors exist, the alignment of the coordinate systems for different beams deviate from their theoretical conditions. A general description of such deviation is modeled as additional rotational and translational matrices between each other. Choose the coordinate system of the first laser beam as a reference; the coordinate system deviations of other beams are defined as ${}^{i}\boldsymbol{R}_{err}$ and ${}^{i}\boldsymbol{T}_{err}$, where $i = 1, 2, \ldots, 15$ is the index difference of the beam with respect to the reference. Correspondingly, the relationship between ${}^{A}\boldsymbol{P}$ and ${}^{B}\boldsymbol{P}$ in Equation (2) is modified as Equation (3).

$$ {}^{B}\boldsymbol{P}_i = {}^{B}\boldsymbol{R}_x \left[{}^{B}_{A}\boldsymbol{R} \left({}^{i}\boldsymbol{R}_{err} {}^{A}\boldsymbol{P}_i + {}^{i}\boldsymbol{R}_{err} \right) {}^{B}_{A}\boldsymbol{T} \right] \tag{3} $$

The ${}^{i}\boldsymbol{R}_{err}$ and ${}^{i}\boldsymbol{T}_{err}$ in 3D space have 6 error parameters corresponding to the rotation and translation on the X, Y, and Z axis, respectively, as shown in Equations (4) and (5).

$$ {}^{i}\boldsymbol{R}_{err} = \boldsymbol{R}_z(\gamma_i)\boldsymbol{R}_y(\beta_i)\boldsymbol{R}_x(\alpha_i) = \begin{bmatrix} \cos\alpha_i \cos\gamma_i - \cos\beta_i \sin\alpha_i \sin\gamma_i & -\cos\beta_i \cos\gamma_i \sin\alpha_i - \cos\alpha_i \sin\gamma_i & \sin\alpha_i \sin\beta_i \\ \cos\gamma_i \sin\alpha_i + \cos\alpha_i \cos\beta_i \sin\gamma_i & \cos\alpha_i \cos\beta_i\gamma_i - \sin\alpha_i \sin\gamma_i & -\cos\alpha_i \sin\beta_i \\ \sin\beta_i \sin\gamma_i & \cos\gamma_i \sin\beta_i & \cos\beta_i \end{bmatrix} \tag{4} $$

$$ {}^{i}\boldsymbol{T}_{err} = \begin{bmatrix} T_{x_i} & T_{y_i} & T_{z_i} \end{bmatrix} \tag{5} $$

where $\alpha_i, \beta_i, \gamma_i, T_{x_i}, T_{y_i}, T_{z_i}$ are rational angle error parameters and translation error parameters of X axis, Y axis, and Z axis, respectively.

Calibration needs to constrain 6 degrees of freedom in the 3D space. To fully consider the 6 DOF constraints, the calibration scene is designed as in Figure 6. The scanner is placed in front of a U-shape wall, which provides three planes as the calibration references. There are three kinds of constraints to be considered: (1) The flatness of the point clouds for all three planes, (2) the continuity of the blocks in plane 1 generated by neighbor beams, and (3) the asymmetry of the relative location relationship between the blocks on plane 2 and 3. The corresponding errors are denoted as ${}^{W_j}err_{P_i}$, ${}^{W_j}err_{C_i}$, ${}^{W_{23}}err_{S_i}$, respectively, where $j = 1, 2, 3$ is the plane index. Inherently, the three kinds of constraints limit the 6 DOF motion of one coordinate system to another, as listed in Table 1. In the following content, we present the error cost formulation for each constraint.

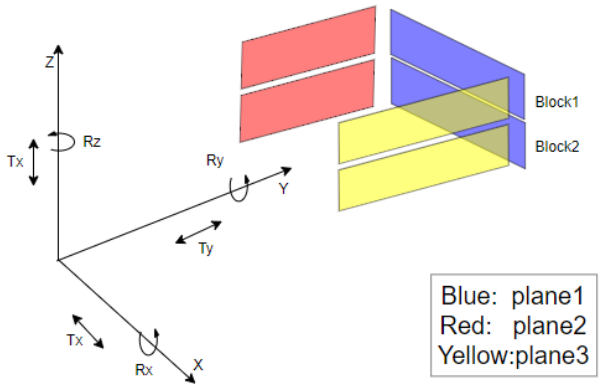


Figure 6. The motion restrictions between beams in the calibration experimental design.

Table 1. Motion restrictions by the calibration constraints.

Constraint	Restricted DOFs
${}^{W_1}err_{P_i}$	T_y & R_z
${}^{W_2}err_{P_i}$	T_x & R_z
${}^{W_3}err_{P_i}$	T_x & R_z
${}^{W_1}err_{C_i}$	T_z & R_z
${}^{W_{23}}err_{S_i}$	R_y

The cost function for the calibration is defined in Equation (6). Since the calibration parameters have the same effect on each plane, the constraint weights of the planes are equal. However, there are two constraints related to plane 1: flatness constraint ${}^{W_1}err_{\boldsymbol{P}_i}$ and continuity constraint ${}^{W_1}err_{\boldsymbol{C}_i}$. We define a weight k to assign the emphasis between the two. In the following content, we present the error cost formulation for each constraint.

$$err_t(\alpha_i, \beta_i, \gamma_i, T_{x_i}, T_{y_i}, T_{z_i}) = k\,{}^{W_1}err_{\boldsymbol{C}_i} + (1-k)\,{}^{W_1}err_{\boldsymbol{P}_i} + {}^{W_2}err_{\boldsymbol{P}_i} + {}^{W_3}err_{\boldsymbol{P}_i} + {}^{W_{23}}err_{\boldsymbol{S}_i} \quad (6)$$

For the constraints of plane flatness, the point clouds belonging to the same plane are fitted with a function of a plane, and the fitting error is used as the calibration cost. We use the principal component analysis (PCA) [30] method to realize the plane fitness since there are no significant outliers in the calibration data and PCA is proven to be robust to measurement noise. The detailed procedure to obtain the plane cost ${}^{W_j}err_{P_i}$ is presented below.

1. Input the plane point cloud set ${}^{W_j}\boldsymbol{P}_i$, where ${}^{W_j}\boldsymbol{P}_i$ is the combination of the point clouds by the first and the $(i+1)^{th}$ beam, and $n_{\boldsymbol{P}_i}$ is the number of elements in ${}^{W_j}\boldsymbol{P}_i$;
2. Define the fitted plane function as Equation (7), where $\vec{\boldsymbol{n}}$ is the normal vector of the plane, and ${}^{W_j}\boldsymbol{P}_0 = (x_0, y_0, z_0)$ is the center coordinates of ${}^{W_j}\boldsymbol{P}_i$;

$$\vec{\boldsymbol{n}} \begin{pmatrix} x \\ y \\ z \end{pmatrix} - \vec{\boldsymbol{n}}\,{}^{W_j}\boldsymbol{P}_0 = 0 \quad (7)$$

3. Define the mean distance of the points to the fitted plane as the fitting error ${}^{W_j}err_{\boldsymbol{P}_i}$ in Equation (8), which is a function of $\vec{\boldsymbol{n}}$, where (x_i, y_i, z_i) are the coordinates of each point in ${}^{W_j}\boldsymbol{P}_i$;

$${}^{W_j}err_{\boldsymbol{P}_i}\left(\vec{\boldsymbol{n}}\right) = \frac{1}{n_{\boldsymbol{P}_i}} \sum_{i=1}^{n_{\boldsymbol{P}_i}} \left| \frac{a(x_i - x_0) + b(y_i - y_0) + c(z_i - z_0)}{\left|\vec{\boldsymbol{n}}\right|} \right| \quad (8)$$

4. Using the PCA [30] method to estimate an initial normal vector $\vec{\boldsymbol{n}}_0$. Define a covariance matrix ${}^{P_i}\boldsymbol{C}_{W_j}$ in Equation (9) and $\vec{\boldsymbol{n}}_0$ is the minimal eigenvalue of ${}^{P_i}\boldsymbol{C}_{W_j}$;

$${}^{P_i}\boldsymbol{C}_{W_j} = \frac{1}{n_{\boldsymbol{P}_i}} \sum_{i=1}^{n_{\boldsymbol{P}_i}} (\boldsymbol{P}_i - \boldsymbol{P}_0)(\boldsymbol{P}_i - \boldsymbol{P}_0)^T \quad (9)$$

5. Minimize ${}^{W_j}err_{\boldsymbol{P}_i}$ in Equation (8) using OQNLP iterative algorithm [31] with the initial value of $\vec{\boldsymbol{n}}$ as $\vec{\boldsymbol{n}}_0$. The resulting optimal normal vector is denoted as $\vec{\boldsymbol{n}}^*$. The final fitted plane and the corresponding flatness error ${}^{W_j}err_{\boldsymbol{P}_i}$ are obtained by Equation (7) with $\vec{\boldsymbol{n}} = \vec{\boldsymbol{n}}^*$.

For the constraint of plane continuity in plane 1, we use curve distance to quantify the splits between two adjacent beam blocks. The scanner rotates downward 20 times with a resolution of 0.1°, which provides a total of 21 scan curves for each beam. The cone of each beam intersects with the three walls with a parabolic curve, as shown in Figure 7. Since the gap between the original beams from the VLP-16 is 2°, the very first curve of the lower beam block should overlap with the last one of the upper beam blocks in the front plane (plane 1), as shown in Figure 8.

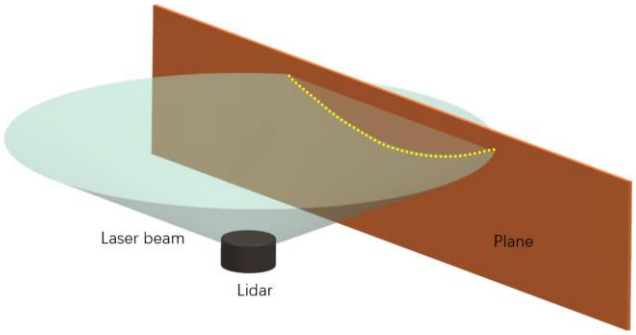


Figure 7. The measured curve on a vertical plane by a single laser beam.

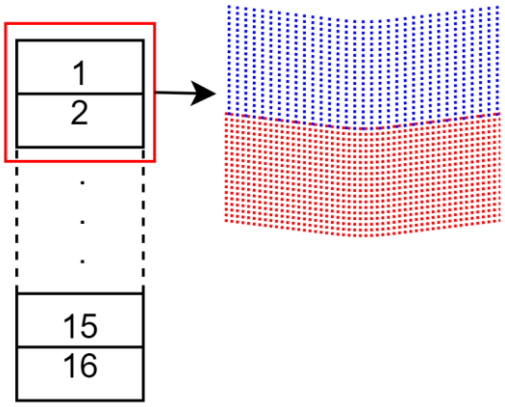


Figure 8. Point cloud blocks of laser beams on the front plane 1.

The detailed procedure to obtain the continuity cost ${}^{W_1}err_{C_i}$ is presented below.

1. Input the point set $\boldsymbol{C}_i$, which is the last scanned line of the i^{th} block, and $m_{\boldsymbol{C}_i}$ is the number of elements in $\boldsymbol{C}_i$;
2. Define the fitted curve function in the YZ plane as Equation (10), where $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are the coordinates matrix in YZ plane. For the i^{th} block, $i = 1, 2, \ldots, 15$, its last beam curve can be fitted with a quadratic curve in the YZ plane, as in Equation (11).

$$\boldsymbol{Y}\begin{bmatrix} t & r & s \end{bmatrix}^{\boldsymbol{T}} - \boldsymbol{Z} = 0 \tag{10}$$

$$\boldsymbol{Y} = \begin{bmatrix} 1 & y_1 & {y_1}^2 \\ 1 & y_1 & {y_1}^2 \\ & \cdots & \\ 1 & y_{m_{C_i}} & {y_{m_{C_i}}}^2 \end{bmatrix}, \boldsymbol{Z} = \begin{bmatrix} z_1 \\ z_2 \\ \cdots \\ z_{m_{C_i}} \end{bmatrix} \tag{11}$$

3. Use the least square method with Equation (12) to calculate the coefficient $\begin{bmatrix} t & r & s \end{bmatrix}^{\boldsymbol{T}}$.

$$\begin{bmatrix} t & r & s \end{bmatrix}^{\boldsymbol{T}} = \left(\boldsymbol{Y}^{\boldsymbol{T}}\boldsymbol{Y}\right)^{-1}\boldsymbol{Y}^{\boldsymbol{T}}\boldsymbol{Z} \tag{12}$$

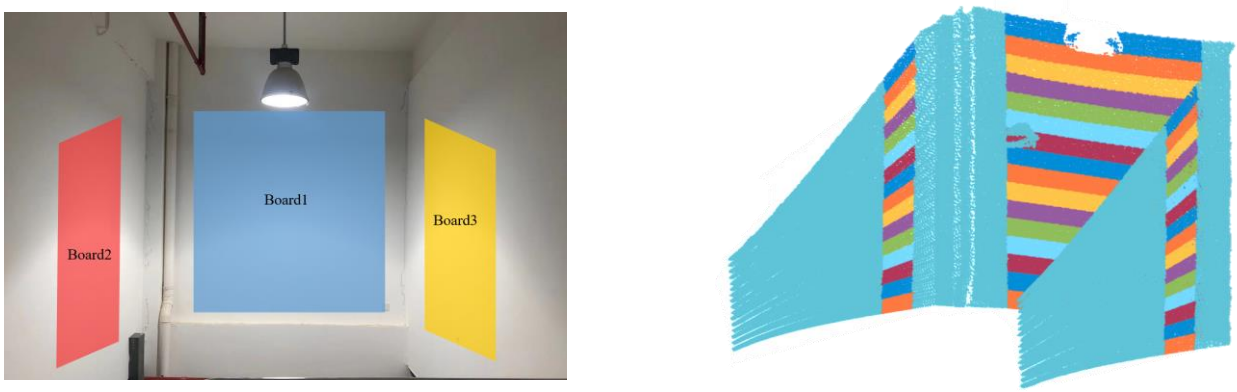
4. Input the point set $\boldsymbol{C}_{i+1}$, where $\boldsymbol{C}_{i+1}$ is the first scanned line of the $(i+1)^{th}$ block, and $m_{\boldsymbol{C}_{i+1}}$ is the number of elements in $\boldsymbol{C}_{i+1}$;
5. The error of distance constraint is the average distance of points in $\boldsymbol{C}_{i+1}$ to the fitted curve in Equation (10). The corresponding continuity cost ${}^{W_1}err_{\boldsymbol{C}_i}$ are obtained by Equation (13), where (y_i, z_i) are the coordinates of each point in YZ plane in $\boldsymbol{C}_{i+1}$.

$${}^{W_1}err_{\boldsymbol{C}_i} = \frac{1}{n_{\boldsymbol{C}_{i+1}}}\sum_{i=1}^{n_{\boldsymbol{C}_{i+1}}}|d_i| = \frac{1}{n_{\boldsymbol{C}_{i+1}}}\sum_{i=1}^{n_{\boldsymbol{C}_{i+1}}}\left|z_i - \left(t + sy_i + r{y_i}^2\right)\right| \tag{13}$$

For the asymmetry constraint, we quantify the error by examining the difference of the gap widths of adjacent blocks between the left and right walls. Using the same procedure of calculating ${}^{W_1}err_{C_i}$, we can obtain the gap distance between any two blocks on planes 2&3, denoted as ${}^{W_2}err_{C_i}$ and ${}^{W_3}err_{C_i}$. Then the symmetry error ${}^{W_{23}}err_{S_i}$ is defined as

$$ {}^{W_{23}}err_{S_i} = \left| {}^{W_1}err_{C_i} - {}^{W_1}err_{C_i} \right| \tag{14}$$

The experimental scene is shown in Figure 9a, which is a corner of a stairwell. The data generated for calibration is shown in Figure 9b, which provides the U-shape geometry we need. Minimize $err_t\left(\alpha_i, \beta_i, \gamma_i, T_{x_i}, T_{y_i}, T_{z_i}\right)$ in Equation (14) using OQNLP iterative algorithm [31], with the initial value set as zero matrices. The resulting optimal calibration parameters results are shown in Table 2. Both the rotational and translational corrections of the beam coordinate systems are minor, which is reasonable considering the small dimensions of the light components in the scanner, the improvement for the environmental modeling accuracy is evident.



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

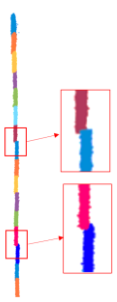
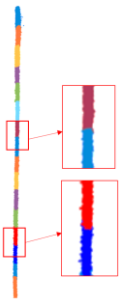
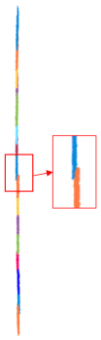
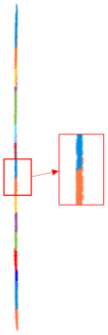
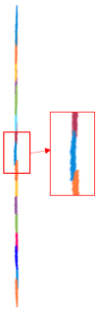
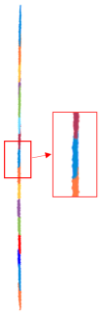
Figure 9. U-shape Experiment wall scene: (**a**) Experimental scene. (**b**) Point clouds for calibration.

Table 2. Parameter Calibration Results (k = 0.2 in Equation (6)).

Laser ID	α_i/rad	β_i/rad	γ_i/rad	T_{x_i}/mm	T_{y_i}/mm	T_{z_i}/mm
1	0.000377	2.26×10^{-5}	0.000659	−0.89445	−0.4671	−0.36628
2	0.004092	−0.0003	0.000416	−5.44071	−1.3504	−2.61062
3	0.005324	−0.00018	0.000924	−1.1268	−2.5268	−1.45165
4	0.004636	0.007249	0.002648	−2.3791	−1.0444	3.57043
5	0.005254	−0.00028	0.003374	−1.9334	−2.137	1.179039
6	0.005564	−0.00108	0.004352	−1.53332	−2.3258	−1.18372
7	0.004925	0.005848	0.004374	−2.357	−1.0122	3.49606
8	0.003797	−0.00323	0.037907	−1.7302	−1.6763	5.236154
9	0.007164	0.005035	0.038873	−1.8049	−3.9924	4.50977
10	0.010316	0.007089	0.039402	−1.9303	−5.9039	0.26249
11	0.011841	0.012014	0.042606	−1.75734	−2.2206	8.65139
12	0.006134	0.020335	0.044059	−1.53601	−2.7426	1.8605
13	0.01163	0.006123	0.049215	−1.6067	−3.0979	3.04924
14	−0.00213	−0.00772	0.049009	−2.104	−4.8593	4.25423
15	0.005891	−0.01073	0.049871	−2.317	−4.3012	4.80066

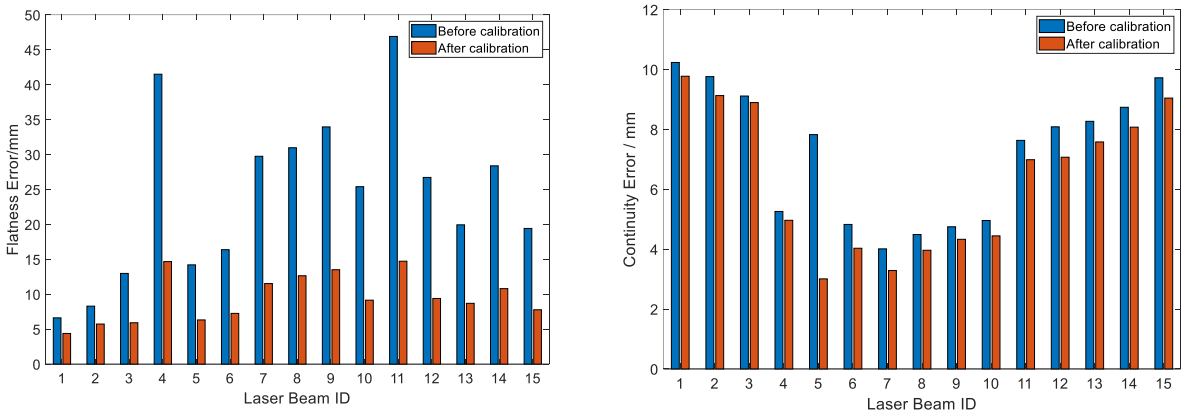
Table 3 shows the comparisons of the three planes before and after calibration from two viewpoints. The flatness is much improved by the side views, and there are no overlaps or unreasonable gaps between any two adjacent point cloud blocks. At the same time, the calibration does not cause any unreasonable changes to plane shape by inspecting the front views. The fitted curve distance constraints ensure the continuity between the blocks.

Table 3. Comparison of the point clouds before and after the calibration.

Plane ID	Front View		Side View	
	Before	After	Before	After
1				
2				
3				

From the side views of the front plane, the inaccurate system parameters caused more serious problems on plane flatness than continuity. Therefore, we set the value of k in Equation (6) as $k = 0.2$ to emphasize more on flatness cost. The calibration results for the two costs are shown in Figure 10. The large flatness errors have been vastly decreased, and the continuity also improves. The calibration goal is fully accomplished. The readers can tune the weight k according to their scanning system status.

In this section, we proposed a 3D Lidar calibration method using three U-shape planes as reference. With all 6 DOF constraints restrained in the cost function, the internal parameters and external parameters are combined in the model to be calibrated. It takes 30 min to optimize with 64,164 points in the data, which is acceptable because it is only done once offline.



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

Figure 10. Errors changes of plane 1 before and after calibration: (**a**) Flatness error change of each point cloud block. (**b**) Continuity error change of each point cloud block.

3. 2D Segmentation with Lidar Images

The general pipeline of locating objects in 3D point clouds includes the tasks of recognition, segmentation, and 3D boxing. As discussed above, it is convenient in our case to accomplish object recognition and rough segmentation with 2D projected images.

3.1. Images Generation

The beams of the scanning system rotate in two directions, as illustrated in Section 2. Therefore, the horizontal and vertical angles can serve as XY indexes to construct a projected image, as shown in Figure 11. The pixel indexes of one point are calculated by discrete angle index in the two directions:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{1}{d\alpha} & 0 \\ 0 & \frac{1}{d\beta} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{15}$$

where u and v are the horizontal and vertical pixel index, and $d\alpha$ and $d\beta$ are the angle resolution in the two directions, respectively. $d\alpha = d\beta = 10^{\circ}$ in our system.

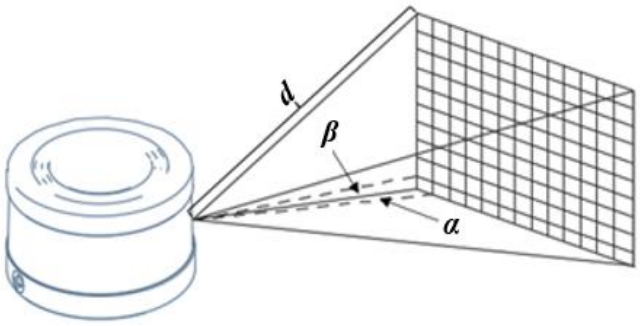


Figure 11. Schematic Diagram of the Projection Method.

A 1200 × 320 image can be obtained with the data in the front view of 120°. The intensity of the pixels can be obtained by either the depth or the reflectivity data, which are named RV Image and ReV Image, respectively. For the RV image, the intensity of a certain pixel is calculated with Equation (16). We choose to emphasize information closer to the LiDAR so that most of the image has sufficient feature details.

$${}^{RV}\boldsymbol{I}_{uv} = \begin{cases} \lfloor \frac{255}{k} d_{uv} \rfloor & d_{uv} < k_d \\ 255 & d_{uv} > k_d \end{cases} \qquad d_{uv} \in [0, r_{max}], k_d < r_{max} \tag{16}$$

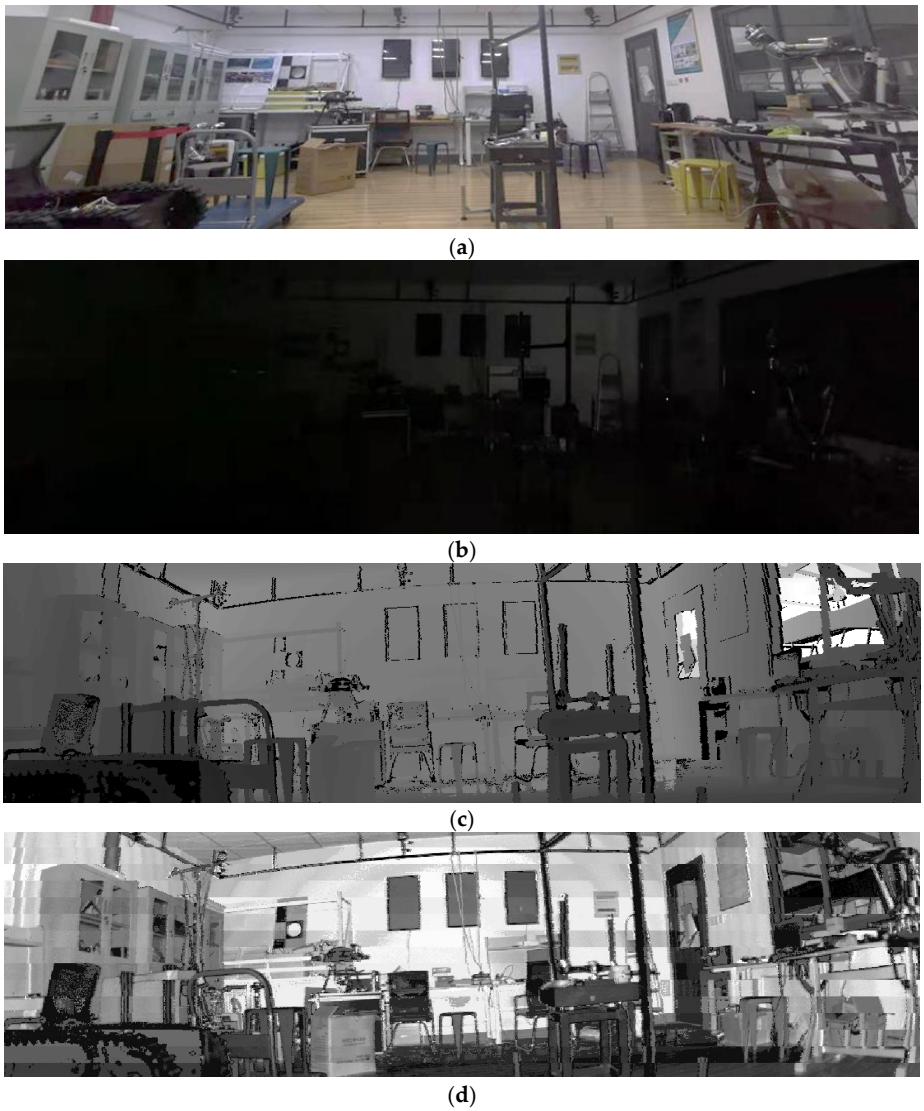
where r_{max} is the maximum measurable range of the current LiDAR, d_{uv} is the measured range for the current pixel, k_d is a user-specified parameter to define the white color threshold, and ${}^{RV}\boldsymbol{I}_{uv}$ is the resulting pixel intensity for the RV image.

For the ReV image, the measured reflectivity is regulated by the cumulative distribution function (CDF). The projected intensity value is then calculated by Equation (17).

$$ {}^{ReV}\boldsymbol{I}_{uv} = \left\lfloor \frac{\mathrm{CDF}(ref_{uv}) - cdf_{min}}{cdf_{max} - cdf_{min}} \times 255 \right\rfloor \tag{17} $$

where cdf_{max} and cdf_{min} are the maximum and minimum values of the CDF for all the points, respectively, and ${}^{ReV}\boldsymbol{I}_{uv}$ is the resulting pixel intensity for the ReV image.

The RV Image and ReV Image are compared with the camera images for the same scene in Figure 12. When the illumination condition deteriorates, the camera is hard to capture most of the details. In contrast, the weakened lighting causes almost no effect on the LiDAR images. The ReV image obviously can obtain more subtle texture information than the RV image, so we choose it as the object identification resource in the following work.



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

Figure 12. Camera and LiDAR image comparison in an indoor scene: (**a**) the image by a camera under normal lighting conditions; (**b**) the image by a camera under very weak lighting conditions; (**c**) the RV Image by our dense scanning system under very weak lighting conditions; (**d**) the ReV. Image by our dense scanning system under very weak lighting conditions.

3.2. Convolution Network on Instance Segmentation

2D segmentation has been a mature technique in the computer vision field. In this paper, we adopt the well-known Mask-RCNN [32] for this purpose. The dataset used in this paper is pre-trained on the COCO dataset [33]. The reader can choose other segmentation tools, which do not affect the general performance of the system.

However, there are still significant differences between a camera image and a Rev image, as illustrated in Figure 12. Therefore, we need to do transfer learning for the pre-trained network using a small dataset of Rev images. In order to ensure the reliability of the training results, we use data enhancement methods to expand the data, including adding gaussian noise, rotating a certain angle, randomly erasing part of the area, randomly clipping part of the area and horizontal flipping. The transfer learning used the model framework of Mask-RCNN [32] with pre-trained parameters on the COCO dataset [33] as the learning start point. The effect of transfer learning on bike recognition is shown in Figure 13, where the false positives are eliminated after re-training the network. It is worth mentioning that the details of the bike are observably recovered in the images, which benefit from the calibration technique introduced in Section 2. It largely improves the recognition possibility of the objects, especially for those in relatively far distances.

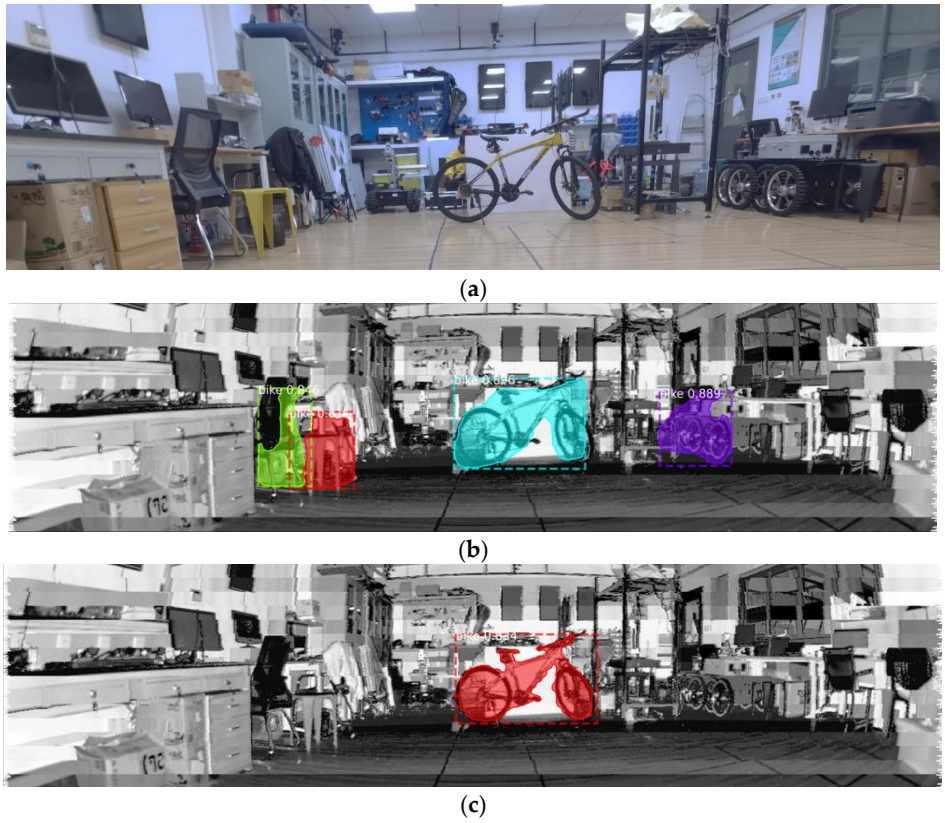
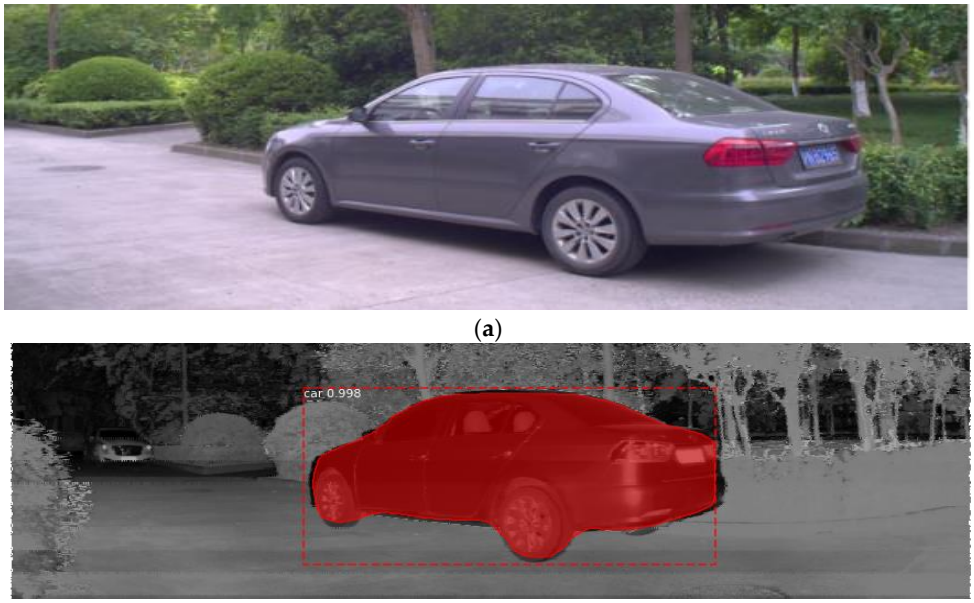


Figure 13. Bike segmentations using the Mask-RCNN before and after the transfer learning in the indoor scene: (**a**) the RGB image in the indoor scene; (**b**) segmentation of bike before transfer learning; (**c**) segmentation of bike after transfer learning.

Similar results can be obtained in the outdoor scene, such as the car segmentation in Figure 14. Comparing the segmentation effects of the bike and the car, the object with frame structure, such as the bike, inevitably includes more background pixels than the car. This could lead to more difficulties in the following work of 3D localization.



(**a**)

(**b**)

Figure 14. Segmentation result in outdoor experiment: (**a**) the RGB image in the outdoor scene; (**b**) segmentation for the ReV image of a car in the outdoor scene.

4. ANFIS-Aided 3D Refinement and Localization

The 2D segmentation extracts the out-contour of the objects, and the encircled area can be reprojected back to the 3D space to get the corresponding 3D point cloud. As it is shown in Figure 15, the noises due to 2D segmentation inaccuracy are limited for objects with minor holes, such as cars. However, for objects with many hollow areas, such as bikes, the outliers can even outsize the object itself in some challenging scenarios. We call such objects framed objects in the following content.

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

Figure 15. Point clouds reconstruction under the Mask-RCNN masking: (**a**) 3D point clouds of the 2D segmented bike; (**b**) 3D point clouds of the 2D segmented car.

In the following content, a fuzzy logic network is used to further remove most of the outliers in the background by adaptively choosing a truncation position in the Y direction. As a result, lightweight networks can be directly used to estimate the location of the 3D object. The outlier removal is equivalent to a Y axis truncation based on the assumptions below:

1. There is a certain distance between the target object and the background on the Y axis;
2. There are no foreground noises.

4.1. ANFIS Construction

4.1.1. ANFIS Structure Design

To find the truncating Y value Tr_y for given point clouds of the target objects, we first discretize the Y axis into multiple sections. The ANFIS outputs the likelihood of each section to be the truncating position. The inputs of the network are chosen as statistically influencing factors: the discretized section index y_{os}, the number of points n_{os} in the sections of y_{os}, the difference of n_{os} along Y axis denoted as dn_{os}, and the mean value of the Z coordinates of the points in each section is denoted as mz_{os}. The four inputs reflect the properties of the point set in different aspects: y_{os} gives the depth range of the point sets, n_{os} reflects the density distribution of the points in depth direction, dn_{os} represents the change of the distribution, and mz_{os} reflects the height distribution. For the q^{th} section, the combination of the four $\left(y_{os\_q}\ n_{os\_q}\ dn_{os\_q}\ mz_{os\_q}\right)^T$ constructs an input vector to the ANFIS. With all the considerations above, the fuzzy logic network is able to output a quantified value Q_q indicating how likely the q^{th} section should be the cutting-off place. With the comparison of all the Q_q, we can deduce a most possible section q corresponding to a truncating Y value Tr_y which is expected to remove most of the background outliers.

The framework of the ANFIS system is shown in Figure 16. The system consists of four modules: fuzzification with membership function, rule-based inference, normalization, and inference result, the details are presented as below.

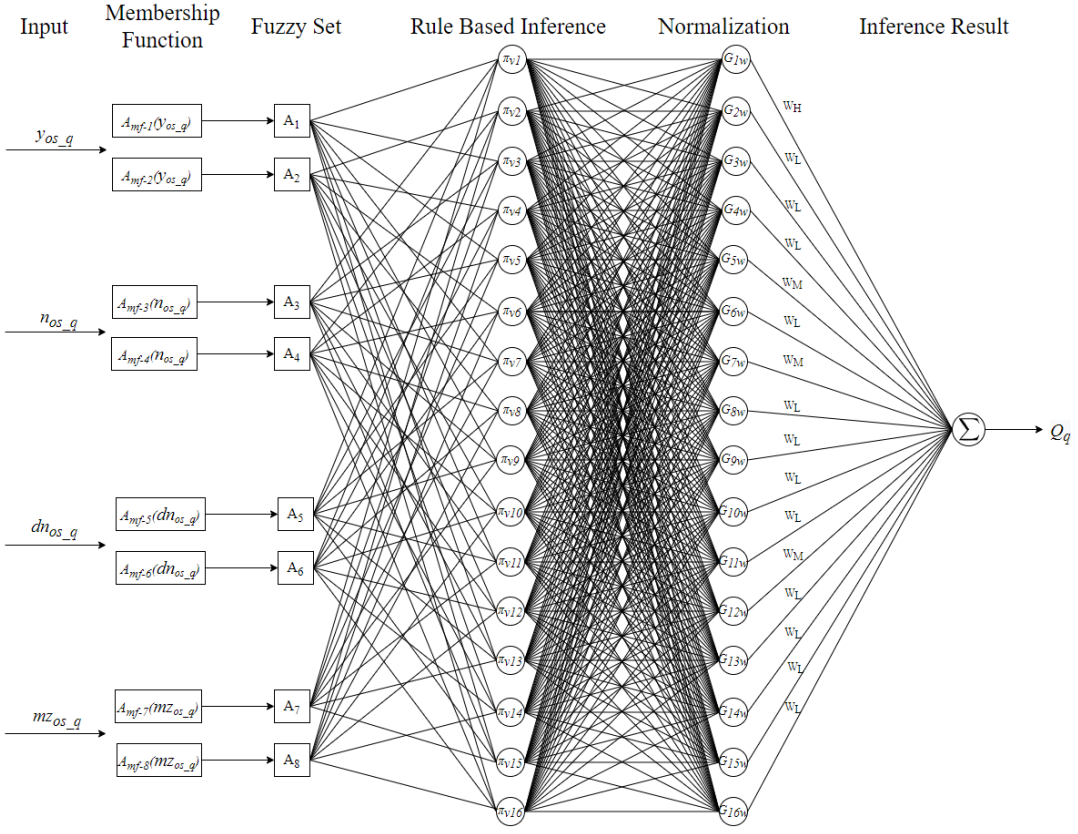


Figure 16. The framework of the ANFIS.

4.1.2. Membership Function for the Fuzzy Set

Membership functions in ANFIS fuzzify the inputs, and the fuzzy set is denoted as $\boldsymbol{A}$. We define two members in $\boldsymbol{A}$ for y_{os_q}, which are *close to* the Lidar (A_1), *far away* from the Lidar (A_2). Define two members for n_{os_q}: *small* (A_3) and *large* (A_4), which correspond to small and large numbers of points in every discretized section in the scene, respectively. Define two members for dn_{os_q}: *small* (A_5) and *large* (A_6), which correspond to small and

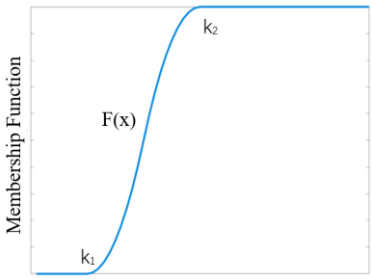
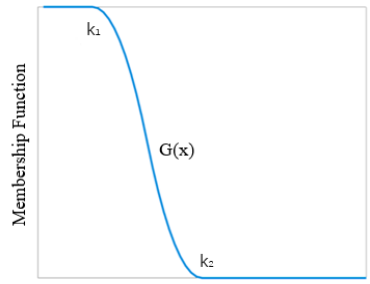
large magnitudes of the number changes of points in every discretized section, respectively. Define two members for mz_{os_q}: *small* (A_7) and *large* (A_8), which correspond to the small and large height of the points in Z direction, respectively. The relationship between the fuzzy set and corresponding point situations is shown in Table 4.

Table 4. The relationship between the fuzzy sets and situations.

Fuzzy Set	Variable	Situation
A_1	y_{os_q}(large)	Far away from the Lidar
A_2	y_{os_q}(small)	Close to the Lidar
A_3	n_{os_q}(small)	Small numbers of points
A_4	n_{os_q}(large)	Large numbers of points
A_5	dn_{os_q}(small)	Small changes of the point number
A_6	dn_{os_q}(large)	Large changes of the point number
A_7	mz_{os_q}(small)	Small average height of the points
A_8	mz_{os_q}(large)	Large average height of the points

The membership functions are designed considering the smooth transitions between the members in the fuzzy sets. For a pair of fuzzy set members, the membership functions are defined as in Table 5, with two tunable parameters to shift the curve shapes. The membership functions define the relationship between the inputs and their corresponding fuzzy sets. Design a membership function for each member A_i in A, and denote it as $A_{mf\_i}(w) \in [0,1]$. The closer $A_{mf\_i}(w)$ is to 1, the more possible that w belongs to A_i; likewise, the closer is to 0, the more impossible that w belongs to A_i.

Table 5. The general membership functions of the ANFIS.

General Membership Function	Diagram
$F(x) = \begin{cases} 0 & x < k_1 \\ 2\left(\frac{x-k_1}{k_2-k_1}\right)^2 & k_1 \le x \le \frac{k_1+k_2}{2} \\ 1-2\left(\frac{x-k_1}{k_2-k_1}\right)^2 & \frac{k_1+k_2}{2} < x < k_2 \\ 1 & x > k_2 \end{cases}$	
$G(x) = \begin{cases} 0 & x < k_1 \\ 1-2\left(\frac{x-k_1}{k_2-k_1}\right)^2 & k_1 \le x \le \frac{k_1+k_2}{2} \\ 2\left(\frac{x-k_1}{k_2-k_1}\right)^2 & \frac{k_1+k_2}{2} < x < k_2 \\ 1 & x > k_2 \end{cases}$	

For a pair of membership functions of a single input variable, their function formulas and diagrams are shown in Table 5. There are two variables to be defined in the functions, which can be tuned to shift the curve shapes better fitting the data.

Since each input has two variables in its membership functions, four inputs totally induce eight adaptive variables, denoted as A_{wi} , $i = 1, 2, \ldots, 8$. The list of the variables and their relationship with the fuzzy set and the corresponding membership functions are shown in Table 6.

Table 6. The relationship between the fuzzy sets and membership function.

Fuzzy Set	Membership Function	Function Formula	Variable
A_1 A_2	$A_{mf\_1}(y_{os\_q})$ $A_{mf\_2}(y_{os\_q})$	$F(y_{os\_q})$ $G(y_{os\_q})$	$k_1 = A_{w1}$, $k_2 = A_{w2}$
A_3 A_4	$A_{mf\_3}(n_{os\_q})$ $A_{mf\_4}(n_{os\_q})$	$G(n_{os\_q})$ $F(n_{os\_q})$	$k_1 = A_{w3}$, $k_2 = A_{w4}$
A_5 A_6	$A_{mf\_5}(dn_{os\_q})$ $A_{mf\_6}(dn_{os\_q})$	$G(dn_{os\_q})$ $F(dn_{os\_q})$	$k_1 = A_{w5}$, $k_2 = A_{w6}$
A_7 A_8	$A_{mf\_7}(mz_{os\_q})$ $A_{mf\_8}(mz_{os\_q})$	$F(mz_{os\_q})$ $G(mz_{os\_q})$	$k_1 = A_{w7}$, $k_2 = A_{w8}$

4.1.3. Fuzzy Rule Base

A fuzzy rule base defines the logical relationship between the fuzzy set and the output. Particularly in this paper, it reasons the possibility Q_q of the q^{th} section to exclude most of the outlier points according to different conditions of the inputs. The rule base we establish in this paper and the corresponding measurement weights of each rule are listed in Table 7. We use three variables W_H, W_M, and W_L to specify three degrees of possibility for a rule to be a truncating situation. For example, when the fuzzy set of the input $\left(y_{os\_q}\ n_{os\_q}\ dn_{os\_q}\ mz_{os\_q}\right)^T$ are π_{w1}(A_1, A_3, A_5, A_7), the possibility for the q^{th} section to be a truncating position is high as W_H. The physical interpretation is: the distance of the section is far away enough from the object frontside; the point number is small, so it should not be on the object; the change of the point number with its previous neighbor is small, so it is in a stable range; the point height is low, so it is possible to be the ground.

Table 7. The rule base of the ANFIS.

y_{os_q}	n_{os_q}	dn_{os_q}	mz_{os_q}	Rule Base	Weight
A_1	A_3	A_5	A_7	π_{w1}	W_H
A_1	A_3	A_6	A_7	π_{w2}	W_L
A_1	A_4	A_6	A_7	π_{w3}	W_L
A_1	A_4	A_5	A_7	π_{w4}	W_L
A_1	A_3	A_5	A_8	π_{w5}	W_M
A_1	A_3	A_6	A_8	π_{w6}	W_L
A_1	A_4	A_5	A_8	π_{w7}	W_M
A_1	A_4	A_6	A_8	π_{w8}	W_L
A_2	A_3	A_5	A_7	π_{w9}	W_L
A_2	A_3	A_6	A_7	π_{w10}	W_L
A_2	A_4	A_6	A_7	π_{w11}	W_L
A_2	A_4	A_5	A_7	π_{w12}	W_M
A_2	A_3	A_5	A_8	π_{w13}	W_L
A_2	A_3	A_6	A_8	π_{w14}	W_L
A_2	A_4	A_5	A_8	π_{w15}	W_L
A_2	A_4	A_6	A_8	π_{w16}	W_L

The degree of the membership for the rule π_{w1} is denoted as G_{1w} and defined by the minimum of the four membership outputs, as shown in Equation (18). Following the same rule, we can obtain all the G_{jw},$j = 1, 2, \ldots, 16$.

$$G_{1w} = \min\left(A_{mf\_1}\left(y_{os\_q}\right), A_{mf\_3}\left(n_{os\_q}\right), A_{mf\_5}\left(dn_{os\_q}\right), A_{mf\_7}\left(mz_{os\_q}\right)\right) \tag{18}$$

4.1.4. Normalization and Inference Output

The normalized degree of membership for all the rules is calculated by Equation (19).

$$\overline{G}_{jw} = \frac{G_{jw}}{\sum_{j=1}^{16} G_{jw}} \tag{19}$$

The inferred truncation possibility for a section is calculated by the combination of all the rule conditions with their associated weights, as shown in Equation (20). Repeating the process for all the sections, the ANFIS provides a vector $\boldsymbol{Q} = \begin{bmatrix} Q_1 & \dots & Q_q & \dots & Q_n \end{bmatrix}$ including the possibility for all the sections to be the truncating place.

$$Q_q = \sum_{j=1}^{16} W_j \overline{G}_{jw} \tag{20}$$

It is straightforward to find Tr_y with a given $\boldsymbol{Q}$, which is simply the Y coordinate value of the section with the maximum value in $\boldsymbol{Q}$.

4.1.5. Parameters Training

There are 11 adaptive parameters $X_{ANFIS} = \begin{bmatrix} A_{wi} & W_H & W_M & W_L \end{bmatrix}^T$, $i = 1, 2, \dots, 8$, in the ANFIS above, including eight membership function variables and three weight parameters. They can be selected by experience but may not be optimal. The following describes how to use experimental data for parameter training.

The ground truth of the truncating value Er_y is user-marked in the training set. Accordingly, define the error function as Equation (21). Minimize the error function with OQNLP iterative algorithm [31] and the initial experience value as input. The trained parameters for the bike segmentation are shown in Table 8. It should be noted that for each kind of object, we need to separately train the network parameters due to their distinct geometric properties.

$$err(X_{ANFIS}) = \frac{1}{n_s} \sum_{k=1}^{n_s} \left| {}^k Tr_y(X_{ANFIS}) - {}^k Er_y \right| \tag{21}$$

where ${}^k Er_y$ is the ground truth cutoff distance in the k^{th} scene in totally n_s training scenes, ${}^k Tr_y$ is the cutoff distance estimated by the ANFIS.

Table 8. The relationship between the fuzzy sets and situations.

Variable	Before Training	After Training
A_{w1}	80	85.256
A_{w2}	180	181.263
A_{w3}	1	0.9584
A_{w4}	25	35.647
A_{w5}	0	0.1403
A_{w6}	1	1.3826
A_{w7}	50	36.08
A_{w8}	100	127.4
W_H	100	102.765
W_M	30	35.2615
W_L	2	1.8621

For the case of bike training, we choose the section width as 10 mm, and the adaptive variables before and after training are shown in Table 8. The initial selection of these values is based on experimental experience. For example, regarding the truncating range for the bike, it is initially set to be 80–180 according to the size of bikes (corresponding to 800–1800 mm) by the parameters A_{w1} and A_{w2}. As for the point numbers in each section,

the truncating section only contains a small number of noises or ground point clouds, so we set the initial parameters of A_{w3} and A_{w4} as small numbers 1 and 25. As for the changes of point numbers in adjacent sections, the truncating section should be in a minor change condition, so 0 and 1 are chosen as the initial values of A_{w5} and A_{w6}. Finally, the average height in the Z direction for each section is set in the range of 50–100 by A_{w7} and A_{w8}, according to the height of a bike. For the weights of fuzzy output, we hope they can distinguish the three conditions, so W_H, W_M, and W_L are set to be with significant differences as100, 30, and 2, respectively.

The last three input curves for the ANFIS are shown in Figure 17a–c, and the trained network outputs a $\boldsymbol{Q}$ curve as in Figure 17d. The largest peak of the $\boldsymbol{Q}$ curve corresponds to the 95th section, which gives a point set truncation place from the side view and bird view in Figure 17e,f, respectively. It is chosen at a place right behind the bike and before most of the background objects. Therefore, all the points on the bike are preserved, and most of the outliers are removed successfully.

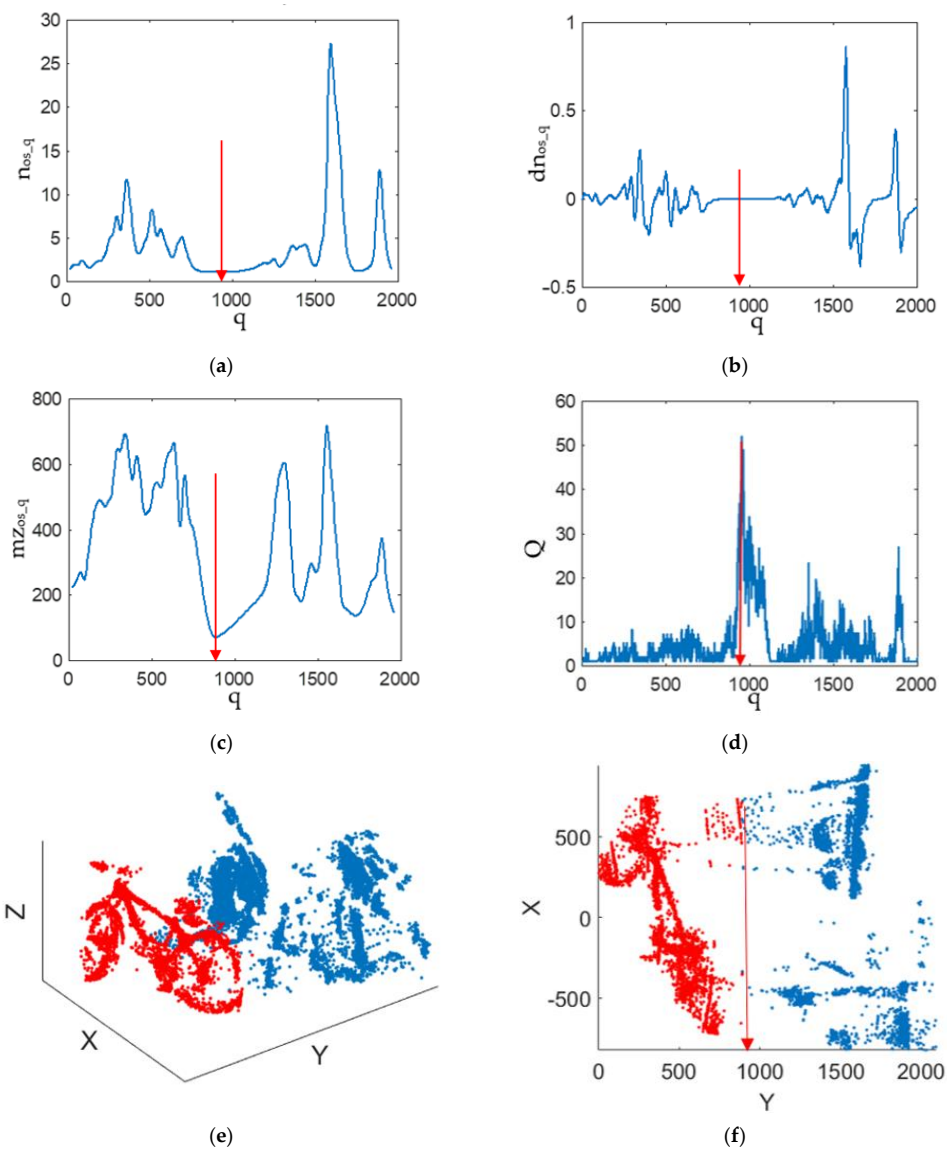


Figure 17. Truncating results for the 3D bike point cloud of Figure 13a: (**a**) the input curve of y_{os_q}; (**b**) the input curve of n_{os_q}; (**c**) the input curve of dn_{os_q}; (**d**) the output curve of mz_{os_q}; (**e**) the side view of the bike segmentation result; (**f**) the bird view of the bike segmentation result.

4.2. 3D Boxing and Refinement

The point cloud of the target object is generally incomplete due to occlusions, so the 3D box is hard to calculate by only parts of the target object directly. The technique to handle this problem is called Amodel Boxing and has been studied by previous researchers with deep learning techniques [24,34]. Out of all the impressive work, we choose to use the Amodel Boxing part of the Pointnet. The network contains two lightweight subnets, as shown in Figure 18, for the centroid regression and 3D box estimation, respectively. Both networks have a shared multilayer perceptron (MLP) on each point and then use a max-pooling layer to get the global features of all the points. A fully connected layer outputs each target's residual center and box parameters using the stacked global features. The inputs of the networks are the 3D point cloud matrices resulting from the point truncation above, and the outputs are the centroid and 3D box of the object, respectively. The localization result of the bike with localization networks is shown in Figure 19.

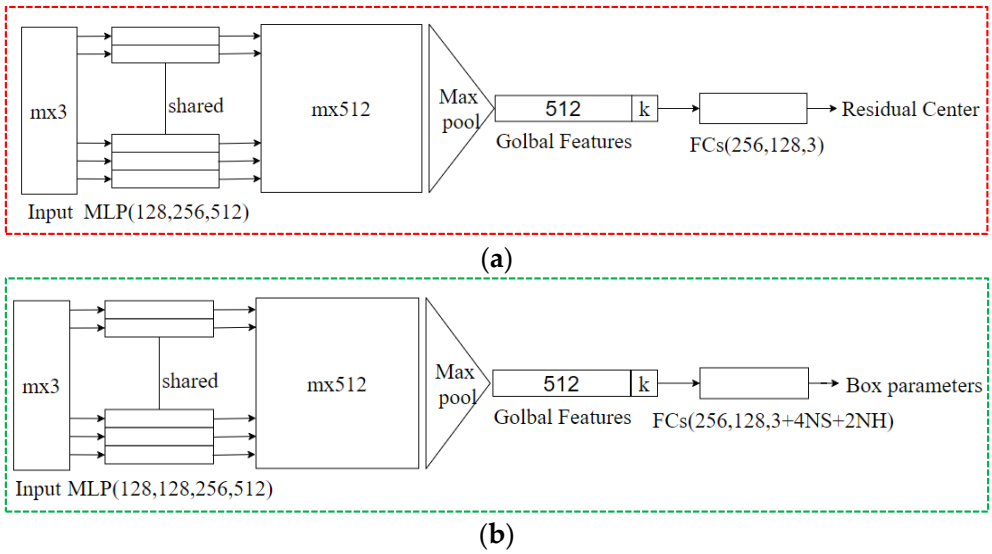


Figure 18. Localization Networks [24]: (**a**) T-net for estimating residual center; (**b**) Amodel 3D Box Estimation Net.

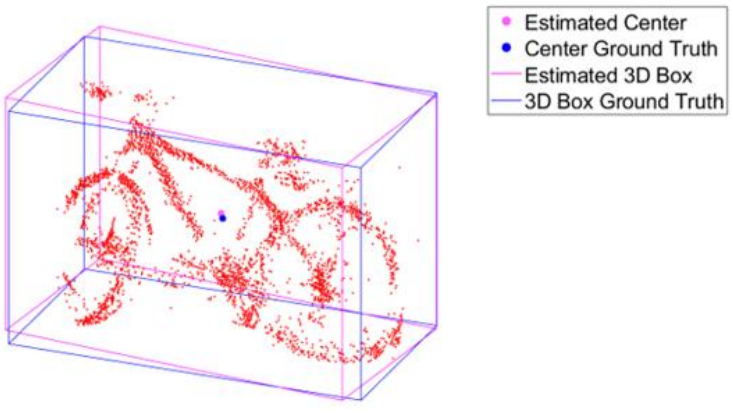


Figure 19. Refinement and Location Results for the bike object in Figure 16.

5. Experimental Results and Discussion

In this section, we designed two kinds of experiments to evaluate the 3D object localization performance of the device and the algorithms. The first experiment is to identify and localize bikes with different distances, backgrounds, and locations. As discussed above, the bike is a typically framed object, and the recognition and 3D segmentation from the

background of such objects are always considered challenging tasks. The second experiment is multi-kind object recognition and localization in different scenes. This is designed to testify to the system performance in complex environments.

The experiments were conducted on the testbed shown in Figure 20. The dense scanning system was installed on a mobile robot, so the techniques in this paper potentially can provide rich environmental information for the robot's automation operations.

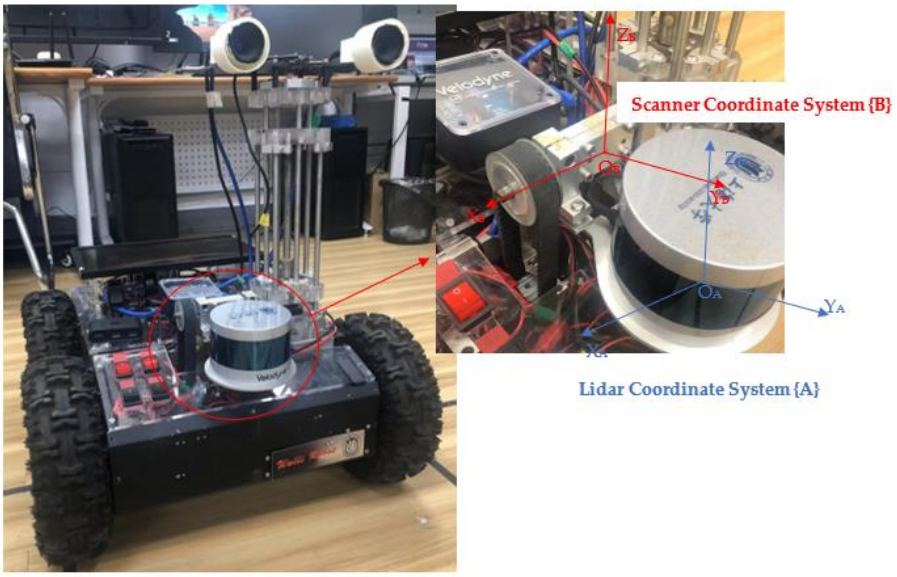


Figure 20. The dense scanning system installed on a mobile robot.

5.1. 3D Localization of Bikes

In the bike test, we leveled the difficulties of the bike instances with similar criteria as the KITTI benchmark [26]:

- Easy: Minimum bounding box height: 60 Px; maximal occlusion level: fully visible; maximal truncation: 15%;
- Moderate: Minimum bounding box height: 40 Px, maximal occlusion level: partly occluded, maximal truncation: 25%;
- Hard: Minimum bounding box height: 35 Px, maximal occlusion level: difficult to see, maximal truncation: 40%.

We collected 150 bikes with different sizes and appearances as the training dataset. The dataset was enhanced by adding Gaussian noise, downsampling, flips, rotations, and truncations, which ultimately provided 1500 images as the training set for the transfer learning. Another 150 images were treated as the dataset for testing. The specific recognition accuracy rate under each difficulty level is shown in Table 9. The average precision (AP) is the precision averaged across all unique recall levels. Though the quality and resolution of the images are limited, the AP of the bike recognition was still high. This indicates that the image by the 3D dense LiDAR is effective in recognizing target objects.

Table 9. The AP of the Bike Recognition (IoU = 0.8).

	Easy	**Middle**	**Hard**
AP	90.84	84.23	76.66

Some examples of the object recognition results are shown in Figures 21 and 22. The masks with different highlight colors show recognized and segmented bikes. Though the bikes were placed in various locations and poses, most of them were identified, and the masks separated the bikes from the background with acceptable contours errors. There

were still some failures for the hard cases, which are marked by red boxes in Figure 22a,b. The missing bike in Figure 22a is due to its small image occupancy, and in Figure 22b due to the lack of a large portion of the bike body in the image.

Figure 21. Some outdoor experimental scenes: (**a**–**j**) illsustrate the raw RBG images of the experimental scenes.

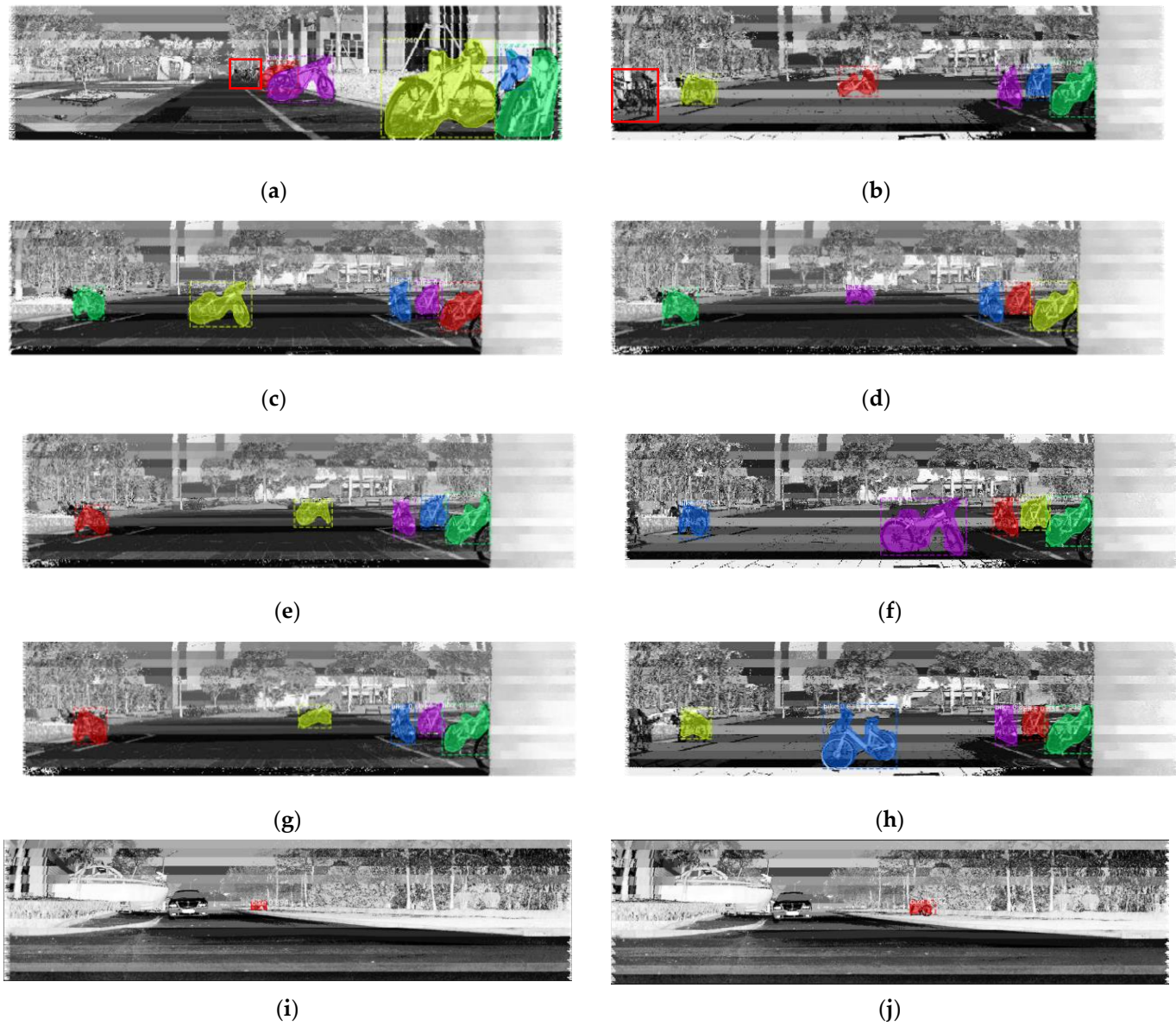


Figure 22. The results of bike recognition and 2D segmentation with Mask-RCNN: (**a**–**j**) the corresponding recognition results of ten scenes in Figure 21.

Several cases of ANFIS truncation results are shown in Figure 23. There are various situations, including a close bike with lots of ground outliers, a close bike with few outliers, a close bike with missing slices, a far bike with sparser points, a close bike with missing parts and complex background outliers, a bike on the image edge with a big portion of body missing. For all the easy, medium, and hard situations, the trained ANFIS made reasonable decisions on the truncating locations and removed most of the outliers. The same conclusion can be drawn from Figure 24, which shows the 3D localization results of the examples in Figure 21. The yellow boxes are the ground truth of the bikes, and the red boxes are the estimated box after outlier exclusion by the ANFIS and 3D box regression by the T-Net. As long as the bike is recognized in the 2D image, the outliers are properly removed and thus the center of the 3D box has no noticeable deviation from the true value.

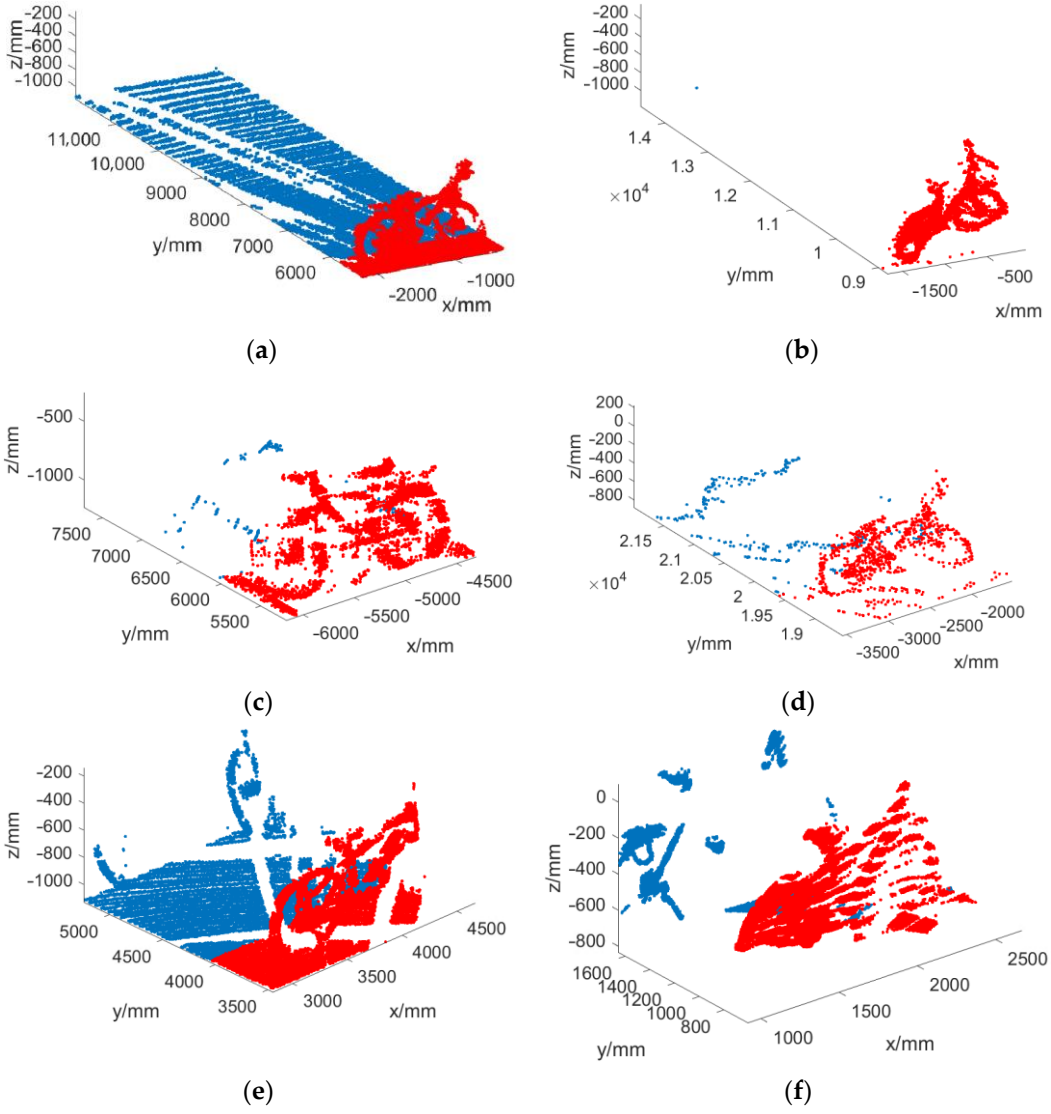


Figure 23. ANFIS truncation of bikes with different levels of difficulty: (**a**) the ANFIS results of one of the bikes in Figure 21c; (**b**) the ANFIS results of one of the bikes in Figure 21b; (**c**) the ANFIS results of one of the bikes in Figure 21g; (**d**) the ANFIS results of the bike in Figure 21i; (**e**) the ANFIS results of one of the bikes in Figure 21e; (**f**) the ANFIS results of one of the bikes in Figure 21a.

Traditionally, the 3D localization accuracy is estimated by the IoU of the 3D bounding box, which only provides box overlapping information between the result and the ground truth. For mobile robots, the more critical criteria are the position and orientation accuracy of the object. Therefore, we define two error terms err_{center} and err_{angle} in Equations (22) and (23). $\boldsymbol{D}_{GT}$ and θ_{GT} are the ground truths of the coordinates and heading angle of the bikes, respectively. h, w, l are the height, width, and length of the bounding box of the bikes, respectively. $\boldsymbol{D}_{final}$ and θ_{final} is the corresponding results output by the T-Nets, respectively. err_{center} describes the center deviation with respect to the dimension of the object, and the err_{angle} is the orientation estimation error. The two errors are critical for many mobile robot operations but have never been carefully studied in previous research.

$$err_{center} = \frac{\left|\boldsymbol{D}_{final} - \boldsymbol{D}_{GT}\right|}{\sqrt{h^2 + w^2 + l^2}} \times 100\% \tag{22}$$

$$err_{angle} = \left|\theta_{final} - \theta_{GT}\right| \tag{23}$$

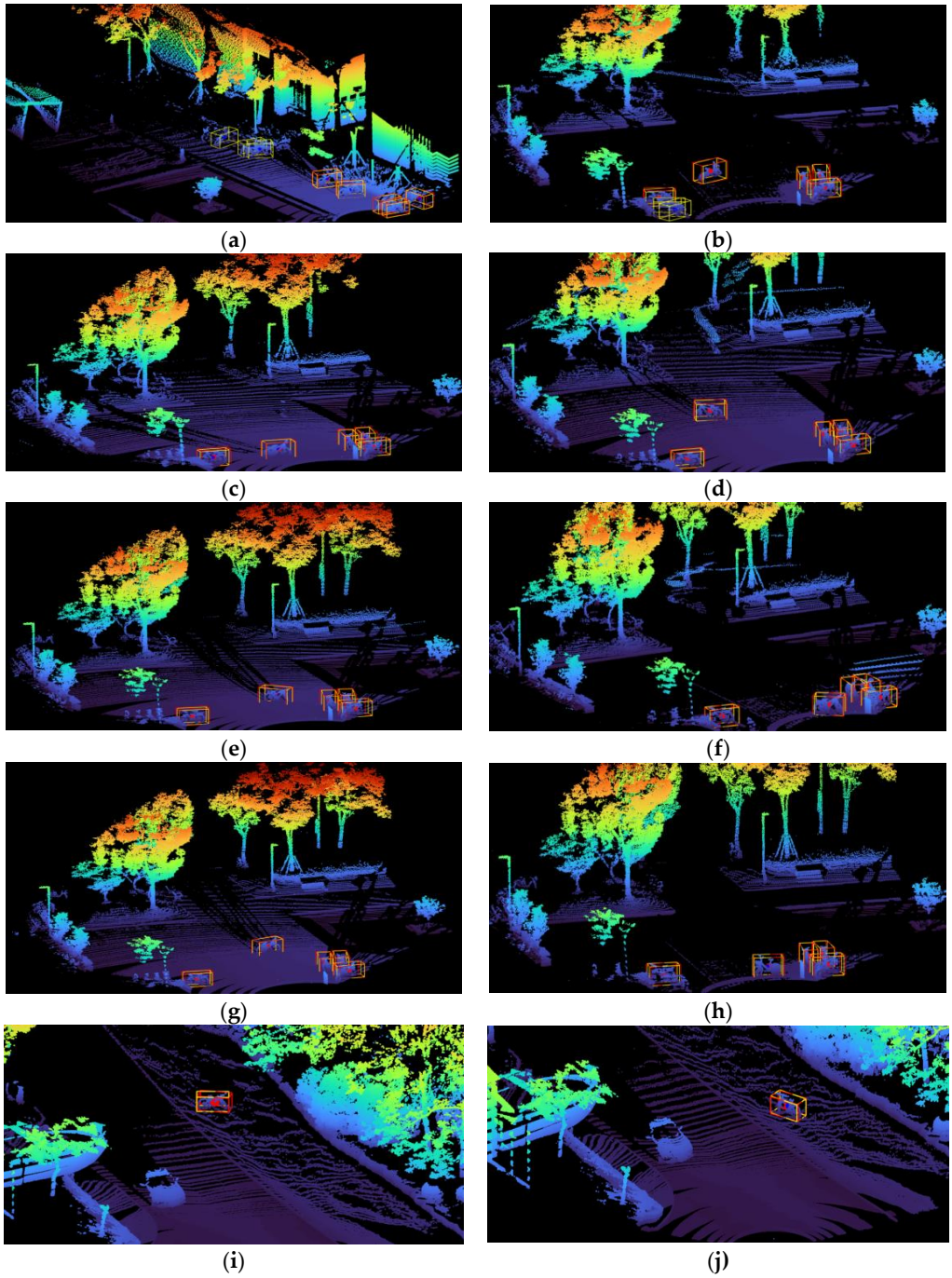


Figure 24. Localization results of the bikes. The yellow box is the ground truth of the bikes, and the red box is the results estimated by T-Net: (**a**–**j**) the corresponding localization results of ten scenes in Figure 21.

On the premise that the objects are recognized in the image, statistically, the 3D localization accuracy and the difficulty level of the object are positively correlated. As it is shown in Table 10, the average center location and heading direction errors are unnoticeable for easy cases, small for middle-level cases, but much larger for hard cases. The accuracy of 3D localization was seldomly analyzed in previous research. Some of them provide a roughly correlated index, the intersection over union (IoU), to have a coarse estimation of the 3D boxing accuracy. For cyclists, they had 0.5 in [23] and 0.34 in [28] as the IoU threshold for all the tests. In our work, the IoU is high as 0.822 for easy cases, and fairly high as 0.723 for middle difficulty level cases, which is much higher than previous studies.

Even for hard cases, the average IoU is 0.519, which is comparable with previous studies. The work in this paper presents an impressive improvement in localization accuracy.

Table 10. Localization error statistics for bikes.

Benchmark	$err_{center}/\%$			err_{angle}/rad			Average 3D IoU		
	Easy	Middle	Hard	Easy	Middle	Hard	Easy	Middle	Hard
Bike	0.831	3.55	10.425	0.021	0.246	0.815	0.822	0.723	0.519

5.2. Indoor and Outdoor Tests with Various Targets

In order to verify that the algorithm mentioned above is also applicable to other kinds of objects, outdoor and indoor experiments proceeded. The targets extended to be bikes and cars in the outdoor scene and chairs and balls in the indoor scene.

The outdoor scene is shown in Figure 25, where four cars and two bikes are the target objects to be recognized and localized. The 2D segmentation result is shown in Figure 26. The two bikes, though one of them is far and to some extent blended in the background, are both found and properly segmented from the background. Three out of the four cars are found. The one under the pink mask is discovered even though part of the car is occluded. Interestingly, we found that even for the cars, the original point cloud after 2D segmentation could be in a very undesirable situation, as seen in Figure 27c,d. It is due to the transparency of the glass window and the inevitable inaccuracy of the 2D segmentation at the edge of the objects. The ANFIS truncation essentially helped to decrease the influence from the outliers, which provided high accuracy in object 3D localization, as seen in Table 11. For the car in Figure 27e, the distance is 25m, and the point cloud has become sparse, but the center location estimation is still in highly accurate, and the orientation error is in an acceptable range. It is worth mentioning that this is an undetectable object if a sparse 64-line Laser scanner, like the one in KITTI [26] dataset, is used. In an ideal case, the whole car would only have less than 200 points, and in the actual KITTI [26] dataset, this number is down to be around 100. The information preserved by such a small number of points is hard to provide sufficient information for either object detection or location estimation.

Figure 25. The view of the outdoor test scene.

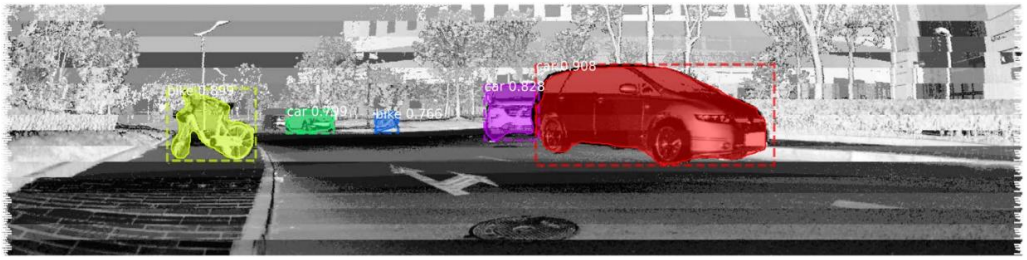


Figure 26. The 2D segmentation results from Mask RCNN of the outdoor scene.

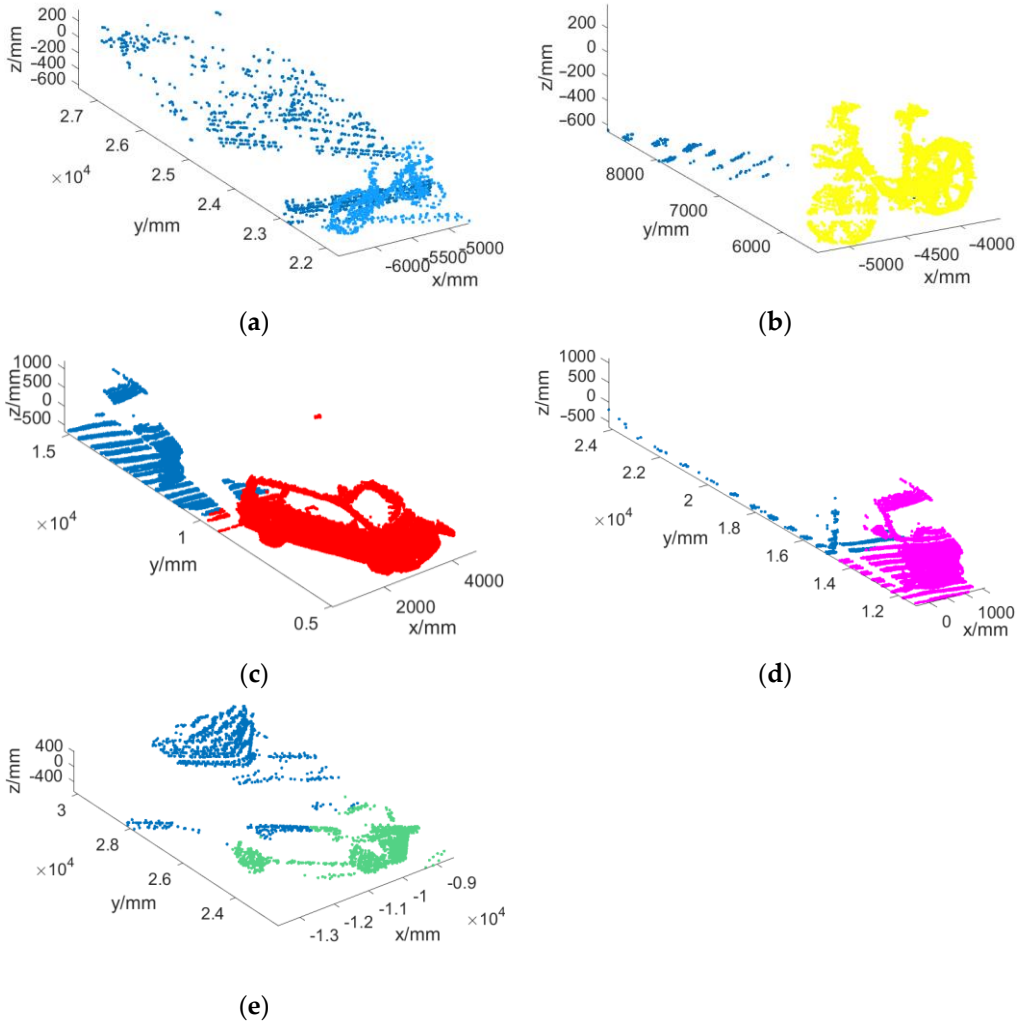


Figure 27. The ANFIS results in the outdoor test scene: (**a**–**e**) the ANFIS results of recognized targets with the same mask color in Figure 26.

Table 11. Localization error statistics in the outdoor test scene.

Benchmark	$\overline{err}_{center}$/%	$\overline{err}_{angle}$/*rad*
Car	4.32	0.528
Bike	2.56	0.024

There is still one failure case, which is the car in the red box as illustrated in Figure 26. The reason is clearly shown in Figure 27. The car is far, and its reflectivity data lack standard vehicle features in the ReV image. As a result, it is missed by the Mask-RCNN. It is evident that the weak spot in the pipeline of this paper is the object detection performance in 2D images. Once the object is detected, the outsider exclusion and 3D boxing technique generally function well in all our tests as shown in Figure 28.

The result for the indoor tests is very similar to the outdoor tests. The indoor test scene is shown in Figure 29. There are seven chairs and three balls scattered in the room, and all of them are recognized and properly segmented by the Mask-RCNN, as shown in Figure 30. The ANFIS helps remove most of the background outliers, as shown by the two examples in Figure 31. The 3D localization results are shown in Figure 32, with minimal errors as listed in Table 12. The distances of the objects with respect to the dense scanner are limited in the indoor scene, so the difficulty level is generally low, and the accuracy is thus higher than the outdoor cases.

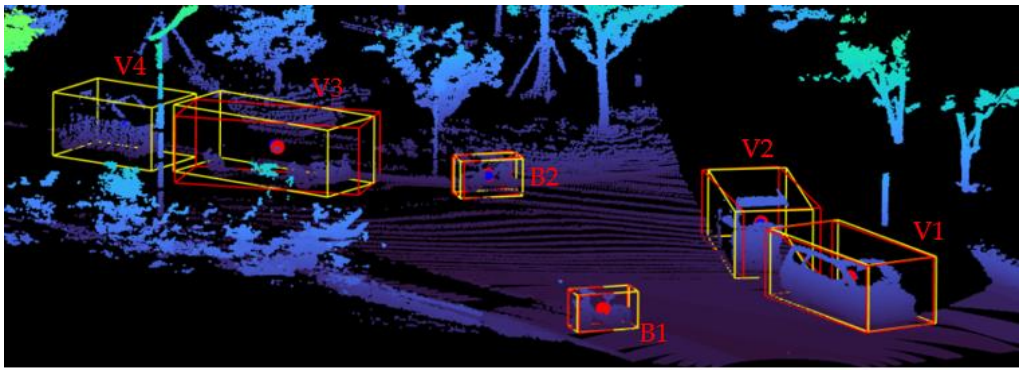
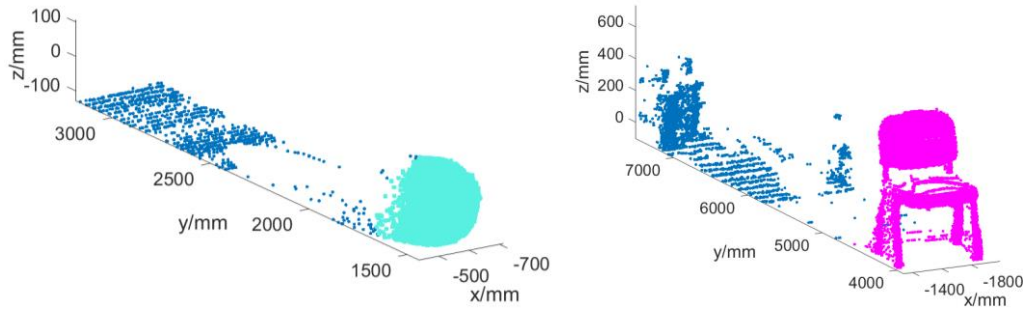


Figure 28. The 3D localization results of the cars and bikes in the outdoor test scene.

Figure 29. The view of the indoor test scene.

Figure 30. The 2D segmentation results from Mask RCNN of the indoor scene.



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

Figure 31. The ANFIS results in the indoor scene: (**a**) the ANFIS results of the sports ball in Figure 30 with the same mask color; (**b**) the ANFIS results of the chair in Figure 30 with the same mask color.

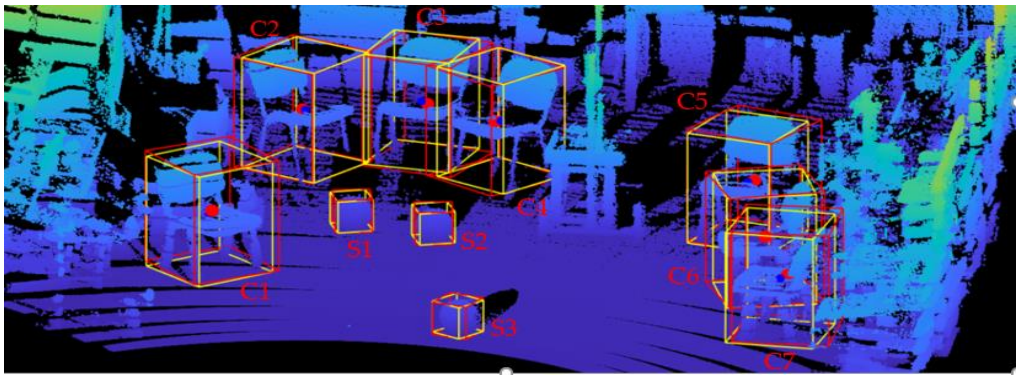


Figure 32. The localization results of the chairs and balls in the indoor scene.

Table 12. Localization error statistics in the indoor test scene.

Benchmark	$\overline{err}_{center}$/%	$\overline{err}_{angle}$/rad
Chair	0.33	0.0122
Sports ball	1.45	NULL

5.3. Comparision with Frustum-PointNet

In this section, we mainly discuss the performance comparison with state-of-the-art work in 3D object recognition and localization, the Frustum-Pointnet [24]. As stated above, the main improvement of this paper is to replace the original deep 3D segmentation network with a lightweight fuzzy logic network for the purpose of low computational cost. The codes for our algorithm and Frustum-Pointnet both ran on NVIDIA TX2 using the dense data in this paper. On average, it takes 3.3 s for one object segmentation by Frustum-Pointnet and only 0.13 s with the ANFIS in this paper as shown in Figure 33.

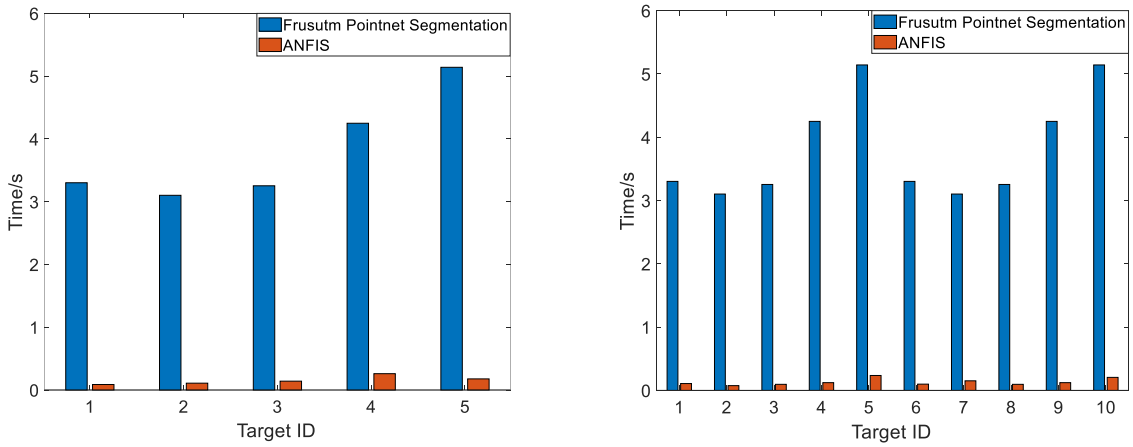


Figure 33. The comparison on time costs of different targets: (**a**) Time cost in the outdoor scene in Figure 25; (**b**) Time cost in the indoor scene in Figure 29.

As for the outlier exclusion effect, we selected cars to compare the performance because cars are originally included in the KITTI dataset used by Frustum-Pointnet. An example is shown in Figure 34. The ANFIS and Frustum-Pointnet have similar outlier exclusion capabilities. The Frustum-Pointnet is slightly better because it eliminates all the ground points, and the ANFIS still preserves a few due to its truncation mechanism. However, the influence on the 3D localization is limited, as shown in Figure 34. The car localization errors are quantitatively compared in Table 13, and the two methods have very similar

localization accuracies. Therefore, we can conclude that the ANFIS essentially improves the computation efficiency with neglectable sacrifice on the localization accuracy.

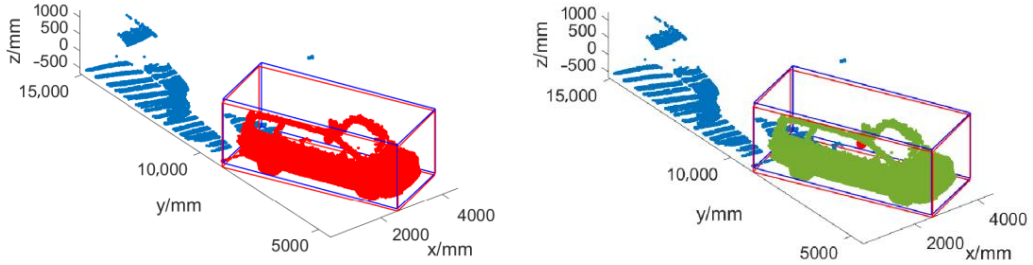


Figure 34. Comparison between the ANFIS method and Frustum-Pointnet [24], where the blue box is the ground truth, and the red box is the estimated results: (**a**) with outlier exclusion by ANFIS; (**b**) with the segmentation by Frustum-Pointnet.

Table 13. Localization error of the cars.

Benchmark	$\overline{err}_{center}$/%	$\overline{err}_{angel}$/*rad*
ANFIS	1.42	0.021
Frustum-Pointnet [24]	1.40	0.019

6. Conclusions

This paper presents a dense scanning system that recognizes and localizes targets from its 3D scanning cloud points. The system is capable of obtaining accurate geometric measurements for the surrounding environment. The fidelity of the data is ensured by a carefully designed system calibration process. As a result, the fine details of the objects are preserved without shape deformations, which helped to improve the recognition accuracy by 2D segmentation techniques applied on projected Lidar images.

In addition, an ANFIS is proposed to exclude the background noises, which is inevitably introduced by edge inaccuracy in the segmentation step. The method has the merits of low computation cost and performance robustness. It has been proven that the ANFISs can output clean 3D points of the objects with different features and placed in different situations. As a result, the system achieved high 3D localization accuracies in both center location and orientation estimations.

It is worth mentioning, though, that our method currently can only process the background outliers. When there are foreground outliers, the current ANFIS needs to be modified to provide both front and back truncations. In addition, though the object segmentation time is largely decreased, the amodel 3D boxing still uses deep networks, which might cause undesirable delays for mobile robots in real applications. Therefore, more research is needed to give fast and accurate estimations on the centroid and orientation of the targets, even with an incomplete point cloud.

Author Contributions: Conceptualization, H.G. and Y.X.; methodology, Y.X. and H.G.; software, H.G., G.F. and Z.G.; validation, H.G., G.F. and Z.G.; formal analysis, H.G., G.F. and Z.G.; investigation, H.G., G.F. and Z.G.; resources, H.G. and G.F.; writing—original draft preparation, H.G.; writing—review and editing, Y.X.; visualization, G.F. and Z.G.; supervision, Y.X.; project administration, Y.X.; funding acquisition, Y.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Shanghai Rising-Star Program, China, [grant number 19QA1403500]; and the Shanghai Natural Science Foundation, China [grant number 20ZR1419100].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Bula, J.; Derron, M.H.; Mariethoz, G. Dense point cloud acquisition with a low-cost Velodyne VLP-16. *Geosci. Instrum. Methods Data Syst.* **2020**, *9*, 385–396. [CrossRef]
2. Zhuang, Y.; Jiang, N.; Hu, H.; Yan, F. 3-D-laser-based scene measurement and place recognition for mobile robots in dynamic indoor environments. *IEEE Trans. Instrum. Meas.* **2012**, *62*, 438–450. [CrossRef]
3. Haala, N.; Peter, M.; Kremer, J.; Hunter, G. Mobile LiDAR mapping for 3D point cloud collection in urban areas—A performance test. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2008**, *37*, 1119–1127.
4. Jaboyedoff, M.; Oppikofer, T.; Abellán, A.; Derron, M.H.; Loye, A.; Metzger, R.; Pedrazzini, A. Use of LIDAR in landslide investigations: A review. *Nat. Hazards* **2012**, *61*, 5–28. [CrossRef]
5. Wang, W.; Zhao, W.; Huang, L.; Vimarlund, V.; Wang, Z. Applications of terrestrial laser scanning for tunnels: A review. *J. Traffic Transp. Eng.* **2014**, *1*, 325–337. [CrossRef]
6. Wellington, C.; Stentz, A. Learning predictions of the load-bearing surface for autonomous rough-terrain navigation in vegetation. In *Field and Service Robotics*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 83–92. [CrossRef]
7. Wulf, O.; Wagner, B. Fast 3D scanning methods for laser measurement systems. In Proceedings of the International Conference on Control Systems and Computer Science, Bucharest, Romania, 2–5 July 2003; pp. 2–5.
8. Bosse, M.; Zlot, R.; Flick, P. Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping. *IEEE Trans. Robot.* **2012**, *28*, 1104–1119. [CrossRef]
9. Zhang, J.; Singh, S. LOAM: Lidar Odometry and Mapping in Real-time. *Robot. Sci. Syst.* **2014**, *2*, 9.
10. Glennie, C.; Lichti, D.D. Static Calibration and Analysis of the Velodyne HDL-64E S2 for High Accuracy Mobile Scanning. *Remote Sens.* **2010**, *2*, 1610–1624. [CrossRef]
11. Yuan, C.; Bi, S.; Cheng, J.; Yang, D.; Wang, W. Low-Cost Calibration of Matching Error between Lidar and Motor for a Rotating 2D Lidar. *Appl. Sci.* **2021**, *11*, 913. [CrossRef]
12. Guo, Y.; Wang, H.; Hu, Q.; Liu, H.; Liu, L.; Bennamoun, M. Deep learning for 3D point clouds: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 4338–4364. [CrossRef] [PubMed]
13. Maturana, D.; Scherer, S. Voxnet: A 3D convolutional neural network for real-time object recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems, Hamburg, Germany, 28 September–2 October 2015; pp. 922–928. [CrossRef]
14. Riegler, G.; Osman Ulusoy, A.; Geiger, A. Octnet: Learning deep 3D representations at high resolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3577–3586. Available online: https://arxiv.org/abs/1611.05009 (accessed on 17 October 2020).
15. Wang, P.S.; Liu, Y.; Guo, Y.X.; Sun, C.Y.; Tong, X. O-CNN: Octree-based convolutional neural networks for 3D shape analysis. *ACM Trans. Graph.* **2017**, *36*, 1–11. [CrossRef]
16. Rethage, D.; Wald, J.; Sturm, J.; Navab, N.; Tombari, F. Fully-convolutional point networks for large-scale point clouds. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 596–611. Available online: https://arxiv.org/abs/1808.06840 (accessed on 18 October 2020).
17. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. PointNet: Deep learning on point sets for 3D classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660. [CrossRef]
18. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5105–5114. Available online: https://arxiv.org/abs/1706.02413 (accessed on 20 June 2019).
19. Wu, W.; Qi, Z.; Fuxin, L. Pointconv: Deep convolutional networks on 3D point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9621–9630. [CrossRef]
20. Thomas, H.; Qi, C.R.; Deschaud, J.E.; Marcotegui, B.; Goulette, F.; Guibas, L.J. Kpconv: Flexible and deformable convolution for point clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 6411–6420. [CrossRef]
21. Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12697–12705. [CrossRef]
22. Milioto, A.; Vizzo, I.; Behley, J.; Stachniss, C. Rangenet++: Fast and accurate lidar semantic segmentation. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, Macau, China, 3–8 November 2019; pp. 4213–4220. [CrossRef]
23. Wu, B.; Wan, A.; Yue, X.; Keutzer, K. Squeezeseg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D lidar point cloud. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1887–1893. [CrossRef]

24. Qi, C.R.; Liu, W.; Wu, C.; Su, H.; Guibas, L.J. Frustum pointnets for 3D object detection from RGB-D data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 918–927. [CrossRef]
25. Zhao, X.; Liu, Z.; Hu, R.; Huang, K. 3D object detection using scale invariant and feature reweighting networks. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 9267–9274. [CrossRef]
26. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The kitti vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361. [CrossRef]
27. Wang, Z.; Jia, K. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, Macau, China, 3–8 November 2019; pp. 1742–1749. [CrossRef]
28. Paigwar, A.; Sierra-Gonzalez, D.; Erkent, O.; Laugier, C. Frustum-pointpillars: A multi-stage approach for 3D object detection using rgb camera and Lidar. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 11–17 October 2021; pp. 2926–2933.
29. Shin, K.; Kwon, Y.P.; Tomizuka, M. Roarnet: A robust 3D object detection based on region approximation refinement. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, 9–12 June 2019; pp. 2510–2515. [CrossRef]
30. Jolliffe, I.T. Principal Components in Regression Analysis. In *Principal Component Analysis*; Springer Series in Statistics; Springer: New York, NY, USA, 1986; pp. 129–155. [CrossRef]
31. Ugray, Z.; Lasdon, L.; Plummer, J.; Glover, F.; Kelly, J.; Martí, R. Scatter search and local NLP solvers: A multistart framework for global optimization. *INFORMS J. Comput.* **2007**, *19*, 328–340. [CrossRef]
32. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969. [CrossRef]
33. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2014; pp. 740–755. Available online: https://arxiv.org/abs/1405.0312 (accessed on 15 July 2019).
34. Meyer, G.P.; Laddha, A.; Kee, E.; Vallespi-Gonzalez, C.; Wellington, C.K. Lasernet: An efficient probabilistic 3D object detector for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12677–12686. [CrossRef]