Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

A Multi-Sensor Fusion Based 2D-Driven 3D Object Detection Approach for Large Scene Applications*

Yingtian Liu[1], Chuanzhe Suo[2], Zhe Liu[2], and Yun-Hui Liu[2,1], *Fellow, IEEE*
[1]*School of Mechanical Engineering and Automation, Harbin Institute of Technology, Shenzhen.*
[2]*T Stone Robotics Institute and Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong.*

***Abstract*— In this paper, we propose a 3D object detection approach for large-scale environments by considering the 2D visual information and 3D lidar data simultaneously. In the proposed multi-sensor fusion framework, first of all, a large-scene application oriented data alignment method is presented, which utilizes the EPnP based joint calibration method in order to achieve an improved data-association performance. Based on which, the 2D bounding box of each object is generated by using the visual detection and the 3D point cloud contained in the corresponding 3D frustum can be obtained. Then, after the ground fitting and point cloud clustering operations, an innovative scoring mechanism is proposed to evaluate each point cloud cluster and find out the cluster belonging to the detected object. Finally, the accurate lidar point cloud segment in the 3D frustum can be generated. Comparing with existing 3D object detection approaches, we propose a more general framework in this paper, which does not require a large number of 3D point cloud detection datasets to train the network and can be used on the 3D lidar with any number of lines, thus a real-time 3D detection system can be achieved. Experiments in large-scale airport environments validate the advanced performance of the proposed approach.**

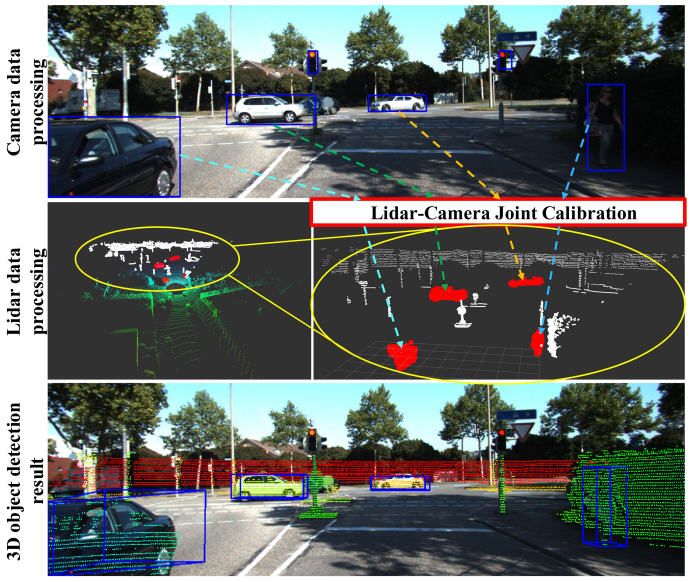

Fig. 1. **2D-driven 3D detection pineline.** Given the RGB image data (top) and the 3D point cloud data (middle), we first generate 2D object detection bounding boxes in the 2D image. Second, based on the LiDAR-camera joint calibration, we extract the 3D point cloud contained in each 2D box and then segment the object points in it. Finally, the multi-sensor fusion based 3D object detection results can be obtained (bottom). The lidar-image data association results are also shown in the bottom figure, which demonstrates the effectiveness of the proposed joint calibration method.

I. Introduction

In recent years, significant progress has been made in 2D image understanding tasks such as object detection [1, 2] and semantic segmentation [3]. Accordingly, especially in autonomous driving scenes, 3D understanding is urgently needed in actual application scenario. However, 3D perception has not been fully explored to a large extent. With the popularity of 3D sensors in mobile devices and autonomous vehicles, more and more 3D data is being acquired and processed. Camera images have rich and orderly RGB information, which can extract a variety of texture features to achieve various difficult perception tasks, but it is not easy to obtain accurate 3D information. In contrast, the lidar 3D point cloud can provide accurate depth and reflection intensity information. However, due to the sparsity and disorder of the lidar point cloud, it is difficult to achieve tasks such as detection and segmentation, especially for the 3D lidar with fewer lines, such as 16-line lidar. RGB-D sensors such as Kinect and RealSense have both image information and depth information, but their sensing range is typically too short, and their performance is seriously affected by complex ambient light. It is difficult to perform well in large scenes with the complex environments such as autonomous driving scene. Considering the respective advantages of the 2D camera and 3D lidar, in this work, we propose a sensor fusion framework to explore one of the most important tasks of 3D perception called 3D object detection, which classifies objects and estimates their position from 3D sensor data. The fusion of lidar point clouds and RGB images should enable higher performance and safety for autonomous vehicles.

*This work is supported in part by the Shenzhen Peacock Plan Team Grant KQTD20140630150243062, in part by the Natural Science Foundation of China under Grant U1613218, in part by the Hong Kong ITC under Grant ITS/448/16FP, in part by the National Key Research and Development Program of China under Grant 2018YFB1309300. The author's E-mail: liuyingtian@stu.hit.edu.cn. Corresponding author: Yun-Hui Liu, E-mail: yhliu@mae.cuhk.edu.hk.

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

There are many methods for 2D image detection, which can be performed well both in efficiency and precision. However, how to efficiently perform 3D object detection in point cloud data is still a challenging problem to be solved. A lot of existing works focus on converting the 3D point cloud into the image plane through the projection [4, 5], and some work uses the expression of the front view [6] and the bird's eye view [7] of the 3D point cloud to enter the deep learning network. There are also some works to convert a 3D point cloud into volumetric grids [8, 9] and then apply the convolution network. However, these data representations are lossy and do not solve the problem with the most realistic data. Recently, there are some papers that propose direct processing of point clouds without converting them to other formats [10, 11], but they are mainly for small scene objects.

In contrast, in large scene applications, we must solve a key challenge: how to effectively extract possible locations of 3D objects in the large 3D space.

Based on the above analysis, this paper proposes a 3D detection framework based on camera and lidar data fusion (as shown in Fig. 1). Due to the limitations of existing large-scale 3D detection datasets for autonomous driving, such as KITTI [12], most 3D detection methods can only be limited to detecting common objects such as pedestrians, cars and cyclists currently. Unlike them, our framework can easily achieve almost as many 3D detection categories as 2D image detection categories. Our 2D-driven 3D object detection method does not require a large number of 3D detection datasets to train the network. We only need to directly embed the currently effective 2D object detection method to perform 3D detection for the objects that we want to detect. The current autonomous driving datasets, such as KITTI and nuScenes, mainly use the 64-line or 32-line lidar, making most 3D object detection networks, which trained with these datasets, perform not normal at 16-line lidar data. In practical applications, high-line lidar is very expensive, also a large number of datasets of 16-line lidar is not easy to obtain and the training labels are very expensive. Thanks to the scoring mechanism proposed in this paper, all point cloud processing parts in the 3D detection framework do not need to use deep learning for training, which means that we only need easy-to-obtain 2D detection datasets to implement 3D detection for the required objects, and can be applied normally on 3D lidar with any number of lines.

The key contributions of our work are as follows:

- A 2D-driven 3D object detection framework based on the camera and lidar data fusion is proposed in this paper, which ensures real-time 3D detection, does not need large number of 3D lidar training datasets and can be applied to the 3D lidar with any number of lines.
- A large-scene application oriented lidar-image joint calibration method is presented which does not require other auxiliary calibration tools on the premise of knowing the camera intrinsic parameters. Based on which, we associate the 3D point cloud into the image plane and an innovative scoring mechanism is further proposed to evaluate point cloud clusters and find out the accurate object point cloud segment in the 3D frustum.
- The proposed approach has been evaluated in Hong Kong Airport for large scene autonomous driving applications. Experiment results successfully demonstrate the advanced performance of 3D object detection, thus showing the practical applicability of the proposed approach.

II. Related Work

3D Object Detection in Point Cloud. Vote3D [13] uses the method of sliding window to locate the object, then uses SVM to classify, and the 3D object classifiers are trained by manual geometry features extracted from point cloud. On the basis of [13], to improve feature representation ability, Vote3Deep [14] firstly voxelates the point cloud into a 3D grid, and replaces SVM with the feature representation of 3D CNN. However, due to the high cost of 3D convolution and the large space of 3D search, the calculation cost of these methods is usually very high. VeloFCN [6] projects point cloud to the front view, and then applying the fully convolution network on the 2D point map, so it could regress 3D boxes densely from the convolutional feature maps. 3D-FCN [7] exploits a BEV representation of the LIDAR and achieves 3D detection by a 3D FCN (fully convolutional network). But these methods of representing point clouds using various maps are lossy that lose some of the 3D geometry of the point cloud. Recently, there are few related works [10, 11] achieve classification and segmentation using raw disordered 3D points directly. Among them, PointNet [11] is the first work to propose such a novel idea. And how to apply them to the autonomous driving scenes is also widely concerned. In this work, we also directly operate on the raw unordered point cloud data to keep the most authentic data of the lidar.

Joint Camera-3D Sensor Detection. In recent years, some works begin to explore the data fusion of camera and 3D sensor for 3D perception tasks. A common approach is based on the depth image processing, which encodes 3D geometry as an additional image channel [15]. However, the output space of these methods is on the camera's image plane. In the context of autonomous driving, we wish to localize objects in 3D space. MV3D [16] is the first 3D RPN architecture using multi-view sensory fusion of autonomous driving scenes. It integrates multiple view information including RGB image, BEV view and front view. It generates 3D object proposals from BEV map and projects them to other views, and then learns the fusion feature from various views to classify the objects and regress

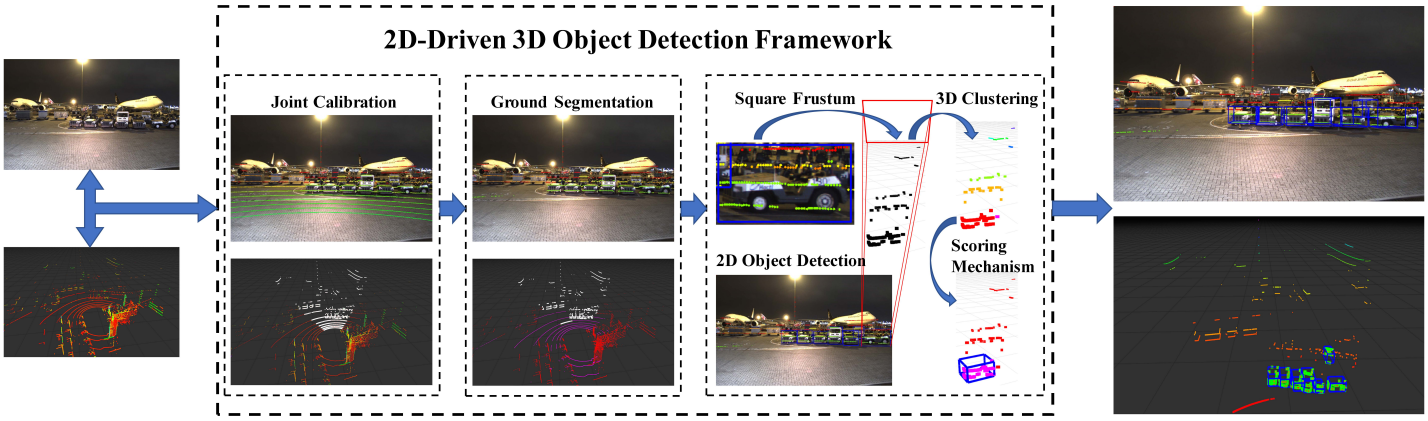

Fig. 2. **System structure of the proposed 2D-driven 3D object detection framework.** In the **joint calibration** step, we obtained a fusion image of the lidar points projection to the image. The white portion of the bottom lidar map represents the lidar points corresponding to the image. In the **ground segmentation** step, the lidar point cloud belonging to the ground is removed, and the pink points represent the ground plane. In the next step, a **2D object detection** is performed on the fusion image, and then every **3D square frustum** is extracted from each 2D bounding box to obtain a point cloud, as shown in black. After **3D clustering**, different color points represent different objects in 3D space. Finally, using our proposed **scoring mechanism**, the target object point cloud can be segmented into pink points. And the final 3D object detection results are represented for airport trailers that lack 3D point cloud training dataset in the fusion image and the 3D lidar map, respectively.

their 3D bounding boxes. Similarly, AVOD [17] utilizes the feature of full resolution feature maps. In ContFuse [18], continuous fusion layers are used to fuse the image features onto the BEV feature maps, achieving better detection result for small objects. For the above networks, due to the lack of lidar point cloud datasets, the performance of sparse point cloud with few lines, such as the data of 16-line lidar, is abnormal.

Monocular-Based Proposal Generation. Another type of the state-of-the-art methods are to use the mature 2D object detector to generate 2D regions and classify their content, then extend the 2D regions to 3D truncated cone proposals which could predict the 3D shapes and locations. This trend has begun with [19] for indoor object detection, they use point histograms and simple FCN to regress the position and orientation of the 3D box. F-pointnet [20] is the first work to apply the 2D-driven 3D approach to autonomous driving scene, using point-wise features of PointNet [11] to directly predict the 3D shape and position of the target in the truncated cone proposal generated by the 2D bounding box and achieving very good results. A similar approach is used in our work, and our approach does not require a large number of expensive point cloud datasets to train the network to detect some specific objects.

III. Multi-Sensor 3D Object Detection Approach

As shown in Fig. 2, the proposed 3D object detection framework aims to classify objects and locate them in 3D space, mainly includes three parts: the data alignment of camera and lidar, the proposal of 3D frustum generated by 2D object detection, and the point cloud processing of 3D lidar. Furthermore, the 3D lidar point cloud processing includes ground segmentation, 3D point cloud clustering and point cloud segmentation of the target object based on a novel scoring mechanism.

A. Data alignment of camera and lidar

The data alignment of camera and lidar includes temporal alignment and spatial alignment. Temporal alignment can be achieved by recording the timestamp of each sensor data and comparing them, using the ROS synchronizer. For spatial alignment, the camera and the 3D lidar need to be jointly calibrated to obtain the projection matrix from the lidar frame to the camera plane. Currently, there are some popular methods for joint calibration of camera and 3D lidar, such as Lidar_Camera_Calibration [21], Autoware, Apollo and other ROS packages. But the first two require special calibration board and calibration process. Apollo's calibration also requires data of odometer and other sensors. What's more, most calibration methods need good initial external parameters, and the calibration results of 16-line lidar can not fulfill the requirements of practical applications.

This paper hopes that we can quickly apply the proposed framework to the actual projects of using arbitrary multi-line 3D lidar, so we designed a joint calibration method (as shown in Fig. 3) that does not need any auxiliary calibration props, and only need to provide part of the synchronous data of the camera and the lidar. According to the principle of the data correspondence between camera and lidar (as in Eq. 1), joint calibration is an estimation problem of 3D-2D projection matrix, which can be regarded as solving the relative pose of camera to lidar, including translation and rotation. So we adopt EPnP [22] method to solve it. EPnP method requires the following conditions: Firstly, it needs to obtain n pairs ($n \geq 4$) of matched 3D lidar coordinate

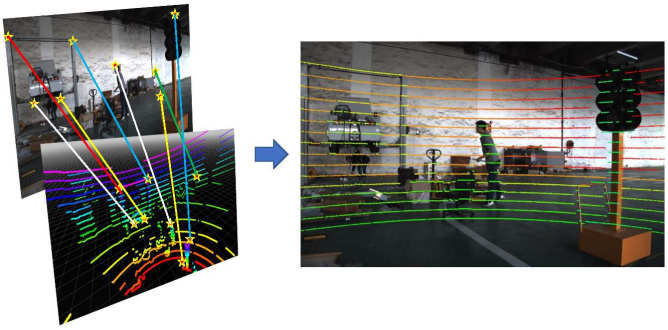

Fig. 3. **Lidar-Camera Joint Calibration.** Find the corresponding points of multiple sets of 2D pixels and 3D lidar, use PnP method to solve the projection matrix, and project the 3D lidar point cloud into the image to obtain the fusion image. In the fusion image, the color of the projected lidar points change from green to red to represent the distance from near to far.

points (x, y, z) and 2D pixel coordinate points (u, v). These matching points can be manually selected and recorded on some software for viewing 3D point cloud and 2D image respectively. Secondly, the intrinsic camera parameters K of the camera needs to be provided, and the precise intrinsic parameters K of the camera can be used to obtain the precise extrinsic parameters T. Even if there is no accurate intrinsic parameters K, the exact projection matrix P value can always be obtained directly, without disrupting the subsequent flow of the framework proposed in this paper.

$$\begin{cases} s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = P\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = KT\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \\ KT = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \end{cases} \tag{1}$$

Through the proposed joint calibration method, even with 16-line lidar, we can obtain an accurate projection matrix and project the 3D lidar point cloud onto the camera plane (as shown in the Fig. 3) to obtain a fusion image that contains both 2D image information and corresponding 3D point cloud information.

B. 3D frustum proposal generated by 2D object detection

Since the resolution of data produced by lidar is much lower than that of RGB image, 2D object detector is more reliable for object positioning and classification. Therefore, this paper firstly proposes 2D target regions in RGB image with a mature 2D object detector and completes the classification task for their content. Then, according to the obtained fusion image that contains both 2D image information and corresponding 3D point cloud information, all 3D points contained in the 2D target regions are extracted, so each 2D region is expanded into a 3D cone region (as shown in Fig. 4), which called a square frustum. Essentially, the 3D frustum provides much smaller space to search than the entire visual area captured by lidar. Each frustum contains all the 3D points of the object we need to detect, and the object's category has been determined.

The 3D detection framework of this paper does not need to specify a specific 2D object detection method. We can choose a state-of-the-art 2D object detection method according to our own needs. In order to consider the effect of precision and speed at the same time, we choose YOLOv3 [2] as the 2D object detection method.

C. 3D detection

*1) **Ground segmentation**:* A large part of the 3D lidar point cloud is a set of points belonging to the ground surface, so the segmentation and removal of the ground point cloud can greatly reduce the number of points involved in the subsequent process calculation. Moreover, due to the line scanning characteristic of lidar, the ground point cloud may cluster multiple objects in the nearby point cloud or cluster itself with the object points, and may also increase the number of clustering categories of distant points, resulting in the difficulty of object point cloud segmentation.

The ground segmentation in this framework can be designed according to our own requirements. In most autonomous driving scenes, the ground is flat. Here, we use a plane fitting method to segment the ground in the whole lidar point cloud. We assume that the ground is a simple linear model (as in Eq. 2). To estimate the plane model, it is equivalent to solving the model parameters (a, b, c, d). We can firstly solve $\mathbf{n} = (a, b, c)^T$, and then solve d by substituting a reliable point into Eq. 2. First of all, since the lowest height points are most likely to belong to the ground surface, we can refer to [23] to obtain some reliable seed points of ground by using the iterative method.

$$\begin{cases} ax + by + cz = d \\ \mathbf{n}^T\mathbf{x} = d \end{cases} \tag{2}$$

$$A\mathbf{n} = \begin{bmatrix} x_1 - \overline{x} & y_1 - \overline{y} & z_1 - \overline{z} \\ x_2 - \overline{x} & y_2 - \overline{y} & z_2 - \overline{z} \\ \cdots & \cdots & \cdots \\ x_n - \overline{x} & y_n - \overline{y} & z_n - \overline{z} \end{bmatrix}\begin{bmatrix} a \\ b \\ c \end{bmatrix} = 0 \tag{3}$$

$$\begin{cases} \mathbf{n}^* = argmin_{\mathbf{n}} \|A\mathbf{n}\| \\ \|\mathbf{n}\| = 1 \end{cases} \tag{4}$$

In the ideal case, these seed points would satisfy Eq. 3, where $(\overline{x}, \overline{y}, \overline{z})$ is the average of all seed points. However, due to the existence of noise, there is no $\mathbf{n}$ that can fully satisfy Eq. 3. Fortunately, we can fit the plane by optimizing the objective function (as in Eq. 4) to obtain the optimal solution $\mathbf{n}^*$ as the model parameters (a, b, c), and $\|\mathbf{n}\| = 1$ is the constraint condition. The physical meaning of Eq. 4 is to minimize the sum of the distances from all seed points to the

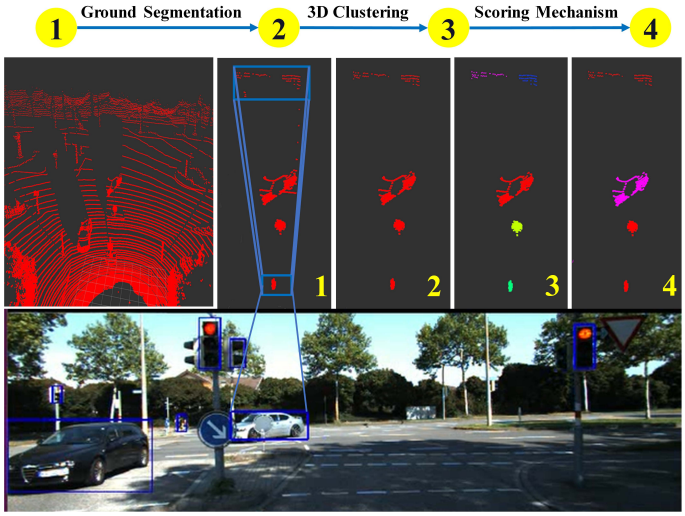

Fig. 4. **3D lidar point cloud processing.** Step 1 extracts the 3D square frustum from the 2D bounding box. After Ground Segmentation and 3D Clustering, the different objects are discretely separated in the 3D region. The challenge is that in step 3, in addition to the red target point cloud, there are some foreground obscurities (green and yellow) and background clutter (pink and blue). In step 4, the innovative scoring mechanism can well segment the target object.

fitted plane. In this paper, we solve the objective function by integrating the decomposition method named singular value decomposition (SVD) [24], and perform SVD on matrix A to obtain Eq. 5, where D is a diagonal matrix and U and V are unitary matrices. Since the diagonal elements of D are the singular values of matrix A, we assume that the last diagonal element is the minimum singular value. Therefore, if and only if Eq. 6 is true, $\|A\mathbf{n}\|$ has the minimum value, then $\mathbf{n}$ obtains the optimal solution (as in Eq. 7). Finally, model parameter d can be obtained by substituting $(\overline{x}, \overline{y}, \overline{z})$ into Eq. 2, also the plane model is allowed to have a fixed threshold error in the z-axis direction to complete ground point cloud segmentation.

$$\begin{cases} \|A\mathbf{n}\| = \left\|UDV^T\mathbf{n}\right\| = \left\|DV^T\mathbf{n}\right\| \\ \left\|V^T\mathbf{n}\right\| = \|\mathbf{n}\| = 1 \end{cases} \tag{5}$$

$$V^T\mathbf{n} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \Rightarrow \mathbf{n}^* = V \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{6}$$

$$\begin{aligned} \mathbf{n}^* &= \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = v_3 \\ \Rightarrow (a, b, c) &= \left(\mathbf{n}^*\right)^T = (v_{3,1}, v_{3,2}, v_{3,3}) \end{aligned} \tag{7}$$

*2) **3D point cloud clustering**:* In a 3D frustum where the ground points are removed, the objects are naturally separated in the physical space. The points belonging to the same object are clustered, and the point clouds belonging to different objects are discretely distributed. Therefore, comparing with the segmentation in the image where pixels from distant objects can be near-by to each other, the segmentation in the 3D point cloud is much more natural and much easier. Segmentation of different objects in 3D point cloud (as shown in Fig. 4) is a 3D point cloud clustering problem.

There is no limited clustering method in the framework of this paper. We can design the clustering method according to our own requirements. In this paper, we use Euclidean cluster extraction, and the search method of point cloud is kd-tree search. The implementation of clustering process can be found in the Point Cloud Library (PCL).

*3) **Scoring mechanism**:* 3D point cloud clustering has helped us to distinguish different objects in the square frustum region. We also need to pick out our target object from these objects. But this problem is not easy, because the frustum region includes not only our target object points, but also some occlusion object points and background clutter points. This paper uses some prior knowledge to design a novel scoring mechanism to grade each object point cloud and filter out the target object:

Firstly, since the heavily occluded object is difficult to detect, the 2D bounding box will only contain as much as possible the visible part of the target object, meaning that in the corresponding 3D frustum region, the target object is mainly in the foreground position, that is, the closer the separated object is to the lidar coordinate origin, the more likely it is the target object. We use the scoring method to express this prior knowledge (as in Eq. 8). In the equation, $\boldsymbol{C}_i$ represents the i-th object point cloud cluster in the frustum region, P^{ij} represents the j-th point in the i-th object point cloud cluster, lmax represents the farthest scanning distance of lidar, and S_{dist} represents the normalized distance score confidence. The closer S_{dist} is to 1.00, the more likely it is that $\boldsymbol{C}_i$ is the target object point cloud.

However, in general, the detection box not only includes the target object point cloud, but also may contain multiple occlusion objects and other foreground clutter (as shown in Fig. 4). The target object cannot be accurately determined by distance information alone. Other prior knowledge will be considered below.

$$S_{dist} = 1 - \frac{\sum_{j=0}^{|\boldsymbol{C}_i|-1} \left(\sqrt{{P_x^{ij}}^2 + {P_y^{ij}}^2} / l_{max} \right)}{|\boldsymbol{C}_i|} \tag{8}$$

Secondly, since the 2D bounding box will try its best to surround all the pixels belonging to the target object and avoid enclosing other noise objects, there will be relatively more points belonging to the target object in the corresponding frustum region. That is to say, the more the number of 3D points contained in the separated object, the more likely

it is the target object. We use Eq. 9 to represent this prior knowledge. In the equation, $\boldsymbol{P}_{in}$ represents all point clouds in the frustum region, and S_{num} represents the normalized quantity score confidence.

Considering both the distance score and the quantity score can solve the problem in most cases. However, when the occlusion is not far to the target object and the number of points of the occlusion is relatively large, the distance score and quantity score of this distracter would both be relatively high, which makes it easy to misjudge the target object.

$$S_{num} = \frac{|\boldsymbol{C}_i|}{|\boldsymbol{P}_{in}|} \tag{9}$$

Thirdly, since the 2D detection bounding box will enclose all visible pixels of the target object with the minimum frame, even when the occlusion is not far to the target object and the occlusion area is relatively large, it is impossible to completely block the target object, meaning the minimum 2D bounding box of each occlusion object does not have much overlap with the detection frame and it is generally less than 50%. That is, the closer the intersection over union (IoU) between the minimum bounding box of the pixels projected by the separated object and detection box is to 100%, the more likely it is the target object. We use Eq. 10 to represent this prior knowledge. In the equation, B_{rect} is the area of the 2D object detection bounding box, and $B_{\boldsymbol{C}_i\_proj}$ is the minimum bounding box area of the pixels projected by the separated object $\boldsymbol{C}_i$, and S_{IoU} represents the normalized IoU score confidence.

$$S_{IoU} = \frac{area\left(B_{rect} \cap B_{\boldsymbol{C}_i\_proj}\right)}{area\left(B_{rect} \cup B_{\boldsymbol{C}_i\_proj}\right)} \tag{10}$$

Finally, in order to consider the above three factors simultaneously, we use a simple fusion technique based on linear combination of the confidence scores of S_{dist}, S_{num} and S_{IoU} (As in Eq. 11 and Eq. 12). It is the complete scoring mechanism proposed in this paper. In the equation, w_1 and w_2 are the customized weights, representing the relative importance of the confidence degree of the corresponding score, and $\boldsymbol{C}^*$ is the point cloud of the target object we seek (as shown in Fig. 4).

$$S\left(\boldsymbol{C}_i\right) = S_{dist} + w_1 * S_{num} + w_2 * S_{IoU} \tag{11}$$

$$\boldsymbol{C}^* = argmax_{\boldsymbol{C}_i} S\left(\boldsymbol{C}_i\right) \tag{12}$$

What's more, the scoring mechanism in the framework of this paper is also very easy to be extended. We can improve it according to our own needs, add other factors that we think are useful, and we just need to imitate the above-mentioned score confidence form to express the consideration we want to add. In order to treat each consideration fairly, remember to normalize the expression of the score confidence.

IV. Experiments

We have tested our framework with the airport's freight scenario data. Our method is suitable for some special scenarios such as airport freight, because there are many objects in the airport that are not in the public data set, such as trailers, forklifts, etc. If you make the 3D object detection dataset of the lidar point cloud, then it will be a huge and expensive job. Secondly, the ground of the airport is relatively flat and the speed of the moving vehicles inside is relatively slow, which is suitable for 3D object detection based on multi-sensor fusion method.

A. Implementation

We have improved the original Euclidean cluster extraction method of PCL according to our own needs:

Firstly, since there are not too few points belonging to the detection object in the frustum region, we change the fixed minimum point threshold to a dynamic minimum point threshold, that is, the minimum threshold is set according to the total number of points included in the frustum region, so the noise clusters can be well judged and removed.In the actual experiment, we set the minimum threshold to 1/20 of the total number of the frustum points.

Secondly, since the data of the 3D lidar in the horizontal direction is dense, but the data in the vertical direction is really sparse, it is easy to cause the two lidar point lines scanned on the same distant object are wrongly divided into two classes due to their long vertical distance. We know that it is almost impossible to have multiple objects simultaneously in the vertical direction of one frustum region, so we conduct a certain proportion compression of z-axis data of the frustum point cloud to reduce the adverse effect of the sparsity of lidar point cloud data in the vertical direction when using Euclidean cluster extraction.We set the compression factor of z-axis data to 10.

In this framework, the cluster distance threshold of Euclidean distance clustering is preferably set to be between 0.5 and 1.0 m. If it is too large, it is easy to cluster different objects together. If it is too small, the point cloud of the same object will be split into different classes. In the scoring mechanism, we found that S_{IoU} is more important than the other two Score Confidence, so it is better to let $w_2 > w_1$, and the range of w_1 and w_2 are better in 1.0 and 2.0.

2D object detection uses the fast and accurate algorithm YOLOv3, which can achieve real-time speed on the Nvidia GTX 1060 with accuracy higher than 50mAP. In the 3D positioning process, we use the 2.6GHz Inter i7 CPU to achieve 10fps. In terms of accuracy, since we are using the original point cloud information, as long as the 3D target point cloud is successfully segmented, the real position is obtained.

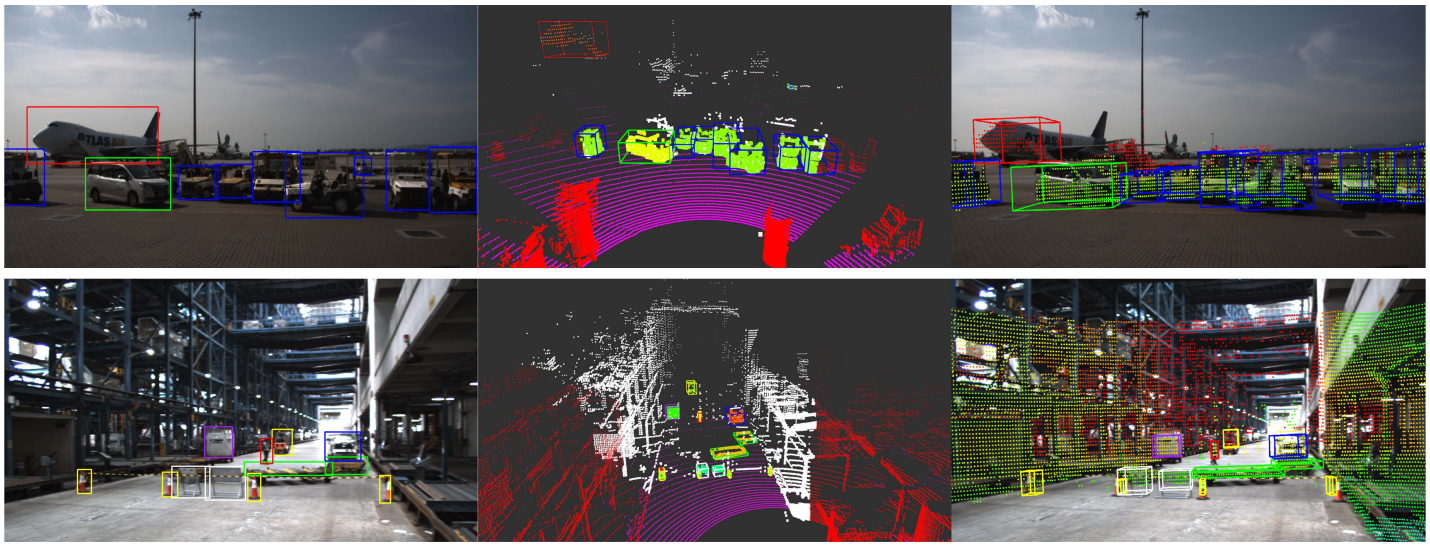

Fig. 5. **3D object detection results in Hong Kong Airport.** The upper part is an outdoor scene, and the lower part is a indoor scene both with complex lighting conditions and using the 64-line lidar. The 2D object detection results based on deep learning training (left), the 3D object segmentation results based on the scoring mechanism (middle), and the results of 3D object detection in the fusion image (right) are shown. Among them, forklifts (yellow), trailers (blue), cargo boxes (purple) and other objects are not found in the public datasets.

B. Result

The objects in the actual scene are often diverse and non-standard, such as modified trailers in the airport, randomly shaped goods and traffic signs, etc. In such scenarios, the dynamic is higher and the environment is more complicated. Therefore, the dataset often cannot cover the needs of the actual application. It is a really expensive and inefficient way to make the 3D point cloud dataset separately of each object, and it also has a negative impact on the universality of the algorithm. And traditional methods rely heavily on datasets, which are poorly scalable in complex environments and difficult to meet actual needs. Therefore, we change the way of thinking, do not rely heavily on the dataset, but only through the easily available 2D image dataset, do simple training, then we can achieve 3D object detection.

The results (as shown in Fig. 5) of different illumination conditions, such as indoor and outdoor, are given in the experiment. Those prove the applicability, robustness and effectiveness of our algorithm in practical applications. Especially for the 3D detection of new type objects, the existing method is not able to achieve satisfactory results in a short period of time.

In the future work, we will compare our method with existing methods in datasets and actual scenarios, and further explore the method of lidar-vision depth fusion.

V. Conclusion

For the lack of large number of expensive 3D lidar point cloud datasets for 3D object detection, we have proposed a 2D-driven 3D object detection framework, using both the camera and the 3D lidar data, to perform accurate 3D localization. The proposed approach can quickly achieve 3D object detection for almost any kind of object in large scene applications, since we only require 2D image labels. Thanks to the proposed novel scoring mechanism, the real-time performance of 3D object point cloud segmentation in the 3D frustum can be ensured even only using the CPU. Furthermore, the proposed approach is robust to sparsity lidar data and also effective even using the 16-line lidar. Practical experiments in real large scene applications at Hong Kong Airport validate the effectiveness of the proposed 3D detection approach, which also show that the proposed framework is easy to be extended and improved according to the actual application scene.

References

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems 28* (C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds.), pp. 91–99, Curran Associates, Inc., 2015.

[2] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018.

[3] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[4] C. R. Qi, H. Su, M. Niessner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[5] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[6] B. Li, T. Zhang, and T. Xia, "Vehicle detection from 3d lidar using fully convolutional network," *CoRR*, vol. abs/1608.07916, 2016.

[7] B. Li, "3d fully convolutional network for vehicle detection in point cloud," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1513–1518, Sep. 2017.

[8] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[9] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[10] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *CoRR*, vol. abs/1801.07829, 2018.

[11] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[12] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[13] D. Z. Wang and I. Posner, "Voting for voting in online point cloud object detection.," in *Robotics: Science and Systems*, vol. 1, pp. 10–15607, 2015.

[14] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks," *CoRR*, vol. abs/1609.06666, 2016.

[15] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *Computer Vision – ECCV 2012* (A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, eds.), (Berlin, Heidelberg), pp. 746–760, Springer Berlin Heidelberg, 2012.

[16] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[17] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," *CoRR*, vol. abs/1712.02294, 2017.

[18] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3d object detection," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[19] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," in *Advances in Neural Information Processing Systems 28* (C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds.), pp. 424–432, Curran Associates, Inc., 2015.

[20] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

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

[22] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *International Journal Of Computer Vision*, vol. 81, pp. 155–166, 2009.

[23] D. Zermas, I. Izzat, and N. Papanikolopoulos, "Fast segmentation of 3d point clouds: A paradigm on lidar data for autonomous vehicle applications," in *IEEE International Conference on Robotics and Automation*, 2017.

[24] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," in *Linear Algebra*, pp. 134–151, Springer, 1971.