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

# Cascaded Sliding Window Based Real-Time 3D Region Proposal for Pedestrian Detection*

Jun Hu, Tao Wu†, Hao Fu and Zhiyu Wang
*College of Intelligence Science and Technology*
*National University of Defense Technology*
*Changsha, Hunan Province, China*
{*hujun17,wutao,fuhao,wangzhiyu09a*}*@nudt.edu.cn*

Kai Ding†
*Science and Technology on Near-Surface Detection Laboratory*
*Wuxi, Jiangsu Province, China*
*winfast113@sina.com*

***Abstract*— Pedestrian detection is an indispensable technology for designing autonomous driving systems. This paper proposes a cascaded sliding window based real-time approach to generate 3D pedestrian proposals on point clouds only. After rasterizing the raw point cloud, a 3D sliding window is adopted to extract pedestrian candidates. Two features are proposed to improve the proposal performance. One is the central points density feature, which acts as a filter to speed up the process and reduce false alarms. The other is the location feature, including the density distribution and height difference distribution of the point cloud, which describes the profile and location of an object in a sliding window. The scores generated by this feature are then applied to non-maximum suppression (NMS) to remove sub-optimal boxes. Experiments on the KITTI 3D object detection benchmark show that our approach achieves state-of-the-art results among comparable methods, while maintaining the speed and accuracy trade-off.**

***Index Terms*— Pedestrian detection, Sliding window, Region proposal, Point clouds.**

## I. Introduction

Pedestrian detection is an important module for the autonomous vehicles. Compared to camera images, LiDAR is becoming more popular due to its ability for generating highly accurate three-dimensional information. However, due to the sparsity and irregularity of point cloud data, LiDAR-based pedestrian detection is still a challenging task.

In recent years, the deep learning based method has become the mainstream of 3D pedestrian detection with the development of neural network algorithms. Most of these deep learning based approaches firstly convert the unordered point clouds into ordered forms, and then adopt a two-stage strategy or a single-shot strategy to generate bounding boxes for target detection.

Although the deep learning based methods have achieved high detection accuracy. These methods are usually very computationally expensive, and require specific computational devices such as GPUs, hindering their practical application. In the field of autonomous driving, the algorithm needs to get a balance between accuracy and efficiency.

To address the limitations mentioned above, we propose in this paper a cascaded sliding window based algorithm which takes point cloud as input and generate 3D region proposals for pedestrian detection in real-time. As the pedestrian target owning a small aspect ratio and a relatively concentrated point cloud distribution, the sliding window algorithm is very suitable for extracting pedestrian candidates in 3D space.

An overview of the proposed approach is shown in Fig. 1. Instead of projecting the point cloud into a lower dimensional space, we firstly rasterize the point cloud on the $x$-$y$ plane of the LiDAR coordinate system. The original information for each point is preserved. Then a 3D sliding window is adopted on the $x$-$y$ plane to generate the proposals. To speed up the sliding process and reduce false alarms, we propose two new types of features, the central points density feature and the location feature, for each sliding bounding box. These two types of features perform as a filter and a coarse classifier, respectively, to reject false positives in the early stage. After this filtering process and a subsequent non-maximum suppression (NMS) stage, more complex features are then computed form each candidate. A fine classifier adopting AdaBoost then performs on these features to make the final decision.

We evaluate our results on the bird's eye view (BEV) detection and 3D detection tasks on the KITTI object detection benchmark [1]. Experiments show that our approach significantly outperforms deep learning based methods such as AVOD [2] and Complexer-YOLO [3], although our algorithm is based on traditional models. Furthermore, our method achieves a good trade-off between efficiency and accuracy, with the calculation speed 50% faster than the deep learning based algorithms.

The rest of this paper is organized as follows. In Section II, we introduce some related works. Section III provides an overview of the proposed approach. In Section IV, experimental results are presented. Finally, the conclusions are summarized in Section V.

*This work is supported by the Foundation of Science and Technology on Near-Surface Detection Laboratory under No.6142414180207.

†Corresponding Authors.

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

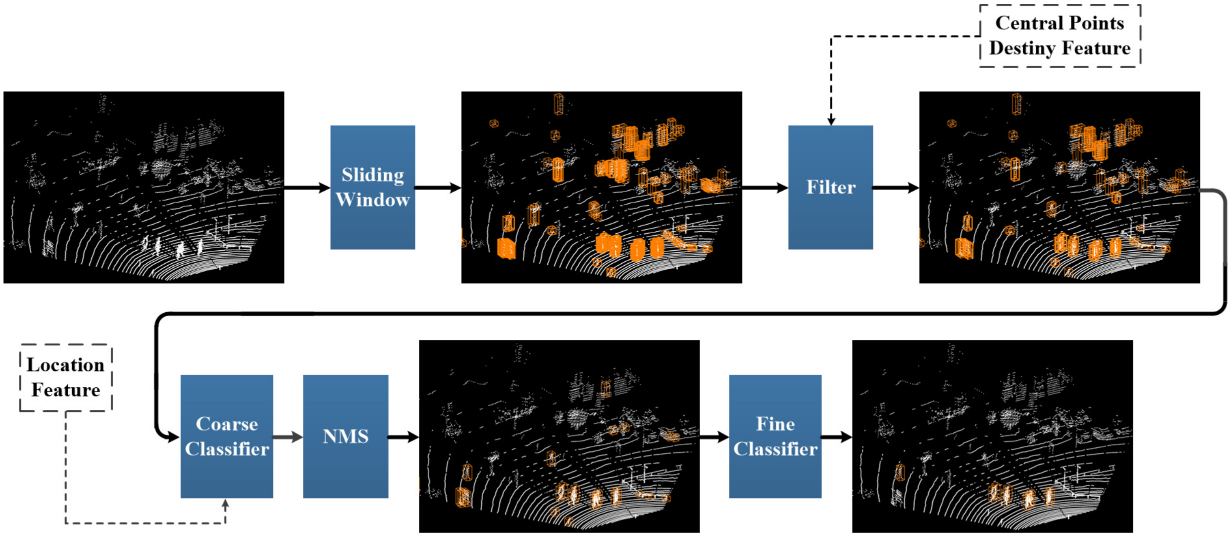


Fig. 1. Overview of the proposed approach. Original proposals are firstly generated from the point cloud by using the sliding window algorithm. To speed up the sliding process and reduce false alarms, we propose two new types of features, the central points density feature and the location feature. The central points density feature performs as a filter to quickly remove the false positives. Then the location feature is used to train a coarse classifier and score for each sliding bounding box. After this these process and a subsequent non-maximum suppression (NMS) stage, more complex features are then computed form each candidate. A fine classifier then performs on these more complex features to make the final decision.

## II. Related work

Currently, the mainstream 3D object detectors are deep learning-based approaches. One type of methods projects the point cloud onto a 2D plane, or uses a voxelized representation of the point cloud. BirdNet [4] projected the 3D point cloud to the BEV and then adopted 2D CNN to estimate the confidence and bounding boxes of the object. VeloFCN [5] projected the point cloud to the front-view depth image to apply a fully convolutional network (FCN) on it. AVOD [2] and MV3D [6] proposed networks which combine 2D LIDAR views and RBG image for 3D object detection. The authors of [7, 8] represent the points with voxel grid and used 3D CNN [9] to learn the features and generate 3D boxes. However, the lower-dimensional projection and voxelization usually suffer from information loss, and the computation of 3D CNN is very expensive.

Another line of work is the Frustum-based approaches. F-PointNet [10] adopted a 2D CNN detector to generate 2D proposals from the RGB images, and then generates frustums for each proposal to group target points. PointNet [11, 12] is then utilized to directly deals with the point cloud for 3D box estimation. However, the drawback is that the detection accuracy of this method depends on the quality of the image-based proposal generation. In addition, the computational cost of this method is large due to the need to run two neural network methods. In general, although deep learning-based approaches have achieved high accuracy, it involves a high computational cost and always rely on specific computational devices, such as a GPU.

Before the deep learning method is widely used, hand-crafted features are often used to generate 3D proposals [13, 14]. Designing simple and effective features in this step could significantly reduce the computation at load. Inspired by their works and based on the features of pedestrian targets, in this paper, we propose a cascaded sliding window algorithm to generate 3D proposals from point clouds, with an aim to achieve a good balance between efficiency and accuracy.

## III. The Proposed Approach

### A. Proposal Generation

The input raw point cloud data is firstly discretized into a 2.5D grid map on the $x$-$y$ plane with a fixed resolution of 0.1 m×0.1 m. The height information for each point is stored in the corresponding grid. We restrict the grid map to 50 m in front of the vehicle and 25 m in each side. The size of a sliding window is set to 0.7 m×0.7 m. The window is initially placed in the upper left corner of the grid map and traverses the entire search area with a step size of 0.1 m.

### B. Filter Based on the Central Points Density Feature

Due to the large number of initial proposals, the generated bounding boxes would be firstly filtered to speed up the sliding process.

We propose a central points density feature for filtering. The definition of the feature is shown in Fig. 2. The left column shows an extracted object candidate. The outermost bounding box denotes a sliding window with a length and width of 0.7 m. Points in the sliding window is firstly projected onto the $x$-$y$ plane. Since the sliding window

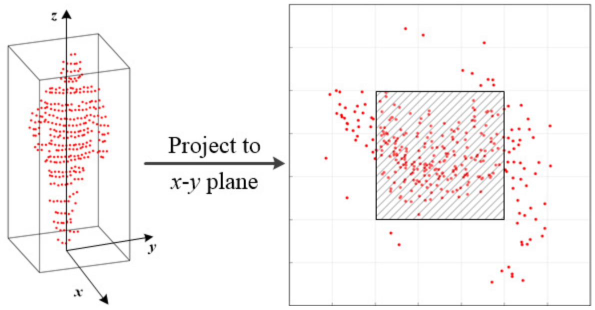


Fig. 2. Definition of the proposed central points density feature.

consists of 7×7 grids with a side length of 0.1 m, the points are then divided into these grids. The central points density feature represents the ratio of the sum of the points falling in the middle 3×3 grids to all points in the sliding window, as shown in the shaded area in the right column.

Let $F$ be the value of central points density feature, it is defined as:

$$F = \frac{\sum n_{ij}}{N}, \quad i, j \in \{3, ..., 5\}, \tag{1}$$

where $N$ represents the number of points in the sliding window. $n_{ij}$ denotes the number of points falling in the corresponding grid.

Meanwhile, we use the following additional rules for filtering: (1) The center grid of a sliding window should not be empty. (2) The difference between the highest point and the lowest point in a sliding window should be within a reasonable range. Bounding boxes that do not meet these criteria will be filtered out.

$$\begin{cases} N_{center} > 0, \\ 0.5 < \Delta h < 2, \\ F > 0.35. \end{cases} \tag{2}$$

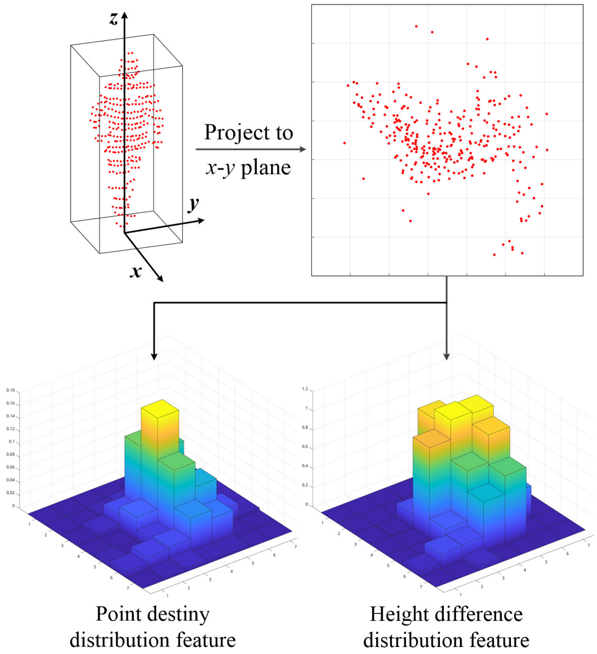
where $N_{center}$ represents the number of points in the center grid of the sliding window. $\Delta h$ denotes the difference between the highest point and the lowest point in a proposal. $F$ is the value of the central points density feature.

### C. Coarse Classifier Based on the Location Feature

To further speed up the process, it is necessary to use a simpler feature to quickly locate the target. In this step, we design a coarse classifier to quickly generate a score for each sliding bounding box.

We assume that most pedestrians are upright, and an ideal bounding box should keep the target in its center. Besides, the extracted point cloud should be complete and avoid including irrelevant points around.

Therefore, we propose a location feature, including the density distribution and height difference distribution of the point cloud, to evaluate the object proposal. Definition of the feature is shown in Fig. 3. Similar to the center point density feature, the points in the sliding window are firstly projected onto the $x$-$y$ plane and divided into corresponding 7×7 grids. Then, a normalized histogram of the number of points as well as a histogram of the height difference between the highest point and the lowest point in each grid is calculated.



Point destiny distribution feature

Height difference distribution feature

Fig. 3. Definition of the proposed location feature.

Let $D_{ij}$ and $\Delta H_{ij}$ be the density and height differences for the points in each grid, the definitions are:

$$\begin{cases} D_{ij} = \frac{n_{ij}}{N}, \quad i, j \in \{1, ..., 7\}, \\ \Delta H_{ij} = h_{ij_{max}} - h_{ij_{min}}, \quad i, j \in \{1, ..., 7\}. \end{cases} \tag{3}$$

where $N$ represents the number of points in the sliding window. $n_{ij}$ denotes the number of points falling in the corresponding grid. $h_{ij_{max}}$ and $h_{ij_{min}}$ indicate the height difference between the highest point and the lowest point in the corresponding grid.

We expect the target to be in the center of the sliding window. This goal can be considered as an anomaly detection problem [15], that is, treating the targets at the center position as positive samples. For test samples, the closer they are to the center of the sliding window, the higher their score. Therefore, we then compress the two histograms into a one-dimensional feature vector and feed it to an one-class support vector machine (SVM) [16] classifier to generate the score, which would be used in the NMS.

### D. Non-Maximum Suppression

NMS usually follows a greedy strategy which uses the detection score to evaluate proposals. For bounding boxes generated from the same target, only the one with the highest score will be retained. However, using a complex classifier to calculate the score for each candidate requires a large amount of computation. Therefore, the strategy of our NMS is as follows: we consider two proposals whose distance is less than a threshold $\theta$ to belong to the same target. We then select candidates based on the scores generated by the proposed location feature, and only the one with the highest score will be retained. Here we set the $\theta$ to 0.3 m.

### *E. Fine Classifier Based on Sophisticated Features*

In the final step, we extract more sophisticated features to determine if the candidate is a pedestrian. We choose to compute seven kinds of features from the point cloud of each candidate and apply AdaBoost to learn the classifier. The description of the features is shown in Table I.

TABLE I

DESCRIPTION OF FEATURES.

| **No.** | **Description** | **Dimension** |
|---|---|---|
| $f_1$ | Number of points | 1 |
| $f_2$ | Distance to the object | 1 |
| $f_3$ | Maximum height difference | 1 |
| $f_4$ | Three-dimensional covariance matrix | 6 |
| $f_5$ | Three-dimensional covariance matrix eigenvalue | 3 |
| $f_6$ | The normalized moment of inertia tensor | 6 |
| $f_7$ | Rotational projection statistics | 135 |

Features $f_1$, $f_2$ , and $f_3$ [17] describe the geometric properties of the point cloud, including the number of points, the distance from the vehicle, and the height difference between the highest point and the lowest point in the $z$ direction.

Features $f_4$ and $f_5$ are the three-dimensional covariance matrix $C$ and its eigenvalues. The covariance matrix is composed of six independent vectors, and the eigenvalues are arranged in descending order. The matrix $C$ is defined as:

$$C = \begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix} . \tag{4}$$

where

$$cov(x,y) = \frac{\sum_{n=1}^{N} (x_i - \overline{x})(y_i - \overline{y})}{n-1}. \tag{5}$$

$f_6$ is an inertia tensor matrix $I$ [18]. It describes the overall distribution of the point cloud. The matrix $I$ is defined as:

$$I = \begin{pmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{pmatrix} . \tag{6}$$

where

$$\begin{cases} I_{xx} = \sum_{n=1}^{N} (y_i^2 + z_i^2), \\ I_{xy} = I_{yx} = -\sum_{n=1}^{N} x_i y_i. \end{cases} \tag{7}$$

The $x, y, z$ above represent the 3D coordinates of each point, and $N$ denotes the number of points of the point cloud.

Feature $f_7$ describes the rotational projection statistics [19], which projects 3D point cloud into lower dimensional space and calculate statistics for each view.

## IV. EXPERIMENTAL RESULTS

We evaluate our approach on the KITTI 3D object detection benchmark [1], which consists of 7,481 training frames and 7,518 test frames as well as the corresponding Velodyne 64E point cloud data. We follow [6] to split the training data into a training set (3712 frames) and a validation set (3769 frames). We focus our experiments on the pedestrian category and compare our approach with state-of-the-art methods. The models are all trained on training split and evaluated on the test split and the validation split. Our experimental platform is a laptop equipped with a quad-core 2.3GHz Intel i5 CPU and 8GB of RAM.

### *A. Evaluation of Detection*

Firstly, our approach is evaluated on the task of 3D detection and BEV detection on the KITTI's official test server. The results are calculated according to the easy, moderate, and hard difficulty levels provided by KITTI. As shown in Table II, we significantly outperform previous state-of-the-art methods. Among them, AVOD [2] and Complexer-YOLO [3] utilize both the point clouds and RGB image. BirdNet [4] and TopNet-HighRes [20] are the LiDAR-only methods using convolutional neural networks (CNNs). Our approach, which is based on traditional models and only takes point clouds as input, achieves more competitive results than AVOD [2] in BEV detection and outperforms the other methods by large margins on all difficulty levels in both the two tasks. Besides, our approach only requires about 0.026s runtime per frame on a quad-core CPU. It is more than twice as fast as AVOD [2] and Complexer-YOLO [3] and four times faster than BirdNet [4].

In addition, the recall at different Intersection over Union (IoU) threshold using less than 100 proposals is investigated, as shown in Fig. 4. Results show that the performance of our method drops significantly at a threshold of 0.7. The reason is partly because we use a fixed-size sliding window a fixed step size, it may result in inaccurate detection. However, when using IoU of 0.5, our method is able to achieve a recall of more than 70% on moderate instances.

TABLE II

Evaluation on KITTI test set for pedestrians. Results are generated by KITTI's official test server.

| Method | 3D detection AP(%) | | | BEV detection AP(%) | | | Time(s) |
|---|---|---|---|---|---|---|---|
| | Easy | Moderate | Hard | Easy | Moderate | Hard | |
| AVOD [2] | **36.10** | **27.86** | **25.76** | 42.58 | 33.57 | 30.14 | 0.08 |
| Complexer-YOLO [3] | 17.60 | 13.96 | 12.70 | 21.42 | 18.26 | 17.06 | 0.06 |
| BirdNet [4] | 12.25 | 8.99 | 8.06 | 20.73 | 15.80 | 14.59 | 0.11 |
| TopNet-HighRes [20] | 10.40 | 6.92 | 6.63 | 19.43 | 13.50 | 11.93 | 0.10 |
| Ours | 33.75 | 26.64 | 23.34 | **49.27** | **37.96** | **33.83** | **0.026** |

TABLE III

Performance on KITTI validation set for pedestrians by adopting different modules.

| Filter | NMS | | 3D detection AP(%) | | | BEV detection AP(%) | | | Time(s) |
|---|---|---|---|---|---|---|---|---|---|
| | Location Feature | Classifier Score | Easy | Moderate | Hard | Easy | Moderate | Hard | |
| ✓ | ✓ | | 47.54 | 43.72 | 37.49 | 65.03 | **59.15** | 51.38 | **0.026** |
| ✓ | | ✓ | **49.77** | **45.53** | **39.20** | **65.30** | 59.07 | **51.46** | 0.039 |
| | ✓ | | 47.41 | 43.66 | 37.47 | 64.93 | 58.95 | 51.30 | 0.032 |
| | | ✓ | 49.40 | 45.18 | 38.78 | 64.25 | 58.29 | 50.57 | 0.096 |

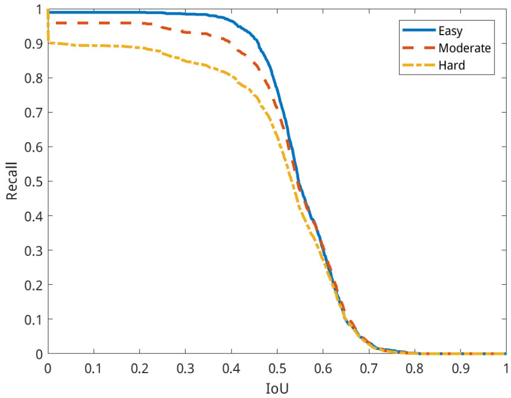


Fig. 4. Recall at different IoU thresholds on KITTI validation set.

### B. Ablation Studies

In order to investigate the effectiveness of different components of the proposed approach, ablation experiments are then conducted. All experiments are evaluated on the validation split and we use the official 3D IoU evaluation metrics of 0.5 for the category of pedestrian.

The effects of the proposed filter and location feature is firstly investigated. In NMS, we compared the differences in the scores generated using the proposed location features or using the final classifier. The influences of each module to the detection performance is analysed by only removing the specific part and keeping all other parts unchanged. The results are listed in Table III. It can be seen that by adopting our proposed filter, not only the processing time is significantly reduced (from 96 ms to 39 ms and from 32 ms to 26 ms), but also the detection performance is improved. This result demonstrates that the proposed filter is effective for accurately filtering out non-pedestrian proposals. Furthermore, generating scores for proposals by the location feature in NMS could further reduces computation time while maintaining similar performance, as compared to directly using the final classifier.

### C. Qualitative Analysis

Some qualitative results of our proposed approach on the test set are shown in Fig. 5. The 3D bounding boxes are projected to the image for better visualization. Results show that our approach still performs well in challenging scenarios. By setting the appropriate sliding window size and NMS distance threshold, our approach is able to distinguish people close to each other and is robust to slightly occluded targets. Note that our method currently only outputs fixed-size non-oriented bounding boxes, which would negatively affect the accuracy of the detection. However, considering that the aspect ratio of pedestrian targets is relatively small, this strategy is still acceptable.

## V. Conclusion

In this work, we propose a cascaded sliding window based algorithm which could generate 3D proposals directly from point cloud in real-time. Two effective features are proposed for to reduce runtime. Experimental results demonstrate that our approach shows competitive performance on KITTI benchmarks [1] in terms of runtime and accuracy, compared to previous state-of-the-art methods.

In future work, we plan to improve our approach by introducing estimates of oriented bounding boxes to make the localization of the target more accurate. In addition, we intend

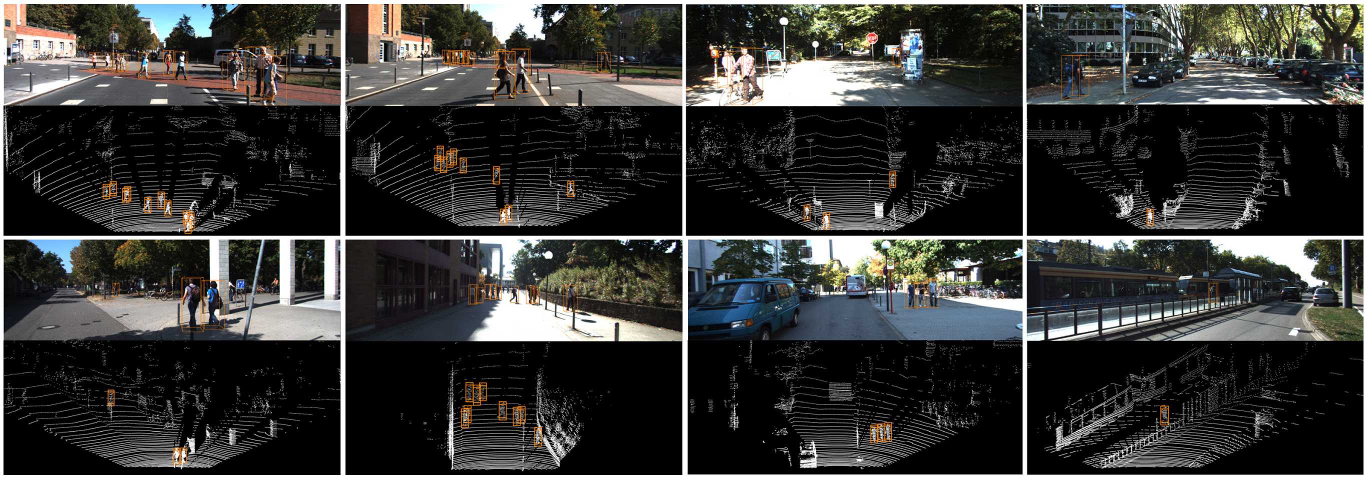


Fig. 5. Visualization of our results on the KITTI test set. The detected pedestrians are shown with orange 3D bounding boxes in the LiDAR view. The 3D bounding boxes are projected onto the corresponding image in the upper row.

to improve the classifier for better detection performance, such as adopting more discriminative features, or combining with lightweight neural networks.

## References

[1] A. Geiger, P. Lenz and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, 2012, pp. 3354-3361.

[2] J. Ku, M. Mozifian, J. Lee, A. Harakeh and S. L. Waslander, "Joint 3D Proposal Generation and Object Detection from View Aggregation," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, 2018, pp. 1-8.

[3] M. Simon, K. Amende, A. Kraus, J. Honer, T. Samann, H. Kaulbersch, S. Milz and H. Michael Gross, "Complexer-YOLO: Real-Time 3D Object Detection and Tracking on Semantic Point Clouds," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops 2019.

[4] J. Beltrán, C. Guindel, F. M. Moreno, D. Cruzado, F. García and A. De La Escalera, "BirdNet: A 3D Object Detection Framework from LiDAR Information," 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, 2018, pp. 3517-3523.

[5] B. Li, T. Zhang and T. Xia, "Vehicle detection from 3D lidar using fully convolutional network," Robotics: Science and Systems XII, AnnArbor, Michigan, USA, 2016, pp. 42.

[6] X. Chen, H. Ma, J. Wan, B. Li and T. Xia, "Multi-view 3D Object Detection Network for Autonomous Driving," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6526-6534.

[7] 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," 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 2017, pp. 1355-1361.

[8] S. Song and J. Xiao, "Deep Sliding Shapes for Amodal 3D Object Detection in RGB-D Images," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 808-816.

[9] D. Tran, L. Bourdev, R. Fergus, L. Torresani and M. Paluri, "Learning Spatiotemporal Features with 3D Convolutional Networks," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 4489-4497.

[10] C. R. Qi, W. Liu, C. Wu, H. Su and L. J. Guibas, "Frustum PointNets for 3D Object Detection from RGB-D Data," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018, pp. 918-927.

[11] R. Q. Charles, H. Su, M. Kaichun and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 77-85.

[12] C. R. Qi, L. Yi, H. Su and L. J. Guibas, "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, 2017, pp. 5105-5114.

[13] X. Chen, K. Kundu, Y. Zhu, A. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," in Neural Information Processing Systems, Montreal, Quebec, Canada, 2015, pp.424-432.

[14] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler and R. Urtasun, "Monocular 3D Object Detection for Autonomous Driving," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 2147-2156.

[15] K. Li, H. Huang, S. Tian and W. Xu, "Improving one-class SVM for anomaly detection," Proceedings of the 2003 International Conference on Machine Learning and Cybernetics, Xi'an, China, 2003, pp. 3077-3081.

[16] B. Scholkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola and R. C. Williamson, "Estimating the Support of a High-Dimensional Distribution," in Neural Computation, vol. 13, no. 7, pp. 1443-1471, July 2001.

[17] C. Premebida, O. Ludwig and U. Nunes, "Exploiting LIDAR-based features on pedestrian detection in urban scenarios," 2009 12th International IEEE Conference on Intelligent Transportation Systems, St. Louis, MO, 2009, pp. 1-6.

[18] L. E.Navarro-Serment, C. Mertz and M. Hebert, "Pedestrian detection and tracking using three-dimensional LADAR data," The International Journal of Robotics Research, vol. 29, pp. 1516-1528, 2010.

[19] Y. Guo, F. Sohel, M. Bennamoun, M. Lu and J. Wan, "Rotational projection statistics for 3D local surface description and object recognition," International Journal of Computer Vision, vol. 105, no. 1, pp. 63-86, Oct. 2013.

[20] S. Wirges, T. Fischer, C. Stiller and J. B. Frias, "Object Detection and Classification in Occupancy Grid Maps Using Deep Convolutional Networks," 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, 2018, pp. 3530-3535.