Automatic 3D Object Detection from RGB-D Data Using PU-GAN

Xueqing Wang[1], Ya-Li Hou[1](✉), Xiaoli Hao[1], Yan Shen[1], and Shuai Liu[2]

[1] School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing, China
ylhou@bjtu.edu.cn

[2] School of Engineering, Honghe University, Mengzi, China

Abstract. 3D object detection from RGB-D data in outdoor scenes is crucial in various industrial applications such as autonomous driving, robotics, etc. However, the points obtained from range sensor scans are usually sparse and non-uniform, which seriously limit the detection performance, especially for far-away objects. By learning a rich variety of point distributions from the latent space, we believe that 3D upsampling techniques may fill up the missing knowledge due to the sparsity of the 3D points. Hence, a 3D object detection method using 3D upsampling techniques has been presented in this paper. The main contributions of the paper are two-fold. First, based on the Frustum PointNets pipeline, a 3D object detection method using PU-GAN has been implemented. A state-of-the-art 3D upsampling method, PU-GAN, is used to complement the sparsity of point cloud. Second, some effective strategies have been proposed to improve the detection performance using upsampled dense points. Extensive experimental results on KITTI benchmark show that the impacts of PU-GAN upsampled points on object detection are closely related to the object distances from the camera. They show their superiority when they are applied on objects located at around 30 meters away. By carefully designing the criteria to employ the upsampled points, the developed method can outperform the baseline Frustum PointNets by a large margin for all the categories, including car, pedestrian and cyclist objects.

Keywords: 3D object detection · RGB-D data · PU-GAN

1 Introduction

3D object detection from RGB-D data in outdoor scenes is crucial in various industrial applications such as autonomous driving, robotics, etc. Given RGB-D data as input, 3D object detection is to classify and localize objects, estimate their size and orientations in 3D space. The RGB data are usually 2D images from a camera and the depth data is usually obtained from LiDAR in outdoor scenarios and represented as a point cloud.

Frustum PointNets [1] leverage mature 2D object detectors and directly operate on raw point clouds, which has achieved remarkable performance and become a baseline of 3D detection using RGB-D data. However, the sparsity and non-uniformity of raw point clouds from range sensor scans is still a big headache especially for far-away object

G. Bebis et al. (Eds.): ISVC 2020, LNCS 12510, pp. 742–752, 2020.
https://doi.org/10.1007/978-3-030-64559-5_59

detections. Without sufficient information about the objects of interest, few 3D points often cause inaccurate pose and size estimation.

In recent years, 3D upsampling methods have achieved great progress with the development of deep learning techniques. By learning a rich variety of point distributions from the latent space, we believe that upsampled points may complement the point cloud from the range sensors and finally lead to better object detection performance. Hence, 3D object detection with RGB-D data using 3D upsampling techniques has been studied in this paper. PU-GAN [2] has been used as an example since it is a state-of-the-art work for 3D upsampling and it can achieve a better distribution uniformity and 3D geometry reconstruction quality.

2 Related Work

3D object detection has made great progress thanks to the deep learning, as a fundamental part of 3D scene perception. It is crucial in various industrial applications such as autonomous driving, robotics and augmented reality. Based on the ways to represent RGB-D data, the approaches can be classified into three categories: Front-view image based methods, bird's eye view based methods and 3D based methods.

Front-View Image Based Methods. [3, 4] infer 3D bounding box by combining monocular RGB images and shape priors. [5, 6] use 2D CNN network to extract feature form images for 3D bounding box regression. FVNet [7] first project point clouds onto a cylindrical surface to generate front-view feature maps which retains rich information, and then introduce a proposal generation network to predict 3D region proposals from the generated maps and further extrude objects of interest from the whole point cloud.

Bird's Eye View Based Methods. MV3D [8] achieves relatively high 3D object detection accuracy by fusing LiDAR projections from different perspective. VoxelNet [9] use voxel feature encoding (VFE) layer to extract learning-based features through each voxel, and then the features are stacked as bird's eye view feature maps. Finally, 2D region proposal network (RPN [10]) is trained to complete the 3D detection. SECOND [11] proposes an improved method of sparse convolution [12] for voxel based object detection to increases the speed of training and inference. PointPillars [13] voxelize the whole point cloud into vertical columns (pillars), then encoded features from each pillar are mapped as 2D bird's eye view image for 3D object detection.

3D Based Methods. Frustum PointNets [1] first uses 2D detector to generate 2D box proposals, then frustums are extracted by lifting 2D regions to 3D space, and a PointNet++ [14] framework is used to complete the prediction of 3D box. [15–18] convert image-based depth maps to pseudo LiDAR representations, and then 3D based methods are used to finish the 3D object detection. STD [19] first uses a proposal generation module (PGM) to generate proposals from point-based spherical anchors. Then a PointsPool layer is used to convert proposal features from sparse expression to compact representation.

Although the fusion of RGB images and point data has achieved great success, the sparsity and non-uniformity of point cloud is still a main challenge of 3D object detection. Insufficient 3D points often cause inaccurate pose and size estimation.

3 The Method

The framework of the developed object detection method using PU-GAN is shown in Fig. 1. As a state-of-the-art work of 3D object detection from RGB-D data, the baseline architecture of Frustum PointNets [1] has been followed. A 2D object detector is used to identify the possible object candidates and greatly reduce the search space of point cloud-based 3D object detection. Based on the monocular vision, the camera projection matrix is used to lift 2D proposals to 3D space. In the 3D frustum formed by each proposal, PointNet++ [14] is used to predict the object types, sizes, positions and orientations. At the output, each 3D object is represented by a category label t, a 3D box parameterized by its size h; w; l, its center cx; cy; cz, and the heading angle θ around the up-axis for orientation.

Although the fusion of RGB images and point data for 3D object detection has achieved some remarkable progress, the sparsity of points is still a serious limitation of the detection performance, especially for small objects, like pedestrians in the street. In the KITTI dataset [20], there are less than 50 points for a pedestrian at 30 meters away from the camera and only around 10 points at 60 meters away. Without sufficient 3D points, it is difficult to identify a pedestrian with a high confidence. Motivated by the development of data-driven 3D upsampling techniques in recent years, we believe that 3D upsampling may be able to complement the sparsity of the 3D points by learning various point distributions from a variety of objects. PU-GAN [2] is a recently proposed 3D upsampling method and shows good 3D geometry reconstruction quality even in large scale scenarios. In the developed method, PU-GAN generated dense points are used for 3D object detection. To effectively use the upsampled points, some criteria have been carefully designed to classify the 2D proposals into easy and hard samples. The upsampled points are only used for those difficult to be identified by raw Lidar points.

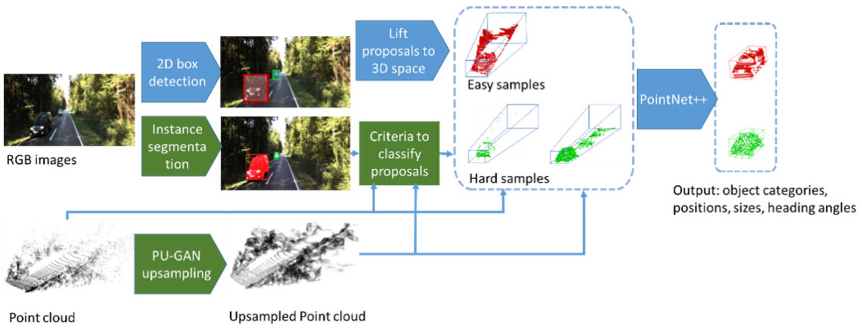

Fig. 1. The proposed framework for 3D object detection using PU-GAN.

PU-GAN Upsampling. To remedy the sparsity of the original lidar points, 3D upsampling techniques are used in the developed method. In recent years, data-driven upsampling approaches have achieved remarkable progress, such as PU-Net [21], EC-Net [22], MPU [23] etc. In our implementations, PU-GAN [2] is used as an example since it can achieve better uniformity of point distribution and better proximity-to-surface.

Given an original sparse LiDAR point set $P_O = \{X_i, Y_i, Z_i, R_i\}_{i=1}^{M}$ of M points, PU-GAN aims to produce uniformly-distributed points which can well preserve the underlying geometry of the original object. After the upsampling, a dense point set $P_G = \{X_i, Y_i, Z_i, R_i\}_{i=1}^{rM}$ of rM points can be obtained.

In PU-GAN, the whole process follows the patch-based strategies. S seeds are first picked using the farthest sampling from P_O and a local patch with 256 points around each seed is extracted. Each patch is normalized and fed into a generator, which is guided by a discriminator to generate uniformly distributed dense points. Finally, the upsampled patches are merged into the final output P_G. S is an important parameter for 3D upsampling in a large scale scenario. We have tried different numbers of seeds and use 800 in all our experiments. As shown in Fig. 2, when a small number of seeds is used, the dense points may not be uniform because patches cannot cover the entire scene and only a portion of Lidar points have been upsampled. When the seed number S is 800, the extracted patches can span the entire space and the generated dense points tend to be uniformly distributed in 3D space.

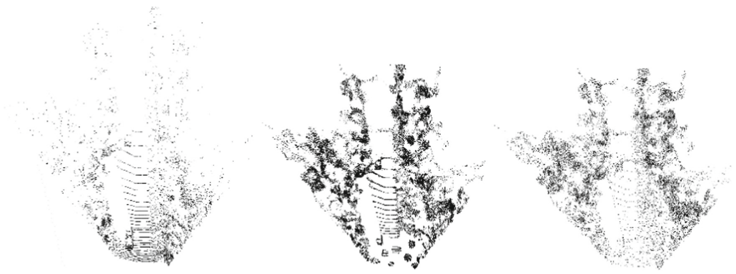

Fig. 2. Upsampled **point cloud data of KITTI dataset from the bird's eye view**. Left: a raw point cloud from KITTI street scene acquired by LiDAR. Middle: upsampled point cloud by PU-GAN with 200 seeds. Right: upsampled point cloud by PU-GAN with 800 seeds.

Criteria to Classify Proposals. A simple way to use PU-GAN for object detection may be to replace all the point clouds from LiDAR with upsampled dense points in the Frustum PointNets architecture. However, the results in our experimental sections show that the effects of upsampled 3D points on object detection are closely related to the object distances. To effectively use 3D upsampling techniques for object detection, some criteria are carefully designed to determine whether Lidar or PU-GAN upsampled points will be used for 3D box estimations.

First, instance segmentation is performed on RGB images to get 2D object masks. In our implementations, Mask R-CNN [24] has been used. Suppose S_O^M and S_G^M are the number of Lidar points and upsampled points from PU-GAN in the mask respectively, we define two criteria to classify the object proposals from a 2D object detector into easy and hard samples, as shown in (1).

$$\begin{cases} crt_1 = S_O^M \\ crt_2 = S_O^M / S_G^M \end{cases} \tag{1}$$

Figure 3 shows the distribution of crt_1 and crt_2 over the distances from the camera. They are plotted based on the ground-truth objects. It can be easily observed that, objects close to the camera tend to have more Lidar points and much higher ratio of Lidar points over upsampled points. We define proposals with crt_1 greater 90 and crt_2 higher than 0.15 as easy samples and it is believed Lidar points are sufficient to make a reliable 3D box estimation for these samples. Other proposals are defined as hard samples and upsampled dense points from PU-GAN will be injected into PointNet++ to predict 3D objects.

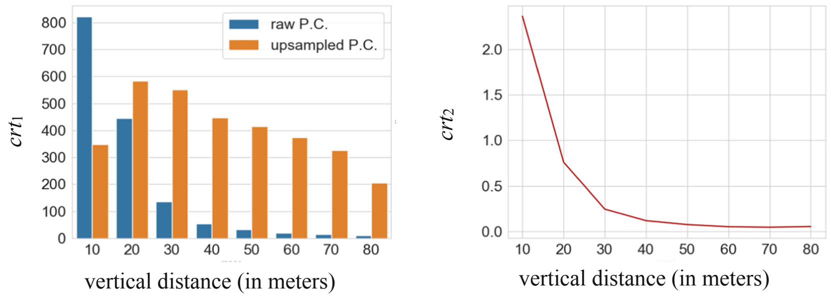

Fig. 3. The distribution of crt_1 and crt_2 over the distances.

Finally, frustums formed by the point clouds are injected into PointNet++ for 3D box estimation. PoinNet++ contains a segmentation network to extract the points of interest in each frustum, a learning-based 3D alignment T-net to refine the centroid of the object based on the masked foreground points, and a PointNet-based 3D box estimation network to finally regress the sizes, positions and heading angles of the 3D objects.

4 Experiments

4.1 Implementation Details

Dataset and Evaluation Metric. All the experimental results are performed on KITTI 3D object detection benchmark [20], which consists of 7481 training images and 7518 test images, as well as the corresponding point clouds. There are a total of 80,256 labeled objects, including three object categories (i.e., car, pedestrian, and cycle). Each category is divided into three different difficulty levels (easy, moderate and hard) according to the object distances and occlusion degrees.

In this experiment, we follow the train/val split as in [8]. The evaluation metric is average precision with the 3D IoU thresholds at 0.7 for car category and 0.5 for pedestrian and cycle categories.

2D Instance Detection. The 2D box proposals come from the same 2D detector model in Frustum PointNets [1]. The open-sourced mask R-CNN [25] is used as the instance segmentation algorithm to obtain the instance masks. The model is pre-trained on the Cityscapes Dataset [26]. For cyclist category, we merge the masks of riders and bicycles to get complete instance mask proposals.

Parameter Settings. The implementation of PU-GAN upsampling method follows the publicly available code in [27]. The farthest sampling strategy is used to choose seeds in the Lidar points and a local patch with 256 points is extracted around each seed. The patches are fed to the generator and upsampled by 12 times. Finally, the upsampled patches are merged and farthest sampling is performed to obtain the final uniformly-distributed points.

The PU-GAN network is trained based on 120 3D models, ranging from simple and smooth models (e.g., Icosahedron) to complex and high-detailed objects (e.g., Statue). A total of 24, 000 patches are collected by cropping 200 patches from each model. These input patches are randomly rotated, scaled to avoid overfitting. The network is trained for 100 epochs with Adam optimizer.

In all the experiments, PointNet++ [14] is used as the backbone of Frustum PointNets for 3D objection. The implementation follows the code in [28]. The models are trained using Adam optimizer with a batch size of 32 frustums, momentum of 0.9. The learning rate is initialized at 0.001 and 0.7 decays every 10 epochs. The models are trained around 80 epochs on one NVIDIA GTX1080Ti GPU. To avoid overfitting, the input frustums are augmented by randomly shifting and expanding the 2D proposals.

4.2 Experimental Results

Results When PU-GAN Points are Applied at Different Distances. To test the impacts of PU-GAN upsampled points on object detection in details, PU-GAN upsampled points are applied on the objects at different distances in this experiment. To decouple the influence of 2D detectors, we use ground truth 2D boxes for region proposals and test the 3D box estimation accuracy. Average precisions of the moderate level in each category are evaluated. In Fig. 3, each point shows the average precision when upsampled points are used for the objects beyond a certain range. For example, when the distance is zero meters, upsampled points are used for all the proposals to estimate the 3D boxes. When the distance is 30 meters, only the proposals beyond 30 meters away from the camera use upsampled points for 3D box estimation and others use original Lidar points. The blue curve is for the car category, the red one is for cyclists and the green one is for pedestrians. The dashed lines indicate the baseline performance of each category in [1].

From Fig. 3, it can be easily observed that the effects of upsampled points on 3D object detection are closely related to the distances of the objects. Using PU-GAN generated dense points for all the objects are not a good option . For objects near the camera, the accuracy of 3D detection significantly decreases when the generated points are used instead of Lidar points. The upsampled points show their superiority when they are applied on the objects beyond 30 meters away from the camera. The superiority decreases when only the objects beyond 40 meters are considered.

It seems that, PU-GAN upsampling can provide dense and reliable 3D points for cars, pedestrians and cyclists at around 30 meters away from the camera. For objects near the camera, Lidar points are dense enough to make a decision on the object types, locations and orientations. On the other hand, the generated point data from PU-GAN may not be as accurate as the Lidar points. This may be the reason why the accuracy decreases when PU-GAN is used for the objects at a short distance. The observations may also indicate that there is still a large gap for the 3D point upsampling techniques to be as accurate as real Lidar points for reliable 3D object detection. For far-away objects, the underlying geometry of the object may not be able to be well indicated by very few Lidar points, which may mislead the generation of 3D point generation by PU-GAN (Fig. 4).

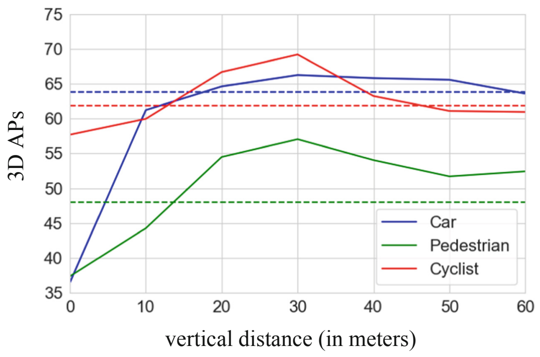

Fig. 4. The 3D APs of the moderate level in each category when PU-GAN generated dense points are used beyond a certain distance.

Detection When PU-GAN Points are Applied with Our Criteria. In the test stage, distances of objects are unknown. In this experiment, criteria proposed in Sect. 3 are tested to verify its efficiency to improve the 3D detection performance using upsampled dense points. Since objects close to the camera tend to have sufficient and accurate Lidar point data, dense points generated from PU-GAN are only used for 2D proposals with S_O^M less than 90 and S_O^M/S_G^M lower than 0.15. Frustum PointNets is used as a baseline for comparison since it is still a state-of-the-art work of 3D object detection using RGB-D data. The 3D object detection AP for each category are shown in Table 1. The results of Frustum PointNets on KITTI *val* set are obtained using the public code in [28] and the models are trained by ourselves.

Table 1. 3D object detection AP on KITTI *val* set.

Category	Method	3D AP		
		Easy	Mod.	Hard
Car	Frustum PointNet	82.39	**69.86**	**62.73**
	Ours	**82.72**	67.88	60.31
Pedestrian	Frustum PointNet	60.96	53.08	47.41
	Ours	**65.81**	**57.34**	**50.14**
Cyclist	Frustum PointNet	67.33	50.73	47.47
	Ours	**71.57**	**52.05**	**48.25**

From Table 1, it can be observed that significant improvements are achieved for pedestrians and cyclists. The average precision is increased by around 5% and 4% respectively for the easy level in the pedestrian and cyclist category. The moderate level and hard level for pedestrians are improved by 4.26% and 2.73% respectively. The results show the potential of the upsampled dense point data from the data-driven approaches for 3D object detection. To further improve its efficiency in the moderate and hard levels, the 3D upsampling techniques may need more prior knowledge about the geometry of the specific categories. For the car category, the detection performance for the easy level is slightly improved while the moderate and hard level decrease a little bit. The reason may be that the threshold setting for S_O^M and S_O^M/S_G^M is strict for the car category and the current upsampling techniques are not powerful enough to produce reliable points for those difficult cases. In the future, the parameters will be tuned carefully and different thresholds may be considered for the proposals with different category labels.

Figure 5 Gives More Qualitative Results for Comparison. In each group, the upper part is the 3D box detection results shown in RGB images while the lower part is the 3D box detection results shown with Lidar point cloud in bird's eye view. In a), c) and d), it can be observed that the original Frustum PointNets algorithm has large errors in the prediction of 3D bounding box in the illustrated samples. With the developed method, the positions and the orientations of the cars can be better estimated due to the complementary knowledge from the upsampled dense points. In b), a partially-occluded pedestrian is missed in the baseline, but the developed method has successfully detected it.

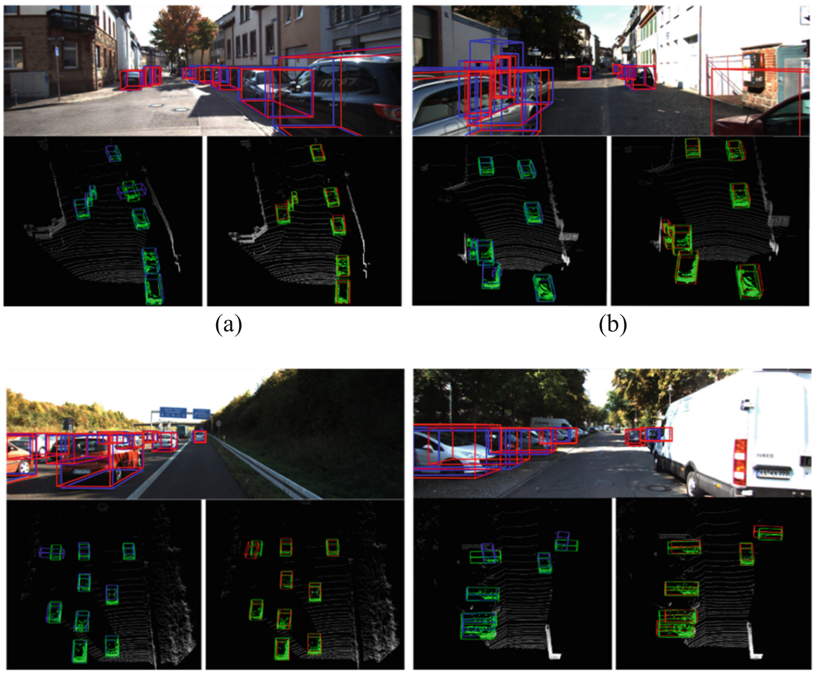

Fig. 5. Qualitative **comparison of the Frustum PointNets and our method.** In each group, the upper part is the 3D box detection results shown in RGB images: our method (red) and the Frustum PoinNet (blue). The lower part is the 3D box detection results shown with Lidar point cloud in bird's eye view. Groundtruth (green), Frustum PoinNets (blue) and our developed method (red). (Color figure online)

5 Conclusions

In this paper, a 3D object detection method using PU-GAN has been developed. PU-GAN upsampled dense points are tested for object detection at different distances. Extensive experimental results show that the impacts of 3D upsampling on object detection are closely related to object distances from the camera. Two criteria are proposed to better combine the advantages of the lidar points and the generated dense points. The results on KITTI 3D object detection benchmark show the potential of the 3D upsampling method for 3D object detection purpose.

Although our algorithm is based on Frustum PointNet and PU-GAN, the idea to complement the sparsity of LiDAR data by upsampling can be easily applied to other architectures. In the future, more upsampling techniques will be examined for 3D object detectioin. Parameter settings will be discussed in details.

In addition, the results of the paper also indicate that there is still much room for the 3D upsampling techniques to be used for object detection. The accuracy and the ability to produce reliable dense points when there are few original point data need to be improved. The speed of upsampling is also a key to the wide applications of upsampling techniques in object detections.

Acknowledgement. This work is supported by the National Natural Science Foundation of China (No. 61771042, 61702032 and 41761079) and Beijing Natural Science Foundation (No. 4202061).

References

1. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3D object detection from RGB-D data. In: CVPR (2018)
2. Li, R., Li, X., Fu, C., Cohen-Or, D., Heng, P.: PU-GAN: a point cloud upsampling adversarial network. In: Proceedings of the IEEE International Conference on Computer Vision (2019)
3. Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., Sun R.U.: Monocular 3D object detection for autonomous driving. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2147–2156 (2016)
4. Xiang, Y., Choi, W., Lin, Y., Savarese, S.: Data-driven 3D voxel patterns for object category recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1903–1911 (2015)
5. Li, B., Zhang, T., Xia, T.: Vehicle detection from 3D lidar using fully convolutional network. In: RSS (2016)
6. Deng, Z., Latecki, L.J.: Amodal detection of 3D objects: inferring 3D bounding boxes from 2D ones in RGB-depth images. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
7. Zhou, J., Tan, X., Shao, Z., Ma, L.: FVNet: 3D front-view proposal generation for real-time object detection from point clouds. In: 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI) (2019)
8. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3D object detection network for autonomous driving. In: IEEE CVPR (2017)
9. Zhou, Y., Tuzel, O.: Voxelnet: end-to-end learning for point cloud based 3D object detection. CoRR (2017)
10. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
11. Yan, Y., Mao, Y., Li, B.: Second: sparsely embedded convolutional detection. Sensors (2018)
12. Graham, B., Engelcke, M., van der Maaten, L.: 3D semantic segmentation with submanifold sparse convolutional networks. In: CVPR (2018)
13. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Jbom, O.B.: Pointpillars: fast encoders for object detection from point clouds. In: CVPR (2019)
14. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: deep hierarchical feature learning on point sets in a metric space. In: NIPS (2017)
15. Wang, Y., Chao, W., Garg, D., Hariharan, B., Campbell, M., Weinberger, K.Q.: Pseudo-lidar from visual depth estimation: bridging the gap in 3D object detection for autonomous driving. In: CVPR (2019)
16. You, Y., et al.: Pseudo-LiDAR++: accurate depth for 3D object detection in autonomous driving (2019)
17. Ma, X., Wang, Z., Li, H., Ouyang, W., Fan, X., Zhang, P.: Accurate monocular object detection via color-embedded 3D reconstruction for autonomous driving. In: ICCV (2019)

18. Weng, X., Kitani, K.: Monocular 3D object detection with pseudo-LiDAR point cloud. In: CVPR (2019)
19. Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J.: STD: sparse-to-dense 3D object detector for point cloud. In: ICCV (2019)
20. Kitti 3D object detection benchmark. http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d (2019)
21. Yu, L., Li, X., Fu, C.W., Cohen-Or, D., Heng, P.A.: PU-net: point cloud upsampling network. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2790–2799 (2018)
22. Yu, L., Li, X., Fu, C.W., Cohen-Or, D., Heng, P.A.: EC-Net: an edge-aware point set consolidation network. In: European Conference on Computer Vision (ECCV), pp. 386–402 (2018)
23. Yifan, W., Wu, S., Huang, H., Cohen-Or, D., Hornung, O.S.: Patch-based progressive 3D point set upsampling. In: CVPR, pp. 5958–5967 (2019)
24. He, K., Gkioxari, G., Doll, P., Girshick, R.: Mask R-CNN. In: ICCV (2017)
25. Wu, Y., Kirillov, A., Massa, F., Lo, W., Girshick, R.: Detectron2 (2019). https://github.com/facebookresearch/detectron2
26. Cordts, M., et al.: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016)
27. Li, R., Li, X., Fu, C., Cohen-Or, D., Heng, P.: PU-GAN (2019). https://liruihui.github.io/publication/PU-GAN/
28. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum-pointnets. https://github.com/charlesq34/frustum-pointnets