2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

mini-PointNetPlus: A Local Feature Descriptor in Deep Learning Model for Real-time 3D Environment Perception

Chuanyu Luo[1], Nuo Cheng[2], Sikun Ma[2], Jun Xiang[2], Xiaohan Li[2], Shengguang Lei[2], Pu Li[3]

***Abstract*— Common deep learning models for 3D real-time environment perception often use pillarization/voxelization methods to convert point cloud data into pillars/voxels and then process it with a 2D/3D convolutional neural network (CNN). The pioneer work PointNet has been widely applied as a local feature descriptor, a fundamental component in deep learning models for 3D perception, to extract features of a point cloud. This is achieved by using a symmetric max-pooling operator which provides unique pillar/voxel features. However, by ignoring most of the points, the max-pooling operator causes an information loss, which reduces the model performance. To address this issue, we propose a novel local feature descriptor, mini-PointNetPlus, as an alternative for plug-and-play to PointNet. Our basic idea is to separately project the data points to the individual features considered, each leading to a permutation invariant. Thus, the proposed descriptor transforms an unordered point cloud to a stable order. The vanilla PointNet is proved to be a special case of our mini-PointNetPlus. Due to fully utilizing the features by the proposed descriptor, we demonstrate in experiment a considerable performance improvement for 3D perception.**

I. INTRODUCTION

LiDAR plays a crucial role in 3D environment perception for autonomous vehicles, robotics, and various other fields. The task of detecting and localizing objects in 3D space presents significant challenges. In contrast to image-based 2D computer vision techniques, LiDAR-generated point clouds are unordered, sparse, and exhibit varying densities due to the nature of distance information they contain. These characteristics render LiDAR-based perception tasks particularly difficult, necessitating advanced algorithms and methodologies to effectively interpret and utilize the data for real-world applications.

Most LiDAR-based approaches [4][3][20] to 3D object detection follow a standardized pipeline, initially converting the raw point cloud data into pillars or voxels. Subsequently, a local feature descriptor aggregates the features within each pillar or voxel from its constituent points. The aggregated features are then processed through a 3D or 2D CNN/Transformer [17] backbone to create a comprehensive feature map. Finally, potential bounding boxes (Bboxes) are predicted using the information from the feature map. A process visually is detailed in Figure 1.

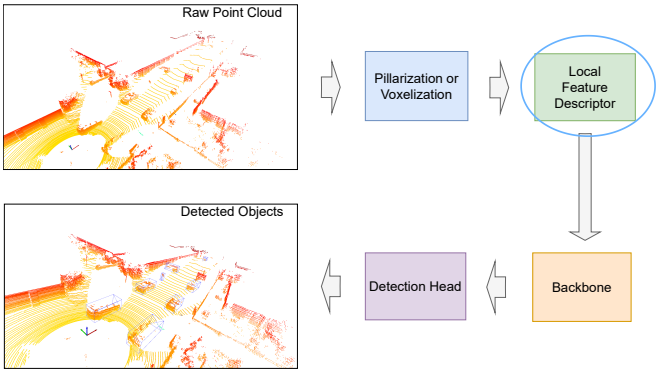

Fig. 1. The detection pipeline for most LiDAR-only 3D real-time object detection models. In our paper, we propose a new local feature descriptor module, which is in the highlighted blue circle.

To process an unordered point cloud, the pioneering work, PointNet [2], is proposed for directly operating on the raw point cloud for 3D indoor object classification and segmentation. Using a symmetric max-pooling layer, PointNet can handle unordered and irregular point clouds without losing the fine-grained information of objects.

Considering large-scale LiDAR-based driving scenes, VoxelNet [5] is proposed to divide an irregular point cloud into a set of dense 3D voxels, where a mini-PointNet is used as a voxel feature extraction module. And then a dense 3D-CNN is used as the backbone for object detection. After that, SECOND [4] is developed for real-time 3D object detection, in which a point cloud is divided into a set of sparse 3D voxels and thus a sparse 3D CNN on the non-empty voxels can be used. By projecting the 3D voxels to a 2D BEV feature map and 2D convolution, SECOND achieves high detection accuracy with a high speed. The mini-PointNet is also used in SECOND as a voxelwise feature extractor (VFE) module [5]. Besides, in a later version of SECOND, the mean value of the point cloud features is also represented as a voxel feature.

Inspired by PointNet and SECOND, PointPillars [3] is developed in which a point cloud is divided into 2D pillars as vertical columns of points. The features are also extracted by the mini-PointNet, on pillars instead of voxels. By only using 2D pillars with vertical spatial information, PointPillars achieves impressively high efficiency and widely used especially in real-time projects.

In the latest state-of-the-art approaches like [7] [6] [16],

*This work is supported by LiangDao GmbH

[1]Chuanyu Luo is with the LiangDao GmbH, Berlin, 12099, Germany and also with the Ilmenau University of Technology, Ilmenau, 98693, Germany chuanyu.luo@tu-ilmenau.de.

[2]Nuo Cheng, Sikun Ma, Jun Xiang, Xiaohan Li, Shengguang Lei are with the LiangDao GmbH, Berlin, 12099, Germany nuo.cheng, sikun.ma, jun.xiang, xiaohan.li, shengguang.lei@liangdao.de.

[3]Prof. Pu Li is with the Ilmenau University of Technology, Ilmenau, 98693, Germany pu.li@tu-ilmenau.de.

the mini-PointNet has become a popular and standard component to extract unordered point cloud features on voxels/pillars. However, is the mini-PointNet really a perfect approach to feature extraction? The answer we think is no, since the non-parametric max-pooling used in it leads to much information loss by ignoring most points. In a similar case of 2D computer vision [1], it is reported that replacing the 2D max-pooling layer with a parametric strided 2D convolution can stabilize the model and improve the performance.

II. Related work

In this section, the popular LiDAR-based 3D detection models and the relevant local feature descriptors are reviewed.

A. LiDAR-based 3D detection models

In the field of LiDAR-based 3D object detection, the framework of most detection modules include the pillarization/voxelization module, the local feature descriptor, the 2D/3D backbone and the detection head.

In the backbone module, VoxelNet [5] use dense 3D-CNN to aggregate the voxel feautures. In SECOND [4], sparse 3D-CNN is used to aggregate the voxels' features, and 2D-CNN to aggregate 2D feature map features. In PointPillars [3], only 2D-CNN is used to aggregate the features information. By leveraging the power of Transformer [17] for long distance information aggregation, SST [18] and DSVT [19] use shifted windows of multiple pillars to better aggregate the features from pillars.

In the detection head module, the vanilla PointPillars uses the Single Shot Detector (SSD) [21] setup to perform 3D object detection. Motivated by 2D CenterNet [22], CenterPoint [20] proposes a new anchor-free key-point detection head from the feature map to predict the centers of objects and other attributes. Furthermore, VoxelNeXt [23] proposes a fully sparse head to directly predict objects based on sparse voxel features.

B. Local feature descriptors

In the local feature descriptor module, which is the focus of our work, like the vanilla PointNet, most works [8][9][10] focus on indoor classification and semantic segmentation from point clouds. For example. PointNet++ [8] use hierarchical architecture to recursively sample and aggregate local features by PointNet. Point Transformer [9] use multiple self-attention layers with farthest point sampling and KNN in a hierarchical neural network for indoor point cloud semantic segmentation task. These complex architecture makes it inefficient for outdoor real-time object detection tasks.

In the outdoor LiDAR-based 3D object detection tasks, the most relevant work is PASNet [11], which uses a learning-based method to sort an unordered point cloud. However, for N points in a voxel/pillar, there is $N!$ permutation possibilities. Generally, all sorting-based methods map high dimensional data to a 1D sequence and thus can be sensitive to the number of points in voxels.

In this study, we consider high dimensional data in a point loud as a combination of the corresponding 1D dimensional data. Therefore, we project the data points to individual dimensions separately. As a result, instead of sorting high dimensional data, we sort the projected 1D dimension data, leading to a stable order. Finally, a learning-based weight extract multiple features of the point cloud.

III. Methodology

In this section, we present a novel grid feature extractor (descriptor), mini-PointNetPlus, to aggregate the features of an unordered point cloud. Here, a grid means a pillar or a voxel. The proposed module can be used as a standard component in almost all 3D CNN for object detection in large-scale driving scenes. At first, the unordered point cloud problem will be stated and then the solution of vanilla PointNet analysed.

Let $x \in \mathbb{R}^C$ denote a point with feature dimensions C. And $S = \{x_1, x_2, ..., x_N\}$ denotes the point set including N unordered points in a grid. We would like to find a solution $O \in \mathbb{R}^C$, which actually describes the grid features, by an extractor function $O = f(x_1, x_2, ..., x_N)$. Since a point cloud is unordered, the output O and function f should be permutation invariant, which means

$$O \equiv f(x_1, x_2, ..., x_N) \equiv f(x_i, x_j, ..., x_k). \tag{1}$$

Here $i, j, ..., k$ represent the permuted index set $\{1, 2, ..., N\}$. In PointNet, it is proposed that the extractor function f is a combination of the multi-layer perceptron, and an invariant symmetric function by max-pooling, denoted as follows

$$\begin{aligned} &O = f(x_1, x_2, ..., x_N) = Max(h(x_1), h(x_2), ..., h(x_n)), \\ &Max : \mathbb{R}^{N \times C} \rightarrow \mathbb{R}^C \end{aligned} \tag{2}$$

Here h is the multi-layer perceptron network. It is noted that the max-pooling function works on the same feature dimension across different points. Other symmetric functions include the addition, average pooling etc. Experiment results [2] show that the max-pooling operator performs better than the other symmetric functions.

However, one obvious limitation of PointNet is that the max-pooling operator ignores the less important points. In our proposed approach, it can be proved that the symmetric function is in fact not the key to keep the permutation invariant property, and all the points can be considered with an adaptive weight.

For N points with dimension C, there exists $N!$ permutations, and there is no stable order to map the high dimensional data to 1D sequence. Fig. 2 shows two options of sorting paths.

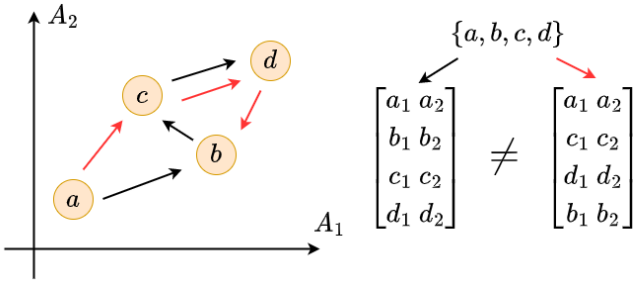

Fig. 2. For illustration purpose, only 2D points are shown. In practice, the points can be high dimensional data after multi-layer perceptron. The black and red arrows indicate two possible orders. There is actually no stable order to sort all the points by a function to map from set to matrix.

However, if projecting high dimensional points to each individual feature/channel dimension separately, there is a permutation invariant and a stable order in each dimension, as illustrated in Fig. 3.

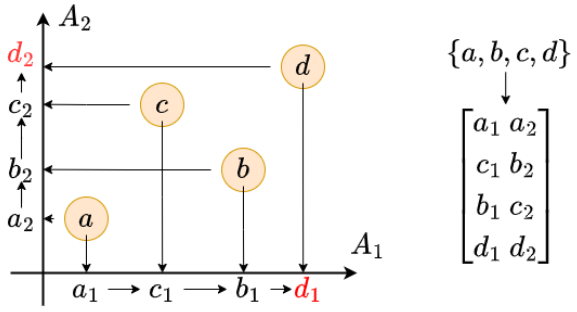

Fig. 3. 2D points are projected to the A_1-axis and A_2-axis separately. There is a stable and permutation invariant order, i.e., to sort along the A_1-axis and A_2-axis, respectively. Note that PointNet considers only the red max values d_1 and d_2.

Therefore, for point set $S = \{x_1, x_2, ..., x_N\}$, we define the set function in Eq. 1 as $f(S) = Sort(Proj(S))$, where f is a permutation invariant set function, $Proj$ function projects all points in each feature dimension, and the $Sort$ function sorts the points ascendingly in each feature dimension.

The permutation invariant property of the set function can be expressed as Eq. 3, corresponding to Fig. 3.

$$\begin{bmatrix} a_1 & a_2 \\ c_1 & b_2 \\ b_1 & c_2 \\ d_1 & d_2 \end{bmatrix} \equiv f(a, b, c, d) \equiv f(c, d, b, a). \tag{3}$$

Generally, the output $A \in \mathbb{R}^{N \times C}$ can be written as a matrix as follows

$$\begin{aligned} A &= f(h(x_1), h(x_2), ..., h(x_n))) \\ &= \begin{bmatrix} a_{11} & \cdots & a_{1C} \\ a_{21} & \cdots & a_{2C} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NC} \end{bmatrix}. \end{aligned} \tag{4}$$

Here $f(S) = Sort(Proj(S))$. It means, the elements of any column index j in A always satisfy $a_{1j} < a_{2j} < \cdots < a_{Nj}$. The final output O of the extractor function will be

$$\begin{aligned} O &= w^T A \\ &= \begin{bmatrix} w_1 & \cdots & w_N \end{bmatrix} \begin{bmatrix} a_{11} & \cdots & a_{1C} \\ a_{21} & \cdots & a_{2C} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NC}. \end{bmatrix} \end{aligned} \tag{5}$$

Here w^T is a vector of the learn-able weight parameters. Especially, in the vanilla PointNet, $w^T = \begin{bmatrix} 0 & 0 & \cdots & 1 \end{bmatrix}$.

In the context of real-time autonomous driving applications, the proposed mini-PointNetPlus module relies solely on the **sort** or **topK** operator—ubiquitous across nearly all platforms. This characteristic facilitates its practical deployment, ensuring high-speed operation.

IV. Experiments

For a numerical verification, we take the pillar-based method PointPillars [3] and the voxel-based method SECOND [4] as our experiment pipeline. The mini-PointNet in the vanilla pipeline is replaced and compared with the adaptive sorting method PASNet [11] and the proposed mini-PointNetPlus, respectively, for comparison. The KITTI [12] and nuScenes [13] dataset are employed as point cloud data in our experiment.

For the pillar-based method using PointPillars, the pillar size is set as $0.16m \times 0.16m \times 4m$ in KITTI and $0.2m \times 0.2m \times 8.0m$ in nuScenes, while the max points number in a pillar is taken as 32 in KITTI and 20 in nuScenes. For the voxel-based method using SECOND, the voxel size is set as $0.05m \times 0.05m \times 0.1m$ and the max points number in voxel is set as 5 in KITTI, respectively.

Except for the grid feature extraction module, all the other modules and training hyper-parameters are take with the same values.

In KITTI, the average precision (AP) and the average orientation similarity (AOS) [12] are used as evaluation metrics. In nuScenes, the evaluation metrics are AP and the nuScenes detection score (NDS) [13].

The results based on the KITTI validation set is listed in Table I and Table II, while the evaluation based on the nuScenes validation set is listed in Table III.

Table I demonstrates that, for both the pillar-based method (PointPillars) and the voxel-based method (SECOND), the proposed mini-PointNetPlus module achieves accuracy improvements comparable to those of the PointNet baseline and PASNet module in terms of mean Average Precision (mAP), with only a negligible effect on processing speed. Notably, Table II reveals a significant enhancement in the object orientation metric (mAOS) attributable to the mini-PointNetPlus module.

Further experimentation conducted on the more extensive nuScenes dataset, as shown in Table III, indicates consistent accuracy enhancements provided by our proposed module, including a 0.44-point increase in mAP and a 0.47-point

Method	Car			Pedestrian			Cyclist			mAP	speed
	Easy	Moderate	hard	Easy	Moderate	hard	Easy	Moderate	hard	-	-
PointPillars [3] + PointNet [2]	85.17	76.29	74.22	56.66	52.38	**47.78**	**80.69**	62.74	58.97	66.10	**22.3ms**
PointPillars [3] + PASNet [11]	86.77	76.48	74.38	**59.11**	**53.02**	47.67	79.32	61.86	59.21	66.42 (+0.32)	25.5ms (+3.2ms)
PointPillars [3] + **miniPointNetPlus**	**87.62**	**77.63**	**75.49**	57.72	52.06	47.19	79.13	**62.88**	**60.14**	**66.65** (+0.55)	23.5ms (+1.2ms)
SECOND [4] + PointNet [2]	90.75	89.79	88.91	63.48	60.99	58.41	86.71	**78.90**	74.56	76.94	34.8ms
SECOND [4] + Mean Point Features	90.79	89.83	89.02	65.03	**62.17**	58.69	87.91	77.29	74.03	77.19 (+0.25)	33.7ms (-1.1ms)
SECOND [4] + **miniPointNetPlus**	**90.82**	**89.92**	**89.08**	**65.11**	61.62	**59.00**	**90.18**	77.02	**74.58**	**77.48** (+0.54)	**33.0ms** (-1.8ms)

TABLE I

RESULTS ON KITTI VAL 3D DETECTION BENCHMARK WITH AP. THE SPEED IS MEASURED ON A TESLA V100 WITH ONE GPU 32G.

Method	Car			Pedestrian			Cyclist			mAOS	speed
	Easy	Moderate	hard	Easy	Moderate	hard	Easy	Moderate	hard	-	-
PointPillars [3] + PointNet [2]	90.72	89.39	88.28	45.53	42.34	40.30	84.53	71.70	68.25	69.00	**22.3ms**
PointPillars [3] + PASNet [11]	90.79	89.52	88.12	47.32	**44.81**	**42.48**	84.32	69.18	66.23	69.19 (+0.19)	25.5ms (+3.2ms)
PointPillars [3] + **miniPointNetPlus**	**90.84**	**89.60**	**88.59**	**47.43**	44.55	41.81	**86.55**	**73.41**	**69.79**	**70.28** (+1.28)	23.5ms (+1.2ms)
SECOND [4] + PointNet [2]	88.12	**78.57**	**77.25**	57.78	**53.90**	**49.17**	80.15	65.82	61.38	68.01	34.8ms
SECOND [4] + Mean Point Features	88.31	78.37	77.22	**57.84**	53.72	48.11	80.88	**66.85**	62.58	68.20 (+0.19)	33.7ms (-1.1ms)
SECOND [4] + **miniPointNetPlus**	**88.38**	78.40	**77.25**	57.54	53.64	48.82	**82.78**	66.83	**62.91**	**68.50** (+0.49)	**33.0ms** (-1.8ms)

TABLE II

RESULTS ON KITTI VAL 3D DETECTION BENCHMARK WITH AOS. THE SPEED IS MEASURED ON A TESLA V100 WITH ONE GPU 32G

Method	mAP	NDS	speed
PointPillars [3] + PointNet [2]	44.07	57.71	**22.6ms**
PointPillars [3] + PASNet [11]	41.73 (-2.34)	56.41 (-1.30)	27.5ms (+4.9ms)
PointPillars [3] + **miniPointNetPlus**	**44.51** (+0.44)	**58.18** (+0.47)	22.7ms (+0.1ms)

TABLE III

RESULTS ON NUSCENES VAL 3D DETECTION BENCHMARK. THE SPEED IS MEASURED ON A TESLA A100 WITH ONE GPU 40G

increase in the NDS metric, while only marginally affecting the time performance (an increase of 0.1ms).

V. CONCLUSION

In this paper, we analyzed the mechanism and limitation of the widely applied method, PointNet, based on which we proposed an approach to separately projecting the individual features to 1D dimensions. It is proved that the PointNet is only one special case of our proposed mini-PointNetPlus. The symmetric function has been widely considered as the key solution to unordered points. But our method shows that if the points are projected separately in the projected dimension, we can find a stable and permutation invariant order. The symmetric function is only one handcrafted function to handle the projected points.

Experimental evidence indicates that for pillar-based or voxel-based methods, our proposed method provides a better performance with only a little cost of efficiency. Given the straightforward implementation and deploy simplicity of our proposed module, it emerges as a viable alternative to PointNet for real-time 3D environmental perception tasks.

REFERENCES

[1] Springenberg, J., Dosovitskiy, A., Brox, T. & Riedmiller, M. Striving for Simplicity: The All Convolutional Net. *ICLR (workshop Track)*. (2015), http://lmb.informatik.uni-freiburg.de/Publications/2015/DB15a

[2] Qi, C., Su, H., Mo, K. & Guibas, L. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*. pp. 652-660 (2017)

[3] Lang, A., Vora, S., Caesar, H., Zhou, L., Yang, J. & Beijbom, O. Pointpillars: Fast encoders for object detection from point clouds. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 12697-12705 (2019)

[4] Yan, Y., Mao, Y. & Li, B. Second: Sparsely embedded convolutional detection. *Sensors*. **18**, 3337 (2018)

[5] Zhou, Y. & Tuzel, O. Voxelnet: End-to-end learning for point cloud based 3d object detection. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*. pp. 4490-4499 (2018)

[6] Yin, T., Zhou, X. & Krahenbuhl, P. Center-based 3d object detection and tracking. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 11784-11793 (2021)

[7] Fan, L., Pang, Z., Zhang, T., Wang, Y., Zhao, H., Wang, F., Wang, N. & Zhang, Z. Embracing single stride 3d object detector with sparse transformer. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 8458-8468 (2022)

[8] Qi, C., Yi, L., Su, H. & Guibas, L. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances In Neural Information Processing Systems*. **30** (2017)

[9] Zhao, H., Jiang, L., Jia, J., Torr, P. & Koltun, V. Point transformer. *Proceedings Of The IEEE/CVF International Conference On Computer Vision*. pp. 16259-16268 (2021)

[10] Li, Y., Bu, R., Sun, M., Wu, W., Di, X. & Chen, B. Pointcnn: Convolution on x-transformed points. *Advances In Neural Information Processing Systems*. **31** (2018)

[11] Cheng, N., Li, X., Li, H., Liu, X., Luo, C., Lei, S. & Li, P. PASNET: A Self-AdaPtive Point Cloud Sorting APProach to an ImProved Feature Extraction. *2022 IEEE International Conference On Image Processing (ICIP)*. pp. 956-960 (2022)

[12] Geiger, A., Lenz, P. & Urtasun, R. Are we ready for autonomous driving? the kitti vision benchmark suite. *2012 IEEE Conference On Computer Vision And Pattern Recognition*. pp. 3354-3361 (2012)

[13] Caesar, H., Bankiti, V., Lang, A., Vora, S., Liong, V., Xu, Q., Krishnan, A., Pan, Y., Baldan, G. & Beijbom, O. nuscenes: A multimodal dataset for autonomous driving. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 11621-11631 (2020)

[14] Zheng, W., Tang, W., Chen, S., Jiang, L. & Fu, C. Cia-ssd: Confident iou-aware single-stage object detector from point cloud. *Proceedings*

Of The AAAI Conference On Artificial Intelligence. pp. 3555-3562 (2021)

[15] Zheng, W., Tang, W., Jiang, L. & Fu, C. SE-SSD: Self-Ensembling Single-Stage Object Detector From Point Cloud. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition (CVPR)*. pp. 14494-14503 (2021,6)

[16] Guangsheng Shi, C. PillarNet: Real-Time and High-Performance Pillar-based 3D Object Detection. *ECCV*. (2022)

[17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, Ł. & Polosukhin, I. Attention is all you need. *Advances In Neural Information Processing Systems*. **30** (2017)

[18] Fan, L., Pang, Z., Zhang, T., Wang, Y., Zhao, H., Wang, F., Wang, N. & Zhang, Z. Embracing single stride 3d object detector with sparse transformer. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 8458-8468 (2022)

[19] Wang, H., Shi, C., Shi, S., Lei, M., Wang, S., He, D., Schiele, B. & Wang, L. Dsvt: Dynamic sparse voxel transformer with rotated sets. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 13520-13529 (2023)

[20] Yin, T., Zhou, X. & Krahenbuhl, P. Center-based 3d object detection and tracking. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 11784-11793 (2021)

[21] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. & Berg, A. Ssd: Single shot multibox detector. *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. pp. 21-37 (2016)

[22] Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q. & Tian, Q. Centernet: Keypoint triplets for object detection. *Proceedings Of The IEEE/CVF International Conference On Computer Vision*. pp. 6569-6578 (2019)

[23] Chen, Y., Liu, J., Zhang, X., Qi, X. & Jia, J. Voxelnext: Fully sparse voxelnet for 3d object detection and tracking. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*. pp. 21674-21683 (2023)