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

IC-FPS: Instance-Centroid Faster Point Sampling Framework for 3D Point-based Object Detection

Haotian Hu
Leapmotor
hu_haotian@leapmotor.com

Fanyi Wang
Zhejiang university
11730038@zju.edu.cn

Yaonong Wang
Leapmotor
wang_yaonong@leapmotor.com

Laifeng Hu
Leapmotor
hu_laifeng@leapmotor.com

Zhiwang Zhang
The University of Sydney
zhiwang.zhang@sydney.edu.au

***Abstract*— 3D object detection is one of the most important tasks in autonomous driving and robotics. Our research focuses on tackling low efficiency issue of point-based methods, and we propose a novel Instance-Centroid Faster Point Sampling (IC-FPS) framework. We design a Neighboring Feature Diffusion Module (NFDM) to extract local features for the purpose of efficiently distinguishing the foreground from the background. Considering Farthest Point Sampling (FPS) strategy for down-sampling is computationally intensive, we propose the Centroid-Instance Sampling Strategy (CISS). CISS samples center point in large-scale point cloud by rapidly sampling the centroid and instance points of the foreground block. The proposed IC-FPS framework can be inserted into every point-based model and effectively replace the first Set Abstraction (SA) layer. Extensive experiments on several public benchmarks demonstrate the superior performance of our proposed IC-FPS. On the Waymo dataset, IC-FPS significantly improves performance of the benchmark model and increases inference speed by 3.8 times. And real-time detection of point-based methods is realized for the first time, which is meaningful for industrial applications.**

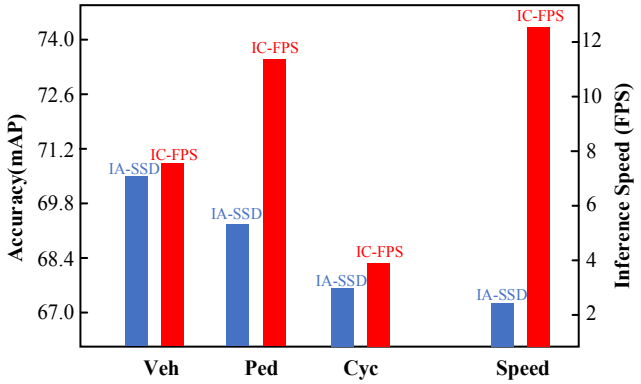

Fig. 1. Comparison of performance and inference time of various models on Waymo dataset [11]. Experiment results are derived by using OpenPCDet [23] framework on a single A40 GPU. More details can be found in Table III.

I. INTRODUCTION

In recent years, with rapid development of sensors such as LiDAR and millimeter wave radar, point cloud has been widely applied in 3D tasks as a common 3D representation. And 3D object detection plays a crucial role in autonomous driving. However, due to the orderless, sparse and irregular nature of point cloud, it is still challenging to predict 3D detection box with multiple degrees-of-free. Previous works projected point clouds into multiple views [1], [2], [3], [4] for feature extraction. But it is inevitable to cause information loss when converting point clouds to intermediate transforms, leading to degradation of model performance. And voxel-based methods [5], [6], [7], [8], [9] rely on 3D sparse convolution, making such methods difficult to deploy in industry. Therefore, point-based methods have become feasible.

As we know, point-based methods [12], [13], [14], [10] extract point cloud features layer by layer, thus rely on sophisticated downsampling strategies, e.g., Distance-Farthest Point Sampling (D-FPS) and Feature-Farthest Point Sampling (F-FPS) [13], [14], [10], to obtain center points. But computational costs of these strategies are too expensive to afford when applied for large-scale 3D object detection. As shown in Table I, On large datasets like Waymo [11], when the number of input points reaches 100k, it takes 627.7ms for D-FPS to sample 16384 points, which severely hinders the applicability of point-based models on tackling large-scale point cloud tasks. While for 3D object detection, to the best of our knowledge, existing works have not provided a sampling strategy that attends efficiency and effectiveness simultaneously.

Moreover, as an object-oriented task, 3D object detection does not require overly dense representations of background context. SASA [16] employs MLP to encode point features, combing with FPS to increase the number of instance points. IA-SSD [10] utilizes MLP to replace the last two layers of FPS for center point selection, in order to improve the recall rates of instance targets. Nevertheless, current methods depend on complicated downsampling strategies or the preceding SA layers to extract neighboring features, for differentiating foreground and background points. And such operation includes a large amount of inefficient computations in the first a few SA layers.

We propose a novel Instance-Centroid Faster Point Sampling (IC-FPS) framework, in order to solve the problem of low efficiency of point-based models in large-scale point cloud scenes and greatly improve baseline performance. IC-FPS consists of three base modules: Neighboring Feature

TABLE I
LATENCY COMPARISON OF D-FPS AND THE PROPOSED CISS WHEN THE NUMBER OF SAMPLING POINT IS 16384.

input point number	25000p	50000p	100000p
D-FPS	171.9ms	322ms	627.7ms
CISS(Ours)	17.4ms	26.2ms	43.7ms

Diffusion Module (NFDM), Background Stripping Module (BSM) and Centroid-Instance Sampling Strategy (CISS). NFDM is fast and efficient, hence used to aggregate neighboring features of blocks, which effectively helps BSM to separate foreground and background blocks. And only the foreground region is processed, ineffective computation of background region is avoided. For better separation of the foreground and background, we propose a Density-Distance Focal Loss to ensure that the sparse foreground points at distant can be sampled effectively. After obtaining the foreground point, the proposed CISS replaces the FPS to sample center point. CISS dramatically improves the overall inference speed, and empowers the point-base model to detect targets in large-scale point cloud scenes. In addition, we utilize centroid point offset loss to offset the sampled centroid point to the nearest real point cloud location, such that the original geometric structure information of the point cloud can be better extracted.

We validate the proposed IC-FPS on multiple large-scale benchmarks and prove that the proposed framework effectively alleviates inefficiency issue of the first SA layer. As shown in Figure 1, IC-FPS improves baseline performance and inference speed by a large margin on Waymo dataset, and demonstrates its effectiveness and efficiency.

To summarize, our contributions are listed as follows,

- We propose Instance-Centroid Faster Point Sampling (IC-FPS) framework for point-based 3D detection, which achieves efficient and accurate detection in large-scale point cloud scenes when inserted to existing point-based methods.
- We propose a Neighboring Feature Diffusion Module (NFDM) to efficiently aggregates local features of blocks, which provides meaningful help for the Background Stripping Module (BSM).
- We propose Centroid Instance Sampling Strategy (CISS) to realize real-time inference of point-based models in large-scale point cloud scenarios, as an efficient alternative of complicated downsampling strategies such as FPS.
- Extensive experiments on multiple large datasets have demonstrated effectiveness and superiority of IC-FPS.

II. RELATED WORKS

Due to the intricate properties of point clouds, researchers attempt to project point clouds to multi-view or regular voxel grids to represent features [7], [8], [9], [32], [31], [33]. But these static projection methods result in information loss. Point-based methods directly use raw point cloud information as the input, and aggregate global features of point cloud from the top to the bottom.

A. Multi-view Based Methods

Early works project unstructured point clouds to multiple 2D views, for the convenience of direct use of convolution operations. MVCNN [28] converts multiple views to global features via max pooling layers, inevitably leading to massive information loss. MV3D-Net [30] only uses top view and front view for feature extraction, which preserves primary feature information and reduces computation cost.

B. Voxel-based Methods

In order to process unstructured 3D point cloud, voxel-based 3D detectors convert point clouds to regular voxel grids, such that the commonly used convolutions can be applied. VoxelNet [5] voxelizes the point cloud, and employs a block feature encoding layer to aggregate global and local information. However, computation and storage cost of 3D convolution increase along with resolution and bring unaffordable burden. SECOND [6] alleviates this issue by introducing 3D sparse convolution to substitute traditional convolution, and effectively optimizes memory usage and computation speed. PointPillars [1] further improves detection speed. It simplifies voxel to pillar with two dimensions, projects features to bird's eye view and applies 2D convolutions to extract deep features. SA-SSD [18] employs segmentation and center point prediction to facilitate model for further extraction of structural information.

C. Point-based Methods

Different to voxel-based methods, point-based methods use original information as the input, and adopt top-down learning to extract unstructured features of point cloud. Existing point-based methods normally adopt architectures similar to PointNet++ [13], which aggregates features by using symmetric aggregation function. PointRCNN [19] is the first 3D object detection model based on the original point cloud. It uses foreground segmentation network to obtain valid points for detection box regression. 3DSSD [14] is a single-stage detection framework that combines advantages of D-FPS and F-FPS. IA-SSD [10] uses FPS and instance-aware downsampling modules to extract features point by point, and utilizes contextual clues around bounding box to predict centroids. Nevertheless, existing methods cannot fully achieve fast and accurate downsampling. Constrained by complicated strategy in SA layer, existing methods are infeasible for large-scale 3D object detection.

The proposed IC-FPS framework differs from the aforementioned methods. It circumvents the disadvantage that SASA and IA-SSD are unable to sample in the first SA layer, and directly optimizes the first downsampling that is the most time-consuming, thus substantially improves baseline efficiency.

III. METHODS

A. Overview

Instance-Centroid Faster Point Sampling (IC-FPS) framework proposed in this paper consists of Neighboring Feature

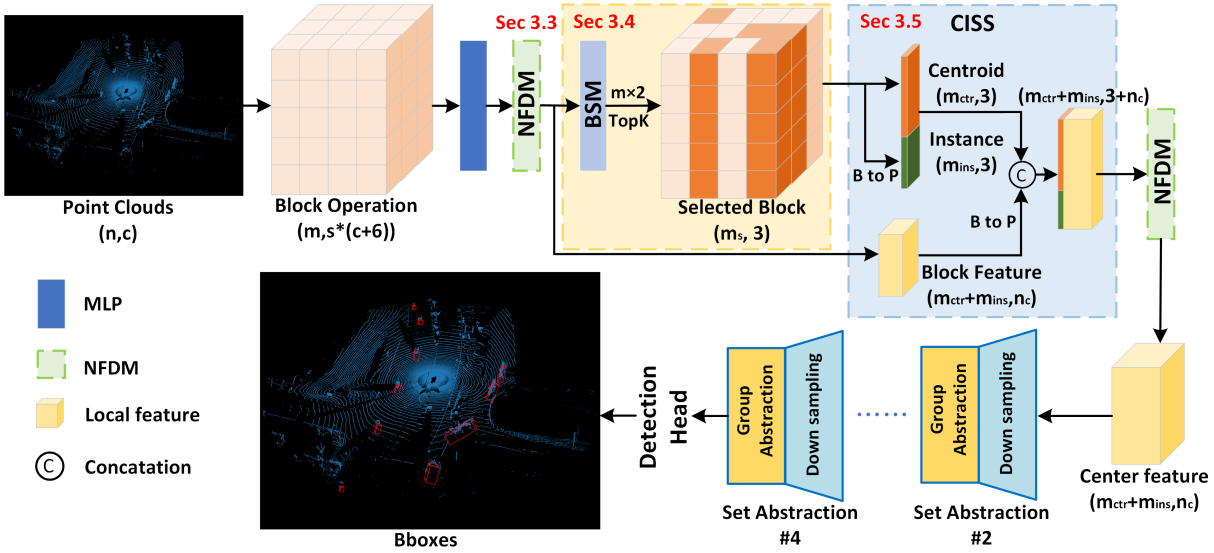

Fig. 2. Diagram of IC-FPS framework. B to P means all points in the block are selected. n represents the input number of point cloud, m the number of valid blocks with points present in the rage, m_s the number of selected blocks, m_{ctr} the number of selected centorid points and m_{ins} the number of selected Instance points. Blocks colored in orange are selected as the foreground blocks. c and n_c represent the number of input channel and feature channel respectively.

Diffusion Module (NFDM), Background Stripping Module, and Centroid Instance Fusion Sampling Strategy (CISS). As shown in Fig. 2, the first NFDM (Sec. III-C) extracts neighboring features of block to ensure that the background stripping module (Sec. III-D) can effectively separate the foreground and background blocks. CISS (Sec. III-E) samples the centroid point and instance point in the foreground block , and combines them as the center point of the output of the first downsampling layer. The second NFDM further expands the diffusion range of foreground block features, reducing the information loss caused by downsampling. Our proposed IC-FPS can be plug-and-played into arbitrary point-based 3D object detection models.

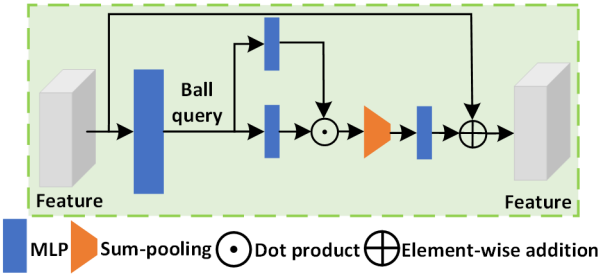

Fig. 3. Diagram of NFDM.

B. Instance-Centroid Faster Point Sampling

Given a set of points $\mathbf{P} = \{p_i \mid i = 1, \cdots, N\} \in \mathcal{R}^{n \times c}$, n represents the number of input points, and c represents the number of channels. As shown in Figure 2, we partition the set of points $\mathbf{P}$ into blocks and derive a matrix of size $[m, s, c]$, where m is the number of valid blocks (including original points), s is the number of points in block. For efficient extraction of local features in each block, we employ the PointPillars [1] method to acquire relative positional information within blocks. And the matrix size is folded to $[m, s \times (c + 6)]$, where the additional six dimensions include relative distances between points to center points and centroid positions in each block.

Valid classification of foreground and background blocks require features of neighboring blocks to be included in each block, while MLPs merely extract independent block features. NFDM is designed to efficiently extract the neighboring feature between blocks, and ensure the performance of the background stripping module. Features extracted by NFDM are used to evaluate each block. Background stripping module calculate the confidence of each block. The higher the confidence, the more likely it is a "foreground block". We select the blocks with confidence greater than 0.45 as "foreground blocks", whose features are characterized by $\mathbf{F}_v \in \mathcal{R}^{m_s * n_c}$. n_c is the number of channels and m_s is the number of foreground blocks. All points in the foreground block are considered as "foreground points".

Subsequently, we only feed the foreground blocks into CISS. Based on the confidence and distance to the origin, m_{ctr} centroid points and m_{ins} instance points are sampled. Based on the mapping relation between blocks and points, block features are mapped back to the point cloud, which helps the network to acquire multi-scale features. To avoid information loss due to downsampling, we use another NFDM to further expand the diffusion range of foreground block features. IC-FPS greatly improves the speed of the first SA layer, and its output can be well adapted to the second SA layer in various point-based networks.

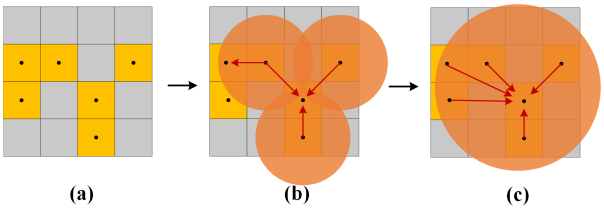

Fig. 4. Operation details of NFDM. (a) represents block features, (b) represents feature diffusion range of valid block in the first NFDM, (c) represents feature diffusion range of valid block in the second NFDM. Yellow regions are valid blocks, orange regions represent diffusion range of features. Each block uses ball query to obtain its neighborhood, and diffuses its features to other valid blocks in the neighborhood.

C. Neighboring Feature Diffusion Module

We propose Neighboring Feature Diffusion Module to efficiently extract point cloud neighboring features. The Diagram of NFDM is shown in Figure 3, different from RandLA-Net[27], we use multi-scale ball query instead of KNN, which effectively accelerates NFDM.

As shown in Figure 4, after obtaining features of each block, We use two ball query with different sampling radius to query the neighboring blocks of each block. Neighboring features are diffused to other blocks in its neighborhood, which makes each block contain features from its neighboring blocks. Besides, connecting two NFDMs further expands receptive field of the model, and helps to mitigate the information loss caused by downsampling.

D. Background Stripping Module

In order to reduce redundant computation, neighboring features obtained from the first NFDM are fed into Background Stripping Module to separate the foreground and background blocks. But it is found that network training with focal loss tends to sample blocks with high density of the point cloud. To better sample the foreground regions at distant with sparse point cloud, we propose a normal distribution based Density-Distance Focal Loss L_{DDFL}. Distant object instances are effectively prevented from being filtered out during training. Density constraint M_{Den} assigns various weights according to point density in the block.

$$M_{Den} = \frac{1}{\sqrt{2\pi\sigma}} exp\left(-\frac{(\frac{N_v}{N_{Max}} - \mu)^2}{2\alpha^2}\right) \times \sqrt{2\pi\sigma}, \quad (1)$$

where μ and σ are position and scale parameters of normal distribution. N_v and N_{max} represent the number of valid points in block and the maximum value, respectively. Distance constraint M_{Dis} assigns more weights on objects at a distance, and is expressed as follows,

$$M_{Dis} = \frac{exp\left(\frac{D}{M_D}\right)}{e}, \quad (2)$$

where D is the distance between the point and the origin in the coordinate system, M_D is the distance from the farthest point to the origin, e is a natural constant. And Density-Distance focal loss L_{DDFL} is written as follows,

$$L_{DDFL} = (1 - p_t)^{\gamma} log\,(p_t) \times M_{Den} \times M_{Dis}, \quad (3)$$

where p_t represents the predicted probability of ground truth, γ is an adjustable factor. M_{Dis} and M_{Den} are the distance constraint and density constraint, respectively.

E. Centroid-Instance Sampling Strategy

CISS aims to sample center points with high efficiency, and replace FPS in the first SA layer. We believe that reasons for the success of FPS are twofold. One is that FPS adaptively selects center points according to point density, i.e. more center points are selected in regions of high density. The other is that FPS samples from the raw point cloud and preserves geometric structural information, which is beneficial to improve accuracy of the subsequent detection box regression. Therefore we add some foreground points to center points, for increasing sample density of instance objects. A centroid point offset module is constructed to restore the raw geometric structure of point cloud.

1) Raw Instance Points Sampling.: After foreground blocks are derived, centroid positions of blocks are calculated and noted as $\mathbf{D}_i \in \mathcal{R}^{m_s \times 3}$. We sort centroid points according to classification confidence from the highest to the lowest, and select the highest m_{ctr} points. In addition, we select m_{ins} points that have shortest distances to the origin. Both instance points and centroid points are regarded as the center points for the first SA layer in subsequent models. Algorithm 1 is the detailed procedure of CISS.

2) Block Centroid Points Offset.: Adding positional information of the original point cloud to the subsequent network layer by layer helps adapting to the original point cloud structure. However there exist deviations between centroid and actual point cloud position. Direct use of centroids might cause losing the original positional information, and the model to fail in predicting accurate size of bounding box during regression. Consequently, we propose a centroid point offset module that moves centroid to the nearest instance point, for effective restoration on original size of targets. And the centroid point offset loss function L_{CB} is written as follows,

$$L_{CB} = SmoothL1\left(\lfloor b \rfloor, \lfloor \tilde{b} \rfloor\right) \quad (4)$$

where b is the predicted offset between the centroid and its nearest instance point, $\tilde{b}$ represents the actual offset of the centroid point to the nearest instance point. $\lfloor \cdot \rfloor$ denote whether centroid is present in the ground truth box or not.

F. Total loss

The proposed IC-FPS framework can be integrated with other models for end-to-end training. Multiple loss functions are combined for optimization. Total loss includes Density-Distance focal loss L_{DDFL}, centroid point offset loss L_{CB}, classification loss L_{cls} and bounding box generation loss L_{box}, as given in Equation 5.

$$L_{total} = L_{DDFL} + L_{CB} + L_{cls} + L_{box} \quad (5)$$

Algorithm 1 Centroid-Instance Sampling Strategy (CISS)

Input: Raw point cloud position $\mathbf{P} \in \mathcal{R}^{n\times 3}$, foreground block features $\mathbf{F}_v \in R^{m_s\times n_c}$ and its index $\mathbf{I}_f \in \mathcal{R}^{m_s\times 1}$, hashtable that maps block to point $HashMap()$. m_{ctr}: the number of selected centroid points; m_{ins}: the number of selected instance points.

Output: center point feature $\mathbf{F}_C \in \mathcal{R}^{(m_{ctr}+m_{ins})\times(3+n_c)}$.

1: Calculate centroid point position of each foreground block and select m_{ctr} centroid points $\mathbf{P}_{ctr} \in \mathcal{R}^{m_{ctr}\times 3}$ by distance.

2: Search the position information of foreground points by using index and hashtable $\mathbf{P}_{fore} \in \mathcal{R}^{m_s\times 3} \leftarrow HashMap(\mathbf{P}, \mathbf{I}_f)$. Select m_{ins} instance points $\mathbf{P}_{ins} \in \mathcal{R}^{m_{ins}\times s\times 3}$ from $\mathbf{P}_{fore}$ by distance.

3: Search correspondent features of centroid points and obtain the centroid features $\mathbf{F}_{ctr} \in \mathcal{R}^{m_{ctr}\times(3+n_c)}$ by merge centroid points and its correspondent features

4: Search correspondent features of instance points and obtain the instance features $\mathbf{F}_{ins} \in \mathcal{R}^{m_{ins}\times(3+n_c)}$ by merge instance points and its correspondent features.

5: Derive center point feature by merging centroid features and instance features : $\mathbf{F}_C \in \mathcal{V}^{(m_{ctr}+m_{ins})\times(3+n_c)} \leftarrow Concat(\mathbf{F}_{ctr}, \mathbf{F}_{ins})$.

IV. Experiments and Discussions

A. Implementation Details

We choose 3DSSD/SASA/IA-SSD [10] as baselines to construct our model. Firstly we partition the input point cloud into blocks. Block size is set to $[0.075, 0.075, 1]$. The diffusion radius of the first NFDM is set to 4.0 and the maximum number of diffusion points is set to 16. We set $\mu = 0.5$ and $\sigma = 0.5$ in the DDFL (Equation 3). Centroid offset module in CISS contains two MLP layers with size of $(16, 3)$. The two diffusion radii of the second NFDM were set to 0.2 and 0.8 and the maximum number of diffusion points is set to 16.

The same training strategy and model structure in each baseline is adopted in our experiment, except that the first SA layer is replaced by IC-FPS. In IA-SSD experiments, batch size is set to 8, learning rate is set to 0.01, Adam[22] optimizer is employed with weight decay set to 0.01. In 3DSSD and SASA experiment, batch size is set to 2, learning rate is set to 0.002. All experiments are conducted on NVIDIA A40 GPU and AMD EPYC 7402 CPU.

B. State-of-the-art Comparison

We configure three IC-FPS with different number of samples, which are IC-FPS-S/IC-FPS-M/IC-FPS-L. And the maximum number of sampled centroids and instance point are set to (16384, 2048), (26000, 4096) and (30720, 8197). In the last three SA layers, the number of samples are set to 1024, 2048 and 4096. IoU threshold is 0.25.

1) 3D Detection on Waymo: To validate performance of IC-FPS on large-scale point cloud scenes, we conduct quantitative experiments on Waymo [11] dataset. Waymo dataset [11] contains 160K samples in training set and 40K samples in validation set with two difficulty levels of challenge, $Level1$ and $Level2$.

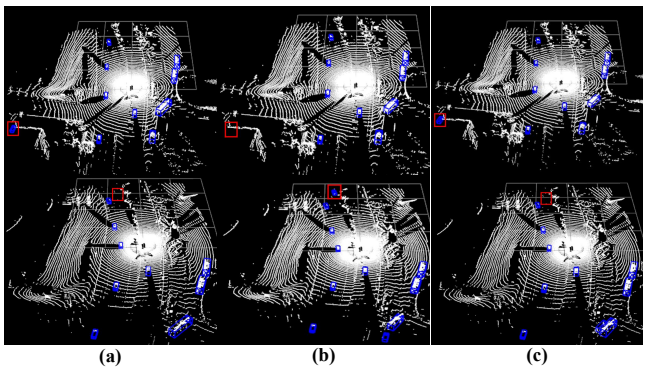

Fig. 5. Visulization results of Waymo dataset $Vehicle$. (a) ground-truth, (b) IA-SSD, (c) IC-FPS+IA-SSD. Red boxes represent differences of various results.

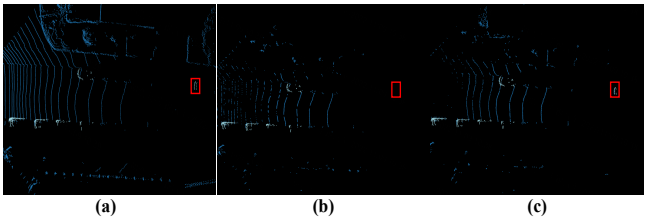

Fig. 6. Visualisation results before and after DDFL. (a) original point cloud input, (b) Without DDFL, (c) With DDFL. White dots are foreground points. Bounding boxes in red are the difference.

Table II demonstrates that the proposed IC-FPS significantly improves the inference speed and accuracy of baseline model. Compared to the baseline model IA-SSD [10], on *Level1* IC-FPS(16384/2048) improves $0.35/0.24, 4.01/5.59, 0.53/1.17$ in terms of mAP/mAPH. Throughput is increased by 0.62 times. For IC-FPS(30720+8192), on *Level2* mAP/mAPH are improved by $(1.14/1.45, 4.42/5.36, 1.04/1.49)$. Inference speed is improved by 20%. IC-FPS also

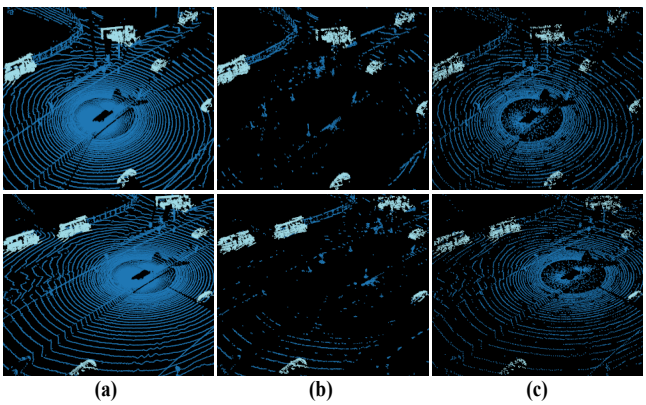

Fig. 7. Distribution of sample points by various sampling strategies, (a) original point cloud input, (b) CISS, (c) FPS. White dots are real instance points, blue dots are background points.

TABLE II

QUANTITATIVE COMPARISON EXPERIMENTS ON THE WAYMO *val* SET FOR 3D OBJECT DETECTION. EVALUATION METRICS ARE MEAN AVERAGE PRECISION (MAP) AND MAP WEIGHTED BY HEADING ACCURACY (MAPH). L1 AND L2 DENOTE *Level1* AND *Level2*. SPEED IS INFERENCE SPEED WHEN PROCESSING ONE FRAME OR FULL-LOADED GPU MEMORY. BOLD TEXTS ARE OUR RESULTS. FOR FAIR COMPARISON, SPEED ARE DERIVED BY REPRODUCING METHODS UNDER OPENPCDET FRAMEWORK [23]. ∗ INDICATES RESULTS AFTER REPRODUCTION OF OFFICIAL OPEN-SOURCE CODE.

Method	Veh.(L1) mAP/mAPH	Veh.(L2) mAP/mAPH	Ped.(L1) mAP/mAPH	Ped.(L2) mAP/mAPH	Cyc.(L1) mAP/mAPH	Cyc.(L2) mAP/mAPH	Speed
Pointpillars [1]	60.67/59.79	52.78/52.01	43.49/23.51	37.32/20.17	35.94/28.34	34.60/27.29	25.0
SECOND [6]	68.03/67.44	59.57/59.04	43.49/23.51	37.32/20.17	35.94/28.34	34.60/27.29	28.6
CenterPoint [24]	71.33/70.76	63.16/62.65	72.09/65.49	64.27/58.23	68.68/67.39	66.11/64.87	18.8
PV-RCNN [20]	74.06/73.38	64.99/64.38	62.66/52.68	53.80/45.14	63.32/61.71	60.72/59.18	3.7
$Part-A^2$ [25]	71.82/71.29	64.33/63.82	63.15/54.96	54.24/47.11	65.23/63.92	62.61/61.35	8.8
3DSSD* [14]	71.27/70.75	62.78/62.32	54.57/48.68	46.79/41.68	62.00/60.74	59.64/58.44	4.9
IC-FPS-M + 3DSSD	**71.72/71.23**	**63.16/62.71**	**56.57/49.66**	**48.51/42.52**	**65.03/63.76**	**62.56/61.34**	**6.7**
SASA* [16]	71.45/71.01	62.99/62.57	63.17/56.99	54.11/49.01	63.82/62.24	61.45/60.18	7.3
IC-FPS-S + SASA	**71.63/71.14**	**63.17/62.73**	**63.80/57.88**	**55.03/49.86**	**64.49/63.23**	**62.04/60.83**	**8.8**
IA-SSD [10]	70.53/69.67	61.55/60.80	69.38/58.47	60.30/50.73	67.67/65.30	64.98/62.71	15.5
IC-FPS-S + IA-SSD	**70.88/69.91**	**61.75/60.96**	**73.39/64.06**	**64.27/56.01**	**68.20/66.47**	**65.79/64.02**	**25.2**
IC-FPS-L + IA-SSD	**71.47/70.81**	**62.84/62.25**	**73.80/64.18**	**64.80/56.09**	**68.71/66.65**	**66.17/64.20**	**19.2**

TABLE III

QUANTITATIVE COMPARISON EXPERIMENTS ON WAYMO *val* SET FOR 3D OBJECT DETECTION. † INDICATES THAT THE WHOLE SCENE IS DIVIDED INTO FOUR PARTS TO ACCELERATE SA LAYER. BOLD TEXTS ARE OUR RESULTS AND THE BEST RESULTS ARE UNDERLINED.

Method	Veh.(L1) mAP/mAPH	Veh.(L2) mAP/mAPH	Ped.(L1) mAP/mAPH	Ped.(L2) mAP/mAPH	Cyc.(L1) mAP/mAPH	Cyc.(L2) mAP/mAPH	*FPS*
IA-SSD [10]	70.53/69.67	61.55/60.80	69.38/58.47	60.30/50.73	67.67/65.30	64.98/62.71	2.7
IA-SSD† [10]	70.38/69.82	61.33/60.19	68.23/58.44	60.11/50.33	66.84/65.06	64.32/62.61	8.3
IC-FPS-L + IA-SSD	**71.47/70.81**	**62.84/62.25**	**73.80/64.18**	**64.80/56.09**	**68.71/66.65**	**66.17/64.20**	**8.5**
IC-FPS-L† + IA-SSD	**71.19/70.48**	**62.62/61.99**	**72.93/63.11**	**63.72/55.21**	**67.22/65.49**	**64.75/63.09**	**12.6**

TABLE IV

INFERENCE SPEED COMPARISON EXPERIMENT OF WAYMO 3D OBJECT DETECTION.

Method	Para. (M)	*FPS*
3DSSD [14]	4.79	2.5
IC-FPS-M + 3DSSD	**4.84**	**3.8**
SASA [16]	4.83	2.6
IC-FPS-S + SASA	**4.88**	**4.0**
IA-SSD [10]	2.70	2.7
IC-FPS-S + IA-SSD	**2.74**	**12.9**
IC-FPS-L + IA-SSD	**2.74**	**8.5**

effectively boosts accuracy and speed of 3DSSD and SASA. IC-FPS(16384+2048) improves mAP/mAPH by $(10.12/8.95, 40.55/35.84, 38.13/36.73)$ on *Level2*, compared to PointPillars that has comparable throughput. Visual results can be found in Figure 5

We adopt the FPS acceleration method in IA-SSD[10], and partition the whole scene into four parts in Table III. Although this method can increase inference speed, it degrades the model performance to some extent. While after inserting IC-FPS, both model inference speed and model accuracy have obvious improvement.

Table IV reports inference speed of each baseline before and after IC-FPS. Since CISS avoids heavy computational burden of the first SA layer, IA-SSD-S inference speed is increased by 3.7 times after inserting IC-FPS-S. And it becomes the first point-based 3D object detection model that realizes real-time detection in large-scale point cloud dataset. As for 3DSSD and SASA, acceleration effect of IC-FPS is degraded because of multi-scale neighboring feature aggregation in the second SA layer.

2) 3D Detection on ONCE: ONCE (One millionN cSenEs) dataset [34] includes 1 million 3D scenes and 7 million corresponding 2D images. Recording duration of 3D scenes lasts for 144 driving hours. And it covers various weathers, traffic conditions, time and zones. We report our results on ONCE dataset in Table V. IC-FPS outperforms the baseline in all three categories.

3) 3D Detection on nuScenes.: nuScenes dataset contains 40K annotated keyframes with 23 object categories. mAP and NDS denote mean Average Precision and nuScenes detection score. In Table VI, We further conduct comparison experiments on nuScenes dataset [**?**] to verify the robustness of IC-FPS. When IC-FPS is inserted, the mAP and NDS of IA-SSD are improved by 0.2 and 0.3, and the inference speed is improved by 1.8 times.

4) 3D Detection on KITTI: KITTI dataset has three categories, car, pedestrian and cyclist. Each category contains three levels, "Easy", "Moderate" and "Hard". "Moderate" is usually taken for evaluation, mAP as primary evaluation metric. We report results of the Car category in the KITTI dataset in Table 8. We can find that after adding IC-FPS-S, the mAPs of 3DSSD and SASA are improved by 0.27/0.16/0.1 and 0.31/0.04/0.23, respectively, and the inference time is reduced by 7.7ms and 5.9ms. Acceleration effect in KITTI

TABLE V

QUANTITATIVE COMPARISON EXPERIMENTS ON ONCE VALIDATION SET. BOLD TEXTS ARE OUR RESULTS, UNDERLINED TEXTS ARE THE BEST RESULTS. PERFORMANCE INDEXES FOLLOW THE SAME CONFIGURATIONS GIVEN IN IA-SSD [10].

Method	Vehicle			Pedestrian			Cyclist			*mAP*
	0-30m	30-50m	>50m	0-30m	30-50m	>50m	0-30m	30-50m	>50m	
PointPillars [1]	80.86	62.07	47.04	19.74	15.15	10.23	58.33	40.32	25.86	44.34
SECOND [6]	84.04	63.02	47.25	29.33	24.05	18.05	69.96	52.43	34.61	51.89
PV-RCNN [20]	89.39	72.55	58.64	25.61	22.84	17.27	71.66	52.58	36.17	53.55
PointRCNN [19]	74.45	40.89	16.81	6.17	2.40	0.91	46.03	20.94	5.46	28.74
IA-SSD [10]	83.01	62.84	47.01	47.45	32.75	18.99	73.78	56.31	39.53	57.43
IC-FPS-L + IA-SSD	**82.73**	**64.47**	**48.75**	**47.64**	**32.57**	**20.51**	**75.64**	**57.65**	**38.14**	**57.81**

TABLE VI

QUANTITATIVE COMPARISON EXPERIMENTS OF POINT-BASED 3D OBJECT DETECTION MODELS ON NUSCENES *val* SET. BOLD TEXTS ARE OUR RESULTS AND BEST RESULTS ARE UNDERLINED. FOR FAIR COMPARISON, INFERENCE TIME AND PERFORMANCE EVALUATION METRICS ARE DERIVED BY REPRODUCING METHODS UNDER OPENPCDET [23].

Method	NDS	mAP	Car	Truck	Bus	Tra.	C.V.	Ped.	Motor	Bicy.	T.C.	Barrier	*FPS*
3D-CVF [26]	49.8	42.2	79.7	37.9	55.0	36.3	-	71.3	37.2	-	40.8	47.1	-
SASA [16]	61.0	45.0	76.8	45.0	66.2	36.5	16.1	69.1	39.6	16.9	29.9	53.6	1.9
3DSSD [14]	56.4	42.6	81.2	47.2	61.4	30.5	12.6	70.2	36.0	8.6	31.1	47.9	2.0
IA-SSD [10]	48.8	44.0	73.8	45.1	67.0	29.7	17.0	66.9	40.6	14.6	32.0	53.2	2.4
IC-FPS-L + IA-SSD	**49.1**	**44.2**	**74.8**	**45.7**	**66.2**	**29.8**	**16.8**	**70.5**	**40.7**	**14.8**	**29.4**	**53.6**	**6.8**

TABLE VII

EXPERIMENT RESULTS ON KITTI DATASET.

Method	Car (AP)			*FPS*
	Easy	Moderate	Hard	
Pointpillar[1]	82.58	74.31	68.99	-
SECOND[6]	84.65	75.96	68.71	-
Part-A2[25]	87.81	78.49	74.29	-
PV-RCNN[20]	90.25	81.43	76.82	-
3DSSD[14]	89.83	81.22	80.01	17.1
IC-FPS + 3DSSD	**90.10**	**81.38**	**80.11**	**19.7**
SASA[16]	90.34	83.91	81.08	17.8
IC-FPS + SASA	**90.65**	**83.95**	**81.31**	**19.9**

TABLE VIII

ABLATION STUDY ON IC-FPS. WE REPORT MAP VALUE OF WAYMO[11] DATASET ON $LEVEL1$, WHERE NFDM STANDS FOR NEIGHBORING FEATURE DIFFUSION MODULE, DDFL DENOTES DENSITY-DISTANCE FOCAL LOSS, AND CTR-OFFSET REPRESENTS CENTROID POINT OFFSET LOSS.

NFDM	DDFL	Ctr-offset	Veh.(L1) mAP	ped.(L1) mAP	Cyc.(L1) mAP
×	×	×	64.89	59.71	61.49
√	×	×	67.02	60.75	63.66
√	√	×	68.15	68.33	67.18
√	√	√	**71.47**	**73.80**	**68.71**
Improvement			**+6.58**	**+14.09**	**+7.22**

is limited because of insufficient number of input points. Another reason roots in the high speed of the first SA layer, making insignificant gain brought by IC-FPS.

C. Ablation Study

We conduct various ablation experiments on Waymo dataset [11]. As shown in Table VIII, we construct three different IC-FPS variants by ablating each proposed module.

TABLE IX

ABLATION STUDY OF IC-FPS VARIANTS USING DIFFERENT DOWNSAMPLING STRATEGIES IN THE FIRST LAYER. INS REPRESENT USING INSTANCE POINTS AS CENTER POINTS. CTR REPRESENT USING CENTROID POINTS AS CENTER POINTS. FPS REPRESENTS USING THE FARTHEST DISTANCE SAMPLING STRATEGY TO CALCULATE CENTER POINTS. *FPS* REPRESENTS THE MODEL INFERENCE SPEED WHEN BATCH SIZE IS SET TO 1.

Ins	Ctr	FPS	Veh.(L1) mAP	ped.(L1) mAP	Cyc.(L1) mAP	*FPS*
×	×	√	70.53	69.38	67.67	2.7
×	√	×	70.17	70.51	67.08	13.3
√	√	×	**70.88**	**73.39**	**68.20**	**12.9**
Improvement			**+0.35**	**+4.01**	**+0.53**	**+10.2**

Experiment results indicate that:

(1) NFDM alleviates the information loss caused by downsampling while enhancing model performance. (2) Adding distance and density constraints is beneficial for distinguishing between foreground and background blocks, especially tiny distant targets. (3) Retrieving raw point cloud geometry from blocks helps subsequent network to better capture scale information of instance object.

Figure 6 compares results before and after adding DDFL. We can find that distant targets are sampled, which effectively improves model recall rate.

Table IX reports ablation study of IC-FPS on various downsampling strategies. It is shown that Centroid-based downsampling significantly improves the inference speed of IC-FPS. But it does not refer to the density distribution of instance target, resulting in decreasing accuracy. Combining instance points with centroid points as sampling center not only improves the performance of baseline by 0.35, 4.01 and 0.53, but also realizes real-time detection (12.9 *FPS*).

As given in Figure 7, compared to FPS, CISS samples more foreground points while excluding most of the background points, which allows the model to better focus on the instance target.

V. CONCLUSIONS

In this paper, we propose an efficient IC-FPS framework for accelerating point-based 3D object detection model. It contains a fast center sampling strategy CISS, which effectively avoids huge computational burdens brought by FPS strategy in the first SA layer. Experiment results on Waymo datasets have demonstrated that our proposed IC-FPS framework is capable of improving baseline model performance and increasing inference speed significantly. Moreover, it is the first time that real-time detection of point-based model is realized in large-scale point cloud scenes.

REFERENCES

[1] Lang Alex H, Vora Sourabh, Caesar Holger, Zhou Lubing, Yang Jiong and Beijbom Oscar, Pointpillars: Fast encoders for object detection from point clouds, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, 12697–12705.

[2] Yang Bin, Luo Wenjie and Urtasun Raquel, Pixor: Real-time 3d object detection from point clouds, Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2018, 7652–7660.

[3] Zhou Yin, Sun Pei, Zhang Yu, Anguelov Dragomir, Gao Jiyang, Ouyang Tom, Guo James, Ngiam Jiquan and Vasudevan Vijay, End-to-end multi-view fusion for 3d object detection in lidar point clouds, Conference on Robot Learning, 2020, 923–932

[4] Lu Haihua, Chen Xuesong, Zhang Guiying, Zhou Qiuhao, M Yanbo and Zhao Yong, SCANet: Spatial-channel attention network for 3D object detection, ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, 1992–1996.

[5] Zhou Yin and Tuzel Oncel, Voxelnet: End-to-end learning for point cloud based 3d object detection, Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, 4490–4499.

[6] Y. Yan, M. Yuxing and L. Bo, Second: Sparsely embedded convolutional detection, Sensors, 2018.

[7] Zhao Na, Chua Tat-Seng and Lee Gim Hee, Sess: Self-ensembling semi-supervised 3d object detection, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, 11079–11087.

[8] Zheng Wu, Tang Weiliang, Chen Sijin, Jiang Li and Fu Chi-Wing, Cia-ssd: Confident iou-aware single-stage object detector from point cloud, Proceedings of the AAAI conference on artificial intelligence, 2021.

[9] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang and Houqiang Li, Voxel r-cnn: Towards high performance voxel-based 3d object detection, Proceedings of the AAAI Conference on Artificial Intelligence, 2021, 1201–1209.

[10] Zhan Yifan, Hu Qingyong, Xu Guoquan, Ma Yanxin, Wan Jianwei and Guo Yulan, Not all points are equal: Learning highly efficient point-based detectors for 3d lidar point clouds, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, 18953–18962.

[11] Sun Pei, Kretzschmar Henrik, Dotiwalla Xerxes, Chouard Aurelien, Patnaik Vijaysai, Tsui Paul, Guo James, Zhou Yin, Chai Yuning and Caine Benjamin, Scalability in perception for autonomous driving: Waymo open dataset, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, 2446–2454.

[12] Qi Charles R, S Hao, Mo Kaichun and Guibas Leonidas J, Pointnet: Deep learning on point sets for 3d classification and segmentation, Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, 652–660.

[13] Q Charles Ruizhongtai, Yi Li, Su Hao and Guibas Leonidas J, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, Advances in neural information processing systemsAdvances in neural information processing systems, 2017, 30.

[14] Yang Zetong, Sun Yanan, Liu Shu and Jia Jiaya, 3dssd: Point-based 3d single stage object detector, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, 11040–11048.

[15] Caesar Holger, Bankiti Varun, Lang Alex H, Vora Sourabh, Liong Venice Erin, Xu Qiang,, Krishnan Anush, Pan Yu, Baldan Giancarlo and Beijbom Oscar, nuscenes: A multimodal dataset for autonomous driving, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, 11621–11631.

[16] Chen Chen, Chen Zhe, Zhan Jing and Tao Dacheng, Sasa: Semantics-augmented set abstraction for point-based 3d object detection, AAAI Conference on Artificial Intelligence, 2022.

[17] Zhen Wu, Tang Weiliang, Jiang Li and Fu Chi-Wing, SE-SSD: Self-ensembling single-stage object detector from point cloud, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, 14494–14503.

[18] He Chenhang, Zeng Hui, Huang Jianqiang, Hu Xian-Sheng and Zhang Lei, Structure aware single-stage 3d object detection from point cloud, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, 11873–11882.

[19] S. Shaoshuai, W. Xiaogang and L. Hongsheng, Pointrcnn: 3d object proposal generation and detection from point cloud, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, 770–779.

[20] Shi Shaoshuai, Guo Chaoxu, Jiang Li, Wang Zhe, Shi Jianping, Wang Xiaogang and Li Hongsheng, Pv-rcnn: Point-voxel feature set abstraction for 3d object detection, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, 10529–10538.

[21] Noh Jongyoun, Lee Sanghoon and Ham Bumsub, Hvpr: Hybrid voxel-point representation for single-stage 3d object detection, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, 14605–14614.

[22] Kingma Diederik P and Ba Jimmy, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980, 2014.

[23] OpenPCDet Development Team, OpenPCDet: An Open-source Toolbox for 3D Object Detection from Point Clouds, https://github.com/open-mmlab/OpenPCDet, 2020.

[24] Y. Tianwei, Z. Xingyi and K Philipp, Centerbased 3d object detection and tracking, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, 11784–11793.

[25] Shaoshuai Shi, Wang Zhe, Shi jianping, Wang Xiaogang and Li Hongsheng, From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.

[26] Yoo J. H., Kim Y., Kim J. S. and Choi J. W., Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection, arXiv preprint arXiv:2004,12636,3, 2020.

[27] H Qingyong, Yang Bo, Xie Linhai, Ros Stefano, Guo Yulan, Wang Zhihua, Trigoni Niki and Markham Andrew, Randla-net: Efficient semantic segmentation of large-scale point clouds, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, 11108–11117.

[28] Su Hang, Maji Subhransu, Kalogerakis Evangelos and Learned-Miller Erik, Multi-view convolutional neural networks for 3d shape recognition, Proceedings of the IEEE international conference on computer vision, 2015, 945–953.

[29] We Xin, Yu Ruixuan and Sun Jian, View-gcn: View-based graph convolutional network for 3d shape analysis, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, 1850–1859.

[30] Chen, Xiaozhi and Ma, Huimin and Wan, Ji and Li, Bo and Xia, Tian, Multi-View 3D Object Detection Network for Autonomous Driving, arXiv preprint arXiv:2004,12636,3, 2020.

[31] David Schinagl, Georg Krispel, Christian Fruhwirth-Reisinger, Horst Possegger and Horst Bischof, GACE: Geometry Aware Confidence Enhancement for Black-box 3D Object Detectors on LiDAR-Data, arXiv preprint arXiv:2310.20319, 2023.

[32] Gang Zhang, Junna Chen, Guohuan Gao, Jianmi Li and Xiaolin Hu, HEDNet: A Hierarchical Encoder-Decoder Network for 3D Object Detection in Point Clouds, arXiv preprint arXiv:2310.20234, 2023.

[33] Se-Ho Kim, Inyong Koo, Inyoung Lee, Byeongjun Park and Changick Kim, DiffRef3D: A Diffusion-based Proposal Refinement Framework for 3D Object Detection, arXiv preprint arXiv:2310.16349, 2023.

[34] Mao Jiageng, Ni Minzhe, Jiang Chenhan, Lian Xiaodan, Li Yamin, Ye Chaoqiang, Zhang Wei, Li Zhenguo, Yu Jie and Xu Chunjiang, One million scenes for autonomous driving: Once dataset, 2021.