Just Add $100 More: Augmenting Pseudo-LiDAR Point Cloud for Resolving Class-imbalance Problem

Mincheol Chang[†,*]
Korea University
m1ncheoree@korea.ac.kr

Siyeong Lee[†]
NAVER LABS
siyeong.lee@naverlabs.com

Jinkyu Kim
Korea University
jinkyukim@korea.ac.kr

Namil Kim[‡]
NAVER LABS
namil.kim@naverlabs.com

Abstract

Typical LiDAR-based 3D object detection models are trained with real-world data collection, which is often imbalanced over classes. To deal with it, augmentation techniques are commonly used, such as copying ground truth LiDAR points and pasting them into scenes. However, existing methods struggle with the lack of sample diversity for minority classes and the limitation of suitable placement. In this work, we introduce a novel approach that utilizes pseudo LiDAR point clouds generated from low-cost miniatures or real-world videos, which is called Pseudo Ground Truth augmentation (PGT-Aug). PGT-Aug involves three key steps: (i) volumetric 3D instance reconstruction using a 2D-to-3D view synthesis model, (ii) object-level domain alignment with LiDAR intensity simulation, and (iii) a hybrid context-aware placement method from ground and map information. We demonstrate the superiority and generality of our method through performance improvements in extensive experiments conducted on popular benchmarks, i.e., **nuScenes**, **KITTI**, and **Lyft**, especially for the datasets with large domain gaps captured by different LiDAR configurations. The project webpage is https://just-add-100-more.github.io.

1 Introduction

LiDAR-based 3D object detection has garnered growing interest thanks to its wide applications, including fully autonomous vehicles (or robots) where the LiDAR point cloud is a main reliable source for 3D scene understanding. There exists a large volume of literature on LiDAR-based 3D object detection models [1–7], and they have achieved remarkable performance improvements via better model architectures and efficient LiDAR points representation. Still, we observe that these models often suffer from the so-called class imbalance problem – they tend to be overfitted to frequently observed objects

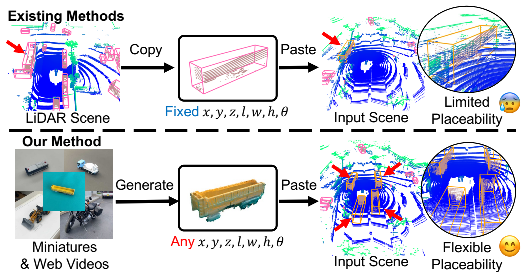


Figure 1: **Overview.** We present PGT-Aug, a novel cost-effective pipeline that generates and augments pseudo-LiDAR samples (from miniatures and web videos) to effectively reduce the performance gap between majority-class vs. minority-class objects.

[*]Work done during intern at NAVER LABS
[†]Equal contribution, [‡]Corresponding author

(of majority classes) and underfitted to rarely observed objects (of minority classes), causing a severe performance gap. Such an imbalance problem is commonly observed in real-world data collections, such as KITTI [8] and nuScenes [9].

A naïve solution for the imbalance problem is collecting more LiDAR data, but achieving a sufficient number of long-tail samples is still practically challenging. Due to the inherent imbalance in the natural distribution, the more data we collect, the worse the imbalance. In literature, one common strategy to deal with class imbalance is a copy-and-paste-based sample augmentation [1, 10–12], where objects (of minority classes) from other frames are copied into the current frame, balancing class-wise occurrences during training (see Fig. 1 top). However, these methods are still limited in that (i) the areas where objects can be pasted depend on their original position (thus limiting the positional diversity), and (ii) sample diversity where the same object (of minority classes) can be repeatedly copied and pasted.

An ideal sample augmentation method for balancing performance across classes may need the 3D shape information of diverse samples (of minority classes) to enable flexible placements into a given scene. Recent Neural Radiance Field (NeRF)-based approaches [13–16] have demonstrated the ability to generate high-quality 3D scenes from multiple viewpoints at low computational costs. Here, we advocate utilizing NeRF-based approaches to obtain the 3D shape of objects from external sources (not from other scenes within the same data). However, generating diverse real-world 3D objects, especially of minority classes, can pose challenges due to the need for images from multiple viewpoints. To address this, we propose using miniatures and public videos to obtain 3D-rendered samples of target objects. Miniatures, commonly used for practical effects in the film industry, offer a practical solution to collect diverse samples, particularly for various types of vehicles.

In this work, we introduce Pseudo Ground Truth Augmentation (PGT-Aug), which generates point clouds of minority-class objects from two sources: (i) surround-view videos of given miniatures and (ii) public videos of real-world objects. As shown in Fig. 1 bottom, we first utilize a 2D-to-3D renderer to reconstruct an object's 3D volumetric representation. Then, we transform it into LiDAR-like 3D point clouds by rearranging and filtering points and estimating their intensities. During training, such generated pseudo-LiDAR points are sampled and placed into appropriate places of a scene without where-to-paste technical constraints. Our experiments with public datasets (i.e. nuScenes [9], KITTI [8], and Lyft [17]) demonstrate that our data augmentation with pseudo-LiDAR points effectively improves detection performance for minority classes. Our contributions can be summarized into three-fold:

- We introduce PGT-Aug, a novel cost-effective pipeline for LiDAR-based object detectors to solve the class imbalance problem by (i) generating pseudo-LiDAR samples (from multi-view images of miniatures and public videos of an object) and (ii) augmenting them during training to balance the performance gap across classes.
- Our pipeline involves a novel view-agnostic pseudo-LiDAR sample generation where we reduce the domain gap between the real-world and the generated LiDAR point clouds. A series of processes, such as spatial distribution matching and data-driven intensity adjustments, achieve this.
- We propose a novel map-aware augmentation technique that determines where to paste the pseudo-LiDAR samples based on map-based scene context (i.e. placing an object into the appropriate locations in the given scene).

2 Related Work

LiDAR-based 3D Object Detection Datasets for Autonomous Driving. The automotive industry has shown a surge of interest in LiDAR-based 3D object detection, enhancing system reliability compared to camera-based detectors. However, creating large-scale annotations face challenges due to sensor costs, spatial distribution variations, and annotation difficulties caused by reflections and occlusions. While several annotated datasets [9, 17–20] exists, the limited quantity, compared to the image domain, leads to both generalization issues and class imbalance [21, 22]. To mitigate this, we propose a novel approach for generating high-quality rare objects inexpensively for data augmentation in 3D object detection, addressing the class imbalance. (see Fig. 7 in the Supplementary Material.)

Data Augmentation of LiDAR Point Clouds. Data augmentation plays a crucial role in increasing the diversity of training samples and addressing class imbalance in datasets. Two common strategies

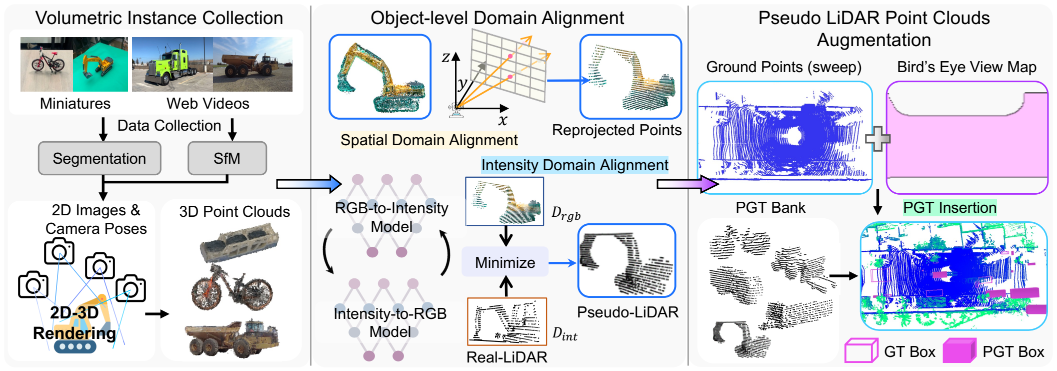


Figure 2: **Overview of Pseudo GT (PGT)-Aug Framework.** Given multiview images, we first reconstruct their volumetric representations (Section 3.1). We post-process RGB point clouds using spatial rearrangement and LiDAR intensity simulator (Section 3.2), producing pseudo-LiDAR point clouds. Such points are stored in a psuedo LiDAR bank, and we paste the sampled objects into the target scene with the proposed augmentation scheme (Section 3.3).

for mitigating overfitting and enhancing model robustness are data resampling and data synthesis. Data resampling [23–27] primarily focus on oversampling minority classes by replicating existing samples, thereby increasing the frequency of minority classes and balancing class distributions during model training. In contrast, data synthesis [28–30] relies on blending of original samples or utilizing synthetic data from generative models. This approach enhances the diversity of the training set, helping to address data scarcity and improve overall model generalization.

Recent works have tackled 3D annotation difficulties through LiDAR simulation [10, 31–33] for 3D renderings or data augmentation [1, 12, 34–37] for utilizing existing data. Some works [10, 32, 38] attempted to reduce a domain gap between simulated and real-world data by combining them, as seen in SHIFT3D [11], which employs deep signed distance functions to refine object's shape and pose adversarially. GT-Aug [1] used copy-and-paste strategies, while Real-Aug [12] enhanced data utilization with realistic placement. Creating diverse, non-standard shapes requires considerable time and effort through 3D CAD modeling. On the other hand, 3D objects in the target dataset are partially observed, making free placement challenging. Different from previous approaches, we collect miniatures and real-world objects to render realistic 3D samples with low cost from diverse domains, and introduce an object-level augmentation with flexible placeability.

Neural Radiance Fields and 3D Rendering. Many works have attempted to represent 3D scenes as continuous implicit representations or differentiable structures [13, 14, 39–41]. NeRF [13] represented 3D geometry by approximating density and view-dependent RGB using a simple MLP architecture. Among many subsequent works [14–16, 42, 43], both Instant-NGP [15] and Plenoxels [14] contributed to faster training and rendering of 3D scenes. Instant-NGP accelerated MLPs using multilevel hash tables, while Plenoxels utilized sparse voxel grids for interpolating color and density field. These advances can make the reconstruction of 3D objects easier and more precise [16], facilitating their use in various perception tasks [44].

3 Method

We introduce PGT-Aug, a fast, realistic, and low-cost pipeline that generates (and collects) pseudo-LiDAR point clouds of minority-class objects from multi-view images of miniatures or real-world objects. Then, to reduce the class imbalance problem, the generated point clouds are augmented into the current scene, balancing the number of occurrences across classes during training. As shown in Fig. 2, our pipeline consists of three main modules: (i) **Volumetric 3D Instance Collection**, where we reconstruct objects' 3D volumetric representation from multi-view images of real-world miniatures or objects (Section 3.1). (ii) **Object-level Domain Alignment**, where we reduce the domain gap (i.e., between the generated vs. the collected from LiDAR sensors) by transforming the point clouds based on sensor configurations and intensity simulation models (Section 3.2). Lastly, (iii) **Pseudo LiDAR Point Clouds Augmentation**, where we augment the generated pseudo-LiDAR point clouds into the current scene by determining more realistic object insertion areas based on the estimated ground areas and map information (Section 3.3).

3.1 Volumetric 3D Instance Collection of Minority Class Objects

Data Collection. Conventional Ground truth (GT) sampling-based data augmentation approaches resample minority-class 3D instances (e.g., trucks, trailers, buses, construction vehicles) from in-domain collections, but their placement flexibility and contextual diversity are often limited. To resolve this issue, we want to augment minority-class 3D instances from out-of-domain sources, efficiently resolving the data imbalance problem by sampling more diverse types and shapes of minority-class samples in the wild. To this end, we advocate for utilizing the following two cost-effective sources: (i) videos capturing the surround view of miniatures and (ii) publicly available videos capturing multiple views of minority-class objects (see Fig. 2). As the title suggests, we purchased dozens of realistic miniatures *for just $100*. Plus, we collect public videos from YouTube, followed by manual filtering to ensure those videos capture an object from multiple viewpoints.

Preprocessing. Given collected video frames, we first estimate the following three pieces of information as a preprocessing for the later 2D-to-3D rendering: (i) a camera intrinsic matrix, (ii) camera poses (3D camera position and its orientation for each frame), and (iii) binary masks for a foreground object. For (i) and (ii), we use COLMAP [45], similar to the preprocess of existing NeRF-based approaches [13, 14, 46]. For (iii), we use the off-the-shelf video segmentation model, i.e., Segment and Track Anything [47], to extract segmentation masks for foreground objects.

2D-to-3D Rendering. Our framework is built on advanced 2D-to-3D renderers, such as Plenoxels [14] and Gaussian-Splatting [16], known for efficient, high-quality 3D reconstruction. Unlike some 3D conditional generative models for specific classes such as Shap-E [48] and Zero-1-to-3 [49], these methods offer dense 3D point clouds suitable for increasing out-of-distribution samples. In general, these methods [14, 16] predict view-dependent representation, but obtaining fully visible and uniformly high-density data from all viewpoints is necessary to generate pseudo-LiDAR objects and axis-aligned bounding boxes. Therefore, we propose an ad-hoc module to obtain representative colors for each voxel grid or point based on the estimated mean color values from different views. More specifically, iterating through N views, it determines which voxel grid or point's camera ray passes through and calculates the color value of voxels or points by doing a dot product between corresponding spherical harmonic coefficients and view-dependent harmonics basis. If a certain voxel or a point is hit multiple times by rays from different views, we update the existing value by adding the new color value and store the total number of passes. Notably, this straightforward module is applicable not only to 2D-to-3D rendering techniques such as Plenoxels and Gaussian Splatting, but also to renderers that do not rely on spherical harmonics[2].

3.2 Object-level Domain Alignment

We aim to augment real LiDAR data with simulated minority-class samples derived from RGB-colored point clouds. However, these point clouds lack crucial information regarding spatial distribution from sensor configurations and LiDAR's intensity, which is essential for accurate LiDAR modeling and evaluation. Thus, we propose *object-level domain alignment* between RGB-colored point clouds $\mathcal{D}_{rgb}$ and real-world point clouds $\mathcal{D}_{int}$, employing two alignment techniques as (i) view-dependent points filtering and rearrangement and (ii) LiDAR intensity simulation.

View-dependent Points Filtering. This step simulates realistic data variations based on the object's relative position to the LiDAR sensors and their settings. Therefore, we require alignment parameters based on factors such as type, position, quantity, and specifications (Field of View and azimuth resolution) of LiDAR sensors in the target dataset. We apply the following three steps: (i) **transformation** of points to the range view representation, (ii) **filtering** of points on the invisible side, and (iii) **reprojection** of points into 3D space.

[2]If a renderer with spherical harmonic lighting is selected, the 0th coefficient of the spherical harmonics can be directly used to extract view-agnostic colors. However, we empirically observed that our approach is more robust at capturing the original object's color and areas of dark shades (see Table 13 in Appendix).

In (i), given points in the spherical coordinates system, we map points into a range view $\mathbf{R}(u, v) \in \mathbb{R}^{H \times W}$, where u and v are the spatial grid indexes. For each grid (u, v), a point with minimum depth remains, and the others are filtered out (i.e., points on visible parts remain). The remaining points are irregularly located in 3D space, generally different from those of real-world LiDAR points. We therefore rearrange each point to be regularly spaced by adjusting their inclination ϕ and azimuth θ in the spherical coordinate system: $\phi' = \left[1 - \left(\frac{v+0.5}{H}\right)\right] \text{FOV}_{\text{total}} - \text{FOV}_{\text{down}}, \theta' = \pi\left[2\left(\frac{u+0.5}{W}\right) - 1\right]$, where FOV_{down} and $\text{FOV}_{\text{total}}$ represent the down part and the total range of the field of view, respectively. Rearranged inclination and azimuth ϕ', θ' are calculated by backprojecting image coordinates u, v to spherical coordinates following the above equations. The results on different LiDAR settings are shown in Fig. 3.

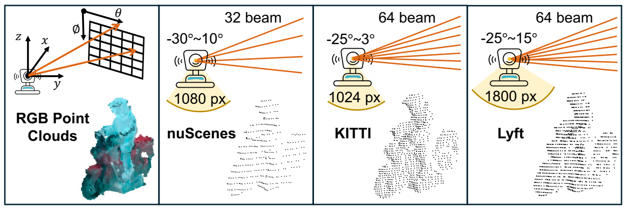


Figure 3: **Effect of Reprojection on Different Datasets.**

LiDAR Intensity Simulation. Since object surface reflectivity cannot be derived from images, we use a data-driven model for intensity estimation. As no real-world LiDAR points match generated RGB points, we create an unpaired domain transfer model specifically for point clouds based on CycleGAN [50] framework. Also, we design a novel region matching loss that directly reduces the intensity error between two samples, even with varying point counts, as shown in Fig. 4. To compute the loss, both generated points $G_{\mathcal{D}_{rgb} \to \mathcal{D}_{int}}(\mathbf{p}_{rgb})$ and real-world points $\mathbf{p}_{int}$ are grouped into an equal number of ball patches whose centers are obtained via farthest point sampling. After dividing into N ball patches, Hungarian matching is used to find the optimal assignment of ball patch pairs by computing the center distances for all matchings $\mathfrak{S}_N$,

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\min} \sum_{i}^{N} ||c_i - c_{\sigma(i)}||_1, \tag{1}$$

where c_i and $c_{\sigma(i)}$ are centers of ball patches from $G_{\mathcal{D}_{rgb} \to \mathcal{D}_{int}}(\mathbf{p}_{rgb})$ and $\mathbf{p}_{int}$, respectively. After finding optimal pairs, we reduce all intensity distances between fake and real pairs of patches. For normalized points x, y and a given ball radius r,

$$\mathcal{L}_{group} = \sum_{j}^{N} ||\mathbb{E}_{x \in b_j^r}(x) - \mathbb{E}_{y \in b_{\hat{\sigma}(j)}^r}(y)||_1, \tag{2}$$

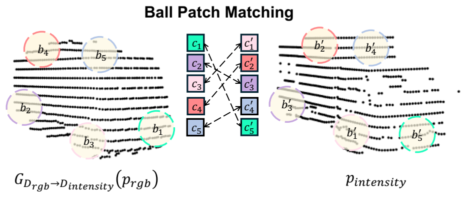


Figure 4: **Region Matching Loss.**

where $\mathbb{E}_{x \in b_j^r}(x)$, $\mathbb{E}_{y \in b_{\hat{\sigma}(j)}^r}(y)$ denote the average of intensity values of optimal ball patch pairs $b_j^r, b_{\hat{\sigma}(j)}^r$ and r is 0.1. The overall objective function is

$$\mathcal{L}_{total} = \mathcal{L}_{\text{CycleGAN}} + \lambda \mathcal{L}_{\text{group}}, \tag{3}$$

where λ is a regularizing factor set to 0.1.

Instance Size Setting. We analyzed the average object size per class in the dataset. We determine the size by adding Gaussian noise with σ of 0.1. The noise exceeding one σ range was clipped to σ.

3.3 Pseudo LiDAR Point Clouds Augmentation

Ground and Map Synthesis for Object Insertion. To determine feasible insertion areas, Real-Aug [12] and Lidar-Aug [10] utilized estimated ground estimation. However, relying solely on this data may not cover areas adequately as shown in Fig. 5. To address this, we propose a method that combines map information and estimated ground areas for more realistic scene composition. We first create a rasterized map with a 0.128m per pixel resolution within a radius of 51.2m around the ego vehicle, assigning proper layouts (e.g., Road, Sidewalk, etc.) for inserted objects. Additionally, we construct a rasterized ground map based on estimated ground points. The two pieces of information may overlap at pixels where dynamic object are present or where ground area is not estimated due to

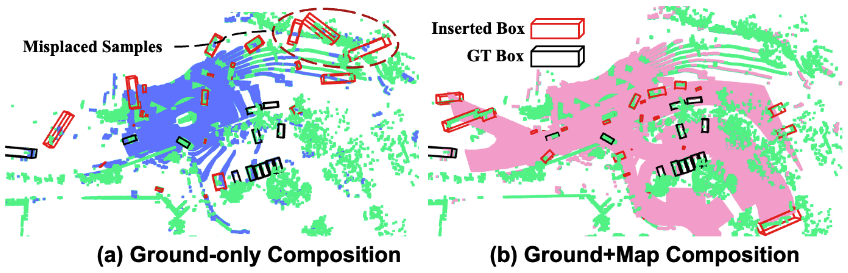


Figure 5: **Comparison of Ground-only and Ground+Map Scene Composition.** Blue and Pink-colored points denote the feasible location of insertion derived from (a) ground-only and (b) ground+map synthesized insertions, respectively.

insufficient points. To deal with it, we prioritize map values for pixels with low point density, while using ground values otherwise. As shown in Fig. 5-(b), the proposed method can predict broader and feasible areas for more realistic data augmentation than ground-only composition.

Aligning to Data Geometry. The generated objects need to be axis-aligned to easily obtain accurate bounding box annotations for the detection task. For this, we first use PCA to align the generated points and rotate them around the normal direction in xy, yz, and zx planes to align its axes. We also need to classify objects' front and back for heading information, so we adopt PointNet++ [51] as a heading classifier trained using binary cross-entropy loss. Since PointNet++ is rotation sensitive [52] and is trained without rotation augmentation, the model can distinguish between front and back.

Virtual Object Sweeps. LiDAR scan sweeping is commonly used to increase the point density in some popular benchmarks [8, 9]. Since our pipeline can generate objects that appear from a single scan, additional techniques are required to apply them to such datasets. We employ a rigid body motion model to stack points over time. Given the dataset, we first collect the velocity and acceleration of each class's center points at each time step and estimate a motion trajectory. We translate the center of generated object points along the selected trajectory. Ultimately, we can generate virtual objects that closely resemble what is acquired from the real LiDAR sweeps.

4 Experiments

Implementation Details. We use Plenoxels [14] to reconstruct a given object's 3D shape from multi-view images. Also, we utilize PointNeXt [53] in our CycleGAN-based LiDAR intensity simulator. Note that we sample 300 instances (that has at least 256 LiDAR points) for each minority class from the nuScenes dataset [9] to train our intensity simulator. We implement and evaluate our proposed

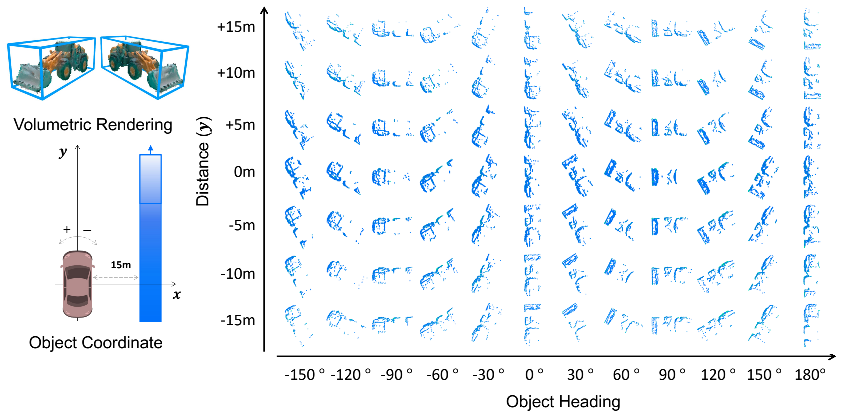


Figure 6: **Examples of generated pseudo-LiDAR point samples** with different orientations and ranges given reconstructed 3D volumetric representations.

Table 1: **Detection performance comparison on nuScenes *val* set in terms of AP, mAP, and NDS.** Based on CV-Voxel [5], we compare different placement methods, such as random [1], ground-based [12], and our placement. We also report the effect of using our pseudo-LiDAR samples.

GT-Aug [1] (Random Placement)		✓	✓	✗	✗	✗	✗
Real-Aug [12] (Ground-based Placement)		✗	✗	✓	✓	✗	✗
Ours (Ground+Map-based Placement)		✗	✗	✗	✗	✓	✓
w/ Pseudo-LiDAR Samples		✗	✓	✗	✓	✗	✓
Class	# of objects in *val*	Performance					
Bus	3,009	69.1	71.3 (+2.2%)	71.8	71.7 (-0.1%)	**73.0**	72.1 (-0.9%)
Construction Vehicle	2,387	18.0	18.1 (+0.1%)	22.7	24.0 (+1.3%)	23.7	**24.2** (+0.5%)
Trailer	3,765	36.1	35.7 (-0.4%)	41.6	41.4 (-0.2%)	**42.3**	40.4 (-1.9%)
Truck	13,950	56.9	57.8 (+0.9%)	57.3	58.3 (+1.0%)	58.8	**59.8** (+1.0%)
Motorcycle	2,227	60.3	61.5 (+1.2%)	67.1	68.0 (+0.9%)	68.7	**70.3** (+1.6%)
Bicycle	2,071	41.5	45.1 (+3.6%)	54.9	57.1 (+2.2%)	56.7	**58.3** (+1.6%)
mAP (*for all 10 classes*)		59.0	59.8 (+0.8%)	62.3	63.0 (+0.7%)	63.3	**63.5** (+0.2%)
NDS (*for all 10 classes*)		66.5	66.9 (+0.4%)	68.4	68.7 (+0.3%)	68.7	**69.1** (+0.4%)

method based on LiDAR-based 3D object detection implementations from OpenPCDet [54] with default parameter settings. All detectors are trained with a batch size of 32 on 4×A100 GPUs for 20 epochs. We conducted all experiments with the fixed seed for a fair comparison. Details on architectures and hyperparameters can be found in the Supplementary Material.

Pseudo Object Bank Details. To validate our system, we have chosen nuScenes [9] dataset. For the pseudo object bank, we generated minority class objects in 13 heading directions (from $-180°$ to $180°$ at intervals of $30°$) within the entire perception range (from -50m to 50m at intervals of 5 meters). During training, we discard objects with less than 16 points to prevent ambiguity between classes and add a small variation at the center. As a result, our bank has 36,960 trucks, 52,800 construction vehicles, 12,960 buses, 19,280 trailers, 25,279 motorcycles, and 4,300 bicycles.

Performance on NuScenes *val* set. We assess the effect of adding pseudo-LiDAR samples during training using CP-Voxel [5] with a voxel size of $[0.075, 0.075, 0.2]$. As shown in Table 1, the incorporation of pseudo-LiDAR samples into the GT database significantly improves the overall detection performance, including minority classes, in terms of mAP and NDS (compare 1st vs. 2nd, 3rd vs. 4th, and 5th vs. 6th columns). Our placement approach surpasses existing methods (compare 1st, 3rd vs. 5th columns). Combining our two methods achieves the best performance, with 69.1% in NDS and 63.5% in mAP, suggesting that our high-quality pseudo-LiDAR samples effectively improve detection performance for minority classes without compromising majority classes.

Performance on NuScenes *test* set. Lastly, we evaluate our model on nuScenes test set based on CP-Voxel [5] and Transfusion-L [55] as baseline detection models. We observe in Table 2 that our model consistently achieves the best NDS and mAP scores (compared to Real-Aug [12]). Notably, our method shows significant performance gains for minor classes such as Trailer, Truck, Motorcycle, etc. Note that we applied the same Test Time Augmentation (TTA) technique to CP-Voxel as their official submission to nuScenes leaderboard.

Table 2: **Detection performance comparison on nuScenes *test* set in terms of minority class AP, mAP for all 10 classes, and NDS**. †: our reproduction, ‡: test time augmentation enabled.

Model	Aug.	Minority classes						mAP	NDS
		Bus	C.V	Trailer	Truck	M.C	B.C		
CP-Voxel‡ [5]	GT-Aug	64.4	**31.0**	60.0	47.2	65.7	41.0	63.8	68.7
	Real-Aug†	64.5	29.0	60.1	57.3	72.2	47.1	65.8	71.3
	PGT-Aug	**68.1**	29.0	**61.7**	**57.7**	**74.0**	**48.6**	**67.1** (+1.3%)	**72.3** (+1.0%)
Transfusion-L [55]	GT-Aug	63.7	29.0	58.7	46.3	67.1	**44.2**	63.9	68.6
	Real-Aug†	64.3	**31.0**	60.0	47.3	65.7	41.0	63.8	68.7
	PGT-Aug	**67.3**	30.1	**60.2**	**56.9**	**68.2**	40.6	**65.1** (+1.3%)	**69.9** (+1.2%)

Table 3: **Comparisons between baseline and PGT-Aug for individual models on nuScenes *val* set.** *Abbr.* C.V: Construction Vehicle, Ped: Pedestrian, T.C: Traffic Cone, M.C: Motorcycle, B.C: Bicycle. †: our reproduction.

Model	Aug.	Majority classes				Minority classes						mAP	NDS
		Car	Ped	Barrier	T.C	Bus	C.V	Trailer	Truck	M.C	B.C		
SECOND [1]	GT-Aug	81.5	77.3	58.3	57.9	67.2	15.2	36.2	50.6	40.7	16.2	50.11	61.56
	Real-Aug†	84.5	**80.1**	61.8	**67.4**	**72.0**	24.1	**44.2**	58.7	61.5	36.6	59.09	67.23
	PGT-Aug	**84.8**	80.0	**62.2**	67.2	71.9	**24.4**	42.4	**58.8**	**64.0**	**38.1**	**59.37**	**67.30**
CP-Pillar [5]	GT-Aug	83.1	82.5	65.0	65.8	63.0	14.1	23.7	54.4	51.4	25.5	52.85	62.57
	Real-Aug†	82.9	**84.0**	65.7	**67.9**	64.9	19.5	26.3	**57.2**	64.6	46.2	58.03	64.85
	PGT-Aug	**83.3**	83.5	**65.7**	67.4	**66.0**	**22.1**	**27.5**	56.3	**65.0**	**47.0**	**58.39**	**65.49**
CP-Voxel [5]	GT-Aug	84.9	**85.4**	**68.3**	69.9	69.1	18.0	36.1	56.9	60.3	41.5	59.04	66.54
	Real-Aug†	84.7	85.0	67.5	70.0	71.8	22.7	**41.6**	57.3	67.1	54.9	62.27	68.39
	PGT-Aug	**85.4**	**85.4**	68.0	**71.1**	**72.1**	**24.2**	40.4	**59.8**	**70.3**	**58.3**	**63.52**	**69.11**
Transfusion-L [55]	GT-Aug	86.6	86.6	69.4	73.6	72.7	23.3	43.9	53.3	69.4	56.1	63.48	68.58
	Real-Aug†	86.6	**86.9**	**69.9**	73.6	73.1	25.9	42.2	**53.8**	68.3	55.1	63.54	68.57
	PGT-Aug	**87.0**	86.4	69.4	**74.0**	**73.4**	**26.7**	**46.9**	50.6	**71.3**	**56.3**	**64.20**	**68.83**
VoxelNeXt [6]	GT-Aug	83.7	84.5	**68.9**	68.4	71.4	20.9	37.5	56.1	62.9	49.8	60.42	67.03
	Real-Aug†	83.4	85.0	67.8	69.6	70.9	22.6	38.7	**58.0**	69.8	**56.7**	62.25	67.62
	PGT-Aug	**84.2**	**85.0**	67.7	**70.6**	**72.4**	**23.9**	**40.6**	57.5	**70.9**	56.6	**62.95**	**68.30**

Comparison with Different Model Architectures. We compared detection performance among GT-Aug, Real-Aug, and PGT-Aug using five baselines, as detailed in Table 3. All detectors were tested under same conditions, ensuring that no additional information about the generated objects was utilized during the evaluation process. PGT-Aug brings significant improvements over other augmentations in all types of models, such as center-based models (CP-Pillar, CP-Voxel [5]), anchor-based model (SECOND [1]), and other types (VoxelNeXt [6], Transfusion-L [55]). This confirms that our method can generally be applied to various detection models, boosting the model's accuracy for minority classes without sacrificing accuracy for majority classes.

5 Ablation Studies

Quality of Pseudo Labels. To evaluate the quality of our generated objects, we measure FID scores using SE(3)-transformer [56] trained on the nuScenes dataset. As shown in Table 4, our generated pseudo-LiDAR objects show similar or lower FID scores (than true samples), which may confirm their plausibility and high quality (compare 5th and 6th vs. 7th and 8th columns). We also observe that the pseudo-LiDAR object quality improves with (i) similar Azimuth resolution to the target dataset sensor, (ii) the use of RGB values additionally as input, and (iii) the regularization by group

Table 4: **Quality of Pseudo LiDAR Point Clouds.** FID scores (squared Wasserstein distance between given samples and nuScenes samples, thus lower is better) comparison between variants of our models and public LiDAR datasets, Lyft [17] and A2D2 [18]. Abbr. G.S: Gaussian Splatting

	Pseudo-LiDAR Point Clouds						Lyft [17]	A2D2 [18]
Volumetric 3D type	Plenoxels [14]					G.S [16]	-	-
Azimuth Resolution (px)	3600	1080	1080	1080	1080	1080	-	-
RGB features	✓	✗	✗	✓	✓	✓	-	-
Group intensity Loss	✓	✗	✓	✗	✓	✓	-	-
Bus	17.7	14.6	13.1	**13.0**	13.2	**11.2**	8.7	19.8
Construction Vehicle	**7.0**	7.5	7.6	7.5	7.6	**7.6**	-	6.0
Trailer	20.6	12.7	11.9	**11.9**	12.2	**13.7**	-	36.5
Truck	8.9	8.4	7.6	7.6	**7.3**	**6.9**	6.6	13.4
Motorcycle	20.7	2.5	7.0	7.2	**3.7**	**1.3**	3.0	10.1
Bicycle	9.0	3.3	2.2	2.4	**2.1**	**1.8**	1.8	0.7
Avg. FID Score	14.2	8.2	8.3	8.3	**7.7**	**7.1**	4.8	14.4
mAP (*for all 10 classes*)	63.40	63.44	63.48	63.41	**63.52**	**63.77**	63.45	63.17
NDS (*for all 10 classes*)	68.83	68.99	69.02	68.87	**69.11**	**69.35**	68.88	68.73

Table 5: **Mixing ratio between GT and PGT objects.**

GT:PGT	0:1	1:3	1:1	3:1	1:0
mAP (↑)	61.29	63.35	**63.52**	63.40	63.34
NDS (↑)	67.63	69.08	**69.11**	68.78	68.71

Table 6: **Continuously increasing pseudo-LiDAR data.**

Size	bank 1/4	bank 1/2	Full bank
mAP (↑)	63.18	63.43	**63.52**
NDS (↑)	68.72	68.94	**69.11**

Table 7: **Comparison between other methods that aim for class imbalance problems.** *Abbr.* C.V: Construction Vehicle, Ped: Pedestrian, T.C: Traffic Cone, M.C: Motorcycle, B.C: Bicycle. †: our reproduction.

Model	Aug.	Majority classes				Minority classes						mAP
		Car	Ped	Barrier	T.C	Bus	C.V	Trailer	Truck	M.C	B.C	
PointPillars [3]	CBLoss†	82.7	**74.7**	54.0	52.1	51.2	61.7	17.9	30.5	48.7	20.5	49.4
	DWA	81.0	72.3	50.2	50.1	49.0	63.4	10.7	34.3	32.9	6.9	44.6
	PGT-Aug	**83.0**	71.8	54.8	51.1	**54.9**	**69.7**	20.2	**39.5**	49.6	14.5	50.9
	PGT Aug + CBLoss	82.7	**74.7**	**56.5**	**55.9**	54.4	68.7	**20.9**	34.1	**53.5**	**20.5**	**52.2** (+1.3%)

intensity loss. Finally, we used objects generated by Plenoxel with the parameters that gave the best FID score for all 3D object detection tasks. We provide more details in the Supplementary Material.

Performance based on FID scores. We conducted experiments to assess how the quality of generated objects as FID scores affects detection performance. We generated objects in all comparison groups to have the same shape distribution. As shown in the last three rows of columns 1 to 5 of Table 4, we observe a positive correlation between the average FID scores and the detection performance.

Samples from Other Datasets. The evaluation was conducted not only through self-validation but also using other public datasets such as Lyft [17] and A2D2 [18]. Despite differences in class ontology and sensor settings from nuScenes, we can prove the effectiveness of our augmentation and confirm a correlation between the FID score and the number of samples in performance gain as shown in Table 4. See the Supplementary Material for matching classes across datasets.

Samples from Different Renderer. To demonstrate the baseline capability of the proposed pipeline, we applied the results generated by replacing the rendering model with Gaussian-splatting [16] instead of Plenoxels [14]. Due to Gaussian-splatting [16] generates higher-quality objects, so the detection performance improves accordingly, as shown in Table 4.

Mixing Ratio for Domain Alignments. Even when several methods are applied to reduce the domain gap between the original and generated pseudo objects, there is still a difference between the two sample groups. Therefore, when out-of-distribution data are only used, we can see that the detection performance might drop. To alleviate this discrepancy, we used a strategy that combines the original object bank with the pseudo object bank to reduce the gap. We experimentally demonstrate that this strategy can mitigate the domain gap, allowing pseudo objects to be used effectively for detection tasks. It is especially effective when the mixing ratio is 1:1, which can be confirmed through Table 5.

Performance based on Bank Size. We discuss the importance of the size of the object bank in the detection task. To examine the impact of varying object shapes, we adjusted the size of the generated bank by decreasing the number of object shapes per class by 1/2 and 1/4. As shown in Table 6, we found that as the types of object increased, the performance improved in the validation set.

Comparison between other approaches for class imbalance problems. As shown in Table 7, we provide the results of comparative experiments with [34, 57]. To align with the baseline of Dynamic Weight Average (DWA) [34], we conducted experiments on PointPillars [3]. We re-implemented Class-Balanced Loss (CBLoss) [57] with a beta value of 0.9999 and resampled the number of objects. While we observed that loss-balancing methods like DWA [34] and CBLoss [57] effectively improve the performance on minority classes, PGT-Aug, a data-augmentation-based method, achieved the largest performance improvement among all methods. Furthermore, our experiments demonstrate that combining these two types of methods can yield a synergistic effect.

Table 8: **PGT Performance on KITTI *val* set in terms of AP and mAP.**

Model	Class	Target classes			Other classes						mAP
		Cyclist			Car			Pedestrian			
	# of objects in *val*	290	262	56	2980	5082	3116	1139	605	434	
	Difficulty	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	
SECOND [1]	GT-Aug	62.3	55.3	49.8	**91.4**	**82.4**	**79.6**	87.1	67.9	63.8	68.5
	PGT-Aug	**63.3**	**56.2**	**50.2**	90.7	82.1	79.3	**90.3**	**72.1**	**67.7**	**70.1** (+1.6%)

Table 9: **PGT Performance on Lyft *val* set.** *Abbr.* E.V: Emergency Vehicle, O.V: Other Vehicle, M.C: Motorcycle, B.C: Bicycle. Ped: Pedestrian

Model	Class	Target classes					Other classes		mAP
		Truck	Bus	O.V	M.C	B.C	Car	Ped.	
	# of objects in *val*	2,721	1,653	4,920	187	3,347	91529	4952	
CP-Voxel [5]	GT-Aug	19.15	20.48	31.91	**4.54**	5.31	**37.14**	6.00	13.84
	PGT-Aug	**19.85**	**21.11**	**31.99**	4.39	**5.48**	37.11	**6.12**	**14.01** (+0.17%)

Performance on Other Datasets. Lastly, we applied the proposed framework to KITTI [8] and Lyft [17] with different sensor configurations, ranges, and the number of sweeps. Both datasets did not provide map information, so we modified our scene composition similar to the ground-only composition by using Patchworks [58] to estimate ground. As shown in Table 8 and Table 9, we can see that our framework demonstrates performance improvement across all datasets and is applicable in various environments.

6 Conclusion, Limitations, and Broader Impacts

In this paper, we propose PGT-Aug, a low-cost yet effective data augmentation framework for class imbalance in 3D object detection. To efficiently obtain rare class objects, we start by generating objects from miniatures or web videos at a low cost, then transform them to resemble real LiDAR data, and finally apply map-assistant data augmentation to insert them consistently into the scene information. Even though PGT-Aug achieves significant improvements on various 3D object detection benchmarks, we believe that domain discrepancies still exist in both datasets and categories. Furthermore, our approach must be extended to dynamic and deformable objects, such as animals. While the generated pseudo objects have the potential to render both images and point clouds, our current implementation is limited to point clouds. Regarding broader impact, our work applies not only to objects for autonomous driving but also to secure facilities or military equipment in the same data format. Thus, it could be used for the development of AI models in military or security applications.

References

[1] Yan Yan, Yuxing Mao, and Bo Li. SECOND: sparsely embedded convolutional detection. *Sensors*, 18(10): 3337, 2018. doi: 10.3390/S18103337. URL `https://doi.org/10.3390/s18103337`.

[2] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4490–4499. Computer Vision Foundation / IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00472. URL `http://openaccess.thecvf.com/content_cvpr_2018/html/Zhou_VoxelNet_End-to-End_Learning_CVPR_2018_paper.html`.

[3] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 12697–12705. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.01298. URL `http://openaccess.thecvf.com/content_CVPR_2019/html/Lang_PointPillars_Fast_Encoders_for_Object_Detection_From_Point_Clouds_CVPR_2019_paper.html`.

[4] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: point-voxel feature set abstraction for 3d object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10526–10535. Computer Vision Foundation / IEEE, 2020. doi: 10.1109/CVPR42600.2020.01054. URL `https://openaccess.thecvf.com/content_CVPR_2020/html/Shi_PV-RCNN_Point-Voxel_Feature_Set_Abstraction_for_3D_Object_Detection_CVPR_2020_paper.html`.

[5] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 11784–11793. Computer Vision Foundation / IEEE, 2021. doi: 10.1109/CVPR46437.2021.01161. URL `https://openaccess.thecvf.com/content/CVPR2021/html/Yin_Center-Based_3D_Object_Detection_and_Tracking_CVPR_2021_paper.html`.

[6] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Voxelnext: Fully sparse voxelnet for 3d object detection and tracking. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 21674–21683. IEEE, 2023. doi: 10.1109/CVPR52729.2023.02076. URL `https://doi.org/10.1109/CVPR52729.2023.02076`.

[7] Haiyang Wang, Chen Shi, Shaoshuai Shi, Meng Lei, Sen Wang, Di He, Bernt Schiele, and Liwei Wang. DSVT: dynamic sparse voxel transformer with rotated sets. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 13520–13529. IEEE, 2023. doi: 10.1109/CVPR52729.2023.01299. URL `https://doi.org/10.1109/CVPR52729.2023.01299`.

[8] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *Int. J. Robotics Res.*, 32(11):1231–1237, 2013. doi: 10.1177/0278364913491297. URL `https://doi.org/10.1177/0278364913491297`.

[9] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 11618–11628. Computer Vision Foundation / IEEE, 2020. doi: 10.1109/CVPR42600.2020.01164. URL `https://openaccess.thecvf.com/content_CVPR_2020/html/Caesar_nuScenes_A_Multimodal_Dataset_for_Autonomous_Driving_CVPR_2020_paper.html`.

[10] Jin Fang, Xinxin Zuo, Dingfu Zhou, Shengze Jin, Sen Wang, and Liangjun Zhang. Lidar-aug: A general rendering-based augmentation framework for 3d object detection. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 4710–4720. Computer Vision Foundation / IEEE, 2021. doi: 10.1109/CVPR46437.2021.00468. URL `https://openaccess.thecvf.com/content/CVPR2021/html/Fang_LiDAR-Aug_A_General_Rendering-Based_Augmentation_Framework_for_3D_Object_Detection_CVPR_2021_paper.html`.

[11] Hongge Chen, Zhao Chen, Gregory P. Meyer, Dennis Park, Carl Vondrick, Ashish Shrivastava, and Yuning Chai. SHIFT3D: synthesizing hard inputs for tricking 3d detectors. *CoRR*, abs/2309.05810, 2023. doi: 10.48550/ARXIV.2309.05810. URL `https://doi.org/10.48550/arXiv.2309.05810`.

[12] Jinglin Zhan, Tiejun Liu, Rengang Li, Jingwei Zhang, Zhaoxiang Zhang, and Yuntao Chen. Real-aug: Realistic scene synthesis for lidar augmentation in 3d object detection. *CoRR*, abs/2305.12853, 2023. doi: 10.48550/ARXIV.2305.12853. URL `https://doi.org/10.48550/arXiv.2305.12853`.

[13] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, 65(1):99–106, 2022. doi: 10.1145/3503250. URL `https://doi.org/10.1145/3503250`.

[14] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 5491–5500. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00542. URL `https://doi.org/10.1109/CVPR52688.2022.00542`.

[15] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. doi: 10.1145/3528223.3530127. URL `https://doi.org/10.1145/3528223.3530127`.

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139:1–139:14, 2023. doi: 10.1145/3592433. URL `https://doi.org/10.1145/3592433`.

[17] John Houston, Guido Zuidhof, Luca Bergamini, Yawei Ye, Long Chen, Ashesh Jain, Sammy Omari, Vladimir Iglovikov, and Peter Ondruska. One thousand and one hours: Self-driving motion prediction dataset. In Jens Kober, Fabio Ramos, and Claire J. Tomlin, editors, *4th Conference on Robot Learning, CoRL 2020, 16-18 November 2020, Virtual Event / Cambridge, MA, USA*, volume 155 of *Proceedings of Machine Learning Research*, pages 409–418. PMLR, 2020. URL `https://proceedings.mlr.press/v155/houston21a.html`.

[18] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. A2D2: audi autonomous driving dataset. *CoRR*, abs/2004.06320, 2020. URL `https://arxiv.org/abs/2004.06320`.

[19] Jiageng Mao, Minzhe Niu, Chenhan Jiang, Hanxue Liang, Jingheng Chen, Xiaodan Liang, Yamin Li, Chaoqiang Ye, Wei Zhang, Zhenguo Li, Jie Yu, Chunjing Xu, and Hang Xu. One million scenes for autonomous driving: ONCE dataset. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL `https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/67c6a1e7ce56d3d6fa748ab6d9af3fd7-Abstract-round1.html`.

[20] Jieru Mei, Alex Zihao Zhu, Xinchen Yan, Hang Yan, Siyuan Qiao, Liang-Chieh Chen, and Henrik Kretzschmar. Waymo open dataset: Panoramic video panoptic segmentation. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXIX*, volume 13689 of *Lecture Notes in Computer Science*, pages 53–72. Springer, 2022. doi: 10.1007/978-3-031-19818-2_4. URL `https://doi.org/10.1007/978-3-031-19818-2_4`.

[21] Kemal Oksuz, Baris Can Cam, Sinan Kalkan, and Emre Akbas. Imbalance problems in object detection: A review. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(10):3388–3415, 2021. doi: 10.1109/TPAMI.2020.2981890. URL `https://doi.org/10.1109/TPAMI.2020.2981890`.

[22] Lanxiao Li and Michael Heizmann. Applying plain transformers to real-world point clouds. *CoRR*, abs/2303.00086, 2023. doi: 10.48550/ARXIV.2303.00086. URL `https://doi.org/10.48550/arXiv.2303.00086`.

[23] Hongyu Guo and Herna L. Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *SIGKDD Explor.*, 6(1):30–39, 2004. doi: 10.1145/1007730.1007736. URL `https://doi.org/10.1145/1007730.1007736`.

[24] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.*, 16:321–357, 2002. doi: 10.1613/JAIR.953. URL `https://doi.org/10.1613/jair.953`.

[25] Xinting Hu, Yi Jiang, Kaihua Tang, Jingyuan Chen, Chunyan Miao, and Hanwang Zhang. Learning to segment the tail. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 14042–14051. Computer Vision Foundation / IEEE, 2020. doi: 10.1109/CVPR42600.2020.01406. URL `https://openaccess.thecvf.com/content_CVPR_2020/html/Hu_Learning_to_Segment_the_Tail_CVPR_2020_paper.html`.

[26] Shin Ando and Chun-Yuan Huang. Deep over-sampling framework for classifying imbalanced data. In Michelangelo Ceci, Jaakko Hollmén, Ljupco Todorovski, Celine Vens, and Saso Dzeroski, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Proceedings, Part I*, volume 10534 of *Lecture Notes in Computer Science*, pages 770–785. Springer, 2017. doi: 10.1007/978-3-319-71249-9_46. URL `https://doi.org/10.1007/978-3-319-71249-9_46`.

[27] Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018. doi: 10.1016/J.NEUNET.2018.07.011. URL `https://doi.org/10.1016/j.neunet.2018.07.011`.

[28] Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 6022–6031. IEEE, 2019. doi: 10.1109/ICCV.2019.00612. URL `https://doi.org/10.1109/ICCV.2019.00612`.

[29] Brandon Trabucco, Kyle Doherty, Max Gurinas, and Ruslan Salakhutdinov. Effective data augmentation with diffusion models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL `https://openreview.net/forum?id=ZWzUA9zeAg`.

[30] Khawar Islam, Muhammad Zaigham Zaheer, Arif Mahmood, and Karthik Nandakumar. Diffusemix: Label-preserving data augmentation with diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 27611–27620. IEEE, 2024. doi: 10.1109/CVPR52733.2024.02608. URL `https://doi.org/10.1109/CVPR52733.2024.02608`.

[31] Alexey Dosovitskiy, Germán Ros, Felipe Codevilla, Antonio M. López, and Vladlen Koltun. CARLA: an open urban driving simulator. In *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*, volume 78 of *Proceedings of Machine Learning Research*, pages 1–16. PMLR, 2017. URL `http://proceedings.mlr.press/v78/dosovitskiy17a.html`.

[32] Sivabalan Manivasagam, Shenlong Wang, Kelvin Wong, Wenyuan Zeng, Mikita Sazanovich, Shuhan Tan, Bin Yang, Wei-Chiu Ma, and Raquel Urtasun. Lidarsim: Realistic lidar simulation by leveraging the real world. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 11164–11173. Computer Vision Foundation / IEEE, 2020. doi: 10.1109/CVPR42600.2020.01118. URL `https://openaccess.thecvf.com/content_CVPR_2020/html/Manivasagam_LiDARsim_Realistic_LiDAR_Simulation_by_Leveraging_the_Real_World_CVPR_2020_paper.html`.

[33] Vlas Zyrianov, Xiyue Zhu, and Shenlong Wang. Learning to generate realistic lidar point clouds. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXIII*, volume 13683 of *Lecture Notes in Computer Science*, pages 17–35. Springer, 2022. doi: 10.1007/978-3-031-20050-2_2. URL `https://doi.org/10.1007/978-3-031-20050-2_2`.

[34] Daeun Lee and Jinkyu Kim. Resolving class imbalance for lidar-based object detector by dynamic weight average and contextual ground truth sampling. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2023, Waikoloa, HI, USA, January 2-7, 2023*, pages 682–691. IEEE, 2023. doi: 10.1109/WACV56688.2023.00075. URL `https://doi.org/10.1109/WACV56688.2023.00075`.

[35] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *CoRR*, abs/1908.09492, 2019. URL `http://arxiv.org/abs/1908.09492`.

[36] Aoran Xiao, Jiaxing Huang, Dayan Guan, Kaiwen Cui, Shijian Lu, and Ling Shao. Polarmix: A general data augmentation technique for lidar point clouds. In *NeurIPS*, 2022. URL `http://papers.nips.cc/paper_files/paper/2022/hash/475b85eb74d201bead9927807e713e95-Abstract-Conference.html`.

[37] Kazuto Nakashima and Ryo Kurazume. Lidar data synthesis with denoising diffusion probabilistic models. *CoRR*, abs/2309.09256, 2023. doi: 10.48550/ARXIV.2309.09256. URL `https://doi.org/10.48550/arXiv.2309.09256`.

[38] Jin Fang, Dingfu Zhou, Feilong Yan, Tongtong Zhao, Feihu Zhang, Yu Ma, Liang Wang, and Ruigang Yang. Augmented lidar simulator for autonomous driving. *IEEE Robotics Autom. Lett.*, 5(2):1931–1938, 2020. doi: 10.1109/LRA.2020.2969927. URL `https://doi.org/10.1109/LRA.2020.2969927`.

[39] Jeong Joon Park, Peter R. Florence, Julian Straub, Richard A. Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 165–174. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00025. URL `http://openaccess.thecvf.com/content_CVPR_2019/html/Park_DeepSDF_Learning_Continuous_Signed_Distance_Functions_for_Shape_Representation_CVPR_2019_paper.html`.

[40] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Hao Li, and Angjoo Kanazawa. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 2304–2314. IEEE, 2019. doi: 10.1109/ICCV.2019.00239. URL `https://doi.org/10.1109/ICCV.2019.00239`.

[41] Songyou Peng, Michael Niemeyer, Lars M. Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part III*, volume 12348 of *Lecture Notes in Computer Science*, pages 523–540. Springer, 2020. doi: 10.1007/978-3-030-58580-8_31. URL `https://doi.org/10.1007/978-3-030-58580-8_31`.

[42] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 5835–5844. IEEE, 2021. doi: 10.1109/ICCV48922.2021.00580. URL `https://doi.org/10.1109/ICCV48922.2021.00580`.

[43] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 5460–5469. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00539. URL `https://doi.org/10.1109/CVPR52688.2022.00539`.

[44] Yoonwoo Jeong, Seungjoo Shin, Junha Lee, Christopher B. Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Perfception: Perception using radiance fields. In *NeurIPS*, 2022. URL `http://papers.nips.cc/paper_files/paper/2022/hash/a76a757ed479a1e6a5f8134bea492f83-Abstract-Datasets_and_Benchmarks.html`.

[45] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 4104–4113. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.445. URL `https://doi.org/10.1109/CVPR.2016.445`.

[46] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *CoRR*, abs/2010.07492, 2020. URL `https://arxiv.org/abs/2010.07492`.

[47] Yangming Cheng, Liulei Li, Yuanyou Xu, Xiaodi Li, Zongxin Yang, Wenguan Wang, and Yi Yang. Segment and track anything. *CoRR*, abs/2305.06558, 2023. doi: 10.48550/ARXIV.2305.06558. URL `https://doi.org/10.48550/arXiv.2305.06558`.

[48] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *CoRR*, abs/2305.02463, 2023. doi: 10.48550/ARXIV.2305.02463. URL `https://doi.org/10.48550/arXiv.2305.02463`.

[49] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 9264–9275. IEEE, 2023. doi: 10.1109/ICCV51070.2023.00853. URL `https://doi.org/10.1109/ICCV51070.2023.00853`.

[50] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2242–2251. IEEE Computer Society, 2017. doi: 10.1109/ICCV.2017.244. URL `https://doi.org/10.1109/ICCV.2017.244`.

[51] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5099–5108, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/d8bf84be3800d12f74d8b05e9b89836f-Abstract.html`.

[52] Chen Zhao, Jiaqi Yang, Xin Xiong, Angfan Zhu, Zhiguo Cao, and Xin Li. Rotation invariant point cloud analysis: Where local geometry meets global topology. *Pattern Recognit.*, 127:108626, 2022. doi: 10.1016/J.PATCOG.2022.108626. URL `https://doi.org/10.1016/j.patcog.2022.108626`.

[53] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. In *NeurIPS*, 2022. URL `http://papers.nips.cc/paper_files/paper/2022/hash/9318763d049edf9a1f2779b2a59911d3-Abstract-Conference.html`.

[54] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. `https://github.com/open-mmlab/OpenPCDet`, 2020.

[55] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 1080–1089. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00116. URL `https://doi.org/10.1109/CVPR52688.2022.00116`.

[56] Fabian Fuchs, Daniel E. Worrall, Volker Fischer, and Max Welling. Se(3)-transformers: 3d roto-translation equivariant attention networks. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/15231a7ce4ba789d13b722cc5c955834-Abstract.html`.

[57] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge J. Belongie. Class-balanced loss based on effective number of samples. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 9268–9277. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00949. URL `http://openaccess.thecvf.com/content_CVPR_2019/html/Cui_Class-Balanced_Loss_Based_on_Effective_Number_of_Samples_CVPR_2019_paper.html`.

[58] Hyungtae Lim, Minho Oh, and Hyun Myung. Patchwork: Concentric zone-based region-wise ground segmentation with ground likelihood estimation using a 3d lidar sensor. *IEEE Robotics Autom. Lett.*, 6 (4):6458–6465, 2021. doi: 10.1109/LRA.2021.3093009. URL `https://doi.org/10.1109/LRA.2021.3093009`.

[59] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`.

[60] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2813–2821. IEEE Computer Society, 2017. doi: 10.1109/ICCV.2017.304. URL `https://doi.org/10.1109/ICCV.2017.304`.

[61] Seungjae Lee, Hyungtae Lim, and Hyun Myung. Patchwork++: Fast and robust ground segmentation solving partial under-segmentation using 3d point cloud. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2022, Kyoto, Japan, October 23-27, 2022*, pages 13276–13283. IEEE, 2022. doi: 10.1109/IROS47612.2022.9981561. URL `https://doi.org/10.1109/IROS47612.2022.9981561`.

[62] Alexander Lehner, Stefano Gasperini, Alvaro Marcos-Ramiro, Michael Schmidt, Mohammad-Ali Nikouei Mahani, Nassir Navab, Benjamin Busam, and Federico Tombari. 3d-vfield: Adversarial augmentation of point clouds for domain generalization in 3d object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 17274–17283. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01678. URL `https://doi.org/10.1109/CVPR52688.2022.01678`.

[63] Xuzhong Hu, Zaipeng Duan, Xiao Huang, Ziwen Xu, Delie Ming, and Jie Ma. Context-aware data augmentation for LIDAR 3d object detection. In *IEEE International Conference on Image Processing, ICIP 2023, Kuala Lumpur, Malaysia, October 8-11, 2023*, pages 11–15. IEEE, 2023. doi: 10.1109/ICIP49359.2023.10222773. URL `https://doi.org/10.1109/ICIP49359.2023.10222773`.

[64] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotný. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 10881–10891. IEEE, 2021. doi: 10.1109/ICCV48922.2021.01072. URL `https://doi.org/10.1109/ICCV48922.2021.01072`.

[65] Daniel Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. A tutorial on thompson sampling. *Found. Trends Mach. Learn.*, 11(1):1–96, 2018. doi: 10.1561/2200000070. URL `https://doi.org/10.1561/2200000070`.

[66] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jérôme Revaud. Dust3r: Geometric 3d vision made easy. *CoRR*, abs/2312.14132, 2023. doi: 10.48550/ARXIV.2312.14132. URL `https://doi.org/10.48550/arXiv.2312.14132`.

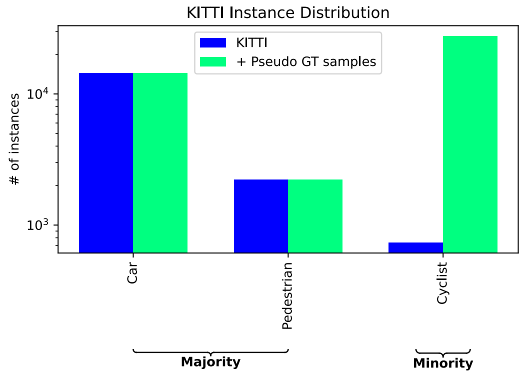
[67] Yawar Siddiqui, Antonio Alliegro, Alexey Artemov, Tatiana Tommasi, Daniele Sirigatti, Vladislav Rosov, Angela Dai, and Matthias Nießner. Meshgpt: Generating triangle meshes with decoder-only transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 19615–19625. IEEE, 2024. doi: 10.1109/CVPR52733.2024.01855. URL `https://doi.org/10.1109/CVPR52733.2024.01855`.

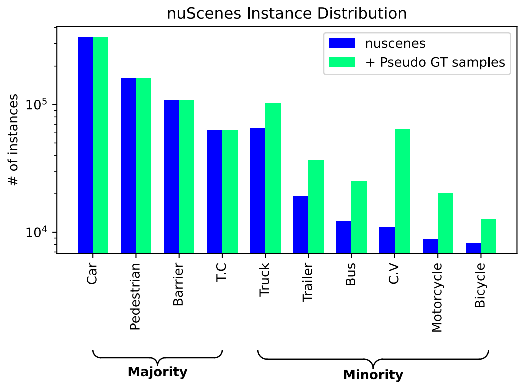
[68] Peihao Wang, Dejia Xu, Zhiwen Fan, Dilin Wang, Sreyas Mohan, Forrest N. Iandola, Rakesh Ranjan, Yilei Li, Qiang Liu, Zhangyang Wang, and Vikas Chandra. Taming mode collapse in score distillation for text-to-3d generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 9037–9047. IEEE, 2024. doi: 10.1109/CVPR52733.2024.00863. URL `https://doi.org/10.1109/CVPR52733.2024.00863`.

A Appendix

A.1 Dataset Details

Motivation. We provide a distribution of annotated instances across different classes for three benchmarks (nuScenes [9], KITTI [18], and Lyft [17]) in Fig. 7. Even in a few categories, severe class imbalance is observed in the datasets. This class imbalance can induce a significant performance gap between detecting the majority and minority classes. The original skewed distribution of ground truth data (Blue) is mitigated by adding pseudo-LiDAR samples generated by the PGT-Aug pipeline for instances of the minority class (Green).





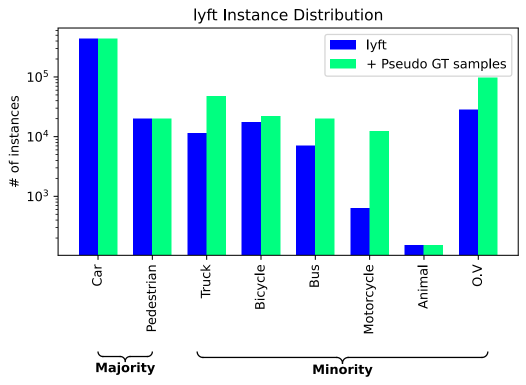


Figure 7: **Instance Distribution on various benchmarks.**

Data Collections. We first provide the number of videos collected from miniatures and web pages as shown in Table 10. We purchased various miniatures within a budget of around \$100, and crawled data using the following keywords on Google: 360° camera, surround view, turntable, sales, and second-hand. Finally, the video collection consists of 64 trucks[3], 21 motorcycles[4], and 10 buses[5] from YouTube, along with additional footage of construction vehicles from the Ritchie Bros Auction website[6]. We provide examples of our dataset collection in Fig. 8.

Table 10: **Statistics for our data collection.** (i) Videos capturing surround view of miniatures and (ii) publicly available videos of given minor-class objects. Abbr. C.V: Construction Vehicle, M.C: Motorcycle, B.C: Bicycle

Data Source	C.V	Trailer	Truck	Bus	M.C	B.C	Total
Miniatures	10	4	5	3	6	3	31
Public Videos	2	-	64	10	21	-	97

[3]https://www.youtube.com/@brucknersusedtruckcenterokc
[4]https://www.youtube.com/@MHDSuperStore
[5]https://www.youtube.com/@kagamotors
[6]https://www.rbauction.com/

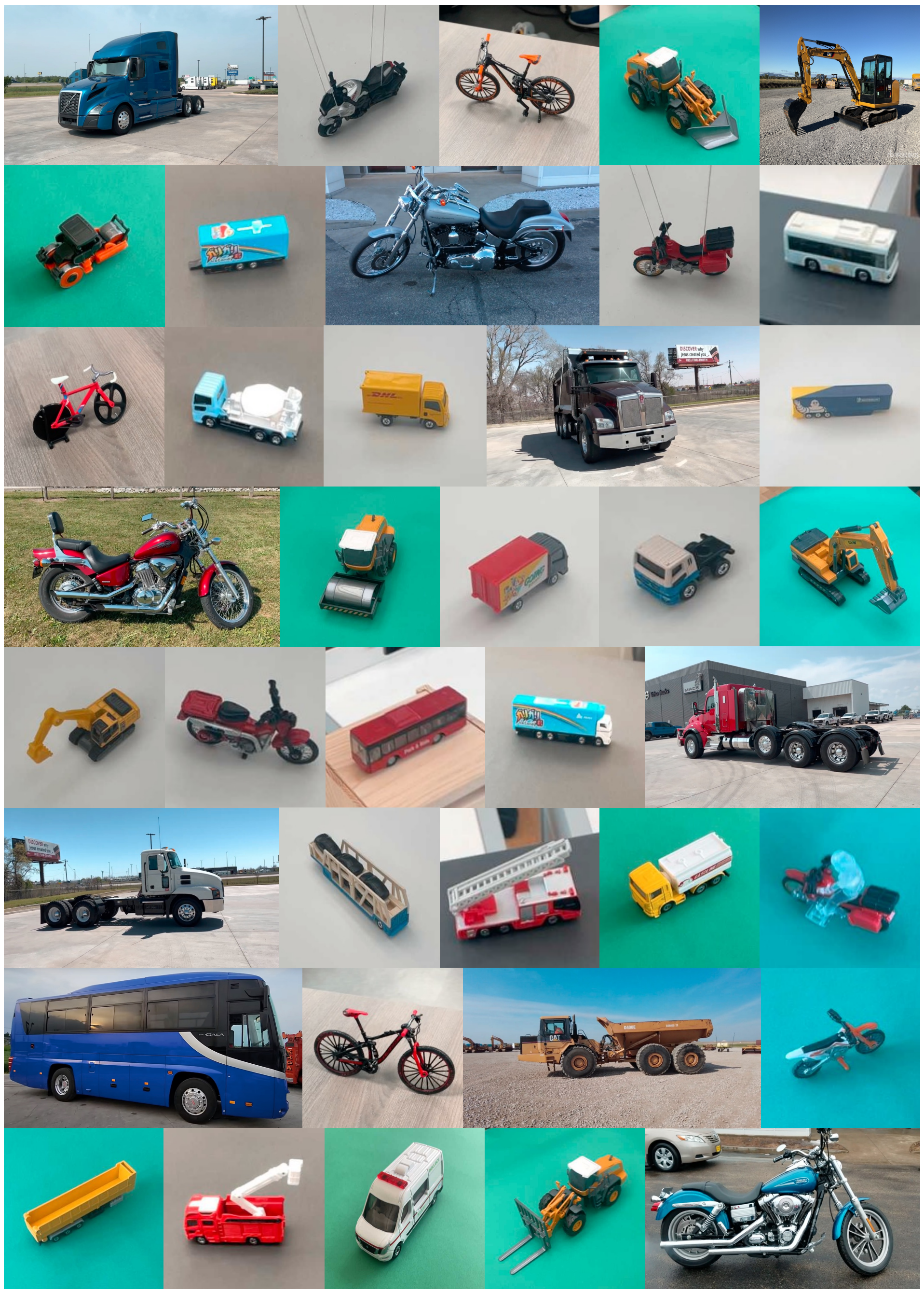


Figure 8: **Dataset Collection.** We demonstrate our collection of miniature images and crawled web videos.

Volumetric 3D Instance Collection. To aid in understanding the operation of the proposed framework, we present all results from input images to output results including foreground segmentation masks, extracted foreground objects, camera poses, and the corresponding dense RGB-colored point clouds in Fig. 9.

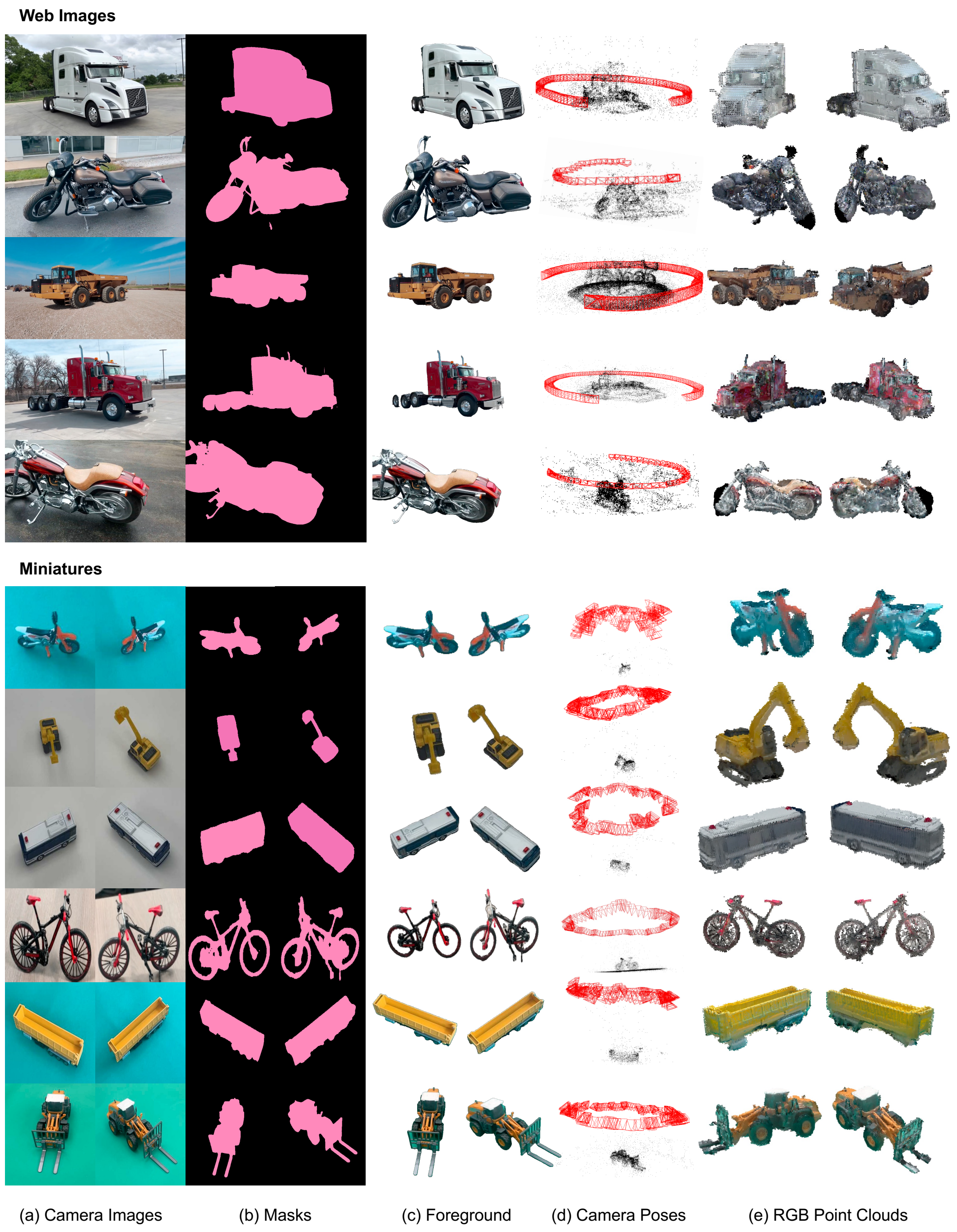


Figure 9: **Volumetric 3D rendering Results of RGB Point Clouds.**

Object-level Domain Alignment. We demonstrate the effect of our rearranged range projection as shown in Fig. 10. Not only qualitative differences, but also by reflecting the sensor configuration, we can reduce the discrepancy to the actual data domain.

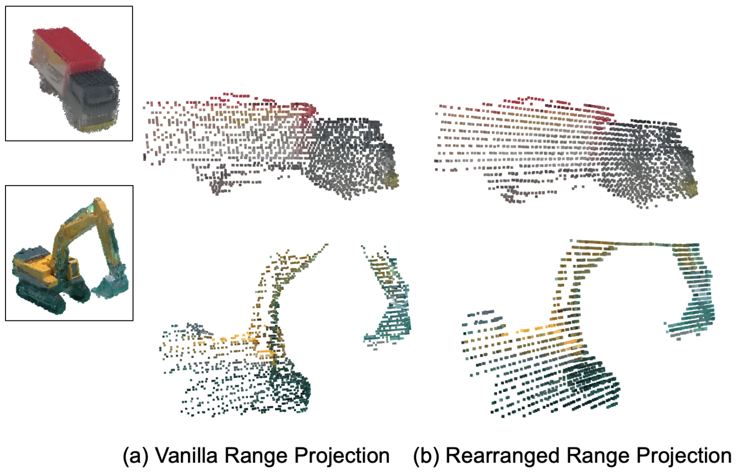


Figure 10: **Examples of Generated Point Clouds (a) with and (b) without rearranged range projection.** We use Azimuth resolution: 1080, Vertical FOV: -30° 10°, Channels: 32 for sensor configuration.

A.2 Implementation Details

View-agnostic Color Representation. We provide a pseudo-code for ad-hoc module to obtain representative colors for each voxel grid or point based on the estimated mean color values from different views. We mainly adopt PeRFception [44] framework to implement view-agnostic 2D-to-3D rendering.

Algorithm 1 Pseudocode for estimating mean RGB value of point cloud in a pretrained voxel grid or point cloud.

```
    Input $(H \times W)$ rays from $N$ views, Spherical Harmonic (SH) coefficients $sh$, step size $t_{step}$,
Spatial indices of the 3D data structure (Voxels or 3D Gaussians) $l$
    Output RGB PointCloud
 1: for each $i$-th view in $N$ do
 2:     for each $j$-th ray in $H \times W$ do
 3:         Define $RGB_{sum}, count$ to store RGB value and counts at $i$-th view.
 4:         Calculate ray origin and direction $r_d$, $r_o \leftarrow$ ray $r_{ij}$
 5:         Calculate SH basis $sh_{basis}$ with $r_d/\hat{r_d}$.
 6:         Calculate the ray $r_{ij}$'s far bound $t_{max}$, and the near bound $t_{min}$.
 7:         Set $t$ as $t_{min}$.
 8:         while $t < t_{max}$ do
 9:             Extract the position in the range $[t, t + t_{step})$ of ray $r_{ij}$.
10:             Obtain spatial index $l_{idx}$ of the position.
11:             Obtain SH coefficients $sh_c \leftarrow sh[l_{idx}]$.
12:             Calculate RGB of voxels or Gaussians by $\Sigma(sh_c \times sh_{basis})$.
13:             Add RGB value and one to $RGB_{sum}, count$ at index $l_{idx}$, respectively.
14:             Increment $t$ by $t_{step}$
15:         end while
16:     end for
17:     Append RGB sum, and counts of voxels or Gaussians in $i$-th view to arrays
18: end for
```

Front-back Classification. For the front-and-back binary classifier, we build a conventional PointNet-based model and train the model during 100 epochs with binary cross-entropy loss, using Adam optimizer [59] with beta values (0.9, 0.999). The learning rate decreases from 0.001 through the step decay scheduler with a step size of 20 and a gamma of 0.5. We used NuScenes [9] point clouds as the training set and flipped the point cloud with a probability of 0.5 to the model input. The model transfers input data into 128-dimensional feature vectors. These are then mapped into scalar probability to classify whether the vehicle is heading forward or backward.

LiDAR Intensity Simulator. Our model is based on CycleGAN [50] for unpaired translation between RGB and intensity, including class-specific PointNeXt [53] generators and discriminators. For the intensity domain, we randomly selected 300 target class samples from nuScenes dataset [9]. We then generated another 300 samples for the RGB domain by performing view-dependent point filtering on RGB dense point clouds from 2D-to-3D renderers according to the location and heading of the nuScenes sample. For batch training, we convert input samples more than 256 points into samples with 256 points using farthest point sampling. Consequently, we trained the intensity simulator for each target class over 200 epochs with batch size 16, using Adam optimizer [59] with beta values (0.5, 0.999) and a learning rate of 0.001.

For two generators $G_{\mathcal{D}_{rgb}\to\mathcal{D}_{int}}$, $G_{\mathcal{D}_{int}\to\mathcal{D}_{rgb}}$, and two discriminators $D_{\mathcal{D}_{int}}$, $D_{\mathcal{D}_{rgb}}$, we use a weighted loss consisting of group intensity loss, cyclic loss, and LSGAN loss [60],

$$\begin{aligned}\mathcal{L}\left(G_{\mathcal{D}_{rgb}\to\mathcal{D}_{int}}, G_{\mathcal{D}_{int}\to\mathcal{D}_{rgb}}, D_{\mathcal{D}_{rgb}}, D_{\mathcal{D}_{int}}\right) = \\ \mathcal{L}_{\text{LSGAN}}\left(G_{\mathcal{D}_{rgb}\to\mathcal{D}_{int}}, D_{\mathcal{D}_{int}}, \mathcal{D}_{int}, \mathcal{D}_{rgb}\right) \\ +\mathcal{L}_{\text{LSGAN}}\left(G_{\mathcal{D}_{int}\to\mathcal{D}_{rgb}}, D_{\mathcal{D}_{rgb}}, \mathcal{D}_{rgb}, \mathcal{D}_{int}\right) \\ +\mathcal{L}_{\text{cyc}}\left(G_{\mathcal{D}_{rgb}\to\mathcal{D}_{int}}, G_{\mathcal{D}_{int}\to\mathcal{D}_{rgb}}\right) \\ +\lambda\mathcal{L}_{\text{group}}\left(G_{\mathcal{D}_{rgb}\to\mathcal{D}_{int}}, \mathcal{D}_{int}, G_{\mathcal{D}_{int}\to\mathcal{D}_{rgb}}, \mathcal{D}_{rgb}\right).\end{aligned} \tag{4}$$

A.3 Experiments Details

Pseudo Object Bank Details. To verify the impact of the proposed pipeline on the 3D object detection task on nuScenes dataset [9], we created a pseudo object bank from RGB points generated by Plenoxel [14]. We used the intensity estimator trained from the point cloud with RGB features, with group intensity loss and CycleGAN loss, and performed point filtering and rearrangement according to the sensor configuration of the nuScenes dataset [9]. This pseudo object bank has the FID score 7.7 on the nuScenes GT object bank (see Table 4). For each dense RGB point cloud object, our 2D-to-3D renderer, including view-agnostic color extraction, took approximately 20 minutes. Training the intensity simulator for each class took approximately 30 minutes, and the generation of pseudo object bank samples, which includes axis alignment and projection, took 1 hour. These processes were completed using 4 AMD EPYC 7313 16-Core Processors and an NVIDIA A100.

Detection Models. We provide all the YAML configuration files for the OpenPCDet [54] models used in the paper. Note that we only modified the augmentation part of the configuration for a fair comparison. We adopt Adam optimizer with one-cycle learning rate policy and Patchwork++ [61] to obtain ground points of the input scenes. Note that SECOND [1] was performed on a voxel with a size of $[0.1, 0.1, 0.2]$. The CP-Pillar [5] was performed on a voxel with a size of $[0.1, 0.1, 8]$, while the other remaining models were performed with a voxel size of $[0.075, 0.075, 0.2]$. Training each model took approximately 36 hours using four Nvidia A100s, 256Gi memory and 4 AMD EPYC 7313 16-Core Processors on a single workstation.

A.4 Ablation Studies Details

To demonstrate the effect of the quality and quantity of the pseudo object banks, we performed various ablation studies. All experiments for the 3D object detection task are conducted based on CP-Voxel [5] operating on the input voxel size [0.075, 0.075, 0.2], while maintaining the setting of the remaining hyperparameters from OpenPCDet [54].

Quality of Pseudo Labels. We conducted experiments to assess how plausible and realistic our generated object points are compared to existing real-world datasets. To measure FID scores, we use a SE(3)-transformer [56] model trained on nuScenes dataset over 100 epochs with batch size 4, using Adam optimizer [59] with a learning rate of 0.002 and a weight decay parameter of 0.5. We first sort the objects in nuScenes in descending order based on the number of points, divide them into 32 groups, and sequentially sample objects with many points, resulting in class sets with 593 objects. Note that A2D2 [18], Lyft [17], and our pseudo GT bank use all available samples. The class mapping between different detection datasets is shown in Table 11. In feature extraction, the object is sampled with 64 points, followed by local grouping with the nearest 32 points. These features are inputted into the SE(3)-transformer [56] to extract the object-aware feature. Our implementation codes originate from `https://github.com/NVIDIA/DeepLearningExamples/tree/master/DGLPyTorch/DrugDiscovery/SE3Transformer`.

Table 11: **Categories agreement among datasets for FID evaluation and 3D object detection.**

A2D2 Dataset [18]	**Lyft Dataset [17]**	**nuScenes Dataset [9]**
Truck	Truck	Truck
Bus	Bus	Bus
MotoBiker, Motorcycle	Motorcycle	Motorcycle
Bicycle, Cyclist	Bicycle	Bicycle
Trailer	-	Trailer
Utility Vehicle	-	Construction Vehicle
Caravan Transporter	Animal Emergency Vehicle Other Vehicle	-

Samples from Other Datasets. Besides pseudo LiDAR generation for nuScenes dataset, we also generated object banks for KITTI [18] and Lyft [17] to demonstrate PGT-Aug's effectiveness on different detecction benchmarks. We adopt perception range of x from -40 to 40m and y from 0 to 70m at the interval of 5m for KITTI [18], and adopt range from -80m to 80m for Lyft [17] according to dataset specific evaluation schemes. We use 13 heading directions (from -180° to 180° at intervals of 30°). We augment pseudo LiDAR for [cyclist] class for KITTI [18], and augment [truck, motorcycle, bicycle, bus, other vehicle] classes for Lyft [17].

A.5 Efficiency and Real-time Applicability of Pre-trained Models

Memory usage and inference time If Object-level Domain Alignment is performed during detection training, it may introduce additional time overhead compared to GT-Aug or Real-Aug. In 3D object detection methods, the objects to be inserted are generated and stored prior to training, and these stored objects are loaded and inserted during the training phase. The process of loading these objects from disk into memory for scene insertion incurs time, and the memory and time complexity per batch is $O(n)$ for the number of objects n to be inserted, similar to Real-Aug. As a result, real-time performance was not a primary consideration in our approach.

We would like to emphasize two key points: (1) the pseudo LiDAR point clouds are generated and stored offline, and (2) these generated point clouds are loaded from memory and augmented during the training of 3D object detectors. Therefore, the requirement to run pre-trained models (e.g., unpaired domain transfer models) in real-time becomes less critical. In other words, we would like to emphasize two key points: (1) the pseudo LiDAR point clouds can be stored offline, and (2) the insertion of sampled objects from a ground truth instance bank is already a common practice, in typical 3D object detector training. Additionally, in Table 12, we analyze the processing time and memory usage for generating pseudo LiDAR point clouds across various object classes, such as construction vehicles, trucks, and trailers. This analysis confirms that our pipeline can efficiently generate point clouds, with a total processing time of less than 300ms.

Table 12: **Average processing time (per instance, in msec) and memory usage (MB)** C.V: Construction Vehicle, Ped: Pedestrian, T.C: Traffic Cone, M.C: Motorcycle, B.C: Bicycle.

	Minority classes					
	Bus	C.V	Trailer	Truck	M.C	B.C
Intensity Estimation	150	140	250	40	34	178
View Dependent Point Sampling	67	44	40	30	6	78
Rigid body motion	8.80	8.65	8.6	8.53	8.55	9.09
Memory	4.006	4.052	4.098	4.013	4.012	4.074

A.6 Qualitative Analysis for 3D Detection Task

We provide additional qualitative results for the model trained with the proposed PGT-Aug in Fig. 11.

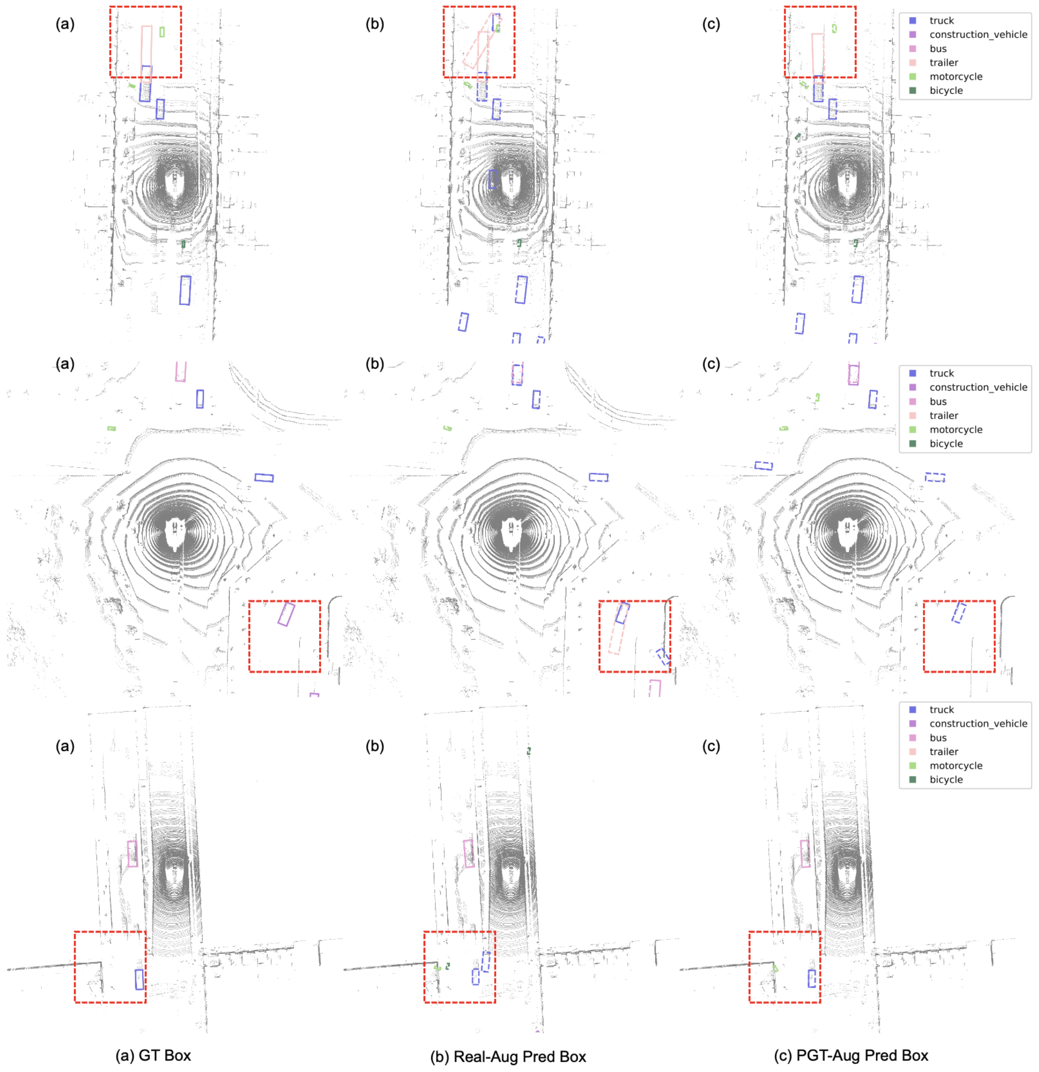


Figure 11: **Detection output comparison between Real-Aug and PGT-Aug (ours). We provide bounding boxes of (a) ground truth and detected by (b) Real-Aug [12] and (c) PGT-Aug.**

B Additional experiments.

B.1 Additional Ablation Studies.

Comparison with using only first spherical harmonic coefficient. As shown in Table 13, we compare our method with a variant model (where Plenoxel's spherical harmonic (SH) coefficients are set to 0) to see the quality of generated pseudo LiDAR point clouds in terms of FID score. The experimental results confirm that our approach can generate point clouds that are more similar to the real LiDAR data.

Table 13: **FID score evaluation between SH coefficient 0 and ours.** *Abbr.* C.V: Construction Vehicle, Ped: Pedestrian, T.C: Traffic Cone, M.C: Motorcycle, B.C: Bicycle.

	Bus	C.V.	Trailer	Truck	M.C	B.C	Average FID
SH coefficient 0	14.9	13.2	8.0	7.2	2.3	3.0	8.1
Ours	13.2	13.2	7.6	7.3	2.1	2.1	7.7

Ablation study on the impact of pre-trained models. To assess the impact of pre-trained models, we conducted additional experiments under the following conditions: (A) constant intensity without intensity generation, (B) random sampling without view-dependent point filtering, (C) no rigid motion model, and (D) and (E) instance generation based on changes in the number of images. As shown in Table 14, the removal of intensity generation in pseudo LiDAR led to a performance decline in downstream detection tasks. Moreover, as the percentage of multiview images used for 3D reconstruction decreased, the overall detection performance decreased accordingly. Lastly, we performed experiments on general alignment by replacing object-level alignment with random sampling of points from dense RGB point clouds and removing the motion models. Both experiments demonstrated suboptimal performance compared to PGT-Aug.

Table 14: **Ablation study in intensity, the number of images, data alignment on nuScenes *val* set.** *Abbr.* C.V: Construction Vehicle, Ped: Pedestrian, T.C: Traffic Cone, M.C: Motorcycle, B.C: Bicycle. †: our reproduction.

Experiments	Majority classes				Minority classes						mAP	NDS
	Car	Ped	Barrier	T.C	Bus	C.V	Trailer	Truck	M.C	B.C		
Constant Intensity (A)	85.4	85.0	68.1	71.5	72.6	23.6	38.9	60.0	68.3	54.2	62.75	68.75
Random sampling (B)	85.4	85.1	67.2	71.4	73.2	22.8	39.2	59.8	68.1	55.4	62.76	68.56
No-motion (C)	85.4	85.2	67.8	71.5	71.9	23.7	41.0	60.1	69.0	56.6	63.22	68.80
25% of the number of image (D)	85.3	85.0	67.3	71.1	73.4	23.1	40.1	59.2	70.2	56.4	63.12	68.78
50% of the number of image (E)	85.5	84.9	68.5	71.4	72.0	23.1	41.4	59.8	69.5	56.4	63.26	69.03
PGT-Aug	85.4	85.4	68.0	71.1	72.1	24.2	40.4	59.8	70.3	58.3	63.52	69.11

B.2 Effect of PGT-Aug on the majority of classes

Our primary objective is to generate and insert the minority classes instead of the majority classes to relieve the class imbalance issue. Thus, we anticipated that the performance of the minority would improve, while the performance of the majority classes would either remain stable or improve slightly due to the synergy effect from addressing the imbalance. To verify the effectiveness of the pipeline to majority classes, we compare PGT-Aug with other 3D data augmentation methods, including LiDAR-Aug [10], 3D-VField [62], and CA-Aug [63], on the KITTI [18] Car benchmark. To match their baseline, we conducted the experiments on PointPillars [3]. We reconstructed 10 cars of 3D RGB point clouds from CO3D dataset [64] and created about 16,000 pseudo LiDARs of Car class. We apply pseudo LiDARs along with GT LiDARs during augmentation, and our PGT-Aug performance largely outperforms GT-Aug, Real-Aug, and other augmentation methods in the car detection benchmark as shown in Table 15.

Instance sample details. As shown in Fig. 12, we create pseudo LiDAR point clouds for our method by randomly sampling ten cars from the CO3D dataset. The instances used

Table 15: **Performance comparison with other data augmentation approaches.** †: our reproduction

Method	AP_car 70 (40% recall)		
	Easy	Moderate	Hard
Baseline	88.08	74.85	70.55
GT-Aug [1]	87.80	78.36	75.41
LiDAR-Aug [10]	87.75	78.24	75.35
Real-Aug† [12]	88.13	78.97	76.06
3D-VField [62]	87.05	77.13	75.55
CA-Aug [63]	88.82	78.66	75.75
PGT-Aug	**90.00**	**79.45**	**76.35**

are as follows: 106_12650_23736, 106_12662_23043, 157_17286_33548, 185_19982_37678, 194_20899_41094, 216_22796_47484, 216_22827_48422, 421_58405_112551, 206_21799_45886, and 421_58407_112553.

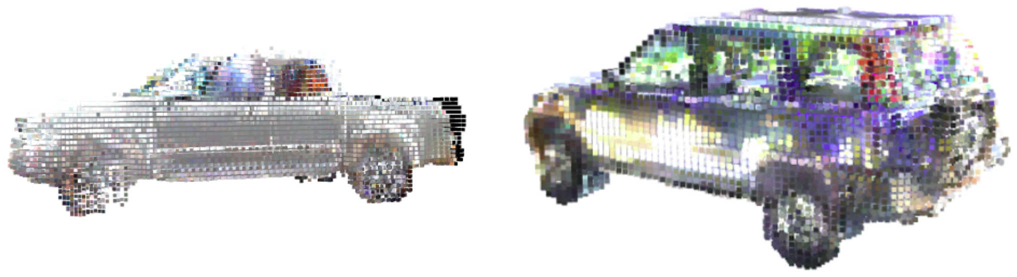


Figure 12: **Examples of pseudo GT samples generated by our pipeline for the Car class from the CO3D dataset.**

B.3 Discussion Towards Better Performance

We introduce some novel techniques for improving the effectiveness of PGT-Aug that were not applied in the main paper for a fair comparison in Table 16. *We expect that these methods will be helpful for future research.* In these experiments, the same pseudo-object bank in Section 4 was used.

Ray tracing and Distance filtering. Ray tracing can simulate a partially occluded form of objects when obstacles exist between the newly inserted object and the location of the LiDAR sensor by a range projection. Through this method, we can generate realistic scenes. For distance filtering, we observed that ground truths located outside the perception range can increase the class uncertainty due to sparsity. Therefore, we can reduce the sample uncertainty by eliminating those samples in the ground truth bank. In this experiment, we set a distance threshold of $54m$.

Bandit Algorithm for Object Placement. The advantage of our pipeline is that it can generate fully volumetric objects, allowing it to be inserted at any location. To maximize the advantage, we used Thompson sampling [65] to determine the candidate positions for pseudo LiDAR samples during training. We formulate this position decision as a bandit problem of selecting cells of a grid representing a quantized space around the ego vehicle. First, for cells with the ground truth box, if the confidence score of the prediction is greater than 0.5, the detector is considered to have performed well in the cell area and the success count of the cell increases. On the other hand, if the confidence score of the negative samples is greater than 0.5, it is considered a failure, and the failure count of the cell increases. For each cell, Thompson Sampling is performed by setting the alpha of the beta distribution as the failure count and the beta as the success count. After that, the top n cells with the highest probability are selected. This process allows us to insert objects where the detector fails during training. We set the grid size $0.075 \times 8m$ in this experiment.

Table 16: **Detection performance comparison for additional modules on nuScenes *val* set in terms of AP, mAP, and NDS**. We use CP-Voxel [5] and Transfusion-L [55] as a baseline model.

Model	CP-Voxel [5]				Transfusion-L [55]			
	A	B	C	D	E	F	G	H
w/ PGT-Aug	✓	✓	✓	✓	✓	✓	✓	✓
w/ Ray Tracing	✗	✓	✓	✓	✗	✓	✓	✓
w/ Distance Filtering	✗	✗	✓	✓	✗	✗	✓	✓
w/ Thompson Sampling	✗	✗	✗	✓	✗	✗	✗	✓
mAP (↑, *for all 10 classes*)	63.5	63.8	63.6	63.7	64.2	65.0	64.4	65.2
NDS (↑, *for all 10 classes*)	69.1	69.3	69.0	69.3	68.8	69.2	68.7	69.4

C Significance of the Proposed Method

In 3D object detection, there have been studies aimed at addressing class imbalance problems by modifying model structures (CBGS) [35] or adjusting the loss function [34]. The most widely used method is the over-sampling known as GT-Aug, which involves inserting objects from other frames into the current frame. However, this method is limited to inserting objects from a predefined pool (training set), which restricts learning intra-class diversity, and generating or collecting 3D data for various objects has been extremely challenging and expensive. Recent advancements in differentiable 3D reconstruction techniques, such as NeRF and Gaussian Splatting, have made it possible to reconstruct dense 3D points at a lower cost. By leveraging these 3D reconstruction techniques, we proposed a novel and practical pipeline that overcomes the limitations of traditional insertion methods, particularly in terms of cost and diversity. Also, our pipeline is not limited to a specific generative model and can be applied to any renderers.

C.1 Generalization by Different Renderers.

To prove the capability of the proposed framework, we show the generated RGB-colored point clouds and the simulated LiDAR point clouds of the yellow loader by different renderers: Plenoxels [14], Gaussian-Splatting [16], and DUSt3R [66] in Fig. 13. Even when different models are applied, we can observe that our proposed framework can consistently generate high-quality LiDAR objects.

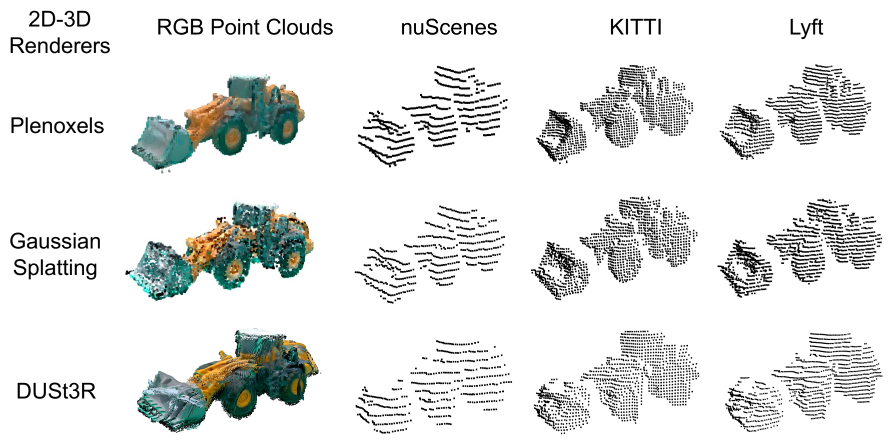


Figure 13: **Examples of RGB Point Clouds and Generated Pseudo LiDAR from Various 2D to 3D Renderers.**

C.2 Reason for using Neural Scene Representations, not Generations.

In order to place the restored objects in various positions, it is necessary to restore the entire shape of the object and create a bounding box. We chose explicit 3D representation models (3DGS Plenoxel, or various NeRF methods) that can reliably reconstruct the entire shape and perform robustly with relatively small objects, rather than Text-to-3D Generation models (Shap-E [48], MeshGPT [67], etc.), which tend to collapse modes when the target image distribution is overly concentrated in a single peak, resulting in abnormal 3D objects that are strongly related to specific views (Janus problem) [68].

C.3 Data Size.

Our dataset is created to provide pseudo-LiDAR point clouds of minority-class objects, which can be augmented into typical driving datasets to compensate for the class imbalance problem. Thus, the volume of our dataset should be smaller than that of these datasets but sufficient to compensate for the class imbalance problem, as we reported in our experiments. Moreover, our approach offers significant advantages in data generation and flexibility: (i) We provide an automated pipeline to generate pseudo-LiDAR point clouds, enabling the community to easily produce and expand datasets with various objects. (ii) Our method can generate view-dependent pseudo-LiDAR point clouds that can be flexibly placed anywhere in the scene. These features allow for rapid and efficient dataset expansion, significantly improving data efficiency and enabling researchers to quickly increase their dataset size with minimal effort.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our claims in abstract and introduction clearly elaborate on our work's motivation to solve class-imbalance in 3D LiDAR-based detection and three contributions. We also provided experimental results to certify our claims.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We addressed the limitations of our synthetic datasets and pipeline in the paper.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA] .

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide a detailed explanation of the methodology for reproducing our results and experimental settings and demonstrate the robustness of our approach through various experimental results. Through a fixed seed, we ensure deterministic behavior in all experiments. More details on the implementation are provided in the supplemental material.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We attached our dense RGB point cloud data, part of the pseudo-LiDAR object bank, and a visualization code as the supplemental material. Our code and data will be made available through the project page.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have specified our details of training and test datasets, and details of experimental settings such as batch size, hyperparameters, hardwares, codebase, size of synthetic datasets.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We answered no but created the fixed object bank for augmentation to eliminate randomness. Through a fixed seed, we ensure deterministic behavior in all experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We clearly address the computer resources for our experiments in both main paper and supplemental material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed our work according to NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have addressed societal impacts of our work in the paper.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our released datasets and detector checkpoints do not pose such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We specified the source of scraped real-world videos of long-tail classes and properly cited the use of existing models, code package, and dataset. The source code of PeRFception and OpenPCDet have Apache License 2.0 and the source code of CycleGAN has BSD License, and Segment-and-Track-Anything has AGPL-3.0 License. The corresponding license is properly inserted in each readme.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided along with the assets?

    Answer: [Yes]

    Justification: We provide detailed description of our newly created pseudo-LiDAR object bank in the main paper, alongside analysis on integration of new assets with existing detection pipeline. We also provide README instructions that include execution methods for the source code and folder structure for our object bank.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.