Instant Domain Augmentation for LiDAR Semantic Segmentation

Kwonyoung Ryu[1*] Soonmin Hwang[2*] Jaesik Park[1]
[1]POSTECH [2]Carnegie Mellon University
http://cvlab.postech.ac.kr/research/LiDomAug

Abstract

Despite the increasing popularity of LiDAR sensors, perception algorithms using 3D LiDAR data struggle with the ***sensor-bias problem****. Specifically, the performance of perception algorithms significantly drops when an unseen specification of the LiDAR sensor is applied at test time due to the domain discrepancy. This paper presents a fast and flexible LiDAR augmentation method for the semantic segmentation task called* ***LiDomAug****. It aggregates raw LiDAR scans and creates a LiDAR scan of any configurations with the consideration of dynamic distortion and occlusion, resulting in instant domain augmentation. Our on-demand augmentation module runs at* ***330 FPS****, so it can be seamlessly integrated into the data loader in the learning framework. In our experiments, learning-based approaches aided with the proposed LiDomAug are less affected by the sensor-bias issue and achieve new state-of-the-art domain adaptation performances on SemanticKITTI and nuScenes dataset without the use of the target domain data. We also present a sensor-agnostic model that faithfully works on the various LiDAR configurations.*

1. Introduction

LiDAR (Light Detection And Ranging) is a modern sensor that provides reliable range measurements of environments sampled from 3D worlds and has become crucial for intelligent systems such as robots [3,56], drones [48], or autonomous vehicles [16,32]. Therefore, developing resilient 3D perception algorithms for LiDAR data [21,27,49] is becoming more crucial.

With the growing interest in LiDAR sensors, various LiDAR sensors from multiple manufacturers have become prevalent. As a result, popular 3D datasets [5, 7, 16, 17, 22] are captured by different LiDAR configurations, which are defined by vertical/horizontal resolutions, a field of view, and a mounting pose. Due to the difference in sampling patterns from various LiDAR configurations, the *sensor-bias problem* arises in 3D perception algorithms [46, 54]. For example, as shown in Figure 1, we observe a severe performance drop in LiDAR semantic segmentation task if the LiDAR used to collect the test set differs from the LiDAR used for the training set.

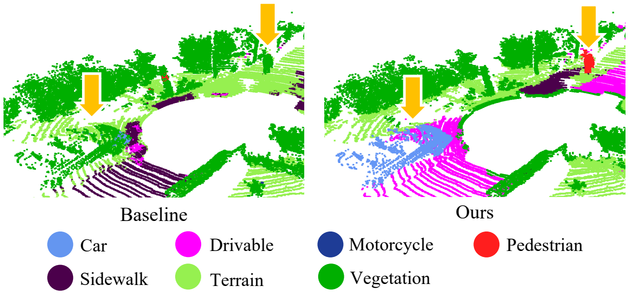

Figure 1. Data-driven LiDAR semantic segmentation methods often show an accuracy drop when they are applied to unseen LiDAR configurations. (Left) A result of a baseline approach, where inaccurate predictions are highlighted with orange arrows. (Right) A result of the baseline approach aided with the proposed LiDomAug. For both results, we use nuScenes [5] (32 ch.) for the training and use SemanticKITTI [2] (64 ch.) for the testing.

Although the sensor-bias problem is crucial, an existing solution, such as domain adaptation, is tuned for a specific LiDAR configuration, which is suboptimal to designing a sensible 3D perception method. Specifically, Supervised Domain Adaptation requires massive labeling costs to learn to adapt to the new data captured with a target sensor. Hence, such an approach is often not viable in practice. Unsupervised Domain Adaptation [23,24,54] aims to make a model adapt to a target domain without using direct annotations. However, there is an accuracy degradation, and such approaches require enough collection of target domain data. Thus, it is demanding to design a new approach that can be applied instantly to an unseen target domain without requesting any target domain data.

By focusing on the widely used cylindrical LiDARs, this paper presents a new approach to alleviate the sensor-bias problem. The proposed method, called *LiDomAug*, augments the training data based on arbitrary cylindrical LiDAR configurations, mounting pose, and motion distortions. The proposed on-demand augmentation module runs at 330 FPS, which can be regarded as an *instant domain*

[*] Equal contribution

augmentation. This flexibility, which is a key strength of our method, enables us to train a sensor-agnostic model that can be directly applied to multiple target domains.

We demonstrate our method on the task of LiDAR semantic segmentation. In particular, we tackle the domain discrepancy problem when the LiDAR sensors used for making the training and the test data are not consistent. Interestingly, learning-based approaches aided with the proposed LiDomAug outperform the state-of-the-art Unsupervised Domain Adaptation approaches [9,29,47,50,54] without access to any target domain data. Our method also beats Domain Mapping [4, 28] and Domain Augmentation approaches [18,35,52,55], showing the practicality of the proposed approach. In addition, we show a semantic segmentation model trained with LiDomAug that works faithfully on the various cylindrical LiDAR configurations.

Our contributions can be summarized as follows:

- We present an *instant LiDAR domain augmentation* method, called LiDomAug, for LiDAR semantic segmentation task. Our on-demand augmentation module runs at 330 FPS.
- Our method can augment arbitrary cylindrical LiDAR configurations, mounting pose, and entangled motions of LiDAR spin and moving platform just from the input data. We empirically validate that such flexible modules are helpful in learning sensor-agnostic LiDAR frameworks.
- Experiments show that LiDAR semantic segmentation networks trained with the proposed LiDomAug outperform the state-of-the-art Unsupervised Domain Adaptation, Domain Mapping, and LiDAR Data Augmentation approaches.

2. Related Work

2.1. LiDAR Domain Adaptation and Mapping

Domain Adaptation. A representative direction to alleviate the sensor-to-sensor domain shift issue is to adopt domain adaptation approaches [46]. Cross-modal learning [23] is exploited to enable controlled information exchange between image predictions and 3D scans. Adversarial domain adaptation methods are introduced for output space [47] or feature space alignment [9] by employing sliced Wasserstein discrepancy [29] or boundary penalty [24]. 3DGCA [50] aligns the statistics between batches from source and target data with geodesic distance. A sparse voxel completion network [54] is proposed to learn a mapping from the source domain to a canonical domain that contains complete and high-resolution point clouds. LiDAR semantic segmentation is performed on the canonical domain, and the result is projected to the target domain. ConDA [26] and CoSMix [40] also construct an intermediate domain by mixing or concatenating the source and the target domains using pseudo-labeled target data to mitigate the domain shift issue. GIPSO [41], a recent online adaptation method, requires an optimization process on target domain data using geometric propagation and temporal regularization with pseudo labels inferred from a source domain model. A common limitation of the above methods is to require additional optimization with access to target domain data, which hinders their practicality. On the other hand, our method only adds slight augmentation overhead in the training phase and circumvents the need for target domain data.

Domain Mapping. Our most relevant approach is domain mapping that directly transforms the source domain data to the target-like LiDAR scan [4, 28] and uses the transformed data for the training. However, the approach by Bešić *et al.* [4] requires access to target domain data, and the method proposed by Langer *et al.* [28] is computationally heavy due to mesh operations that recover surfaces and check occlusions. Instead, our method can produce various LiDAR scans considering multiple LiDAR configurations in 330 FPS. Our experiment shows the efficacy of our LiDAR scans on the LiDAR semantic segmentation task.

2.2. LiDAR Data Augmentation

Approaches for LiDAR data augmentation have been explored in various ways. Inspired by seminal work in image augmentation [55], augmentation methods for the LiDAR object detection task [10, 11, 13, 14, 20, 30] are proposed. However, these works are crafted for the detection task and assume bounding box labels are provided. For the 3D semantic segmentation task, CutMix [55] and Copy-Paste [18] extend the successful ideas applied for 2D image augmentation. Mix3D [35] aggregates the two 3D scenes to make objects implicitly placed into a novel out-of-context environment, which encourages the model to focus more on the local structure. Recently, PolarMix [52] introduces the scene- and object-level mix in cylindrical coordinates. PolarMix shows an impressive performance gain in domain adaptation tasks but is limited to demonstrating its synthetic-to-real adaptation capability. To the best of our knowledge, our approach is the first work on comprehensive LiDAR data augmentation to address the sensor-bias issue, and it shows superior performance compared with existing 3D data augmentation approaches.

2.3. LiDAR Semantic Segmentation

Existing approaches for 3D semantic segmentation can be categorized into three groups: 2D projection-based, point-based, and voxel-based methods. The 2D projection-based approaches [25,34,53] project 3D point clouds to 2D space and apply a neural network architecture crafted for

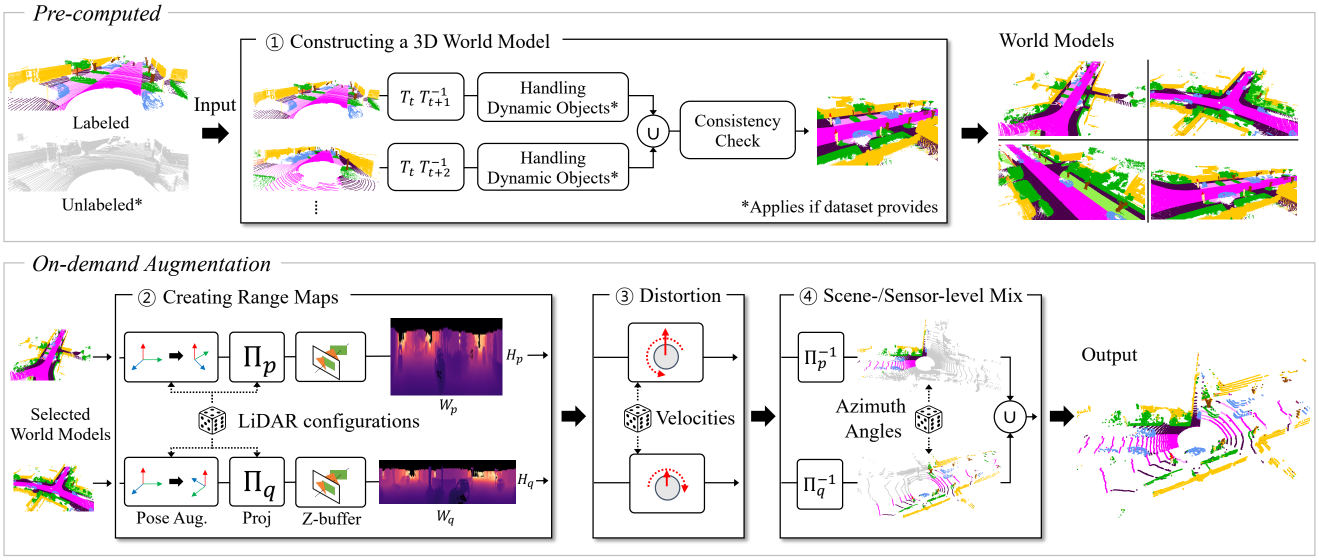

Figure 2. **Overview of the proposed LiDomAug pipeline.** ① A dense 3D world model is constructed from raw LiDAR frames with the consideration of unlabeled frames and dynamic objects (Sec. 3.1). ② Range maps of arbitrary LiDAR configuration are rendered by projecting the world model after applying random pose augmentation (Sec. 3.2). ③ The distortion induced by LiDAR spin and ego-motion is applied (Sec. 3.3). ④ Range maps are mixed using random azimuth ranges and back-projected to make an output 3D point cloud (Sec. 3.4). Note that the on-demand augmentation module that comprises ②, ③, and ④ runs at 330FPS.

image perception. Point-based methods directly work on unstructured and scattered point cloud data. Approaches in this category utilize point-wise multi-layered perceptron [8, 38], point convolution [31, 33, 45], or lattice convolution [43]. Voxel-based methods handle voxelized 3D points. Early work [39, 51] adopts dense 3D convolutions, but recent approach [12] regards voxel as a sparse tensor and presents an efficient semantic segmentation framework.

Among these approaches, we select KPConv [45] (point-based) and MinkowskiNet [12] (voxel-based) for the semantic segmentation experiments due to their efficiency and fidelity in the field. We apply the proposed LiDomAug to the selected networks to see the improvements.

3. Fast LiDAR Data Augmentation

We introduce a new augmentation method, called **LiDomAug**, that instantly creates a new LiDAR frame considering LiDAR mounting positions, various LiDAR configurations, and distortion caused by LiDAR spin and ego-motion. In this work, we craft our augmentation approach for cylindrical LiDARs. As shown in Fig. 2, our method consists of four steps: ① Constructing a world model from LiDAR frames, ② Creating a range map of arbitrary LiDAR configurations and poses, ③ Applying motion distortion to the augmented frames caused by ego-motion, and ④ Scene-level & sensor-level mix. The proposed method is flexible enough to produce a combined LiDAR frame having multiple LiDAR configurations.

3.1. Constructing a 3D World Model

A LiDAR frame is partial geometric capture of a 3D world. Therefore, we can aggregate multiple LiDAR frames of similar regions to build a rough 3D world model. In this step, we separately care for static scenes and dynamic objects by utilizing semantic label annotations on 3D points and trajectories of moving objects in the scene. Such information is available in standard LiDAR datasets [15].

Static scene. We construct a static world model by aggregating multiple LiDAR frames using ego-motion. Specifically, a set of motion-compensated LiDAR frames $\mathcal{P}_t^{world}$ is built as a world model at time t by aggregating N adjacent LiDAR frames. We determine the adjacent LiDAR frames using geometric adjacency (based on the LiDAR center coordinates) rather than temporal adjacency (based on frame indices) to cover the 3D scene better. This scheme helps to build a denser world map when the ego vehicle revisits the same place, formulated as follows:

$$\mathcal{P}_t^{world} = \bigcup_{k\in K_t} T_t \circ T_{t+k}^{-1}(\mathcal{P}_{t+k}), \tag{1}$$

where the geometrically adjacent set of frames $K_t = \arg\min_K \sum_{k\in K} ||\mathbf{R}_{t+k}^\top \mathbf{t}_{t+k} - \mathbf{R}_t^\top \mathbf{t}_t||_2$ s.t. $|K| = N$, $T_t(\mathbf{x}) = \mathbf{R}_t\mathbf{x} + \mathbf{t}_t$ is the ego-motion from the world origin at time t, and $\mathcal{P}_t$ is 3D points captured at time t.

Dynamic objects. When we aggregate 3D points on dynamic objects in the world model, we should avoid unintended flying points occurring by object-wise motion. To alleviate the issue, we leverage temporally consecutive LiDAR frames, not the geometrically adjacent frames, and we consider trajectory information of the dynamic objects over time. In short, the sparse observations of dynamic objects across multiple frames are aggregated by applying inverse motions of each dynamic object and ego-motions.

Label consistency and label propagation. After the world model construction, we examine the labeling consistency for all the aggregated 3D points in $\mathcal{P}_t^{world}$. This verification step is a safeguard to remove noisy points from various sources of errors, such as incorrect annotation and inaccurate ego-motion. To make consistent labels, we examine a set of 3D points assigned to a single voxel in the voxel grid (10cm). The majority voting determines a representative semantic label for each voxel, and we can get clean labels. Note that the majority label in a voxel can be propagated to the unlabeled points in the same voxel. This step helps assign pseudo labels to sparsely annotated datasets like nuScenes [15] that only provides dense annotations for keyframes selected at 2Hz.

3.2. Creating a Range Map

Pose augmentation. Once we have a world model, the LiDAR pose is augmented by applying a rigid transformation $T_{aug}(\mathbf{x}) = \mathbf{R}_{aug}\mathbf{x} + \mathbf{t}_{aug}$ to give variations of the LiDAR frames. In our experiments, random rotation along the z-axis, i.e., $\mathbf{R}_{aug} = \mathbf{R}_z(\theta_{yaw})$, and random translation are considered[1]. The yaw angle θ_{yaw} and the translation vector $\mathbf{t}_{aug}$ are drawn from uniform distributions.

Randomized LiDAR configurations. A LiDAR frame can be expressed as a range map, and the configuration of LiDAR is defined by the vertical field of view (f_{up}, f_{down}) and the resolution of the range map (H, W). In the case of cylindrical LiDARs, the projection $\Pi(\mathbf{x}) \rightarrow [u, v, r]^\top$ of the 3D points is calculated[2] as follows [3, 28]:

$$\Pi(\begin{bmatrix}x\\y\\z\end{bmatrix}) = \begin{bmatrix}\frac{1}{2}[1-(\arctan\frac{y}{x})/\pi]W \\ [1-(\arcsin\frac{z}{r} - f_{down})/f]H \\ ||\mathbf{x}||_2\end{bmatrix} = \begin{bmatrix}u\\v\\r\end{bmatrix}, \quad (2)$$

where $f = |f_{up}| + |f_{down}|$. With $\Pi(\cdot)$, we can project the world model $\mathbf{x} \in T_{aug}(\mathcal{P}_t^{world})$ using a given LiDAR configuration. Here, we *randomize LiDAR configuration* (H, W, f_{up}, f_{down}) to augment LiDAR frames further. With this procedure, a range map of random LiDAR configuration can be rendered, and LiDAR patterns not observed in the training data can be provided. This step is shown to be very effective in our experiment.

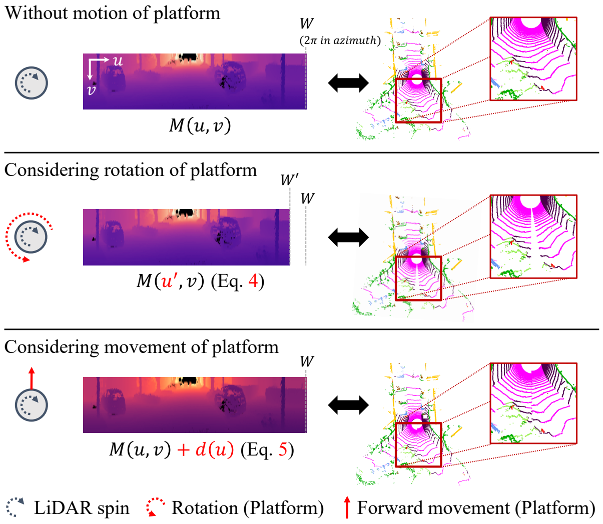
Without motion of platform
W
(2π in azimuth)
u
v
M(u, v)
Considering rotation of platform
W'
W
M(u', v) (Eq. 4)
Considering movement of platform
W
M(u, v) + d(u) (Eq. 5)
LiDAR spin
Rotation (Platform)
Forward movement (Platform)

Figure 3. Implementation of LiDAR distortion induced by entangled motion. Given movement/rotation velocities, distortion maps are generated and used to implement the motion distortion.

The world models are constructed by aggregating LiDAR frames of different viewpoints, which can result in occlusion from a desired viewpoint. To filter out these points, we employ z-buffer-based raycasting [37] that selects the nearest 3D points to the desired viewpoint. Therefore, we formulate the step for range image rendering as follows:

$$Prj(\mathcal{P}_t) = \mathcal{Z}(\Pi(T_{aug}(\mathcal{P}_t^{world}))), \quad (3)$$

where the $\mathcal{Z}$ means the z-buffer-based ray-casting.

3.3. Adding Motion Distortion

Cylindrical LiDARs have a spinning motion with a fixed rate for omnidirectional capture. Also, LiDARs are often mounted on a moving platform, such as a vehicle. The two entangled motions, i.e., movement of the vehicle and spinning motion of LiDAR, result in distortion on framed data. We observe such distortions from the real LiDAR frames (See the supplement).

More specifically, the rotation of the platform affects the effective LiDAR angular velocity, resulting in a gap or overlap between starting and ending points of a single LiDAR frame shown in the middle of Fig. 3. If the platform has a forward movement, as depicted at the bottom of Fig. 3, the starting and ending points are not aligned because each 3D point has a different travel distance. Although this distortion could significantly change the coordinates of 3D points in a LiDAR frame, this phenomenon is rarely addressed in the literature.

We formulate the distortion with LiDAR spin angular velocity ω_0, platform rotation angular velocity ω, and platform

[1] While our method is capable of incorporating arbitrary rotations, it is worth noting that most public datasets utilize *upright LiDARs*. Therefore, applying full rotations may result in an unintended severe domain gap.

[2] We set the x-axis as the vehicle's forward direction, the y-axis as the left from the vehicle, and the z-axis as the top direction from the ground.

forward movement velocity V under constant velocity assumption.

$$M(u', v) = M(\frac{(\omega_0 + \omega)}{\omega_0} \cdot u, v), \tag{4}$$

$$M(u, v) + d(u) = M(u, v) + V \cdot \frac{u}{\omega_0}, \tag{5}$$

where M is a range map projection of a LiDAR frame. The effective angular velocity is $\omega + \omega_0$ for distortion by rotation, which results in a resampling of each 3D point in the range map along u-axis, as shown in Eq. 4. The travel distance compensation due to the forward movement is given by Eq. 5. These equations lead us to an efficient implementation of the *distortion in the range map* by applying coordinate resampling and depth adjustment.

3.4. Scene-level & Sensor-level Mix

PolarMix [52] shows a scene-level mix and demonstrates strong data augmentation performance. Inspired by this work, we propose an extended augmentation module that mixes frames of *different scenes captured by different LiDARs*. As described in Fig. 2, after rendering range maps with random LiDAR configurations, we mix the range maps using random azimuth angle ranges. The mixed range map is transformed to a 3D point cloud using the inverse of the projection model Π^{-1}. Generally, the diversity of training data in a single training step is proportional to the batch size. The mixing module helps data-driven approaches by providing diverse LiDAR patterns in a single batch, reducing efforts to keep a large batch size.

4. Experiment

We conduct a series of experiments to test the generalization ability over the sensor-bias issue in the domain adaptation setting, in which a different LiDAR sensor is used at test time. We compare our method with domain adaptation and data augmentation approaches (Sec. 4.3). Next, we demonstrate the effectiveness of our method in training a sensor-unbiased model (Sec. 4.4). Last, we perform an ablation study on each technical contribution (Sec. 4.5).

4.1. Implementation details

In our experiment, as described in Sec. 3.2, the yaw angle in the rotation matrix is randomly sampled from a uniform distribution, $\theta_{yaw} \sim U(-\frac{\pi}{6}, \frac{\pi}{6})$, and each element in the translation vector is drawn from another uniform distribution, $\mathbf{t}_{aug} = [x, y, z]^T$ where $x \sim U(-1, 1)$m, $y \sim U(-0.5, 0.5)$m, and $z \sim U(-0.1, 0.1)$m. In addition, we set random LiDAR configuration parameters that are described in Sec. 3.3 as follows: $H \in \{1024, 2048\}$px., $W \in [16, 128]$px., $f_{up} \in [0, \pi/12)$, $f_{down} \in [-\pi/6, 0)$. Note that we sample W, f_{up}, and f_{down} from certain ranges to render arbitrary configurations of LiDARs. The forward movement velocity V is sampled from $U(0, 60)$ km/h, and the rotation angular velocity of the vehicle ω is sampled from $U(-\frac{\pi}{8}, \frac{\pi}{8})$. We mix two augmented LiDAR frames in our experiment.

We implement every module with GPU primitives for speed gain. As a result, our method can be seamlessly plugged into the data loader in the training pipeline due to its efficiency. For example, our method integrated into the data loader of MinkNet42 network training adds just 3ms to render a new LiDAR frame.[3]

4.2. Datasets

SemanticKITTI [2] is a large-scale dataset for LiDAR semantic segmentation task built upon the popular KITTI Vision Odometry Benchmark [16]. It consists of 22 sequences with 19 annotated classes. The dataset was collected by a Velodyne HDL-64E that has 64 vertical beams for 26.9° of vertical field of view (+2.0° to -24.9°) corresponding to 64×2048 range map. Following the standard protocol, we use sequences 00 to 10 (19k frames) for training except sequence 08 (4k frames), reserved for a validation set. Since SemanticKITTI does not provide the 3D bounding boxes, we treat the dynamic objects as a part of the static scene when constructing the world models described in Sec. 3.1.

nuScenes-lidarseg [15] is another large dataset providing 1,000 driving scenes (850 for training and validation, 150 for testing), including per-point annotation for 16 categories. However, as only the keyframes sampled at 2Hz are annotated, the label propagation scheme described in Sec. 3.1 is applied. This dataset was captured with a Velodyne HDL-32E, providing 32 vertical beams for 41.33° of vertical field of view (+10.67° to -30.67°), resulting in 32×2048 range map. We consider each motion of dynamic objects in constructing world models using the given 3D bounding box trajectory information.

Label Mapping. Since the annotated classes in Semantic KITTI [2] and nuScene [15] differ, we evaluate only ten overlapping categories in our experiments: {Car, Bicycle, Motorcycle, Truck, Other vehicles, Pedestrian, Drivable surface, Sidewalk, Terrain, and Vegetation} as suggested by [54]. We use mean Intersection-over-Union(mIoU) as our evaluation metric.

4.3. Results

We compare our method with unsupervised domain adaptation and domain mapping methods. All the methods are trained on SemanticKITTI [2] and evaluated on nuScenes [15] (K→N) or vice versa (N→K). Note that our approach *does not utilize the target dataset nor utilize the*

[3] We use a workstation equipped with AMD EPYC 7452 CPU and Nvidia GeForce RTX 3090 GPU.

Unit: mIoU (Rel.%)

Backbone (# of params)	Methods	Source → Target $K(64) \rightarrow N(32)$	$N(32) \rightarrow K(64)$
	Baseline	27.9	23.5
Complete &Label [54] (8.39M)	FeaDA [9]	27.2 (↓ 2.5)	21.4 (↓ 8.9)
	OutDA [47]	26.5 (↓ 5.0)	22.7 (↓ 3.4)
	SWD [29]	27.7 (↓ 0.7)	24.5 (↓ 4.3)
	3DGCA [50]	27.4 (↓ 1.8)	23.9 (↑ 1.7)
	C&L [54]	31.6 (↑ **13.3**)	33.7 (↑ 43.4)
	Baseline + LiDomAug	**39.2** (↑ **40.5**)	**37.9** (↑ **61.3**)

(a) Comparison with unsupervised domain adaptation approaches

Unit: mIoU (Rel.%)

Backbone (# of params)	Methods	Source → Target $K(64) \rightarrow N(32)$	$N(32) \rightarrow K(64)$
	Baseline	37.8	36.1
MinkNet42 [12] (37.8M)	CutMix [55]	37.1 (↓ 1.9)	37.6 (↑ 4.2)
	Copy-Paste [18]	38.5 (↑ 1.9)	41.1 (↑ 13.9)
	Mix3D [35]	43.1 (↑ 14.0)	44.7 (↑ 23.8)
	PolarMix [52]	45.8 (↑ 21.2)	39.1 (↑ 8.3)
	Baseline + LiDomAug	**45.9** (↑ **21.4**)	**48.3** (↑ **33.8**)

(b) Comparison with data augmentation approaches

Table 1. An experiment with domain adaptation settings. We train networks with SemanticKITTI (64 ch.) [16] and test with nuScenes (32 ch.) [15] (K→N) and vice versa (N→K). Baseline with the proposed LiDomAug is more effective than state-of-the-art Domain Adaptation and Data Augmentation approaches.

Unit: mIoU (Rel.%)

Method	Retraining	Source → Target $K(64) \rightarrow N_{0103}(32)$
CP [28]	No	28.8
MB [28]	No	30.0 (↑ 4.2)
MB+GCA [28]	Required	32.6 (↑ 13.2)
CP+GCA [28]	Required	35.9 (↑ 24.7)
BonnetalPS+AdaptLPS [4]	Required	37.5 (↑ 30.2)
EfficientLPS+AdaptLPS [4]	Required	38.5 (↑ 33.7)
MinkNet42 + LiDomAug	No	**52.4** (↑ 81.9)

Table 2. Comparison with domain mapping approaches. We follow the evaluation protocol used in [28], i.e., trained on SemanticKITTI (K) and evaluated on a subset of the nuScene (N-0103), to ensure a fair comparison. Note that our approach does not require retraining and achieves performance improvement.

target LiDAR sensor information. As stated in Sec. 3.2, our approach is trained with randomized LiDAR configurations described in Sec. 4.1 for this experiment.

Unsupervised Domain Adaptation. As shown in Table 1-(a), our method shows consistent improvement by a large margin over the state-of-the-art methods in both adaptation settings (K→N and N→K). In the N→K setting, for example, even though the model is trained on sparse data (32-ch) than the target domain (64-ch), our augmentation method provides more density-varied examples than what is available in the source domain, which helps improve the learning of sensor-agnostic representations.

Adversarial domain alignment methods, FeaDA [9], OutDA [47], SWD [29], and 3DGCA [50], show similar performance with the baseline and reveal the limitation in learning sensor-unbiased representations.[4] Compared with C&L [54] that requires additional back-and-forth mapping to the canonical domain at the test time, our augmentation method is just applied at the training time, and it does not add additional computational burdens at the test time.

Domain Mapping. Domain Mapping methods [4, 28] try to convert the source domain data to target domain data as closely as possible, so they are required to access the target domain data. Some approaches, such as GCA [28] and AdaptLPS [4], as shown in Table 2, even require retraining networks. On the other hand, as discussed in Sec. 2, our method is an effective instant domain augmentation approach, which provides diverse LiDAR patterns beyond the target domain patterns during training, so it is a good alternative to the domain mapping approaches. Table 2 shows the comparison result. We follow the same evaluation protocol used in [28] for a fair comparison, and our model shows superior performance over the state-of-the-art (BPS+AdaptLPS) by a large margin (38.5 vs. 52.4 mIOU), without re-training process required by the other approaches.

Data Augmentation. Although the 3D augmentation methods [18,35,52,55] have shown their effectiveness in learning a good representation on a *single domain*, it is rarely studied to show the effectiveness of domain adaptation settings particularly caused by sensor discrepancy. We experiment to see whether the existing LiDAR augmentation methods and our approach are good at domain adaptation.

As shown in Table 1-(b), interestingly, the 3D augmentation methods [18, 35, 52, 55] are helpful in domain adaptation settings, even though they are not designed for adapting to an unseen domain. In particular, Mix3D [35] shows impressive improvements (37.8 → 43.1 and 36.1 → 44.7) by simply aggregating two 3D scenes. However, we speculate that the point cloud aggregation by Mix3D can induce unusual local structures (e.g., two cars overlapped perpendicularly), which may result in a suboptimal model.

Furthermore, PolarMix [52], works well in the K→N setting (37.8 → 45.8). Our conjecture for the success of PolarMix in the K→N setting is that it can provide patterns of sparse 3D points similar to those found in nuScenes (32-ch) when faraway portions of a KITTI frame (64-ch) are selected for mixing. However, if the source domain does not

[4] As reported in [54], previous domain adaptation methods, namely FeaDA, OutDA, SWD, and 3DGCA, are ineffective in handling 3D LiDAR data. For more details, please refer to Sec. 4.2 in [54].

Unit: mIoU (Rel.%)

Backbone	Training data	Testing data					Avg. rank
		V64	V32	V16	O64	O128	
KPConv [45]	V64	54.70	0.02 (↓ 99.95)	0.01 (↓ 99.98)	0.01 (↓ 99.98)	0.01 (↓ 99.98)	6.2
MinkNet42 [12]	V64	**62.80**	24.32 (↓ 43.65)	13.77 (↓ 59.64)	25.80 (↓ 36.76)	24.05 (↓ 46.14)	4.8
	V32	45.59 (↓ 27.40)	43.16	25.35 (↓ 25.70)	29.05 (↓ 28.80)	29.28 (↓ 34.42)	4.0
	V16	33.70 (↓ 46.33)	29.66 (↓ 31.27)	**34.12**	34.07 (↓ 16.50)	31.48 (↓ 29.50)	4.0
	O64	43.01 (↓ 31.51)	39.13 (↓ 9.34)	27.96 (↓ 18.05)	40.80	43.08 (↓ 3.516)	3.2
	O128	42.25 (↓ 32.72)	27.41 (↓ 36.49)	10.54 (↓ 69.11)	37.81 (↓ 7.33)	44.65	4.4
	LiDomAug (Rand)	61.51 (↓ 2.05)	**44.73** (↑ **3.64**)	33.38 (↓ 2.17)	**46.54** (↑ **14.07**)	**48.34** (↑ **8.26**)	**1.4**

Table 3. Experiment on the sensor-bias issue of LiDAR semantic segmentation models. We tested KPConv [45] and MinkowskiNet [12] models on SemanticKITTI [2] by generating multiple LiDAR configurations using the proposed LiDomAug. 'LiDomAug (Rand)' denotes that the model is trained with our final method, i.e., with pose-augmentation, randomized LiDAR config., random distortion, and scene-level & sensor-level mixing. For each test configuration, **the best** and the second-best performances are highlighted, the reference cases (the LiDAR scans come from the same LiDAR configurations are used at the training and the testing time) are colored in gray.

provide enough diversity as in the opposite N→K scenario, PolarMix shows reduced improvement (36.1 → 39.1). This result shows that learning a rich sensor-agnostic representation is challenging. Our method aims to reduce the domain gap induced by sensor discrepancy by explicitly rendering various LiDAR patterns. As a result, our method achieves superior performances in both K→N and N→K settings by a large margin (37.8 → 45.9 and 36.1 → 48.3).

4.4. Towards Sensor-agnostic Model

Our method encourages models to learn a sensor-agnostic representation, and no data from the target domain is required during training. In this experiment, we discuss the effectiveness of our approach in training a model unbiased to any LiDAR configuration. This experiment is challenging to conduct because there is no real-world dataset captured by different kinds of LiDARs at once[5]. Therefore, it is not straightforward to configure a dataset of the same scene captured with different LiDARs.

To proceed with this experiment, we use the proposed LiDomAug to create LiDAR frames of various LiDAR configurations from the SemanticKITTI dataset. These frames of specific LiDAR configurations were then mix-and-matched for training and testing datasets. Specifically, we create the frames of 16-, 32-, and 64-ch Velodyne LiDARs [19] (denoted by V16, V32, and V64) and the frames of 64- and 128-ch Ouster LiDARs [36] (denoted by O64 and O128) based on the manufacturer-provided LiDAR specification[6]. We tested KPConv [45] and MinkowskiNet [12], which are representative methods in point- and voxel-based approaches, respectively.

In Table 3, we present evaluation results on various LiDAR patterns (columns) of a model trained on a specific LiDAR pattern (row). As shown in rows 1-2, the models show severe performance drops if the LiDAR got changed at test time. For instance, a MinkNet42 [12] model trained on V64 LiDAR pattern in the second row achieves 62.80 mIoU if tested on the same LiDAR. However, the model shows significant performance drops if evaluated on different LiDAR patterns (24.32 mIoU on V32, 13.77 mIoU on V16, etc.). Especially, KPConv [45] fails under this sensor-discrepancy scenario while MinkowskiNet [12] model is less affected. We speculate that the U-Net style of architectural design makes it resilient to variations of the geometric patterns of 3D points. Therefore, we choose MinkowskiNet [12] as the backbone model for the rest of the experiment.

We also trained MinkowskiNet [12] models on the other LiDAR configurations, such as V32, V16, O64, and O128, shown in rows 3-6 of Table 3. As expected in the sensor-discrepancy evaluation scenarios, the best performances are achieved when the same LiDAR is applied at test time (colored in gray). Otherwise, the performances fluctuate a lot. This result indicates that a data-driven model tends to be biased towards a specific LiDAR configuration of the training data, which could be a hurdle in deploying them to real-world applications.

As a remedy for the sensor-bias issue, we propose to train models with the proposed LiDomAug using *randomized LiDAR configurations*, shown in row 7 of Table 3. Our model gets to learn sensor-agnostic representations since LiDomAug provides various LiDAR patterns with realistic distortions. Given the *no free lunch theorem* [42], LiDomAug (Rand) does not beat all the test settings, especially when the LiDAR configuration used for the training and testing is the same. Our model, however, achieves the highest generalization ability across the diverse LiDAR configurations we tested, measured by average rank in the last column of Table 3. This result shows that the proposed method helps alleviate the sensor bias.

[5] Carballo *et al.* [6] proposed a multi-lidar dataset, but the dataset is not released to the public domain.

[6] More detailed configurations are described in the supplement.

Unit: mIoU (Rel.%)

Training data (Sec 3.2)	Pose-Aug (Sec 3.2)	Distortion (Sec 3.3)	S&S Mix (Sec 3.4)	$K(64) \rightarrow N(32)$
K(64)				36.57±0.56
K(32)				37.05±1.15 (↑ 1.31)
Random				40.05±1.03 (↑ 9.51)
Random	✓			42.70±0.91 (↑ 16.8)
Random	✓	✓		43.04±0.56 (↑ 17.7)
Random	✓	✓	✓	**44.98±1.42 (↑ 23.0)**

Table 4. Ablation study on the proposed augmentation modules. The results are averaged over three runs.

Unit: mIoU (Rel.%)

Backbone (# of params)	Methods	Source → Target	
		$K(64) \rightarrow N(32)$	$N(32) \rightarrow K(64)$
SPVCNN [44] (37.9M)	Baseline	43.4	41.9
	Baseline + LiDomAug	**51.7** (↑ **19.1**)	**51.2** (↑ **22.2**)

Table 5. Training with SemanticKITTI [16] and testing with nuScenes-lidarseg [15] (K→N). The vice versa is denoted as (N→K). We utilize state-of-the-art NAS-based 3D neural network architecture (SPVCNN [44]) for this experiment.

4.5. Ablation Study

We perform an ablation study on the impact of each proposed contribution. In this experiment, we train MinkNet42 models [12] on SemanticKITTI (64 ch.) [2] and test them on nuScenes (32 ch.) [15], i.e., $K(64) \rightarrow N(32)$ scenario.

Training with randomized LiDARs. We compare models trained on three types of LiDAR patterns: (1) the original SemanticKITTI dataset, denoted as K(64). (2) 32 ch. LiDAR, created by LiDomAug using SemanticKITTI dataset and target nuScene LiDAR specification, denoted as K(32). (3) LiDAR scans created by LiDomAug using randomized configurations, denoted as Random. As shown in rows 1-3 of Table 4, K(32) shows improvement over K(64) because K(32) resembles the data in the target domain, denoted as N(32). Random data provides examples with abundant patterns to help learn a better representation, achieving additional performance gain (36.57 → 40.05).

Pose augmentation is another effective source of providing a diversity of the scan patterns. As shown in Table 4, adding pose augmentation to the Random LiDAR inputs leads to extra improvement (40.05 → 42.70).

Distortion induced by entangled motion. In the real world, LiDAR data have distortions due to the entangled motion of the vehicle and LiDAR. Our distortion module implements various degrees of distortion from randomly selected forward and angular velocities of the vehicle. As this module enhances the realism of LiDAR frames, we achieve another enhancement (42.70 → 43.04), shown in row 5 of Table 4.

Scene-level & Sensor-level mix module provides extra diversity to a single LiDAR frame by swapping scenes captured with different LiDAR configurations. Eventually, the final model shown in row 6 of Table 4 learns a better representation from the rich LiDAR frames (43.04 → 44.98).

NAS-based backbone. We additionally validate that our approach can be applied to an advanced 3D neural network, SPVCNN [44] that was found by extensive neural architecture search (NAS). We use the same experimental setting used in Sec. 4.3, and we utilize the pre-trained network provided by the authors. After fine-tuning the network with the proposed LiDomAug, we observe the prediction accuracy on the unseen target domain significantly enhanced in both $K(64) \rightarrow N(32)$ and $N(32) \rightarrow K(64)$ scenarios.

5. Conclusion

This paper proposes a new LiDAR augmentation method to remedy the sensor-bias issue in LiDAR semantic segmentation models. Our method efficiently transforms real-world LiDAR data to another LiDAR domain having the desired configuration. Due to its efficiency, our method can be deployed as an online data augmentation module in the learning frameworks, which leads us to call our method instant domain augmentation. Our method does not require access to any target data, so it encourages models to learn a sensor-agnostic representation by providing random LiDAR configurations of data. Extensive experiments show that training with our method significantly improves the LiDAR semantic segmentation performance in the unseen datasets collected by a different LiDAR.

Limitation and future work. Our method requires accurate 6-DoF ego-motions to construct the world models, but it could be estimated by off-the-shelf LiDAR SLAM method [1]. Our method is crafted for a cylindrical LiDAR, the most common type utilized in most of the existing public datasets. However, our trivial extension to a more complex setting, i.e., two LiDAR settings consisting of a solid-state LiDAR and a cylindrical LiDAR, shows a promising result (see Section E in the supplement). Since our method can be used for a generic LiDAR domain augmentation, our future work is to apply our method to other 3D perception tasks, such as object detection or instance semantic segmentation.

Acknowledgement. This work was supported by IITP grant (No.2019-0-01906, POSTECH Artificial Intelligence Graduate School Program) and NRF grant (No.2023R1A1C200781211) funded by the Korean government (MSIT). Hyundai Motor Group provided generous support for this research.

References

[1] Chunge Bai, Tao Xiao, Yajie Chen, Haoqian Wang, Fang Zhang, and Xiang Gao. Faster-lio: Lightweight tightly coupled lidar-inertial odometry using parallel sparse incremental voxels. *IEEE Robotics and Automation Letters*, 2022. 8

[2] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *International Conference on Computer Vision*, 2019. 1, 5, 7, 8

[3] Jens Behley and Cyrill Stachniss. Efficient surfel-based slam using 3d laser range data in urban environments. In *Robotics: Science and Systems Conference*, 2018. 1, 4

[4] Borna Bešić, Nikhil Gosala, Daniele Cattaneo, and Abhinav Valada. Unsupervised domain adaptation for lidar panoptic segmentation. *Robotics and Automation Letters*, 2022. 2, 6

[5] 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. *arXiv preprint*, 2019. 1

[6] Alexander Carballo, Jacob Lambert, Abraham Monrroy, David Wong, Patiphon Narksri, Yuki Kitsukawa, Eijiro Takeuchi, Shinpei Kato, and Kazuya Takeda. Libre: The multiple 3d lidar dataset. In *Intelligent Vehicles Symposium*, 2020. 7

[7] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Conference on Computer Vision and Pattern Recognition*, 2019. 1

[8] R. Qi Charles, Hao Su, Mo Kaichun, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2017. 3

[9] Yi-Hsin Chen, Wei-Yu Chen, Yu-Ting Chen, Bo-Cheng Tsai, Yu-Chiang Frank Wang, and Min Sun. No more discrimination: Cross city adaptation of road scene segmenters. In *International Conference on Computer Vision*, Oct 2017. 2, 6

[10] Shuyang Cheng, Zhaoqi Leng, Ekin Dogus Cubuk, Barret Zoph, Chunyan Bai, Jiquan Ngiam, Yang Song, Benjamin Caine, Vijay Vasudevan, Congcong Li, Quoc V. Le, Jonathon Shlens, and Dragomir Anguelov. Improving 3d object detection through progressive population based augmentation. In *European Conference on Computer Vision (ECCV)*, 2020. 2

[11] Jaeseok Choi, Yeji Song, and Nojun Kwak. Part-aware data augmentation for 3d object detection in point cloud*. *International Conference on Intelligent Robots and Systems*, pages 3391–3397, 2021. 2

[12] Christopher Bongsoo Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. *Conference on Computer Vision and Pattern Recognition*, 2019. 3, 6, 7, 8

[13] Jin Fang, Dingfu Zhou, F. L. Yan, Tongtong Zhao, Feihu Zhang, Yu Ma, Liang Wang, and Ruigang Yang. Augmented lidar simulator for autonomous driving. *Robotics and Automation Letters*, 2020. 2

[14] 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 *Conference on Computer Vision and Pattern Recognition*, June 2021. 2

[15] Whye Kit Fong, Rohit Mohan, Juana Valeria Hurtado, Lubing Zhou, Holger Caesar, Oscar Beijbom, and Abhinav Valada. Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking. *arXiv preprint*, 2021. 3, 4, 5, 6, 8

[16] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Conference on Computer Vision and Pattern Recognition*, 2012. 1, 5, 6, 8

[17] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, et al. A2d2: Audi autonomous driving dataset. *arXiv preprint*, 2020. 1

[18] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D. Cubuk, Quoc V. Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2918–2928, June 2021. 2, 6

[19] David S. Hall. High definition lidar system, U.S. Patent, US7969558B2. 7

[20] Jordan S. K. Hu and Steven L. Waslander. Pattern-aware data augmentation for lidar 3d object detection. *Computing Research Repository*, 2021. 2

[21] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Agathoniki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. *Conference on Computer Vision and Pattern Recognition*, 2020. 1

[22] Xinyu Huang, Xinjing Cheng, Qichuan Geng, Binbin Cao, Dingfu Zhou, Peng Wang, Yuanqing Lin, and Ruigang Yang. The apolloscape dataset for autonomous driving. In *Conference on Computer Vision and Pattern Recognition Workshops*, 2018. 1

[23] Maximilian Jaritz, Tuan-Hung Vu, Raoul de Charette, Emilie Wirbel, and Patrick Pérez. xmuda: Cross-modal unsupervised domain adaptation for 3d semantic segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2020. 1, 2

[24] Peng Jiang and Srikanth Saripalli. Lidarnet: A boundary-aware domain adaptation model for point cloud semantic segmentation. In *International Conference on Robotics and Automation*, 2021. 1, 2

[25] Deyvid Kochanov, Fatemeh Karimi Nejadasl, and Olaf Booij. Kprnet: Improving projection-based lidar semantic segmentation. *European Conference on Computer Vision Workshops*, 2020. 2

[26] Lingdong Kong, Niamul Quader, Venice Erin Liong, and Hanwang Zhang. Conda: Unsupervised domain adaptation for lidar segmentation via regularized domain concatenation. 2021. 2

[27] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. *Conference on Computer Vision and Pattern Recognition*, 2019. 1

[28] F. Langer, A. Milioto, A. Haag, J. Behley, and C. Stachniss. Domain Transfer for Semantic Segmentation of LiDAR Data using Deep Neural Networks. In *International Conference on Intelligent Robots and Systems*, 2020. 2, 4, 6

[29] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, June 2019. 2, 6

[30] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17295–17304, June 2022. 2

[31] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*, 2018. 3

[32] You Li and Javier Ibanez-Guzman. Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems. *IEEE Signal Processing Magazine*, 37(4):50–61, 2020. 1

[33] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *Advances in Neural Information Processing Systems*, 2019. 3

[34] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Rangenet++: Fast and accurate lidar semantic segmentation. In *International Conference on Intelligent Robots and Systems*, 2019. 2

[35] Alexey Nekrasov, Jonas Schult, Or Litany, B. Leibe, and Francis Engelmann. Mix3d: Out-of-context data augmentation for 3d scenes. *2021 International Conference on 3D Vision (3DV)*, pages 116–125, 2021. 2, 6

[36] Angus Pacala, Mark Frichtl, Marvin Shu, and Eric Younge. Rotating compact light ranging system, U.S. Patent, US10481269B2. 7

[37] Matt Pharr and Greg Humphreys. *Physically Based Rendering: From Theory To Implementation*, volume 2. 08 2004. 4

[38] C. Qi, L. Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, 2017. 3

[39] Charles R. Qi, Hao Su, Matthias Niessner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Conference on Computer Vision and Pattern Recognition*, June 2016. 3

[40] Cristiano Saltori, Fabio Galasso, Giuseppe Fiameni, Niculae Sebe, Elisa Ricci, and Fabio Poiesi. Cosmix: Compositional semantic mix for domain adaptation in 3d lidar segmentation. In *European Conference on Computer Vision (ECCV)*, 2022. 2

[41] Cristiano Saltori, Evgeny Krivosheev, Stéphane Lathuilière, Niculae Sebe, Fabio Galasso, Giuseppe Fiameni, Elisa Ricci, and Fabio Poiesi. Gipso: Geometrically informed propagation for online adaptation in 3d lidar segmentation. In *European Conference on Computer Vision (ECCV)*, 2022. 2

[42] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014. 7

[43] Hang Su, V. Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. *Conference on Computer Vision and Pattern Recognition*, 2018. 3

[44] Haotian* Tang, Zhijian* Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *European Conference on Computer Vision (ECCV)*, 2020. 8

[45] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. *Conference on Computer Vision and Pattern Recognition*, 2019. 3, 7

[46] Larissa T Triess, Mariella Dreissig, Christoph B Rist, and J Marius Zöllner. A survey on deep domain adaptation for lidar perception. In *Intelligent Vehicles Symposium Workshops*, 2021. 1, 2

[47] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker. Learning to adapt structured output space for semantic segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2018. 2, 6

[48] Luke Wallace, Arko Lucieer, Christopher Stephen Watson, and Darren Turner. Development of a uav-lidar system with application to forest inventory. *Remote. Sens.*, 2012. 1

[49] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark E. Campbell, and Kilian Q. Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. *Conference on Computer Vision and Pattern Recognition*, 2019. 1

[50] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *International Conference on Robotics and Automation*, 2019. 2, 6

[51] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Conference on Computer Vision and Pattern Recognition*, 2015. 3

[52] Aoran Xiao, Jiaxing Huang, Dayan Guan, Kaiwen Cui, Shijian Lu, and Ling Shao. Polarmix: A general data augmentation technique for lidar point clouds. *ArXiv*, abs/2208.00223, 2022. 2, 5, 6

[53] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. In *European Conference on Computer Vision*, 2020. 2

[54] Li Yi, Boqing Gong, and Thomas Funkhouser. Complete & label: A domain adaptation approach to semantic segmentation of lidar point clouds. In *Conference on Computer Vision and Pattern Recognition*, June 2021. 1, 2, 5, 6

[55] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Young Joon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031, 2019. 2, 6

[56] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems*, 2014. 1