# LiDAR-CS Dataset: LiDAR Point Cloud Dataset with Cross-Sensors for 3D Object Detection

Jin Fang[1], Dingfu Zhou[1], Jingjing Zhao[1], Chenming Wu[1], Chulin Tang[2], and Liangjun Zhang[1]

*Abstract*— Over the past few years, there has been remarkable progress in research on 3D point clouds and their use in autonomous driving scenarios has become widespread. However, deep learning methods heavily rely on annotated data and often face domain generalization issues. Unlike 2D images whose domains usually pertain to the texture information present in them, the features derived from a 3D point cloud are affected by the distribution of the points. The lack of a 3D domain adaptation benchmark leads to the common practice of training a model on one benchmark (e.g. Waymo) and then assessing it on another dataset (e.g. KITTI). This setting results in two distinct domain gaps: scenarios and sensors, making it difficult to analyze and evaluate the method accurately. To tackle this problem, this paper presents LiDAR Dataset with Cross-Sensors (LiDAR-CS Dataset), which contains large-scale annotated LiDAR point cloud under six groups of different sensors but with the same corresponding scenarios, captured from hybrid realistic LiDAR simulator. To our knowledge, LiDAR-CS Dataset is the first dataset that addresses the sensor-related gaps in the domain of 3D object detection in real traffic. Furthermore, we evaluate and analyze the performance using various baseline detectors and demonstrated its potential applications. Project page: *https://opendriving.github.io/lidar-cs*.

## I. INTRODUCTION

In recent years, deep learning techniques have advanced quickly, and the hardware that supports them has also improved. As a result, deep-learning-related applications have become more prominent, and one of the most noteworthy is Autonomous Driving (AD). To achieve 3D perception tasks, Light Detection and Ranging (LiDAR) sensors have become the leading choice due to their reliable range measurement. To well promote the technique development, many AD-related 3D perception datasets have been released such as the KITTI [1], nuScene [2] and Waymo [3].

Despite the notable accomplishments, pressing concerns still confront the community. Evaluating models across different datasets can be a complex task, which requires careful consideration of various factors. Usually, this kind of evaluation is necessary to verify the model's generalization ability, however, this process is very complex and time-consuming due to several reasons: **(1) different sensors.** The LiDAR points are collected from different sensors, e.g., KITTI (Velodyne 64 beams LiDAR), nuScene (32 beams LiDAR); **(2) different labeling rules.** The labeling rule is different, in the KITTI dataset, only the objects in the front views have been annotated while the nuScene and Waymo are labeled in $360°$ view; **(3) different categories.** KITTI and Waymo share the same labeled categories while nuScene
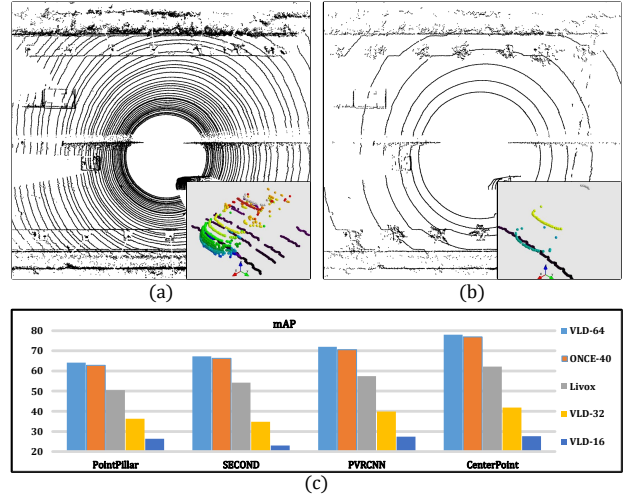


Fig. 1: (a) and (b) are LiDAR point cloud examples collected from different types of sensors which are from 64-beams and 16-beams LiDAR sensors respectively. The vehicle has been cropped and zoomed in for detailed visualization purposes. Sub-figure (c) gives a cross-sensors evaluation of experimental results where four baseline detectors are trained on VLD 64 LiDAR data and evaluated on five different sensors in the same scenario. The results show that the domain gaps are obvious across different sensors.

defines more specific categories. **(4) different evaluation metrics.** Last but not least, the evaluation metric for each dataset is also quite different which confuses the comparison over different datasets.

Secondly, there have been many different LiDAR sensor designs and functions due to the rapid development of autonomous driving technology recently, such as Velodyne [4], Pandora [5] and Livox [6], For each manufacturer, there are lots of LiDAR sensor types. However, the existing LiDAR point cloud benchmarks are trying to enlarge the scenario diversity while the diversity of sensors has been ignored. However, none of them provide more diverse data with different LiDAR sensors. As shown in Fig. 1, the detectors' performances drop dramatically when the training and the testing are based on different sensors. Therefore, a dataset that can enlarge the sensor's diversity will be very helpful for the community.

Thirdly, domain adaptation is a crucial challenge in point cloud perception research. Previous work [7], [8], [9] employed multiple datasets for the research of domain adaptation. However, one fact is often ignored both the scenario domain and sensor domain exist across different datasets. The existence of two variables at the same time makes it difficult to evaluate and analyze different domain adaptation approaches. In most cases, it is easy to collect data from

various situations using a single sensor. However, it becomes challenging to obtain data with identical scenarios using different sensors in real-world applications. This is because the background may stay constant, while the foregrounds are constantly changing.

Motivated by the issues mentioned above, a novel LiDAR-CS (LiDAR-Cross-Sensors) dataset has been proposed here which can well address these issues. In general, compared with existing benchmarks, LiDAR-CS dataset has the following unique properties.

**Cross Sensors.** Currently, LiDAR-CS includes samples from 6 types of LiDAR sensors. For each LiDAR frame, we have generated 6 duplicates with the same scenarios and the same foreground position, the only difference is the point distribution which comes from the inherent attributes of the sensors, such as the beam number, field of view, angle resolution, etc. In addition, the data can be easy to be extended by adding another LiDAR sensor while only needing to provide some real scanned frames.

**Consistent Annotation.** We generate all the sensors' data with the same background and foreground, therefore, we can keep consistent annotations for all the sensors. With the same annotation, the evaluation metric therefore can remain the same for all the data.

## II. RELATED WORK

**LiDAR Dataset for AD Perception.** Recently, numerous large-scale datasets have been released due to the rapid development of deep learning and autonomous driving. For the 3D object detection task, the most popular benchmarks are KITTI [1], nuScenes [2], and Waymo [3]. KITTI is the first large-scale public dataset for perception tasks in real AD scenarios. The point cloud is collected by a 64-beam Velodyne (VLD) LiDAR device. Three categories have been annotated in this benchmark while only the objects visible in the front camera view have been annotated. NuScenes is another dataset collected by a 32-beam LiDAR device. The point cloud frames are fully annotated with a 360° view. Objects are divided into 23 classes with attributes like visibility. Waymo Open Dataset [3] was conducted using 1 mid-range LiDAR and 4 short-range LiDARs. The categories of objects follow KITTI, but the frames are annotated in the full 360° view.

**Cross Domains Dataset.** Domain (or Sensor) generalizability is a very important property for deep learning-based approaches. A lot of benchmarks have been released for the 2D image domain [10] while few works have been proposed for the 3D point cloud. In [11], a **cross-sensor** dataset has been proposed with two types of LiDAR sensors (e.g., HDL-64 and VLP-32) and the transferability of features extracted from different sensors are also discussed, but unfortunately, the dataset is not released to the public. Waymo [3] proposes a **cross-scenario** domain adaptation task, which spans data from three different cities and split the data into two different domains. Experiments show that remarkable domain gaps exist between those cities.

**LiDAR Simulation.** LiDAR simulation [12] [13] is very popular in the field of robotics and AD. [14] collects the calibrated images and point cloud using the APIs provided by the video game engine, and applies these data for vehicle detection in their later work [15]. CARLA [16] and AutonoVi-Sim [17] also furnish the function to simulate LiDAR points data from the virtual world. Unlike previous simulators that entirely rely on CG models and game engines, in [18], an augmented simulator based on the "scan-and-simulate" pipeline has been proposed. Specifically, a vehicle with a LiDAR scanner is deployed to sweep the street of interest to obtain the background points cloud and then the foreground objects (e.g., CAD models) are placed on the background. LiDARsim [19] leverage the multi sweeps to build the dense background and reconstruct the foreground from cropped point cloud instead of CAD models. Nowadays popular NeRF [20] technique is introduced for LiDAR simulation [21], [22], [23].

**LiDAR-based 3D Object Detection.** LiDAR-based 3D object detection has been well-studied for many years. Generally, all these approaches can be divided into three categories by the way of LiDAR point cloud representation as 2D projection-based or view-based [24], [25], [26], [27], [28], [29], voxel-based [30], [31], [32], [33], [34], [35], pillar-based [36], [37], [38], [39] and raw point cloud-based [40], [41], [42], [43], etc. To evaluate the performance of existing detectors on our proposed cross-sensor benchmark, PointPillars, SECOND, PointRCNN, PV-RCNN and CenterPoint are utilized, and the details of the detectors are demonstrated in Sec. IV-A.

## III. LiDAR-CS DATASET

In this section, we present the LiDAR-CS Dataset and the methodology to build it. A pattern-aware LiDAR Simulator was designed to generate a realistic and high-quality dataset that can be used with most LiDAR sensors by inputting a few real scanned frames. Additionally, the section provides information about the sensors used and the settings employed in the dataset. III-B, followed by the details of the dataset and evaluation metric.

### A. Cross-Sensor Data Generation

As previously mentioned, obtaining cross-sensor data with identical scenarios in real-world applications is a challenging task. To overcome this challenge, data simulation techniques can be employed to generate data. In order to reduce the domain gap between real and simulated data, we utilize an augmented LiDAR simulator proposed in [18]. This simulator employs high-precision point cloud scans as a background and only synthesizes foreground objects with CAD models. Graphics rendering techniques are used to generate a realistic LiDAR point cloud. Our experimental results show that the generated point cloud performs similarly to real data for the 3D object detection task.

Furthermore, based on [18], we propose a pattern-aware LiDAR Simulation framework that is more general to suitable for different LiDAR sensors and combinations. The
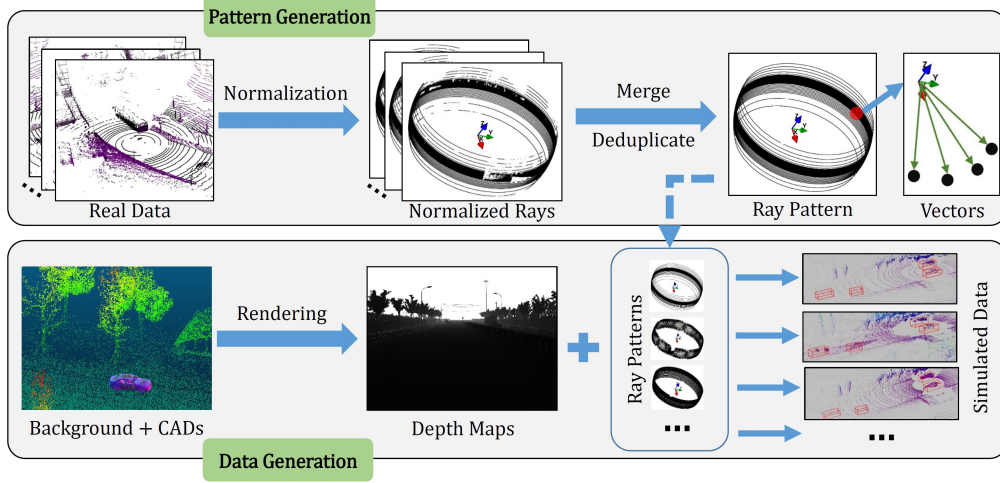
Fig. 2: The overview of the proposed pattern-aware LiDAR Simulator framework. First of all, the real LiDAR points are normalized to a spherical surface. Due to points missing, statistics information from multiple scans is required to build the LiDAR *Ray Pattern*. Then, the *Ray Pattern* vectors are simultaneously projected and query the depth value from the depth map to generate the simulation point cloud.

overview of the framework is shown in Fig. 2, which consists of two modules, the *Pattern Generation* module and the *Data Generation* module.

*1) Pattern Generation Module:* For sensor simulation, parameters azimuth resolution, vertical angles, and spin rate are prerequisites. However, not all the LiDAR point clouds can find the corresponding parameters of the sensor, and multi-sensor equipment is common for nowadays AD applications, therefore it is complicated to search the parameters for all the sensors. Moreover, the parameters in the manual usually are physically different from the real applications.

To well handle this problem, we present a *Pattern Generation* module, which can recover the LiDAR ray patterns from the scanned LiDAR points automatically and the LiDAR parameters are not prerequisites anymore. First of all, the point $p_i = (x_i, y_i, z_i), i \le N$, (here $N$ denotes the number of points in this LiDAR frame) is normalized into a unit spherical surface by

$$(\widetilde{x}_i, \widetilde{y}_i, \widetilde{z}_i) = \frac{(x_i, y_i, z_i)}{r}, \tag{1}$$

where

$$r = \sqrt{x_i{}^2 + y_i{}^2 + z_i{}^2}. \tag{2}$$

We call the normalized point cloud as *Ray Pattern* which represents the distribution of the rays of this sensor. Note that some laser rays will not return (some points are missing) in a certain frame due to different reasons, therefore the *Ray Pattern* from one frame may be incomplete. To handle this, we propose to merge the ray patterns across multi-frames and remove the duplicate rays to get a complete *Ray Pattern*.

*2) Data Generation Module:* With the estimated *Ray Pattern*, simulation LiDAR point cloud can be generated by leveraging the CG (Computer Graphics) techniques. We project the scene onto 6 faces of a cube centered at the LiDAR center to form the cube maps, the environment point cloud is rendered with surface splatting techniques to obtain

the smooth and holeless projection while the obstacle CAD models are rendered with regular methods.

However, the rendering module is the speed bottleneck of the system during the data generation processing. For multi-sensors simulation, we can project all the *Ray Patterns* to the depth map together and query the depth values simultaneously for each sensor. By this technique, the time-consuming point cloud simulation can remain nearly constant with the increasing number of sensors. Since the location and attributes of the synthesis CAD models are predefined, the ground truth of the foreground objects is annotated automatically during the point cloud simulation processing.

### B. LiDAR-CS Dataset

Currently, 6 different types of LiDAR sensors have been simulated here. The detailed information for each LiDAR sensor is listed below.

1) **Velodyne Puck 16** (VLD-16) has 16 channels and maximum 100m range with $30°$ vertical field-of-view (FOV).
2) **Velodyne 32E** (VLD-32) has 32 channels, and vertical FOV from $-30°$ to $+10°$.
3) **Velodyne HDL-64E** (VLD-64) has 64 channels, and 120m sensing range with $26.9°$ vertical FOV.
4) **Velodyne 128** (VLD-128) can capture the range of maximum 245m, and vertical FOV from $-25°$ to $+15°$.
5) **ONCE-40** is used in ONCE dataset [44] which is a 40-beams LiDAR sensor with vertical FOV from $-25°$ to $+15°$. However, the parameters of the sensor have not been provided in the original paper [44] which illustrates the necessity of our pattern-aware framework.
6) **Livox.** Different from the mechanical scanning LiDAR devices introduced above, Livox is a solid-state LiDAR with incommensurable retina-like [45] scanning patterns. We follow Livox Simu dataset [46] and remove the dynamic pattern character to simplify the simulation processing. The sensor contains five Horizon LiDAR
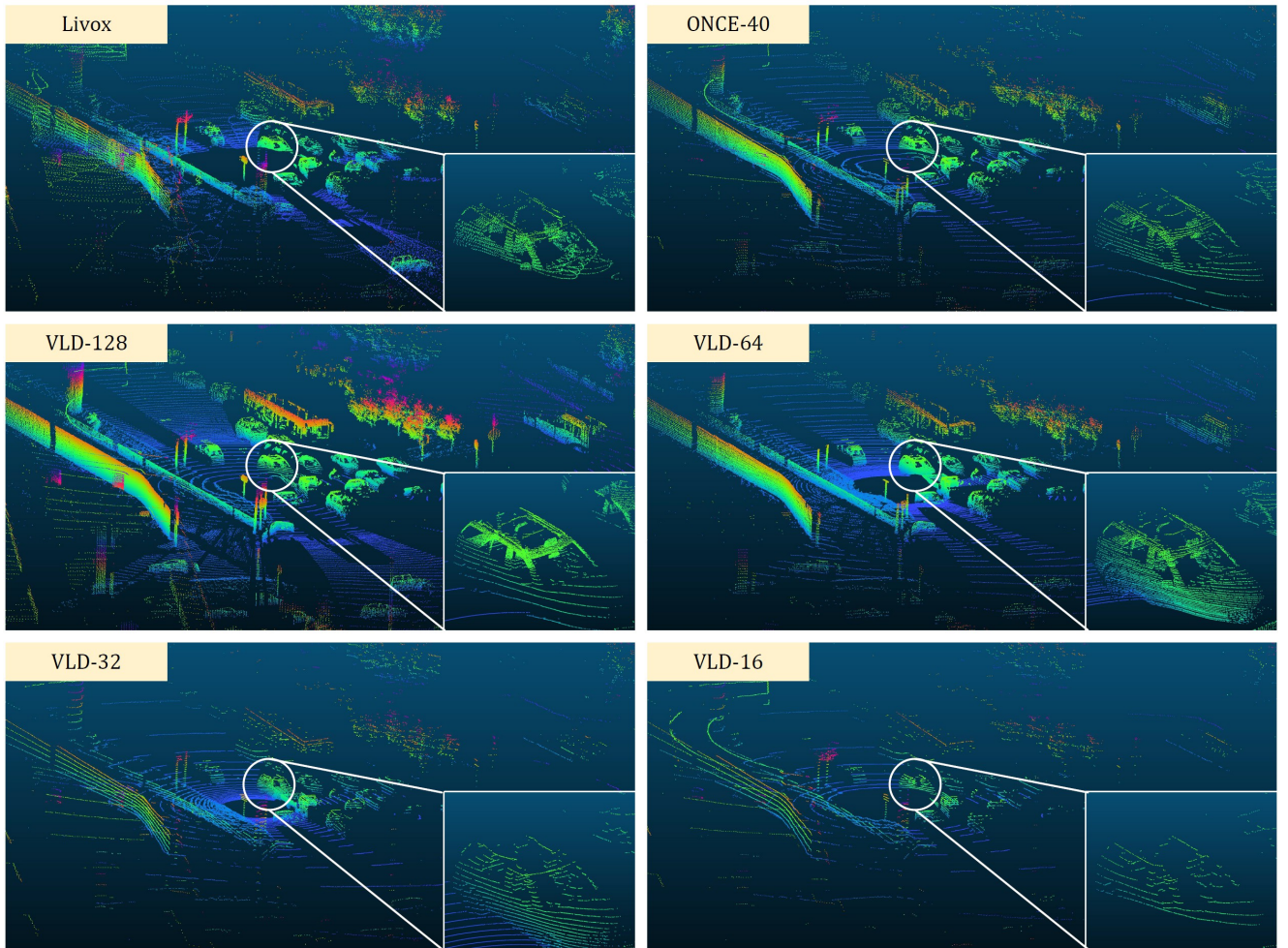
Fig. 3: An example of the LiDAR-CS dataset. All the point clouds are generated from the same scenario under different sensor patterns. The points in the cycle are zoomed in and shown in the white boxes for a better view. The point clouds are colorized by the height of the points. Better viewed in color.

devices and one Tele-15 LiDAR device.

All the sensors are set up at the same height (e.g.,, 2m in this work), and with $360°$ horizontal FOV.

### C. Dataset Details

We simulate 14,000 point cloud frames for each pattern from 6 different sensors. In total, LiDAR-CS Dataset contains 84,000 point cloud frames with 6 groups. We provide a semantic category and a cuboid bounding box as $(x, y, z, width, length, height, yaw\ angle)$ for the foreground object annotation. For each group, the data is divided into two splits, 7,000 frames for training and 7,000 frames for testing. For better comparison, all the groups share the same data splitting. For the foreground objects, five categories have been annotated in this dataset as *Car*, *Truck*, *Pedestrian*, *Bicyclist* and *Motorcyclist*. To be clear, the *Truck* category includes not only trucks but also other big vehicles such as vans and buses, etc. The number of objects in each category has been provided in Tab. I. Although the object number for "Car" is much larger than other categories, the object numbers for the rest categories are relatively balanced.

In Fig. 3, we illustrate an example of the LiDAR-CS data,

TABLE I: Data distribution of annotated 3D objects in LiDAR-CS dataset.

| Types | Car | Truck | Pedestrian | Bicyclist | Motorcyclist |
|---|---|---|---|---|---|
| 3D Object | 1218k | 164k | 230k | 69k | 130k |

in which all the background is from the same scenario and all the foreground objects are placed at the same location. In each sub-figure, we illustrate the LiDAR point cloud generated by different LiDAR patterns. For better visualization, a small part has been cropped and zoomed in to show the details of the object.

### D. Evaluation Metric

We follow the evaluation metric in ONCE benchmark [44]. Mean average precision (mAP) is calculated over the five categories while the orientations of the objects are also considered, objects will be regarded as false positive when the orientation fails to fall in $-90°$ to $90°$ of the ground truth orientation. Only objects within 70m are evaluated in our metric.

TABLE II: Cross evaluation on LiDAR-CS benchmark under five different LiDAR sensor patterns with five baseline detectors. "Ped." is short for "Pedestrian".

| Train \ Val | Methods | Livox mAP | Car | Truck | Ped. | ONCE-40 mAP | Car | Truck | Ped. | VLD-64 mAP | Car | Truck | Ped. | VLD-32 mAP | Car | Truck | Ped. | VLD-16 mAP | Car | Truck | Ped. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Livox | PointPillar | 54.13 | 77.21 | 83.29 | 25.41 | 57.46 | 77.29 | 85.25 | 27.65 | 58.15 | 77.34 | 85.37 | 27.09 | 38.79 | 51.76 | 62.54 | 23.02 | 30.38 | 42.87 | 47.54 | 20.38 |
| | SECOND | 57.39 | 75.56 | 83.15 | 25.54 | 61.04 | 77.04 | 85.28 | 27.25 | 61.52 | 76.97 | 85.20 | 26.67 | 40.14 | 49.48 | 60.63 | 22.89 | 29.39 | 38.88 | 42.89 | 18.51 |
| | PointRCNN | 36.50 | 53.82 | 63.66 | 10.07 | 42.90 | 53.87 | 69.68 | 17.40 | 38.86 | 51.82 | 67.69 | 11.64 | 27.05 | 36.96 | 40.83 | 10.21 | 22.23 | 30.75 | 30.06 | 8.71 |
| | PV-RCNN | 62.34 | 82.66 | 88.04 | 26.16 | 66.63 | 84.71 | 90.27 | 27.64 | 67.43 | 84.76 | 90.28 | 28.04 | 44.84 | 56.37 | 66.38 | 22.95 | 33.09 | 45.33 | 47.23 | 18.01 |
| | CenterPoint | 68.55 | 79.31 | 83.41 | 49.36 | 73.75 | 81.05 | 85.55 | 54.37 | 74.66 | 81.23 | 85.82 | 55.15 | 48.20 | 51.10 | 60.20 | 39.47 | 35.54 | 40.45 | 43.23 | 30.57 |
| ONCE-40 | PointPillar | 49.53 | 68.34 | 78.75 | 23.37 | 62.96 | 82.08 | 88.14 | 30.98 | 63.06 | 82.10 | 87.71 | 29.65 | 36.90 | 51.62 | 58.69 | 20.74 | 27.45 | 41.35 | 43.34 | 15.77 |
| | SECOND | 53.30 | 66.76 | 76.72 | 26.92 | 66.14 | 81.80 | 87.83 | 31.61 | 66.46 | 81.88 | 87.93 | 31.13 | 35.22 | 45.34 | 53.67 | 20.35 | 23.45 | 33.76 | 34.16 | 13.41 |
| | PointRCNN | 37.61 | 55.73 | 61.58 | 12.02 | 48.02 | 63.91 | 69.87 | 18.83 | 45.38 | 63.83 | 71.73 | 14.26 | 28.03 | 36.92 | 38.90 | 11.67 | 20.47 | 28.49 | 26.04 | 8.96 |
| | PV-RCNN | 56.26 | 70.18 | 79.12 | 25.74 | 70.42 | 87.12 | 90.59 | 30.66 | 71.26 | 87.22 | 90.62 | 31.39 | 39.60 | 52.52 | 56.90 | 18.32 | 27.20 | 41.61 | 35.49 | 12.48 |
| | CenterPoint | 61.92 | 68.61 | 76.73 | 46.33 | 77.19 | 84.23 | 88.36 | 58.17 | 77.50 | 84.20 | 88.09 | 58.97 | 42.07 | 48.77 | 53.98 | 32.77 | 29.26 | 37.22 | 35.81 | 22.86 |
| VLD-64 | PointPillar | 50.50 | 70.42 | 78.69 | 23.95 | 62.77 | 82.03 | 87.74 | 30.54 | 64.16 | 82.19 | 88.04 | 31.76 | 36.29 | 51.50 | 57.98 | 19.45 | 26.42 | 40.96 | 41.85 | 14.82 |
| | SECOND | 54.24 | 68.42 | 76.31 | 27.24 | 66.31 | 81.86 | 86.98 | 30.52 | 67.30 | 82.09 | 87.85 | 31.28 | 34.87 | 45.77 | 50.89 | 19.42 | 22.99 | 35.16 | 32.96 | 12.56 |
| | PointRCNN | 36.01 | 51.77 | 57.61 | 10.22 | 44.72 | 59.92 | 63.84 | 16.35 | 41.69 | 57.89 | 63.80 | 13.69 | 27.56 | 35.26 | 38.80 | 12.51 | 21.72 | 28.87 | 27.86 | 9.70 |
| | PV-RCNN | 57.39 | 72.15 | 81.03 | 25.32 | 70.55 | 88.91 | 90.77 | 29.68 | 71.97 | 89.15 | 90.95 | 30.74 | 39.94 | 54.38 | 57.49 | 17.51 | 27.47 | 42.16 | 37.60 | 11.29 |
| | CenterPoint | 62.16 | 68.89 | 75.93 | 47.10 | 76.88 | 84.18 | 87.95 | 57.14 | 78.00 | 86.08 | 88.18 | 59.23 | 41.86 | 48.51 | 53.72 | 32.92 | 27.70 | 36.34 | 35.25 | 20.66 |
| VLD-32 | PointPillar | 34.86 | 46.60 | 63.58 | 17.04 | 44.60 | 61.34 | 74.18 | 21.45 | 42.96 | 60.76 | 71.64 | 19.94 | 46.08 | 64.11 | 75.60 | 23.64 | 36.55 | 52.73 | 55.40 | 21.75 |
| | SECOND | 34.42 | 40.22 | 55.37 | 20.33 | 45.77 | 60.21 | 69.42 | 22.87 | 41.28 | 56.69 | 63.00 | 20.76 | 48.40 | 62.01 | 73.77 | 26.45 | 37.89 | 50.44 | 54.04 | 22.08 |
| | PointRCNN | 32.32 | 45.32 | 54.87 | 8.93 | 39.55 | 57.71 | 61.44 | 13.44 | 36.11 | 55.51 | 61.29 | 9.13 | 35.76 | 49.09 | 51.23 | 16.11 | 33.47 | 43.95 | 44.32 | 16.27 |
| | PV-RCNN | 35.46 | 43.67 | 56.81 | 17.96 | 49.05 | 71.07 | 70.14 | 19.89 | 44.97 | 66.61 | 63.68 | 18.38 | 53.45 | 71.48 | 79.37 | 24.64 | 43.61 | 59.75 | 61.38 | 21.54 |
| | CenterPoint | 41.84 | 42.97 | 55.43 | 25.73 | 55.14 | 62.47 | 69.69 | 36.68 | 51.23 | 59.74 | 67.01 | 34.52 | 56.94 | 63.55 | 73.26 | 41.50 | 43.83 | 51.56 | 52.35 | 34.28 |
| VLD-16 | PointPillar | 22.52 | 29.71 | 42.82 | 12.62 | 28.41 | 40.27 | 49.54 | 14.80 | 25.50 | 35.13 | 46.32 | 13.84 | 35.83 | 53.70 | 54.45 | 19.27 | 39.70 | 57.09 | 64.36 | 22.64 |
| | SECOND | 17.45 | 18.12 | 35.83 | 12.63 | 26.02 | 37.31 | 44.80 | 14.45 | 19.22 | 25.26 | 38.50 | 12.00 | 39.68 | 54.74 | 61.37 | 20.50 | 39.07 | 54.33 | 61.72 | 20.26 |
| | PointRCNN | 25.63 | 34.35 | 49.05 | 4.76 | 36.83 | 51.33 | 60.79 | 10.06 | 33.38 | 48.78 | 62.37 | 5.55 | 38.47 | 49.09 | 56.73 | 17.92 | 37.45 | 46.43 | 49.93 | 19.61 |
| | PV-RCNN | 16.21 | 18.19 | 37.62 | 8.15 | 26.05 | 41.23 | 50.46 | 10.22 | 17.65 | 26.66 | 40.64 | 5.12 | 46.18 | 65.20 | 72.13 | 21.27 | 46.28 | 64.43 | 69.61 | 22.63 |
| | CenterPoint | 20.41 | 20.94 | 32.83 | 10.69 | 26.09 | 36.12 | 41.47 | 15.78 | 17.42 | 26.03 | 34.02 | 5.39 | 48.69 | 56.36 | 62.15 | 34.86 | 47.56 | 54.49 | 61.44 | 34.39 |

## IV. EMPIRICAL STUDIES

On top of LiDAR-CS benchmark, we evaluate several 3D object detection baselines on the generalization ability and explore the influence of the points distribution domain gap.

### A. 3D Object Detection Baselines

To well analyze the performances of different detectors across different sensor data, we employ five state-of-the-art 3D object detection baselines here which include points-based, voxel-based, and anchor-free-based frameworks respectively. Details of these baselines are introduced below:

- **PointPillars** [36] transforms the points into vertical pillars to form the 2D feature map and the feature in each pillar is extracted with a light-weighted MLP network, then the 2D backbone is employed for the following detection task;
- **SECOND** [31] is an improved version of VoxelNet by introducing the sparse 3D convolution to replace the traditional 3D convolution which can boost both the 3D detection performance and efficiency;
- **PointRCNN** [47] is a purely points-based framework that utilizes PointNet++ as the backbone to extract features for the segmentation branch and the bounding box proposal branches respectively. In addition, the second refine module is also designed to refine the bound box proposal and leverage the cropped point cloud together with the point features extracted from the first stage;
- **PV-RCNN** [48] is a hybrid approach that learns discriminative features by combining both the 3D voxel and points representation;
- **CenterPoint** [49] which is an anchor-free-based method and achieves state-of-the-art performance on various benchmarks.

### B. Cross-Sensors Evaluation

Five mainstream detectors are employed here to reveal perception domain gaps between different LiDAR sensors. In our experiments, all the detectors are trained independently on the training split (2,000 frames of them are used as validation set during the training) of each point cloud group and evaluated on the testing split of all the groups, respectively.

The detailed evaluation results are shown in Tab. II. Limited by the space, the results for VLD-128 sensor data and categories of "Bicyclist" and "Motorcyclist" have not been given in this table, readers could refer to our dataset homepage for the complete evaluation results. To be clear, the **mAP** is still computed over all the five categories.

**Domain Gaps from Points Distribution.** From this table, we can obviously find some interesting observations. **Firstly**, the model trained and tested on the same sensor gives the best performance in most situations. This is easy to understand because they come from the same sensor therefore the point cloud distribution is exactly identical. **Secondly**, the detectors trained on a low-resolution dataset can achieve comparable performance on the high-resolution testing dataset while the opposite is not true. Taking the detectors trained on VLD-32, they achieve similar performances by testing on VLD-32 and VLD-64 while the performances drop dramatically on VLD-16. **Thirdly**, the domain gaps positively correlate to the similarity of point ray distribution. For example, models trained on VLD-32 data outperform models trained on VLD-64 data when testing on VLD-16 dataset, and vice versa. **Finally**, we find that some sensors exhibit similar performance, such as ONCE-40 and VLD-64, and the mAP results are very close whether as training data or testing

data. This is because even though they have different beam numbers, they share similar vertical angles' distribution.

**Detectors Evaluation.** We also compare the performances of different detectors. Overall, PV-RCNN achieves the best performance on large obstacles such as "Car" and "Truck", while CenterPoint outperforms other baseline detectors with a remarkable gap in the "Pedestrian" category. When facing the domain gap situation, the performances of all the baselines decrease dramatically, especially for PointRCNN which relies on a Points-based backbone for feature extraction and the extracted features will change dramatically with the points distribution change. Relatively speaking, voxel-based methods have the ability to deal with the change of point distribution to some extent. Experimental results illustrate that mainstream baseline detectors do not have the points-distribution-insensitive ability. We propose the hypothesis that the extracted features from the detectors are disabled to learn the real structure of the objects or underlying surfaces, but are confined to learning the distribution of points.

### C. Domain Alignment Evaluation

To handle the domain gap between different sensor data, an intuitive way is to align the laser beams from the source sensor to the target sensor. To our knowledge, few approaches and datasets have been proposed to solve the cross-sensors gap problem. Two baseline domain alignment methods (e.g., ST3D [8] and SN [7]) are employed for evaluation here. SN takes the BBoxs' size from the target dataset to normalize the BBoxs in the source dataset during the training while ST3dD uses a *Random Object Scaling* strategy to expand the diversity of object sizes.

For comparison, we propose a simple strategy, NNDS (Nearest-neighbor scan down-sampling), which reduces the domain of point distribution by downsampling the source point cloud along the target scan distribution, as in [50]. But different from [50] simply dropping the scan lines uniformly, NNDS utilizes nearest-neighbor scan searching with target data as a reference and drops the other lines.

Tasks of VLD-64 → VLD-32 and VLD-64 → VLD-16 are given in Tab. III. PointPillars and CenterPoint are employed here as baseline detectors. From the table, we can see that the baselines perform worse than the NNDS. Both *ST3D* and *SN* are proposed for solving the domain gap that comes from the different scenarios. But our dataset is proposed to simulate the domain gap coming from different LiDAR sensors and the statistical information of objects from different sensors are the same. Therefore, our dataset introduces a new challenge for domain alignment.

### V. POTENTIAL APPLICATIONS AND DISCUSSION

The potential applications of the LiDAR-CS dataset are virtually limitless. Although originally designed for 3D object detection, it could also be repurposed for other purposes.

**Data Domain Adaptation** The domain gaps between different sensors, simulation, and real-world, are very important topics for real AD applications. The proposed pattern-aware

TABLE III: Evaluation for cross-sensor domain alignment. Two detectors are used here: SECOND with IoU head and PV-RCNN. The improvement based on the source-only method is highlighted in blue.

| Task | Method | mAP (%) | |
|---|---|---|---|
| | | SECOND-IoU | PV-RCNN |
| VLD-64 → VLD-32 | Source Only | 34.56 | 39.94 |
| | SN | 34.56 (**0** ↑) | 39.94 (**0** ↑) |
| | ST3D | 34.89 (**0.33** ↑) | 40.43 (**0.49** ↑) |
| | NNDS | **48.02** (**13.46** ↑) | **52.29** (**12.35** ↑) |
| VLD-64 → VLD-16 | Source Only | 23.01 | 27.47 |
| | SN | 23.01 (**0** ↑) | 27.47 (**0** ↑) |
| | ST3D | 22.95 (**-0.06** ↑) | 27.78 (**0.31** ↑) |
| | NNDS | **39.76** (**16.75** ↑) | **44.85** (**17.38** ↑) |

point cloud simulation technique can be employed for generating any type of LiDAR sensor.

**Distribution-insensitive features.** Our LiDAR-CS dataset could be applied to evaluate the anti-interference ability of features extracted from different backbones, as we showed in Tab. II. Furthermore, perception systems with sensor-invariant ability can also be evaluated.

**Large-scale Point Cloud Up-sampling or Re-sampling.** While there are many methods for up-sampling point clouds, most have only been tested on small samples from CAD models with a few thousand points. However, few studies have focused on up-sampling LiDAR point clouds in large-scale autonomous driving scenarios. Our dataset includes realistic point clouds from various scenarios and large-scale frames with different resolutions, making it an ideal resource for researching point cloud up-sampling and re-sampling.

**Semantic Segmentation.** Since the ground truth bounding box of all objects is provided, obtaining the semantic label for foreground objects is easy. Our dataset can also be used for semantic segmentation.

**Sensor Selection.** Our approach can also be used to select LiDAR hardware and adjust parameters. Instead of relying on an intuitive selection of device type and installation parameters, or choosing device combinations, and then collecting and annotating data, and training perception models, it would be more scientific and safer to evaluate devices and settings with perception systems beforehand using low-cost simulation data.

### VI. CONCLUSION

In this paper, we propose LiDAR-CS Dataset, which fills in the missing benchmark on points distribution domain adaptation and sensor-insensitive detectors for 3D object detection. For generating LiDAR-CS Dataset, we present a *pattern-aware LiDAR Simulator* to simplify the calculation of sensor ray tracing and accelerate the data generation. Through empirical experiments on baseline detectors using various sensor devices, we have gained insightful conclusions and heuristic understanding which can positively impact future research. Our dataset has been made available to the public, with the hope that it will encourage further research on 3D point cloud technology. The LiDAR-CS dataset will continue to support more LiDAR sensors and perception tasks in the future.

## REFERENCES

[1] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *CVPR*, 2012, pp. 3354–3361. 1, 2

[2] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020, pp. 11 621–11 631. 1, 2

[3] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *CVPR*, 2020, pp. 2446–2454. 1, 2

[4] "Velodyne lidar," https://velodynelidar.com/. 1

[5] "Hesai tech," https://www.hesaitech.com/en/. 1

[6] "Livox tech," https://www.livoxtech.com/. 1

[7] Y. Wang, X. Chen, Y. You, L. E. Li, B. Hariharan, M. Campbell, K. Q. Weinberger, and W.-L. Chao, "Train in germany, test in the usa: Making 3d object detectors generalize," in *CVPR*, 2020, pp. 11 713–11 723. 1, 6

[8] J. Yang, S. Shi, Z. Wang, H. Li, and X. Qi, "St3d: Self-training for unsupervised domain adaptation on 3d object detection," in *CVPR*, 2021, pp. 10 368–10 378. 1, 6

[9] B. Zhang, J. Yuan, B. Shi, T. Chen, Y. Li, and Y. Qiao, "Uni3d: A unified baseline for multi-dataset 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 9253–9262. 1

[10] X. Peng, B. Usman, K. Saito, N. Kaushik, J. Hoffman, and K. Saenko, "Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation," *arXiv preprint arXiv:1806.09755*, 2018. 2

[11] C. B. Rist, M. Enzweiler, and D. M. Gavrila, "Cross-sensor deep domain adaptation for lidar detection and segmentation," in *IV*. IEEE, 2019, pp. 1535–1542. 2

[12] S. Lee, D. Kang, S. Cho, S. Sim, Y. W. Park, K. Um, and K. Cho, "Lidar simulation method for low-cost repetitive validation," in *Advances in computer science and ubiquitous computing*. Springer, 2015, pp. 237–242. 2

[13] A. Tallavajhula, Ç. Meriçli, and A. Kelly, "Off-road lidar simulation with data-driven terrain primitives," in *ICRA*. IEEE, 2018, pp. 7470–7477. 2

[14] X. Yue, B. Wu, S. A. Seshia, K. Keutzer, and A. L. Sangiovanni-Vincentelli, "A lidar point cloud generator: from a virtual world to autonomous driving," in *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, 2018, pp. 458–464. 2

[15] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud," in *ICRA*. IEEE, 2018, pp. 1887–1893. 2

[16] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16. 2

[17] A. Best, S. Narang, L. Pasqualin, D. Barber, and D. Manocha, "Autonovi-sim: Autonomous vehicle simulation platform with weather, sensing, and traffic control," in *CVPR Workshops*, 2018, pp. 1048–1056. 2

[18] J. Fang, D. Zhou, F. Yan, T. Zhao, F. Zhang, Y. Ma, L. Wang, and R. Yang, "Augmented lidar simulator for autonomous driving," *RAL*, vol. 5, no. 2, pp. 1931–1938, 2020. 2

[19] S. Manivasagam, S. Wang, K. Wong, W. Zeng, M. Sazanovich, S. Tan, B. Yang, W.-C. Ma, and R. Urtasun, "Lidarsim: Realistic lidar simulation by leveraging the real world," in *CVPR*, 2020, pp. 11 167–11 176. 2

[20] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021. 2

[21] C. Li, Y. Ren, and B. Liu, "Pcgen: Point cloud generator for lidar simulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 676–11 682. 2

[22] T. Tao, L. Gao, G. Wang, P. Chen, D. Hao, X. Liang, M. Salzmann, and K. Yu, "Lidar-nerf: Novel lidar view synthesis via neural radiance fields," *arXiv preprint arXiv:2304.10406*, 2023. 2

[23] J. Zhang, F. Zhang, S. Kuang, and L. Zhang, "Nerf-lidar: Generating realistic lidar point clouds with neural radiance fields," *arXiv preprint arXiv:2304.14811*, 2023. 2

[24] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–8. 2

[25] W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net," in *CVPR*, 2018, pp. 3569–3577. 2

[26] B. Yang, W. Luo, and R. Urtasun, "Pixor: Real-time 3d object detection from point clouds," in *CVPR*, 2018, pp. 7652–7660. 2

[27] Z. Liang, M. Zhang, Z. Zhang, X. Zhao, and S. Pu, "Rangercnn: Towards fast and accurate 3d object detection with range image representation," *arXiv preprint arXiv:2009.00206*, 2020. 2

[28] D. Zhou, J. Fang, X. Song, C. Guan, J. Yin, Y. Dai, and R. Yang, "Iou loss for 2d/3d object detection," in *3DV*. IEEE, 2019, pp. 85–94. 2

[29] P. Hu, J. Ziglar, D. Held, and D. Ramanan, "What you see is what you get: Exploiting visibility for 3d object detection," in *CVPR*, 2020, pp. 11 001–11 009. 2

[30] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *CVPR*, 2018, pp. 4490–4499. 2

[31] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018. 2, 5

[32] H. Kuang, B. Wang, J. An, M. Zhang, and Z. Zhang, "Voxel-fpn: Multi-scale voxel feature aggregation for 3d object detection from lidar point clouds," *Sensors*, vol. 20, no. 3, p. 704, 2020. 2

[33] J. Yin, J. Shen, C. Guan, D. Zhou, and R. Yang, "Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention," in *CVPR*, 2020, pp. 11 495–11 504. 2

[34] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, "Voxel r-cnn: Towards high performance voxel-based 3d object detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 2, 2021, pp. 1201–1209. 2

[35] J. Mao, Y. Xue, M. Niu, H. Bai, J. Feng, X. Liang, H. Xu, and C. Xu, "Voxel transformer for 3d object detection," in *ICCV*, 2021, pp. 3164–3173. 2

[36] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *CVPR*, 2019, pp. 12 697–12 705. 2, 5

[37] G. Shi, R. Li, and C. Ma, "Pillarnet: Real-time and high-performance pillar-based 3d object detection," in *European Conference on Computer Vision*. Springer, 2022, pp. 35–52. 2

[38] J. Li, C. Luo, and X. Yang, "Pillarnext: Rethinking network designs for 3d object detection in lidar point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 567–17 576. 2

[39] S. Zhou, Z. Tian, X. Chu, X. Zhang, B. Zhang, X. Lu, C. Feng, Z. Jie, P. Y. Chiang, and L. Ma, "Fastpillars: A deployment-friendly pillar-based 3d detector," *arXiv preprint arXiv:2302.02367*, 2023. 2

[40] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *CVPR*, 2018, pp. 918–927. 2

[41] D. Zhou, J. Fang, X. Song, L. Liu, J. Yin, Y. Dai, H. Li, and R. Yang, "Joint 3d instance segmentation and object detection for autonomous driving," in *CVPR*, 2020, pp. 1839–1849. 2

[42] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3dssd: Point-based 3d single stage object detector," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 040–11 048. 2

[43] X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, "3d object detection with pointformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7463–7472. 2

[44] J. Mao, M. Niu, C. Jiang, H. Liang, J. Chen, X. Liang, Y. Li, C. Ye, W. Zhang, Z. Li, *et al.*, "One million scenes for autonomous driving: Once dataset," *arXiv preprint arXiv:2106.11037*, 2021. 3, 4

[45] Z. Liu, F. Zhang, and X. Hong, "Low-cost retina-like robotic lidars based on incommensurable scanning," *IEEE/ASME Transactions on Mechatronics*, 2021. 3

[46] "Livox simulation dataset," https://livox-wiki-en.readthedocs.io/en/latest/data_summary/dataset.html#livox-simu-dataset-v1-0-cn/. 3

[47] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *CVPR*, 2019, pp. 770–779. 5

[48] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection," in *CVPR*, 2020, pp. 10 529–10 538. 5

[49] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *CVPR*, 2021, pp. 11 784–11 793. 5

[50] I. Alonso, L. R. Montesano, A. C. Murillo, *et al.*, "Domain adaptation in lidar semantic segmentation," *arXiv preprint arXiv:2010.12239*, 2020. 6