Simulation-to-Reality Domain Adaptation for Offline 3D Object Annotation on Pointclouds with Correlation Alignment

Weishuang Zhang, B. Ravi Kiran, Thomas Gauthier, Yanis Mazouz and Theo Steger
Navya, France

Keywords: Pointclouds, Object Detection, 3D, Simulation, Unsupervised Domain Adaptation.

Abstract: Annotating objects with 3D bounding boxes in LiDAR pointclouds is a costly human driven process in an autonomous driving perception system. In this paper, we present a method to semi-automatically annotate real-world pointclouds collected by deployment vehicles using simulated data. We train a 3D object detector model on labeled simulated data from CARLA jointly with real world pointclouds from our target vehicle. The supervised object detection loss is augmented with a CORAL loss term to reduce the distance between labeled simulated and unlabeled real pointcloud feature representations. The goal here is to learn representations that are invariant to simulated (labeled) and real-world (unlabeled) target domains. We also provide an updated survey on domain adaptation methods for pointclouds.

1 INTRODUCTION

Many self-driving vehicles (SDV) rely on LiDAR (Light Detection And Ranging) technology to perceive their surroundings. There are multiple real-world SDV largescale LiDAR annotated datasets including KITTI (Geiger et al., 2013), nuScenes (Caesar et al., 2019), Waymo (Sun et al., 2020), Lyft (Kesten et al., 2019), Semantic-KITTI (Behley et al., 2019), nuScenes LiDAR-Seg (Fong et al., 2021). This has provided a large performance gain across various supervised 3D detection and segmentation perception pipelines. Though generating annotated pointcloud datasets is a costly, meticulous & time consuming process requiring a large number of human annotators. Annotating real data also poses physical constraints on the position, number of obstacles as well as type of dynamic scenarios.

Simulators have become a cheaper and scalable alternative in terms of scenario diversity and time of training. In comparison to simulation, real world annotation pipelines have these key operational issues:

1. Ensuring sensor calibration and synchronization (e.g. Camera-Lidar or Radar-Lidar) to achieve precise annotations when the pointclouds are sparse.
2. Most road-datasets contain car as majority classes due to the domain of operation. Though in real world operations, certain zones can contain more pedestrians. There is a change in class distribution between the training and test domains.
3. Furthermore, annotation is often performed on dense LiDAR pointclouds (64/32 layers). Transfer learning from datasets to sparse multi-Lidar pointclouds remains a big challenge.

To address these problems, autonomous driving simulators such as CARLA (Dosovitskiy et al., 2017) can provide inexpensive source of synthetic annotated data. Our contributions include :

- A short incremental review on the state of domain adaptation for pointclouds tasks, classification, semantic segmentation & detection.
- A case study on 3D-object detection on our deployment vehicle's pointclouds which evaluates the application of domain invariant representation learning using the correlation alignment loss (CORAL) between simulation and real pointclouds
- A qualitative analysis of the sources of the domain gap between simulated and real pointclouds.

1.1 Domain Adaptation (DA) on Pointclouds

In a typical deep learning application such as object detection using LiDAR or Camera, a crucial assumption made is that the training dataset domain

Zhang, W., Kiran, B., Gauthier, T., Mazouz, Y. and Steger, T.
Simulation-to-Reality Domain Adaptation for Offline 3D Object Annotation on Pointclouds with Correlation Alignment.
DOI: 10.5220/0011059200003209
In *Proceedings of the 2nd International Conference on Image Processing and Vision Engineering (IMPROVE 2022)*, pages 142-149
ISBN: 978-989-758-563-0; ISSN: 2795-4943

Table 1: Categorization of Domain adaptation methods for LiDAR pointclouds.

DA Methods	Description
Aligning I/O representations	Dataset-to-Dataset(D2D) transfer (Triess et al., 2021) involves transfer learning between LiDAR Datasets collected with different LiDAR configurations (number, scanning patter, spatial resolution, different classes/label spaces) would require alignment either by upsampling, downsampling, re-sampling of pointclouds. (Alonso. et al., 2021) class sensitive data augmentation. These methods are frequently hand-engineered. (Tomasello et al., 2019) present a Deep Sensor Cloning methods which enables the generation of pointclouds from expensive LiDARs (HDL64) using CNNs along with in-expensive LiDARs (Scala).
Modeling Physics	Authors (Hahner et al., 2021) have proposed a fog simulation method in pointclouds that is applicable to any LiDAR dataset. (Zhao et al., 2020) learn dropout noise from real world data.
Adversarial Domain Mapping (ADM)	Learns a conditional mapping from source domain samples to their target domain samples using Generative Adversarial Networks (GANs). ADM can enable Simulation-to-Real(S2R) DA. Annotations from simulation can be leveraged by mapping Simulated clouds to real target domain clouds with subsequent training using source domain labels. Authors (Sallab et al., 2019) map simulated BEV images to real world equivalents while improving object detection performance.
Domain Invariant Learning	These methods are usually adversarial methods that align the feature spaces between source & target pointcloud domains, thus enforcing consistent prediction on target domain. CORAL loss based on (Sun et al., 2017) belongs to this family. Authors (Langer et al., 2020) generate semi-synthetic pointclouds from the source data while performing correlation alignment between synthetic target scans and target scans.
Simulation-To-Real (S2R)	These families of methods focus on reducing the domain gap between simulation and reality. Authors (DeBortoli et al., 2021) claims to have encouraged the 3D feature encoder to extract features that are invariant across simulated and real scenes. Authors (Huang and Qiao, 2021) generate synthetic pointclouds to train classification models instead of aligning features. They are thus able to highlight which part of the object is transferred.

(also called source domain) and test data domain (target domain) share the same feature space distribution. This could be broken in multiple ways (non IID sampling, IID referring to Independent and identically distributed) and is a key constraint in the performance of DNNs in open operational domains. Domain adaption is a set of transformations (or representation learning) that aligns the features between source and target domains. Based on the availability of labels in the target domain, DA can be supervised or unsupervised. Transfer learning is a subset of supervised DA where labels in source domain can be used to fine tune DNNs to their target domains, though this is usually a costly process. Unsupervised DA usually operates in target domains where there are either no or very few labels.

Authors (Triess et al., 2021) and (Borna Bešić and Valada, 2021) provide surveys on DA methods for the perception tasks (segmentation/detection/classification of pointclouds) in LiDAR. We provide a summary of the taxonomy of methods described by this survey in table 1. We have updated the survey with new references and methods from recent literature. Majority of these methods are focused on unsupervised DA where there are no target domain labels available. The goal here is to highlight potential DA methods that could be used to perform Simulation-To-Real(S2R) domain adaptation.

1.2 Simulation-to-Real DA

In this subsection, we summarize key studies using simulation-to-real(S2R) domain adaptation (DA) methods. This implies pre-training on simulated pointclouds while evaluating on real-world pointclouds. Authors (Yue et al., 2018) demonstrate the first pointcloud simulator while showing a significant improvement in accuracy (+9%) in pointcloud semantic segmentation by augmenting the training dataset with the generated synthesized data.

Another key issue in simulating LiDAR pointclouds is generating sensor & material dependant intensity channel output. Most simulators do not model the intensity function, though mostly modeling ray tracing and illumination operations. Authors (Wu

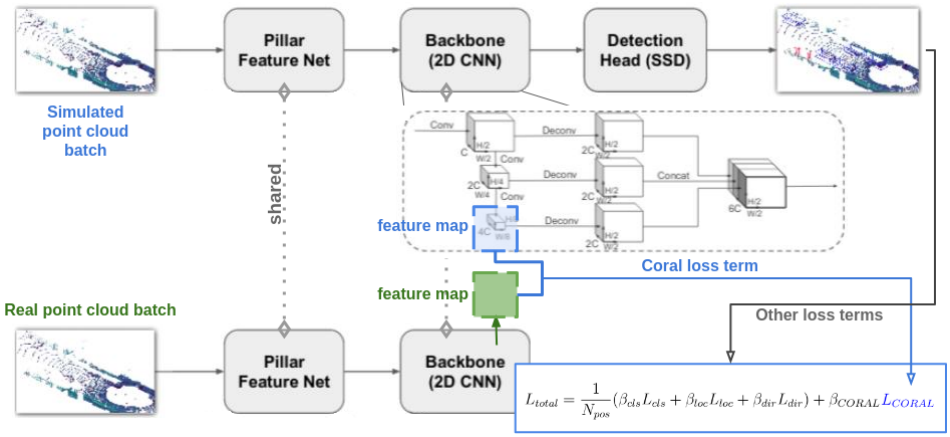


Figure 1: We reproduce the Pointpillars pipeline along with the CORAL loss between real & simulated feature maps.

et al., 2019) proposed a learned intensity rendering network which regresses the intensity as a function of the xyz coordinate channels in the range image. However this mapping from xyz-intensity is a highly non-stationary function, since similar geometrical surfaces (such as building wall and metal) could have drastically different intensity values. Thus learning these mappings is a difficult process. (Vacek et al., 2021) propose the use of RGB data along with xyz coordinates to improve intensity regression on polished parts of car bodyworks, windows, traffic signs and license/registration plates.

(Dworak et al., 2019) demonstrate the Sim-to-Real transferability for 3D object detection between CARLA and KITTI, using different mixtures (training on different combinations of datasets, sequential fine tuning on the 2 datasets) of real and simulated data to train Yolo3D, Voxelnet and Pointpillar architectures. They demonstrate there are significant gains in performance for object detection. (Brekke et al., 2019) evaluate simulated pre-training on both camera images and lidar scans from CARLA, while training a AVOD-FPN network. Authors remark that real world data cannot be replaced though simulated data can considerably reduce the amount of training data required to achieve target accuracy levels in the detection task.

Authors (Fang et al., 2020) and (Manivasagam et al., 2020) perform Real-To-Simulation (R2S) modeling of pointclouds, where real-world scans (3D maps of background) are used to build a catalog of diverse scenarios. Recorded dynamic objects are then inserted into these existing scenarios. This is then used to train a robust 3D object detection. Authors (Saltori et al., 2020) study dataset-to-dataset domain adaptation between KITTI-nuScenes leveraging motion coherence across detections, reversible scale transformations and pseudo-annotations. Authors (Deschaud et al., 2021) have created the Paris-CARLA-3D dataset, with simulated pointcloud and camera data generated in CARLA while the real world data was logged in Paris. The goal of the dataset is to evaluate the unsupervised domain adaptation from CARLA to Paris data.

Authors (Meng et al., 2020) have also explored weakly-supervised learning (annotating horizontal centers of objects in bird's view scenes) while learning to predict a full 3D bounding box.

2 SIMULATION-TO-REAL ON CUSTOM LiDAR DATASET

In this section we focus on our experimental demonstration. Our goal here is to summarize the different components of the simulation-to-real domain adaptation experiment, namely the object detection architecture on pointclouds, the features being used by the CORAL loss to perform correlation alignment between feature maps. This experiment shall be carried out on our proprietary simulated and real pointcloud dataset.

2.1 Simulated & Real Datasets

The simulated pointcloud dataset was generated with 16k multi-lidar scans. There was no "Cyclist" class included in the dataset and mainly constituted of "Pedestrian" and "Car" classes. The class frequencies in the simulated dataset is show in figure 2. The plots also demonstrate the polar histograms (range, azimuth), where each cell in the plot contains the normalized frequencies of bounding boxes visible across the dataset.

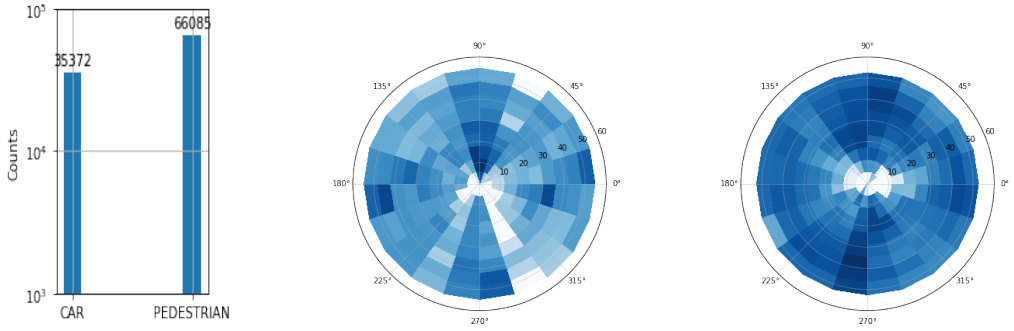
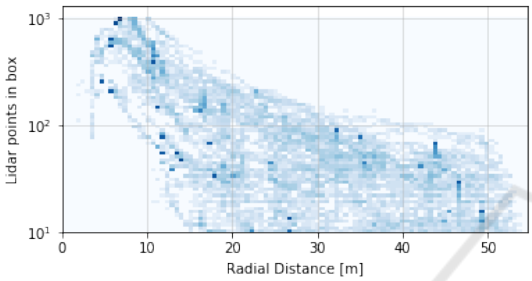
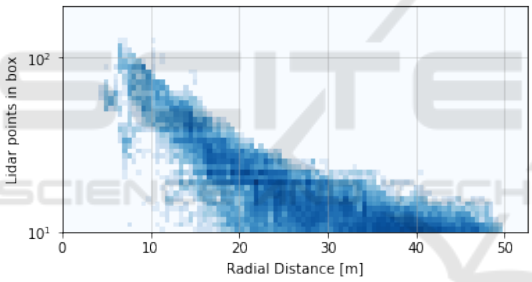


Figure 2: Class histogram in simulated dataset, along with polar density map for box annotations in dataset. The intensity values were log-scaled.



Car.



Pedestrian.

Figure 3: Number of points with ground truth boxes as the range of the box varies in the simulated dataset.

Real Pointclouds. The real pointcloud dataset was constructed on board the target vehicle in operation at our test site.

Labeled: We annotated a small dataset of LiDAR pointclouds coming from the vehicles real domain of operation. This contains merely 223 scans, arranged in 4 continuous sequences. Each LIDAR scan contains 16.9 points, 2k in minimum and 20.7 in maximum, with a median equals to 16.7k.

Unlabeled: To performed unsupervised domain adaptation we used a large collection (1000 scans) of LiDAR pointclouds coming from the vehicles real domain of operation with different obstacles (pedestrian and cars mainly) in varied configurations.

Point Density per Box. The plot in figure 3 shows the number points within ground truth boxes for each category in the y axis, while the range/radius at which the box is present. The plot for the car category is very interesting as it doesn't follow the pattern in the public dataset, the relationship between point density per box and distance to box center is looser in this dataset. This empirical parameter is key to ensure robust feature extraction for the car category at different point density within each bounding box. Filtering out bounding boxes that contain very few points in the real world dataset thus is demonstrated as a key manual engineering step that directly affects the quality of features being extracted within any given object detection framework.

2.2 Pointpillars Architecture

The pipeline of the Pointpillars consists of three main parts: Pillar Feature Network, Backbone and SSD Detection Head, as shown below.

Pillar Feature Net is a feature encoder network that converts a pointcloud to a sparse pseudo-image composed by two modules. The input pointcloud is first discretised into an evenly spaced grid in the x-y plane. A tensor of size (D,N) is then calculated for each voxel, where D is the dimension of feature for each sampled point in the pillar and N is the maximum number of points per pillar. Thus the pointcloud is converted into a stacked-pillars tensor of size (D,P,N), where P denotes the number of non-empty pillars per pointcloud.

A layer consists of a Linear-BatchNorm-ReLU follows to extract pillar-wise features, with max pooling over the channels to create an output tensor of size (C,P). Then the features are scattered back to the original pillar locations to create a pseudo-image of size (C,H,W), where C is number of channels fixed to 64.

Backbone consists in blocks of top-down 2D CNNs, which can be characterized by a series of blocks. Each top block has 2D convolutional layers to reduce the 2D tensor sizes into half, followed by BatchNorm and a ReLU. The processed tensors are

combined through upsampling and concatenation, as shown above.

SSD Detection Head is a detection head that finally detects and regresses 3D boxes in Pointpillars. The prior boxes are matched to the ground truth using IoU.

2.3 Coral Loss

We use the CORAL loss (Sun et al., 2017) to reduce the geodesic distance between the simulated and real pointclouds. The same method could also be used to minimize distance between embeddings of pointclouds coming from different LiDAR configurations.

The CORAL loss is described between two domains for a single feature layer. Given source-domain training examples $D_S = \{x_i\}, x \in R^d$ with labels $L_s = \{y_i\}, i \in \{1,2,...,L\}$, and unlabeled target data $D_T = \{u_i\}, x \in R^d$. Suppose the number of source and target data are n_S and n_T respectively. Here, both x and u are the d-dimensional deep layer activations $\phi(I)$ of input I that we are trying to learn. Let D_S^{ij}/D_T^{ij} be the j-th dimension of the i-th source/target data, C_S/C_T denote the 2D feature covariance matrices. The CORAL loss ie. the distance between the source and target features is defined as:

$$L_{DA} = \frac{1}{4d^2}||C_S - C_T||_F^2 \quad (1)$$

where $||\cdot||_F^2$ denotes the squared matrix Frobenius norm.

2.4 Pipeline

Figure 1 shows the Pointpillars architecture using CORAL loss for deep domain adaptation.

The two mini-batches batches of simulate and real pointclouds pass through the shared backbone pipeline. The CORAL loss is evaluated at the end of the 2D backbone as the feature map considering its reduced size as shown in figure 1.

The shape of this feature map is (bn, 4C, H/8, W/8), where bn is the batch size, C is the channel number output by PFN, which is set to 64, while H and W represent the size of the pillar grids in the xy plane.

For a pointcloud ranged in $\pm 50m$ both in x and y with a grid size of 0.25m, the feature map shape in our experiment is (bn, 256, 50, 50). As the CORAL loss L_{CORAL} needs a pair of 2-dimension inputs, we chose to reshape the feature map into (256bn, 2500). And the Loss function is modified to:

$$L_{total} = \frac{1}{N_{pos}}(\beta_{cls}L_{cls} + \beta_{loc}L_{loc} + \beta_{dir}L_{dir}) + \beta_{DA}L_{DA} \quad (2)$$

Where β_{DA} is the weight for the CORAL loss. L_{cls}, L_{loc} and L_{DA} represent the classification, localization and CORAL domain adaptation loss terms.

3 EXPERIMENT & RESULTS

In this section, We describe the dataset setup and experiments performed with different hyperparameters to demonstrate the effect of adding a domain adaptation loss based on CORAL.

In our study we use the data from simulation which contains 12646 scans as labeled pipeline input, and 2 sequences of unlabeled real vehicle data which contains around 500 scans. Both of the input pointclouds are ranged within 50m in both x and y axis, $[-3,1]$m in z axis. Similar to what is done in nuScenes, the point pillar is sized to 0.25m×0.25m with max number of points per pillar set to 60. Zero padding is applied for not fully filled pillars.

Performance metrics measured were using the official KITTI evaluation toolkit (Geiger et al., 2013) and nuScenes devkit. The four metrics used in our experiments denote respectively as mean average precision in BEV, 3D, 2D (image plane), and Average Orientation Similarity (AOS). In this section we'll step from basic conceptions to explain how these metrics are calculated. Some of them are not directly used but fundamental to understand the PASCAL criteria introduced in object detection.

The performance metrics of simulation trained model on real data can be seen in table 2 . We present 3 experiments on sim-to-real domain adaptation, the loss curves are shown below in 4. The three models are represented by different colors, where the blue ones denotes the model with batch size 8, CORAL loss weight $\beta_{DA} = 1e8$, the violet ones denote the model with batch size 4, CORAL loss weight $\beta_{DA} = 1e8$, and the red ones denotes the model with batch size 4, CORAL loss weight $\beta_{DA} = 1e9$. The localization, classification and CORAL losses are shown. The weight β_{DA} was chosen to balance the contribution of the localization, classification & CORAL losses.

From these loss curves we find that the CORAL loss converges quickly in the first several training epochs. The final converged loss shares a similar value despite the large weighting, while using a larger batch size seems to result in a smaller CORAL loss magnitude. A higher CORAL loss weight hinders the descent of classification/location loss curves over the source domain simulated Shuttle A. We conclude that applying appropriate weighted CORAL loss with short training epochs improves the performance on real data.

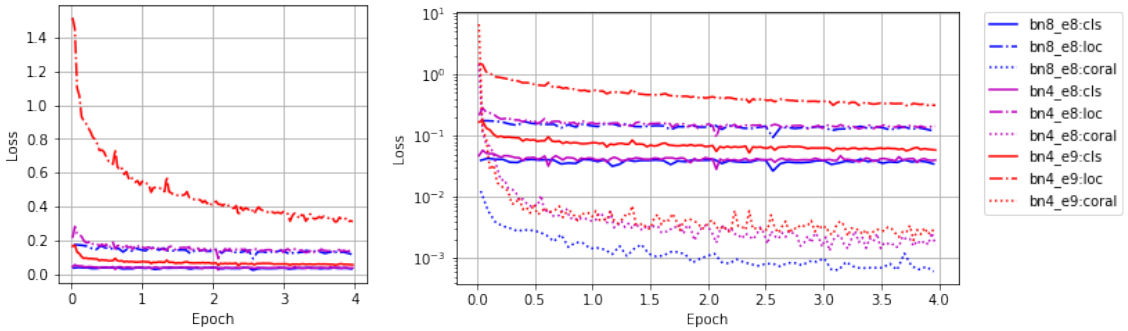


Figure 4: The plot on the left represents the global loss, while plot on the right represent the localization, classification and Coral Loss curves. Different experiments with batch sizes 4 & 8 (bn4 and bn8 respectively), and CORAL loss weighting $\beta_{DA} = 1e8$ & $1e9$ are demonstrated. The loc, cls refer to the localization and classification losses.

4 SOURCES OF S2R DOMAIN GAP

In our experiments with simulator we noticed a few key issues with the way simulated pointclouds were generated, and subsequently causing problems during training. We highlightg these qualitative issues for future improvement of Simulation to Real (S2R) domain transferability.

Synthetic vs Real Pointclouds: We observed a sharp dropout noise in the real world pointclouds, while simulated pointclouds where spatio-temporally consistent. This corresponded to 30% drop in points in the real world pointclouds. This drop in point density has a direct negative effect on the quality recall in the real world. Pointcloud subsampling procedures are a key component to tune the S2R domain gap here. This sampling pattern has been studied separately as a topic by authors in (Yang et al., 2020). Further on, while real world pointclouds had multiple return outputs (range) simulated pointclouds had a single return value. We also have not modeled the presence of dropout noise in the LiDAR pointclouds yet in this study, which reduces the number of points drastically. Finally real world pointclouds undergo motion distortion due to ego-vehicle motion, while simulated pointclouds do not explicitly model this behavior. This might be important for vehicles driving at high speeds.

Table 2: The table demonstrates the gain in performance in 3D detection metrics, with and without the application of DA using the CORAL loss term.

	w/o DA	with DA
Car-bev@0.50	8.82	12.56
Car-bev@0.70	6.06	4.54
Car-aos	36.77	56.76
Pedestrian bev@0.25	0.06	0.03
Pedestrian bev@0.50	0.00	0.00
Pedestrian aos	10.88	12.40

The pointclouds in the real world contain shadowed regions with no points, created due to the presence of the ego-vehicle. While in the simulated pointclouds, the pointclouds contain no ego-vehicle shadow. It is as if the ego-vehicle is transparent and the sensors on the vehicle gather data without any visibility through ego-vehicle taken into account. This is demonstrated via a BEV image over simulated and real vehicle pointclouds in figure 5. Further more objects found partially within the shadow are annotated in simulation while real pointclouds frequently contain very few points on the object. This might lead to over-fitting issues while training on simulation data.

Pointcloud Sparsity: To avoid the sparse generated pointclouds it could be better to have a preprocessing step that removes ground truth (GT) boxes with very low point occupancy. Also, ground truth bounding boxes contain variable number of points. Thus varying the sparsity of the pointclouds in the voxels could help train a better Pointpillar backbone.

Selecting GT Boxes within Sensor FOV and Range: The CARLA simulator provides bounding boxes of all agents in the virtual city irrespective of their visibility to the LiDAR sensor. Points on objects at large range values are no more visible though their corresponding bounding boxes have are still provided by CARLA. The same is applicable with objects in non line positions, eg. behind another vehicle. We manually filter out such detections by thresholding boxes based on the number of points within them.

Simulator Rendering Issues: LiDAR pointclouds are a spatio-temporal stream in the real-world while the simulated pointclouds are received

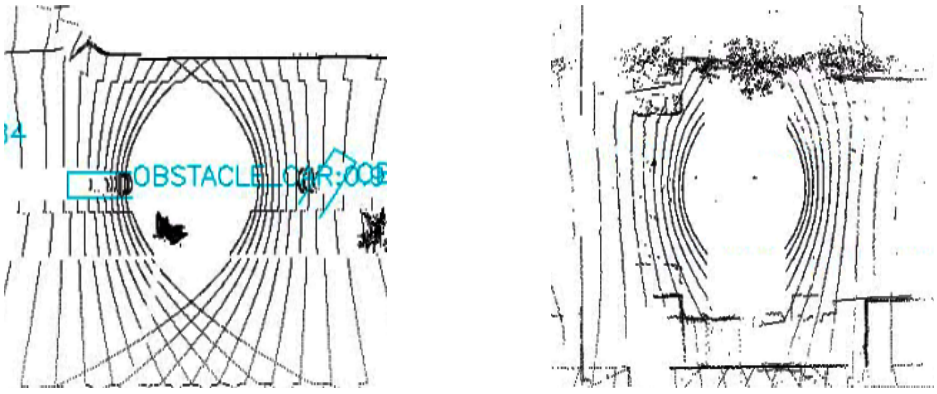


Figure 5: Simulated vs Real vehicle shadow.

as frames. This might lead to some differences in how object bounding boxes might be rendered by the simulator leading to un-correlated shifts between the bounding box and rendered pointclouds. As a result, we observed part of the points that actually belongs to a car (20-30% at maximum) may fall outside its ground truth bounding box. This issue appears in frames mainly when the objects are dynamic with respect to the ego vehicle location.

5 CONCLUSION

The key goal of this study is to evaluate the simulation-to-real transferability of point-cloud-based 3D object detection model performance. We evaluated the performance of the Pointpillars 3D detector on KITTI and nuScenes datasets. This model was further modified to be trained on simulated datasets generated with the CARLA simulator, before predicting on real data collected with vehicles in production. The final goal was to obtain bounding box predictions on the real vehicle pointclouds and alleviate the annotators work with automation of the annotation process.

One of the main down sides in using CORAL loss is the size of the covariance matrices over real and simulated feature maps. Large matrices can not be evaluated and thus we are limited to low resolution feature maps.

In future work we aim to study the usage of domain randomization (Johnson-Roberson et al., 2017) to help reduce the simulation-to-real gap by randomizing parameters of the simulator.

ACKNOWLEDGEMENTS

We would like to thank Barthélemy Picherit & members of simulation team for their support and collaboration during this project. We thank Alexandre Almin from Navya for his comments on the paper. This work is part of the Deep Learning Segmentation (DLS) project financed by ADEME.

REFERENCES

Alonso., I., Riazuelo., L., Montesano., L., and Murillo., A. (2021). Domain adaptation in lidar semantic segmentation by aligning class distributions. In *Proceedings of the 18th International Conference on Informatics in Control, Automation and Robotics - ICINCO,*, pages 330–337. INSTICC, SciTePress.

Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., and Gall, J. (2019). Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9297–9307.

Borna Bešić, Nikhil Gosala, D. C. and Valada, A. (2021). Unsupervised domain adaptation for lidar panoptic segmentation. *arXiv preprint arXiv:2109.15286*.

Brekke, Å., Vatsendvik, F., and Lindseth, F. (2019). Multimodal 3d object detection from simulated pretraining. In *Symposium of the Norwegian AI Society*, pages 102–113. Springer.

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2019). nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*.

DeBortoli, R., Fuxin, L., Kapoor, A., and Hollinger, G. A. (2021). Adversarial training on point clouds for sim-to-real 3d object detection. *IEEE Robotics and Automation Letters*, 6(4):6662–6669.

Deschaud, J.-E., Duque, D., Richa, J. P., Velasco-Forero, S., Marcotegui, B., and Goulette, F. (2021). Paris-carla-

3d: A real and synthetic outdoor point cloud dataset for challenging tasks in 3d mapping. *Remote Sensing*, 13(22).

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16.

Dworak, D., Ciepiela, F., Derbisz, J., Izzat, I., Komorkiewicz, M., and Wójcik, M. (2019). Performance of lidar object detection deep learning architectures based on artificially generated point cloud data from carla simulator. In *2019 24th International Conference on Methods and Models in Automation and Robotics (MMAR)*, pages 600–605. IEEE.

Fang, J., Zhou, D., Yan, F., Zhao, T., Zhang, F., Ma, Y., Wang, L., and Yang, R. (2020). Augmented lidar simulator for autonomous driving. *IEEE Robotics and Automation Letters*, 5(2):1931–1938.

Fong, W. K., Mohan, R., Hurtado, J. V., Zhou, L., Caesar, H., Beijbom, O., and Valada, A. (2021). Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking. *arXiv preprint arXiv:2109.03805*.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237.

Hahner, M., Sakaridis, C., Dai, D., and Van Gool, L. (2021). Fog simulation on real lidar point clouds for 3d object detection in adverse weather. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15283–15292.

Huang, J. and Qiao, C. (2021). Generation for adaption: A gan-based approach for 3d domain adaption with point cloud data. *ArXiv*, abs/2102.07373.

Johnson-Roberson, M., Barto, C., Mehta, R., Sridhar, S. N., Rosaen, K., and Vasudevan, R. (2017). Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 746–753. IEEE.

Kesten, R., Usman, M., Houston, J., Pandya, T., Nadhamuni, K., Ferreira, A., Yuan, M., Low, B., Jain, A., Ondruska, P., Omari, S., Shah, S., Kulkarni, A., Kazakova, A., Tao, C., Platinsky, L., Jiang, W., and Shet, V. (2019). Level 5. Perception Dataset 2020.

Langer, F., Milioto, A., Haag, A., Behley, J., and Stachniss, C. (2020). Domain transfer for semantic segmentation of lidar data using deep neural networks. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8263–8270. IEEE.

Manivasagam, S., Wang, S., Wong, K., Zeng, W., Sazanovich, M., Tan, S., Yang, B., Ma, W.-C., and Urtasun, R. (2020). Lidarsim: Realistic lidar simulation by leveraging the real world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11167–11176.

Meng, Q., Wang, W., Zhou, T., Shen, J., Gool, L. V., and Dai, D. (2020). Weakly supervised 3d object detection from lidar point cloud. In *European Conference on Computer Vision*, pages 515–531. Springer.

Sallab, A. E., Sobh, I., Zahran, M., and Essam, N. (2019). Lidar sensor modeling and data augmentation with gans for autonomous driving. *arXiv preprint arXiv:1905.07290*.

Saltori, C., Lathuiliére, S., Sebe, N., Ricci, E., and Galasso, F. (2020). Sf-uda 3d: Source-free unsupervised domain adaptation for lidar-based 3d object detection. In *2020 International Conference on 3D Vision (3DV)*, pages 771–780. IEEE.

Sun, B., Feng, J., and Saenko, K. (2017). Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*, pages 153–171. Springer.

Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al. (2020). Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454.

Tomasello, P., Sidhu, S., Shen, A., Moskewicz, M. W., Redmon, N., Joshi, G., Phadte, R., Jain, P., and Iandola, F. (2019). Dscnet: Replicating lidar point clouds with deep sensor cloning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0.

Triess, L. T., Dreissig, M., Rist, C. B., and Zöllner, J. M. (2021). A survey on deep domain adaptation for lidar perception. In *Proc. IEEE Intelligent Vehicles Symposium (IV) Workshops*.

Vacek, P., Jašek, O., Zimmermann, K., and Svoboda, T. (2021). Learning to predict lidar intensities. *IEEE Transactions on Intelligent Transportation Systems*.

Wu, B., Zhou, X., Zhao, S., Yue, X., and Keutzer, K. (2019). Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4376–4382. IEEE.

Yang, Z., Sun, Y., Liu, S., and Jia, J. (2020). 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048.

Yue, X., Wu, B., Seshia, S. A., Keutzer, K., and Sangiovanni-Vincentelli, A. L. (2018). A lidar point cloud generator: from a virtual world to autonomous driving. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, pages 458–464.

Zhao, S., Wang, Y., Li, B., Wu, B., Gao, Y., Xu, P., Darrell, T., and Keutzer, K. (2020). epointda: An end-to-end simulation-to-real domain adaptation framework for lidar point cloud segmentation. *arXiv preprint arXiv:2009.03456*, 2.