2024 IEEE Intelligent Vehicles Symposium (IV)
June 2-5, 2024. Jeju Island, Korea

SimBusters: Bridging Simulation Gaps in Intelligent Vehicles Perception

Alberto Justo[1], Javier Araluce[1], Javier Romera[2], Mario Rodriguez-Arozamena[1], Leonardo González[1] and Sergio Díaz[1]

Abstract—**Recent advances in automated vehicle technology rely heavily on simulated environments for training and testing. However, a significant challenge lies in bridging the gap between simulated and real-world scenarios, as discrepancies between these environments can affect the performance and reliability after that transition, especially in perception. Particularly, LiDAR sensors are highly affected in this matter due to disparities in pointcloud distribution and intensity. Therefore, this paper presents an innovative approach to bridge the gap between simulation and reality. For it, we test and validate a realistic LiDAR library, PCSim, within the CARLA simulator, providing an enhanced simulation environment. Our method involves integrating perception models, pre-trained on real-world datasets, in this environment. Then, we develop a Real2Sim domain adaptation method to transfer these models into the library, leveraging their performance. Finally, we evaluate the 3D object detection models in PCSim LiDARs to prove our methodology.**

We have assessed this proposal in PCSim, obtaining promising results in mitigating the simulation-reality gap. Our evaluations provide a guidance for future effective transition from virtual environments to real-world applications.

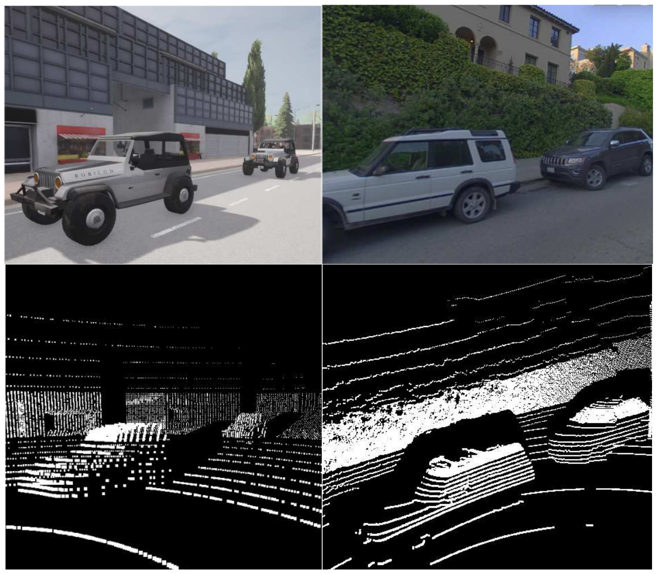


Fig. 1: Composite visualization of vehicles from synthetic and real-world data. On the left, top image shows two vehicles captured by a camera in the CARLA simulator. Bottom image displays the corresponding LiDAR point cloud generated by Hesai Pandar64 within PCSim [10] library. On the right, top and bottom images represent analogous views from a real-world dataset, Pandaset [12], using the same LiDAR device.

I. INTRODUCTION

Currently, LiDARs have become increasingly important in perception for autonomous driving, due to their ability to accurately represent their surroundings in three dimensions. Thus the information they provide is helpful for 3D object detection [1], [2], tracking [3], [4], and semantic segmentation [5], [6] purposes, amongst many others. In this field, virtual environments are highly desirable, as they provide unlimited data capturing and easiness for labeling. They provide better scalability, since there is no need of real vehicle for this purpose. Nevertheless one of the main problems encountered in 3D object detection is the transition from simulation to reality, because of the disparity of data characteristics between these two [7].

In real-world scenarios, LiDAR sensors capture complex data, including the distribution and intensity of the points that make up the 3D representation of the environment. These attributes rely on various factors like the objects' distance, material properties, and ambient conditions. It is worth noting that these attributes make each LiDAR brand model different, despite the current trend towards data homogenization. Simulated LiDAR data, however, often lacks this level of detailed representation [8]. It may not accurately capture the disparity of point distribution nor the intensity variations seen in real-world data. Therefore, bridging this 'reality gap' requires improving the overall realism of simulated environments and necessitates a focus on the detailed modeling of LiDAR sensor behavior.

Currently, mitigating this breach presents a major challenge. This challenge is the replication of real-world sensor variability and complexity in simulated environments, like CARLA [9]. One brand-new solution for implementing realistic LiDARs into CARLA is PCSim [10]. This library can simulate various widely-used LiDAR devices, each with distinct features, such as beam properties and motion distortion [11]. PCSim, illustrated in Figure 1, enhances the default CARLA LiDAR configuration, and will be used throughout the work presented in this paper.

Moreover, another integral aspect of this bridge is the concept of Domain Adaptation (DA) [7], [13]. This technique adapts a model trained in one domain (the source) into another, different domain (the target). In the context of LiDAR simulation, domain adaptation takes a capital role. Since there are many different approaches applicable to LiDAR point clouds [14], [15], we build a Real2Sim method, which transforms LiDAR intensity into a normalized form, used in simulated environments.

[1]Alberto Justo, Javier Araluce, Mario Rodriguez-Arozamena, Leonardo González and Sergio Díaz are with TECNALIA, Basque Research and Technology Alliance (BRTA), 48160 Derio, Bizkaia, Spain alberto.justo, javier.araluce, mario.rodriguez, leonardo.gonzalez, sergio.diaz{@tecnalia.com}

[2]Javier Romera is with Dept. of Computer Systems Engineering, Mondragon University, 48001 Bilbao, Bizkaia, Spain javier.romera@alumni.mondragon.edu

Main contributions given by this paper are the following:

- We present a novel assessment framework that evaluates how much PCSim mimics real LiDARs by SOTA metrics.
- We build a Real2Sim domain adaptation procedure, to transfer real-world 3D object detection models in CARLA through PCSim.
- In order to make fair ground truth measurements for our validations, we have developed a Ground Truth Pointcloud Clustering method.

II. RELATED WORKS

A. Simulation vs Reality

The exploration of 3D object detection methodologies has seen a significant divergence in the use of simulated and real-world datasets. Synthetic datasets, like SHIFT [16], allow specific scenario fitting and easy data collection and annotation. In contrast, real-world datasets, such as NuScenes [17], offer a richer variety of natural scenarios, holding labor-intensive data collection and annotation. Although simulated data is quite accurate overall, it may still have differences from real-world data due to factors such as sensor noise, intensity, point cloud distributions, environmental conditions, and the physical properties of objects.

Regarding these issues, several studies [18], [19] have shown that LiDAR-based 3D object detection models trained solely on simulated data often underperform when tested on real-world scenes. Therefore, solutions for making LiDAR simulations better resemble reality have emerged over the years. In one hand, data-driven generative models synthesize LiDAR data based on a given scene representation [20]. Although these methods are realistic and computationally efficient, they tend to not adapt well to new environments or be easily configurable. On the other hand, hybrid methods [21], [22] combine different parts of data-driven generative models and simulation environments. They constitute digital twins with CAD models and real-world sensor data, but find the same generalization problems.

Within this frame of reference, even though new hybrid solutions are taking more importance lately, automated driving simulators are still a common ground for many researchers. From our point of view, we believe that the implementation of realistic LiDAR libraries shows a cornerstone in advancing the fidelity of simulated environments in this field. That is the reason why we choose to evaluate PCSim to prove this statement in 3D object detection. To the best of our knowledge, this library has not been evaluated the way we do in our research before.

B. Domain Adaptation

Currently, there are still many challenges for applying models across different domains, sensor types, and from simulated to real-world environments.

Domain-to-domain and sensor-to-sensor adaptations [23] primarily involve techniques like transfer learning and convolutional-network-based feature extraction. These strategies focus on using the knowledge gained in one area and applying it successfully in another.

In simulation-to-reality (Sim2Real) [24] adaptation, Generative Adversarial Networks (GANs) are frequently utilized to transform synthetic LiDAR data into a form that recreates reality more closely.

However, all these methods previously mentioned rely on deep learning methods. They are seen as 'black boxes', making it difficult to understand how they handle domain adaptations or to diagnose why they fail. Their effectiveness is dependable on having large, diverse datasets and significant computational resources. Handling domain adaptations through simpler algorithms, like normalization, gives a more practical solution and less intensive in computation.

III. METHODOLOGY

Simulated environments, when compared to actual LiDAR readings, have two major drawbacks: disparity in point cloud distribution and intensity [7].

In terms of LiDAR simulation, point clouds are obtained through ray tracing [25]. Ray intersections are determined by projecting rays from the original sensor position outward to the surface of the environmental geometry, as shown mathematically in Equation 1.

$$p_i = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = r_i \begin{pmatrix} \cos(\phi)\cos(\theta) \\ \cos(\phi)\sin(\theta) \\ \sin(\phi) \end{pmatrix} \tag{1}$$

Here we can see the orientation of the laser, defined by azimuth angle θ and polar angle ϕ. Distance r_i is obtained from ray casting [25]. These variables are used to compute the coordinates of point p_i, translated to Cartesian coordinates. In CARLA, these parameters are idealized, resulting in more uniform distributions and consistent measurements. A Gaussian noise can be added also in these calculations to make them closer to reality. PCSim incorporates r_i, θ and ϕ from the channel distribution specifications of each real LiDAR device.

Intensity of reflected signals in LiDAR provides critical information about the surface properties of detected objects. Generally, a lower or higher intensity implies the measurement estimation will be less or more reliable. Intensity is calculated using the Beer-Lambert Law [26], shown in Equation 2.

$$\frac{I}{I_0} = e^{-a \cdot d} \tag{2}$$

In this equation, I represents the intensity of light after it has passed through the material. I_0 is the initial intensity of the light before entering the material. The exponent $-a \cdot d$ corresponds to the product of the absorption coefficient of the medium (a) and the thickness of the medium (d) where light passes through.

CARLA encodes intensity information using the RGB color values in a virtual image. It is a simplified model

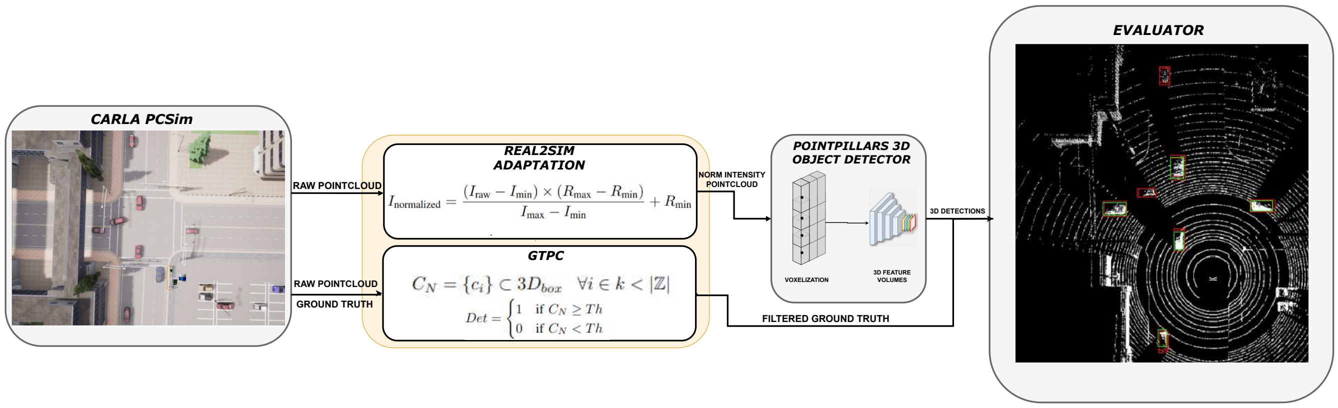

CARLA PCSim
RAW POINTCLOUD
RAW POINTCLOUD
GROUND TRUTH
REAL2SIM ADAPTATION
$I_{\text{normalized}} = \frac{(I_{\text{raw}} - I_{\text{min}}) \times (R_{\text{max}} - R_{\text{min}})}{I_{\text{max}} - I_{\text{min}}} + R_{\text{min}}$
GTPC
$C_N = \{c_i\} \subset 3D_{box} \quad \forall i \in k < |\mathbb{Z}|$
$Det = \begin{cases} 1 & \text{if } C_N \geq Th \\ 0 & \text{if } C_N < Th \end{cases}$
NORM INTENSITY POINTCLOUD
POINTPILLARS 3D OBJECT DETECTOR
VOXELIZATION
3D FEATURE VOLUMES
3D DETECTIONS
FILTERED GROUND TRUTH
EVALUATOR

Fig. 2: SimBusters. Our assessment framework for PCSim library

TABLE I: Datasets characteristics for training PointPillars

Dataset	LiDAR	Channels	vFOV(º)	Range(m)	Frames	BBoxes
KITTI [29]	Velodyne HDL-64	64	27	120	7.4k	80k
NuScenes [17]	Velodyne HDL-32	32	41	100	390k	1.4M
Pandaset [12]	Hesai Pandar64	64	40	200	16k	-

compared to the complexity of LIDAR systems in the real world. When detecting an object with a LiDAR, there is an additional chance ('drop-off') that the point will be disregarded. This chance is based on the intensity of the detected point. Regarding this matter, PCSim uses CARLA's intensity, but provides different drop-off intensity parameters for each sensor model.

A. 3D Object Detection

From all open-source LiDAR-based 3D object detection models, we have chosen PointPillars [27] , due to its good overall performance, low training time and fast runtime (16.2 ms on a GTX 1080Ti [28]). As shown in Table I, we have selected three different real-datasets for training these models: KITTI, NuScenes and Pandaset. They encompass large amounts of data, collected under various environmental conditions, giving a comprehensive training ground. Moreover, they offer standarized benchmarks. In particular, KITTI and Pandaset provide intensity values alongside spatial coordinates. NuScenes does not give intensity information, still contributes with more varied environmental and lighting conditions.

B. Real2Sim Domain Adaptation

A normalization process is necessary to enable a consistent interpretation of intensity data across different LiDAR models. Deep learning methods [13] require extensive data and computational resources for training models to learn sensor-specific patterns. However, a mathematical approach to normalization gives a simpler alternative. This procedure is shown in Equation 3.

$$I_{\text{normalized}} = \frac{(I_{\text{raw}} - I_{\text{min}}) \times (R_{\text{max}} - R_{\text{min}})}{I_{\text{max}} - I_{\text{min}}} + R_{\text{min}} \tag{3}$$

Here, I_{raw} represents the raw intensity value detected by LiDAR sensor. The terms I_{min} and I_{max} correspond to the minimum and maximum detectable intensity values for the sensor. R_{min} and R_{max} denote the normalized range, typically 0 and 255 to align with an 8-bit scale. Our normalization not only makes intensity values from different LiDAR sensors comparable, but also aligns them with standard image intensity ranges. This normalization technique aims to transform the sensor-specific intensity values into a standardized scale. Therefore, we enable direct comparability across different LiDAR sensors and real-to-simulated domains.

C. Ground Truth Pointcloud Clustering (GTPC)

In many situations, LiDAR-based 3D object detection in automated vehicles is conditioned by factors like sensor range or measurement occlusions in particular situations, affecting the detection's reliability. Due to these factors, comparing all ground truth measurements against the current frame's 3D object detections makes little sense. This means there is a need of a fairer method for comparing 3D object detections to ground truth measurements that count on occlusions and sensor range. Common methods like Euclidean Distance are highly extended to rule out ground truth measurements. However, they only rely on threshold distances in the surroundings of the ego-vehicle, but don't consider situations where the ego-vehicle cannot detect nearby agents due to obstructions. Therefore, we choose to develop a Ground Truth Pointcloud Clustering (GTPC) filter. This approach quantifies the point cloud data by counting the points enclosed within a pre-annotated bounding box. Thus, Equation 4 determines the set of points (C_N) in the point cloud, consisting of k samples (c_i), that are within the bounding box ($3D_{BBox}$).

$$C_N = \{c_i\} \subset 3D_{BBox} \quad \forall i \in k < |\mathbb{Z}| \tag{4}$$

$$Det = \begin{cases} 1 & \text{if } C_N \geq Th \\ 0 & \text{if } C_N < Th \end{cases} \tag{5}$$

In Equation 5, if the number of points C_N inside each bounding box are equal or higher than a certain threshold Th (in our case, 10 points as done in [17]) then we take that as ground truth measurement. Otherwise, we rule out this position. GTPC provides an intuitive method for filtering the actual agent positions near the ego-vehicle, as it considers occlusions.

IV. EXPERIMENTS

A. Implementation Details

In our experimental setup, we recorded a sample dataset comprising 5.000 frames with an average of 20 cars per frame. This data was sourced from the default environments of Town04 and Town05 from CARLA. For the detection phase, we employed the open-source platform OpenPCDet [28], tailored for 3D object detection from point cloud data. The computational backbone of our experiments feautures: Nvidia GeForce RTX 2080 SUPER GPU and an AMD® Ryzen 9 3950x 16-core processor × 32 GB of RAM.

B. Metrics

Intersection over Union (IoU): Intersection over Union (IoU) is a metric used to measure the accuracy of a 3D object detector on a particular dataset [2]. Equation 6 displays IoU as the ratio of the area of overlap between the predicted bounding box and the ground truth bounding box to the area of union of these two boxes.

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}} \tag{6}$$

Mean Average Precision (mAP): It is the most popular metric used to evaluate the performance of a detector across different classes and IoU thresholds, and many other metrics are redefinitions of this one [2]. The Average Precision (AP), referred in Equation 7, for a single class is calculated as the average of the precision values at the points where recall changes. $P(R)$ represents the precision-recall curve and AP is the area under the curve. The mAP, illustrated in Equation 8, is then the mean of these AP values.

$$\text{AP} = \int_0^1 P(R)\, dR \tag{7}$$

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i \tag{8}$$

V. RESULTS

This section shows the final results after conducting our experiment. We evaluated PointPillars model, pre-trained in real-world datasets, using 4 different sensor models with PC-SIM + DA technique. We incorporated our Real2Sim domain adaptation, to see how detector's performance behaves after intensity normalization. In this context, we specify detection metrics only for "Car" class. Table II shows PointPillars performance in real datasets, which states the context for our performance metrics in Table III, which we compared against.

To make a more equitable comparison between these metrics, we decided to take mAP within the highest IoU threshold (0.7). So, our overall detections are more strict, hence mAP values are lower. We are comparing our experiment against entire real datasets which consider more environments and situations, so we demand more from our detections. Table IV shows that the implemented Real2Sim states an upgrade in both KITTI and Pandaset pre-trained PointPillars. Improvements are between 2.81 % and 10.26 % for mAP and between 0 % and 1.42 % for IoU. However, NuScenes pre-trained model underperformed, regardless of applying Real2Sim. Thus we can infere two statements. First, models that use intensity (unlike NuScenes) perform better after normalization. Second, pointcloud distribution affects the evaluation. We can see this in NuScenes metrics comparison between HDL-32 and CARLA-32 LiDARs. They differ when it comes to object detection performance, as their pointcloud distribution is not the same.

In essence, these results not only validate the reliability of PCSim library, but also our Real2Sim method. Although improvements showed are promising, we have to consider the scope of our experiment, in comparison to the datasets stated in the SOTA. There are still some improvements in terms of PCSim intensity simulation to be done.

Moreover, Figure 3 illustrates the representation of the same scene, using the four LiDAR devices we have evaluated. After applying Real2Sim, we can appreciate the improvements mentioned before. Figures 3(a) and 3(b) illustrate the detections vs ground truth positions in Velodyne HDL-64, using pre-trained PointPillars in KITTI dataset. Figures 3(c) and 3(d) show the same comparison, but in Pandar64, using the same pre-trained model in Pandaset. However, our Real2Sim method applied at NuScenes, shown in Figures 3(e) to 3(h), does not improve as models trained in this dataset do not count on intensity.

VI. CONCLUSIONS AND FUTURE WORKS

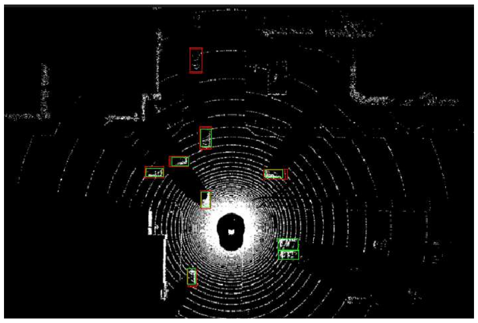
This paper presents our assessment framework for mitigating the gap between simulation and reality in intelligent vehicles perception. For this purpose, we have used a

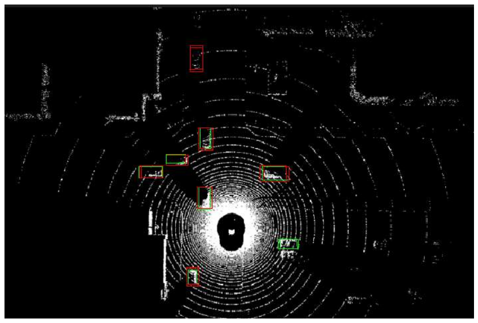
TABLE II: "Car" Baseline metrics of PointPillars in real datasets

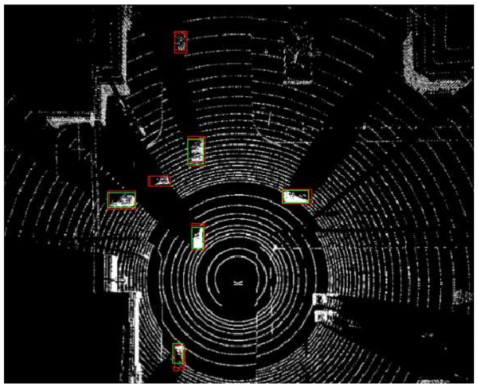
Dataset	IoU	mAP@0.7
KITTI	0.7	0.39
NuScenes	0.84	0.68
Pandaset	0.88	0.71

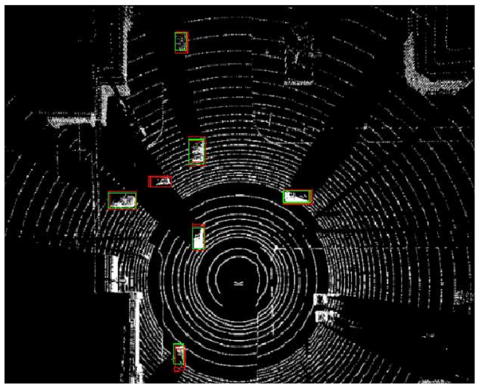
TABLE III: Comparison of procedures in PCSim. We show in which dataset PointPillars was trained, PCSim LiDAR where is tested, domain adaptation (DA) and the performance metrics. The "-" implies that no method was applied.

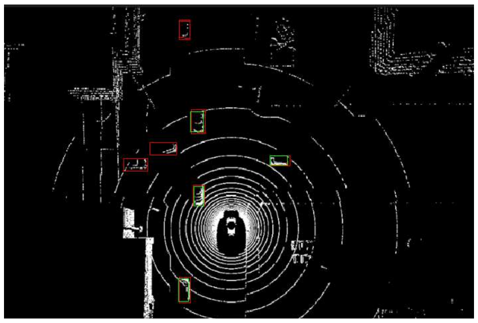
Dataset	PCSim LiDAR	DA	IoU	mAP@0.7
KITTI	HDL-64	-	0.67	0.34
	HDL-64	Real2Sim	**0.71**	**0.43**
NuScenes	HDL-32	-	0.77	0.59
	HDL-32	Real2Sim	0.77	0.59
	CARLA-32	-	0.76	0.54
	CARLA-32	Real2Sim	0.76	0.54
Pandaset	Pandar64	-	0.86	0.65
	Pandar64	Real2Sim	**0.88**	**0.73**

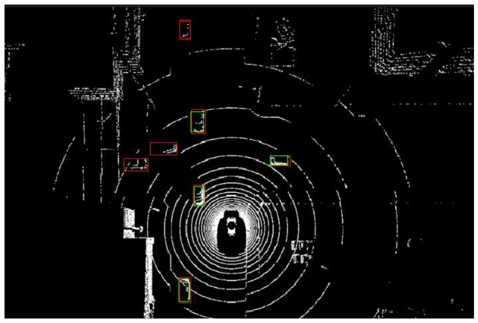

(a) Detections vs Ground Truth in PCSim Velodyne HDL-64, without DA

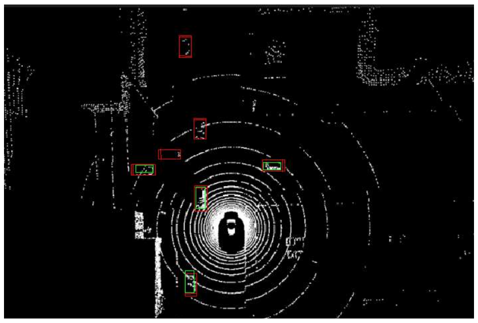

(b) Detections vs Ground Truth in PCSim Velodyne HDL-64, with Real2Sim


(c) Detections vs Ground Truth in PCSim Pandar64, without DA


(d) Detections vs Ground Truth in PCSim Pandar64. with Real2Sim


(e) Detections vs Ground Truth in PCSim Velodyne HDL-32, without DA


(f) Detections vs Ground Truth in PCSim Velodyne HDL-32, with Real2Sim


(g) Detections vs Ground Truth in CARLA 32-beams LiDAR, without DA

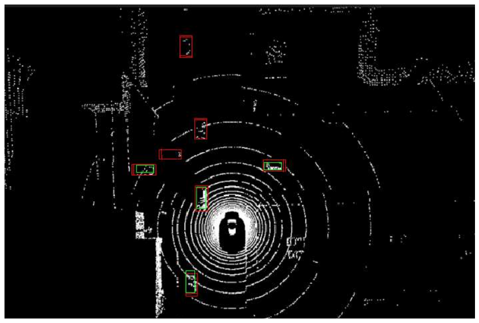

(h) Detections vs Ground Truth in CARLA 32-beams LiDAR, with Real2Sim

Fig. 3: Qualitative results showing our assesment framework. We represent: the ground truth measurements against the detections in the current frame. Ground truth is represented after applying GTPC. We also compare the detections with and without DA.

TABLE IV: Percentages in performance metrics between our experiment in PCSim and real datasets. Decreases and increases are represented here.

Dataset	PCSim LiDAR	DA	IoU	mAP@0.7
KITTI	HDL-64	-	↓4.29%	↓12.34%
	HDL-64	Real2Sim	↑**1.42%**	↑**10.26%**
NuScenes	HDL-32	-	↓8.33%	↓13.23%
	HDL-32	Real2Sim	↓8.33%	↓13.23%
	CARLA-32	-	↓9.52%	↓20.58%
	CARLA-32	Real2Sim	↓9.52%	↓20.58%
Pandaset	Pandar64	-	↓2.27%	↓8.45%
	Pandar64	Real2Sim	**0.0%**	↑**2.81%**

SOTA model, PointPillars, pre-trained in real datasets, from a vehicle 3D object detection, and evaluated it on an enhanced CARLA LiDAR library (PCSim [10]). Moreover, we have implemented a ground truth filter in CARLA default's agent positions, Ground Truth Pointcloud Clustering, to make comparisons with PointPillars detections. On top of that, we have developed a novel Real2Sim domain adaptation method, to increase our 3D object detection performances in this library. Our results, based on SOTA metrics, highlight the importance of considering the intensity and pointcloud distribution in LiDAR-based 3D detectors. Furthermore, our Real2Sim method enables the normalization of intensities to a common format, tailored for sensor-to-sensor and real-to-simulated DA.

To sum up, our assessment in PCSim shows that this library proves to be a promising solution for simulating realistic LiDARs in CARLA. We aim to work on further validations in this field, comparing more 3D object detection models, real-world datasets and sensors to this library. Moreover, we think that PCSim has potential to be a starting point for transitioning models trained in simulated data to real-world situations.

ACKNOWLEDGMENT

This research has been conducted as part of the EVENTS project, which is funded by the European Union, under grant agreement No 101069614 and within the MEDUSA program under the grant number CER-2023101, Red de Excelencia CERVERA which was founded by the MICIN thorught CDTI under the MRR of the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Commission. Neither the European Union nor the granting authority can be held responsible for them.

REFERENCES

[1] W. Zimmer, E. Ercelik, X. Zhou, X. J. D. Ortiz, and A. Knoll, "A survey of robust 3d object detection methods in point clouds," 2022.
[2] S. Y. Alaba and J. E. Ball, "A survey on deep-learning-based lidar 3d object detection for autonomous driving," *Sensors*, vol. 22, no. 24, 2022.
[3] M. Sualeh and G.-W. Kim, "Dynamic multi-lidar based multiple object detection and tracking," *Sensors (Basel, Switzerland)*, vol. 19, 2019.
[4] Y. Park, L. M. Dang, S. Lee, D. Han, and H. Moon, "Multiple object tracking in deep learning approaches: A survey," *Electronics*, vol. 10, no. 19, 2021.
[5] A. Xiao, J. Huang, D. Guan, F. Zhan, and S. Lu, "Transfer learning from synthetic to real lidar point cloud for semantic segmentation," 2021.
[6] R. Zhang, Y. Wu, W. Jin, and X. Meng, "Deep-learning-based point cloud semantic segmentation: A survey," *Electronics*, vol. 12, no. 17, 2023.
[7] A. Farahani, S. Voghoei, K. Rasheed, and H. R. Arabnia, "A brief review of domain adaptation," in *Advances in Data Science and Information Engineering*, (Cham), Springer International Publishing, 2021.
[8] S. Manivasagam, I. A. Bârsan, J. Wang, Z. Yang, and R. Urtasun, "Towards zero domain gap: A comprehensive study of realistic lidar simulation for autonomy testing," in *ICCV*, 2023.
[9] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," 2017.
[10] X. Cai, W. Jiang, R. Xu, W. Zhao, J. Ma, S. Liu, and Y. Li, "Analyzing infrastructure lidar placement with realistic lidar simulation library," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2023.
[11] W. Jiang, H. Xiang, X. Cai, R. Xu, J. Ma, Y. Li, G. H. Lee, and S. Liu, "Optimizing the placement of roadside lidars for autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18381–18390, 2023.
[12] P. Xiao, Z. Shao, S. Hao, Z. Zhang, X. Chai, J. Jiao, Z. Li, J. Wu, K. Sun, K. Jiang, W. Yunlong, and D. Yang, "Pandaset: Advanced sensor suite dataset for autonomous driving," pp. 3095–3101, 09 2021.
[13] J. Richter, F. Faion, D. Feng, P. B. Becker, P. Sielecki, and C. Glaeser, "Understanding the domain gap in lidar object detection networks," 2022.
[14] L. T. Triess, M. Dreissig, C. B. Rist, and J. Marius Zollner, "A survey on deep domain adaptation for lidar perception," in *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*, IEEE, July 2021.
[15] T. Shan, J. Wang, F. Chen, P. Szenher, and B. Englot, "Simulation-based lidar super-resolution for ground vehicles," 2020.
[16] T. Sun, M. Segu, J. Postels, Y. Wang, L. Van Gool, B. Schiele, F. Tombari, and F. Yu, "SHIFT: a synthetic driving dataset for continuous multi-task domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21371–21382, June 2022.
[17] 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.
[18] G. Gusmão, C. Barbosa, and A. Raposo, "Development and validation of lidar sensor simulators based on parallel raycasting," *Sensors*, vol. 20, p. 7186, 12 2020.
[19] S. Huch, L. Scalerandi, E. Rivera, and M. Lienkamp, "Quantifying the lidar sim-to-real domain shift: A detailed investigation using object detectors and analyzing point clouds at target-level," 03 2023.
[20] V. Zyrianov, X. Zhu, and S. Wang, "Learning to generate realistic lidar point cloud," in *ECCV*, 2022.
[21] 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," 2020.
[22] B. Guillard, S. Vemprala, J. K. Gupta, O. Miksik, V. Vineet, P. Fua, and A. Kapoor, "Learning to simulate realistic lidars," 2022.
[23] X. Liu, C. Yoo, F. Xing, H. Oh, G. E. Fakhri, J.-W. Kang, and J. Woo, "Deep unsupervised domain adaptation: A review of recent advances and perspectives," 2022.
[24] J. Truong, S. Chernova, and D. Batra, "Bi-directional domain adaptation for sim2real transfer of embodied navigation agents," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2634–2641, 2021.
[25] S. Royo and M. Ballesta-Garcia, "An overview of lidar imaging systems for autonomous vehicles," *Applied Sciences*, vol. 9, no. 19, 2019.
[26] D. F. Swinehart, "The beer-lambert law," *Journal of Chemical Education*, vol. 39, no. 7, p. 333, 1962.
[27] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," 2019.
[28] O. D. Team, "Openpcdet: An open-source toolbox for 3d object detection from point clouds." https://github.com/open-mmlab/OpenPCDet, 2020.
[29] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012.