Verification of LiDAR-Based Detection Models in the Context of Autonomous Racing

1st Utkarsh Chirimar
Dept. of Computer Science
University of Virginia
Charlottesville, VA, USA
uc6gq@virginia.edu

2nd John Chrosniak
Dept. of Computer Science
University of Virginia
Charlottesville, VA, USA
jlc9wr@virginia.edu

***Abstract*—Autonomous driving frameworks require intense machine learning algorithms to be successful in detecting obstacles within the car's environment. These models are trained on thousands of examples, whether it is for data coming from the LiDAR or camera. However, we can never be fully confident in the ability of these models 100% of the time. Specifically, LiDAR data is prone to perturbations due to multiple reasons and these models are more likely to fail at making accurate predictions. Therefore, we need verification processes that can help properly evaluate the robustness of machine learning models employed to sensor data. This paper focuses on mimicking different degrees of perturbations on LiDAR data from autonomous driving scenarios on everyday roads and race tracks. We aim to understand at what points popular machine learning models tend to fail and fif machine learning models for one scenario outperforms the other with more perturbations.**

I. Introduction

Autonomous driving is a growing research field with many complex problems to be solved. Most autonomous driving frameworks are tackling the problem through three modules: perception, planning, and control [1]. Perception enables autonomous vehicles to understand the driving environment by localizing the ego vehicle and sensing surrounding obstacles' locations. Planning uses the information provided by the perception module to make decide the next actions to take towards a specified goal, usually to avoid obstacles. Finally, the control module needs to safely execute the planned actions by providing the inputs to the hardware. More specifically, within the perception module, self-driving vehicles not only need to estimate the position and velocities of obstacles in the environment but also need to predict the trajectory of these obstacles. This problem space is known as obstacle detection and tracking (ODT), for which many sensors, including LiDAR, camera, and RADAR, are needed. Each sensor has its own advantage and disadvantage, so the redundancy in sensors is key to minimizing error [2].

In order to solve ODT problems, supervised machine learning (ML) techniques are usually employed on the sensor data to understand the surrounding environment. For example, well-trained models have been trained on thousands of examples coming from various available datasets, such as KITTI [3] and NuScenes [4], and achieve high accuracy when detecting obstacles. The reason why these ML models are successful is because of the quality and quantity of annotated data available for driving scenarios from the datasets mentioned above and more. However, if there are not enough annotations available, solving complex problems, such as ODT in autonomous driving, becomes more difficult. There are other ML methods to training models without annotations, known as unsupervised learning, but these models tend to be less accurate for problems as complex as ODT.

LiDARs are a powerful sensor used heavily to solve ODT problems, including autonomous cars, in order to detect obstacles in the environment. The LiDAR is able to render an accurate geometric representation of the environment, called point clouds. Most algorithms for identifying objects on the road, such as cars, pedestrians, and bicycles, utilize ML. Point clouds are sent through ML models that, then, tend to output bounding boxes of certain objects

with their specific classification. OpenPCDet is a project that provides multiple kinds of open-source ML models for LiDAR based detections [5].

Problems arise when the point clouds are not the most perfect, and the ML models are not able to make accurate predictions. Point clouds are vulnerable to distortions, which can be caused for many reasons: the vehicle's motion, reflecting materials in the environment, spoofed objects from attackers, and more [6]. Since perturbations in LiDAR points can be common, we want to understand how ML models react to variabilities in point clouds. We aim to create certain random (simple and complex) LiDAR perturbations and send them through the OpenPCDet models to evaluate each model's robustness.

II. Related Work

There are many ways to attack LiDAR data to trick autonomous vehicles. In [7], the authors attack LiDAR points through a technique known as spoofing. They use a photo-diode, kept near the LiDAR, that receives the LiDAR lasers and outputs lasers to simulate real echo pulses but with a delay. This causes the car to perceive an object in the spoofed LiDAR even though there isn't one.

In [8], the paper does a similar experiment to our proposed plan. They have implemented adversarial attacks on three different ML models: PointRCNN 3D detector, PIXOR 3D, and PV-RCNN. All three of these models are from the OpenPCDet project and the paper tested the adversarial attacks on the KITTI dataset. They were able to conclude these popular LiDAR detection models were vulnerable to the adversarial attacks they implemented.

We propose an evaluation similar to what was performed in [8]. However, we defer from their methods in two ways: (1) we will also evaluate a model that was developed in an autonomous racecar environment, where speeds are much higher and LiDAR distortion can be a bigger problem, and (2) we will evaluate different models than the ones the paper evaluated using different perturbation methods.

III. Experimental Setup

For our experiments, we trained PointPillars [9] and VoxelRCNN [10] object detection models on both the KITTI Vision Dataset [3] and RACECAR Dataset [11], resulting in four overall models. The KITTI dataset is a popular self-driving dataset with professionally annotated 3D bounding boxes of pedestrians, cyclists, and vehicles across a variety of urban driving scenarios. In contrast, the RACECAR dataset is a recently released dataset containing scenes from multi-vehicle laps in the Indy Autonomous Challenge [12]. While this dataset does not have professional annotations, reasonably accurate annotations can be derived using the pose information of both vehicles on the track. Figure 1 shows examples of raw data from both of these datasets.

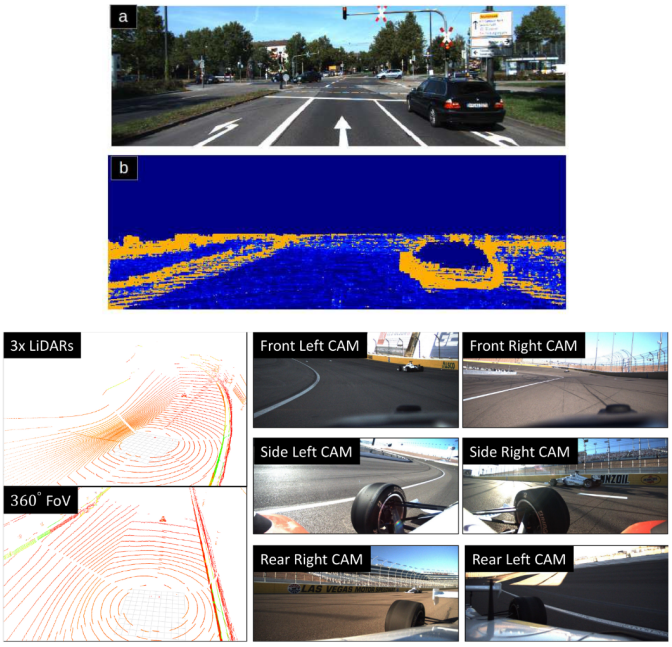

Fig. 1. Raw data from the KITTI (top) and RACECAR datasets (bottom).

For perturbation testing, two datasets were created with 100 samples from the RACECAR and KITTI datasets, each with three varying levels of perturbations. The first dataset put more emphasis on injecting Gaussian noise into the point clouds, with the three levels containing noise levels of 0.05, 0.1, and 0.15 meters. This dataset also had minor dropout levels of 5%, 10%, and 15% of points being randomly discarded. Figure 2 shows these perturbations applied to a scene in the KITTI dataset. The second dataset put more emphasis on random dropout with 10%, 20%, and 30% of points being

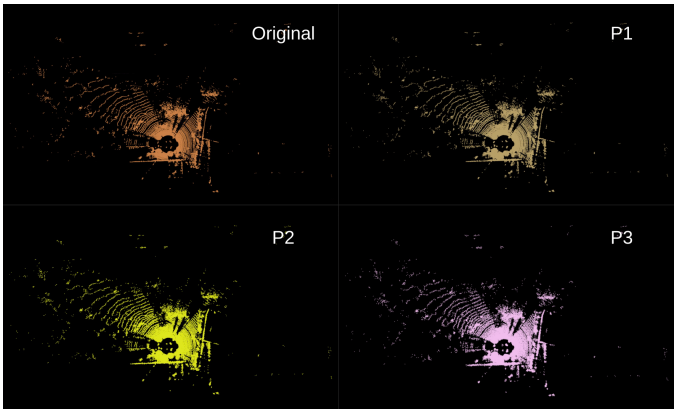

Fig. 2. Perturbations applied to the KITTI dataset with high levels of Gaussian noise and lower levels of dropout. The noise and dropout increase from P1 to P3.

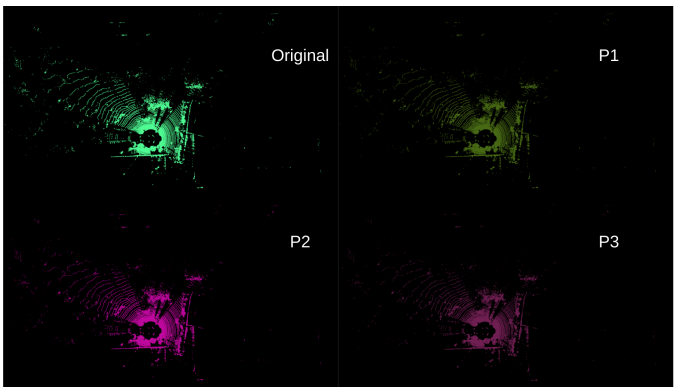

Fig. 3. Perturbations applied to the KITTI dataset with high levels of dropout and lower levels of Gaussian noise. The noise and dropout increase from P1 to P3.

randomly discarded. Small amounts of Gaussian noise were also injected at levels of 0.01, 0.02, and 0.03 meters. Figure 3 shows these perturbations applied to the same scene in the KITTI dataset.

IV. Experimental Results

The trained models were evaluated using the OpenPCDet evaluation script on the perturbation datasets. The evaluation metrics include recall with 50% or more Intersection over Union (0.5 IoU) and the average number of predictions. The first metric indicates the model's success in detecting objects while the second indicates the noisiness of the model. The results for the performance on the dataset with larger amounts of Gaussian noise are displayed in Table I. The PointPillars and VoxelRCNN models displayed different behavior when subjected to perturbations in the KITTI dataset. The

Model	Dataset	Unperturbed	Level 1	Level 2	Level 3
PointPillars	KITTI	0.923288, 31.650	0.583562, 37.630	0.550685, 45.610	0.476712, 52.290
VoxelRCNN	KITTI	0.966887, 6.260	0.599338, 6.650	0.437086, 5.040	0.135762, 3.920
PointPillars	RACECAR	0.950000, 1.620	0.930000, 2.350	0.900000, 9.480	0.840000, 22.400
VoxelRCNN	RACECAR	0.980000, 1.040	0.960000, 1.220	0.970000, 2.870	0.920000, 4.370

TABLE I
EVALUATION RESULTS ON THE DATASET WITH LARGER GAUSSIAN NOISE. TOP NUMBERS INDICATE THE PERCENTAGE OF OBJECTS DETECTED WITH 0.5 IOU AND THE BOTTOM NUMBER INDICATES THE AVERAGE NUMBER OF DETECTIONS PER SCENE.

PointPillars model was able to detect objects at a higher rate than VoxelRCNN at higher perturbation levels, but this came at the cost of misclassifying many empty areas as vehicles, pedestrians, or cyclists. While the detection capabilities of VoxelRCNN decreased more than the PointPillars model, this behavior should be considered more favorable because the high levels of noise caused vehicles to no longer look like vehicles. This behavior differed with the models trained on the RACECAR dataset, as both models saw decreases in accuracy and increases in the average number of detections. VoxelRCNN still saw less performance degradation compared to PointPillars.

The results for the performance on the dataset with larger amounts of dropout are displayed in Table II. The models trained on the KITTI dataset both exhibited the same large drop in performance for the smallest level of dropout that mostly plateaued for the remaining perturbation levels. The large initial spike is likely caused by the models failing to detect objects at further distances, as the sparse amount of points on the objects were likely removed by the dropout. The models trained on the RACECAR dataset saw very little performance degradation from the increased dropout. The PointPillars model only saw a 7% reduction in accuracy on the highest level of dropout, while VoxelRCNN saw no reductions in accuracy.

V. Conclusion

From these experiments, we can conclude that Gaussian noise is a more effective test for evaluating model robustness to perturbations. We found that the

Model	Dataset	Unperturbed	Level 1	Level 2	Level 3
PointPillars	KITTI	0.923288, 31.650	0.597260, 30.740	0.586301, 29.770	0.586301, 29.670
VoxelRCNN	KITTI	0.966887, 6.260	0.619205, 6.180	0.609272, 6.060	0.596026, 6.030
PointPillars	RACECAR	0.950000, 1.620	0.920000, 1.580	0.890000, 1.680	0.880000, 1.690
VoxelRCNN	RACECAR	0.980000, 1.040	0.980000, 0.980	0.980000, 1.020	0.980000, 1.020

TABLE II
EVALUATION RESULTS ON THE DATASET WITH LARGER DROPOUT RATES. TOP NUMBERS INDICATE THE PERCENTAGE OF OBJECTS DETECTED WITH 0.5 IOU AND THE BOTTOM NUMBER INDICATES THE AVERAGE NUMBER OF DETECTIONS PER SCENE.

VoxelRCNN model is more robust to both Gaussian noise and dropout perturbations and exhibits the desired characteristic of having stable outputs in response to noise. The outputs of the PointPillars model, however, became increasingly unstable with higher levels of noise. Although it successfully detected objects at a higher rate than VoxelRCNN, the added noise in the output would likely cause severe issues for the planning and control stack of an autonomous vehicle.

A limitation of this work is that only Gaussian noise and dropout perturbations were explored. Other perturbations, such as random rotation of points and flipping points across axes, could also be tested for a more thorough analysis of testing models subject to perturbations.

REFERENCES

[1] S. Pendleton, H. Andersen, X. Du, *et al.*, "Perception, planning, control, and coordination for autonomous vehicles," *Machines*, vol. 5, no. 1, p. 6, Feb. 2017, ISSN: 2075-1702. DOI: 10.3390/machines5010006. [Online]. Available: http://dx.doi.org/10.3390/machines5010006.

[2] S. Kollazhi Manghat, "Multi Sensor Multi Object Tracking in Autonomous Vehicles," en, Accepted: 2019-12-12T15:20:43Z, Thesis, Indiana University–Purdue University Indianapolis, Dec. 2019. DOI: 10.7912/C2/2564. [Online]. Available: https://scholarworks.iupui.edu/handle/1805/21459.

[3] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[4] H. Caesar, V. Bankiti, A. H. Lang, *et al.*, *Nuscenes: A multimodal dataset for autonomous driving*, 2020. arXiv: 1903.11027 [cs.LG].

[5] *OpenPCDet: An open-source toolbox for 3d object detection from point clouds*, original-date: 2020-03-14T14:48:31Z, Sep. 2, 2022. [Online]. Available: https://github.com/open-mmlab/OpenPCDet (visited on 03/16/2023).

[6] Y. Li, C. Wen, F. Juefei-Xu, and C. Feng, *Fooling lidar perception via adversarial trajectory perturbation*, 2021. arXiv: 2103.15326 [cs.CV].

[7] Y. Cao, C. Xiao, B. Cyr, *et al.*, "Adversarial sensor attack on LiDAR-based perception in autonomous driving," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, ACM, Nov. 2019. DOI: 10.1145/3319535.3339815. [Online]. Available: https://doi.org/10.1145%2F3319535.3339815.

[8] X. Wang, M. Cai, F. Sohel, N. Sang, and Z. Chang, "Adversarial point cloud perturbations against 3d object detection in autonomous driving systems," *Neurocomputing*, vol. 466, pp. 27–36, 2021, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2021.09.027. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231221013850.

[9] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, *Pointpillars: Fast encoders for object detection from point clouds*, 2019. arXiv: 1812.05784 [cs.LG].

[10] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, *Voxel r-cnn: Towards high performance voxel-based 3d object detection*, 2021. arXiv: 2012.15712 [cs.CV].

[11] A. Kulkarni, J. Chrosniak, E. Ducote, *et al.*, *Racecar dataset*, https://github.com/linklab-uva/RACECAR_DATA, 2023.
[12] *Indy autnomous challenge*, https://www.indyautonomouschallenge.com/, 2022.