
COMPREHENSIVE ASSESSMENT OF LiDAR EVALUATION METRICS: A COMPARATIVE STUDY USING SIMULATED AND REAL DATA

Syed Mostaquim Ali

National Research Council Canada
London, ON, Canada
Civil and Environmental Engineering
Western University, London, ON, Canada
syedmostaquim.ali@nrc-cnrc.gc.ca; sali632@uwo.ca

Taufiq Rahman

National Research Council Canada
London, ON, Canada
taufiq.rahman@nrc-cnrc.gc.ca

Ghazal Farhani

National Research Council Canada
London, ON, Canada
ghazal.farhani@nrc-cnrc.gc.ca

Mohamed H. Zaki

Civil and Environmental Engineering
Western University, London, ON, Canada
m.zaki@uwo.ca

Benoit Anttil

Transport Canada
Ottawa, Canada
benoit.anttil@tc.gc.ca

Dominique Charlebois

Transport Canada
Ottawa, Canada
dominique.charlebois@tc.gc.ca

November 6, 2025

ABSTRACT

For developing safe Autonomous Driving Systems (ADS), rigorous testing is required before they are deemed safe for road deployments. Since comprehensive conventional physical testing is impractical due to cost and safety concerns, Virtual Testing Environments (VTE) can be adopted as an alternative. Comparing VTE-generated sensor outputs against their real-world analogues can be a strong indication that the VTE accurately represents reality. Correspondingly, this work explores a comprehensive experimental approach to finding evaluation metrics suitable for comparing real-world and simulated LiDAR scans. The metrics were tested in terms of sensitivity and accuracy with different noise, density, distortion, sensor orientation, and channel settings. From comparing the metrics, we found that Density Aware Chamfer Distance (DCD) works best across all cases. In the second step of the research, a Virtual Testing Environment was generated using real LiDAR scan data. The data was collected in a controlled environment with only static objects using an instrumented vehicle equipped with LiDAR, IMU and cameras. Simulated LiDAR scans were generated from the VTEs using the same pose as real LiDAR scans. The simulated and LiDAR scans were compared in terms of model perception and geometric similarity. Actual and simulated LiDAR scans have a similar semantic segmentation output with a mIoU of 21% with corrected intensity and an average density aware chamfer distance (DCD) of 0.63. This indicates a slight difference in the geometric properties of simulated and real LiDAR scans and a significant difference between model outputs. During the comparison, density-aware chamfer distance was found to be the most correlated among the metrics with perception methods.

1 Introduction

Autonomous driving systems (ADS) are significantly different from traditional vehicles, which are composed of mostly mechanical components. Even the few software components used in traditional vehicles (e.g., anti-lock braking system control, stability control, etc.) are developed from classical control theory to operate in a deterministic manner. In contrast, the software components in an ADS are much more extensive. Some algorithms, such as those that are machine learning based, are stochastic in nature. Unlike traditional automotive software components, a causality based relationship between inputs and outputs cannot always be established. Furthermore, they are expected to operate in non-deterministic environments characterized by infinite numbers of permutations and combinations of different factors involving the operating environment, which renders obtaining absolute proof of safety impossible [1]. In addition, traditional testing methods used for safety evaluation of automotive systems cannot be adapted for ADS [2]. Unsurprisingly, it was estimated in [3] that an ADS system must be driven 11 billion miles requiring 500 years of real-world operation just to prove with acceptable statistical significance that its capabilities are 20% better than the safety benchmark of the average human driver of 1.09 fatalities per 100 million miles. Therefore, it is deemed impractical to rely on on-road and track testing alone to generate the test coverage needed for ADS. A feasible alternative is virtual testing [4] which can be used to generate effective test coverage at a much lower time and resource cost.

Regulatory bodies such as the United Nations Economic Commission for Europe (UNECE) and the US Department of Transportation (USDOT) have already recognized virtual testing as an ADS validation methodology to complement track testing, real-world testing, in-service monitoring and reporting [5, 6]. However, many open questions must be answered first before virtual testing can be adopted as a regulatory tool. Some of these questions have been identified in literature [6, 7]: (a) how reliability is defined in relation to autonomous operation, (b) the minimum degree of fidelity of the simulation models to enable virtual testing, (c) how the simulation models are validated, and (d) logistical considerations such as what models are used, who supplies these models and how or if proprietary algorithms may be supplied. The scope of this paper will focus on many aspects of (a), (b) and (c) of the above mentioned questions in relation to the virtual testing environments (VTE).

A VTE serves as a replacement for resource-intensive real-world testing. It includes models that describe the geometry, appearance and behavior of static (e.g., road lanes, traffic signs, buildings) and dynamic (e.g., vehicular traffic, pedestrians, weather) elements of the roadway environment, and the software that evaluates these models to provide a view of how the driving environment evolves in time and how the VTE under test performs in different test cases. Since the concept of operational design domain (ODD) [8] is used to determine the safety boundary of an ADS or feature thereof, it is expected that a VTE also encapsulates the ODD of the ADS under test. A VTE should closely mimic the defined ODD and any deviation should be statistically comparable. Therefore it is essential to define evaluation metrics to establish confidence for virtual testing. In this paper, available literature on evaluating high-fidelity 3D virtual testing environments for ADS is reviewed with a view to discuss the metrics and benchmarks proposed for evaluating VTEs. We aim to provide a comprehensive overview that can guide future research in establishing confidence for VTEs for ADS.

2 Literature Review

2.1 LiDAR Sensing Technology

LiDAR and cameras are two critical technologies for environmental perception in ADS technologies. Although cameras provide high-resolution images and can read traffic signs and lights, they are more susceptible to environmental factors like fog and rain. LiDAR sensor's ability to function in diverse lighting and weather conditions makes it a more suitable option than camera-based systems for autonomous vehicle developers [9]. Although LiDAR is more expensive than cameras, it provides more accurate spatial representation than cameras [10]. It is a crucial sensing system for autonomous vehicle technology, particularly for L3 and higher-level capabilities [11]. The sensor uses pulsed Laser waves to provide precise spatial data for localization and tracking [12]. With the advancement of LiDAR technologies, such as solid-state sensors, there is expected to be a reduction in LiDAR costs [9]. The ADS development and research community has been developing perception tools based on multi-object detection [13, 14, 15] and semantic segmentation [16, 17]. State-of-the-art deep learning techniques such as point transformers [18] and sparse convolution networks [19] have significantly improved model perceptions for LiDAR point clouds.

2.2 Software

Simulation software has been an integral part of the ADS research and development cycle since the early days. There are as many as 72 distinct simulators used in ADS research [20]. Out of the simulators CARLA, Airsim, SUMO are

one of the most popular. Besides conventional software, racing simulators and commercial games like GTA V have been used for ADS development [21, 22].

The article [23] reviewed available simulation tools to find their advantages, disadvantages, and testing objectives based on different criteria such as fidelity, licensing, interface compatibility, access to data, sensor compatibility, graphics and physics realism, and development effort. Previous literature was surveyed and 5 main simulation software analyzed: CARLA, LGSIM, Gemini, SUMO, and OpenPASS. Sumo is primarily a microscopic simulator, and OpenPASS is a text-based traffic simulator. Carla is an open-source high-fidelity driving simulator based on an Unreal Engine. Although CARLA has a difficult installation and build process, it is most flexible because of its versatile and well-maintained Python API. CARLA also has advanced support for simulated sensors. LGSIM is another open-source simulation software that runs on the Unity engine. LGSIM has an easy-to-use GUI but it lacks a well-maintained Python API. Both LGSIM and CARLA are high-fidelity simulators that require high processing power. esmini is more lightweight compared to CARLA or LGSIM. The disadvantage of esmini is the lack of capability to simulate sensors. CARLA is often chosen simulation software for autonomous driving simulation because of graphics and physics realism, the ability to be customized, and support for sensor simulation.

2.3 Credibility Assessment Framework

Using a virtual testing environment (VTE) as an alternative to real-world testing scenarios requires the VTE to be validated. A pipeline for building trust in VTEs should include tools, models, operators, and documentation. The article [24] provides guidelines for assessing the credibility of modelling & simulation for VTEs and proposes a credibility assessment framework (CAF). The framework evaluates capability, accuracy, correctness, usability, and fit for purpose. The CAF process includes documentation, code, calculation verification, and sensitivity analysis for the VTE platform. The article provides guidelines for establishing credibility by focusing on management, the team’s experience and expertise, input and output data, and the validation process. It also emphasizes the importance of proper documentation of the credibility assessment process.

2.4 LiDAR simulations

The first type of LiDAR simulation relied on physics-based point cloud simulation in a 3D virtual world [25, 26, 27, 28]. The 3D virtual world is mainly developed by computer-aided design (CAD) objects, which only sometimes represent the complexity and details of the real world. To minimize this semantic complexity, real-world datasets have been utilized [29]. The more accurate LiDAR simulation method relies on reconstructing real-world scenarios into 3D meshes and then running physics-based simulations [30, 31]. Point cloud-based generative models are also used to develop virtual environments. Neural Radiance Field (Nerf) is also used for generating simulated LiDAR point clouds [32]. There are also hybrid methods of simulating LiDAR using hybrid methods [33, 34, 35, 36].

2.5 Research Gap

The current state of the LiDAR simulation method validates simulation using qualitative methods and based on the visual comparison method. The qualitative method may be useful but it does not produce a reliable benchmark for comparing different simulation methods. Also, using 3D perception methods for comparison highly depends on model performance. The accuracy of the perception method when comparing real and simulated LiDAR scans depends highly on the model and training data. Also, as most 3D perception methods are black box models, having similar output does not guarantee that both scans are similar. Also, training these models requires a considerable training time for accurate inferences. For this reason, an alternative method for assessing the similarity of real and simulated LiDAR scans has been proposed in the paper. Using geometric similarity methods can eliminate the model bias of 3D perception methods while also producing a benchmark to compare different simulation methods.

3 Proposed Qualification Protocol

The literature provides many general concepts of how VTEs should be qualified for the purpose of ADS validation. Given the enormity of the scope involving a general VTE qualification protocol, even in conceptual form, the individual articles were found to focus on certain aspects of VTE qualification; e.g., scenario generation and definition, validating software tool-chain for safety-critical systems, traffic simulation, vehicle dynamics modeling, etc. Because sensing and perception is a core ADS competency, this paper will focus on qualification of sensors in a VTE.

The synthetic sensor data produced by sensor models can be used in end-to-end simulation testing wherein every aspect of the VTE is simulated or in a X-in-loop testing methodology. Since the physical processes of how sensors

produce perception data (e.g., point cloud generation for LiDAR, image formation for cameras, inertial measurements for an IMU, etc.) are well understood, the most prevalent gap exists in the process of modeling how different ODD affects sensor performance. This gap will be addressed in this paper by specifically focusing on: (a) generation of a ground-truth driving environment, (b) generation of simulated sensor data under different ODDs, (c) development of comparison metrics, (e) development and demonstration how these comparison metrics can be applied for the purpose of qualifying the sensor models in a VTE.

3.1 Generation of Ground-Truth Driving Environment

A survey vehicle equipped with high definition perception sensors (e.g., LiDAR, cameras, IMU, GNSS-RTK) will be used to create a 3D map of a real-world driving environment. The driving environment will be represented by 3D point clouds captured by one or multiple LiDAR sensors. The textures for the different objects in the driving environment will be captured and recreated from calibrated camera data. The GNSS-RTK system will be used to geo-reference the different elements of the ground-truth driving environment.

3.2 Generation of Simulated Sensor Data Under Different ODDs

The ground-truth driving environment will be imported into the CARLA simulator and an ego-vehicle will be programmed to drive in it. The ego-vehicle will be equipped with different sensors (e.g., LiDAR, camera, radar) and the built-in models will be used to generate synthetic data. Subsequently, additional weather simulation models (e.g., simulation of LiDAR in adverse weather [37], weather image translation [38]) will be augmented with the synthetic data to simulate sensor data in different ODDs.

3.3 Development of Comparison Metrics

Comparison metrics that quantify the similarity of simulated sensor data with respect to the ground-truth data will be used to evaluate the simulator’s ability to produce realistic synthetic data. For point-cloud data, cloud matching techniques that accounts for both 3D geometry and the environmental context will be used. Image similarity measures can be used for the image data.

4 Developing Evaluation Metrics

Maximizing the similarity between real LiDAR scans and their simulated counterparts is necessary to minimize the sim2real gap. Consider we have a perfect simulator F_{sim} , which we use to simulate a LiDAR scan $F_{sim}(x_i) = Y_{sim}$, where x_i represents all relevant variables from the real world. Under perfect conditions, both simulated and real LiDAR scans should be identical. However, since it is nearly impossible to develop perfect scenarios, there will always be discrepancies between simulated and real outputs. A similarity metric is necessary to measure the similarity between a simulator and its real-world counterpart. This section outlines the criteria for selecting evaluation metrics and assessing their performance, sensitivity, and validity as a similarity metric for LiDAR scans.

4.1 Conditions for Evaluation metrics

We assume any output of sensor X results from a function F . The function F can result from a real sensor under real-world conditions x_i or a simulated sensor in a virtual testing environment under simulated conditions x_i . The sensor output is determined by the conditions presented x_i in both real and simulated scenarios. The conditions x_i can be different factors such as:

- **Environmental Factors:** location of objects, weather conditions, and temperature.
- **Odometry:** Orientation, translation, rotation, and vehicle speed.
- **System Factors:** Number of sensor channels, field of view, LiDAR hardware characteristics, and computational limitations.

The similarity metrics for comparing LiDAR scans should satisfy the following conditions:

1. **Self-consistency:** For any LiDAR scan a and evaluation metric f , the method should produce the most accurate result when comparing a scan with itself. For instance, $f(a, a)$ should be precisely zero for distance functions, indicating zero distance between the same LiDAR scan.

2. **Symmetry:** For any two scans a and b , the evaluation metric should satisfy the similarity, such that $f(a, b) = f(b, a)$. This property ensures logical consistency across multiple comparisons.
3. **Sensitivity:** The metric should be sensitive to scan variations. Any change in the similarity between two scans should be reflected in the evaluation metric’s output. This ensures that the metric accurately captures differences in sensor outputs.
4. **Computation Efficiency:** The evaluation metric should be computationally efficient with CPU time and memory, allowing fast evaluation of large datasets.
5. **Well-defined range:** The evaluation metric should have a well-defined range that provides an interpretable scale for comparison. This allows for consistent comparison across different simulations and units.

4.2 Generating Cases to Evaluate Similarity Metrics

To assess the metrics’ sensitivity to identifying changes in LiDAR scans, we generated a series of cases using both real and simulated LiDAR data. These cases evaluate the metric’s responsiveness to changes in LiDAR scan characteristics.

4.2.1 Channel settings

To evaluate the metrics response to different LiDAR settings, we simulate LiDAR scans in a Carla Environment using three widely used models: the 16-channel VLP-16, the 32-channel VLP-32c, and the 64-channel HDL-64E which are commonly adopted in autonomous driving research [39]. These different LiDAR sensors differ in the number of points and vertical resolution. Figure 1 illustrates LiDAR scans generated with different simulated LiDARs. A Carla scene in an urban environment simulated these LiDAR scans with the same position and orientations. The objective of the experiment is to scan the same objects with exact positions and orientations with different LiDAR sensors. Furthermore, these scans will be used to determine the sensitivity of metrics to different LiDAR settings and the accuracy of metrics in identifying LiDAR scans of similar geometry, despite varying LiDAR settings.

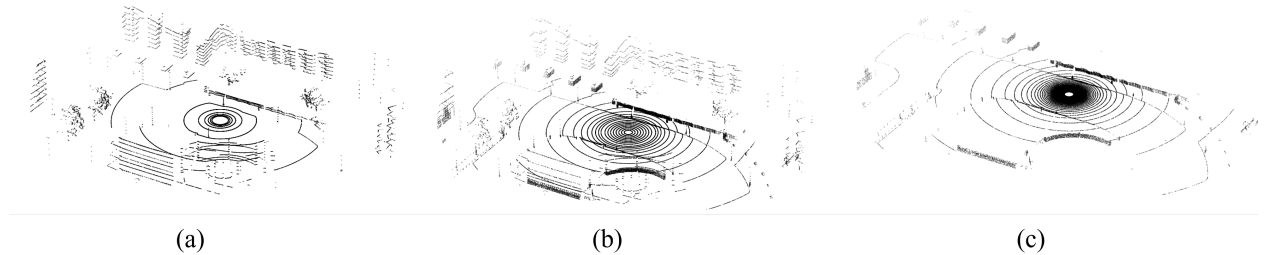


Figure 1: Simulated LiDAR scans of different channel settings from left VLP-16, VLP-32c, HDL-64E (top plots provide the spatial channel distribution for different LiDAR models)

4.2.2 Orientation

LiDAR scans with different orientations were simulated in a Carla Urban driving scene [25]. This approach generates LiDAR scans with different orientations by simulating the rotation of the LiDAR in three axes. Simulated LiDAR scans were recorded in positions with different rotation values. This study was done in a simulated Environment, as replicating such conditions in real-world settings poses significant challenges. This experiment aims to assess the sensitivity of metrics to changes in orientation and determine whether metrics can accurately identify similar LiDAR scans of the exact location, despite different orientations. Figure 2 shows simulated LiDAR scans with different orientations.

4.2.3 Distortion

Distortion in LiDAR scans can occur due to various factors, including the Doppler effect resulting from movement, vibrations, or inaccuracies in laser timing [40]. For our experiment with evaluation metrics, we introduced the distortion using spatial transformation. Real LiDAR scans are rotated, skewed, and translated to generate distorted and undistorted pairs for comparison. These pairs are compared through each evaluation metric to assess their performance in terms of sensitivity to distortion and accuracy of identifying original pairs. Figure 3 shows examples of cases of distortion.

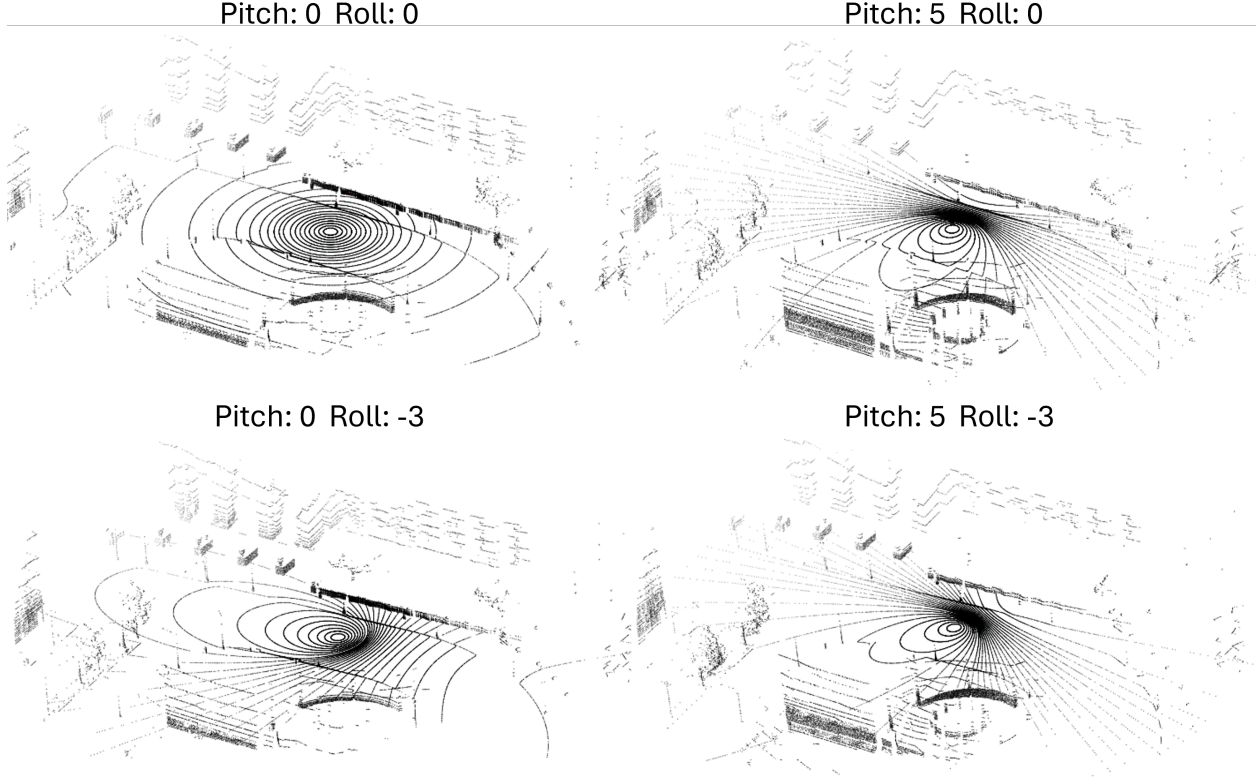


Figure 2: Simulated LiDAR scans with different orientations

4.2.4 Noise

Noise in LiDAR scans can occur due to various factors, including weather conditions, sensor inaccuracies, or other environmental influences [41]. Random noise of varying intensities was introduced into the LiDAR scans. The noise intensity added was measured using standard deviation to ensure a controlled and measurable approach. Noises of varying intensities were added to real LiDAR scans to create multiple original and noisy scan pairs. Figure 4 shows LiDAR scans with added noise. Each scan pair was used as a metrics input to assess performance for noisy data. Generating multiple cases with varying noise intensities allows us to compare metric performance for different noise levels in a controlled manner.

4.2.5 Outliers

In real-world LiDAR systems, scans often contain outliers generated by physical, environmental, or sensor-related issues [42]. Outliers are defined as points that do not conform to the geometric properties of the scanned scene. Two types of outliers usually occur in real-world scenarios. Random scattered outliers, which are isolated points, appear randomly across the scene, far from any surface in the scan. These often occur due to smoke, fog, dust or weather conditions such as snow or rain. Cluster scattered outliers are groups of erroneous points that do not correspond to any object in the scene. These are usually caused by reflections from glass, water or other reflective objects, and sensor errors. Controlled outliers were introduced into real LiDAR scan point clouds to create test pairs for evaluation metrics. We added two outliers for our experiment: randomly scattered and clustered points. The random scattered points were distributed within the bounds of the LiDAR scan to simulate isolated outliers in the scan. The number of points measured the intensity of the outliers added. Figure 5 shows a LiDAR scan with randomly scattered outliers. The clustered outliers consisted of points within a fixed radius and position. The clusters were added randomly to real LiDAR scans of varying positions, radii, and number of points within a threshold. Figure 6 shows a LiDAR scan with clustered outliers. Here, the number of clusters determined the intensity of the outliers. The approach of generating LiDAR scans with varying outliers allows for a detailed analysis of how different metrics are affected by outliers.

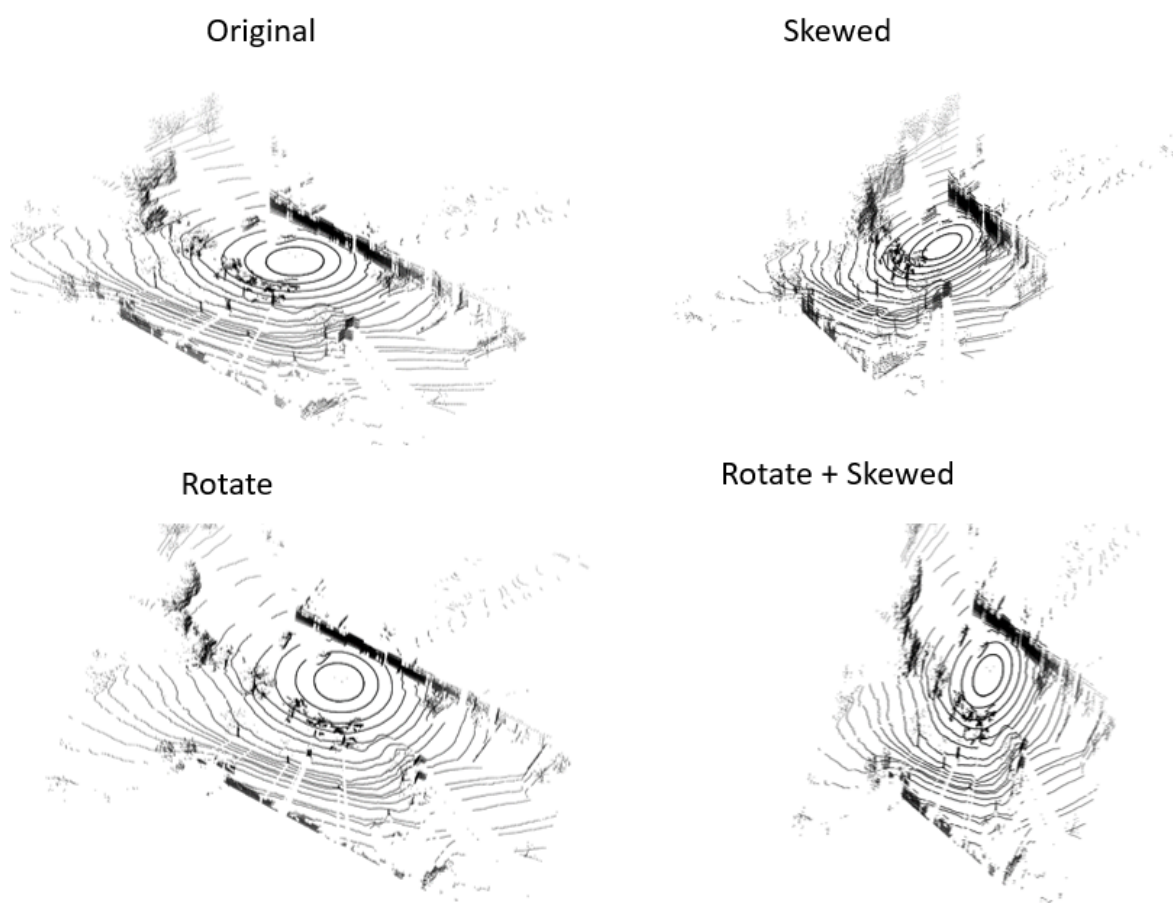


Figure 3: Simulated LiDAR scans with different distortions

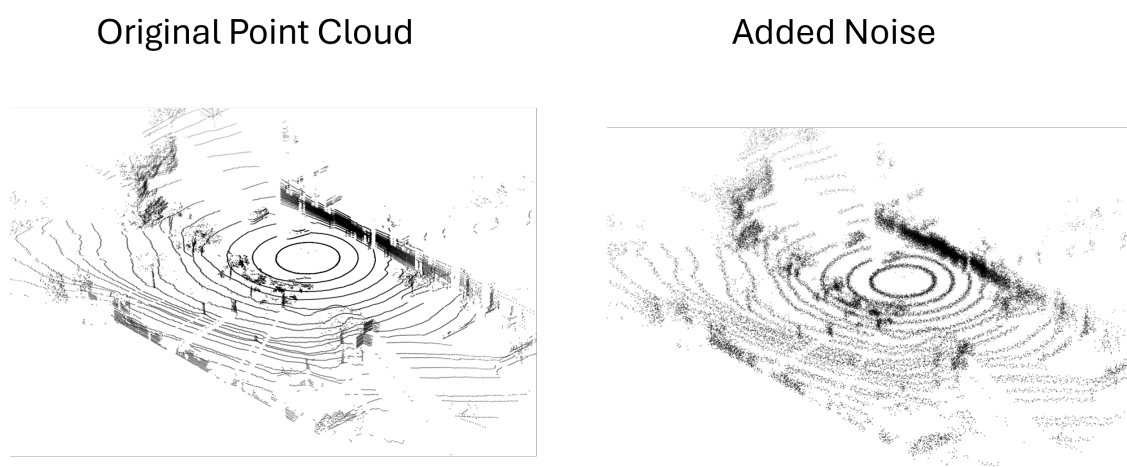


Figure 4: Added Noise to Real LiDAR Scans. In this example the magnitude of the noise is 0.2 normal standard deviation.

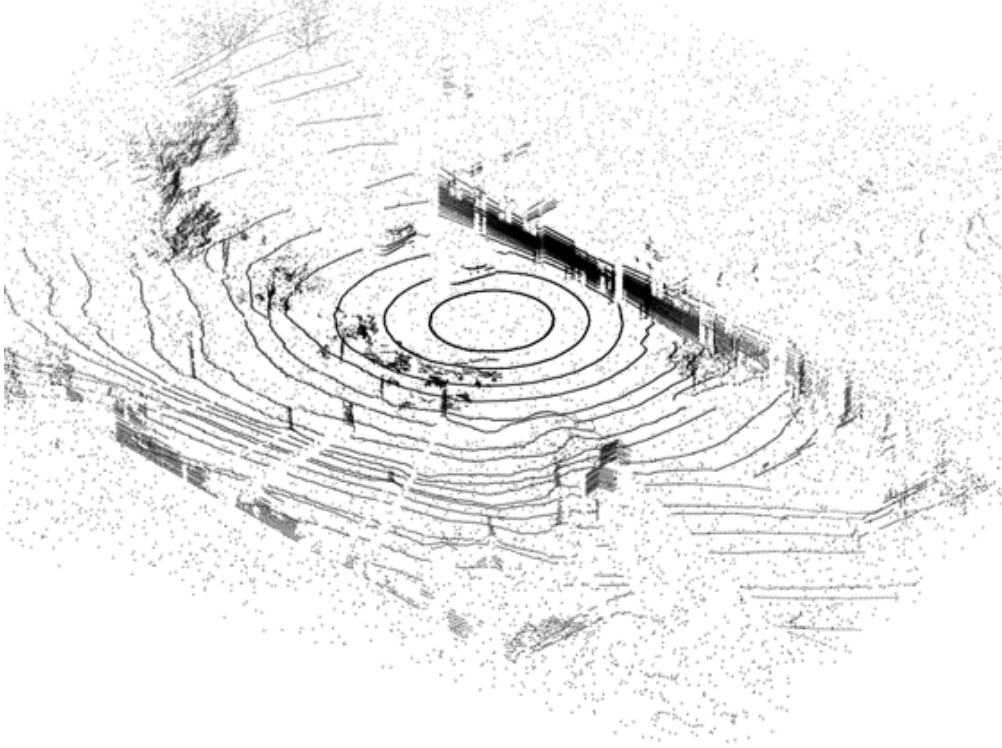


Figure 5: Added Random Scattered points into real LiDAR scans. The figure shows 10,000 random outlier points within the point cloud bounds.

4.2.6 Density

Point density is one of the critical properties of LiDAR scans affected by sensor type, distance to objects, and environmental factors [43]. To evaluate the sensitivity of metrics to variations in point density, we generated LiDAR scans with lower density by downsampling real-world LiDAR data. From real LiDAR scans, a fraction of points is randomly sampled to generate lower-density point clouds. Random sampling is used instead of voxel-based down-sampling [44] because random sampling introduces an imbalance in point cloud density, whereas voxel-based down-sampling maintains a uniform distribution. Generating point clouds with varying densities enables us to test the metric’s sensitivity to varying densities and its ability to identify original pairs.

4.3 Geometric Similarity Measures

Simulating LiDAR scans should closely resemble their real-world counterparts in terms of their geometric properties. The similarity metrics proposed focus on finding geometric similarity to generate an evaluation score that quantifies the similarity between LiDAR scans. In this naive approach, we only measure the similarity of LiDAR scans in terms of geometry while ignoring other features, such as intensity. This method two LiDAR scans P_1 and P_2 with n and m number of points are compared where,

- $P_1 = p_1^{(1)}, p_1^{(2)}, \dots, p_1^{(n)};$
- $P_2 = p_2^{(1)}, p_2^{(2)}, \dots, p_2^{(m)};$

here, p refers to each point in the pointcloud. The exponent refers to each unique point.

4.3.1 Chamfer Distance

The chamfer distance [45] is a commonly used method for point clouds in various applications. This method measures the minimum average Euclidean distance between two sets of points. The chamfer distance between two point clouds can be defined in equation 1. This method is symmetric because of its two components. It measures the average squared distance from each point P_1 to its nearest point in P_2 . Then, in the second term for P_2 and P_1 , vice versa. A smaller

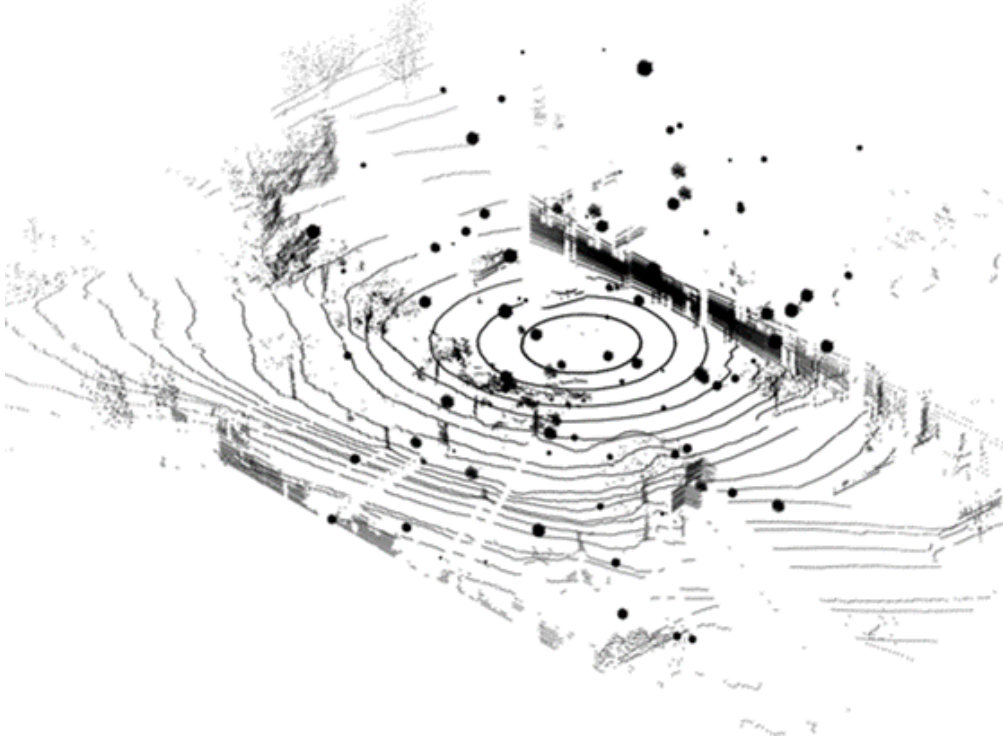


Figure 6: Added Clustered Outliers into real LiDAR scans. The figure shows 100 clustered points with random sizes with a limit of 1000 points. The clusters have a maximum radius of 1 unit. The clustered point clouds are to be added within a maximum of 20 units radius from the center of the LiDAR scan.

chamfer distance indicates greater similarity between point clouds, making it an excellent geometric similarity measure between two LiDAR data sets. The range of Chamfer distance is $0 \rightarrow \infty$. Where 0 indicates the maximum similarity.

$$d_{chamfer}(P_1, P_2) = \frac{1}{|P_1|} \sum_{p_1 \in P_1} \|p_1 - p_2\|^2 + \frac{1}{|P_2|} \sum_{p_2 \in P_2} \|p_2 - p_1\|^2 \quad (1)$$

In the equation 1, $d_{chamfer}$ refers to the chamfer distance, P refers to pointcloud. p_1 and p_2 refers to each point in the P_1 and P_2 pointclouds, respectively.

4.3.2 Earth Mover's Distance

The Earth Mover's Distance (EMD) is one of the most widely used methods for quantifying similarity between point clouds [46]. EMD calculates the minimum work required to transform one set of points into another. The earth mover distance is formulated as an optimization problem that finds a one-to-one bijection mapping, denoted as $\phi : P_1 \hookrightarrow P_2$. The algorithm is presented in equation 2, where the goal is to minimize the total cost of transforming the point cloud P_1 into P_2 . Although this method provides a more detailed measure of dissimilarity than simple metrics like Chamfer Distance, this method requires both point clouds to be the same size. This property of EMD makes it impractical to compare LiDAR scans, as the number of point clouds generated in a LiDAR scan can not be guaranteed.

$$d_{EMD}(P_1, P_2) = \min_{\phi: P_1 \rightarrow P_2} \sum_{p_1 \in P_1} \|p_1 - \phi(p_1)\| \quad (2)$$

2, d_{EMD} refers to the earth mover's distance, P refers to pointcloud. p_1 and p_2 refers to each point in the P_1 and P_2 pointclouds, respectively. ϕ refers to the bijection mapping.

4.3.3 Density Aware Chamfer Distance

Wu et al. [47] introduced Density Aware Chamfer Distance (DCD) as an advanced method for comparing point clouds. The DCD is calculated as shown in equation 3. The formulation of this method incorporates the term e^z , where $z < 0$, ensuring the resulting distance is bounded between 0 and 1. The metric also introduces a scaling parameter for the distance α , which can be used to adjust the variations in distances, allowing for a more precise comparison of point clouds. This method is more sensitive to disparities in point cloud than Chamfer distance and computationally efficient than Earth mover’s distance. Also, the metric does not require a point cloud of similar sizes to compare, which makes it perfect as a geometric evaluation metric for LiDAR scans.

$$d_{\text{DCD}}(P_1, P_2) = \frac{1}{2} \left(\frac{1}{\|P_1\|} \sum_{p_1 \in P_1} \left(1 - \frac{1}{n_{\hat{p}_1}} e^{-\alpha \|p_1 - \hat{p}_2\|_2} \right) + \frac{1}{\|P_2\|} \sum_{p_2 \in P_2} \left(1 - \frac{1}{n_{\hat{p}_2}} e^{-\alpha \|p_2 - \hat{p}_1\|_2} \right) \right) \quad (3)$$

3, d_{DCD} refers to the density aware chamfer distance, P refers to pointcloud. p_1 and p_2 refers to each point in the P_1 and P_2 pointclouds, respectively, α refers to the sensitivity, and n refers to number of points in the pointcloud.

4.3.4 Histogram Method

This method converts the distance between points into a histogram [48]. The histogram is generated by finding the pairwise distance between points from the object’s surface. This method takes random point samples from the point cloud to control the run time. The histogram method is very suitable for unstructured point clouds. Wallace et al. [49, 50] propose a custom histogram method for point clouds. In their method, distances are normalized by using the largest object distance. They claimed this method of normalization helps to preserve distance. The Minkowski distance was used to compare the absolute differences between the two histograms. Since this method calculates pairwise distances, the memory requirement is $O(n^2)$. This limitation means that to compare two point clouds with N points, it needs to compute a histogram from a matrix of size $2N^2$ elements. For this reason, the author suggested random sampling point clouds to N elements. We also tested the histogram method with Voxel downsampling to test the performance of metrics.

4.3.5 Iterative Closest Point

Iterative closest point is a point cloud registration method that registers two point clouds by finding the nearest point in the other cloud and calculating the distance between them. This method uses different objective functions to measure overlaps, such as point-to-point distance, point-to-plane distance, and surface normal alignment. Chamfer distance is one of the most popular optimization metrics used for the iterative closest point method. For our tests of geometric comparison, Iterative closest points with chamfer distance as a loss function were used as a similarity metric. The metric registers LiDAR scans together and generates a final inlier root mean square error between the registered point clouds. The final score is used as a metric for evaluation. One of the disadvantages of this method is that the Iterative Closest Point is not symmetric, and the registration process is computationally expensive.

5 Comparison of Geometric Similarity Metrics

This section aims to discuss the capability of the similarity metrics. The similarity metrics are tested in different scenarios, and the results are discussed.

5.1 Sensitivity of Metrics

5.1.1 Distortion: Rotation

The metrics were tested with multiple LiDAR scans with different rotations to find the sensitivity of the metrics. The results show that Chamfer distance and Birds Eye View increase consistently with rotation. Density-aware Chamfer Distance and Voxel IoU are seen to be very sensitive to slight changes in rotation. The histogram method and the Iterative Closest point show the slightest sensitivity to rotation.

5.1.2 Distortion: Translation

The Chamfer distance and Bird’s Eye View increase linearly with translation. The density-aware Chamfer distance and Voxel comparison are very sensitive to translation. The histogram approach is not sensitive to translation. The key takeaway from this comparison is that if the LiDAR scan has any translation distortion, it is required to find scans with similar geometries. The histogram method is the metric for this purpose.

5.1.3 Distortion: Scale

We see the metrics Chamfer Distance and Bird’s eye view increase linearly when scaling up, but these metrics show the least distance for scaling down. Density aware chamfer distance, ICP (Iterative Closest Point) compare, voxel IoU, and histogram show maximum accuracy for scaling down and up. The Chamfer distance falsely identifies scaled-down LiDAR scans as similar LiDAR scans with zero distance, which might make chamfer distance impractical for scaled-down LiDAR scans.

5.1.4 Noise

The evaluation metrics were tested across different noise levels, up to 2 standard deviations. Figure 7 shows the sensitivity of the metrics. The density-aware chamfer distance (DCD) demonstrated high sensitivity to noise levels. Decreasing the sensitivity α increases the metric DCD’s sensitivity range. The Chamfer distance demonstrated a linear increase in response to noise and downsampling point cloud before comparing increases in sensitivity. Birds Eye View comparison and ICP comparison also showed a linear increase with noise. The Voxel IoU metric followed a similar trend. However, the histogram approach using random sampling showed minimal sampling to noise, although voxel downsampling with the histogram approach showed linear sensitivity toward noise levels. Overall, the results indicate that all the metrics, except for the histogram with random sampling, exhibit moderate sensitivity to noise

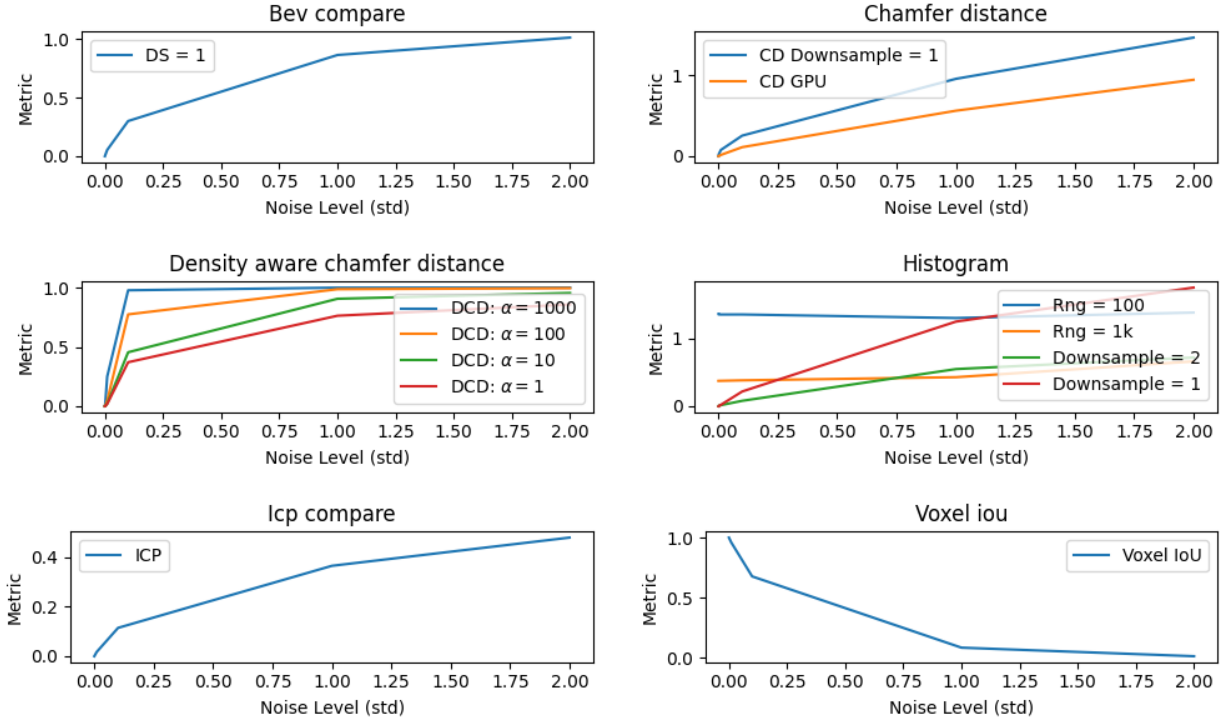


Figure 7: Metric Sensitivity to Added Noise

5.1.5 Outliers

Testing outliers sensitivity we have two different cases. One where we compare the outlier sensitivity to adding random points. And another is for clustered points. **Adding random points:** For testing outliers, we tested by adding points up

to 10,000. Figure 8 shows the metric output with different numbers of points. Density-aware chamfer distance shows the most linear increase with the number of added points. The trend for this metric holds for all α values. Bird eye view comparison and chamfer distance also show a linear trend. Downsampling the point cloud before calculating chamfer distance seems to increase the sensitivity of the metric. The iterative closest point method does not show any sensitivity towards added outliers. The histogram approach shows a linear increase with added points across all variations, where downsampling the point cloud shows a similar trend to Chamfer distance.

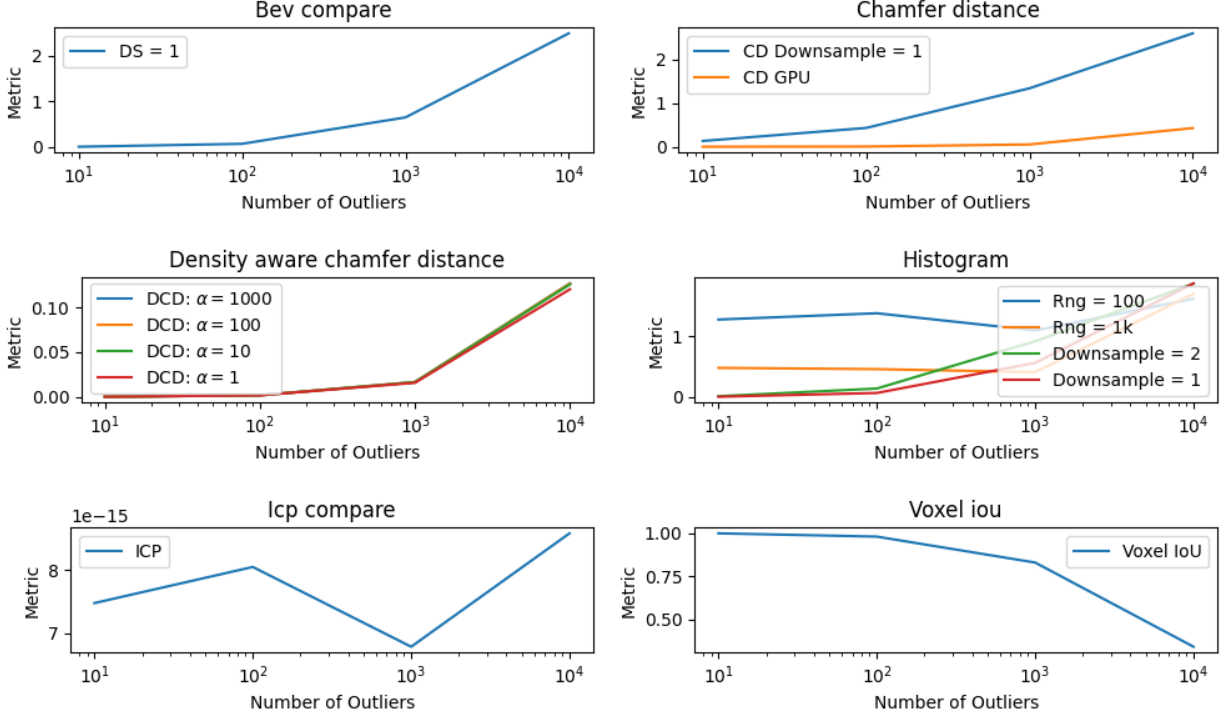


Figure 8: Metric Sensitivity to Random Outliers

Adding Clustered outliers: We tested the metrics comparison for point clouds with up to 1000 added clusters of a maximum size of 1000 points and 10 unit radius. In the figure 9, we see the output of metrics with different numbers of clusters. The comparison shows a linear increase for density-aware chamfer distance for all α values. Chamfer Distance shows a near linear increase. The ICP comparison and voxel method do not show sensitivity towards this approach. From the sensitivity analysis with outliers, it seems that all the methods are sensitive except for the ICP comparison method. Density-aware chamfer distance shows the most consistent increase with outliers.

5.1.6 Density

Lidar scans are randomly sampled to generate scans with different densities. Figure 10 shows metric results with LiDAR scan sampled randomly with different proportions of the original scan. Chamfer distance shows moderate sensitivity, and gradually, the sensitivity decreases. Density-aware chamfer distance shows linear decreases for $\alpha = 1000$. Bird’s eye view, ICP, and voxel IoU show a similar trend as chamfer distance. The histogram method with random sampling is least sensitive to a density-based method, where downsampling shows a similar decrease as chamfer distance. From the analysis, density-aware chamfer distance shows the most linear increase with density proportion to the original scan.

5.2 Accuracy of metrics

LiDAR scans of the same positions were compared with each other with different modifiers (noise, outlier, distortion, orientation and channels). Each of the metrics was used to compare and generate an accuracy score. The accuracy score determines the metrics resilience to such modifications in LiDAR scans. The table 11 shows the accuracy of different metrics. Out of all the metrics, the histogram method works best for translation and rotation scenarios without other distortions. From the table, we see Chamfer distance and density-aware Chamfer distance with $\alpha = 1$ worked best

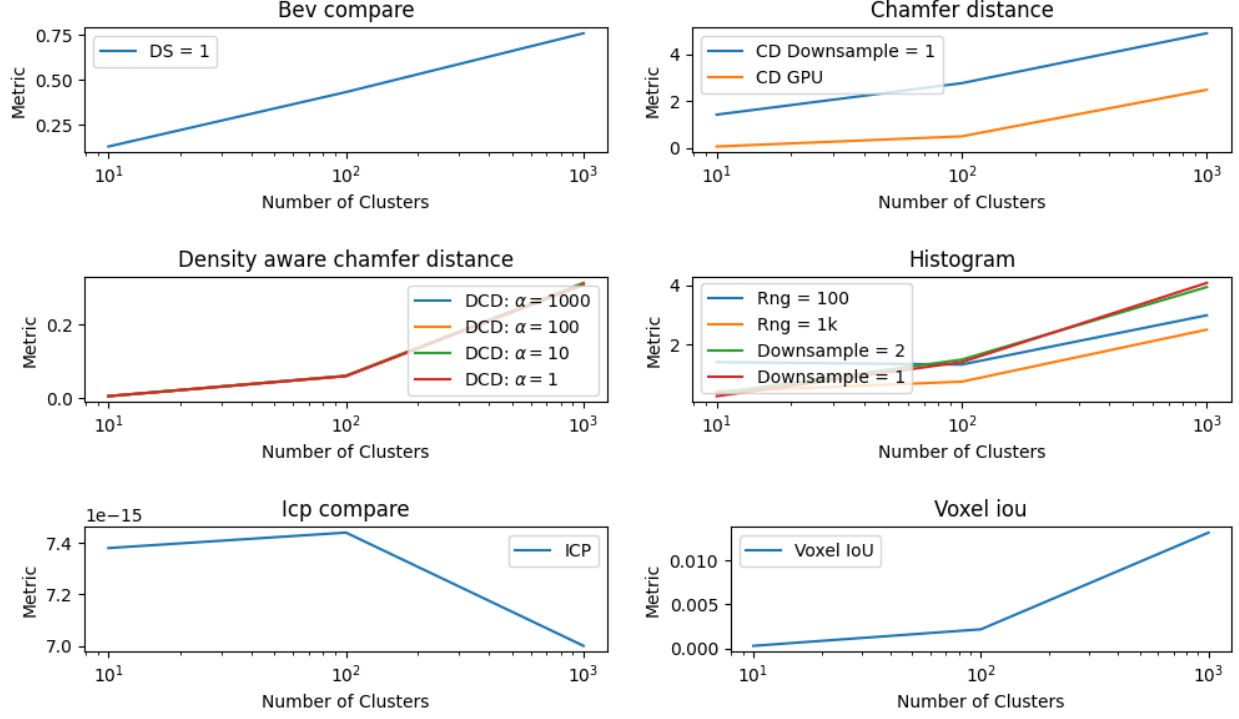


Figure 9: Metric Sensitivity to Clustered Outliers

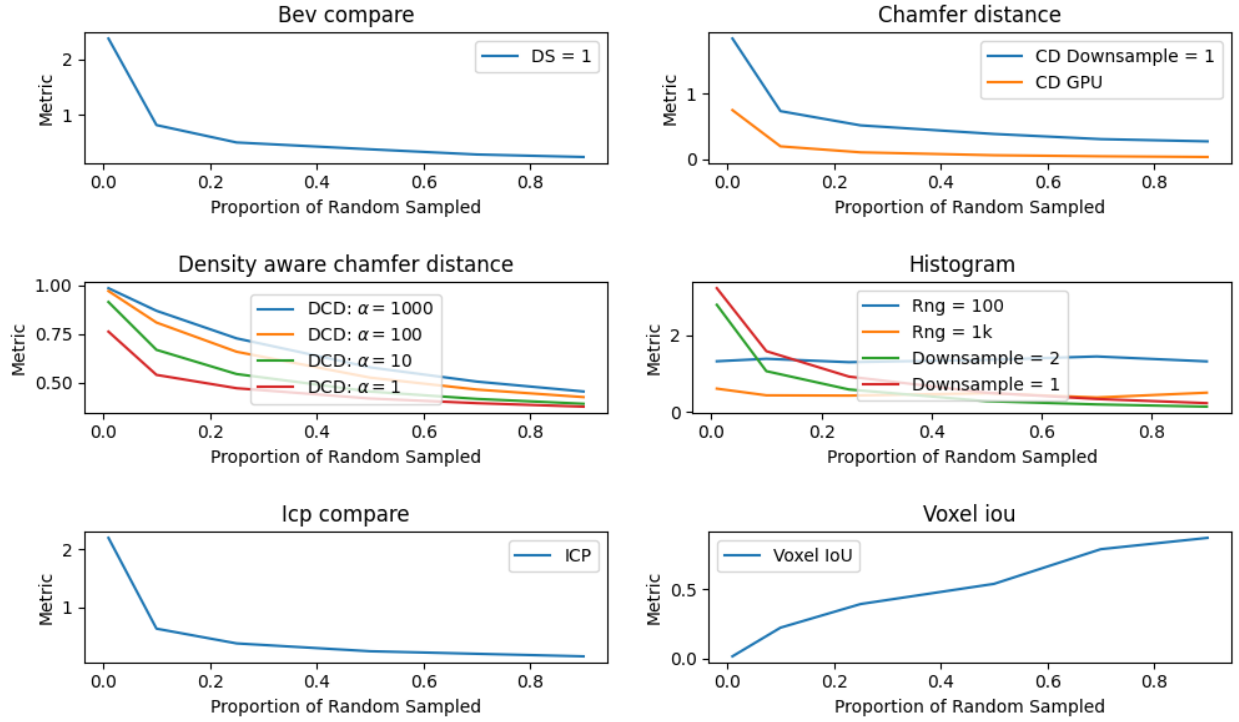


Figure 10: Metric Sensitivity to Density

Metrics	Parameters	Distortion					Noise	Points	Outliers			Channels	Density	Overall
		Scale	Skew	Translate	Rotation	Mean			Cluster	Orientation	Mean			
Birds Eye View	Voxel Down Sample 1	0.33	0.46	0.54	0.50	0.46	1.00	0.95	1.00	1.00	1.00	0.97	1.00	0.90
Chamfer Distance	Voxel Down Sample 1	0.31	0.49	0.51	0.44	0.44	1.00	1.00	1.00	1.00	1.00	0.97	1.00	0.90
		0.34	0.52	0.53	0.44	0.46	1.00	0.95	1.00	1.00	1.00	0.97	1.00	0.90
Density Aware Chamfer Distance	$\alpha=1$	0.29	0.71	0.53	0.51	0.51	0.99	1.00	1.00	1.00	1.00	1.00	1.00	0.92
	$\alpha=10$	0.28	0.70	0.49	0.54	0.50	0.93	1.00	1.00	1.00	1.00	1.00	1.00	0.91
	$\alpha=100$	0.25	0.68	0.47	0.54	0.48	0.87	1.00	1.00	1.00	1.00	1.00	1.00	0.89
	$\alpha=1000$	0.26	0.69	0.41	0.45	0.45	0.83	1.00	1.00	1.00	1.00	1.00	1.00	0.88
	Random Sampling = 100	0.21	0.27	0.70	0.82	0.50	0.71	0.75	0.52	0.42	0.47	0.40	0.80	0.61
Histogram Compare	Random Sampling = 1000	0.24	0.29	1.00	1.00	0.63	0.93	0.80	0.79	0.62	0.70	0.43	0.96	0.74
	Random Sampling = 10000	N/A	0.29	1.00	1.00	0.76	0.96	0.79	0.79	0.57	0.68	0.30	N/A	0.70
	Voxel Down Sample 1	N/A	0.29	1.00	1.00	0.76	0.33	0.64	0.43	0.78	0.61	0.43	0.54	0.55
	Voxel Down Sample 2	0.25	0.27	1.00	1.00	0.63	0.64	0.57	0.38	0.83	0.61	0.43	0.52	0.57
	Iterative Closest Point	0.35	0.33	0.74	0.55	0.49	0.96	1.00	1.00	1.00	1.00	0.80	0.96	0.87
Voxel Intersection over Union		0.24	0.40	1.00	0.31	0.49	0.61	1.00	0.17	0.75	0.46	0.83	0.94	0.72

Figure 11: Accuracy of metrics when compared with different modifiers

for all methods except for the distortion cases. None of the produced accurate results for Scale and Skew distortion scenarios.

5.3 Computation Time

The computation time for each metric is given in Figure 12. The slowest evaluation metric is the histogram method with 10k points, followed by a bird’s eye view without downsampling. The computation time for these metrics makes it impractical for large-scale use. Density-aware chamfer distance is the fastest approach among the computation metrics.

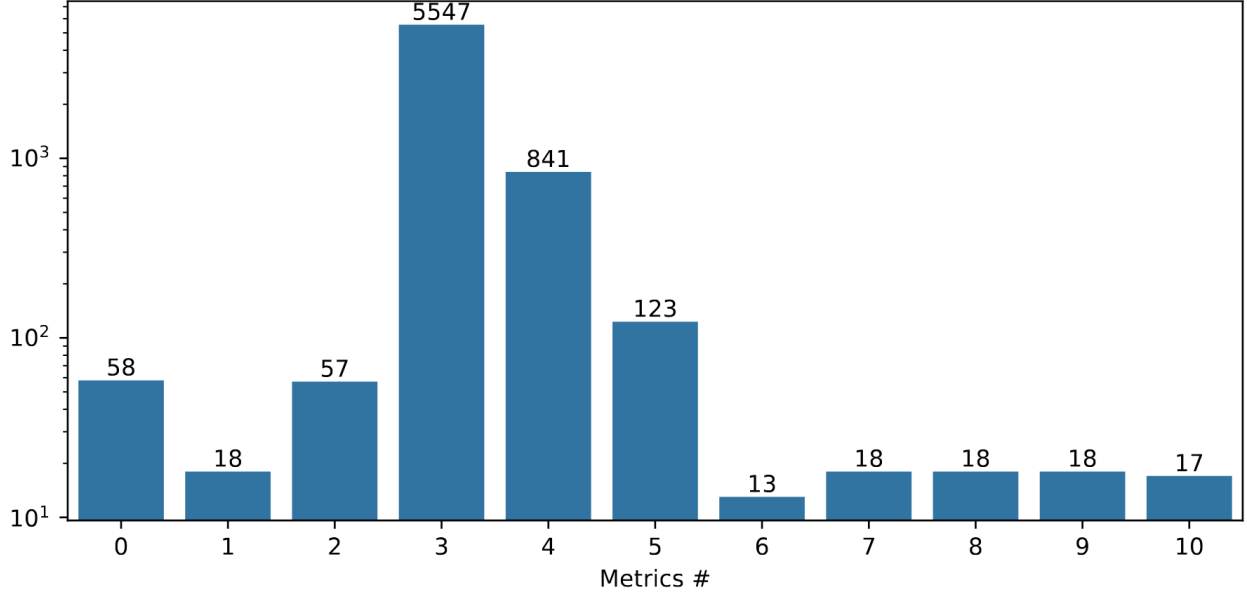


Figure 12: Computation Time. Birds Eye View (0), Chamfer Distance (1), Histogram with Rng = 1000 points (2), with Rng = 10000 points (3), with Downsample = 1 (4), ICP (5), Voxel IoU (6), Density Aware Chamfer Distance; $\alpha = 1000$ (7), $\alpha = 100$ (8), $\alpha = 10$ (9), $\alpha = 1$ (10),

5.4 Reflexivity of Metrics

The metrics f compared random LiDAR scans x with themselves to check if $f(a, a)$ should produce maximum similarity. All the metrics, except the histogram with random sampling, successfully output the maximum similarity. During our testing, we found that comparing the same point cloud with the metric histogram method with a random sampling of 100, 1000, and 10000 points produced values of 1.46 ± 0.38 , $0.46 \pm .12$, and $.15 \pm 0.4$, respectively.

5.5 Summary of Comparison

From the Comparison, it is observed that each of the similar methods is well-suited for specific tasks. However, to find geometric similarity between real LiDAR scans, we aim to find a metric that shows linear sensitivity to deformation, has good accuracy, and has reasonable computation time. Chamfer Distance, Density Aware Chamfer Distance, and Bird's Eye View are suitable geometric similarity measures. Based on our analysis, we recommend a two-step approach for geometric similarity analysis. First, test with density-aware chamfer distance with $\alpha = 1$. If the score is higher than 1, it means the scans are not very similar; testing again with chamfer distance will generate a better idea of how far real and simulated scans are. The combined approach optimizes accuracy, sensitivity, and computation efficiency.

6 Virtual Test Environment Generation from Vehicle Data

This section describes the digital twin development for the Carla simulator with collected real-world data. The data is collected with a vehicle mounted with LiDAR, IMU, and Camera sensors. These scans generate a 3D Carla scene using SLAM and 3D mesh generation techniques. The digital twin environment aims to create simulated replications of LiDAR scans collected in the real world. The LiDAR scans will then be evaluated to measure deviation from real-world scenarios.

6.1 Data Collection

A 2021 Toyota RAV4 Hybrid is equipped with a Velodyne 32c LiDAR, 3 Zed 2i stereo depth camera, and an inertial measurement unit (IMU). The sensors were equipped with sufficient computing and power supply equipment. Figure 13 illustrates the vehicle setup. We employed the Robot Operating System (ROS) framework [41] for effective sensor management, deployment, and data handling. ROS provides a versatile environment for processing, recording, and analyzing sensor data through its modular node-based architecture. Communication within ROS occurs via nodes that exchange data as messages using a publisher-subscriber model. In this model, nodes publish messages to specific topics, and other nodes can subscribe to these topics to access the data. Additionally, ROS supports the recording of topic messages into ROS bag files, facilitating convenient offline analysis. During the survey, we recorded all sensor data into ROS bag files for subsequent detailed examination. During the survey, the ROS framework recorded the messages for all sensors with accurate timestamps, allowing multi-sensor fusion.

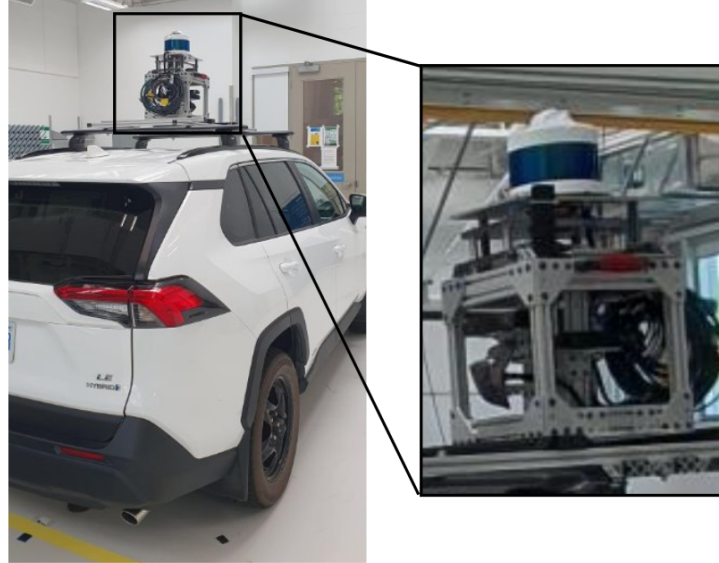


Figure 13: Data collection vehicle setup.

Western University Research Park was selected as the study area. The Figure shows the path taken by the vehicle. We strategically conducted the survey early in the morning, a time when the presence of pedestrians and cars was minimal. We aim to generate a digital twin to test LiDAR performance with static objects, as having dynamic objects in the scene may compromise the accuracy of digital twin generation. The vehicle covered a distance of 1 kilometre in

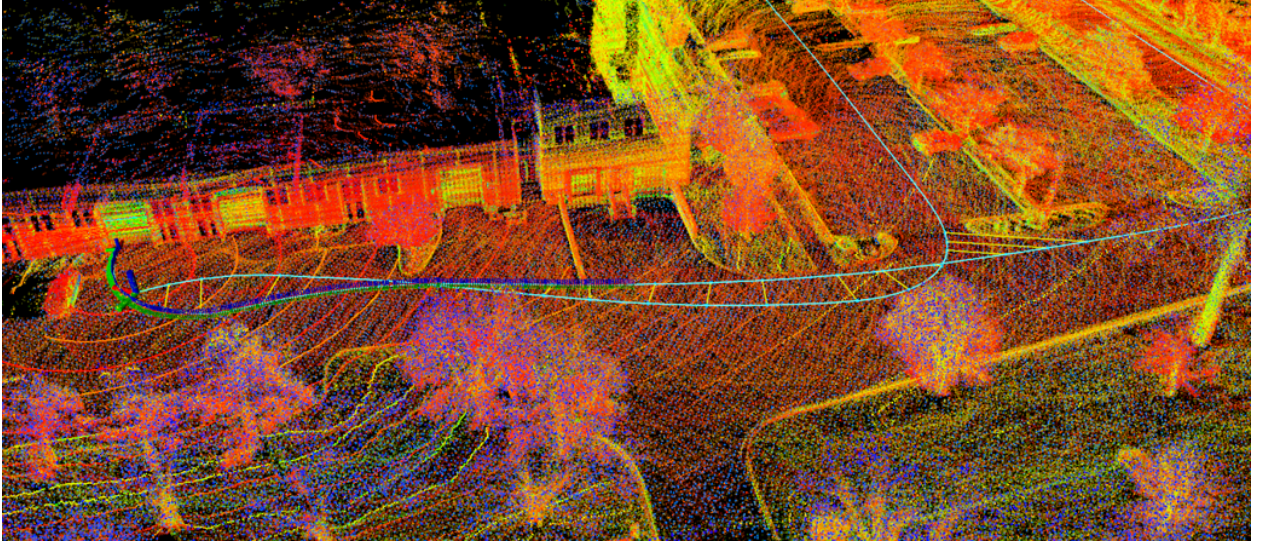


Figure 14: 3D point-cloud reconstruction with LIO-SAM. The point colour shows the intensity of LiDAR scans. The green line indicates the trajectory of the vehicle.

driving, returning to its starting position. A closed-loop trajectory was required for the accuracy of the Simultaneous Localization and Mapping (SLAM) algorithm.

6.1.1 Map Generation with SLAM

LiDAR (Light Detection and Ranging) is a remote-sensing technology that measures a target’s position by analyzing reflected light. Our experiment uses a 32-channel LiDAR VLP-32c, emitting laser pulses at 32 angles. This enables scanning multiple vertical planes, creating a detailed 3D point cloud. A single LiDAR scan provides an accurate 3D representation of points on targets from which lasers are reflected, providing a 3D snapshot of an area. The Simultaneous Localization and Mapping (SLAM) algorithm can be utilized to find the odometry of each LiDAR scan. This odometry information can be utilized to reconstruct the snapshots into a 3D point cloud map. The odometry of each scan describes the position (t_x, t_y, t_z) and orientation (r_x, r_y, r_z, r_w) of the sensor during data collection.

This research utilizes LiDAR Inertial Odometry via Smooth Mapping (LIO-SAM) [51] for reconstructing 3D point clouds. This method integrates motion data from the Inertial Measurement Unit (IMU) and optimized odometry solution from LiDAR scans to generate accurate odometry of the LiDAR scans. The generated odometry is further corrected using loop closure methods. This fusion of multiple data enhances the precision of the generated 3d map. Figure 14 shows a snapshot of LiDAR reconstruction with SLAM. This process generates an accurate 3D map of the scanned area in meter coordinates.

6.1.2 Point Cloud Processing

We use registered point clouds in conjunction with other environments and sensor information to generate the point cloud. Initially, the point cloud was generated in .ply format using the Python open3D library. The point cloud is a collection of points as each row contains position information (x, y, z) . We also add sensor position (x, y, z) during recording into the point cloud. Using the normal estimation method, we use the sensor position to estimate the direction of normals for each point.

6.1.3 Mesh Generation

Before generating the mesh, the point cloud was sub-sampled to reduce the number of points. Then, we generated the mesh from the subsampled point cloud using the Poisson reconstruction method. Initial faces on the mesh generated by this method are filtered based on the point density. Figure 15 shows the mesh generated. The mesh is then converted to FBX file format for importing to the unreal engine.

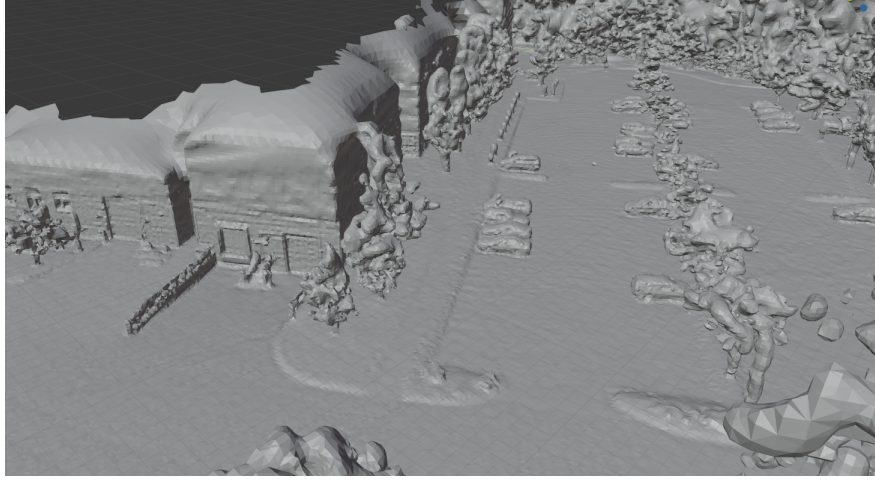


Figure 15: Generated Mesh in Unreal Engine Environment.

6.1.4 Carla Scene Generation

We used Carla’s development build to develop the digital twin scenario. Generally, Carla requires a mesh and an open drive file to generate custom maps. However, as our goal was to generate a static scene for simulating sensors, we used an empty opendrive file. We created a blank project in Carla and imported the mesh. The original point cloud coordinate was in meters, so no additional scaling was required. The mesh’s zero point origin was translated to the unreal engine’s origin. The result is that the unreal coordinate matches 1:1 with the ROS coordinate. Since the objective of our research is LiDAR simulation, no additional texture or light adjustment was made for the scene. Figure 15 shows the scenario in Carla.

6.1.5 LiDAR Simulation

Carla Ray cast LiDAR simulation, which was used to simulate LiDAR in a simulated environment. This method simulates rotating LiDAR, where each point is calculated using a single ray-casting vector for each channel. The rotation is simulated by capturing the horizontal angle at which the LiDAR can rotate in each frame interval. Since this method allows the simulation of lasers for each channel angle, we modified the Carla engine to accommodate channels of the following sensors: VLP-16, VLP-32c, HDL-64e. Each sensor has 16,32 and 64 channels. The Carla simulator calculates points’ intensity using the system’s attenuation and distance to point by the formula 4. Here, a refers to the attenuation of medium and d refers to distance.

$$I = e^{-a \cdot d} \quad (4)$$

6.2 Simulation Results

From the virtual testing environment, LiDAR scans were generated for similar odometers. Figure 16 visually compares Real and LiDAR scans. From visual inspection, the LiDAR scans have similar rings and geometry for ground elements. The simulated LiDAR scans and accurate LiDAR scans are different when simulating trees and materials such as glasses. One key reason is trees are difficult to map with point clouds, and generating a mesh for trees results in poor surface reconstruction. Also, since the approach for simulation only generates mesh and does not consider reflectivity and transparency of materials. Materials such as glasses are regarded as solid in simulation. In Figure 17, we see in the real scan the reflectance of glass produces a point cloud inside the wall, where we see no such outliers in the simulated scans. The trees in real LiDAR scans and Simulated LiDAR scans vary greatly; due to the mesh generation process, the trees act as a solid element in the VTE, whereas in the real world, the leaves scatter the LiDAR scans Figure 19 shows a comparison of trees in both scenarios. In Figure 20 we see LiDAR scan with bushes, the point cloud for the bushes in real world is much more sparse than simulated LiDARs.

6.2.1 Geometric Similarity

The surveyed LiDAR scans were compared with their simulated twin with the same position and orientation in the VTE to set a benchmark score. With our approach of generating a virtual testing environment, the LiDAR scans have an average of 0.76 Density Aware Chamfer Distance and 1.81 Chamfer Distance. This suggests the LiDAR scans have a

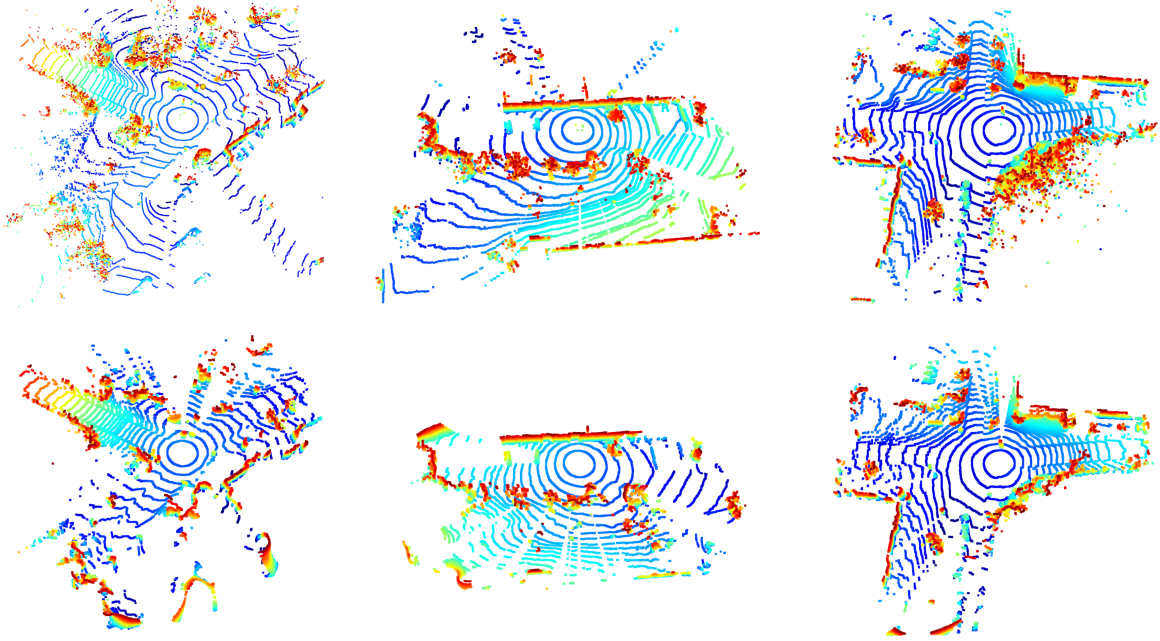


Figure 16: Visual Comparison of Real (Top) and Simulated (Bottom) LiDAR

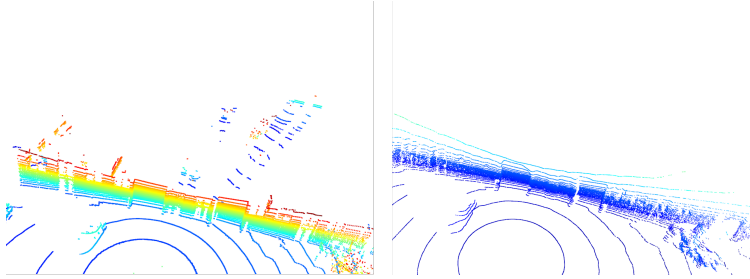


Figure 17: Visual Comparison of Scenarios with Windows. Real Scan (left) and Simulated LiDAR Scan (Right)

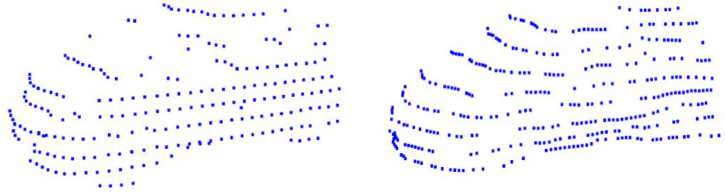


Figure 18: Visual Comparison of Scenarios with Cars. Real Scan (Left) and Simulated LiDAR Scan (Right)

moderate level of geometric similarity. Table 1 contains comparison results for another similarity metric. In Figure 21, we show the distribution of different metrics for the comparison results.

6.2.2 Model Perception Comparison: Original LiDAR scans and Simulation output

To test model perception we compared the output of Sphereformer [52] a LiDAR segmentation model pre-trained on Semantic Kitti Dataset. The model uses a Sparse Convolution Network-based U-Net architecture with transformer heads for semantic segmentation tasks. The model achieves a 74.8% mIoU on the SemKITTI dataset. This model takes

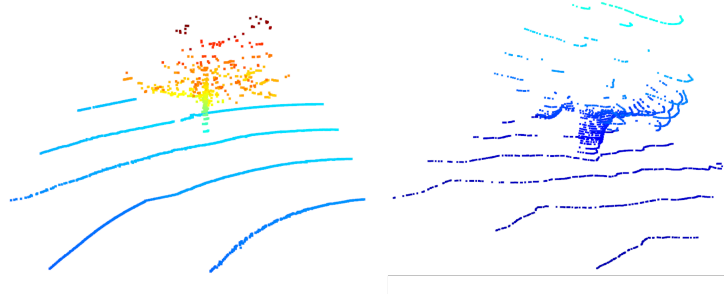


Figure 19: Visual Comparison of Scenarios with Trees. Real Scan (Left) and Simulated LiDAR Scan (Right)

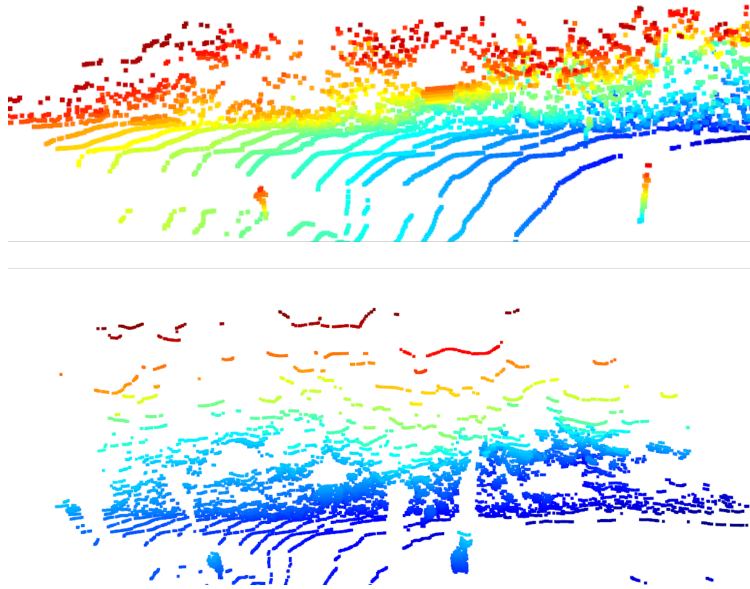


Figure 20: Visual Comparison of Scenarios with trees and bushes. Real Scan (Top) and Simulated LiDAR Scan (Bottom)

a single LiDAR scanned point cloud with intensity and input and generates semantic label data for each point. We use the model as a perception measure to check if real LiDAR scans and simulated LiDAR scans have similar perception results. Ideally, if the simulated LiDAR and original LiDAR are identical, the model should produce similar output. The comparison is by calculating the mean intersection by overlap (mIoU) of prediction output from the model. If both scans are identical, the mIoU should be 1. From the initial test with prediction LiDAR scans generated from the virtual testing environment and the original LiDAR scan of the same odometry, we see an average mIoU of 4.7%. This shows the detection model perceives both scans very differently. Although there is some level of geometric similarity, the intensity of both scans is very different. It can be assumed that if both scans have similar intensity, the model should have better similarity in terms of perception.

6.2.3 Model Perception Comparison: Original LiDAR scans and Simulation output with corrected intensity

To verify the assumption that if both the real and simulated LiDAR scans have similar intensity, the detection will improve. We corrected the simulation intensity by overlaying the intensity of real LiDAR scans on the simulated one. We used the nearest neighbour approach for each point in the simulated LiDAR scan and got the intensity from the closest point in the real LiDAR scan within a threshold of 1 unit.

Table 1: Comparison results with different metrics

Metrics	Mean	STD
Birds Eye View Compare	2.55	0.90
Chamfer Distance	1.81	0.46
Chamfer Distance Voxel Down Sample=1	3.23	0.83
Histogram with RnS 100 points	1.87	0.84
Histogram with RnS 1k points	1.36	0.88
Histogram with RnS 10k points	1.33	0.90
Histogram with VDS 1 unit	0.80	0.44
Histogram with VDS 2 unit	0.73	0.41
Iterative Closest Point	1.42	0.38
Voxel Intersection Over Union	0.04	0.02
Density Aware Chamfer Distance $\alpha = 1k$	0.98	0.01
Density Aware Chamfer Distance $\alpha = 100$	0.93	0.04
Density Aware Chamfer Distance $\alpha = 10$	0.85	0.07
Density Aware Chamfer Distance $\alpha = 1$	0.76	0.08

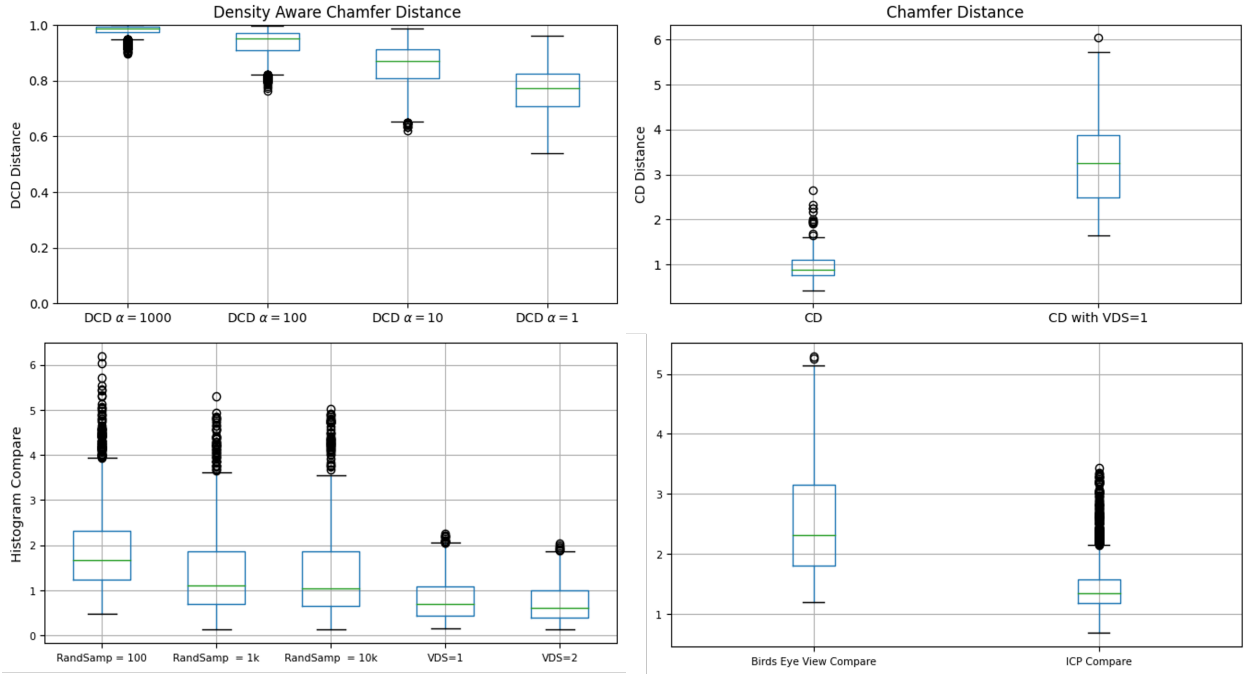


Figure 21: Distribution of different distance metrics when comparing simulated LiDAR and Real LiDAR of the same odometry. Top Left: Density Aware Chamfer Distances. Top Right: Chamfer Distance. Bottom Left; Histogram Methods. Bottom Left: Birds Eye View and ICP compare.

By adopting this approach, we see an improvement in model perception from 4.7% to 21.2%. Figure 23 shows the distribution of mIoU of detection between LiDAR scans from both sources. This result shows that the intensity of LiDAR scans influences model perception outputs. Figure 22 shows the intensity of the real, simulated LiDAR scan and intensity-corrected simulated LiDAR scan of the same odometry. The Figure shows the semantic segmentation output with the Sphereformer model.

7 Conclusion

This paper proposes a method to compare simulated and real LiDAR scans using geometric similarity measures. We made pair-to-pair comparisons across real, augmented, and simulated LiDAR scans of different case studies. From the comparison of the case studies, we find that Density Aware Chamfer Distance is by far the most suitable geometric

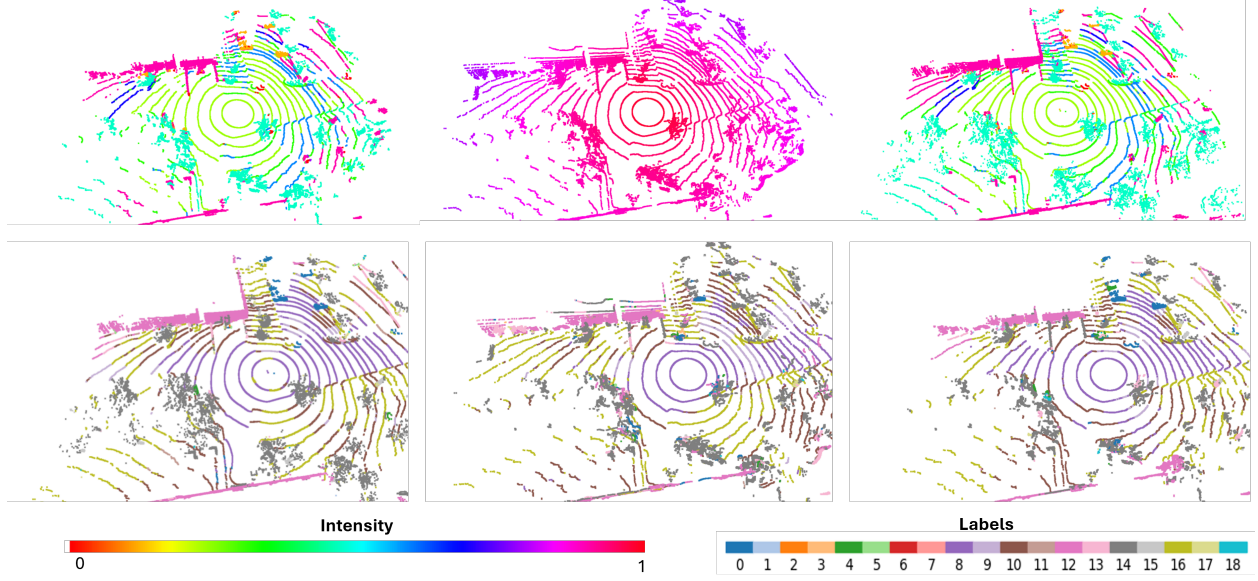


Figure 22: Intensity of LiDAR scans. Top Left: Intensity of Real LiDAR scan. Top Center: Intensity of simulated LiDAR scan. Top Right: Intensity of Simulated Scan with Corrected Intensity. Bottom Left: Segmentation output with Real LiDAR scan. Bottom Center: Segmentation output from simulated LiDAR scan. Segmentation Output from simulated LiDAR with corrected Intensity. The labels are as follows, 0: Car, 1: Bicycle, 2: Motorcycle, 3: Truck, 4: Other-vehicle, 5: Person, 6: Bicyclist, 7: Motorcyclist, 8: Road, 9: Parking, 10: Sidewalk, 11: Other-ground, 12: Building, 13: Fence, 14: Vegetation, 15: Trunk, 16: Terrain, 17: Pole, 18: Traffic-sign

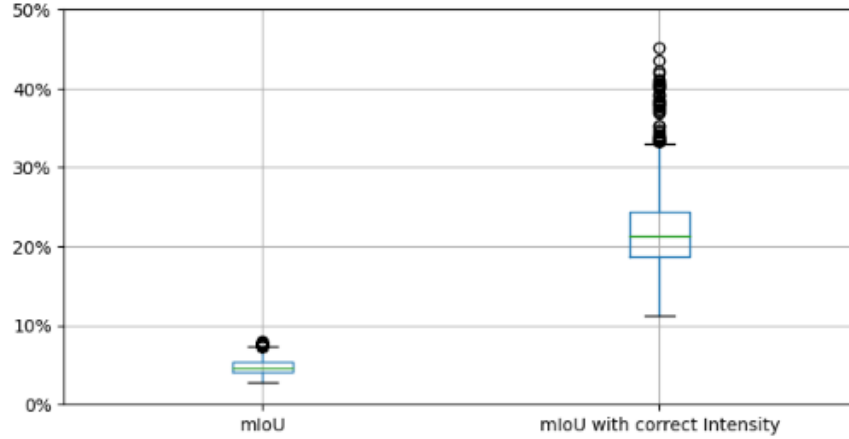


Figure 23: Mean Intersection over Union of prediction from Sphreforformer model for real LiDAR scan and simulated LiDAR scan. The left boxplot shows the result for comparison with simulated intensity. The right box plot shows the result with corrected intensity for simulated scans. Here, the y-axis represents the mIoU values.

comparison method. Because it is sensitive to all modifiers, self-consistent, symmetrical, and most computationally efficient, and has a well-defined range of 0 to 1. Additionally, this metric contains a sensitivity variable α that allows for measuring the slightest differences in LiDAR scans, making it more robust across different scenarios. The sensitivity of the tested scenarios and computational efficiency determine the suitability. Second, we developed a LiDAR scan comparison methodology for comparing real and simulated LiDAR scans. The simulated copy of the LiDAR scans was generated using the CARLA simulator with exact sensor settings and pose. The simulated scans produced acceptable DCD values. Third, we compared the 3D object detection performance of the Sphreforformer model using real, simulated LiDAR scans, and simulated LiDAR scans with corrected intensity. Our findings show that the 3D perception model detects similar intensities for corrected intensities.

Study Limitations. The paper finds Density Aware Chamfer Distance a key similarity metric for comparing LiDAR scan pairs of the same odometry. Generating LiDAR scans in a real-world environment with the same odometry can pose a significant challenge. The metric only compares the geometry of point clouds, ignoring intensity. From the model perception comparison, intensity plays a vital role in model perception. Since our approach only converts 3D point clouds into a mesh, intensity is not considered. LiDAR intensity depends on factors such as medium attenuation, material reflectivity, and sensor settings. Generating a simulator that can accurately simulate LiDAR scans of diverse sensors is challenging. It is also difficult to validate the intensity of LiDAR scans. Another key limitation found in VTE development is that due to meshing, all the objects are not similar in real and simulated scans, such as vegetation reflecting materials. A standard geometric approach would not be the best way to validate these scans. Also, LiDAR scans in snow and rain generate scattered points in specific patterns. Evaluating these scatter patterns is also a challenge with Geometric approaches. Due to the lack of ground truth data, the model perception assessment compares the segmented output of the real and simulated scans. Assessing the accuracy of model perception for both data and ground truth data can open new insights into simulated model perception. Also, due to technical challenges, LiDAR scans with varying channels are only simulated in Carla.

Future Research. With these limitations at hand, our future goal is to find evaluation metrics sensitive to geometry and other LiDAR properties, such as intensities. Digital twins can be developed with an advanced approach, such as using neural kernel fields to generate the mesh. We can also generate ground truth labelled data to better evaluate model perceptions. In future work, we will use 128-channel LiDAR to map real-world environments to generate a more accurate digital twin with advanced mesh generation methods.

References

- [1] Stefan Riedmaier, Thomas Ponn, Dieter Ludwig, Bernhard Schick, and Frank Diermeyer. Survey on scenario-based safety assessment of automated vehicles. *IEEE access*, 8:87456–87477, 2020.
- [2] Joshua L Every, Frank Barickman, John Martin, Sugosh Rao, Scott Schnelle, and Bowen Weng. A novel method to evaluate the safety of highly automated vehicles. In *25th International Technical Conference on the Enhanced Safety of Vehicles (ESV) National Highway Traffic Safety Administration, Detroit, Michigan*, 2017.
- [3] N. Kalra and S. M. Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transp. Res. Part A Policy Pr.*, 94:182–193, 2016.
- [4] Xizhe Zhang, Siddhartha Khastgir, Hamid Asgari, and Paul Jennings. Test framework for automatic test case generation and execution aimed at developing trustworthy avs from both verifiability and certifiability aspects. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 312–319. IEEE, 2021.
- [5] WP UNECE. New assessment/test method for automated driving (natm) master document, united nations, 2021b.
- [6] Myra Blanco, Michelle Chaka, L Stowe, H Clay Gabler, Kenneth Weinstein, Ronald B Gibbons, Luke Neurauter, Joshua McNeil, Kaitlyn E Fitzgerald, Whitney Tatem, et al. Fmvss considerations for vehicles with automated driving systems: Volume 1. Technical report, United States. Department of Transportation. National Highway Traffic Safety . . . , 2020.
- [7] Christof Ebert and Michael Weyrich. Validation of autonomous systems. *IEEE Software*, 36(5):15–23, 2019.
- [8] On-Road Automated Driving (ORAD) Committee. *Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles*. SAE International, 2021.
- [9] Vedant Rane, Hrithik Raj C. Poojari, Prasan Sharma, Soham Phansekar, and Prajakta Pawar. Lidar based self-driving car. *International Journal for Research in Applied Science and Engineering Technology*, 2022.
- [10] Peide Wang. Research on comparison of lidar and camera in autonomous driving. *Journal of Physics: Conference Series*, 2093, 2021.
- [11] Fuquan Zhao, Hao Jiang, and Zongwei Liu. Recent development of automotive lidar technology, industry and trends. In *International Conference on Digital Image Processing*, 2019.
- [12] Mahdi Elhousni and Xinming Huang. A survey on 3d lidar localization for autonomous vehicles. *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1879–1884, 2020.
- [13] Haiyang Wang, Chen Shi, Shaoshuai Shi, Meng Lei, Sen Wang, Di He, Bernt Schiele, and Liwei Wang. Dsvt: Dynamic sparse voxel transformer with rotated sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13520–13529, 2023.
- [14] Zhe Liu, Jinghua Hou, Xinyu Wang, Xiaoqing Ye, Jingdong Wang, Hengshuang Zhao, and Xiang Bai. Lion: Linear group rnn for 3d object detection in point clouds. *arXiv preprint arXiv:2407.18232*, 2024.

- [15] Guangsheng Shi, Ruifeng Li, and Chao Ma. Pillarnet: Real-time and high-performance pillar-based 3d object detection. In *European Conference on Computer Vision*, pages 35–52. Springer, 2022.
- [16] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4840–4851, 2024.
- [17] Bohao Peng, Xiaoyang Wu, Li Jiang, Yukang Chen, Hengshuang Zhao, Zhuotao Tian, and Jiaya Jia. Oa-cnns: Omni-adaptive sparse cnns for 3d semantic segmentation. In *CVPR*, 2024.
- [18] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021.
- [19] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. Sparse convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 806–814, 2015.
- [20] Yavuz Koroglu and Franz Wotawa. Towards a review on simulated adas/ad testing. In *2023 IEEE/ACM International Conference on Automation of Software Test (AST)*, pages 112–122, 2023.
- [21] Bernhard Wymann, Eric Espié, Christophe Guionneau, Christos Dimitrakakis, Rémi Coulom, and Andrew Sumner. Torcs, the open racing car simulator. *Software available at <http://torcs.sourceforge.net>*, 4(6):2, 2000.
- [22] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part II 14*, pages 102–118. Springer, 2016.
- [23] University of Warwick – WMG. Test objective vs simulation fidelity, survey of virtual testing tools. Technical report, University of Warwick – WMG, March 2022. Contract Number T8080200621 Qualification of Virtual Testing Tools for Automated Driving Systems.
- [24] Ece/trans/wp.29/2022/58 — credibility assessment for using virtual toolchain in ads validation. *UNECE’s World Forum for Harmonization of Vehicle Regulations (WP.29), Informal Working Group on Validation Methods for Automated Driving (VMAD)*, 2022.
- [25] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.
- [26] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ international conference on intelligent robots and systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2149–2154. Ieee, 2004.
- [27] Xiangyu Yue, Bichen Wu, Sanjit A Seshia, Kurt Keutzer, and Alberto 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*, pages 458–464, 2018.
- [28] Michael Gschwandtner, Roland Kwitt, Andreas Uhl, and Wolfgang Pree. Blensor: Blender sensor simulation toolbox. In *Advances in Visual Computing: 7th International Symposium, ISVC 2011, Las Vegas, NV, USA, September 26–28, 2011. Proceedings, Part II 7*, pages 199–208. Springer, 2011.
- [29] Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9068–9079, 2018.
- [30] Jin Fang, Dingfu Zhou, Feilong Yan, Tongtong Zhao, Feihu Zhang, Yu Ma, Liang Wang, and Ruigang Yang. Augmented lidar simulator for autonomous driving. *IEEE Robotics and Automation Letters*, 5(2):1931–1938, 2020.
- [31] Sivabalan Manivasagam, Shenlong Wang, Kelvin Wong, Wenyuan Zeng, Mikita Sazanovich, Shuhan Tan, Bin Yang, Wei-Chiu Ma, and Raquel Urtasun. 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, 2020.
- [32] Junge Zhang, Feihu Zhang, Shaochen Kuang, and Li Zhang. Nerf-lidar: Generating realistic lidar point clouds with neural radiance fields. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7178–7186, 2024.
- [33] Benoît Guillard, Sai Vemprala, Jayesh K Gupta, Ondrej Miksik, Vibhav Vineet, Pascal Fua, and Ashish Kapoor. Learning to simulate realistic lidars. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8173–8180. IEEE, 2022.

- [34] Benjamin Attal, Eliot Laidlaw, Aaron Gokaslan, Changil Kim, Christian Richardt, James Tompkin, and Matthew O'Toole. Törf: Time-of-flight radiance fields for dynamic scene view synthesis. *Advances in neural information processing systems*, 34:26289–26301, 2021.
- [35] Eric Heiden, Ziang Liu, Ragesh K Ramachandran, and Gaurav S Sukhatme. Physics-based simulation of continuous-wave lidar for localization, calibration and tracking. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2595–2601. IEEE, 2020.
- [36] Chenqi Li, Yuan Ren, and Bingbing Liu. Pcgcn: Point cloud generator for lidar simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11676–11682. IEEE, 2023.
- [37] Velat Kilic, Deepti Hegde, Vishwanath Sindagi, A Brinton Cooper, Mark A Foster, and Vishal M Patel. Lidar light scattering augmentation (lisa): Physics-based simulation of adverse weather conditions for 3d object detection. *arXiv preprint arXiv:2107.07004*, 2021.
- [38] Jeong-gi Kwak, Youngsaeng Jin, Yuanming Li, Dongsik Yoon, Donghyeon Kim, and Hanseok Ko. Adverse weather image translation with asymmetric and uncertainty-aware gan. *arXiv preprint arXiv:2112.04283*, 2021.
- [39] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.
- [40] Craig Glennie. Rigorous 3d error analysis of kinematic scanning lidar systems. 2007.
- [41] Xu Yan, Chaoda Zheng, Zhen Li, Sheng Wang, and Shuguang Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5589–5598, 2020.
- [42] Francesco Pirotti, Roberta Ravanelli, Francesca Fissore, and Andrea Masiero. Implementation and assessment of two density-based outlier detection methods over large spatial point clouds. *Open Geospatial Data, Software and Standards*, 3(1):14, 2018.
- [43] Timo Hackel, Jan D Wegner, and Konrad Schindler. Fast semantic segmentation of 3d point clouds with strongly varying density. *ISPRS annals of the photogrammetry, remote sensing and spatial information sciences*, 3:177–184, 2016.
- [44] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236*, 2016.
- [45] M Akmal Butt and Petros Maragos. Optimum design of chamfer distance transforms. *IEEE Transactions on Image Processing*, 7(10):1477–1484, 1998.
- [46] Ofir Pele and Michael Werman. Fast and robust earth mover's distances. In *2009 IEEE 12th international conference on computer vision*, pages 460–467. IEEE, 2009.
- [47] Tong Wu, Liang Pan, Junzhe Zhang, Tai Wang, Ziwei Liu, and Dahua Lin. Density-aware chamfer distance as a comprehensive metric for point cloud completion. *arXiv preprint arXiv:2111.12702*, 2021.
- [48] Vladimir Muliukha, Aleksey Lukashin, and Aleksandr Ilyashenko. An intelligent method for comparing shapes of three-dimensional objects. In *2019 25th Conference of Open Innovations Association (FRUCT)*, pages 234–240. IEEE, 2019.
- [49] University of Warwick – WMG. Virtual simulation verification using derived metrics. Technical report, University of Warwick – WMG, March 2022. Contract Number T8080200621 Test Objective vs Simulation Fidelity, Survey of Virtual Testing Tools.
- [50] Albert Wallace, Siddhartha Khastgir, Xizhe Zhang, Simon Brewerton, Benoit Anctil, Peter Burns, Dominique Charlebois, and Paul Jennings. Validating simulation environments for automated driving systems using 3d object comparison metric. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 860–866. IEEE, 2022.
- [51] Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Daniela Rus. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 5135–5142. IEEE, 2020.
- [52] Xin Lai, Yukang Chen, Fanbin Lu, Jianhui Liu, and Jiaya Jia. Spherical transformer for lidar-based 3d recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17545–17555, 2023.