2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Text3DAug - Prompted Instance Augmentation for LiDAR Perception

Laurenz Reichardt[1,*], Luca Uhr[1,*] and Oliver Wasenmüller[1]
[1]Mannheim University of Applied Sciences, Germany
{l.reichardt, l.uhr, o.wasenmueller}@hs-mannheim.de

***Abstract*— LiDAR data of urban scenarios poses unique challenges, such as heterogeneous characteristics and inherent class imbalance. Therefore, large-scale datasets are necessary to apply deep learning methods. Instance augmentation has emerged as an efficient method to increase dataset diversity. However, current methods require the time-consuming curation of 3D models or costly manual data annotation. To overcome these limitations, we propose Text3DAug, a novel approach leveraging generative models for instance augmentation. Text3DAug does not depend on labeled data and is the first of its kind to generate instances and annotations from text. This allows for a fully automated pipeline, eliminating the need for manual effort in practical applications. Additionally, Text3DAug is sensor agnostic and can be applied regardless of the LiDAR sensor used. Comprehensive experimental analysis on LiDAR segmentation, detection and novel class discovery demonstrates that Text3DAug is effective in supplementing existing methods or as a standalone method, performing on par or better than established methods, however while overcoming their specific drawbacks. The code is publicly available.** [1]

I. INTRODUCTION

LiDAR sensors enable the 3D perception of environments and are crucial for applications such as autonomous navigation, robotics, mapping and various industrial applications. While deep learning applications have become the de facto standard for many tasks such as LiDAR detection and segmentation, the data still poses unique challenges.

Firstly, LiDAR data is heterogeneous, with characteristics highly dependend on the sensor. Point cloud structure and distribution varies with the number of scanlines, field of view, rotation frequency, mounting height, etc. This leads to a significant decline in performance, when deep learning methods trained on data from one sensor are applied to data from another sensor. The magnitude of this so called sensor domain gap is unique to 3D point clouds with ongoing research on how to pre-train networks on different datasets or enable multi-dataset training [1], [2].

Secondly, data-imbalance is inherent to LiDAR point clouds, due to multiple factors. In the case of urban scenarios, large objects such as buildings are represented by more points compared to smaller objects or individuals. Due to the radiating alignment of the vertical LiDAR scanlines, point cloud density decreases with increased object distance, meaning that small objects are represented by few or no points beyond a certain distance. This results in adverse effects on network performance for long range perception [3]. This is exasperated by the fact that some objects, especially road participants such as motorcycles, are rare. Such factors result in data-imbalance in large scale datasets. For example, the SemanticKITTI dataset contains building points exceeding those representing people by a factor 709 and for motorcyclist by a factor of $16,205$.

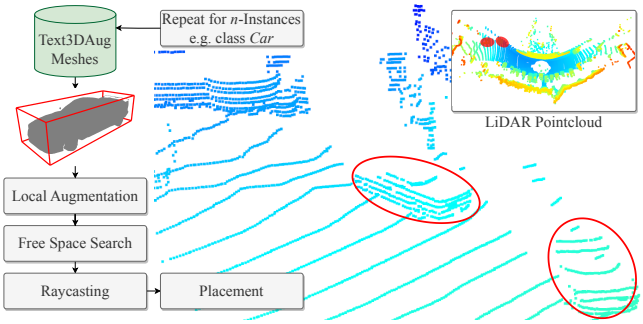

Fig. 1: The Text3DAug augmentation pipeline. We prompt our instance generation engine in order to create and annotate meshes for desired classes. These are then realistically placed and rendered in LiDAR point clouds as instances.

These challenges impose the need for large scale and diverse datasets in order to apply deep learning methods to LiDAR data, in order to obtain sufficient points for all classes. Data augmentation is a standard technique to artificially increase data diversity and in the context of LiDAR scans, instance augmentation has emerged as an effective approach to tackle data-imbalance. Specifically, training data is enriched by "cut and pasting" object instances (e.g. road participants for SemanticKITTI) from different scans. However, the practical application of this concept has extensive requirements. Creating instance cut-outs necessitate semantic and instance labels, however labeling point clouds is significantly more time-consuming when compared to image data due to additional dimensions involved [4].

The labeled data use for cut-out instances also has to exhibit sufficient objects of the desired class, which can be challenging due to data-imbalance, possibly requiring further data collection. Moreover, these instances retain the point structure and remission values specific to their original position, LiDAR sensor and possible occlusion. Additional factors such as the sensor domain gap, different semantic classes, or missing instance labels, mean that objects from other datasets can rarely be employed.

In this work, we tackle the above mentioned limitations, presenting Text3DAug as the first fully-automated and label-

* Equal Contribution
[1]http://github.com/CeMOS-IS/Text3DAug-Augmentation

free instance augmentation method (see Figure 1). We aim to establish Text3DAug as a practical alternative or addition to existing methods. Our method and its contributions can be summarized as follows:

- Text3DAug pioneers the use of generative models, prompting instances for augmentation. We evaluate the effectiveness of our novel pipeline through comprehensive experiments on LiDAR segmentation and detection benchmarks.
- Text3DAug does not require labels or trajectory information. Our instance engine is fully automated and generates a plethora of annotated instances without manual effort. This approach enables the scalability of our method, able to augment with potentially thousands of instances.
- Our method implements realistic placement and rendering of instances according to sensor characteristics. As such, Text3DAug is sensor agnostic, which we evaluate using various datasets.
- As a prompt-based method, Text3DAug is not constrained by dataset classes, and lends itself to the label-free training of new classes. We evaluate this with experiments on novel class discovery.

II. RELATED WORK

The natural imbalance in LiDAR point clouds requires extensive and varied datasets for deep learning. Simulated data has emerged as a viable alternative to real-world data acquisition. Additionally, data augmentation, including the particularly effective instance augmentation, has become a standard method for enhancing data diversity.

A. Data Simulation

Data simulation has emerged as a natural alternative to the time-consuming and expensive process of data acquisition and labeling. Urban simulators such as SYNTHIA [5] and Carla [6] are based on game engines, while others extend existing video games [7], [8]. Simulators allow for data generation under various lighting and weather conditions, with differing dynamic object behaviour and new viewpoints, enabling the collection of diverse data for different sensor modalities. Nonetheless, simulated scenes are created with significant manual effort for 3D asset creation, realistic placement, dynamic animation and rendering. VirtualKITTI [9] and LiDARsim [10] instead use real world LiDAR scans to initialize digital twins, but again rely on manually labeled data in order to transfer object classes and positions.

Despite its apparent benefits, a large domain gap remains between synthetic and real world data, resulting in a significant performance gap [11], [12], [13]. Currently, methods using synthetic data lag behind those using a modest amount of real labeled data and even further behind those using large scale datasets [12]. So called ”sim2real” methods attempt to map real LiDAR characteristics to synthetic data [11], [10], [14] or mix real data into the training process [13], [15]. LiDAR-Aug [16] inserts synthetic CAD models into real LiDAR pointclouds, followed by ray casting. However, LiDAR-Aug requiring the costly curation or manual creation to obtain such models. Moreover, CAD models themselves might vary in quality (poly count and detail) and in factors such as coordinate system definition. For example, the mesh axis of a CAD model does not necessary align with the real center point, varying by object class and standard (e.g. ISO8855 [17] defining the rear axle of a car as the vehicle center), requiring post processing after curation. Lastly, most simulation methods, including LiDAR-Aug do not account for the LiDAR-remission values absent from CAD models.

B. Instance Augmentation

Instance augmentation has been a significant step towards increasing 3D point cloud data diversity, especially of under-represented classes, an aspect crucial for safety critical applications such as autonomous driving. The pioneering work of Yan et al. [18] laid the foundation, creating a database of ”cut and paste” ground-truth instances for integration into LiDAR scans. Zhou et al. [19] extend this concept by oversampling rare class instances and adding local instance transformation in order to maximize data variance. However, ”cut and paste” methods preserve the point distribution and structure specific to the instances original position relative to the LiDAR sensor. Due to the radiating scanlines, the point density of an object decreases with increasing distance to the sensor. Placing an instance closer or further from the sensor, results in a different point density when compared to its surroundings. These issues, in combination with the random placement of instances, leads to unrealistic representations in a LiDAR scan. Subsequent works [20], [21] add some realism by removing points occluded by the added instances, based on the range-view representation of point clouds. However, depending on the grids angular resolution, the range-view representation can lead to data loss.

Real3DAug [22] instead precomputes placement and occlusion maps to identify suitable scene positions, but with significant limitations. These maps are prohibitively time-consuming and computationally expensive. Because of this, the dataset is modified once prior to training, meaning that augmentation remains identical between epochs. Besides trajectories and labelled instances, Real3DAug also requires further semantic labels for its maps, regardless of the LiDAR perception task, as its placement strategy differentiates between different ground types such as road and sidewalks. During placement, instance orientation is computed based on estimated bounding boxes, an approximation since LiDAR data only covers the sensor-facing sides of objects [10].

III. APPROACH

We identify three key observations regarding the current state-of-the-art in LiDAR instance augmentation. Firstly, ”cut and paste” instance augmentation methods [18], [19], [20], [21], [22] depend on the availability of point-wise semantic and instance labels. Secondly, the amount of instances available to these methods is limited by the size and variance of the dataset. Thirdly, obtaining CAD models instead of instance cut-outs can be time consuming and expensive, with

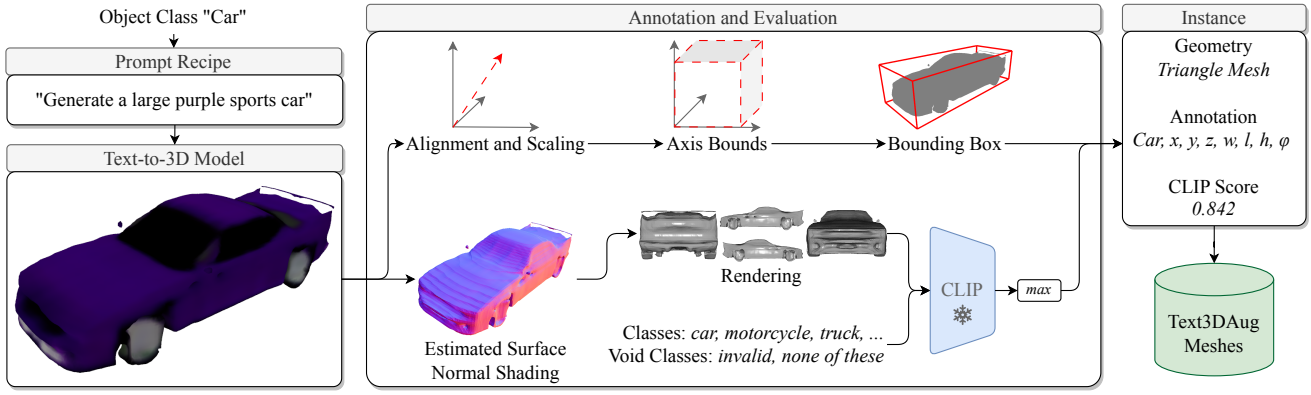

Fig. 2: Our instance generation engine prompts text-to-3D models to generate mesh models. Annotations are derived from the mesh and CLIP scoring is used as a measure of quality. These are added to a database which will be used for the augmentation of LiDAR scans. The shown mesh model in this figure was generated by Shap-E [23].

models potentially requiring post processing and varying in quality.

Thus, we propose Text3DAug, a fully automated and label free instance augmentation pipeline, the first leveraging generative models for 3D content creation. Section III-A describes our standardized prompt recipe which we use to generate object meshes from text-to-3D models. These are then automatically post-processed, evaluated and labeled. Our instance generation engine is described in Section III-B and in Figure 2. For augmentation, meshes are randomly selected, then placed and rendered as instances in LiDAR point clouds using the algorithms described in Section III-C. This process is further elaborated in Figure 1. Text3DAug is designed to be modular, allowing its components to be modified and extended with future research developments.

A. Prompting

Initially we explored the generation of meshes from text-to-3D models [24], [23], [25], [26], [27] using prompts derived from the Q&A output of a Large Language Model (LLM) [28]. However, we observed that LLM generated prompts with multiple attributes such as "A man wearing a hat, walking and holding an umbrella" led to the generation of indistinguishable meshes across a variety of generative models. In contrast, simple prompts such as "a man walking" produced plausible results. Based on this observation and with efficiency in mind, we deemed the use of LLMs excessive and instead designed a fixed prompt recipe. For a desired object class, we define a data store of synonyms and (if applicable) brand-names, e.g. for the class car: *sportscar*, *convertible*, *sedan*, *SUV*, *Ford*, etc. From this, we build the prompt instruction and incorperate attributes like size and color to provide context. Utilizing this recipe, we randomly sample prompts such as "Generate a large purple sports car" for the class car.

Our observation that simple prompts perform better is backed by current research, in that generative models struggle with prompts containing concepts or multiple attributes. Specifically, this issue stems from coarse textual annotations in public datasets, with only a few attributes such as color and size [23], [25], [26], or from drawbacks in the mesh optimization process resulting in imprecise geometries [29], [30], [27]. With increased research and data in the space of generative models for 3D content, more complex prompt strategies will become possible.

B. Instance Generation

By employing the aforementioned prompt recipe, we utilize pre-trained text-to-3D models to generate object meshes for the desired classes. These meshes are added to a database and later sampled for instance augmentation of the LiDAR scans. By pregenerating the meshes, we enable efficient augmentation during training. A notable advantage of text-to-3D models is that meshes are generated with a specific orientation. Contrary to methods such as Real3DAug [22], the orientation does not have to be estimated from partial LiDAR data. We exploit this characteristic to derive precise bounding box annotations. This is achieved by axis aligning and transferring meshes into a common coordinate system, followed by fitting the bounding box based on the axis bounds. Mesh vertices are scaled to a maximum height of one. This later allows for the randomization of instance height during placement, during which the bounding box is transformed accordingly (refer to Section III-C). For the semantic label we assign the object class that was used for prompting.

The mesh quality is automatically evaluated based on the CLIP score [31]. Since color information is not necessary for LiDAR data, we substitute mesh textures with shading derived from estimated surface normals, and then render the mesh from four views (front, back, both sides). The removal of textures encourages CLIP to focus on shape. CLIP similarity is determined for these four views in relation to the desired classes, as well as for the void classes such as *none of these* or *invalid*. The highest score corresponding to the prompted class is assigned as the quality value. This approach is based on the understanding that certain classes have characteristic views that are more representative or identifiable. For example, a person is best recognized from the front, while a car might be best viewed from its side. We evaluate the effectiveness of CLIP for this task in Section IV. Our instance generation engine is depicted in Figure 2.

C. Placement and Local Augmentation

Our Text3DAug pipeline introduces a systematic approach to instance placement in 3D point clouds. For further detail on this procedure we refer to Alg. 1. First, n random object meshes are sampled from the previously created database D, according to the desired classes C, ensuring a diverse representation of object classes. To add additional realism, we process the LiDAR data by mapping remission values according to their range, denoted as R. This approach enables the unsupervised acquisition of realistic remission values. Remission values are assigned to the mesh vertices by sampling random values from R corresponding to the range of each vertex relative to its position during placement.

Each mesh undergoes a random local transformation, which includes height scaling within a class-dependent appropriate range, followed by rotation, in order to reflect the stochastic nature of real-world data. This is followed by a free-space analysis, which transforms a point cloud P and the mesh's vertices into polar coordinates. A random placement distance r of the object to the sensor is chosen, and the relative azimuth span $\Delta\Phi(r)$ is derived from the mesh. Our algorithm filters the polar coordinates of the point cloud P based on $\Delta\Phi$ and the mesh height, delineating viable regions. Then, one of these regions is randomly chosen and the mesh is positioned within it. If no regions are found, the process is repeated with a different object distance. After a suitable region is found, we consider all points of P above and below the mesh and take their minimum z-coordinate z_{min} as the estimated ground level. If no points are found, we also search within the area around the mesh, until z_{min} is determined. The mesh z-positioning is then corrected with z_{min}, anchoring it to the ground. We found this approach to be particularly effective, enabling a seamless integration into urban landscapes, e.g. accurately placing objects on sidewalks or inclined roads. Through this placement algorithm, meshes are inserted into the point cloud respecting its existing structure and spatial constraints. For the last step of realistic placement, we calculate Φ_{min} and Φ_{max} for each vertical scanline (ring) and use these as a limit to remove points in front of the mesh or occluded by it.

After placement, instances are rendered from the meshes using ray casting based on the LiDAR sensor parameters, crucial for simulating an authentic point distribution. Realism is enhanced by introducing noise with the weight w_{noise} and point dropout with probability p_{drop}, emulating imperfections found in real-world data. Annotations L are adjusted accordingly, with the mesh class being assigned as the semantic label. For detection tasks, the bounding boxes of the instances are adjusted based on the instance's position, rotation, and height. The augmented point cloud P_{aug} and the updated annotations L_{aug} are passed on for network training.

Algorithm 1 Instance Augmentation of Text3DAug

Initialization:
  D: Database matching mesh paths to classes
  C: List of dataset classes for augmentation
  R: Remission file
  Augmentation parameters (n, w_{noise}, p_{drop})
  LiDAR sensor parameters

Input:
  P: Point cloud
  L: Annotations ▷ bounding box or semantic label

C_n = Randomly select n classes from C
for class c in C_n **do**
  Select a random mesh from the class c in D
  Add remission to mesh from R
  Random mesh height scaling and rotation
  Find viable regions in P using free-space analysis
  Place mesh and adjust to estimated ground level z_{min}
  Render instance from mesh with ray casting
  Remove points behind or in front of instance
  Point removal with p_{drop} and noise w_{noise}
  Add instance to P
  if Detection **then**
    Update L according to instance placement
  else if Segmentation **then**
    Update L with class c for instance
Return:
  Augmented point cloud P_{aug} with instances
  Updated annotations L_{aug}

IV. EXPERIMENTS

We conduct a series of experiments to assess the capabilities of our prompted Text3DAug pipeline and its components. In Section IV-A and Section IV-B, we evaluate Text3DAug as an instance augmentation method for LiDAR perception, comparing it to relevant methods on the tasks of LiDAR segmentation and detection. The label-free nature of our method lends itself to novel class discovery and we investigate whether text-generated meshes are sufficient for this task in Section IV-C. Our choice of a generative text-to-3D model and CLIP scoring is examined in Section IV-D. Also, we investigate the scalability of our method through a trade-off between mesh quality and quantity in Section IV-F. Finally, placement and local augmentation for realistic LiDAR rendering are assessed in Section IV-E.

For our experiments, we identify eight instance classes (*car*, *person*, *bicycle*, *bicyclist*, *motorcycle*, *motorcyclist*, *truck*, and *bus*) which are shared or can be remapped between the SemanticKITTI [34] and NuScenes semantic segmentation datasets [35], as well as the KITTI [32] and NuScenes [36] detection datasets. We also use these for our evaluation. Unless otherwise stated, we use the original implementations for generative models, segmentation models and augmentation pipelines. Detection experiments are based on the OpenPCDet [37] framework. We use CLIP [31] with ViT-L/14 [38] for mesh evaluation. Training configurations and settings will be made available along with our code release.

TABLE II: Comparison of instance augmentation methods for LiDAR semantic segmentation mIoU.

Method	Type	SemanticKITTI [34]		NuScenes [35]	
		C3D [39]	SPVCNN [40]	C3D [39]	SPVCNN [40]
None (Baseline)	-	61.70	63.72	75.74	69.50
Lidar-Aug* [16]	CAD	63.47	64.40	75.49	69.58
Real3DAug [22]	Labeled	63.25	65.00	-	-
GT-Aug [18]	Labeled	66.07	65.33	75.05	67.96
Ours	Text	64.80	65.18	**75.98**	**69.85**
Ours +GT-Aug	Text + Labeled	**66.94**	**67.52**	74.80	68.78

* Re-implemented

A. Segmentation

We evaluate our method Text3DAug on the SemanticKITTI and NuScenes semantic segmentation datasets using the state-of-the-art networks SPVCNN [40] and Cylinder3D [39]. For all methods we place five instances from 1,000 available meshes per class into each scan, generated by Shap-E [23]. Results are shown in Table II.

In our study on the SemanticKITTI dataset, all instance augmentation methods led to improvements. GT-Aug, a representative "cut and paste" approach with random placement, performs well, but is constrained by a limited pool of instances from the dataset with under-represented classes. Since Text3DAug operates independently of dataset content, its scalability can address this issue. Combining both GT-Aug and Text3DAug yielded the best results, indicating a synergistic effect and further improving model performance. Using SPVCNN, Text3DAug performed almost equal to GT-Aug. The worse results in Table II of Real3DAug can be attributed to the fact, that instance placement is precomputed prior to training, not between epochs. LiDAR-Aug does not consider the remission values of the CAD models, which has a negative effect on network performance (see Section IV-E).

Unexpectedly, GT-Aug was detrimental to network performance on NuScenes. One downside of the "cut and paste" approach is that instances retain properties of their original location. The sensor used for the data acquisition of NuScenes has less scanlines and a different field of view, leading to a significantly reduced point density when compared to the sensor used for KITTI. The combination of this specific sensor and the random instance placement may result in an unrealistic augmentation that negatively impacts network performance. As such, GT-Aug does not generalize well between these two datasets. Generally, improvements were less pronounced compared to those on the SemanticKITTI dataset, which can be attributed to the diversity of the NuScenes dataset. NuScenes has significantly more LiDAR scans and a greater variety in their 1,000 short scenes compared to SemanticKITTIs 22 long scenes. Furthermore NuScenes has ca. 4.4× as many instances [35]. Considering the substantial differences between the datasets, it is logical that instance augmentation has less of an impact.

In summary, Text3DAug leads to improvements over the baseline in all cases, synergizing with GT-Aug for KITTI and leading to the best results. For NuScenes, Text3DAug performs the best among all methods. In general, our standalone method performs on par with or better than established state-of-the-art methods, albeit completely label free.

TABLE III: Evaluation comparing our method Text3DAug to other instance augmentation methods on the KITTI [32] detection task.

	Method	$AP^{70\%}_{car}$			$AP^{50\%}_{pedestrian}$			$AP^{50\%}_{cyclist}$		
		easy	mod.	hard	easy	mod.	hard	easy	mod.	hard
PV-RCNN [41]	None (Baseline)	89.11	79.24	78.58	66.72	60.23	58.47	76.93	57.58	56.55
	Lidar-Aug* [16]	88.89	79.08	78.57	61.36	58.96	56.77	83.92	64.78	58.49
	GT-Aug [18]	89.25	83.27	78.78	64.79	58.32	54.20	86.13	71.96	68.50
	Ours	88.80	78.94	78.29	**67.17**	**60.57**	**58.92**	85.60	65.05	63.71
	Ours+GT-Aug	**89.75**	**83.78**	**78.98**	65.76	59.53	54.79	**93.37**	**73.84**	**70.16**
SECOND [18]	None (Baseline)	86.39	75.50	72.64	52.04	47.20	44.81	65.03	49.61	45.76
	Lidar-Aug* [16]	86.54	76.10	73.06	53.10	47.81	44.83	65.28	51.94	49.13
	GT-Aug [18]	88.06	**78.50**	77.10	**56.40**	**53.12**	**48.33**	82.42	64.04	60.88
	Ours	86.51	75.78	72.75	53.74	50.82	47.32	70.62	57.61	53.20
	Ours+GT-Aug	**88.40**	78.40	**77.13**	54.71	50.12	45.47	**82.44**	**66.26**	**62.03**
PointPillar [33]	None (Baseline)	84.24	72.51	67.35	46.69	43.49	40.47	62.84	44.13	41.88
	Lidar-Aug* [16]	86.33	73.54	67.92	40.06	36.77	35.24	62.52	42.88	40.96
	GT-Aug [18]	86.43	77.24	75.28	**55.79**	**50.94**	**46.66**	**82.56**	**63.68**	**60.79**
	Ours	86.39	73.91	68.49	40.93	37.88	35.59	56.89	43.81	41.65
	Ours+GT-Aug	**87.65**	**77.48**	**75.59**	54.85	49.76	46.35	78.33	61.91	58.92

* Re-implemented

B. Detection

We evaluate 3D object detection performance on the KITTI and NuScenes datasets. We use the detectors PV-RCNN [41], SECOND [18] and PointPillar [33] for KITTI and PointPillar for NuScenes. Following the KITTI benchmark convention, the mean Average Precision (mAP) for pedestrians and cyclists is calculated with a 50% overlap, while for cars it is calculated with a 70% overlap. As for semantic segmentation, the same 1,000 meshes generated by Shap-E [23] are used for each class, with five being randomly inserted per scan.

Table I shows that Text3DAug leads to an improvement over the baseline. Similar to the segmentation results in Table IV-A, Text3DAug best complements GT-Aug on KITTI detection. However, NuScenes results show the same negative effect for GT-Aug, in which the combination of "cut and

TABLE I: Evaluation comparing Text3DAug to other instance augmentation methods on the NuScenes [32] detection task using PointPillars [33]. Classes *trailer*, *construction vehicle*, *traffic cone* and *barrier* were not augmented.

	$mAP^{\%}$	$AP^{\%}_{car}$	$AP^{\%}_{truck}$	$AP^{\%}_{bus}$	$AP^{\%}_{trailer}$	$AP^{\%}_{construct.}$	$AP^{\%}_{pedestrian}$	$AP^{\%}_{motorcycle}$	$AP^{\%}_{bicycle}$	$AP^{\%}_{traffic\ cone}$	$AP^{\%}_{barrier}$
None (Baseline)	49.0	81.8	49.4	58.8	34.1	16.4	74.6	47.5	**23.0**	50.2	53.9
Lidar-Aug* [16]	49.2	82.3	47.1	59.2	**35.8**	16.5	**74.9**	51.0	19.7	51.0	54.2
GT-Aug [18]	44.4	80.9	48.8	**62.4**	35.2	12.5	72.0	31.2	6.2	46.0	48.4
Ours	**49.4**	**82.4**	**49.5**	58.7	34.3	**16.8**	74.3	**51.1**	20.4	**51.6**	**54.8**
Ours+GT-Aug	44.0	81.4	49.0	61.0	34.4	12.2	72.0	30.7	5.2	45.3	48.6

*Re-implemented

TABLE IV: Evaluation of generative models in our instance generation engine. We compare the segmentation mIoU for all 19 training classes and the mIoU for augmented instance classes only on SemanticKITTI [34].

Generative Model		None (Baseline)	Point-E [24]	Shap-E [23]	One-2-3-45 [27] (SDXL-T [42])	Cap3D [25] (Point-E)	Cap3D (Shap-E)	Gpt4Point [26]
All class mIoU		61.71	62.29	**63.93**	62.58	63.53	63.88	63.12
Instances mIoU		59.10	61.46	**64.49**	60.66	62.91	64.15	63.03
Instances CLIP		-	0.585	**0.588**	0.454	0.555	0.575	0.552
Person	IoU	68.33	69.82	67.89	67.12	72.79	73.09	72.29
	CLIP	-	0.355	0.357	0.426	0.436	0.257	0.389
Bicyclist	IoU	86.13	89.25	82.77	85.55	88.00	86.15	84.96
	CLIP	-	0.827	0.825	0.435	0.750	0.779	0.467
Bicycle	IoU	47.79	40.97	45.40	45.47	44.68	45.41	46.84
	CLIP	-	0.451	0.444	0.322	0.467	0.479	0.777
Motorcyclist	IoU	1.51	12.68	37.85	0.03	9.11	37.17	26.35
	CLIP	-	0.268	0.287	0.131	0.395	0.299	0.415
Motorcycle	IoU	63.73	45.25	60.16	51.79	62.79	60.73	62.48
	CLIP	-	0.593	0.611	0.268	0.480	0.560	0.495
Car	IoU	96.06	96.38	94.51	95.78	96.57	95.28	95.38
	CLIP	-	0.813	0.787	0.805	0.776	0.871	0.761
Bus	IoU	44.26	54.74	51.41	52.86	52.16	40.71	49.23
	CLIP	-	0.249	0.254	0.599	0.115	0.288	0.110
Truck	IoU	64.99	85.61	75.93	86.66	77.20	74.66	66.72
	CLIP	-	0.582	0.576	0.385	0.380	0.535	0.372

paste" instances with data of this specific sensor results in an unrealistic augmentation. Here GT-Aug is detrimental to NuScenes detection performance, a problem that does not affect our method. In the case on NuScenes, Text3DAug performs best as a standalone method. The results on both datasets complement previous experiments, highlighting the effectiveness of Text3DAug across sensors and tasks.

C. Novel Class Discovery

Due to its prompt-based nature, Text3DAug lends itself to novel class discovery. We perform experiments for both detection and segmentation, comparing fully-supervised network performance to performance learning a class without any real labeled data. To achieve this, we remove LiDAR points and labels of a specific class in the training split of a dataset and instead placing Text3DAug instances of this class during training.

The evaluation split remains unchanged, with the respective metric serving as the benchmark for class learning. The aim of these experiments is to determine if a network can distinguish a new class in real data using only text prompts in Text3DAug. Cylinder3D [39] is used for SemanticKITTI [34] segmentation, while SECOND [18] is used for KITTI [32] detection. The results are depicted in Tables VI and V. Without any labels, Text3DAug achieves notable results,

TABLE V: Novel class discovery using Text3DAug for KITTI [32] object detection with SECOND [18].

Method	$AP^{50\%}_{pedestrian}$			$AP^{50\%}_{cyclist}$		
	easy	mod	hard	easy	mod	hard
Fully-Supervised [18]	52.04	47.20	44.81	65.03	49.61	45.76
Novel Class Discovery (ours)	28.14	27.07	24.02	12.16	10.50	10.59

TABLE VI: Novel class discovery using Text3DAug for SemanticKITTI [34] segmentation with Cylinder3D [39].

Method	max. class IoU	
	Car	Pedestrian
Fully-Supervised [39]	96.09	75.71
Novel Class Discovery (ours)	49.00	16.80

showing that new classes can be learned just from text. These results are demonstrate the significant potential of Text3DAug, especially when compared to the baseline using fully-labeled data. This opens future paths of research, integrating Text3DAug into label-efficient or unsupervised methods. A downside of removing existing classes in this evaluation, is that empty regions are left in the LiDAR scan, potentially leading to worse results.

D. Instance Generation

We examine the influence of various state-of-the-art mesh generation models on the performance of Text3DAug. To ensure a fair comparison, we use identical prompts and generate 250 meshes per class (without CLIP filtering). We choose LiDAR semantic segmentation as the measure of effectiveness, as fine grained point-level classification is required. Experiments are conducted on the SemanticKITTI dataset using Cylinder3D [39] for segmentation. We add five instances using Text3DAug during training, which are randomly chosen from the eight instance classes (*car*, *person*, *bicycle*, *bicyclist*, *motorcycle*, *motorcyclist*, *truck*, and *bus*).

For Point-E [24], Shap-E [23], Cap3D [25] and GPT4Point [26] we generate the out-of-domain classes *motorcyclist* and *bicyclist* by combining the meshes of *person* with those of *motorcycle* or *bicycle* together. This limitation to these classes does not affect One-2-3-45 [27] and as such we include both classes in the evaluation. We determined a failure rate of 23.2% for One-2-3-45 stemming from an unstable optimization process, in such cases requiring re-prompting with new prompts. Results are listed in Table IV.

Independent of the generative model used, our method improves performance over the baseline, with Shap-E showing the best results. Notably, combining *person* with *motorcycle* meshes to create the out-of-domain class *motorcyclist* proved effective. Furthermore, no single generative model excels across all classes; rather, considerable variation remains

in which model performs best for a specific class. For the classes *bicycle* and *motorcycle*, no improvement was observed compared to the baseline. This variability and existence of out-of-domain classes indicates the potential remaining in the field of 3D content generation. As our Text3DAug is not reliant on one single generative model, we hypothesize that Text3DAug will synergize well with future research as such issues are resolved.

We also evaluate the average CLIP score of all meshes in comparison to the resulting performance. There is a moderate positive correlation of $R = 0.69$ between the CLIP score and resulting mIoU. Although the sample size of six models may be too small to establish a reliable correlation value, it does suggest a discernible trend: the highest CLIP score aligns with the best performing model measured by mIoU.

E. Placement and Local Augmentation

The experiments in this section cover the design choices for the realistic placement and rendering of meshes in LiDAR scans as instances (described in Section III-C). For this we use Cylinder3D [39] for LiDAR segmentation on SemanticKITTI [34] with 250 random meshes per class for Text3DAug.

TABLE VII: Evaluation of the placement algorithm and remission sampling for our Text3DAug.

Placement	Remission	mIoU
Random	✓	63.62
Realistic	✓	**63.93**
Realistic		62.49

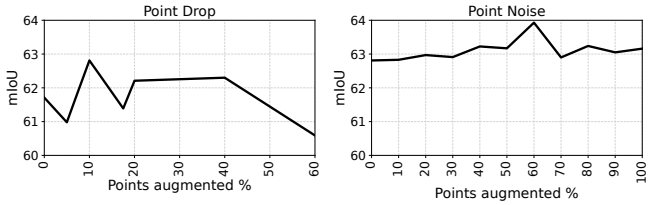

Fig. 3: Impact of removing points and adding noise on Text3DAug instances on SemanticKITTI [34] semantic segmentation performance.

First, we evaluate our choice of point drop and refer to Figure 3. We find that removing points with a probability of 10% from added instances performs best. Using this value, we then assess point noise, determining that affecting 60% of points with noise further improves results. Then, we evaluate our placement algorithm and the addition of remission values to meshes. We find that randomly placing instances performs worse and that remission values are critical for a network. Results are depicted in Table VII, with the combination of realistic placement and remission values performing best.

F. Quality vs. Quantity

In this section, we investigate whether quality or quantity of meshes is more crucial for Text3DAug. For this, we extend the Shap-E database of meshes to 1,000 meshes per class. We then filter available meshes using the CLIP score as a measure of quality, selecting the top 250, 500 and 750 meshes for random sampling, in contrast to using all 1,000. Experiments

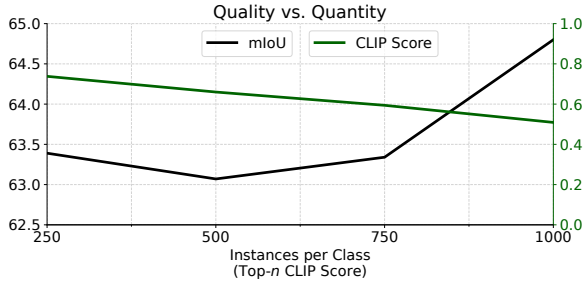

Fig. 4: We assess the impact of quality versus quantity on our pipeline, filtering the meshes by CLIP score and comparing results for the segmentation mIoU on SemanticKITTI [34].

are conducted using Cylinder3D [39] on SemanticKITTI. We add five instances using Text3DAug during training. Results are plotted in Figure 4, showing that the amount of available meshes for instance placement has a greater impact than the hypothetical quality based on the CLIP score. This showcases the benefit of scalability provided by our method. Notably, with 250 meshes filtered by CLIP score, Text3DAug performs slightly worse compared to the 250 random meshes from Section IV-D. These results contrast the moderate correlation of CLIP / mIoU from Section IV-D, leading us to the conclusion that a diversity of meshes is most important for resulting model performance. Low quality meshes possibly have a regularizing effect, in turn leading to better data variance.

V. CONCLUSION

In this work, we propose Text3DAug, exploiting the recent advent of generative text-to-3D models for LiDAR instance augmentation. Current instance augmentation methods rely on large datasets, labels or manual effort. Tackling these limitation, we design an automatic engine to generate meshes and annotations from text prompts. These are then realistically placed into LiDAR scans and rendered as instances according to the sensor characteristics. The number of instances available to our method is not limited by dataset size, offering a practical and scalable approach. We show the effectiveness of Text3DAug through comprehensive evaluation on the tasks of LiDAR semantic segmentation and detection, outperforming or performing on-par with comparable methods, however without their drawbacks. We further experiment with Text3DAug for novel class discovery, demonstrating the capability to learn new classes solely from text, potentially enhancing label-efficient or unsupervised research. In future work, our modular pipeline can accommodate new generative networks or prompting techniques, developing with progress in the state of the art. Furthermore, we see potential in expanding Text3DAug to different sensor modalities such as radar [43] and various methods of depth sensing.

ACKNOWLEDGMENT

This work was funded by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) under the grant AuReSi (KK5335501JR1) and SERiS (KK5335502LB3).

REFERENCES

[1] H. Zhu, H. Yang, X. Wu, D. Huang, S. Zhang, X. He, T. He, H. Zhao, C. Shen, Y. Qiao, *et al.*, "Ponderv2: Pave the way for 3d foundataion model with a universal pre-training paradigm," *arXiv preprint arXiv:2310.08586*, 2023.

[2] X. Wu, Z. Tian, X. Wen, B. Peng, X. Liu, K. Yu, and H. Zhao, "Towards large-scale 3d representation learning with multi-dataset point prompt training," in *Computer Vision and Pattern Recognition (CVPR)*, 2024.

[3] M. Fürst, O. Wasenmüller, and D. Stricker, "Lrpd: Long range 3d pedestrian detection leveraging specific strengths of lidar and rgb," in *International Conference on Intelligent Transportation Systems (ITSC)*, 2020.

[4] D. Stumpf, S. Krauß, G. Reis, O. Wasenmüller, and D. Stricker, "Salt: A semi-automatic labeling tool for rgb-d video sequences," *arXiv preprint arXiv:2102.10820*, 2021.

[5] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[6] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on Robot Learning (CoRL)*, 2017.

[7] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *European Conference on Computer Vision (ECCV)*, 2016.

[8] X. Yue, B. Wu, S. A. Seshia, K. Keutzer, and A. L. Sangiovanni-Vincentelli, "A lidar point cloud generator: From a virtual world to autonomous driving," in *International Conference on Multimedia Retrieval (ICMR)*, 2018.

[9] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[10] S. Manivasagam, S. Wang, K. Wong, W. Zeng, M. Sazanovich, S. Tan, B. Yang, W.-C. Ma, and R. Urtasun, "Lidarsim: Realistic lidar simulation by leveraging the real world," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[11] A. Xiao, J. Huang, D. Guan, F. Zhan, and S. Lu, "Transfer learning from synthetic to real lidar point cloud for semantic segmentation," in *AAAI Conference on Artificial Intelligence*, 2022.

[12] L. Reichardt, N. Ebert, and O. Wasenmüller, "360° from a single camera: A few-shot approach for lidar segmentation," in *International Conference on Computer Vision Workshops (ICCVW)*, 2023.

[13] C. Saltori, F. Galasso, G. Fiameni, N. Sebe, E. Ricci, and F. Poiesi, "Cosmix: Compositional semantic mix for domain adaptation in 3d lidar segmentation," in *European Conference on Computer Vision (ECCV)*, 2022.

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

[15] C. Saltori, E. Krivosheev, S. Lathuiliére, N. Sebe, F. Galasso, G. Fiameni, E. Ricci, and F. Poiesi, "Gipso: Geometrically informed propagation for online adaptation in 3d lidar segmentation," in *European Conference on Computer Vision (ECCV)*, 2022.

[16] J. Fang, X. Zuo, D. Zhou, S. Jin, S. Wang, and L. Zhang, "Lidar-aug: A general rendering-based augmentation framework for 3d object detection," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[17] I. S. 33, "Iso8855:2011 road vehicles - vehicle dynamics and road-holding ability vocabulary," International Organization for Standardization, Geneva, CH, Standard, 2011.

[18] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, 2018.

[19] Z. Zhou, Y. Zhang, and H. Foroosh, "Panoptic-polarnet: Proposal-free lidar point cloud panoptic segmentation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[20] F. Hasecke, M. Alsfasser, and A. Kummert, "What can be seen is what you get: Structure aware point cloud augmentation," in *Intelligent Vehicles Symposium (IV)*, 2022.

[21] L. Kong, Y. Liu, R. Chen, Y. Ma, X. Zhu, Y. Li, Y. Hou, Y. Qiao, and Z. Liu, "Rethinking range view representation for lidar segmentation," in *International Conference on Computer Vision (ICCV)*, 2023.

[22] P. Šebek, Š. Pokorný, P. Vacek, and T. Svoboda, "Real3d-aug: Point cloud augmentation by placing real objects with occlusion handling for 3d detection and segmentation," *arXiv preprint arXiv:2206.07634*, 2022.

[23] H. Jun and A. Nichol, "Shap-e: Generating conditional 3d implicit functions," *arXiv preprint arXiv:2305.02463*, 2023.

[24] A. Nichol, H. Jun, P. Dhariwal, P. Mishkin, and M. Chen, "Point-e: A system for generating 3d point clouds from complex prompts," *arXiv preprint arXiv:2212.08751*, 2022.

[25] T. Luo, C. Rockwell, H. Lee, and J. Johnson, "Scalable 3d captioning with pretrained models," *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[26] Z. Qi, Y. Fang, Z. Sun, X. Wu, T. Wu, J. Wang, D. Lin, and H. Zhao, "Gpt4point: A unified framework for point-language understanding and generation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[27] M. Liu, C. Xu, H. Jin, L. Chen, M. Varma T, Z. Xu, and H. Su, "One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization," *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[28] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, *et al.*, "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[29] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, "Dreamfusion: Text-to-3d using 2d diffusion," *arXiv preprint arXiv:2209.14988*, 2022.

[30] L. Melas-Kyriazi, I. Laina, C. Rupprecht, and A. Vedaldi, "Realfusion: 360deg reconstruction of any object from a single image," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[31] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning (ICML)*, 2021.

[32] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[33] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[34] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences," in *International Conference on Computer Vision (ICCV)*, 2019.

[35] W. K. Fong, R. Mohan, J. V. Hurtado, L. Zhou, H. Caesar, O. Beijbom, and A. Valada, "Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking," *Robotics and Automation Letters (RA-L)*, 2022.

[36] 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 *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[37] O. D. Team, "Openpcdet: An open-source toolbox for 3d object detection from point clouds," https://github.com/open-mmlab/OpenPCDet, 2020.

[38] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *ICLR*, 2021.

[39] X. Zhu, H. Zhou, T. Wang, F. Hong, Y. Ma, W. Li, H. Li, and D. Lin, "Cylindrical and asymmetrical 3d convolution networks for lidar segmentation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[40] H. Tang, Z. Liu, S. Zhao, Y. Lin, J. Lin, H. Wang, and S. Han, "Searching efficient 3d architectures with sparse point-voxel convolution," in *European Conference on Computer Vision (ECCV)*, 2020.

[41] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[42] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," in *International Conference on Learning Representations (ICLR)*, 2024.

[43] A. Musiat, L. Reichardt, M. Schulze, and O. Wasenmüller, "Radarpillars: Efficient object detection from 4d radar point clouds," in *International Conference on Intelligent Transportation Systems (ITSC)*, 2024.