Outdoor Scene Extrapolation with Hierarchical Generative Cellular Automata

Dongsu Zhang[1 *] Francis Williams[2] Zan Gojcic[2] Karsten Kreis[2]
Sanja Fidler[2,3,4] Young Min Kim[1] Amlan Kar[2, 3, 4]
[1]Seoul National University [2]NVIDIA [3]Vector Institute [4]University of Toronto

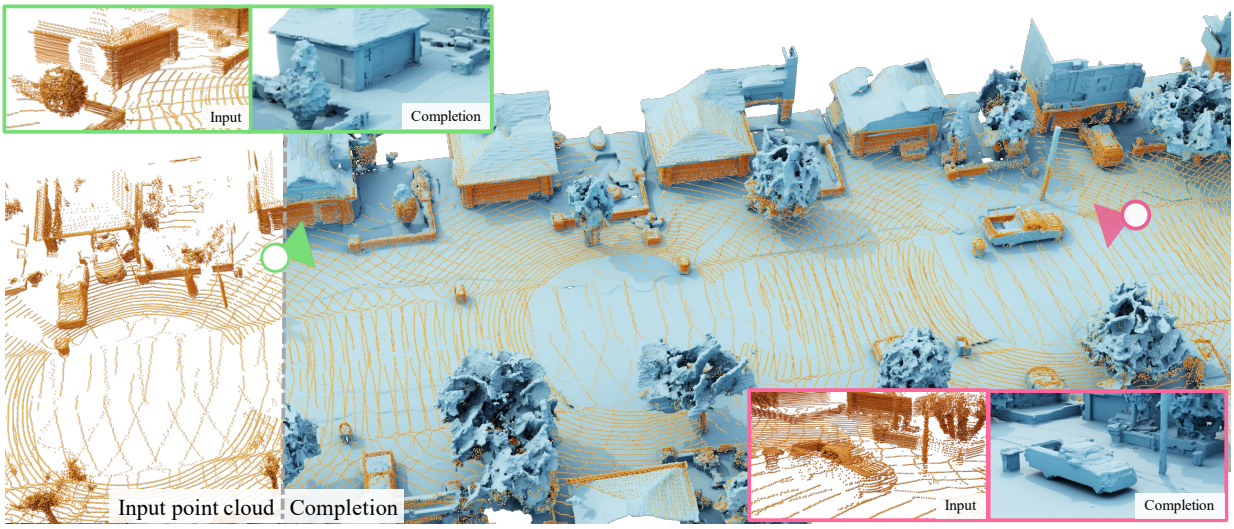


Figure 1. Geometry generation from hGCA (blue) from five accumulated LiDAR scans (yellow spheres) on real-world Waymo-open dataset. hGCA is a conditional 3D generative model that can generate geometry beyond occlusions (vehicles, facades) and input field of view (roofs, trees, poles), from sparse and noisy LiDAR scans. Our method is also spatially scalable, completing this whole scene (120 meters) at high resolution on a single 24GB GPU without additional tricks.

Abstract

We aim to generate fine-grained 3D geometry from large-scale sparse LiDAR scans, abundantly captured by autonomous vehicles (AV). Contrary to prior work on AV scene completion, we aim to extrapolate fine geometry from unlabeled and beyond spatial limits of LiDAR scans, taking a step towards generating realistic, high-resolution simulation-ready 3D street environments. We propose hierarchical Generative Cellular Automata (hGCA), a spatially scalable conditional 3D generative model, which grows geometry recursively with local kernels following [46, 47], in a coarse-to-fine manner, equipped with a light-weight planner to induce global consistency. Experiments on synthetic scenes show that hGCA generates plausible scene geometry with higher fidelity and completeness compared to state-of-the-art baselines. Our model generalizes strongly from sim-to-real, qualitatively outperforming baselines on the Waymo-open dataset. We also show anecdotal evidence of the ability to create novel objects from real-world geometric cues even when trained on limited synthetic content. More results and details can be found on our project page.

1. Introduction

How can we scalably build large-scale, diverse and realistic digital worlds for applications in simulation for autonomous vehicles (AV) or gaming and entertainment? Manually authoring a realistic scene requires significant effort in creating individual objects and positioning them in realistic spatial configurations. Procedural models are a promising alternative which back recent AAA games such as No Man's Sky. However, authoring procedural models of objects and environments are usually time consuming manual tasks. Densely scanning the world is now an increasingly popular and more scalable option, using Neural Radiance Field (NeRF) based approaches. However, these reconstruction methods typically don't capture content beyond what is observed. Sparse LiDAR scans from autonomous vehicles – a by-product of their development and deployment – also provide cues to the geometry of street environments in the world. Our work aims to use these sparse LiDAR scans as input to a

[*] Work started during Dongsu's internship at NVIDIA

conditional 3D generative model that learns to extrapolate plausible high-resolution scene geometry.

Prior work in the domain has focused on semantic scene completion (SSC) from a single LiDAR scan [40, 42], using accumulated sequential LiDAR scans with labeled semantic classes as supervision [2]. This is useful for AV perception to learn to expect 3D semantic occupancy beyond instantaneous observations. However, using such accumulated scans as supervision typically results in outputs unsuitable for simulation, since they have low-resolution geometry and suffer from heavy occlusions, exacerbated by scans being taken from a single drive through a dynamic scene [2, 34]. Moreover, typical LiDAR scanners in AV have a restricted height range which prohibits learning to generate scene geometry beyond this limit in SSC. From sparse LiDAR scans, we instead aim to generate high-resolution scene geometry and go beyond the LiDAR range (Fig. 1), to take a step towards simulation ready scene geometry. To differentiate from the task of semantic scene completion (SSC), we name our task outdoor scene extrapolation. However, for ease of expression, we use the terms completion, generation, and extrapolation interchangeably through the rest of the paper. We train and evaluate on synthetic scenes which allow fine and complete geometric supervision while maintaining the ability to complete geometry from real LiDAR scans. We use a conditional 3D generative model, which is more suited to this challenging inverse problem, as opposed to prior SSC methods that typically use discriminative autoencoder.

We propose a spatially scalable 3D generative model of geometry, with a two-stage hierarchical coarse-to-fine formulation, called hGCA. hGCA builds on top of the recent Generative Cellular Automata (GCA) framework [46, 47], which is a 3D generative model that recursively applies local kernels to incrementally grow geometry from a sparse set of active cells. GCA was shown to perform competitively with state-of-the-art for geometry completion from dense indoor scans. The sparsity and locality of GCA allows spatial scalability. However, we find that naively applying GCA for fine geometry extrapolation on large outdoor scenes from sparse LiDAR leads to performance deterioration stemming from a lack of global context and the need to use a large number of recursive steps, the latter motivating our coarse-to-fine approach. To introduce global context, hGCA's coarse stage uses a GCA conditioned on features from a light-weight bird's eye view *planner* to generate scene geometry in a low-resolution voxel grid, without losing spatial scalability. The second stage synthesizes finer details with cGCA [47], generating high resolution voxels augmented with local implicit functions that allow promoting the output to a 3D mesh.

We train on synthetic street scenes, using data from the CARLA simulator [13], and a city asset from Turbosquid, using simulated LiDAR scans as input. On synthetic scenes, hGCA outperforms state-of-the-art SSC and indoor scene completion methods on multiple metrics for geometry extrapolation. Quantitatively evaluating 3D generative models in the real world is challenging. Qualitatively, we observe that hGCA shows strong sim-to-real generalization to real LiDAR scans compared to prior work, generating more complete and higher fidelity geometry, demonstrated on the Waymo-open dataset [34]. We also demonstrate with examples that despite being trained on limited synthetic content, hGCA can generate some novel content beyond its training data, by taking geometric cues from input LiDAR scans.

2. Related Work

3D Shape Completion. Earlier works [9, 44] on data-driven 3D shape completion regressed a single shape from partial 3D observation using deep neural networks. [17] learn to complete 3D shapes using partial geometry supervision coming from LiDAR scans. Multiple works [10, 11, 32] tackle completion of indoor scenes from dense RGB-D scans. [10, 11] proposed a hierarchical coarse-to-fine approach for fine-grained completion of indoor scenes. Recent works [5, 20, 33] have employed deep implicit fields to learn to generate continuous surfaces [7, 22, 23, 26]. We take inspiration from these using a coarse-to-fine approach and local implicit functions at the finest level. Most prior works on outdoor scene completion focus on semantic scene completion (SSC) [2, 8, 25, 40, 42] for autonomous vehicles (AV), i.e. completing semantic voxel occupancy given a single LiDAR scan using accumulated sequential point clouds with semantic labels for supervision. JS3CNet [42] proposes a novel point-voxel interaction module for better feature extraction and SCPNet [40] utilizes student-teacher distillation from a multi-frame input teacher, and improves network design without any dowsampling modules. While the works show suitable results for AV perception, the methods produce low resolution geometry and suffer from occlusions arising from supervision, deficient for simulation. We show superior performance to both indoor scene completion and SSC methods for AV scene geometry extrapolation on synthetic data, as well as qualitatively improved sim-to-real generalization on the Waymo-open [34] dataset.

3D Generative models [4, 15, 18, 37, 43, 45], have typically focused on synthesizing single objects, leveraging GANs [1, 38], and diffusion based generative models [45, 48]. Recent methods in text-to-3D generative models [28], have shown impressive results in generating novel shapes and small scenes. [41, 49] learn generative models of LiDAR scans demonstrating scene level point cloud synthesis in autonomous driving. Inspired by the cellular automaton [36], the GCA [46, 47] framework can generate multimodal completions for both objects and indoor scenes. GCA scales to scenes as it recursively applies local kernels to grow a sparse set of active cells in its generative process. This resembles diffusion models [3, 19, 31] where samples

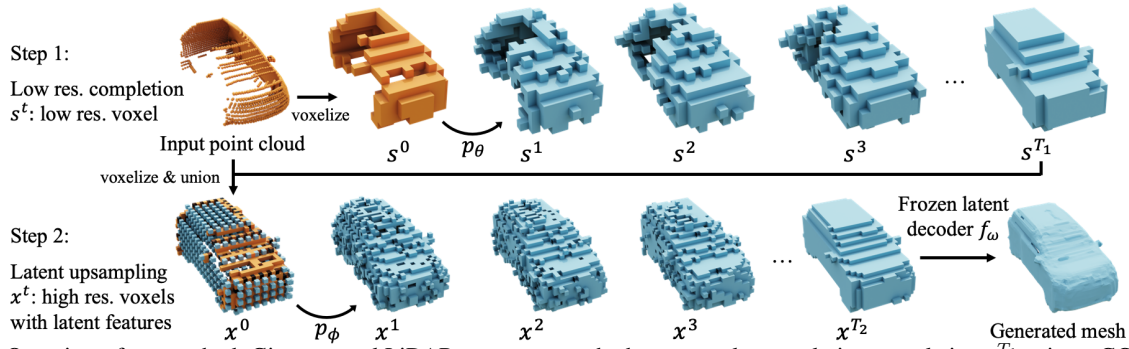


Figure 2. Overview of our method. Given several LiDAR scans, our method generates low resolution completion s^{T_1} using a GCA attached with a planner that adds global consistency. Then given s^{T_1} and the input, we upsample the completion using a cGCA into high resolution voxel with a local latent x^{T_2} and decode it to obtain the final generated mesh.

are generated with a recursive learned denoising kernel, and Neural Cellular Automata [24]. We find that the locality of GCA fails to capture global context, generating artifacts in large scenes. hGCA extends GCA to capture global context and efficiently generate fine geometry.

3. Hierarchical Generative Cellular Automata

Given LiDAR scans captured from an ego-vehicle, the task is to generate complete scene geometry, including regions beyond the LiDAR range or away from the street. To efficiently handle expansive scales of outdoor scenes with fine detail (Fig. 1), hierarchical Generative Cellular Automata (hGCA) proposes a conditional generative model in a two-step, coarse-to-fine manner as shown in Fig. 2. The first step of hGCA extrapolates the scene in a low resolution voxel representation using a model based on Generative Cellular Automata (GCA) [46]; a sparse, local and hence spatially scalable generative model, which we briefly introduce in Sec. 3.1 for completeness. However, the local generation of GCA can introduce artifacts in extrapolating larges scenes beyond sensor measurements. We propose to induce global context into GCA by jointly training a light-weight bird's eye view encoder, called *planner* (Sec. 3.2). Then we transform the coarse geometry into high-resolution continuous scene geometry using local implicit functions [47] (Sec. 3.3). Together, the proposed method can create large outdoor scenes with spatial scalability, global consistency, higher fidelity from sparse, partial real-world scans.

3.1. Background: Generative Cellular Automata

Generative process. GCA recursively grows an incomplete shape to completion as illustrated in step 1 of Fig. 2, by locally updating occupancies around the current shape. GCA represents shapes as sparse voxel occupancies, $s = \{(c, o_c) | c \in \mathbb{Z}^3, o_c \in \{0, 1\}\}$, where o_c indicates binary occupancy of a voxel / cell with its coordinates c. In the following text, we use voxel and cell interchangeably. Given an observed, incomplete state s^0, it generates a completed state s^T by recursively sampling $s^{1:T}$:

$$s^{t+1} \sim p_\theta(\cdot | s^t), \tag{1}$$

where T is a predefined number of transition steps and p_θ is a local transition kernel with parameters θ. The transition kernel uses a U-Net [30] architecture using sparse convolutions [16] *i.e.*, the convolution only processes occupied cells for efficiency. The transition kernel p_θ is computed *locally* on the neighborhood of the occupied cells, $\mathcal{N}(s^t) = \{c' \in \mathbb{Z}^3 \mid d(c, c') \leq r, o_c = 1, c \in \mathbb{Z}^3\}$, *i.e.*, cells within a radius r from current occupied cells under a distance metric d. For efficient sampling, the transition kernel is computed for each cell in $\mathcal{N}(s^t)$ independently,

$$p(s^{t+1} | s^t) = \prod_{c \in \mathcal{N}(s^t)} p_\theta(o_c | s^t), \tag{2}$$

$$p_\theta(o_c | s^t) = Ber(\lambda_{\theta,c}), \tag{3}$$

where $p_\theta(o_c | s^t)$ is a Bernoulli variable with mean $\lambda_{\theta,c}$ estimated by the neural network for cell c, given s_t. This sparse and local generative process of GCA allows more spatial scalability over traditional encoder-decoder methods that process whole scenes at once. In this work, we use a variant of GCA where the transition kernel p_θ is conditioned on both the initial state s^0 and the current state s^t, improving conditioning on the input s^0 [47].

Training GCA. GCA is trained with infusion training [3] to converge to a desired shape s^{gt}, given s^0 and s^{gt}. Infusion training supervises the transition kernel $p_\theta(s^{t+1} | s^t)$ at each step, by defining an infusion kernel

$$q_\theta^t(\tilde{s}^{t+1} | \tilde{s}^t, s^{\text{gt}}) = \prod_{c \in \mathcal{N}(\tilde{s}^t)} q_\theta^t(o_c | \tilde{s}^t, s^{\text{gt}}) \tag{4}$$

where the infusion kernel $q_\theta^t(\tilde{s}^{t+1} | \tilde{s}^t, s^{\text{gt}})$ is factorized per cell as in Eq. 2. The infusion kernel for a single cell c,

$$q_\theta^t(o_c | \tilde{s}^t, s^{\text{gt}}) = Ber((1 - \alpha^t)\lambda_{\theta,c} + \alpha^t \mathbb{1}[c \in s^{\text{gt}}]) \tag{5}$$

is a Bernoulli variable with its mean defined as the estimated occupancy probability $\lambda_{\theta,c}$ *infused* with target s^{gt} with weight $\alpha^t = \alpha_1 t + \alpha_0 \mid \alpha^t \in [0, 1]$, which increases linearly with t.

For training input s^0 to generate $s^{\text{gt}} \sim s^T$, intermediate infusion states $\tilde{s}^{1:T}$ are sampled from the infusion kernel $q_\theta(\tilde{s}^{t+1} | \tilde{s}^t, s^{\text{gt}})$ recursively. For each sampled infusion state $\tilde{s}^t$, GCA is trained with binary cross entropy loss against the

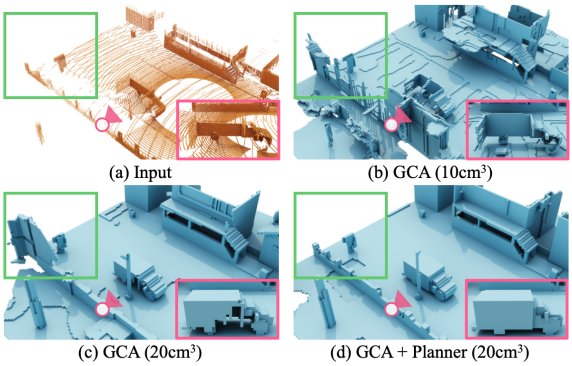


Figure 3. Left: (a) Input LiDAR scans. (b), (c) GCA completion in 10cm^3 and 20cm^3 voxel resolution. (d) GCA + planner completion in 20cm voxel resolution. GCA is local and often cannot capture the global context, generating imperfect completions (pink box) or artifacts (green box).

ground-truth s^{gt} on the neighborhood $\mathcal{N}(\tilde{s}^t)$ by minimizing:

$$\mathcal{L}_{\text{GCA}} = -\sum_{c\in\mathcal{N}(\tilde{s}^t)}\sum_{o_c\in\{0,1\}} \mathbb{1}[o_c = o_{c,s^{\text{gt}}}] \log p_\theta(o_c|\tilde{s}^t), \tag{6}$$

where $o_{c,s^{\text{gt}}} \in \{0,1\}$ is the occupancy of cell c for ground truth shape s^{gt}. We refer to [47] for theoretical foundation of the loss function.

3.2. Planner

While GCA has been shown to complete small indoor scenes, we find that it lacks global consistency on extrapolating large-scale scenes. Take the fence in green box in left of Fig. 3 for instance. The walls of buildings are generated inconsistently by GCA, at both low and high resolution, showing symptoms of lack of global consistency. This issue is exacerbated in sim-to-real inference shown in Fig. 7. We hypothesize that while GCA's sparse and recursive kernel brings scalability, it cannot maintain global context both spatially and temporally. Spatially, the sparse convolutions deliver information only through occupied cells, making it difficult to observe wide spatial context without immediate connection. Moreover, the Markov transition kernel transmits no other memory except binary occupancy between transitions, thus inhibiting long-range "planning" across transition steps.

Hence, we introduce a light-weight *planner* module that provides the global context of the scene into GCA, while it maintains the recursive local operations. Specifically, we provide the consistent bird's eye view (BEV) features to GCA kernels, independent of time step t. The features are trained to plan ahead and predict very low-resolution, yet dense, final occupancy from the initial state s^0.

BEV features. The planner module is depicted in Fig. 4 inside the green box. We first voxelize the input point cloud to initial state s^0 and transform it to $h_r \times w_r$ bird's eye view (BEV) image. Akin to PointPillars [21], each pixel on the BEV image contains the feature extracted from the voxels within the vertical 'pillar.' Each pillar aggregates

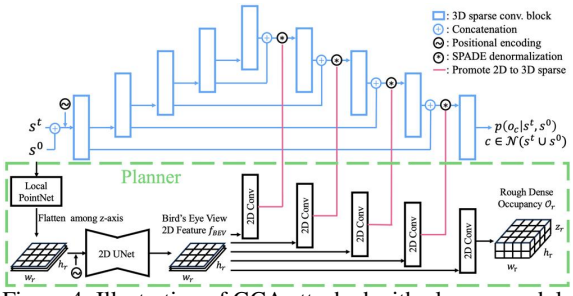


Figure 4. Illustration of GCA attached with planner module.

$3 \times 3 \times z_{max}$ voxels (z is the up-axis, z_{max} is the maximum voxels along z axis) of the original voxel grid. After the x and y coordinates of occupied voxels are converted into offsets from the pillar center and z coordinate is normalized by z_{max}, a local PointNet [29] processes them to obtain a feature for the corresponding pixel in the low-resolution BEV image. We further add 2D positional encoding [35] to encode relative position, and pass them through a dense 2D UNet [30] to obtain *global* BEV features f_{BEV}.

Training GCA with BEV features. As shown in Fig. 4, we use 2D convolutions on f_{BEV} to provide the global guidance (shown in pink) in the decoder layers of the sparse UNet of the GCA kernel (shown in blue). Inspired by the spatial conditioning mechanism in SPADE [27], the 2D convolutions compute a mean and variance per pillar. The means and variances per pillar are added and multiplied to the 3D sparse features falling within the pillar, effectively de-normalizing them. To ensure f_{BEV} contains necessary information to plan geometry, we apply an auxiliary guidance loss. Specifically, f_{BEV} is trained to decode low-resolution 3D occupancy $\mathcal{O}_r$ of shape (h_r, w_r, z_r) (we typically use $z_r = 4$, implying voxels of 2 meter height) with a 2D convolution layer. It is supervised with a cross-entropy loss,

$$\mathcal{L}_{\text{BEV}} = CE(\mathcal{O}_r, \mathcal{O}_r^{\text{gt}}), \tag{7}$$

where $\mathcal{O}_r^{\text{gt}}$ is the ground truth coarse occupancy in the resolution of $\mathcal{O}_r$. The final loss is a weighted combination of Eq. 6 and 7 with weight β,

$$\mathcal{L} = \mathcal{L}_{\text{GCA}} + \beta\mathcal{L}_{\text{BEV}}. \tag{8}$$

3.3. Upsampling to Continuous Geometry

Most prior works in semantic scene completion target AV perception, and it suffices to predict occupancy at 20cm^3 voxel resolution given the LiDAR scan. In contrast, hGCA can create high-resolution surfaces that are suitable for content creation. Given the low-resolution generation of the scene from the previous step, hGCA generates voxels with latent vectors for local implicit functions [5, 20, 47] in a higher resolution. hGCA's hierarchical generation achieves efficiency in both space and time complexity for the large-scale under-constrained problem, disentangling geometry completion and upsampling into separate steps.

Generative process. We utilize a continuous version of GCA, named cGCA [47], as our generative model for conditional up-sampling into an implicit representation. cGCA

extends GCA to generate continuous surface, using an augmented state x of each cell c, adding a local implicit latent feature z_c, i.e. $x = \{(c, o_c, z_c)|c \in \mathbb{Z}^3, o_c \in \{0, 1\}, z_c \in \mathbb{R}^K\}$, where K is the dimension of the latent feature. The local implicit latent features z_c are in the latent space of a pre-trained auto-encoder, as in [47]. The encoder g_ξ encodes coordinate-distance input pairs to x and the decoder $f_\omega(x)$ decodes any point in $\mathbb{R}^3$ into an unsigned distance to surface. We additionally double the voxel resolution (*i.e.*, 10cm^3) from our low resolution completion (20cm^3 voxels). While one could theoretically obtain continuous surface even with implicit latent vectors in low-resolution voxels, we observe improved shape fidelity when using finer resolution voxels.

As in GCA, a state x^{t+1} is sampled at each transition step $x^{t+1} \sim p_\phi(x^{t+1}|x^t, x^0)$, where x^0 is the initial state. We set coordinates in the sparse tensor x^0 to be the union of the input LiDAR scans and our low resolution completion s^{T_1}, all provided in a finer voxel resolution. For cells c that belong to the input point cloud, we set their latent feature z_c using the pretrained encoder g_ξ. We set the initial features z_c to zeros for cells c in x^0 that come solely from s^{T_1}. After recursively sampling T_2 steps from p_ϕ, the final state x^{T_2} is decoded into a distance function using the pretrained decoder f_ω, yielding an output mesh. Further details regarding cGCA are in [47].

Training. The training for upsampling is similar to that of GCA, derived for a continuous case. Specifically, the training operates in the augmented state x, mapped using the encoder g_ξ from coordinate-distance pairs. For the initial state x_0, we use the ground truth low-resolution voxels s^{gt} instead of the generated output from the first stage s^{T_1}. This enforces the upsampler to solely learn to upsample, and the two stages are trained independently and therefore efficiently. If we use the outputs s^{T_1}, the stochasticity may lead to inconsistent supervision and training instability. We refer the readers to the Appendix for further details.

4. Experiments

We evaluate hGCA on street scene generation given LiDAR scans captured from AVs. We assume registered sequential scans *i.e.* relative poses between captures are known. We train and evaluate on synthetic street scenes from CARLA [13] and Turbosquid [2] against state-of-the-art methods in Sec. 4.1 for scene extrapolation quality, using ground truth geometry. We use synthetic LiDAR scans to train, matching the LiDAR scan pattern from Waymo-Open [34]. In Sec. 4.2, we test generalization abilities of hGCA on real LiDAR scans from Waymo-Open [34] and on novel objects, unseen in training. Lastly, in Sec. 4.3, we investigate the planner module. We provide further analysis in Appendix.

Datasets. 1) **Karton City** is a synthetic city comprised of 20 city blocks, obtained from the Turbosquid 3D asset marketplace[2]. We split 20 blocks into train/val/test splits and re-combine 4 blocks in each split randomly per scene. We simulate parked cars by placing car assets from ShapeNet [6]. 2) **CARLA [13]** is an open source driving simulator with diverse environments. We use 5/1/1 towns as train/val/test split with randomly placed static vehicles. We simulate random ego-vehicle trajectories on synthetic data for training and evaluation, detailed in the Appendix. 3) **Waymo-open** [34] is a real-world AV dataset with registered LiDAR scans, used here to demonstrate sim-to-real generalization qualitatively. We discuss issues with quantitatively evaluating 3D generative models on real data in Sec. 4.2 and the Appendix.

Evaluation Metrics. We evaluate our generated scenes using three metrics to capture various aspects of scene extrapolation. 1) **High LiDAR ReSim** evaluates geometry fidelity, focusing on regions visible from the street. It measures the chamfer distance (CD) between two LiDAR scans from poses distant from the center, one from the ground truth (GT) mesh, and the other from the completed scene. For this metric, we add high elevation angles to the LiDAR sensor to evaluate generation *beyond* maximum input height. The metric (deliberately) avoids evaluating on inconsistent geometry in interior walls of buildings in GT, which are invisible to LiDAR from the street (green boxes in Fig. 5). Following evaluation in semantic scene completion [2], we compute 2) **IoU** at 20cm^3 voxel resolution, for all voxels visible to the high elevation LiDAR (to measure beyond input height) from all novel sampled poses on the ego-vehicle trajectory, also used in LiDAR ReSim. In contrast to the point-wise High LiDAR ReSim metric, IoU evaluates rough occupancy of large scene context, independently of the 3D representation used in generated geometry. Additionally, we propose 3) **Street CD** to include evaluation on geometry completely occluded from the ego-trajectory, such as the sidewalk side of parked cars. On Karton City dataset, where the scene is a simple crossroad junction, we compute Chamfer distance between the generated geometry against GT, only on the objects on the main street. Due to the simplicity and abundance of flat ground in Karton City, we remove it from evaluation by simple height thresholding. For generative models, we generate $k = 3$ generations and measure minimum and average metrics to account for stochasticity, and also measure Total Mutual Difference (TMD) [39] to capture generation diversity.

Baselines. We compare hGCA with state-of-the-art AV semantic scene completion methods (JS3CNet [42], SCPNet [40]), indoor scene completion methods (SG-NN [11], ConvOcc [33]), and generative models based on GCA (GCA [46], cGCA [47]), training all models from scratch. We adapt semantic scene completion methods to our setting by changing the semantic class output to binary occupancy. We refer to Appendix for more details on datasets, evaluation

[2] Clickable link to asset on turbosquid.com

		5 scans										10 scans									
		CARLA				Karton City						CARLA				Karton City					
Method	Represe -ntation	High LiDAR ReSim min. ↓	avg. ↓	TMD ↑	IoU	High LiDAR ReSim min. ↓	avg. ↓	TMD ↑	IoU	Street CD min. ↓	avg. ↓	TMD ↑	High LiDAR ReSim min. ↓	avg. ↓	TMD ↑	IoU	High LiDAR ReSim min. ↓	avg. ↓	TMD ↑	IoU	Street CD min. ↓	avg. ↓	TMD ↑
ConvOcc	implicit	15.52		-	13.40	10.35		-	25.64	17.13		-	14.62		-	13.74	9.25		-	26.54	15.13		-
SCPNet	$20cm^3$	5.77		-	49.82	4.82		-	68.53	3.64		-	5.47		-	52.49	4.28		-	72.48	3.14		-
JS3CNet	$20cm^3$	6.58		-	51.02	5.28		-	63.76	3.92		-	6.29		-	53.39	5.02		-	65.68	3.26		-
	$10cm^3$	6.64		-	43.46	3.86		-	70.28	2.30		-	4.99		-	46.76	3.46		-	73.11	1.92		-
SG-NN	$10cm^3$	5.06		-	50.76	4.06		-	70.18	2.61		-	4.53		-	54.29	3.42		-	73.58	2.04		-
GCA	$20cm^3$	5.58	5.83	1.45	50.91	3.95	4.03	0.61	74.95	2.87	3.34	1.16	5.30	5.54	1.36	54.40	3.79	3.85	0.45	78.04	2.52	2.73	0.77
	$10cm^3$	5.66	6.17	2.25	44.26	3.93	4.10	1.16	68.23	3.38	4.16	**2.28**	5.13	5.52	1.96	48.26	3.28	3.40	0.83	72.38	2.58	3.11	**1.45**
cGCA	implicit	7.04	7.59	3.19	35.43	4.29	4.42	1.23	59.79	2.49	3.36	2.00	6.84	7.43	2.97	36.17	4.00	4.09	0.91	63.56	1.65	2.02	0.92
hGCA	$10cm^3$	4.60	4.72	0.80	**53.84**	**3.20**	3.25	0.51	**75.97**	2.09	2.27	0.64	4.38	4.48	0.78	**56.85**	2.97	3.01	0.41	**79.63**	1.79	1.89	0.54
	implicit	**4.53**	4.65	0.92	52.17	**3.20**	3.25	0.56	70.45	**1.85**	2.02	0.51	**4.30**	4.40	0.88	54.68	**2.95**	2.99	0.45	73.38	**1.55**	1.65	0.42
input		6.33		-	34.43	6.83		-	38.32	5.63		-	5.46		-	40.42	5.45		-	47.71	4.99		-

Table 1. Quantitative results on CARLA and Karton City with 5 and 10 scans given as input. All results except IoU are multiplied by 10 in meter scale. LiDAR Resim and Street CD evaluates the fidelity of completion and TMD measures the diversity of generation. High LiDAR Resim uses high elevation LiDAR to evaluate the extrapolation. IoU is computed with ground truth geometry.

metrics and baselies.

Implementation details. For our coarse stage, we use $20cm^3$ voxel size, $T_1 = 30$ transition steps with radius $r = 1$. In the upsampling model, we use $10cm^3$ size, $T_2 = 15$ transition steps with radius $r = 2$. We set BEV loss weight $\beta = 0.1$ for all experiments, with planner parameters $h_r = h_{max}/3, w_r = w_{max}/3, z_r = 4$ unless stated otherwise. We train and infer (unless specified) on scenes in a volume of $38.4 \times 38.4 \times 8$ meters with height ranging from $[-1, 7]$ meters in a ego vehicle frame, randomly selected from one of the poses the input scans. At $20cm^3$ voxel resolution, this corresponds to $h_{max} = 192$, $w_{max} = 192$ and $z_{max} = 40$. All experiments were performed on a single 24GB RTX 3090 GPU. We obtain and reuse the pre-trained latent autoencoder g_ξ and f_ω for local-implicits used in cGCA [47], trained on indoor scenes from 3DFront [14], which generalizes well to our data. We simulate synthetic LiDAR with simple ray-casting and add noise to LiDAR scan coordinates and relative poses, which improves sim-to-real generalization. We report results without LiDAR noise in the Appendix.

4.1. Synthetic Scene Completion

Scene extrapolation results on synthetic scenes are reported in Tab. 1. We accumulate 5 or 10 LiDAR scans from random poses as input. We train all models on combined CARLA and Karton City data for added diversity, but evaluate separately. Output representation for baselines are voxels or continuous surfaces when available. hGCA outperforms all baselines by a margin in reconstruction metrics while generating diverse outputs. For hGCA, we report scores of $10cm^3$ voxel occupancy and implicit representation, both obtained after upsampling. We notice that IoUs are slightly higher with our $10cm^3$ voxels, resulting from our unsigned distance fields sometimes not generating clear zero-level iso-surfaces, creating thick meshes for thin structures after thresholding, similar to [47]. We find that the planner trades off diversity for quality and global consistency, discussed further in Sec. 4.3. We show qualitative results in Fig. 5, and many more in the Appendix. Deterministic completion models (ConvOcc [33], SCPNet [40], JS3CNet [42], SG-NN [11]) tend to conservatively generate geometry beyond the input, such as bus stops or cars in Fig. 5. These approaches lack multi-modality and we hypothesize that it limits generation to high-confidence geometry near sparse inputs by tending to model a mean distribution. In contrast, hGCA generates well completed geometry with high fidelity.

4.2. Generalization across Domains

Generalization to real LiDAR scans. We find that hGCA generalizes well from sim-to-real, successfully completing cars and trees from Waymo-open LiDAR scans, using 5 accumulated scans as input, visualized in Fig. 6, Fig. 7, and more in the Appendix. Fig. 7 shows that deterministic baselines again exhibit conservative behavior, whereas naive-GCA suffers from inconsistency, in one case generating a tree from a house. Both GCA and hGCA complete occluded cars in the parking lot or even inside of a garage (Fig. 7), where the latter has never been seen in the synthetic training data. We hypothesize that this generalization stems from the locality of GCA and the two-stage approach where the coarser GCA is more robust to real-world noise. Overall, hGCA can generate convincing completions, exhibiting geometric quality inferior, yet closer, to the synthetic data it was trained on compared to baselines, taking a step towards simulation-ready environment creation using AV LiDAR as a content scanner. Evaluating a 3D generative model on real AV data is challenging. The best source of ground truth geometry available to us is using all accumulated scans as in semantic scene completion, also shown in Fig. 7, which is highly incomplete and has limited height range. For example, in the left of Fig. 7, hGCA generates more complete geometry, but is worser on LiDAR Resim or IoU scores compared to baselines (SCPNet: 3.94/63.93, SG-NN: 2.97/58.67, hGCA: 3.58/63.46), using held-out real scans in the scene for LiDAR ReSim and IoU against accumulated scans. We discuss difficulties of evaluation on real-world datasets further in the Appendix.

Out-of-distribution inputs. A fair concern with training on synthetic content is the limited diversity of assets the generative model is trained on, which could be reflected in the outputs. We show anecdotal evidence of novel asset completion in Fig. 8. On the right, we show geometry completion from sythetic LiDAR of a three-wheeler vehicle asset taken from Objaverse-XL [12][3]. We verify that no three-wheelers exist in our training data. hGCA generates a convincing

[3] Clickable link to asset on sketchfab.com

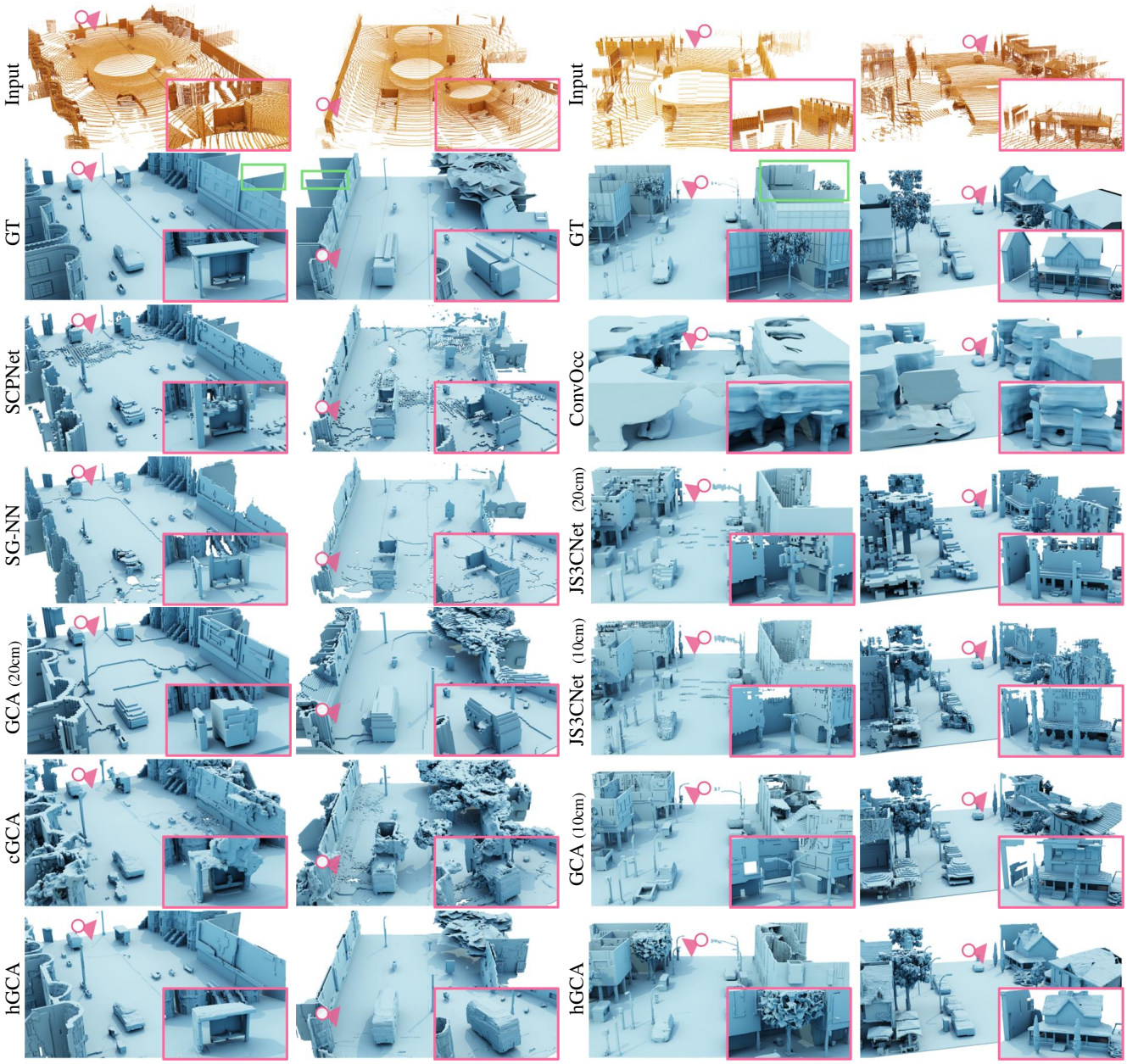


Figure 5. Visualizations on CARLA (first 2 columns) and Karton City (last 2 columns) from 5 scans. hGCA generates high-resoluton geometry beyond field of view (bus stops, trees, roofs) and occlusions (cars) compared to existing baselines. Deterministic baselines tend to conservatively complete high-confidence regions near the input. Green boxes demonstrate inconsistency of building interiors in GT data.

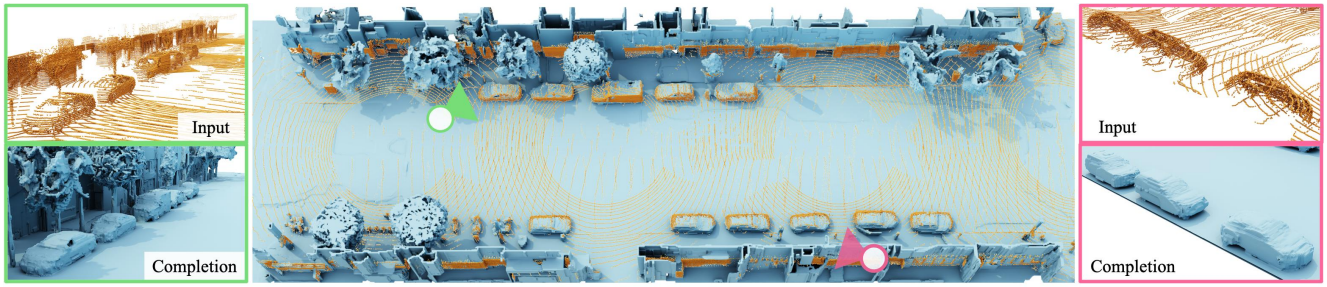


Figure 6. Completion given accumulation of 5 LiDAR scans (yellow spheres) on real-world Waymo-open dataset. hGCA can extrapolate beyond input field of view (walls) and occlusion (cars). Walls in pink boxes are cut off for ease of visualization.

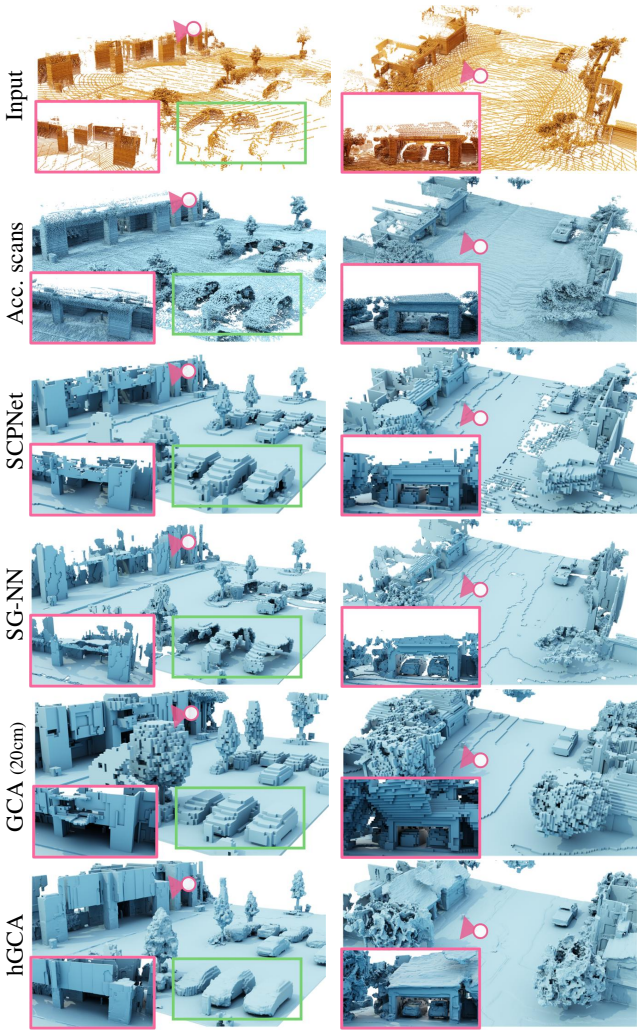
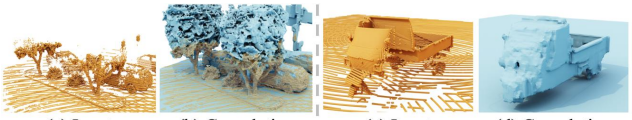


Figure 7. Visualizations on real-world Waymo-open dataset. hGCA exhibits great sim-to-real performance compared to existing method with high fidelity (pink box) and can generate more complete shapes than accumulated scans (green box).



(a) Input (b) Completion (c) Input (d) Completion

Figure 8. (a), (b): Completion on LiDAR scan from Waymo-open. (c), (d): Completion on synthetic LiDAR of a three-wheeler asset from sketchfab 3 hGCA can realistically complete from tree trunks or three-wheeler cars unseen in training, taking geometric cues from the input (yellow spheres).

three-wheeler, respecting the input, better than deterministic baselines, which we visualize in Appendix. We also show how hGCA can complete this asset with some lower density input scans in Appendix. The left of Fig. 8, shows LiDAR scans from Waymo-open of complex, unique tree trunks. hGCA, trained only on single trunk trees, is able to generate realistic trees that preserve the observed structure. These results require further rigorous validation, but show promise towards environment creation with diverse assets from geometric cues taken from real LiDAR scans.

		CARLA				Karton City			
Voxel Size	z_r	LiDAR ReSim min. ↓	LiDAR ReSim avg. ↓	LiDAR ReSim TMD ↑	IoU	LiDAR ReSim min. ↓	LiDAR ReSim avg. ↓	LiDAR ReSim TMD ↑	IoU
$20cm^3$	✗	5.58	5.83	**1.45**	50.91	3.95	4.03	**0.61**	74.95
	27	5.44	5.57	0.94	49.37	**3.92**	3.97	0.41	**76.34**
	16	5.37	5.54	0.91	51.67	3.96	4.00	0.41	76.2
	4	**5.28**	5.40	0.77	**53.81**	3.97	4.03	0.44	75.86
	2	5.32	5.46	0.86	52.49	3.99	4.05	0.47	75.49
$10cm^3$	✗	5.66	6.17	**2.25**	44.26	3.93	4.10	1.16	68.23
	4	**4.58**	4.74	0.93	**51.40**	**3.44**	3.52	0.67	**72.12**

Table 2. Ablation study on effects of Planner from 5 input scans. ✗ in z_r refers to vanilla GCA without Planner module.

4.3. Ablation Studies on Planner

The planner module aims to induce global consistency in hGCA. Table 2 shows quantatively that it trades of diversity for completion performance compared to vanilla GCA We test different resolutions of planner occupancy prediction in height, indicated by z_r. We found that $z_r = 4$ is a good balance between providing global context without hurting generation performance on CARLA, which has a more diverse validation set. We can infer that predicting occupancy in finer vertical resolution (large z_r) may be beyond the capacity of our simple planner module and hinders joint optimization of the local GCA loss with the global planner loss. We observed that the planner does not boost the performance of the upsampling module, indicating that local upsampling does not benefit from coarse global context.

5. Conclusion

We proposed hierarchical Generative Cellular Automata (hGCA), a spatially scalable generative model that generates 3D scenes beyond occlusions and input field of view from several LiDAR scans. Our model generates scenes in a two-stage hierarchical coarse-to-fine manner, where the first stage generates coarse geometry by providing global consistency to GCA with a light-weight planner module. The second stage synthesizes finer details by applying cGCA conditioned on the coarse geometry. On synthetic scenes, hGCA generates plausible scenes with higher fidelity and completness compared to prior state-of-the-art works. hGCA demonstrates strong sim-to-real generalization, capable of extrapolating LiDAR scans on real-world Waymo dataset. While hGCA takes a step towards content creation from LiDAR scans, several desiderata remain. Improving fidelity of geometry and generating textures, materials etc. is needed for usability of the completed geometry. For example, in Fig. 7 right column, hGCA generates inconsistent roofs. The current generative process of hGCA is slow, disabling use of the model in real-time, which we leave to future work.

Acknowledgements. This work was supported by Institute of Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government(MSIT) [NO.2021-0-01343, Artificial Intelligence Graduate School Program (Seoul National University)] and Creative-Pioneering Researchers Program through Seoul National University.

References

[1] Himanshu Arora, Saurabh Mishra, Shichong Peng, Ke Li, and Ali Mahdavi-Amiri. Multimodal shape completion via imle, 2021. 2

[2] 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 *ICCV*, 2019. 2, 5

[3] Florian Bordes, Sina Honari, and Pascal Vincent. Learning to generate samples from noise through infusion training. In *ICLR*, 2017. 2, 3

[4] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[5] Rohan Chabra, Jan Eric Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard A. Newcombe. Deep local shapes: Learning local SDF priors for detailed 3d reconstruction. *CoRR*, abs/2003.10983, 2020. 2, 4

[6] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 5

[7] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[8] Ran Cheng, Christopher Agia, Yuan Ren, Xinhai Li, and Liu Bingbing. S3cnet: A sparse semantic scene completion network for lidar point clouds. In *Conference on Robot Learning*, pages 2148–2161. PMLR, 2021. 2

[9] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6545–6554. IEEE Computer Society, 2017. 2

[10] Angela Dai, Daniel Ritchie, Martin Bokeloh, Scott Reed, Jürgen Sturm, and Matthias Nießner. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. In *CVPR*, pages 4578–4587. IEEE Computer Society, 2018. 2

[11] Angela Dai, Christian Diller, and Matthias Nießner. Sg-nn: Sparse generative neural networks for self-supervised scene completion of rgb-d scans. In *CVPR*, 2020. 2, 5, 6

[12] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *arXiv preprint arXiv:2307.05663*, 2023. 6

[13] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. 2, 5

[14] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10933–10942, 2021. 6

[15] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances in Neural Information Processing Systems*, 2022. 2

[16] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. *CVPR*, 2018. 3

[17] Jiayuan Gu, Wei-Chiu Ma, Sivabalan Manivasagam, Wenyuan Zeng, Zihao Wang, Yuwen Xiong, Hao Su, and Raquel Urtasun. Weakly-supervised 3d shape completion in the wild. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 283–299. Springer, 2020. 2

[18] Philipp Henzler, Niloy J. Mitra, and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 2

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 2

[20] Chiyu Max Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas A. Funkhouser. Local implicit grid representations for 3d scenes. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 6000–6009. IEEE, 2020. 2, 4

[21] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4

[22] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[23] Mateusz Michalkiewicz, Jhony K. Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders P. Eriksson. Deep level sets: Implicit surface representations for 3d shape inference. *CoRR*, abs/1901.06802, 2019. 2

[24] Alexander Mordvintsev, Ettore Randazzo, Eyvind Niklasson, and Michael Levin. Growing neural cellular automata. *Distill*, 5(2):e23, 2020. 3

[25] Yancheng Pan, Biao Gao, Jilin Mei, Sibo Geng, Chengkun Li, and Huijing Zhao. Semanticposs: A point cloud dataset with large quantity of dynamic instances. *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 687–693, 2020. 2

[26] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous

signed distance functions for shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[27] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 4

[28] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv*, 2022. 2

[29] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *arXiv preprint arXiv:1612.00593*, 2016. 4

[30] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015. (available on arXiv:1505.04597 [cs.CV]). 3, 4

[31] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 2256–2265. JMLR.org, 2015. 2

[32] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. *CVPR*, 2017. 2

[33] Lars Mescheder Marc Pollefeys Andreas Geiger Songyou Peng, Michael Niemeyer. Convolutional occupancy networks. In *European Conference on Computer Vision (ECCV)*, 2020. 2, 5, 6

[34] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, pages 2446–2454, 2020. 2, 5

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. 4

[36] John Von Neumann, Arthur W Burks, et al. Theory of self-reproducing automata. *IEEE Transactions on Neural Networks*, 5(1):3–14, 1966. 2

[37] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in neural information processing systems*, 29, 2016. 2

[38] Rundi Wu, Xuelin Chen, Yixin Zhuang, and Baoquan Chen. Multimodal shape completion via conditional generative adversarial networks. In *The European Conference on Computer Vision (ECCV)*, 2020. 2

[39] Rundi Wu, Xuelin Chen, Yixin Zhuang, and Baoquan Chen. Multimodal shape completion via conditional generative adversarial networks. In *ECCV*, 2020. 5

[40] Zhaoyang Xia, Youquan Liu, Xin Li, Xinge Zhu, Yuexin Ma, Yikang Li, Yuenan Hou, and Yu Qiao. Scpnet: Semantic scene completion on point cloud. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2023. 2, 5, 6

[41] Yuwen Xiong, Wei-Chiu Ma, Jingkang Wang, and Raquel Urtasun. Learning compact representations for lidar completion and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1074–1083, 2023. 2

[42] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. In *AAAI*, pages 3101–3109, 2021. 2, 5, 6

[43] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. *arXiv*, 2019. 2

[44] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *3D Vision (3DV), 2018 International Conference on*, 2018. 2

[45] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2

[46] Dongsu Zhang, Changwoon Choi, Jeonghwan Kim, and Young Min Kim. Learning to generate 3d shapes with generative cellular automata. *arXiv preprint arXiv:2103.04130*, 2021. 1, 2, 3, 5

[47] Dongsu Zhang, Changwoon Choi, Inbum Park, and Young Min Kim. Probabilistic implicit scene completion. *ICLR*, 2022. 1, 2, 3, 4, 5, 6

[48] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5826–5835, 2021. 2

[49] Vlas Zyrianov, Xiyue Zhu, and Shenlong Wang. Learning to generate realistic lidar point clouds. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 2