GenAssets: Generating in-the-wild 3D Assets in Latent Space

Ze Yang[1,2] Jingkang Wang[1,2] Haowei Zhang[1]
Sivabalan Manivasagam[1,2] Yun Chen[1,2] Raquel Urtasun[1,2]
[1]Waabi [2]University of Toronto
{zyang, jwang, hzhang, siva, ychen, urtasun}@waabi.ai

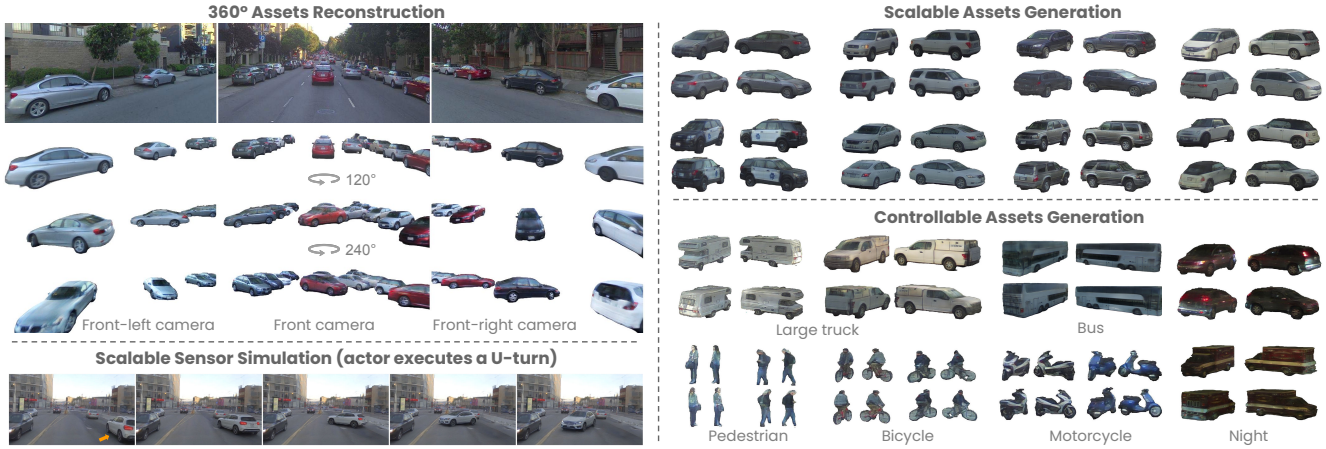

Figure 1. **GenAssets takes in-the-wild camera image(s) and point cloud(s), and automatically reconstruct or generate 360° assets**. Our 3D assets are diverse and high-quality with complete geometry and appearance, allowing for realistic and scalable sensor simulation.

Abstract

High-quality 3D assets for traffic participants are critical for multi-sensor simulation, which is essential for the safe end-to-end development of autonomy. Building assets from in-the-wild data is key for diversity and realism, but existing neural-rendering based reconstruction methods are slow and generate assets that render well only from viewpoints close to the original observations, limiting their usefulness in simulation. Recent diffusion-based generative models build complete and diverse assets, but perform poorly on in-the-wild driving scenes, where observed actors are captured under sparse and limited fields of view, and are partially occluded. In this work, we propose a 3D latent diffusion model that learns on in-the-wild LiDAR and camera data captured by a sensor platform and generates high-quality 3D assets with complete geometry and appearance. Key to our method is a "reconstruct-then-generate" approach that first leverages occlusion-aware neural rendering trained over multiple scenes to build a high-quality latent space for objects, and then trains a diffusion model that operates on the latent space. We show our method outperforms existing reconstruction and generation based methods, unlocking diverse and scalable content creation for simulation. Please visit https://waabi.ai/genassets *for more details.*

1. Introduction

Simulation environments enable testing the performance of autonomy systems in long-tail and safety critical scenarios safely, efficiently, and scalably [41, 61, 77, 98]. To test the complete autonomy stack, the virtual environments should simulate the sensor data (*e.g.*, LiDAR, camera) of the robot. Realistic and consistent sensor simulation across multiple modalities depends on the availability of high-quality 3D *assets* that accurately represent the geometry and appearance of traffic participants such as cars and motorcycles.

Commoditized simulation environments rely on artists to manually build such 3D assets [14, 59, 62]. This process is slow, costly, and requires precise specification of attributes such as geometry, UV mapping, and materials. As a consequence, existing methods utilize very limited content, failing to capture the full diversity of objects in the real world.

An alternative and much more appealing approach that has gained traction is to reconstruct assets from real world data captured by the sensor platform, which enables diversity and realism. Reconstruction-based neural rendering methods have demonstrated impressive results by optimizing 3D representations that can render and match input sensor data for self-driving scenes [11, 15, 29, 70, 74, 86, 94, 95, 98, 104]. These approaches render high-quality data

when the viewpoint is close to the training views, but can suffer from severe artifacts at novel viewpoints due to incomplete observations. Additionally, each asset must be observed in the real world and reconstructed via optimization, making it computationally expensive and limiting diversity.

Recent 3D generative models [6, 7, 17, 47, 57, 63] have shown promise in generating complete assets and rendering novel views far from source. These methods leverage a combination of generative adversarial networks (GANs), diffusion models, and neural rendering, and train on large corpuses of images to learn shape and appearance priors. NeRF-based diffusion models that optimize via score-distillation sampling (SDS) [34, 52], or denoise in 3D space [9, 20, 45, 48, 85], have demonstrated particularly high-quality complete shape and appearance generation, and also offer controllability through text/image conditioning. However, such models usually rely on synthetic datasets with ground-truth 3D models to render the dense 2D views necessary to learn a 3D representation prior. They also primarily work on "clean" images that fully capture the object of interest. This reliance on clean synthetic data limits their ability to handle in-the-wild real data from a moving platform, where the data is typically captured from sparse viewpoints, under partial occlusions, limited resolution, and with sensor noise (*e.g.*, lighting artifacts, rolling shutter).

In this work, we propose a latent diffusion 3D generative model that learns directly from sparse, in-the-wild data, enabling high-quality asset generation at scale. We tackle the challenges of asset completion and generative modelling via a two-stage "reconstruct-then-generate" methodology. In the first stage, we learn a low-dimensional object latent space that generates complete assets by training across multiple scenes via neural rendering. To handle partial occlusion, we jointly learn a scene representation for both the static scene and dynamic actors, enabling occlusion-aware rendering during learning. In the second stage, we train a diffusion model to operate on the 3D latent space, enabling realistic asset generation that can be conditioned on individual views, day or night, or the actor class, enabling asset variations that are applicable to new scenes. We evaluate our model on the public self-driving dataset PandaSet [88], demonstrating its capability to reconstruct and generate high-quality assets from sparse, in-the-wild data. Additionally, we showcase applications such as conditional generation, as well as realistic rendering of our assets for downstream simulation systems, highlighting our approach's flexibility, scalability, and practical utility.

2. Related Work

3D Reconstruction in the Wild: Reconstructing objects from in-the-wild data is essential for creating 3D digital twins and simulations, but poses significant challenges due to the complexity (*e.g.*, occlusions, sparse viewpoints, partial observations) and noise (*e.g.*, localization and calibration inaccuracies [100], sensor noise). Object-based reconstructions fall into optimizing explicit meshes [46, 78, 96] or implicit representations [27, 43, 49, 80, 97, 99, 101] methods via differentiable rendering. However, they rely on accurate object segmentation masks and struggle to handle complex occlusions. To address this issue, recent works propose compositional neural fields [35, 50, 53, 74, 98] or 3D Gaussians [11, 29, 94] to decompose 3D world into static background and dynamic actors. These works often involve costly per-scene optimizations and cannot generalize well to novel views with large shifts nor produce complete (*i.e.*, 360-degree) objects. To mitigate these limitations, recent works propose generalizable reconstruction models [10, 23, 44, 56, 83] that generate a 3D representation with a single feed-forward network trained on many scenes. Another line of work aims to incorporate data-driven priors from pre-trained models (*e.g.*, stable diffusion [58]) via score-distillation sampling (SDS) [25, 52, 81, 84]. However, existing methods either require object-centric synthetic training data [23, 83, 84] (*e.g.*, Objaverse [12]) and/or perform poorly on in-the-wild sensor data [10, 25, 81]. Our approach builds high-fidelity and complete assets by directly learning on real-world data.

GAN-based 3D Generation: Generative Adversarial Networks [19] (GANs) have shown great potential for 3D asset generation [17, 91], where the generator transforms random noise into the target 3D representation, and the discriminator is jointly trained using 2D images rendered from the generated representation and the ground truth. This paradigm allows generation without requiring explicit 3D ground truth. Specifically, π-GAN [6] introduces a generative model for high-quality 3D-aware image synthesis based on neural radiance fields. GIRAFFE [47] further proposes to represent scenes as compositional generative neural feature fields, allowing for better object-background disentanglement and controllable scene generation. EG3D [7] and GET3D [17] further improve 3D generation efficiency and quality by integrating the triplane representation (for volume rendering or explicit mesh rendering) with a StyleGAN [28] architecture. Although they achieve robust and consistent 3D generation, they primarily focus on "clean" object-centric images with full coverage and no occlusions and have difficulties generalizing to in-the-wild driving data. Most recently, DiscoScene [91] and GINA-3D [63] learn 3D generative models from in-the-wild data. However, they often produce significant visual artifacts (*e.g.*, floating particles, low-fidelity shapes and texture). In contrast, our method generates high-quality assets.

Diffusion-based 3D Generation: Diffusion-based methods have achieved tremendous success in image generation

[22, 51, 58, 67, 68], leveraging denoising diffusion probabilistic models [22] to transform white noise into high-quality outputs through iterative refinement. To bridge the gap between 2D and 3D generation, a common strategy involves lifting 2D images to 3D via score distillation sampling (SDS) [18, 25, 34, 52, 64, 71, 81, 84]. These methods rely on pre-trained or fine-tuned 2D diffusion models and optimize 3D representations with diffusion-prior guidance. The optimization is slow and the supervision can be multi-view inconsistent. To address this issue, recent works [32, 36, 37, 39, 64, 79] propose to synthesize multi-view consistent images via diffusion models and then perform 3D reconstruction with explicit meshes [90], neural radiance fields [32, 79] or 3D Gaussians [72, 93]. Another line of work performs diffusion directly in 3D, either by training 3D-aware diffusion model from posed 2D imges without requiring explicit 3D supervision [1, 5, 9, 20, 45, 65, 69, 73, 87, 92] or by operating directly on 3D data [38, 40, 75]. Inspired by latent diffusion models (LDM) [58], recent works diffuse within a latent space, where models like UNet [60] or VAEs [30, 76] map 3D data to a compressed latent representation for faster, higher-quality and more scalable synthesis of both objects [57, 85] and large-scale scenes [24, 42, 54, 55, 89, 102]. However, existing works primarily focus on synthetic objects [9, 42] or static scenes [24, 54, 55, 89] and cannot handle complex real-world scenarios. In contrast, we propose a latent diffusion 3D generative model with compositional radiance fields that learns directly from in-the-wild dynamic scenes, enabling high-quality 3D asset generation at scale.

3. Method

Our goal is to build a generative model that can create high-fidelity assets scalably for self-driving simulation. The model should enable unconditional generation of diverse assets for a variety of classes (e.g., cars, buses, trucks, pedestrians), and also support generating complete assets conditioned on new in-the-wild camera images and LiDAR point clouds for actors that may be occluded or far-away. Importantly, to avoid the quality degradation [9] that occurs when generative models trained on synthetic data are transferred to real data, our generative model should directly learn over a corpus of real-world driving scenes collected by a robot sensor platform. Towards this goal, we propose a two-stage "reconstruct-then-generate" framework. In the first stage, we jointly learn a set of latent codes through reconstruction-based neural rendering, where each latent code represents a foreground actor in our dataset. We devise a compositional scene representation that enables rendering of actors of driving scenes in an occlusion-aware manner, enabling us to decouple the foreground actors from the background (*e.g.*, road, sky, vegetation). Fig. 2 provides an overview of this process. In the second stage, we train a diffusion model to learn generative priors in this asset latent space, enabling the generation of realistic and diverse neural assets. This enables both unconditional generation and conditional generation through score-based guidance. Fig. 3 shows the overview of our latent asset diffusion model. We first introduce our compositional scene representation (Sec. 3.1). We then describe how we compress the actor representations into a latent space to learn the set of latent codes via neural rendering (Sec. 3.2), and detail the diffusion process in the latent space and the training process (Sec. 3.3).

3.1. Compositional Scene Representation

We aim to learn a generative 3D asset model from real-world driving scenes. Unlike existing object-centric approaches [7, 9, 17, 18, 45, 48, 85], our sensor data contains many actors captured under different viewpoint, distances, and occlusions. To handle multiple instances and their occlusions, we leverage a compositional scene representation. Each scene is decomposed into a static background $\mathcal{B}$ and a set of dynamic actors $\{\mathcal{A}_i\}_{i=1}^{K}$, with each actor enclosed within a 3D box and localized by a trajectory of $SE(3)$ poses. We model the static background and dynamic actors with separate neural feature fields parameterized by triplane representations [7]. Specifically, for each dynamic actor's volume, explicit features are aligned along three orthogonal planes, each with resolution $N_{\mathcal{A}} \times N_{\mathcal{A}} \times C$, where $N_{\mathcal{A}}$ is the spatial resolution, and C is the feature dimension. Similarly, triplane features of resolution $N_{\mathcal{B}} \times N_{\mathcal{B}} \times C$ represent the background volume. For a spatial query coordinate $\mathbf{x} \in \mathbb{R}^3$, we project it onto each of the three feature planes $\mathbf{t} = (\mathbf{t}^{xy}, \mathbf{t}^{xz}, \mathbf{t}^{yz})$ and bilinearly interpolate the corresponding feature vectors. The interpolated features are concatenated with view direction $\mathbf{d} \in \mathbb{S}^2$ and processed by an MLP network f_{feat} to yield geometry as a signed-distance function (SDF) s and a neural feature $\mathbf{f}$. This querying process is defined by:

$$s, \mathbf{f} = f_{\text{feat}}(\{\texttt{interp}(\mathbf{x}^p, \mathbf{t}^p)\}_{p \in \{xy, xz, yz\}}, \mathbf{d}), \quad (1)$$

where $\mathbf{x}^p$ represents the 2D projection of $\mathbf{x}$ onto feature plane p. To render the scene representation, we first transform the object-centric actor neural feature fields to world coordinates similar to [74, 98]. Then we composite the background and actor feature fields. The composited geometry and appearance features are then rendered into sensor observations through neural volume rendering.

3.2. Learning Latent Asset Representations

Encoding to a Latent Space: Per-scene reconstruction methods [74, 94, 98] train each scene separately, with its own set of actor and background triplanes. This prevents effective handling of ambiguities in occluded or unseen regions, resulting in actor representations that generalize poorly to unseen viewpoints. Naively training jointly over

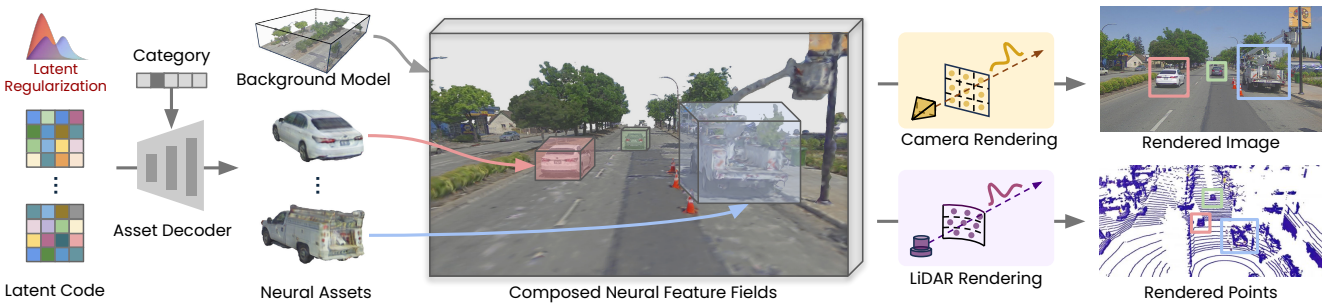

Figure 2. **Learning latent asset representation.** We learn a low-dimensional object latent space that generates complete assets by training across multiple scenes via occlusion-aware neural rendering. The asset decoder is trained to map low-dimension latent codes into neural assets which are then composed with learnable per-scene background models to match real-world sensor observations.

all the explicit triplanes for every actor in the dataset containing tens of thousands of objects requires vast amounts of GPU memory and still does not force the learned triplanes to have complete actor shape and appearance. To address these limitations, we propose learning a latent code $\mathbf{c}_i \in \mathbb{R}^{n_\mathcal{A} \times n_\mathcal{A} \times c}$ for each actor representation and a class embedding $\mathbf{e}_i \in \mathbb{R}^{n_\mathcal{A} \times n_\mathcal{A} \times c}$, combined with a shared asset triplane decoder f_{dec} that maps the latent code into the triplane representation:

$$f_{\text{dec}} : \mathbf{e}_i, \mathbf{c}_i \in \mathbb{R}^{n_\mathcal{A} \times n_\mathcal{A} \times c} \rightarrow \mathbf{t}_i \in \mathbb{R}^{N_\mathcal{A} \times N_\mathcal{A} \times 3C}. \quad (2)$$

The class embedding $\mathbf{e}_i$ has the same spatial resolution as the latent code and is shared among actors of the same class. We concatenate it with the actor latent code before passing it to the decoder. The decoder f_{dec} upsamples the latent code by a factor $f = N_\mathcal{A}/n_\mathcal{A}$. The intuition is that different actors are observed from various viewpoints, so their feature planes $\mathbf{t}_i$ capture different informative regions. By compressing them into a latent code bottleneck, we enable learning shape and appearance priors, allowing inference of invisible parts from sparse observations.

Rendering Sensor Observations: Now that we have decoded the asset neural fields $\{\mathbf{t}_i\}_{i=1}^N$ from the latent codes $\{\mathbf{c}_i\}_{i=1}^N$, the next step is to composite them with the background neural fields $\mathbf{t_B}$ and render into the sensor data. In this work, we focus on camera images and LiDAR point clouds, as they are primary sensory modalities for SDVs.

For camera rendering, we use a hybrid volume and neural rendering framework for both efficiency and realism. Given a camera ray $\mathbf{r}(t) = \mathbf{o} + h\mathbf{d}$ sent from the camera center $\mathbf{o}$ to the pixel in direction $\mathbf{d}$, we sample 3D points along the ray, querying geometry s_t and neural feature $\mathbf{f}_t$ via Eq. (1). We then aggregate these samples to obtain the pixel feature through volume rendering:

$$\mathbf{f}(\mathbf{r}) = \sum_{t=1}^{N_t} w_t \mathbf{f}_t, \quad w_i = \alpha_t \prod_{j=1}^{t-1} (1 - \alpha_j), \quad (3)$$

where α_t is the opacity for the t-th point, derived from the SDF s_t following [74, 98]. We repeat this process to volume render all camera pixels, generating a 2D feature map $\mathbf{F} \in \mathbb{R}^{H_f \times W_f \times C_f}$. Then we leverage 2D CNN network f_{rgb} [98] to convert this feature map into an image $\mathbf{I}_{\text{rgb}}$:

$$f_{\text{rgb}} : \mathbf{F} \in \mathbb{R}^{H_f \times W_f \times C_f} \rightarrow \mathbf{I}_{\text{rgb}} \in \mathbb{R}^{H \times W \times 3}. \quad (4)$$

The CNN network f_{rgb} upsamples the rendered map from resolution $H_f \times W_f$ to $H \times W$. This allows us to significantly reduce the number of neural feature field queries. This approach also enhances model capacity and improves image quality by capturing spatial relationships effectively.

In addition to camera images, we use LiDAR rendering for additional geometry supervision. LiDAR sensors emit laser pulses and measure the time of flight to determine distances to reflective surfaces. This depth information provides valuable supervision for asset geometry. Using similar notation, we define $\mathbf{r}(t) = \mathbf{o} + h\mathbf{d}$ as a ray cast from the LiDAR sensor center $\mathbf{o}$ in the direction $\mathbf{d}$. We render LiDAR depth similarly to Eq. (3) by aggregating sample depths via volume rendering: $D(\mathbf{r}) = \sum_{t=1}^{N_t} w_t h_t$, where h_t is the depth of the t-th sampled point, and w_t is the sample weight computed as in Eq. (3).

Learning: We jointly optimize the asset code $\{\mathbf{c}_i\}_{i=1}^{N_c}$, class embedding $\{\mathbf{e}_i\}_{i=1}^{N_e}$, background neural fields $\mathbf{t}_\mathcal{B}$, asset decoder f_{dec}, neural feature fields MLP f_{feat}, and RGB CNN network f_{rgb} by minimizing the differences between rendered and observed sensor data. To achieve high-fidelity reconstruction and rendering, we incorporate perceptual and patch-based adversarial objectives. We also regularize the latent space by applying a Kullback–Leibler (KL) penalty. Our full objective is:

$$\mathcal{L} = \mathcal{L}_{\text{rgb}} + \lambda_{\text{perp}}\mathcal{L}_{\text{perp}} + \lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \lambda_{\text{lid}}\mathcal{L}_{\text{lid}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}}, \quad (5)$$

where L_{rgb} and L_{perp} represent the ℓ_2 photometric loss and the perceptual loss [103] between rendered and observed images, $\mathcal{L}_{\text{adv}}$ is the adversarial objective using patchGAN [26], $\mathcal{L}_{\text{lid}}$ is the ℓ_1 depth loss between rendered and observed LiDAR points, and $\mathcal{L}_{\text{KL}}$ is the injected KL penalty

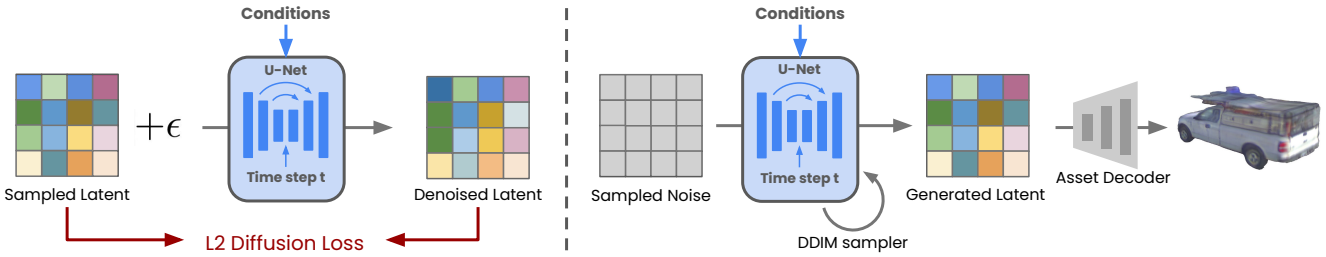

Figure 3. **Left:** Training asset diffusion model in latent space. **Right:** Sampling diffusion model for (un)conditional neural asset generation.

guiding the latent space towards a standard normal distribution, similar to [30, 58]: $\mathcal{L}_{\text{KL}} = \frac{1}{2}\|\boldsymbol{\mu}_i^2 + \boldsymbol{\sigma}_i^2 - 1 - \log(\boldsymbol{\sigma}_i^2)\|_1$, where $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$ represent the mean and standard deviation components of latent code $\mathbf{c}_i$, *i.e.*, $\mathbf{c}_i = \boldsymbol{\mu}_i^2 + \boldsymbol{\sigma}_i \odot \boldsymbol{\epsilon}$, with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. This regularization prevents overfitting and encourages the latent space to be compact, continuous, smooth, and low-variance, enabling easier learning of generative models. Similar to [16, 91], we introduce additional perceptual losses on object patches for enhanced object-level supervision, where the patches are obtained by projecting the 3D instance boxes onto the image plane. Please refer to supp. for more details.

3.3. Latent Asset Diffusion Model

Given our learned asset code library $\{\mathbf{c}_i\}_{i=1}^N$ from the dataset, we aim to learn generative priors using a diffusion model. Diffusion models [22, 66] are probabilistic frameworks that model the data distribution by progressively denoising a Gaussian noise variable. In our approach, the diffusion model operates directly in the latent space and begins with Gaussian noise, progressively denoising it to recover the underlying latent distribution. The *forward* or *diffusion* process is a discrete-time Markov chain that iteratively adds Gaussian noise to the latent code $\mathbf{c}^{(0)}$, producing progressively noisier codes following the transition probability $q(\mathbf{c}^{(t)}|\mathbf{c}^{(t-1)}) = \mathcal{N}(\sqrt{1-\beta^{(t)}}\mathbf{c}^{(t-1)}, \beta^{(t)}\mathbf{I})$, according to a noise schedule $\beta^{(t)}$. After T steps, the code approximates pure Gaussian noise. This *forward* process can be directly sampled at timestep t using closed-form expression:

$$\mathbf{c}^{(t)} = \sqrt{\bar{\alpha}^{(t)}}\mathbf{c}^{(0)} + \sqrt{1-\bar{\alpha}^{(t)}}\boldsymbol{\epsilon}, \tag{6}$$

where $\bar{\alpha}^{(t)} = \prod_{s=1}^{t} \alpha^{(s)}$ with $\alpha^{(t)} = 1 - \beta^{(t)}$, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is drawn from a Gaussian distribution. The goal of the diffusion model is to learn the *reverse* process, progressively denoising the noisy code to recover a clean version. We parameterize the denoising network f_{diff} as a U-Net [60], which takes the noisy code $\mathbf{c}^{(t)}$ and timestep t as inputs, predicting the noise estimate $f_{\text{diff}}(\mathbf{c}^{(t)}, t)$ and removing it to yield a denoised code. The denoising network is trained with a weighted ℓ_2 objective:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{\mathbf{c},\boldsymbol{\epsilon},t}\left[\frac{1}{2}w^{(t)}\|f_{\text{diff}}(\mathbf{c}^{(t)}, t) - \boldsymbol{\epsilon}\|_2^2\right], \tag{7}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is standard Gaussian noise, $t \sim \mathcal{U}(1, T)$ is uniformly sampled from the interval $[1, T]$, and $\mathbf{c}^{(t)}$ is derived from $\mathbf{c}^{(0)}$ (Eq. (6)), where $\mathbf{c}^{(0)} \sim \{\mathbf{c}_i\}_{i=1}^N$ is sampled from the asset code library. The term $w^{(t)}$ provides time-dependent weighting. Learning the diffusion process in the latent space, rather than in high-dimensional triplane space like prior work [2, 9, 45, 65, 85], offers key advantages for likelihood-based generative modeling by: (i) focusing on essential contents of the data and (ii) operating in a computationally efficient, compact space.

Unconditional Asset Generation: Unconditional asset generation involves generating a latent code using the learned diffusion model f_{diff} and then decoding it into neural assets using the learned asset decoder f_{dec}. Sampling from the diffusion prior can be performed with various solvers (*e.g.*, DDPM [22], DDIM [67]), similar to methods used in image generation. Taking DDIM [67] for example, we begin with a random Gaussian code $\mathbf{c}^{(T)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and iteratively denoise it over T steps until reaching $t = 0$:

$$\mathbf{c}^{(t-1)} = \sqrt{\alpha^{(t-1)}}\left(\frac{\mathbf{c}^{(t)} - \sqrt{\beta^{(t)}}\tilde{\boldsymbol{\epsilon}}}{\sqrt{\alpha^{(t)}}}\right) + \sqrt{\beta^{(t-1)}}\tilde{\boldsymbol{\epsilon}}, \tag{8}$$

$$\text{s.t.} \quad \tilde{\boldsymbol{\epsilon}} = f_{\text{diff}}(\mathbf{c}^{(t)}, t), \tag{9}$$

where $\tilde{\boldsymbol{\epsilon}}$ is the noise estimate at timestep t from the denoising network f_{diff}. The final denoised code $\mathbf{c}^{(0)}$ is a sample from the learned latent distribution. We then pass this code to the asset decoder to generate the asset neural fields, $\mathbf{t} = f_{\text{dec}}(\mathbf{c}^{(0)})$, for scene composition and rendering.

Conditional / Guided Assets Generation: For conditional generation, we aim to model the conditional distribution of the latent space $p(\mathbf{c}|y)$. This can be achieved with a conditional denoising network $f_{\text{diff}}(\mathbf{c}^{(t)}, t, y)$, with condition signals such as actor class and time-of-day. As shown in Fig. 1, classifier-free diffusion guidance enables effective control over asset generation. Alternatively, for some inverse problems such as image-to-3D, conditional generation can be approximated from an unconditional model using classifier guidance [13, 68], leveraging gradients of the rendering loss w.r.t. known observations. This flexibility re-

moves the need for separate models tailored to different inverse problems. Given camera image and/or LiDAR point cloud of a new actor to reconstruct, during the sampling process (Eq. (8)) we compute the gradient of the rendering loss with respect to the current code: $\mathbf{g} = \nabla_{\mathbf{c}^{(t)}}(\mathcal{L}_{\text{rgb}} + \lambda_{\text{lid}}\mathcal{L}_{\text{lid}})$, and then steer the sampling process by updating the noisy code similar to [9, 45]:

$$\hat{\mathbf{c}}^{(t-1)} \leftarrow \mathbf{c}^{(t-1)} - \lambda \cdot \mathbf{g}, \tag{10}$$

where $\mathbf{c}^{(t-1)}$ is computed following Eq. (8) and $\hat{\mathbf{c}}^{(t-1)}$ is the updated code; λ is a small guidance weight.

4. Experiments

In this section, we first describe our experimental setup. We then compare our model to state-of-the-art reconstruction methods across multiple novel view synthesis settings. Next, we compare to SoTA generative models for 3D generation. Finally, we demonstrate conditional generation from different classes, single-image, and appearance (day/night), and show that the generated assets enhance downstream detection performance. Please refer to the supp. for implementation details and ablation of our model components.

4.1. Experimental Setup

Dataset: We conduct experiments on the PandaSet [88] dataset, which includes 103 driving scenes captured in San Francisco, with each scene spanning 8 seconds (80 frames at 10hz). The data collection platform features a 360° mechanical spinning LiDAR, a forward-facing LiDAR, along with six 1920×1080 cameras surrounding the vehicles. Object bounding boxes are provided with the dataset. As our focus is object reconstruction and generation, we extract object segmentation masks for metric evaluation. Specifically, we leverage 3D object boxes and LiDAR points to identify actor patches in the camera images, and then apply a visual foundation model [31] to generate instance masks for the actors. Our method does not require instance masks for learning. We employ them strictly to evaluate foreground-only metrics. Please see supp. for details.

Baselines: We compare our method with several state-of-the-art reconstruction and generation approaches. For reconstruction, we benchmark against per-scene NeRF-based *NeuRAD* [74] and 3D Gaussian Splatting-based *Street Gaussians* [94], as well as generalizable reconstruction methods *G3R* [10] and *PixelSplat* [8]. During evaluation, these methods take as input all source views for the full scene, except for PixelSplat, which can only handle two views. For generation, we evaluate against the GAN-based *EG3D* [7] and *DiscoScene* [91], as well as diffusion-based *SSDNeRF* [9]. As *EG3D* and *SSDNeRF* are object-centric models, we train them on our object-centric benchmark with instance masks.

4.2. Reconstruction Evaluation

Evaluation Settings and Metrics: We evaluate our method across three challenging settings: (1) *Sparse view synthesis:* Using every 10th frame for training and the remaining frames for testing, with both training and testing frames captured from the front camera. (2) *Novel camera synthesis:* Training on frames from the front camera and evaluating on frames from the front-left camera. (3) 360° *View synthesis:* Rotating actors (0°–360°) to simulate various behaviors, evaluated on front camera views. We select 7 diverse scenes to assess reconstruction performance, with the remaining 96 scenes for learning generalizable priors. For *sparse view synthesis* and *novel camera synthesis*, we report PSNR, SSIM [82], and LPIPS [103]. For 360° *view synthesis*, since no ground truth images are available, we report FID metrics [21]. As PixelSplat [8] is designed to reconstruct with two input views, we only evaluate it on the *sparse view synthesis* setting.

Sparse View Synthesis: We first compare our method against state-of-the-art approaches for sparse view synthesis in Tab. 1. Our method outperforms the baselines across all metrics. Unlike prior methods that uses 90% frames [11, 95], 75% frames [94] or 50% frames [10, 74, 98] for training, our setup is more challenging, relying on only 10% of frames for training. We found that baseline methods struggle to learn robust geometry, leading to severe artifacts in novel viewpoints. Qualitative results in Fig. 4 (top) show that our method renders camera images more accurately.

Novel Camera Synthesis: We compare our method against SoTA for novel camera synthesis in Tab. 1. A visual comparison is provided in Fig. 4 (middle). All baseline methods effectively reconstruct observed camera data (front camera). However, for new sensor poses (front-left camera), previously unobserved regions become visible, requiring the model to "hallucinate" these areas. For instance, the front-right part of the SUV in Fig. 4 (middle) is invisible in the front camera but becomes visible in front-left camera views. Reconstruction methods fail on these regions. Our method, by learning generative priors, is capable of rendering complete and high-quality simulations in this setting.

360° View Synthesis: Sensor simulation requires not only simulating novel viewpoints but also synthesizing entirely new scenarios. Consider a four-way intersection with a car crossing in front of you from a perpendicular lane. To explore a situation where that car suddenly turns sharply into your lane, we must render actors in significantly different poses than those seen in the original scene. To that end, we investigate a 360° view synthesis setting by rotating actors in the scene from 0° to 360°, simulating various

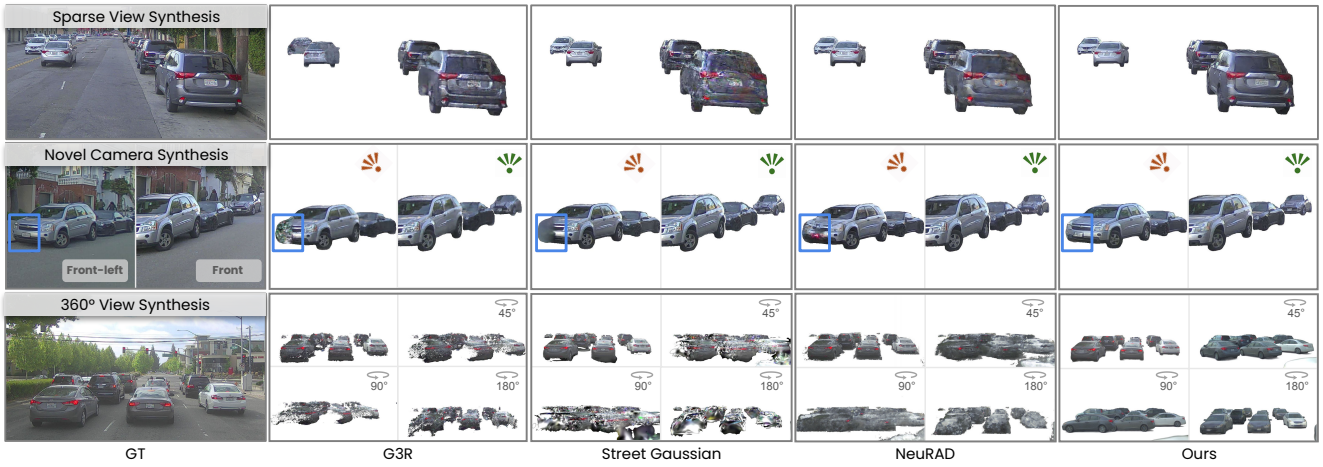

Figure 4. **Top: Sparse view synthesis.** GenAssets generalizes well on this extreme setting thanks to low-dimensional latent space learned across many scenes, while the SoTA reconstruction methods are less robust and produce noticeable visual artifacts (*e.g.*, missing, blurry or distorted appearance). **Middle: Novel camera synthesis.** We train on frames from the front camera and evaluate on frames from the front-left camera. Our method generates assets that have better extrapolation results, while retaining high quality in the original view. **Bottom:** 360° **view synthesis.** Through our multi-scene training and latent space, our method enables higher-fidelity asset completion for different object orientations compared to SoTA reconstruction-based methods.

Methods	Sparse View Synthesis			Novel Camera Synthesis			360° View Synthesis
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
PixelSplat* [8]	17.67	0.704	0.336	-	-	-	-
G3R [10]	18.37	0.711	0.255	17.40	0.723	0.221	191.92
Street Gaussian [94]	20.01	0.763	0.156	17.81	0.724	0.226	162.37
NeuRAD [74]	21.07	**0.825**	0.129	17.49	0.723	0.214	159.38
Ours	**21.34**	**0.825**	**0.113**	**18.36**	**0.805**	**0.147**	**100.28**

Table 1. **Quantitative comparison with SoTA reconstruction approaches on PandaSet.** We evaluate on sparse view synthesis (10% frames for training and remaining frames for testing), novel camera synthesis (training on front camera and testing on front-left camera), and 360° view synthesis (rotating actors from 0° to 360°). GenAssets outperforms existing SoTA per-scene or generalizable reconstruction approaches across all settings. PixelSplat* [8] (2 input views) only for sparse view synthesis evaluation.

orientations. Since ground-truth images are unavailable, we report FID [21] in Tab. 1. Qualitative results in Fig. 4 (bottom) show that our method effectively hallucinates unseen parts of the actors, whereas all baselines struggle to do so.

4.3. Generation Evaluation

Evaluation Metrics: To evaluate the generation quality, we report both Fréchet Inception Distance (FID) [21] and Kernel Inception Distance (KID) [3] scores. The metrics are calculated by comparing distributions of 10K generated images with all available images in the validation set, providing a quantitative assessment of similarity in terms of perceptual quality and realism. FID measures the distance between the mean and covariance of feature representations from real and generated image distributions. KID complements FID by measuring differences in feature embeddings through a kernel-based approach. All generated objects are rendered at a resolution of 256×256.

Unconditional Generation: We compare our method against SoTA GAN-based method EG3D [7] and DiscoScene [91], as well as diffusion-based method SSDNeRF [9] in Tab. 2. All baselines struggle with occlusion, leading to suboptimal assets. In addition, EG3D and DiscoScene suffers from mode collapse, producing limited asset variety (mainly sedans). DiscoScene struggles to differentiate foreground and background and exhibits boundary artifacts, likely due to the absence of LiDAR or mask supervision. SSDNeRF yields blurry results, potentially due to challenges posed by our real-world sparse setting. Our model improves over these baselines across all metrics. Fig. 5 shows a visual comparison.

Figure 5. **Qualitative comparison on unconditional generation.** Our methods generates more diverse, complete and higher-quality 3D assets compared to SoTA 3D generative models.

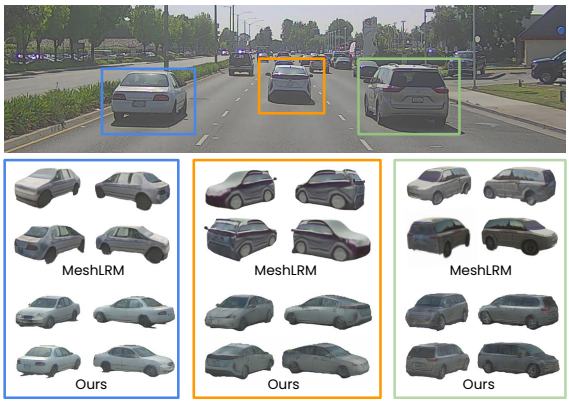

Figure 6. **Qualitative results on single-image to 3D.**

4.4. Applications

Conditional Generation: The flexibility of our framework enables various conditional generation tasks. Specifically, we freeze the learned latent codes and train a conditional diffusion model $f_{\text{diff}}(\mathbf{c}^{(t)}, t, y)$ using classifier-free guidance. We explore conditioning on fine-grained actor classes and time-of-day (day/night), with results presented in Fig. 1. We can generate complete assets for a variety classes, with intra-class shape and appearance variation.

Single Image to 3D: Our approach also enables generating 3D assets from single-view images using a rendering-guided denoising process (Eq. (10)), where the denoising gradient is directed to minimize rendering loss against the observed image. Fig. 6 shows qualitative results of single-image reconstruction for three nearby vehicles, alongside comparisons to SoTA large reconstruction model MeshLRM [83]. MeshLRM produces distorted shapes and synthetic cartoon-ish appearance for unobserved views. Our approach generates higher quality completion and is consistent with the input. Please refer to supp. for more comparisons with other SoTA large models.

Data Augmentation with GenAssets: We now showcase that our generated assets boost downstream performance when training a BEVFormer [33] 3D object detector (Tab. 3). Specifically, we augment the training dataset by swapping out existing actors with generated ones for the same scene layouts. Please see supp. for more details.

Methods	FID↓	KID↓
SSDNeRF [9]	191.30	156.13
DiscoScene [91]	138.48	116.27
EG3D [7]	80.56	38.97
Ours	**59.50**	**28.32**

Table 2. **Unconditional generation evaluation on PandaSet.** KID scores are multiplied by 10^3. GenAssets outperforms SoTA baselines on both metrics and generates higher-quality 3D assets.

	mAP↑	AP@1m↑	AP@2m↑	AP@4m↑
Real	27.08	8.58	26.99	45.67
Real + Sim	**29.32**	**9.78**	**29.18**	**49.00**

Table 3. **Data augmentation with GenAssets helps 3D detection.** We report distance-based APs [4] at $1m$, $2m$ and $4m$.

5. Conclusion

In this work, we tackled the challenge of generating high-quality and complete assets from in-the-wild LiDAR and camera data captured by a moving sensor platform. Towards this goal, we developed a "reconstruct-then-generate" approach where we first learn to reconstruct foreground actors over multiple scenes with compositional scene neural rendering and encode them to a latent space. We then train a diffusion model to operate within this latent space to enable generation. We show our method generates high-quality, complete assets for actors such as vehicles and motorcycles, outperforming both per-scene reconstruction methods and generative models. We also show that our approach can be conditioned for controllable asset generation such as on sparse sensor data, actor class, and time of day, enabling diverse and scalable content creation for simulation. Future work involves generating dynamic lighting and animation, intrinsic decomposition, and adopting more efficient scene representation and rendering techniques.

Acknowledgement: We sincerely thank the Waabi team for their invaluable assistance and support. We also thank the anonymous reviewers for their insightful suggestions.

References

[1] Titas Anciukevičius, Zexiang Xu, Matthew Fisher, Paul Henderson, Hakan Bilen, Niloy J Mitra, and Paul Guerrero. Renderdiffusion: Image diffusion for 3d reconstruction, inpainting and generation. In *CVPR*, 2023. 3

[2] Miguel Angel Bautista, Pengsheng Guo, Samira Abnar, Walter Talbott, Alexander Toshev, Zhuoyuan Chen, Laurent Dinh, Shuangfei Zhai, Hanlin Goh, Daniel Ulbricht, et al. Gaudi: A neural architect for immersive 3d scene generation. *NeurIPS*, 2022. 5

[3] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv*, 2018. 7

[4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 8

[5] Ang Cao, Justin Johnson, Andrea Vedaldi, and David Novotny. Lightplane: Highly-scalable components for neural 3d fields. *arXiv*, 2024. 3

[6] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021. 2

[7] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, 2022. 2, 3, 6, 7, 8

[8] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *CVPR*, 2024. 6, 7

[9] Hansheng Chen, Jiatao Gu, Anpei Chen, Wei Tian, Zhuowen Tu, Lingjie Liu, and Hao Su. Single-stage diffusion nerf: A unified approach to 3d generation and reconstruction. In *ICCV*, 2023. 2, 3, 5, 6, 7, 8

[10] Yun Chen, Jingkang Wang, Ze Yang, Sivabalan Manivasagam, and Raquel Urtasun. G3r: Gradient guided generalizable reconstruction. In *ECCV*, 2025. 2, 6, 7

[11] Ziyu Chen, Jiawei Yang, Jiahui Huang, Riccardo de Lutio, Janick Martinez Esturo, Boris Ivanovic, Or Litany, Zan Gojcic, Sanja Fidler, Marco Pavone, et al. Omnire: Omni urban scene reconstruction. *arXiv*, 2024. 1, 2, 6

[12] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *CVPR*, 2023. 2

[13] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 2021. 5

[14] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *CoRL*, 2017. 1

[15] Tobias Fischer, Jonas Kulhanek, Samuel Rota Bulò, Lorenzo Porzi, Marc Pollefeys, and Peter Kontschieder. Dynamic 3d gaussian fields for urban areas. *arXiv*, 2024. 1

[16] Raghudeep Gadde, Qianli Feng, and Aleix M Martinez. Detail me more: Improving gan's photo-realism of complex scenes. In *ICCV*, 2021. 5

[17] 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. *NeurIPS*, 2022. 2, 3

[18] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T Barron, and Ben Poole. Cat3d: Create anything in 3d with multi-view diffusion models. *arXiv*, 2024. 3

[19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS*, 2014. 2

[20] Jiatao Gu, Alex Trevithick, Kai-En Lin, Joshua M Susskind, Christian Theobalt, Lingjie Liu, and Ravi Ramamoorthi. Nerfdiff: Single-image view synthesis with nerf-guided distillation from 3d-aware diffusion. In *ICML*, 2023. 2, 3

[21] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 2017. 6, 7

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 3, 5

[23] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. LRM: Large reconstruction model for single image to 3d. *arXiv*, 2023. 2

[24] Qianjiang Hu, Zhimin Zhang, and Wei Hu. Rangeldm: Fast realistic lidar point cloud generation. In *ECCV*, 2025. 3

[25] Sungwon Hwang, Min-Jung Kim, Taewoong Kang, Jayeon Kang, and Jaegul Choo. VEGS: View extrapolation of urban scenes in 3d gaussian splatting using learned priors. *arXiv*, 2024. 2, 3

[26] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 4

[27] Wonbong Jang and Lourdes Agapito. Codenerf: Disentangled neural radiance fields for object categories. In *ICCV*, 2021. 2

[28] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 2

[29] Mustafa Khan, Hamidreza Fazlali, Dhruv Sharma, Tongtong Cao, Dongfeng Bai, Yuan Ren, and Bingbing Liu. Autosplat: Constrained gaussian splatting for autonomous driving scene reconstruction. *arXiv*, 2024. 1, 2

[30] Diederik P Kingma. Auto-encoding variational bayes. *arXiv*, 2013. 3, 5

[31] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, 2023. 6

[32] Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg

Shakhnarovich, and Sai Bi. Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. *arXiv*, 2023. 3

[33] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *ECCV*, 2022. 8

[34] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *CVPR*, 2023. 2, 3

[35] Jeffrey Yunfan Liu, Yun Chen, Ze Yang, Jingkang Wang, Sivabalan Manivasagam, and Raquel Urtasun. Neural scene rasterization for large scene rendering in real time. In *ICCV*, 2023. 2

[36] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *NeurIPS*, 2024. 3

[37] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *ICCV*, 2023. 3

[38] Zhen Liu, Yao Feng, Michael J Black, Derek Nowrouzezahrai, Liam Paull, and Weiyang Liu. Meshdiffusion: Score-based generative 3d mesh modeling. *arXiv*, 2023. 3

[39] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3d: Single image to 3d using cross-domain diffusion. In *CVPR*, 2024. 3

[40] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *CVPR*, 2021. 3

[41] Sivabalan Manivasagam, Ioan Andrei Bârsan, Jingkang Wang, Ze Yang, and Raquel Urtasun. Towards zero domain gap: A comprehensive study of realistic lidar simulation for autonomy testing. In *ICCV*, 2023. 1

[42] Quan Meng, Lei Li, Matthias Nießner, and Angela Dai. Lt3sd: Latent trees for 3d scene diffusion. *arXiv*, 2024. 3

[43] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 2021. 2

[44] Norman Müller, Andrea Simonelli, Lorenzo Porzi, Samuel Rota Bulò, Matthias Nießner, and Peter Kontschieder. Autorf: Learning 3d object radiance fields from single view observations. In *CVPR*, 2022. 2

[45] Norman Müller, Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulo, Peter Kontschieder, and Matthias Nießner. Diffrf: Rendering-guided 3d radiance field diffusion. In *CVPR*, 2023. 2, 3, 5, 6

[46] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting triangular 3d models, materials, and lighting from images. In *CVPR*, 2022. 2

[47] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *CVPR*, 2021. 2

[48] Evangelos Ntavelis, Aliaksandr Siarohin, Kyle Olszewski, Chaoyang Wang, Luc V Gool, and Sergey Tulyakov. Autodecoding latent 3d diffusion models. *NeurIPS*, 2023. 2, 3

[49] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *ICCV*, 2021. 2

[50] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *CVPR*, 2021. 2

[51] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *ICLR*, 2024. 3

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

[53] Ava Pun, Gary Sun, Jingkang Wang, Yun Chen, Ze Yang, Sivabalan Manivasagam, Wei-Chiu Ma, and Raquel Urtasun. Neural lighting simulation for urban scenes. In *NeurIPS*, 2023. 2

[54] Haoxi Ran, Vitor Guizilini, and Yue Wang. Towards realistic scene generation with lidar diffusion models. In *CVPR*, 2024. 3

[55] Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. Xcube: Large-scale 3d generative modeling using sparse voxel hierarchies. In *CVPR*, 2024. 3

[56] Xuanchi Ren, Yifan Lu, Hanxue Liang, Zhangjie Wu, Huan Ling, Mike Chen, Sanja Fidler, Francis Williams, and Jiahui Huang. SCube: Instant large-scale scene reconstruction using voxsplats. *arXiv*, 2024. 2

[57] Barbara Roessle, Norman Müller, Lorenzo Porzi, Samuel Rota Bulò, Peter Kontschieder, Angela Dai, and Matthias Nießner. L3dg: Latent 3d gaussian diffusion. *arXiv*, 2024. 2, 3

[58] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 2, 3, 5

[59] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, and Shalin Mehta. Lgsvl simulator: A high fidelity simulator for autonomous driving. In *ITSC*, 2020. 1

[60] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 3, 5

[61] Jay Sarva, Jingkang Wang, James Tu, Yuwen Xiong, Sivabalan Manivasagam, and Raquel Urtasun. Adv3d: Generating safety-critical 3d objects through closed-loop simulation. In *CoRL*, 2023. 1

[62] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. AirSim: High-fidelity visual and physical simula-

tion for autonomous vehicles. In *Field and service robotics*, 2018. 1

[63] Bokui Shen, Xinchen Yan, Charles R Qi, Mahyar Najibi, Boyang Deng, Leonidas Guibas, Yin Zhou, and Dragomir Anguelov. Gina-3d: Learning to generate implicit neural assets in the wild. In *CVPR*, 2023. 2

[64] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv*, 2023. 3

[65] J Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion. In *CVPR*, 2023. 3, 5

[66] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 5

[67] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv*, 2020. 3, 5

[68] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv*, 2020. 3, 5

[69] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Viewset diffusion:(0-) image-conditioned 3d generative models from 2d data. In *ICCV*, 2023. 3

[70] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *CVPR*, 2022. 1

[71] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv*, 2023. 3

[72] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *ECCV*, 2025. 3

[73] Ayush Tewari, Tianwei Yin, George Cazenavette, Semon Rezchikov, Josh Tenenbaum, Frédo Durand, Bill Freeman, and Vincent Sitzmann. Diffusion with forward models: Solving stochastic inverse problems without direct supervision. *NeurIPS*, 2023. 3

[74] Adam Tonderski, Carl Lindström, Georg Hess, William Ljungbergh, Lennart Svensson, and Christoffer Petersson. NeuRAD: Neural rendering for autonomous driving. In *CVPR*, 2024. 1, 2, 3, 4, 6, 7

[75] Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, Karsten Kreis, et al. Lion: Latent point diffusion models for 3d shape generation. *NeurIPS*, 2022. 3

[76] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *NeurIPS*, 2017. 3

[77] Jingkang Wang, Ava Pun, James Tu, Sivabalan Manivasagam, Abbas Sadat, Sergio Casas, Mengye Ren, and Raquel Urtasun. Advsim: Generating safety-critical scenarios for self-driving vehicles. In *CVPR*, 2021. 1

[78] Jingkang Wang, Sivabalan Manivasagam, Yun Chen, Ze Yang, Ioan Andrei Bârsan, Anqi Joyce Yang, Wei-Chiu Ma, and Raquel Urtasun. CADSim: Robust and scalable in-the-wild 3d reconstruction for controllable sensor simulation. In *6th Annual Conference on Robot Learning*, 2022. 2

[79] Peng Wang and Yichun Shi. Imagedream: Image-prompt multi-view diffusion for 3d generation. *arXiv*, 2023. 3

[80] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv*, 2021. 2

[81] Qitai Wang, Lue Fan, Yuqi Wang, Yuntao Chen, and Zhaoxiang Zhang. FreeVS: Generative view synthesis on free driving trajectory. *arXiv*, 2024. 2, 3

[82] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 2004. 6

[83] Xinyue Wei, Kai Zhang, Sai Bi, Hao Tan, Fujun Luan, Valentin Deschaintre, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. MeshLRM: Large reconstruction model for high-quality mesh. *arXiv*, 2024. 2, 8

[84] Rundi Wu, Ben Mildenhall, Philipp Henzler, Keunhong Park, Ruiqi Gao, Daniel Watson, Pratul P Srinivasan, Dor Verbin, Jonathan T Barron, Ben Poole, et al. Reconfusion: 3d reconstruction with diffusion priors. In *CVPR*, 2024. 2, 3

[85] Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Jingxi Xu, Philip Torr, Xun Cao, and Yao Yao. Direct3d: Scalable image-to-3d generation via 3d latent diffusion transformer. *arXiv*, 2024. 2, 3, 5

[86] Zirui Wu, Tianyu Liu, Liyi Luo, Zhide Zhong, Jianteng Chen, Hongmin Xiao, Chao Hou, Haozhe Lou, Yuantao Chen, Runyi Yang, et al. Mars: An instance-aware, modular and realistic simulator for autonomous driving. In *CAAI International Conference on Artificial Intelligence*, 2023. 1

[87] Ziyi Wu, Yulia Rubanova, Rishabh Kabra, Drew Hudson, Igor Gilitschenski, Yusuf Aytar, Sjoerd van Steenkiste, Kelsey Allen, and Thomas Kipf. Neural assets: 3d-aware multi-object scene synthesis with image diffusion models. In *NeurIPS*, 2024. 3

[88] Pengchuan Xiao, Zhenlei Shao, Steven Hao, Zishuo Zhang, Xiaolin Chai, Judy Jiao, Zesong Li, Jian Wu, Kai Sun, Kun Jiang, et al. Pandaset: Advanced sensor suite dataset for autonomous driving. In *ITSC*, 2021. 2, 6

[89] Yuwen Xiong, Wei-Chiu Ma, Jingkang Wang, and Raquel Urtasun. Learning compact representations for lidar completion and generation. In *CVPR*, 2023. 3

[90] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *arXiv*, 2024. 3

[91] Yinghao Xu, Menglei Chai, Zifan Shi, Sida Peng, Ivan Skorokhodov, Aliaksandr Siarohin, Ceyuan Yang, Yujun Shen, Hsin-Ying Lee, Bolei Zhou, et al. Discoscene: Spatially disentangled generative radiance fields for controllable 3d-aware scene synthesis. In *CVPR*, 2023. 2, 5, 6, 7, 8

[92] Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, et al. Dmv3d: Denoising multi-view diffusion using 3d large reconstruction model. *arXiv*, 2023. 3

[93] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wet-

zstein. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. *arXiv*, 2024. 3

[94] Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. Street gaussians for modeling dynamic urban scenes. In *ECCV*, 2024. 1, 2, 3, 6, 7

[95] Jiawei Yang, Boris Ivanovic, Or Litany, Xinshuo Weng, Seung Wook Kim, Boyi Li, Tong Che, Danfei Xu, Sanja Fidler, Marco Pavone, et al. Emernerf: Emergent spatial-temporal scene decomposition via self-supervision. *arXiv*, 2023. 1, 6

[96] Ze Yang, Sivabalan Manivasagam, Ming Liang, Bin Yang, Wei-Chiu Ma, and Raquel Urtasun. Recovering and simulating pedestrians in the wild. In *Conference on Robot Learning*, 2021. 2

[97] Ze Yang, Shenlong Wang, Sivabalan Manivasagam, Zeng Huang, Wei-Chiu Ma, Xinchen Yan, Ersin Yumer, and Raquel Urtasun. S3: Neural shape, skeleton, and skinning fields for 3d human modeling. In *CVPR*, 2021. 2

[98] Ze Yang, Yun Chen, Jingkang Wang, Sivabalan Manivasagam, Wei-Chiu Ma, Anqi Joyce Yang, and Raquel Urtasun. Unisim: A neural closed-loop sensor simulator. In *CVPR*, 2023. 1, 2, 3, 4, 6

[99] Ze Yang, Sivabalan Manivasagam, Yun Chen, Jingkang Wang, Rui Hu, and Raquel Urtasun. Reconstructing objects in-the-wild for realistic sensor simulation. In *ICRA*, 2023. 2

[100] Ze Yang, George Chen, Haowei Zhang, Kevin Ta, Ioan Andrei Bârsan, Daniel Murphy, Sivabalan Manivasagam, and Raquel Urtasun. Unical: Unified neural sensor calibration. In *ECCV*, 2025. 2

[101] Jason Zhang, Gengshan Yang, Shubham Tulsiani, and Deva Ramanan. NeRS: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. *NeurIPS*, 2021. 2

[102] Lunjun Zhang, Yuwen Xiong, Ze Yang, Sergio Casas, Rui Hu, and Raquel Urtasun. Copilot4d: Learning unsupervised world models for autonomous driving via discrete diffusion. *arXiv*, 2023. 3

[103] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 4, 6

[104] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In *CVPR*, 2024. 1