PartGen: Part-level 3D Generation and Reconstruction with Multi-View Diffusion Models

Minghao Chen[1,2]   Roman Shapovalov[2]   Iro Laina[1]   Tom Monnier[2]
Jianyuan Wang[1,2]   David Novotny[2]   Andrea Vedaldi[1,2]
[1]Visual Geometry Group, University of Oxford   [2]Meta AI
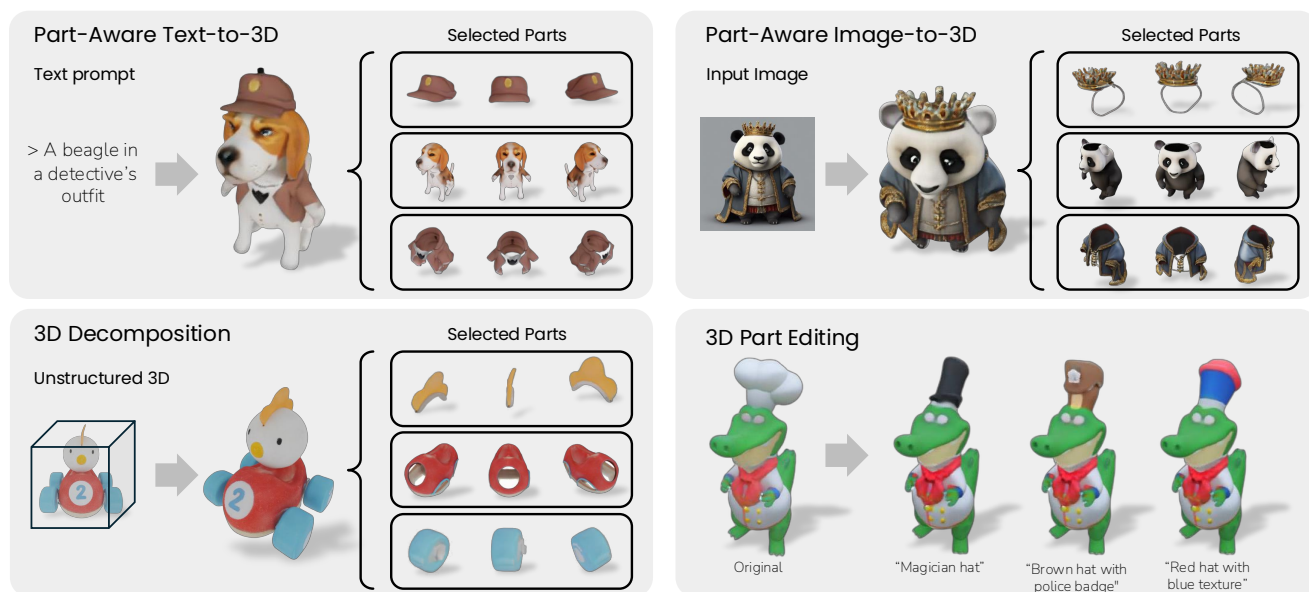
silent-chen.github.io/PartGen

Figure 1. We introduce **PartGen**, a pipeline that generates compositional 3D objects similar to a human artist. It can start from text, an image, or an existing, unstructured 3D object. It consists of a multi-view diffusion model that identifies plausible parts automatically and another that completes and reconstructs them in 3D, accounting for their context, *i.e.*, the other parts, to ensure that they fit together correctly. Additionally, PartGen enables 3D part editing based on text instructions, enhancing flexibility and control in 3D object creation.

Abstract

Text- or image-to-3D generators and 3D scanners can now produce 3D assets with high-quality shapes and textures. These assets typically consist of a single, fused representation, like an implicit neural field, a Gaussian mixture, or a mesh, without any useful structure. However, most applications and creative workflows require assets to be made of several meaningful parts that can be manipulated independently. To address this gap, we introduce PartGen, a novel approach that generates 3D objects composed of meaningful parts starting from text, an image, or an unstructured 3D object. First, given multiple views of a 3D object, generated or rendered, a multi-view diffusion model extracts a set of plausible and view-consistent part segmentations, dividing the object into parts. Then, a second multi-view diffusion model takes each part separately, fills in the occlusions, and uses those completed views for 3D reconstruction by feeding them to a 3D reconstruction network. This completion process considers the context of the entire object to ensure that the parts integrate cohesively. The generative completion model can make up for the information missing due to occlusions; in extreme cases, it can hallucinate entirely invisible parts based on the input 3D asset. We evaluate our method on generated and real 3D assets and show that it outperforms segmentation and part-extraction baselines by a large margin. We also showcase downstream applications such as 3D part editing.

Work completed during Minghao's internship at Meta.

1. Introduction

High-quality textured 3D assets can now be obtained through generation from text or images [12, 14, 18, 51, 56, 58, 76, 83], or through photogrammetry techniques [15, 63, 89]. However, the resulting objects are *unstructured*, consisting of a single, monolithic representation, such as an implicit neural field, a mixture of Gaussians, or a mesh. This is not good enough in a professional setting, where the *structure* of an asset is also of paramount importance. While there are many aspects to the structure of a 3D object (*e.g.*, the mesh topology), parts are especially important as they enable reuse, editing and animation.

In this paper, we thus consider the problem of obtaining *structured* 3D objects that are formed by a collection of meaningful *parts*, akin to the models produced by human artists. For example, a model of a person may be decomposed into its clothes and accessories, as well as various anatomical features like hair, eyes, teeth, limbs, etc. However, if the object is generated or scanned, different parts are usually 'fused' together, missing the internal surfaces and the part boundaries. This means that physically detachable parts appear glued together, with a jarring effect. Furthermore, parts carry important information and functionality that those models lack. For example, different parts may have distinct animations or different materials. Parts can also be replaced, removed, or edited independently. For instance, in video games, parts are often reconfigured dynamically, *e.g.*, to represent a character picking up a weapon or changing clothes. Due to their semantic meaning, parts are also important for 3D understanding and applications like robotics, embodied AI, and *spatial intelligence* [48, 53].

Inspired by these requirements, we introduce PartGen, a method to upgrade existing 3D generation pipelines from producing unstructured 3D objects to generating objects as compositions of meaningful 3D parts. To do this, we address two key questions: (1) how to automatically *segment* a 3D object into parts, and (2) how to extract high-quality, *complete* 3D parts even when these are only partially—or not at all—visible from the exterior of the 3D object.

Crucially, both part segmentation and completion are highly ambiguous tasks. First, since different artists may find it useful to decompose the same object in different ways, there is no 'gold-standard' segmentation for any given 3D object. Hence, a segmentation method should model the distribution of plausible part segmentations rather than a single one. Second, current 3D reconstruction and generation methods only model an object's visible outer surface, omitting inner or occluded parts. Therefore, decomposing an object into parts often requires completing these parts or even entirely hallucinating them.

To model this ambiguity, we base part segmentation and reconstruction on 3D generative models. We note that most state-of-the-art 3D generation pipelines [12, 14, 18, 39, 51,

56, 58, 76, 83] start by generating several consistent 2D views of the object, and then apply a 3D reconstruction network to those images to recover the 3D object. We build upon this two-stage scheme to address both part segmentation and reconstruction ambiguities.

In the first stage, we cast part segmentation as a *stochastic multi-view-consistent colouring problem*, leveraging a multi-view image generator fine-tuned to produce colour-coded segmentation maps across multiple views of a 3D object. We do not assume any explicit or even deterministic taxonomy of parts; the segmentation model is learned from a large collection of artist-created data, capturing how 3D artists decompose objects into parts. The benefits of this approach are twofold. First, it leverages an image generator which is already trained to be view-consistent. Second, a generative approach allows for multiple plausible segmentations by simply re-sampling from the model. We show that this process results in better segmentation than that obtained by fine-tuning a model like SAM [35] or SAM2 [70] for the task of multi-view segmentation: while the latter can still be used, our approach better captures the artists' intent.

For the second problem, namely reconstructing a segmented part in 3D, an obvious approach is to mask the part within the available object views, and then use a 3D reconstructor network to recover the part in 3D. However, when the part is heavily occluded, this task amounts to *amodal reconstruction*, which is highly ambiguous and thus badly addressed by the deterministic reconstructor network. Instead, and this is our core contribution, we propose to tune another multi-view generator to *complete* the views of the part while *accounting for the context* of the object as a whole. In this manner, the parts can be reconstructed reliably even if they are only partially visible, or even not visible, in the original input views. Furthermore, the resulting parts fit together well and, when combined, form a coherent 3D object.

We show that PartGen can be applied to different input modalities. Starting from text, an image, or a areal-world 3D scan, PartGen can generate 3D assets with meaningful parts. We assess our method empirically on a large collection of 3D assets produced by 3D artists or scanned, both quantitatively and qualitatively. We also demonstrate that PartGen can be easily extended to the 3D part editing task.

2. Related Work

3D generation from text and images. The problem of generating 3D assets from text or images has been thoroughly studied in the literature. Some authors have built generators from scratch. For instance, CodeNeRF [30] learns a latent code for NeRF in a Variational Autoencoder fashion, and Shap-E [31] and 3DGen [21] does so using latent diffusion, PC^2 [55] and Point-E [62] diffuse a point cloud, and MosaicSDF a semi-explicit SDF-based representation [94]. However, 3D training data is scarce, which

makes it difficult to train text-based generators directly.

DreamFusion [65] demonstrated for the first time that 3D assets can be extracted from T2I diffusion models with *Score Distillation Sampling* (SDS) loss. Variants of Dream-Fusion explore representations like hash grids [41, 66], meshes [41] and 3D Gaussians (3DGS) [8, 79, 96], tweaks to the SDS loss [27, 85, 87, 104], conditioning on an input image [54, 66, 78, 80, 98], and regularizing normals or depth [68, 74, 78].

Other works focus on improving the 3D awareness of the T2I model, simplifying extracting a 3D output and eschewing the need for slow SDS optimization. Inspired by 3DIM [88], Zero-1-to-3 [47] fine-tunes the 2D generator to output novel views of the object. Two-stage approaches [6, 9, 18, 22, 23, 25, 45, 49, 50, 56, 74, 81, 86, 91–93] take the output of a text- or image-to-multi-view model that generates multiple views of the object and reconstruct the latter using multi-view reconstruction methods like NeRF [59] or 3DGS [32]. Other approaches reduce the number of input views generated and learn a fast feed-forward network for 3D reconstruction. Perhaps the most notable example is Instant3D [39] based on the *Large Reconstruction Model* (LRM) [26]. Recently, there are works focusing on 3D compositional generation [11, 40, 64, 105]. D3LL [17] learns 3D object composition through distilling from a 2D T2I generator. ComboVerse [7] starts from a single image, but mostly at the levels of different objects instead of their parts, performs single-view inpainting and reconstruction, and uses SDS optimization for composition.

3D segmentation. Our work decomposes a given 3D object into parts. Several works have considered segmenting 3D objects or scenes represented in an unstructured manner, lately as neural fields or 3D Gaussian mixtures. Semantic-NeRF [101] was the first to fuse 2D semantic segmentation maps in 3D with neural fields. DFF [36] and N3F [84] propose to map 2D features to 3D fields, allowing their supervised and unsupervised segmentation. LERF [33] extends this concept to language-aware features like CLIP [69]. Contrastive Lift [2] considers instead instance segmentation, fusing information from several independently-segmented views using a contrastive formulation. GARField [34] and OminiSeg3D [97] consider that concepts exist at different levels of scale, which they identify with the help of SAM [35]. LangSplat [67] leverages both CLIP and SAM, creating distinct 3D language fields to model each SAM scale explicitly, while N2F2 [3] automates binding the correct scale to each concept. Neural Part Priors [4] completes and decomposes 3D scans with learned part priors in a test-time optimization manner. Finally, Uni3D [102] learns a 'foundation' model for 3D point clouds that can perform zero-shot segmentation.

Primitive-based representations. Some authors proposed to represent 3D objects as a mixture of primitives [99], which can be seen as related to parts, although they are usually non-semantic. For example, SIF [19] represents an occupancy function as a 3D Gaussians mixture. LDIF [20] uses the Gaussians to window local occupancy functions implemented as neural fields [57]. Neural Template [28] and SPAGHETTI [1] learn to decompose shapes in a similar manner using an auto-decoding setup. SALAD [37] uses SPAGHETTI as the latent representation for a diffusion-based generator. PartNeRF [82] is conceptually similar, but builds a mixture of NeRFs. NeuForm [42] and Diff-Facto [61] learn representations that afford part-based control. DBW [60] decomposes real-world scenes with textured superquadric primitives.

Semantic part-based representations. Other authors have considered 3D parts that are semantic. PartSLIP [46] and PartSLIP++ [103] use vision-language model to segment objects into parts using point clouds as representation. Part123 [44] is conceptually similar to Contrastive Lift [2], but applied to object than scenes, and to the output of a monocular reconstruction network instead of NeRF.

In this paper, we address a problem different from the ones above. We generate compositional 3D objects from various modalities using multi-view diffusion models for segmentation and completion. Parts are meaningfully segmented, fully reconstructed, and correctly assembled. We handle the ambiguity of these tasks in a generative way.

3. Method

This section introduces PartGen, our framework for generating 3D objects that are fully decomposable into *complete* 3D parts. Each part is a distinct, human-interpretable, and self-contained element, representing the 3D object compositionally. PartGen can take different modalities as input (text prompts, image prompts, or 3D assets) and performs part segmentation and completion by repurposing a powerful multi-view diffusion model for these two tasks. An overview of PartGen is shown in Figure 2.

The rest of the section is organised as follows. In Sec. 3.1, we introduce the necessary background on multi-view diffusion and how PartGen can be applied to text, image, or 3D model inputs briefly. Then, in Secs. 3.2 to 3.4 we describe how PartGen automatically segments, completes, and reconstructs meaningful parts in 3D.

3.1. Background on 3D generation

First, we provide essential background on multi-view diffusion models for 3D generation [39, 74, 76]. These methods usually adopt a two-stage approach to 3D generation.

In the first stage, given a prompt y, an image generator Φ outputs several 2D views of the object from different van-
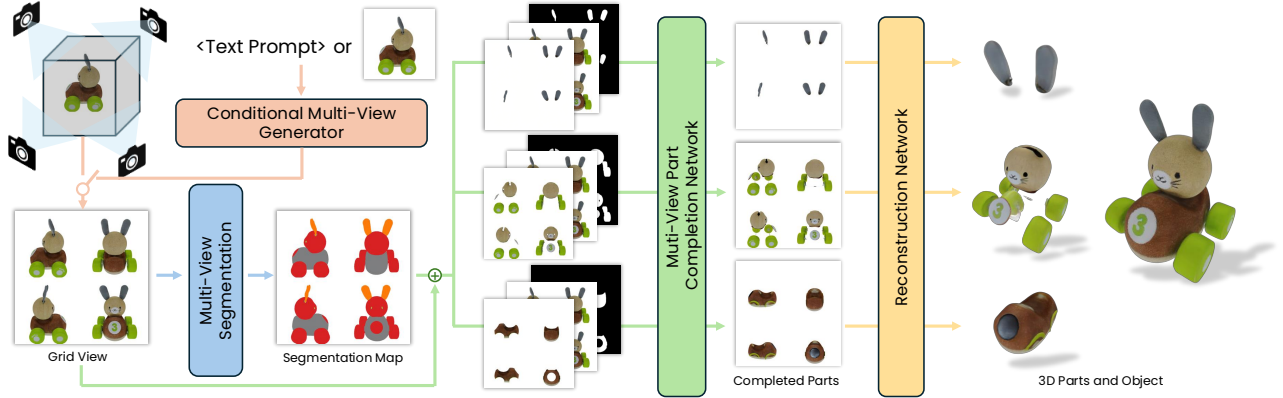
Figure 2. **Overview of PartGen.** Our method begins with text, single images, or existing 3D objects to obtain an initial grid view of the object. This view is then processed by a diffusion-based segmentation network to achieve multi-view consistent part segmentation. Next, the segmented parts, along with contextual information, are input into a multi-view part completion network to generate a fully completed view of each part. Finally, a pre-trained reconstruction model generates the 3D parts.

tage points. Depending on the nature of y, the network Φ is either a text-to-image (T2I) model [39, 74] or a image-to-image (I2I) one [73, 86]. These are fine-tuned to output a single 'multi-view' image $I \in \mathbb{R}^{3 \times 2H \times 2W}$, where views from the four cardinal directions around the object are arranged into a 2×2 grid. This model thus provides a probabilistic mapping $I \sim p(I \mid \Phi, y)$. The 2D views I are subsequently passed to a Reconstruction Model (RM) [39, 76, 90] Ψ, *i.e.*, a neural network that reconstructs the 3D object $\mathbf{L}$ in both shape and appearance. Compared to direct 3D generation, this two-stage paradigm takes full advantage of an image generation model pre-trained on internet-scale 2D data.

This approach is general and can be applied with various implementations of image-generation and reconstruction models. Our work in particular follows a setup similar to AssetGen [76]. Specifically, we obtain Φ by finetuning a pre-trained text-to-image diffusion model with an architecture similar to Emu [13], a diffusion model in a 8-channel latent space, the mapping to which is provided by a specially trained variational autoencoder (VAE). The detailed fine-tuning strategy can be found in Sec. 4.4 and supplementary material. When the input is a 3D model, we render multiple views to form the grid view. For the RM Ψ we use LightplaneLRM [5], trained on our dataset.

3.2. Multi-view part segmentation

The first major contribution of our paper is a method for segmenting an object into its constituent parts. Inspired by multi-view diffusion approaches, we frame object decomposition into parts as a *multi-view segmentation* task, rather than as direct 3D segmentation. At a high-level, the goal is to map I to a collection 2D masks $M^1, \dots, M^S \in \{0, 1\}^{2H \times 2W}$, one for each visible part of the object. Both image I and masks M_i are multi-view grids.

Addressing 3D object segmentation through the lens of multi-view diffusion offers several advantages. First, it al-

lows us to repurpose existing multi-view models Φ, which, as described in Sec. 3.1, are already pre-trained to produce multi-view consistent generations in the RGB domain. Second, it integrates easily with established multi-view frameworks. Third, decomposing an object into parts is an inherently non-deterministic, ambiguous task as it depends on the desired verbosity level, individual preferences, and artistic intent. By learning this task with probabilistic diffusion models, we can effectively capture and model this ambiguity. We thus train our model on a curated dataset of artist-created 3D objects, where each object $\mathbf{L}$ is annotated with a possible decomposition into 3D parts, $\mathbf{L} = (\mathbf{S}^1, \dots, \mathbf{S}^S)$. The dataset details are provided in Sec. 3.5.

Consider that the input is a multi-view image I, and the output is a set of multi-view part masks $M^1, M^2, \dots, M^S$. To finetune our multi-view image generators Φ for mask prediction, we quantize the RGB space into Q different colors $c_1, \dots, c_Q \in [0, 1]^3$. For each training sample $\mathbf{L} = (\mathbf{S}^k)_{k=1}^S$, we assign colors to the parts, mapping part $\mathbf{S}^k$ to color c_{π_k}, where π is a random permutation on $\{1, \dots, Q\}$ (we assume that $Q \geq S$). Given this mapping, we render the segmentation map as a multi-view RGB image $C \in [0, 1]^{3 \times 2H \times 2W}$ (Fig. 4). Then, we fine-tune Φ to (1) take as conditioning the multi-view image I, and (2) to generate the color-coded multi-view segmentation map C, hence sampling a distribution $C \sim p(C \mid \Phi_{\text{seg}}, I)$.

This approach can produce alternative segmentations by simply re-running Φ_{seg}, which is stochastic. It further exploits the fact that Φ_{seg} is stochastic to discount the specific 'naming' or coloring of the parts, which is arbitrary. Naming is a technical issue in instance segmentation which usually requires ad-hoc solutions, and here is solved 'for free'.

To extract the segments at test time, we sample the image C and simply quantize it based on the reference colors $c_1, \dots, c_Q$, discarding parts that contain only a few pixels.
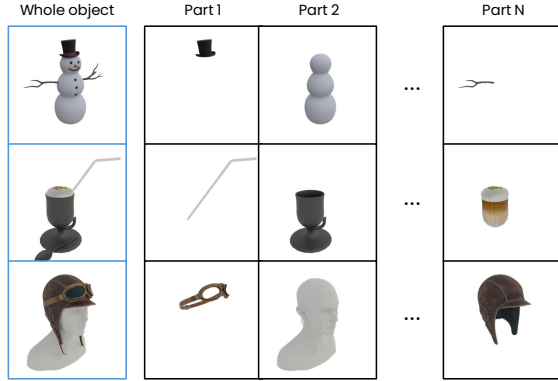
Figure 3. **Training data.** We obtain a dataset of 3D objects decomposed into parts from assets created by artists. These come 'naturally' decomposed into parts according to the artist's design.

Implementation details. The network Φ_{seg} has the same architecture as the network Φ with some changes to allow conditioning on the multi-view image I: we encode it into latent space with the VAE and stack it with the noised latent as the input to the diffusion network.

3.3. Contextual part completion

The method so far has produced a multi-view image I of the 3D object along with 2D segments $M^1, M^2, \ldots, M^S$. What remains is to convert those into the *full* 3D part reconstructions. Given a mask M, in principle we could simply submit the masked image $I \odot M$ to the RM Ψ to obtain a 3D reconstruction of the part, *i.e.*, $\hat{\mathbf{S}} = \Psi(I \odot M)$. However, in multi-view images, some parts can be heavily occluded by other parts and, in extreme cases, entirely invisible. While we could train the RM to handle such occlusions directly, in practice this does not work as part completion is inherently a stochastic problem, whereas the RM is deterministic.

To handle this ambiguity, we repurpose yet again the multi-view generator Φ, this time to perform part completion. The latter model is able to generate a 3D object from text or single image, so, properly fine-tuned, it should be able to hallucinate any missing portion of a part.

Formally, we consider fine-tuning Φ to sample a view $J \sim p(J \mid I \odot M)$, mapping the masked image $I \odot M$ to the completed multi-view image J of the part. However, we note that sometimes parts are barely visible, so the masked image $I \odot M$ provides very little information. Furthermore, we need the generated part to *fit well with the other parts and the whole object*. Hence, we provide to the model also the un-masked image I for *context*. Thus, condition $p(J \mid I \odot M, I, M)$ on the masked image $I \odot M$, the unmasked image I, and the mask M. The importance of the context I increases with the extent of the occlusion.

Implementation details. The network architecture resembles that of Sec. 3.2, but extends the conditioning, motivated

by the inpainting setup in [71]. We apply the pre-trained VAE separately to the masked image $I \odot M$ and context image I, yielding 2×8 channels, and stack them with the 8D noise image and the unencoded part mask M to obtain the 25-channel input to the diffusion model. Example results are shown in Figure 5.

3.4. Part reconstruction

Given a multi-view part image J, the final step is to reconstruct the part in 3D. Because the part views are now complete and consistent, we can simply use the RM to obtain a predicted reconstruction $\hat{\mathbf{S}} = \Psi(J)$ of the part. We found that the model does not require special finetuning to move from objects to their parts, so any good quality reconstruction model can be plugged into our pipeline directly.

3.5. Training data

To train our models, we require a dataset of 3D models consisting of multiple parts. We have built this dataset from a collection of 140k 3D-artist generated assets that we licensed for AI training from a commercial source. Each asset $\mathbf{L}$ is stored as a GLTF scene that contains, in general, several watertight meshes $(\mathbf{S}^1, \ldots, \mathbf{S}^S)$ that often align with semantic parts due to being created by a human who likely aimed to create an editable asset. Example objects from the dataset are shown in Fig. 3. We preprocess data differently for each of the three models we fine tuned.

Multi-view generator data. To train the multi-view generator models Φ, first of all, we have to render the target multi-view images I consisting of 4 views to the full object. Following Instant3D [39], we rendered shaded colours I from the 4 views from the orthogonal azimuths and $20°$ elevation and arranged them in a 2×2 grid. In case of *text conditioning*, training data consists of the pairs $\{(I_n, y_n)\}_{n=1}^N$ of multi-view images and their text captions Following Asset-Gen [76], we choose 10k highest quality assets and generated their text captions using CAP3D-like pipeline [52] that used LLAMA3 model [16]. In case of *image conditioning*, we use all 140k models, and the conditioning y_n comes in form of single renders from a randomly sampled direction (not just one of the four used in I_n).

Part segmentation and completion data. To train the part segmentation and completion networks, we need to additionally render the multi-view part images and their depth maps. Since different creators have different ideas on part decomposition, we filter the dataset to avoid having excessively granular parts which likely lack semantic meaning. To this end, we first cull the parts that take less than 5% of the object volume, and then remove the assets that have more than 10 parts or consist of a single monolithic part. This results in the dataset of 45k objects contain the total of 210k parts. Given the asset $\mathbf{L} = (\mathbf{S}^1, \ldots, \mathbf{S}^S)$, we ren-

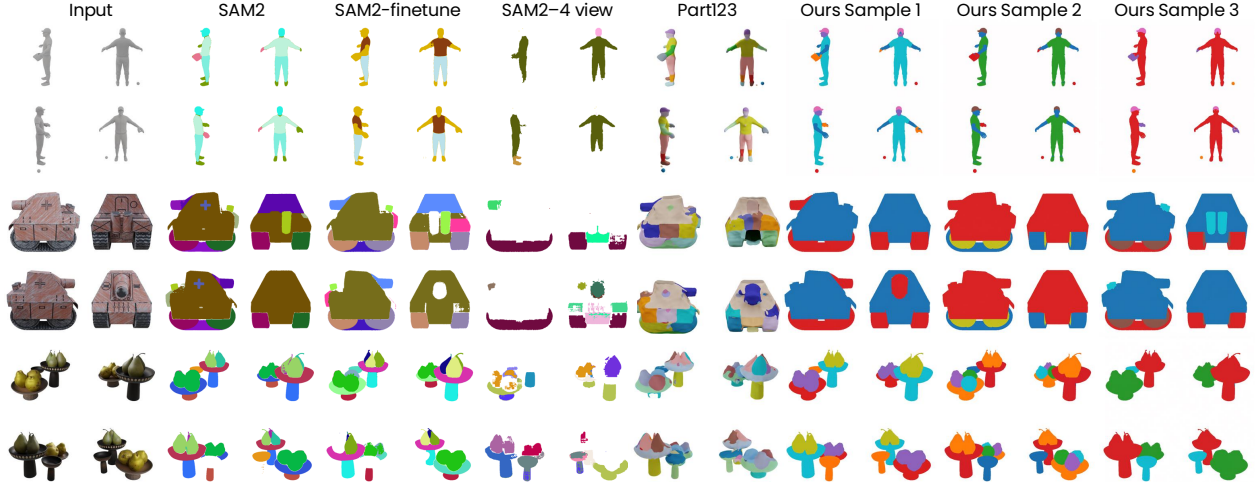| Input | SAM2 | SAM2-finetune | SAM2−4 view | Part123 | Ours Sample 1 | Ours Sample 2 | Ours Sample 3 |

Figure 4. **Examples of automatic multi-view part segmentations.** By running our method several times, we obtain different segmentations, covering the space of artist intents.

	Automatic		**Seeded**	
Method	mAP$_{50}\uparrow$	mAP$_{75}\uparrow$	mAP$_{50}\uparrow$	mAP$_{75}\uparrow$
Part123 [44]	11.5	7.4	10.3	6.5
SAM2† [70]	20.3	11.8	24.6	13.1
SAM2* [70]	37.4	27.0	44.2	30.1
SAM2 [70]	35.3	23.4	41.4	27.4
PartGen (1 sample)	45.2	32.9	44.9	33.5
PartGen (5 samples)	54.2	33.9	51.3	32.9
PartGen (10 samples)	**59.3**	**38.5**	**53.7**	**35.4**

Table 1. **Segmentation results.** SAM2* is fine-tuned our data and SAM2† is fine-tuned for multi-view segmentation.

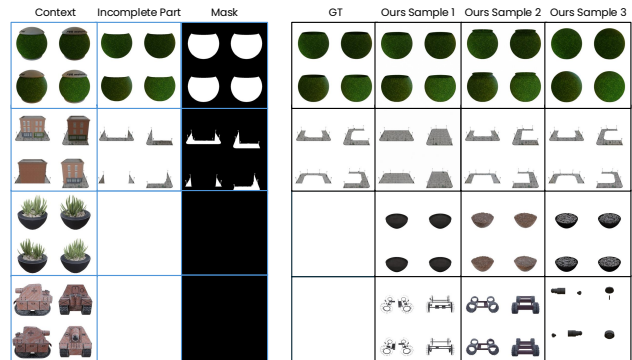| Context | Incomplete Part | Mask | | GT | Ours Sample 1 | Ours Sample 2 | Ours Sample 3 |

Figure 5. **Qualitative results of part completion.** The images with blue borders are the inputs. Our algorithm produces various plausible outputs across different runs. Even if given an empty part, PartGen attempts to generate internal structures inside the object, such as sand or inner wheels.

der a set of multi-view images $\{J^s\}_{s=1}^{S}$ (shown in Fig. 3) and the corresponding depth maps $\{\delta^s\}_{s=1}^{S}$ from the same viewpoints as above.

The *segmentation diffusion network* is trained on the dataset of pairs $\{(I_n, \mathbf{M}_n)\}_{n=1}^{N}$, where the segmentation map $\mathbf{M} = [M^k]_{k=1}^{S}$ is a stack of multi-view binary part masks $M^k \in \{0,1\}^{2H \times 2W}$. Each mask shows the pixels where the appropriate part is visible in I: $M_{i,j}^k = [k = \mathrm{argmin}_l \delta_{i,j}^l]$, where $k, l \in \{1, \ldots, S\}$ and brackets denote Iverson brackets. The *part completion network* is trained on the dataset of triplets $\{(I_{n'}, J_{n'}, M_{n'})\}_{n'=1}^{N'}$. All the components are produces in the way described above.

4. Experiments

Evaluation protocol. We first individually evaluate the two main components of our pipeline, namely part segmentation (Sec. 4.1) and part completion and reconstruction (Sec. 4.2). We then evaluate how well the decomposed reconstruction matches the original object (Sec. 4.3). For all experiments, we use the held out 100 objects from the dataset described in Sec. 3.5.

4.1. Part segmentation

Evaluation protocol. We set up two settings for the segmentation tasks. One is *automatic part segmentation*, where the input is the multi-view image I and requires the method to output all parts of the object $M^1, \ldots, M^S$. The other is *seeded segmentation*, where we assume that users give a point as an additional input for a specific mask. Now the segmentation algorithm is regarded as a black box $\hat{\mathbf{M}} = \mathcal{A}(I)$ mapping the multi-view image I to a ranked list of N part segmentations (which can in general partially overlap). This ranked list is obtained by scoring candidate regions and removing redundant ones. See the sup. mat. for more details. We then match these segments to the ground-truth segments M_k and report *mean Average Precision* (mAP). This precision can be low in practice due to the inherent ambiguity of the problem: many of the parts predicted by the algorithm will not match any particular artist's choice.

Method	Compl.	Multi-view	Context	View completion J			3D reconstruction S		
				CLIP↑	LPIPS↓	PSNR↑	CLIP↑	LPIPS↓	PSNR↑
Oracle ($\hat{J} = J$)	GT	—	—	1.0	0.0	∞	0.957	0.027	18.91
PartGen ($\hat{J} = \mathcal{B}(I \odot M, I)$)	✓	✓	✓	**0.974**	**0.015**	**21.38**	**0.936**	**0.039**	**17.16**
w/o context[†] ($\hat{J} = \mathcal{B}(I \odot M)$)	✓	✓	✗	0.951	0.028	16.80	0.923	0.046	14.83
single view[‡] ($\hat{J}_v = \mathcal{B}(I_v \odot M_v, I_v)$)	✓	✗	✓	0.944	0.031	15.92	0.922	0.051	13.25
None ($\hat{J} = I \odot M$)	✗	—	—	0.932	0.039	13.24	0.913	0.059	12.32

Table 2. **Part completion results.** We first evaluate view part completion by computing scores w.r.t. the ground-truth multi-view part image J. Then, we evaluate 3D part reconstruction by reconstructing each part **S** and rendering it. See text for details.
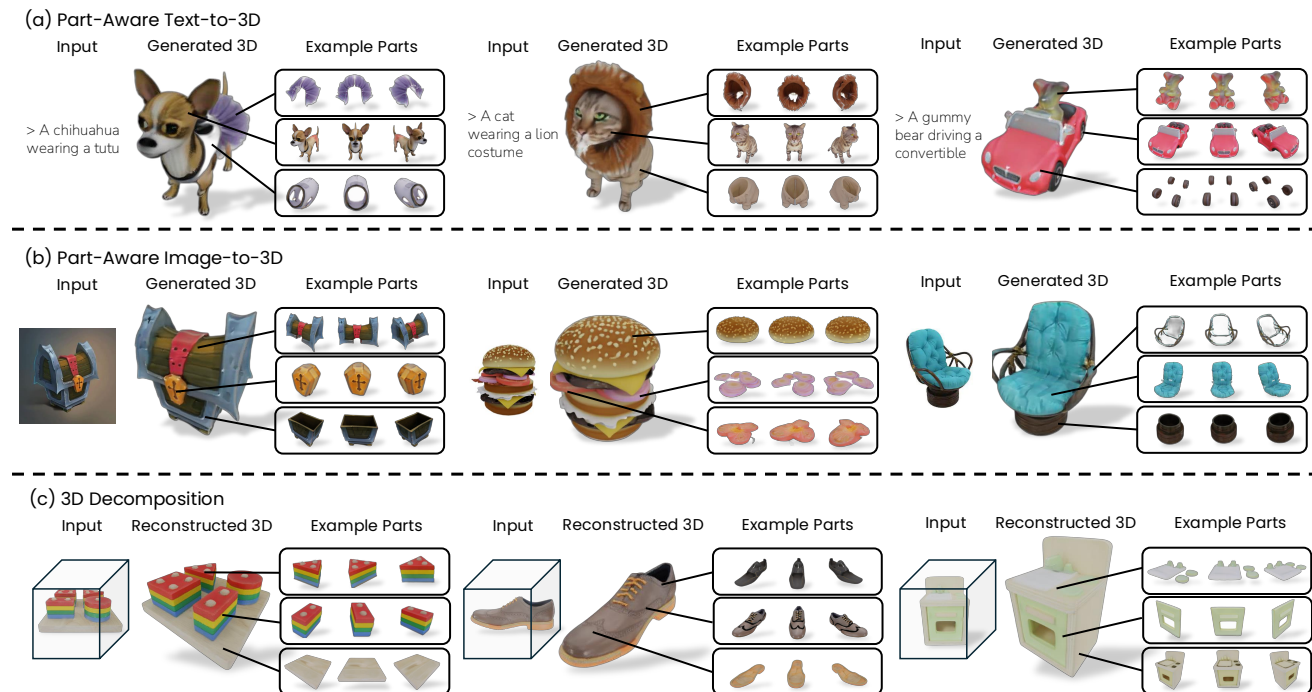

Figure 6. **Examples of applications.** PartGen can effectively generate or reconstruct 3D objects with meaningful and realistic parts in different scenarios: a) Part-aware text-to-3D generation; b) Part-aware image-to-3D generation; c) 3D decomposition.

Baselines. We consider the original and fine-tuned SAM2 [70] as our baselines for multi-view segmentation. We fine-tune SAM2 in two different ways. First, we fine-tune SAM2's mask decoder on our dataset, given the ground truth masks and randomly selected seed points for different views. Second, we concatenate the four orthogonal views in a multi-view image I and fine-tune SAM2 to predict the multi-view mask **M** (in this case, the seed point randomly falls in one of the views). SAM2 produces three regions for each input image and seed point. For automatic segmentation, we seed SAM2 with a set of query points spread over the object, obtaining three different regions for each seed point. For seeded segmentation, we simply return the regions that SAM2 outputs for the given seed point. We also provide a comparison with recent work, Part123 [44].

Results. We report the results in Tab. 1. As shown in the table, mAP results for our method are *much* higher than others, including SAM2 fine-tuned on our data. This is pri-

marily because of the ambiguity of the segmentation task, which is better captured by our generator-based approach. We further provide qualitative results in Fig. 4.

4.2. Part completion and reconstruction

We utilize the same test data as in Sec. 4.1, forming tuples $(\mathbf{S}, I, M^k, J^k)$ consisting of the 3D object part **S**, the full multi-view image I, the part mask M^k and the multi-view image J^k of the part, as described in Section 3.5. We choose one random part index k per model, and will omit it from the notation below to be more concise.

Evaluation protocol. The completion algorithm and its baselines are treated as a black box $\hat{J} = \mathcal{B}(I \odot M, I)$ that predicts the completed multi-view image $\hat{J}$. We then compare $\hat{J}$ to the ground-truth render J using Peak Signal to Noise Ratio (PSNR) of the foreground pixels, Learned Perceptual Image Patch Similarity (LPIPS) [100], and CLIP similarity [69]. The latter is an important metric since the

Figure 7. **3D part editing.** We can edit the appearance and shape of the 3D objects with text prompt.

Method	CLIP↑	LPIPS↓	PSNR↑
PartGen ($\hat{\mathbf{L}} = \bigcup_k \Phi(\hat{J}_k)$)	0.952	0.065	20.33
Unstructured ($\hat{\mathbf{L}} = \Phi(I)$)	0.955	0.064	20.47

Table 3. **Model reassembling result.** The quality of 3D reconstruction of the object as a whole is close to that of the part-based compositional reconstruction, which proves that the predicted parts fit together well.

completion task is highly ambiguous, and thus evaluating *semantic* similarity can provide additional insights. We also evaluate the quality of the reconstruction of the predicted completions by comparing the reconstructed object part $\hat{\mathbf{S}} = \Phi(\hat{J})$ to the ground-truth part $\mathbf{S}$ using the same metrics, but averaged after rendering the part to four random novel viewpoints.

Results. We compare our part completion algorithm ($\hat{J} = \mathcal{B}(I \odot M, I)$) to several baselines and the oracle, testing using no completion ($\hat{J} = I \odot M$), omitting context ($\hat{J} = \mathcal{B}(I \odot M)$), completing single views independently ($\hat{J}_v = \mathcal{B}(I_v \odot M_v, I_v)$), and the oracle ($\hat{J} = J$). The latter provides the upper-bound on the part reconstruction performance, where the only bottleneck is the RM.

As shown in the table Tab. 2, our model largely surpasses the baselines. Both joint multi-view reasoning and contextual part completion are important for good performance. We further provide qualitative results in Fig. 5.

4.3. Reassembling parts

Evaluation protocol. Starting from multi-view image I of a 3D object $\mathbf{L}$, we run the segmentation algorithm to obtain segmentation $(\hat{M}^1, \dots, \hat{M}^S)$, reconstruct each 3D part as

$\hat{\mathbf{S}}^k = \Phi(\hat{J}^k)$, and reassemble the 3D object $\hat{\mathbf{L}}$ by merging the 3D parts $\{\hat{\mathbf{S}}^1, \dots, \hat{\mathbf{S}}^N\}$. We then compare $\hat{\mathbf{L}} = \bigcup_k \Phi(\hat{J}_k)$ to the unsegmented reconstruction $\hat{\mathbf{L}} = \Phi(I)$ using the same protocol as for parts.

Results. Table 3 shows that our method achieves performance comparable to directly reconstructing the objects using the RM ($\hat{\mathbf{L}} = \Phi(I)$), with the additional benefit of producing the reconstruction structured into parts, which are useful for downstream applications such as editing.

4.4. Applications

Part-aware text-to-3D generation. First, we apply Part-Gen to part-aware text-to-3D generation. We train a text-to-multi-view generator similar to [76], which takes a text prompt as input and outputs a grid of four views. For illustration, we use the prompts from DreamFusion [65]. As shown in Fig. 6, PartGen can effectively generate 3D objects with distinct and completed parts, even in challenging cases with heavy occlusions, such as the gummy bear. Additional examples are provided in the supp. mat.

Part-aware image-to-3D generation. Next, we consider part-aware image-to-3D generation. Building upon the text-to-multi-view generator, we further fine-tune the generator to accept images as input with a strategy similar to [95]. Further training details are provided in supplementary materials. Results are shown in Fig. 6 demonstrating that PartGen is successful in this case as well.

Real-world 3D object decomposition. PartGen can also decompose real-world 3D objects. We show this using objects from Google Scanned Objects (GSO) [15] for this purpose. Given a 3D object from GSO, we render different views to obtain a an image grid and then apply PartGen as above. The last row of Figure 6 shows that PartGen can effectively decompose real-world 3D objects too.

3D part editing. Finally, we show that once the 3D parts are decomposed, they can be further modified through text input. As illustrated in Fig. 7, a variant of our method enables effective editing of the shape and texture of the parts based on textual prompts. The details of the 3D editing model are provided in supplementary materials.

5. Conclusion

We have introduced PartGen, a novel approach to generate or reconstruct compositional 3D objects from text, images, or unstructured 3D objects. PartGen can reconstruct in 3D parts that are even minimally visible, or not visible at all, utilizing the guidance of a specially-designed multi-view diffusion prior. We have also shown several application of PartGen, including text-guided part editing. This is a promising step towards the generation of 3D assets that are more useful in professional workflows.

References

[1] Hertz Amir, Perel Or, Giryes Raja, Sorkine-Hornung Olga, and Cohen-Or Daniel. SPAGHETTI: editing implicit shapes through part aware generation. In *ACM Transactions on Graphics*, 2022. 3

[2] Yash Sanjay Bhalgat, Iro Laina, Joao F. Henriques, Andrea Vedaldi, and Andrew Zisserman. Contrastive Lift: 3D object instance segmentation by slow-fast contrastive fusion. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 3

[3] Yash Sanjay Bhalgat, Iro Laina, Joao F. Henriques, Andrew Zisserman, and Andrea Vedaldi. N2F2: Hierarchical scene understanding with nested neural feature fields. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 3

[4] Aleksei Bokhovkin and Angela Dai. Neural part priors: Learning to optimize part-based object completion in rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9032–9042, 2023. 3

[5] Ang Cao, Justin Johnson, Andrea Vedaldi, and David Novotny. Lightplane: Highly-scalable components for neural 3d fields. *arXiv preprint arXiv:2404.19760*, 2024. 4, 2

[6] Eric R. Chan, Koki Nagano, Matthew A. Chan, Alexander W. Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. Generative novel view synthesis with 3D-aware diffusion models. In *Proc. ICCV*, 2023. 3

[7] Yongwei Chen, Tengfei Wang, Tong Wu, Xingang Pan, Kui Jia, and Ziwei Liu. Comboverse: Compositional 3d assets creation using spatially-aware diffusion guidance. *arXiv preprint arXiv:2403.12409*, 2024. 3

[8] Zilong Chen, Feng Wang, and Huaping Liu. Text-to-3D using Gaussian splatting. *arXiv*, 2309.16585, 2023. 3

[9] Zilong Chen, Yikai Wang, Feng Wang, Zhengyi Wang, and Huaping Liu. V3D: Video diffusion models are effective 3D generators. *arXiv*, 2403.06738, 2024. 3

[10] Zheng Chong, Xiao Dong, Haoxiang Li, Shiyue Zhang, Wenqing Zhang, Xujie Zhang, Hanqing Zhao, and Xiaodan Liang. Catvton: Concatenation is all you need for virtual try-on with diffusion models. *arXiv preprint arXiv:2407.15886*, 2024. 1

[11] Dana Cohen-Bar, Elad Richardson, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. Set-the-scene: Global-local training for generating controllable nerf scenes. In *Proc. ICCV Workshops*, 2023. 3

[12] CSM. CSM text-to-3D cube 2.0, 2024. 2

[13] Xiaoliang Dai, Ji Hou, Chih-Yao Ma, Sam S. Tsai, Jialiang Wang, Rui Wang, Peizhao Zhang, Simon Vandenhende, Xiaofang Wang, Abhimanyu Dubey, Matthew Yu, Abhishek Kadian, Filip Radenovic, Dhruv Mahajan, Kunpeng Li, Yue Zhao, Vladan Petrovic, Mitesh Kumar Singh, Simran Motwani, Yi Wen, Yiwen Song, Roshan Sumbaly, Vignesh Ramanathan, Zijian He, Peter Vajda, and Devi Parikh. Emu: Enhancing image generation models using photogenic needles in a haystack. *CoRR*, abs/2309.15807, 2023. 4, 1

[14] Deemos. Rodin text-to-3D gen-1 (0525) v0.5, 2024. 2

[15] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022. 2, 8

[16] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, and Kevin Stone. The Llama 3 herd of models. *arXiv*, 2407.21783, 2024. 5

[17] Dave Epstein, Ben Poole, Ben Mildenhall, Alexei A. Efros, and Aleksander Holynski. Disentangled 3d scene generation with layout learning, 2024. 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*, 2405.10314, 2024. 2, 3

[19] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T. Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Proc. CVPR*, 2019. 3

[20] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas A. Funkhouser. Local deep implicit functions for 3D shape. In *Proc. CVPR*, 2020. 3

[21] Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oguz. 3DGen: Triplane latent diffusion for textured mesh generation. *corr*, abs/2303.05371, 2023. 2

[22] Junlin Han, Jianyuan Wang, Andrea Vedaldi, Philip Torr, and Filippos Kokkinos. Flex3d: Feed-forward 3d generation with flexible reconstruction model and input view curation. *arXiv preprint arXiv:2410.00890*, 2024. 3

[23] Junlin Han, Filippos Kokkinos, and Philip Torr. Vfusion3d: Learning scalable 3d generative models from video diffusion models. In *European Conference on Computer Vision*, pages 333–350. Springer, 2025. 3

[24] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proc. NeurIPS*, 2020. 1

[25] Lukas Höllein, Aljaz Bozic, Norman Müller, David Novotný, Hung-Yu Tseng, Christian Richardt, Michael Zollhöfer, and Matthias Nießner. ViewDiff: 3D-consistent image generation with text-to-image models. In *Proc. CVPR*, 2024. 3

[26] 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. In *Proc. ICLR*, 2024. 3

[27] Yukun Huang, Jianan Wang, Yukai Shi, Xianbiao Qi, Zheng-Jun Zha, and Lei Zhang. Dreamtime: An improved optimization strategy for text-to-3D content creation. *CoRR*, abs/2306.12422, 2023. 3

[28] Ka-Hei Hui, Ruihui Li, Jingyu Hu, and Chi-Wing Fu. Neural template: Topology-aware reconstruction and disentangled generation of 3d meshes. In *Proc. CVPR*, 2022. 3

[29] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J. Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver IO: A general architecture for structured inputs & outputs. In *Proc. ICLR*, 2022. 1

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

[31] Heewoo Jun and Alex Nichol. Shap-E: Generating conditional 3D implicit functions. *arXiv*, 2023. 2

[32] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for real-time radiance field rendering. *Proc. SIGGRAPH*, 42(4), 2023. 3

[33] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. LERF: language embedded radiance fields. In *Proc. ICCV*, 2023. 3

[34] Chung Min Kim, Mingxuan Wu, Justin Kerr, Ken Goldberg, Matthew Tancik, and Angjoo Kanazawa. Garfield: Group anything with radiance fields. *arXiv.cs*, abs/2401.09419, 2024. 3

[35] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *Proc. CVPR*, 2023. 2, 3

[36] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing NeRF for editing via feature field distillation. *arXiv.cs*, 2022. 3

[37] Juil Koo, Seungwoo Yoo, Minh Hieu Nguyen, and Minhyuk Sung. SALAD: part-level latent diffusion for 3D shape generation and manipulation. In *Proc. ICCV*, 2023. 3

[38] D. Larlus, G. Dorko, D. Jurie, and B. Triggs. Pascal visual object classes challenge. In *Selected Proceeding of the first PASCAL Challenges Workshop*, 2006. 3

[39] 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. *Proc. ICLR*, 2024. 2, 3, 4, 5, 1

[40] Yuhan Li, Yishun Dou, Yue Shi, Yu Lei, Xuanhong Chen, Yi Zhang, Peng Zhou, and Bingbing Ni. Focaldreamer: Text-driven 3d editing via focal-fusion assembly, 2023. 3

[41] 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. *arXiv.cs*, abs/2211.10440, 2022. 3

[42] Connor Lin, Niloy Mitra, Gordon Wetzstein, Leonidas J. Guibas, and Paul Guerrero. NeuForm: adaptive overfitting for neural shape editing. In *Proc. NeurIPS*, 2022. 3

[43] Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. Common diffusion noise schedules and sample steps are flawed. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 5404–5411, 2024. 1

[44] Anran Liu, Cheng Lin, Yuan Liu, Xiaoxiao Long, Zhiyang Dou, Hao-Xiang Guo, Ping Luo, and Wenping Wang. Part123: Part-aware 3d reconstruction from a single-view image. *arXiv*, 2405.16888, 2024. 3, 6, 7

[45] 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. In *Proc. NeurIPS*, 2023. 3

[46] Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. PartSLIP: low-shot part segmentation for 3D point clouds via pretrained image-language models. In *Proc. CVPR*, 2023. 3

[47] 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 *Proc. ICCV*, 2023. 3

[48] Weiyu Liu, Jiayuan Mao, Joy Hsu, Tucker Hermans, Animesh Garg, and Jiajun Wu. Composable part-based manipulation. In *CoRL 2023*, 2023. 2

[49] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. SyncDreamer: Generating multiview-consistent images from a single-view image. *arXiv*, 2309.03453, 2023. 3

[50] Xiaoxiao Long, Yuanchen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, and Wenping Wang. Wonder3D: Single image to 3D using cross-domain diffusion. *arXiv.cs*, abs/2310.15008, 2023. 3

[51] LumaAI. Genie text-to-3D v1.0, 2024. 2

[52] Tiange Luo, Chris Rockwell, Honglak Lee, and Justin Johnson. Scalable 3d captioning with pretrained models. *arXiv preprint arXiv:2306.07279*, 2023. 5

[53] Oier Mees, Jessica Borja-Diaz, and Wolfram Burgard. Grounding language with visual affordances over unstruc-

tured data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, London, UK, 2023. 2

[54] Luke Melas-Kyriazi, Christian Rupprecht, Iro Laina, and Andrea Vedaldi. RealFusion: 360 reconstruction of any object from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[55] Luke Melas-Kyriazi, Christian Rupprecht, and Andrea Vedaldi. PC2: Projection-conditioned point cloud diffusion for single-image 3d reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[56] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, Natalia Neverova, Andrea Vedaldi, Oran Gafni, and Filippos Kokkinos. IM-3D: Iterative multiview diffusion and reconstruction for high-quality 3D generation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024. 2, 3

[57] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy Networks: Learning 3D reconstruction in function space. In *Proc. CVPR*, 2019. 3

[58] Meshy. Meshy text-to-3D v3.0, 2024. 2

[59] 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. In *Proc. ECCV*, 2020. 3

[60] Tom Monnier, Jake Austin, Angjoo Kanazawa, Alexei Efros, and Mathieu Aubry. Differentiable blocks world: Qualitative 3d decomposition by rendering primitives. *Advances in Neural Information Processing Systems*, 36: 5791–5807, 2023. 3

[61] George Kiyohiro Nakayama, Mikaela Angelina Uy, Jiahui Huang, Shi-Min Hu, Ke Li, and Leonidas Guibas. Diff-Facto: controllable part-based 3D point cloud generation with cross diffusion. In *Proc. ICCV*, 2023. 3

[62] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-E: A system for generating 3D point clouds from complex prompts. *arXiv.cs*, abs/2212.08751, 2022. 2

[63] Xiaqing Pan, Nicholas Charron, Yongqian Yang, Scott Peters, Thomas Whelan, Chen Kong, Omkar Parkhi, Richard Newcombe, and Carl Yuheng Ren. Aria digital twin: A new benchmark dataset for egocentric 3d machine perception, 2023. 2

[64] Ryan Po and Gordon Wetzstein. Compositional 3d scene generation using locally conditioned diffusion. *ArXiv*, abs/2303.12218, 2023. 3

[65] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D diffusion. In *Proc. ICLR*, 2023. 3, 8

[66] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, and Bernard Ghanem. Magic123: One image to high-quality 3D object generation using both 2D and 3D diffusion priors. *arXiv.cs*, abs/2306.17843, 2023. 3

[67] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. LangSplat: 3D language Gaussian splatting. In *Proc. CVPR*, 2024. 3

[68] Lingteng Qiu, Guanying Chen, Xiaodong Gu, Qi Zuo, Mutian Xu, Yushuang Wu, Weihao Yuan, Zilong Dong, Liefeng Bo, and Xiaoguang Han. Richdreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3D. *arXiv.cs*, abs/2311.16918, 2023. 3

[69] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proc. ICML*, pages 8748–8763, 2021. 3, 7, 1

[70] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. *arXiv*, 2408.00714, 2024. 2, 6, 7

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

[72] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. 1

[73] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multiview diffusion base model. *arXiv.cs*, abs/2310.15110, 2023. 4

[74] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. MVDream: Multi-view diffusion for 3D generation. In *Proc. ICLR*, 2024. 3, 4

[75] Aleksandar Shtedritski, Christian Rupprecht, and Andrea Vedaldi. What does clip know about a red circle? visual prompt engineering for vlms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11987–11997, 2023. 2

[76] Yawar Siddiqui, Filippos Kokkinos, Tom Monnier, Mahendra Kariya, Yanir Kleiman, Emilien Garreau, Oran Gafni, Natalia Neverova, Andrea Vedaldi, Roman Shapovalov, and David Novotny. Meta 3D Asset Gen: Text-to-mesh generation with high-quality geometry, texture, and PBR materials. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 2, 3, 4, 5, 8

[77] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *Proc. ICLR*, 2021. 1

[78] Jingxiang Sun, Bo Zhang, Ruizhi Shao, Lizhen Wang, Wen Liu, Zhenda Xie, and Yebin Liu. DreamCraft3D: Hierarchical 3D generation with bootstrapped diffusion prior. *arXiv.cs*, abs/2310.16818, 2023. 3

[79] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. DreamGaussian: Generative gaussian splatting for efficient 3D content creation. *arXiv*, 2309.16653, 2023. 3

[80] Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. Make-It-3D: High-fidelity 3d creation from A single image with diffusion prior. *arXiv.cs*, abs/2303.14184, 2023. 3

[81] Shitao Tang, Jiacheng Chen, Dilin Wang, Chengzhou Tang, Fuyang Zhang, Yuchen Fan, Vikas Chandra, Yasutaka Furukawa, and Rakesh Ranjan. MVDiffusion++: A dense high-resolution multi-view diffusion model for single or sparse-view 3d object reconstruction. *arXiv*, 2402.12712, 2024. 3

[82] Konstantinos Tertikas, Despoina Paschalidou, Boxiao Pan, Jeong Joon Park, Mikaela Angelina Uy, Ioannis Z. Emiris, Yannis Avrithis, and Leonidas J. Guibas. PartNeRF: Generating part-aware editable 3D shapes without 3D supervision. *arXiv.cs*, abs/2303.09554, 2023. 3

[83] TripoAI. Tripo3D text-to-3D, 2024. 2

[84] Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. Neural Feature Fusion Fields: 3D distillation of self-supervised 2D image representation. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2022. 3

[85] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A. Yeh, and Greg Shakhnarovich. Score Jacobian chaining: Lifting pretrained 2D diffusion models for 3D generation. In *Proc. CVPR*, 2023. 3

[86] Peng Wang and Yichun Shi. ImageDream: Image-prompt multi-view diffusion for 3D generation. In *Proc. ICLR*, 2024. 3, 4

[87] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. ProlificDreamer: High-fidelity and diverse text-to-3D generation with variational score distillation. *arXiv.cs*, abs/2305.16213, 2023. 3

[88] Daniel Watson, William Chan, Ricardo Martin-Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel view synthesis with diffusion models. In *Proc. ICLR*, 2023. 3

[89] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Liang Pan Jiawei Ren, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, Dahua Lin, and Ziwei Liu. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[90] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. GRM: Large gaussian reconstruction model for efficient 3D reconstruction and generation. *arXiv*, 2403.14621, 2024. 4

[91] Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, and Kai Zhang. DMV3D: Denoising multi-view diffusion using 3D large reconstruction model. In *Proc. ICLR*, 2024. 3

[92] Jiayu Yang, Ziang Cheng, Yunfei Duan, Pan Ji, and Hongdong Li. ConsistNet: Enforcing 3D consistency for multi-view images diffusion. *arXiv.cs*, abs/2310.10343, 2023.

[93] Yunhan Yang, Yukun Huang, Xiaoyang Wu, Yuan-Chen Guo, Song-Hai Zhang, Hengshuang Zhao, Tong He, and Xihui Liu. DreamComposer: Controllable 3D object gener-ation via multi-view conditions. *arXiv.cs*, abs/2312.03611, 2023. 3

[94] Lior Yariv, Omri Puny, Natalia Neverova, Oran Gafni, and Yaron Lipman. Mosaic-SDF for 3D generative models. *arXiv.cs*, abs/2312.09222, 2023. 2

[95] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arxiv:2308.06721*, 2023. 8, 1

[96] Taoran Yi, Jiemin Fang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. GaussianDreamer: Fast generation from text to 3D gaussian splatting with point cloud priors. *arXiv.cs*, abs/2310.08529, 2023. 3

[97] Haiyang Ying, Yixuan Yin, Jinzhi Zhang, Fan Wang, Tao Yu, Ruqi Huang, and Lu Fang. Omniseg3d: Omniversal 3d segmentation via hierarchical contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20612–20622, 2024. 3

[98] Wangbo Yu, Li Yuan, Yan-Pei Cao, Xiangjun Gao, Xiaoyu Li, Long Quan, Ying Shan, and Yonghong Tian. HiFi-123: Towards high-fidelity one image to 3D content generation. *arXiv.cs*, abs/2310.06744, 2023. 3

[99] Guanqi Zhan, Qingnan Fan, Kaichun Mo, Lin Shao, Baoquan Chen, Leonidas J Guibas, Hao Dong, et al. Generative 3d part assembly via dynamic graph learning. *Advances in Neural Information Processing Systems*, 33:6315–6326, 2020. 3

[100] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. CVPR*, pages 586–595, 2018. 7

[101] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J. Davison. In-place scene labelling and understanding with implicit scene representation. In *Proc. ICCV*, 2021. 3

[102] Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3D: Exploring unified 3D representation at scale. In *Proc. ICLR*, 2024. 3

[103] Yuchen Zhou, Jiayuan Gu, Xuanlin Li, Minghua Liu, Yunhao Fang, and Hao Su. PartSLIP++: enhancing low-shot 3d part segmentation via multi-view instance segmentation and maximum likelihood estimation. *arXiv*, 2312.03015, 2023. 3

[104] Junzhe Zhu and Peiye Zhuang. HiFA: High-fidelity text-to-3D with advanced diffusion guidance. *CoRR*, abs/2305.18766, 2023. 3

[105] Yan Zizheng, Zhou Jiapeng, Meng Fanpeng, Wu Yushuang, Qiu Lingteng, Ye Zisheng, Cui Shuguang, Chen Guanying, and Han Xiaoguang. Dreamdissector: Learning disentangled text-to-3d generation from 2d diffusion priors. *ECCV*, 2024. 3

PartGen: Part-level 3D Generation and Reconstruction with Multi-View Diffusion Models

Supplementary Material

This supplementary material contains the following parts:

- **Implementation Details.** Detailed descriptions of the training and inference settings for all models used in Part-Gen are provided.
- **Additional Experiment Details.** We describe the detailed evaluation metrics employed in the experiments and provide additional experiments.
- **Additional Examples.** We include more outputs of our method, showcasing applications with part-aware text-to-3D, part-aware image-to-3D, real-world 3D decomposition, and iteratively adding parts.
- **Failure Case.** We analyse the modes of of failure of Part-Gen.
- **Ethics and Limitation.** We provide a discussion on the ethical considerations of data and usage, as well as the limitations of our method.

A. Implementation Details

We provide the details of training used in PartGen (Appendices A.1 to A.4). In addition, we provide the implementation details for the applications: for part composition (Appendix A.5) and for part editing (Appendix A.6).

A.1. Text-to-multi-view generator

We fine-tune the text-to-multi-view generator starting with a pre-trained text-to-image diffusion model trained on billions of image-text pairs that uses an architecture and data similar to Emu [13]. We change the target image to a grid of 2×2 views as described in Section 3.5 following Instant 3D [39] via v-prediction [72] loss. The resolution of each view is 512×512, resulting in the total size of 1024×1024. To avoid the problem of the cluttered background mentioned in [39], we rescale the noise scheduler to force a zero terminal signal-to-noise ratio (SNR) following [43]. We use the DDPM scheduler with 1000 steps [24] for training. During the inference, we use DDIM [77] scheduler with 250 steps. The model is trained with 64 H100 GPUs with a total batch size of 512 and a learning rate 10^{-5} for 10k steps.

A.2. Image-to-multi-view generator

Building on the text-to-multi-view generator, we further fine-tune the model to accept images as input conditioning instead of text. The text condition is removed by setting it to a default null condition (an empty string). We concatenate the conditional image to the noised image along the spatial
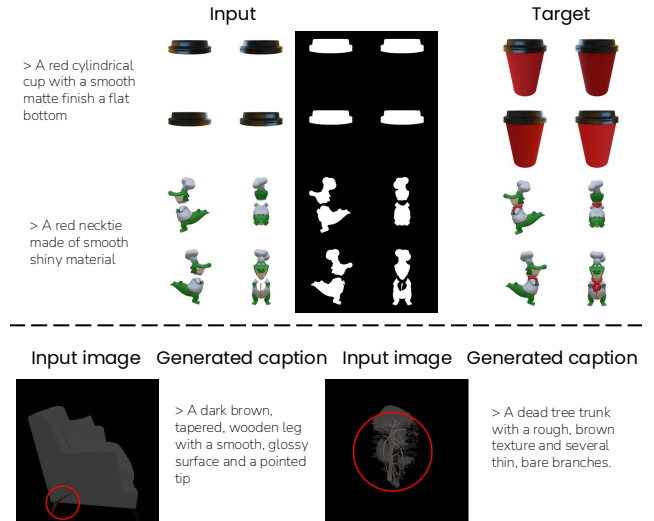

Figure 8. **3D part editing and captioning examples.** The top section illustrates training examples for the editing network, where a mask, a masked image, and text instructions are provided as conditioning to the diffusion network, which fills in the part based on the given textual input. The bottom section demonstrates the input for the part captioning pipeline. Here, a red circle and highlights are used to help the large vision-language model (LVLM) identify and annotate the specific part.

dimension, following [10]. Additionally, inspired by IP-adapter [95], we introduce another cross-attention layer into the diffusion model. The input image is first converted into tokens using CLIP [69], then reprojected into 157 tokens of dimension 1024 using a Perceiver-like architecture [29]. To train the model, we utilize all 140k 3D models of our data collection, selecting conditional images with random elevation and azimuth but fixed camera distance and field of view. We use the DDPM scheduler with 1000 steps [24], rescaled SNR, and v-prediction for training. Training is conducted with 64 H100 GPUs, a batch size of 512, and a learning rate of 10^{-5} over 15k steps.

A.3. Multi-view segmentation network

To obtain the multi-view segmentation network, we also fine-tune the pre-trained text-to-multi-view model. The input channels are expanded from 8 to 16 to accommodate the additional image input, where 8 corresponds to the latent dimension of the VAE used in our network. We create segmentation-image pairs as inputs. The training setup follows a similar recipe to that of the image-to-multi-view
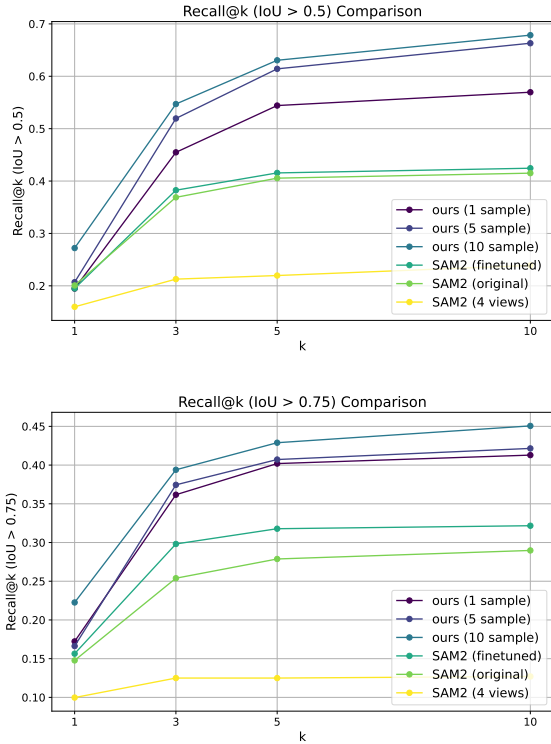
Figure 9. **Recall curve of different methods.** Our method achieve better performance comparing with SAM2 and its variants.

generator, employing a DDPM scheduler, v-prediction, and rescaled SNR. The network is trained with 64 H100 GPUs, a batch size of 512, a learning rate of 10^{-5}, for 10k steps.

A.4. Multi-view completion network

The training strategy for the multi-view completion network mirrors that of the multi-view segmentation network, with the key difference in the input configuration. The number of input channels (in latent space) is increased to 25 by including the context image, masked image, and binary mask, where the mask remains a single unencoded channel. Example inputs are illustrated in Figure 5 of the main text. The network is trained with 64 H100 GPUs, a batch size of 512, a learning rate of 10^{-5}, and for approximately 10k steps.

A.5. Parts assembly

When compositing an object from its parts, we observed that simply combining the implicit neural fields of parts reconstructed by the Reconstruction Model (RM) in the rendering process with their respective spatial locations achieves satisfactory results.

To describe this formally, we first review the rendering function of LightplaneLRM [5] that we use as our reconstruction model. LightplaneLRM employs a generalized

Emission-Absorption (EA) model for rendering, which calculates transmittance T_{ij}, representing the probability of a photon emitted at position x_{ij} (the j_{th} sampling point in the i_{th} ray) reaching the sensor. Then the rendered feature (*e.g.* color) v_i of ray r_i is computed as:

$$v_i = \sum_{j=1}^{R-1}(T_{i,j-1} - T_{i,j})f_v(x_{ij})$$

where $f_v(x_{ij})$ denotes the feature of the 3D point x_{ij}; $T_{i,j} = \exp(-\sum_{k=0}^{j} \Delta \cdot \sigma(x_{ik}))$, where Δ is the distance between two sampled points and $\sigma(x_{ik})$ is the opacity at position x_{ik}, $T_{i,j-1} - T_{i,j}$ captures the visibility of the point.

Now we show how we generalise it to rendering N parts. Given feature functions $f_v^1, \dots, f_v^N$ and their opacity functions $\sigma^1, \cdots, \sigma^N$, the rendered feature of a specific ray r_i becomes:

$$v_i = \sum_{j=1}^{R-1}\sum_{h=1}^{N}(\hat{T}_{i,j-1} - \hat{T}_{i,j})w_{ij}^h \cdot f_v^h(x_{ij}).$$

where $w_{ij}^h = \sigma^h(x_{ij})/\sum_{l=1}^{N}\sigma^l(x_{ij})$ is the weight of the feature $f_v^h(x_{ij})$ at x_{ij} for part h; $\hat{T}_{i,j} = \exp(-\sum_{k=0}^{j}\sum_{h=1}^{N}\Delta\cdot\sigma^h(x_{ik}))$, Δ is the distance between two sampled points and $\sigma^h(x_{ik})$ is the opacity at position x_{ik} for part h, and $\hat{T}_{i,j-1} - \hat{T}_{i,j}$ is the visibility of the point.

A.6. 3D part editing

As shown in the main text and Figure 7, once 3D assets are generated or reconstructed as a composition of different parts through PartGen, specific parts can be edited using text instructions to achieve 3D part editing. To enable this, we fine-tune the text-to-multi-view generator using part multi-view images, masks, and text description pairs. Example of the training data are shown in Figure 8 (top). Notably, instead of supplying the mask for the part to be edited, we provide the mask of the remaining parts. This design choice encourages the editing network to imagine the part's shape without constraining the region where it has to project. The training recipe is similar to multi-view segmentation network.

To generate captions for different parts, we establish an annotation pipeline similar to the one used for captioning the whole object, where captions for various views are first generated using LLAMA3 and then summarized into a single unified caption using LLAMA3 as well. The key challenge in this variant is that some parts are difficult to identify without knowing the context information of the object. We thus employ the technique inspired by [75]. Specifically, we use red annulet and alpha blending to emphasize the part being annotated. Example inputs and generated captions are shown in Figure 8 (bottom). The network is trained with 64 H100 GPUs, a batch size of 512, and the learning rate of 10^{-5} over 10,000 steps.
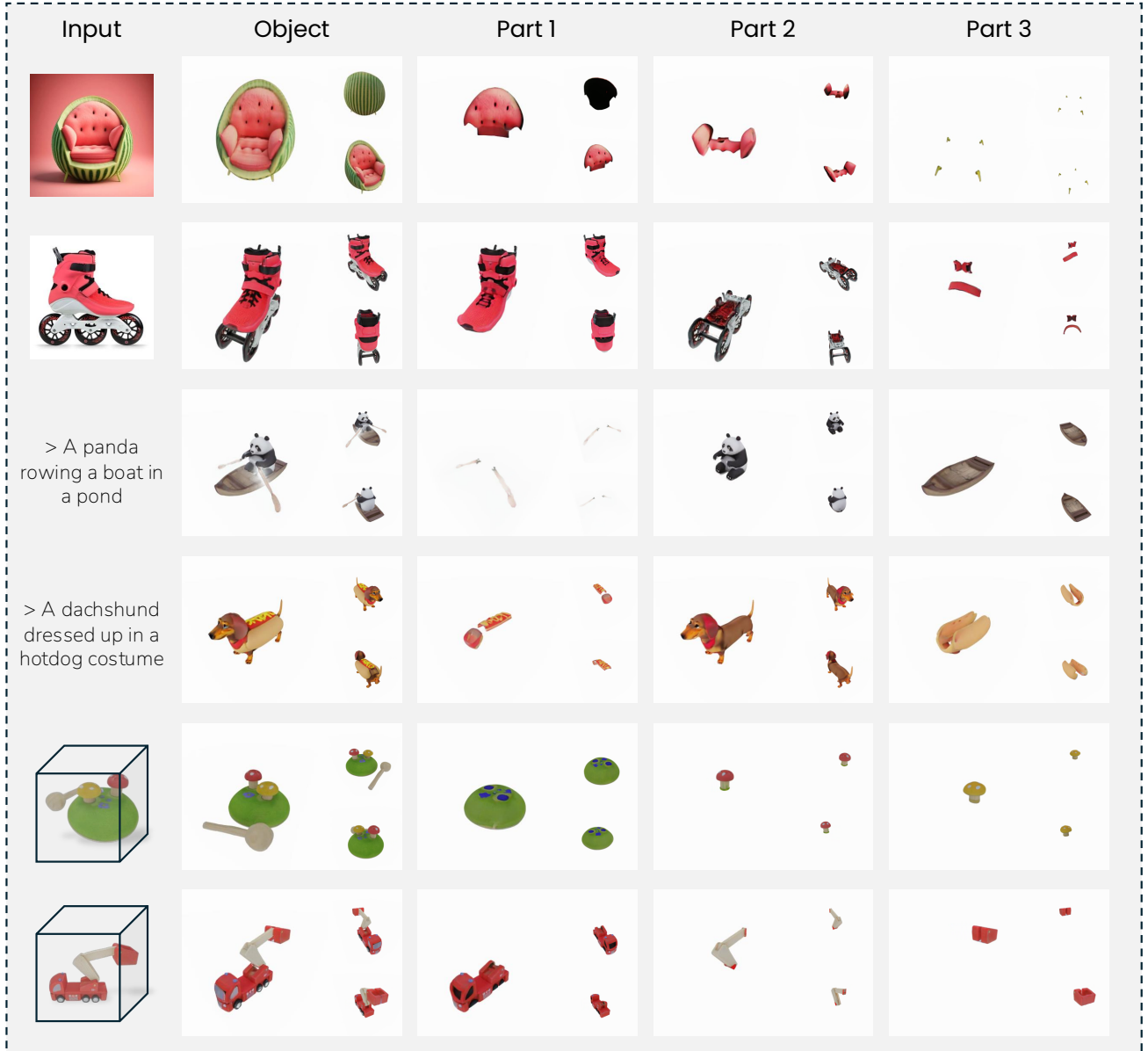
Figure 10. **More examples.** Additional examples illustrate that PartGen can process various modalities and effectively generate or reconstruct 3D objects with distinct parts.

B. Additional Experiment Details

We provide a detailed explanation of the ranking rules applied to different methods and the formal definition of mean average precision (mAP) used in our evaluation protocol. Additionally, we report the recall at K in the automatic segmentation setting.

Ranking the parts. For evaluation using mAP and recall at K, it is necessary to rank the part proposal. For our method, we run the segmentation network several times and concatenate the results into an initial set $\mathcal{P}$ of segment proposals.

Then, we assign to each segment $\hat{M} \in \mathcal{P}$ a reliability score based on how frequently it overlaps with similar segments in the list, *i.e.*,

$$s(\hat{M}) = \left| \left\{ \hat{M}' \in \mathcal{P} : m(\hat{M}', \hat{M}) > \frac{1}{2} \right\} \right|$$

where the *Intersection over Union* (IoU) [38] metric is given by:

$$m(\hat{M}, M) = \mathrm{IoU}(\hat{M}, M) = \frac{|\hat{M} \cap M| + \epsilon}{|\hat{M} \cup M| + \epsilon}.$$
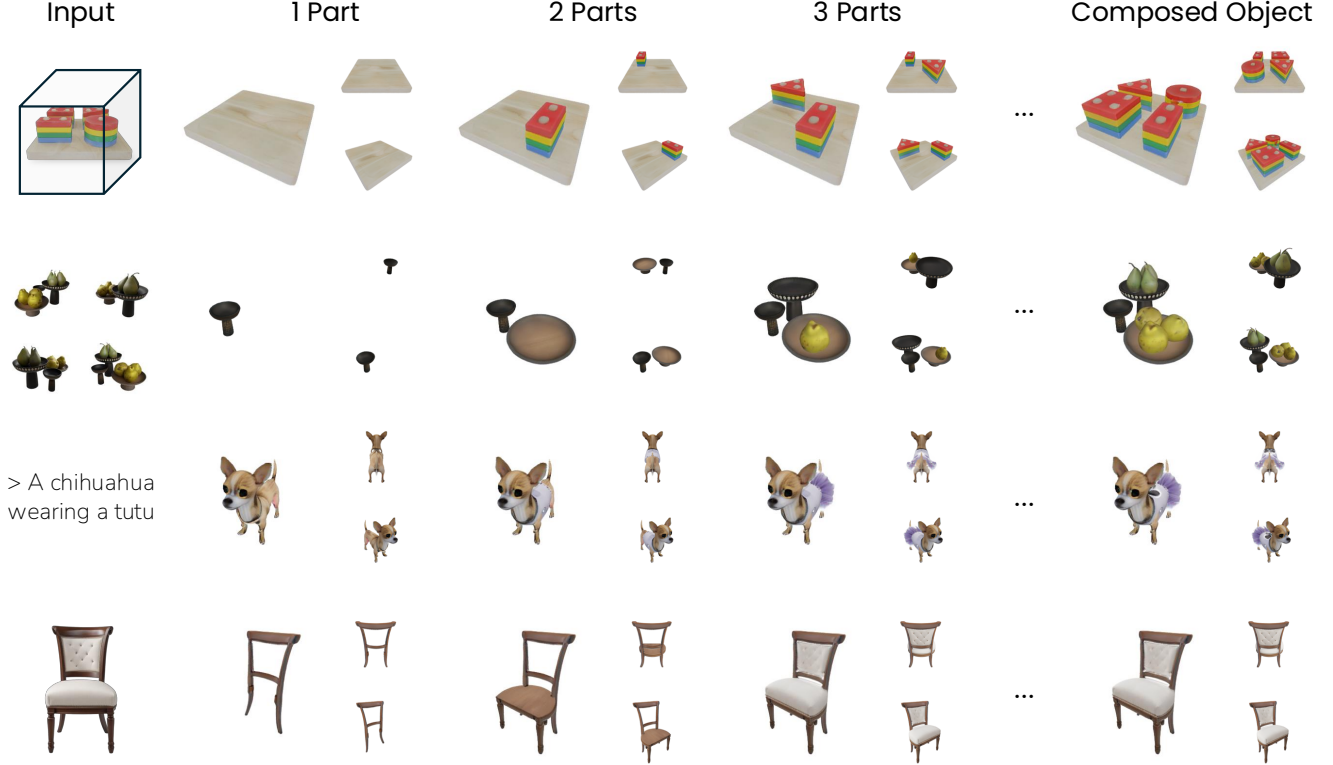
Figure 11. **Iteratively adding parts.** We show that users can iteratively add parts and combine the results of PartGen pipeline.

The constant $\epsilon = 10^{-4}$ smooths the metric when both regions are empty, in which case $m(\phi, \phi) = 1$, and will be useful later.

Finally, we sort the regions M by decreasing score $s(M)$ and, scanning the list from high to low, we incrementally remove duplicates down the list if they overlap by more than $1/2$ with the regions selected so far. The final result is a ranked list of multi-view masks $\mathcal{M} = (\hat{M}_1, \ldots, \hat{M}_N)$ where $N \leq |\mathcal{P}|$ and:

$$\forall i < j : \ s(\hat{M}_i) \geq s(\hat{M}_j) \ \wedge \ m(\hat{M}_i, \hat{M}_j) < \frac{1}{2}.$$

Other algorithms like SAM2 come with their own region reliability metric s, which we use for sorting. We otherwise apply non-maxima suppression to their ranked regions in the same way as ours.

Computing mAP. The image I comes from an object $\mathbf{L}$ with parts $(\mathbf{S}^1, \ldots, \mathbf{S}^S)$ from which we obtain the ground-truth part masks $\mathcal{S} = (M^1, \ldots, M^S)$ as explained in Section 3.5 in the main text. We assign ground-truth segments to candidates following the procedure: we go through the list $\mathcal{M} = (\hat{M}_1, \ldots, \hat{M}_N)$ and match the candidates one by one to the ground truth segment with the highest IOU, exclude that ground-truth segment, and continue traversing the candidate list. We measure the degree of overlap

between a predicted segment and a ground truth segment as $m(\hat{M}, M) \in [0, 1]$. Given this metric, we then report the *mean Average Precision* (mAP) metric at different IoU thresholds τ. Recall that, based on this definition, computing the AP curve for a sample involves matching predicted segments to ground truth segments in ranking order, ensuring that each ground truth segment is matched only once, and considering any unmatched ground truth segments.

In more detail, we start by scanning the list of segments $\hat{M}_k$ in order $k = 1, 2, \ldots$. Each time, we compare $\hat{M}_k$ to the ground truth segments $\mathcal{S}$ and define:

$$s^* = \underset{s=1,\ldots,S}{\operatorname{argmax}} \, m(\hat{M}_k, M_s).$$

If $m(\hat{M}_k, M_{s^*}) \geq \tau$, then we label the region M_s as retrieved by setting $y_k = 1$ and removing M_s from the list of ground truth segments not yet recalled by setting

$$\mathcal{S} \leftarrow \mathcal{S} \setminus \{M_{s^*}\}.$$

Otherwise, if $m(\hat{M}_k, M_{s^*}) < \tau$ or if $\mathcal{S}$ is empty, we set $y_k = 0$. We repeat this process for all k, which results in labels $(y_1, \ldots, y_N) \in \{0, 1\}^N$. We then set the *average precision* (AP) at τ to be:

$$\mathrm{AP}(\mathcal{M}, \mathcal{S}; \tau) = \frac{1}{S} \sum_{k=1}^{N} \sum_{i=1}^{k} \frac{y_i y_k}{k}.$$

(a) Grid view generation failure

(b) Segmentation Failure
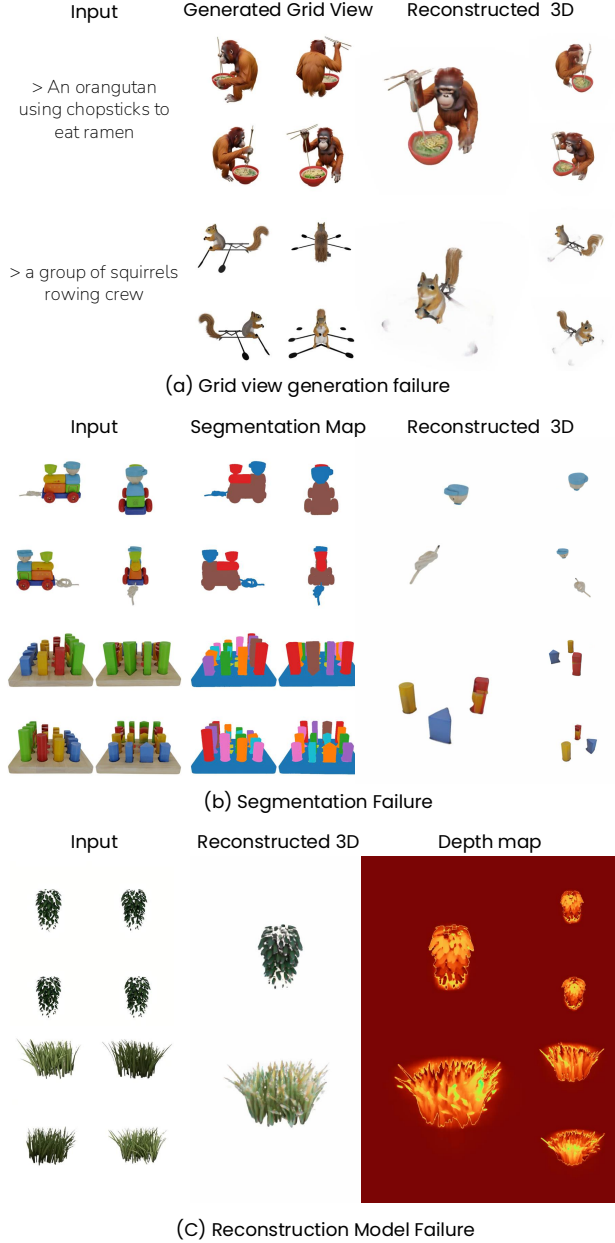

(C) Reconstruction Model Failure

Figure 12. **Failure Cases.** (a) Multi-view grid generation failure, where the generated views lack 3D consistency. (b) Segmentation failure, where semantically distinct parts are incorrectly grouped together. (c) Reconstruction model failure, where the complex geometry of the input leads to inaccuracies in the depth map.

Note that this quantity is at most 1 because by construction $\sum_{i=1}^{N} y_i \leq S$ as we cannot match more proposal than there are ground truth regions. mAP is defined as the average of the AP over all test samples.

Computing recall at K. For a given sample, we define *recall at K* the curve

$$R(K; \mathcal{M}, \mathcal{S}, \tau) = \frac{1}{S} \sum_{s=1}^{S} \chi \left(\max_{k=1,\ldots,K} m(\hat{M}_s, M_k) > \tau \right).$$

Hence, this is simply the fraction of ground truth segments recovered by looking up to position K in the ranked list of predicted segments. The results in Figure 9 demonstrate that our diffusion-based method outperforms SAM2 and its variants by a large margin and shows consistent improvement as the number of samples increases.

Seeded part segmentation. To evaluate *seeded part segmentation*, the assessment proceeds as before, except that a single ground truth part **S** and mask M is considered at a time, and the corresponding seed point $u \in M$ is passed to the algorithm $(\hat{M}_1, \ldots, \hat{M}_K) = \mathcal{A}(I, u)$. Note that, because the problem is still ambiguous, it makes sense for the algorithm to still produce a ranked list of possible part segments.

C. Additional Examples

More application examples. We provide additional application examples in Figure 10, showcasing the versatility of our approach to varying input types. These include part-aware text-to-3D generation, where textual prompts guide the synthesis of 3D models with semantically distinct parts; part-aware image-to-3D generation, which reconstructs 3D objects from a single image while maintaining detailed part-level decomposition; and real-world 3D decomposition, where complex real-world objects are segmented into different parts. These examples demonstrate the broad applicability and robustness of PartGen in handling diverse inputs and scenarios.

Iteratively adding parts. As shown in Figure 11, we demonstrate the capability of our approach to compose a 3D object by iteratively adding individual parts to it. Starting with different inputs, users can seamlessly integrate additional parts step by step, maintaining consistency and coherence in the resulting 3D model. This process highlights the flexibility and modularity of our method, enabling fine-grained control over the composition of complex objects while preserving the semantic and structural integrity of the composition.

D. Failure Cases

As outlined in the method section, PartGen incorporates several steps, including multi-view grid generation, multi-view segmentation, multi-view part completion, and 3D part reconstruction. Failures at different stages will result in specific issues. For instance, as shown in Figure 12(a), failures in grid view generation can cause inconsistencies

in 3D reconstruction, such as misrepresentations of the orangutan's hands or the squirrel's oars. The segmentation method can sometimes group distinct parts together, and limited, in our implementation, to objects containing no more than 10 parts, otherwise it merges different building blocks into a single part. Furthermore, highly complex input structures, such as dense grass and leaves, can lead to poor reconstruction outcomes, particularly in terms of depth quality, as illustrated in Figure 12(c).

E. Ethics and Limitation

Ethics. Our models are trained on datasets derived from artist-created 3D assets. These datasets may contain biases that could propagate into the outputs, potentially resulting in culturally insensitive or inappropriate content. To mitigate this, we strongly encourage users to implement safeguards and adhere to ethical guidelines when deploying PartGen in real-world applications.

Limitation. In this work, we focus primarily on object-level generation, leveraging artist-created 3D assets as our training dataset. However, this approach is heavily dependent on the quality and diversity of the dataset. Extending the method to scene-level generation and reconstruction is a promising direction but it will require further research and exploration.