SceneFactor: Factored Latent 3D Diffusion for Controllable 3D Scene Generation

Alexey Bokhovkin
Technical University of Munich

Quan Meng
Technical University of Munich

Shubham Tulsiani
Carnegie Mellon University

Angela Dai
Technical University of Munich

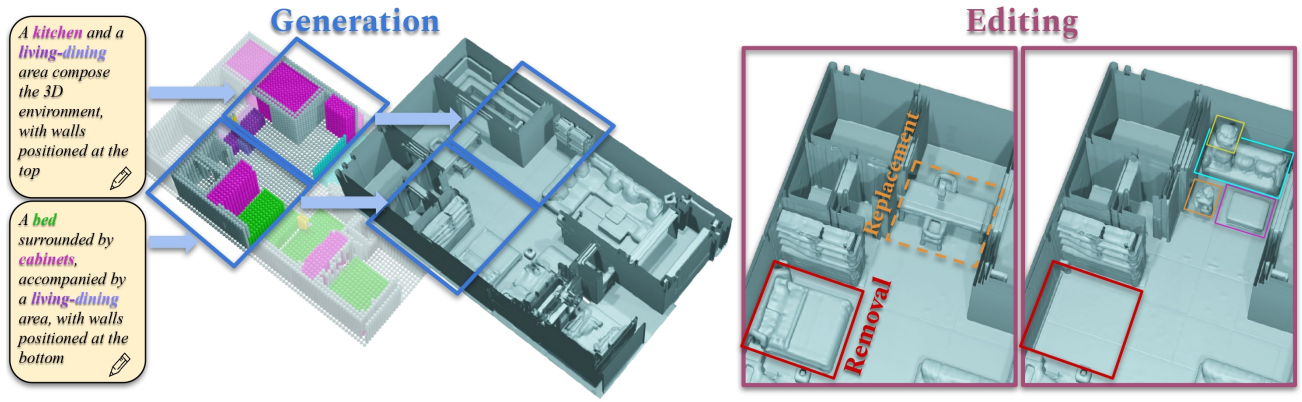

Figure 1. SceneFactor factors the complex task of text-guided 3D scene generation into forming a coarse semantic structure, followed by refined geometric synthesis. Rather than require a learned model to decide the location, type, size, and local geometry of scene elements directly, our generation of a coarse semantic box layout enables training a simpler task of layout-guided geometric synthesis. To achieve this factorized generation, we train semantic and geometric latent diffusion models. Crucially, the proxy semantic map generation enables user-friendly localized editing of generated scenes by editing in the semantic map with simple box operations (by clicking two box corners), without requiring re-synthesis of the full scene. Note that input text is colored by semantic categories for visualization purposes only.

Abstract

We present SceneFactor, a diffusion-based approach for large-scale 3D scene generation that enables controllable generation and effortless editing. SceneFactor enables text-guided 3D scene synthesis through our factored diffusion formulation, leveraging latent semantic and geometric manifolds for generation of arbitrary-sized 3D scenes. While text input enables easy, controllable generation, text guidance remains imprecise for intuitive, localized editing and manipulation of the generated 3D scenes. Our factored semantic diffusion generates a proxy semantic space composed of semantic 3D boxes that enables controllable editing of generated scenes by adding, removing, changing the size of the semantic 3D proxy boxes that guides high-fidelity, consistent 3D geometric editing. Extensive experiments demonstrate that our approach enables high-fidelity 3D scene synthesis with effective controllable editing through our factored diffusion approach.

Project page: alexeybokhovkin.github.io/mesh2tex/

1. Introduction

3D editable generative modeling is crucial to create immersive environments for many applications, such as augmented or virtual reality, video games and films, architectural design, or creating interactive simulations. Such content creation is inherently creative by nature, and is typically performed in an iterative process controlled by the user, with the ability to control and edit in localized regions to produce the desired output. Thus, a key requirement in generative 3D modeling is an underlying representation that en-

ables such intuitive, localized control and editing for users.

While remarkable advances in 2D generative modeling have been achieved with diffusion-based methods [26, 48, 52, 58] (even enabling controllability through semantic layouts, human poses, or depth [42, 78]), 3D generative modeling has largely focused on the unconditional or text-conditioned synthesis of 3D shapes [11, 18, 36, 56, 57], while the more challenging problem of large-scale 3D scene generation remains underexplored. Moreover, these methods also tend to lack editability, which is a key requirement of the content creation process – to be able to edit the generated representation in localized regions without requiring a re-synthesis of the full output. Editable generative approaches often lack the ease of editing operations, requiring the user to specify an accurate editing region boundary [3, 40, 53] or conduct extensive prompt engineering to avoid editing of undesired regions [5, 14, 60].

We thus propose a diffusion-based 3D generative approach for the synthesis of large-scale 3D scenes that enables intuitive, localized editing of the generated 3D representation in two clicks (defining a bounding box) per edited object. Key to our approach is a learned, latent semantic feature space which enables localized editability and control of the 3D scene generation. We learn to map text descriptions of scene regions to 3D semantic layout maps, which then guide the high-fidelity geometric synthesis of scene geometry corresponding to the proxy semantics. We formulate a two-stage latent semantic diffusion approach, first learning latent semantic and geometric feature spaces through VQ-VAE training. The latent semantic space is then modeled by diffusion, condition on text inputs, to produce a proxy semantic map. We then model the 3D scene geometry in its latent geometric space, conditioned on the semantic layout maps through spatial cross-attention to enable effective localized modeling of the semantic structure corresponding to geometric outputs.

Edits can then easily be performed in the semantic space by specifying the two points defining a bounding box (which can be automatically filled to match the proxy semantic map characteristics). To characterize the complexity in 3D scenes and handle larger scales, SceneFactor is trained on scene chunks, which can then be consistently outpainted to generate arbitrary-sized 3D scene outputs. Experiments show that SceneFactor enables text-guided synthesis as well as intuitive editing in the proxy semantic domain (e.g., adding objects by introducing new semantic boxes, as well as removing, moving, and editing generated objects by manipulating two corners of their semantic boxes).

In summary, our contributions are:

- the first method for text-guided large-scale 3D scene generation that enables easy, localized spatial editing for generated 3D scenes, performed in several mouse clicks.
- a latent semantic diffusion approach to enable two-stage generation of semantic and geometric latent manifolds characterizing coarse 3D scene layout and high-fidelity geometric structures, leveraging spatial cross-attention for strong spatial guidance of geometric synthesis.
- our latent semantic space enables intuitive, localized editing of generated 3D scenes without requiring re-synthesis of the full scene, enabling object addition, removal, replacement, and object manipulation while maintaining global scene consistency.

2. Related Work

2.1. 3D Shape Generation

Recent remarkable advances in 2D image generation have re-invigorated research in 3D generative modeling, which has largely focused on shape generation. Directly inspired by 2D generative models such as latent diffusion models [52], various methods have been developed to distill information from large, pretrained 2D models for text-to-3D radiance field generation [6, 10, 38, 49, 67, 77].

Alternatively, many other methods have focused on directly generating 3D shape representations by training on large 3D shape datasets such as ShapeNet [7]. In order to generate significant detail for high-dimensional 3D objects, recent approaches focus on generating compressed latent representations for 3D shapes [9, 41, 69, 73] and efficient mesh representations [32, 44, 57]. These methods focus on single-object generation in a canonicalized domain, while we focus on large-scale scene generation.

3D diffusion-based methods have also been developed for high-fidelity 3D shape generation. PVD [82] generates 3D point clouds with a hybrid point-voxel representation. Diffusion-SDF [12], HyperDiffusion [18], NFD [56], and SDFusion [11] leverage trained 1D, 2D, and 3D representations to more efficiently encode 3D shape geometry. While these approaches focus on single object generation, they can be conceptually applied to 3D scene generation by training on crops of 3D scenes. Our approach not only focuses on a factored diffusion approach for high-fidelity 3D scene generation, but learning a 3D scene representation that enables intuitive, localized editing for content creation scenarios.

2.2. 3D Scene Generation

Generating 3D scenes remains significantly more challenging than objects, due to the complexity of scene arrangements, high resolution required to resolve local detail, and strongly varying sizes [47]. Several approaches have thus relied on the capacity of image generative models to iteratively generate RGB views from text queries in order to form 3D scenes [20, 28, 59]; this results in impressive local appearance, but the lack of 3D reasoning often results in more incoherent global 3D structures. GAN-based approaches have enabled 3D-aware scene generation as radi-

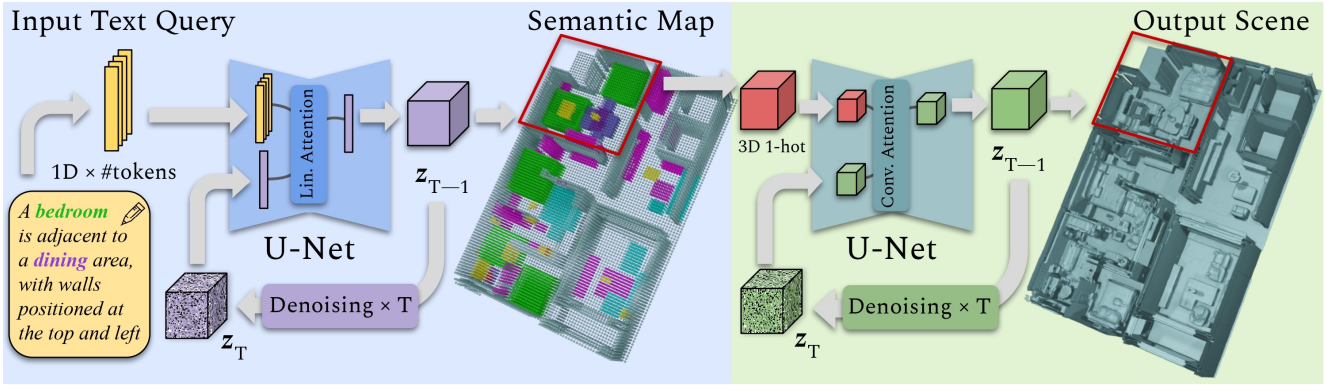

Figure 2. Method overview. We formulate text-guided 3D scene generation as a factored diffusion process, first generating a coarse semantic box layout representing the text input (left), followed by synthesis of scene geometry corresponding to the generated semantics (right). This factorization makes complex 3D scene generation more tractable and enables generation of locally editable 3D scenes, which can be manipulated through box manipulations in the semantic maps. Left: Our high-level semantic generation produces a coarse, box-level representation of a scene through latent diffusion on a pretrained semantic manifold, conditioned on text captions. This enables accurate alignment between text input and scene layout, without requiring solving a highly ambiguous generation task for geometric detail. Right: Conditioned on the coarse semantic box map, we use another latent diffusion model to generate 3D scene geometry, enabling spatial semantic grounding of generated scene objects and structures. Object categories in the text input are colored for visualization only.

ance fields using depth priors [55] or scene layouts [4], for improved view synthesis.

A popular approach is to leverage object retrieval in order to create 3D scenes with high-fidelity object structures, and instead synthesize the scene graph of object layouts [2, 8, 13, 16, 23, 35, 45, 46, 61, 65, 68, 75, 76, 79]. Due to the use of object retrieval, scene geometry remains limited to the object database used for retrieval. Most similar to our approach are several recent approaches, DiffInDScene [30], BlockFusion [70], SemCity [34] and XCube [51], which have been developed directly for large-scale scene generation, leveraging more flexible 3D representations unrestricted by object retrieval. In particular, BlockFusion produces a scene in a sliding window fashion, conditioned on a given layout, employing a single triplane latent diffusion stage. XCube generates the structure of an entire scene at once, without relying on sliding windows, instead generating a scene in a hierarchically coarse-to-fine fashion. Our approach also takes a chunked approach to scene generation, enabling large-scale synthesis of 3D scenes by chunk-based outpainting. However, in contrast to state-of-the-art 3D diffusion approaches that focus on direct scene generation that do not enable editing of scene outputs, we develop a factored diffusion approach to enable both high-fidelity geometric synthesis while enabling localized editing of output scenes.

2.3. 3D Object and Scene Editing

Generating controllable 3D object or scene representations has largely focused on conditional generative modeling formulations, using input text, images, or partial scans to guide output synthesis. For 3D shapes, methods such as AutoSDF [41] and ShapeFormer [73] enable 3D shape generation conditioned on image or partial 3D object inputs. 3D diffusion models can also be formulated as conditional diffusion models to enable text- or image-based 3D generation [11, 31, 36, 80]. Research on 3D scenes has also emphasized conditional generation, largely based on text and/or scene layout information to generate 3D scenes [19, 54, 70, 72]. While such conditional generative approaches enable high-level control over generated outputs based on adapting the input text, image, or layout, they would require re-synthesis of the generated output for adapted inputs, making localized editing challenging.

For 3D shapes, several approaches have been developed to enable more fine-grained localized shape editing, through local attention [81] or part-based reasoning [37, 43, 63]. Our approach formulates a factored diffusion approach to enable localized editing of generated 3D scenes.

3. Method

SceneFactor is a factored diffusion-based approach that generates large-scale 3D indoor scenes from text, using a proxy 3D semantic space to enable synthesis of high-fidelity, controllable 3D scenes. From an input text caption τ, we first synthesize a coarse 3D semantic layout S, representing a scene as 3D semantic boxes corresponding to the text τ. Based on semantic layout S, we then synthesize output scene geometry G. This factors the complex 3D scene generation process to high-level structural generation, followed by synthesis of geometric detail, enabling high-fidelity synthesis. Moreover, this enables the output scene G to be locally edited by simple manipulations performed on S. Both factored semantic and geometric representations are generated through conditional latent diffusion to produce compressed feature representations S and G.

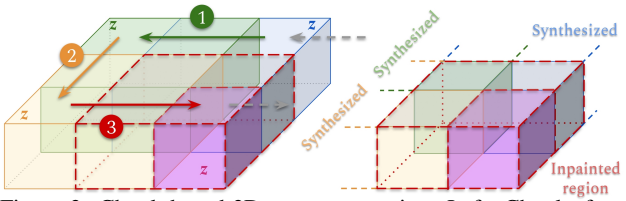

Figure 3. Chunk-based 3D scene generation. Left: Chunks for a scene are generated in sliding-window fashion (1-2-3), with overlap between generated chunks to ensure scene consistency along boundaries. Right: Synthesis of a chunk (chunk 3) is based on regions of previously generated chunks (1,2). The purple incomplete region is then synthesized by inpainting based on the previously generated blue, green, and yellow regions.

In order to synthesize large-scale scene environments, training is performed on scene chunks, and a 3D scene is generated chunk-by-chunk through outpainting. A set of input text descriptions $\{\tau_k\}_{k=1}^{N_c}$ provides high-level user control over the scene chunk generation, where N_c is the number of chunks to generate for an output scene.

We first describe the chunking of scenes for training our factored latent spaces as well as diffusion training in Sec. 3.1. We then optimize our factored latent semantic and geometric spaces (Sec. 3.2), followed by diffusion training over these spaces (Sec. 3.3). Finally, 3D scenes are synthesized by chunk-by-chunk outpainting (Sec. 3.4), and our factored generation approach enables localized 3D scene editing of synthesized scenes (Sec. 3.5).

3.1. Chunk-based 3D Scene Generation

To produce 3D scenes of arbitrary sizes, we train our approach on scene chunks and synthesize output scenes in chunk-by-chunk fashion. As shown in Fig. 3, a train scene X is chunked into N_c chunks in sliding-window fashion along the x and y axes (z remains a constant height). Chunks are generated with half-chunk overlap.

For a scene X, we then generate N_c chunks with text captions $\{\tau_k\}_{k=1}^{N_c}$, and corresponding semantic grids $\{S_k\}_{k=1}^{N_c}$ and geometric grids $\{G_k\}_{k=1}^{N_c}$. Each semantic chunk S_k contains a grid of one-hot-encodings of semantic boxes for each class category, where the first channel corresponds to free space, the second to wall/floor and the remaining 8 channels for object categories. The object categories are shown in Fig. 4. Each geometric chunk G_k describes a truncated unsigned distance field representation of the scene geometry in the chunk. We use cubic-sized chunks for the VQ-VAE training, with chunks twice as large (except in the up direction) for diffusion training.

To generate 3D scenes of arbitrary sizes, we describe chunk-by-chunk synthesis in Sec. 3.4, first generating the full semantic scene map based on a set of text captions, and then generating refined scene geometry, leveraging our factored generation process to disentangle the complex task of 3D scene generation into high-level semantic mapping followed by finer-grained geometric synthesis.

3.2. Factored Semantic and Geometric Latent Optimization

SceneFactor leverages dual semantic and geometric latent spaces for factored scene generation, enabling high-fidelity and editable 3D scene synthesis through disentangling the 3D scene generation task. To obtain both latent semantic and geometric spaces, we first optimize two models to encode compressed feature representations f_S and f_G that can be decoded to semantic and geometric chunks S and G.

Geometric distance field chunks $G \in \mathbb{R}^{128\times64\times128}$ are spatially compressed by a factor of 4 to $f_G \in \mathbb{R}^{32\times16\times32}$. Here, the latent space size is designed to be as small as possible to encourage effective generative modeling while still being able to decode to high-fidelity geometry. Semantic one-hot chunks $S \in \mathbb{Z}^{c\times32\times16\times32}$, where $c = 10$ denotes the number of class categories, are also spatially compressed by a factor of 4 to $f_S \in \mathbb{R}^{8\times4\times8}$.

In order to construct latent spaces that are memory efficient and produce smooth manifolds for efficient generation and editing using diffusion, we optimize for both semantic and geometric latent spaces using 3D VQ-VAEs [64]. Empirically, we found that using VQ-VAEs enabled high spatial compression with low feature dimensionality, enabling significant parameter reduction in encoding high-dimensional 3D data. Note that both latent semantic and geometric feature grids f_S and f_G maintain compressed feature representations with feature dimensionality of 1, enabled through VQ-VAE latent space training. In particular, we maintain 3D latent grids for both semantics and geometry to enable learning spatial correlation with the decoded spatial 3D domains. This enables our localized semantic editing of generated 3D scenes.

We then train a fully-convolutional 3D VQ-VAE for our geometric latent space, with encoder $\mathcal{E}^G$ and decoder $\mathcal{D}^G$ optimized for geometric reconstruction:

$$\mathcal{L}^{\text{geo}} = \|G - \mathcal{D}^G(\mathcal{E}^G(G))\|_1 + \mathcal{L}^{\text{quant}}(f_G), \quad (1)$$

where $\mathcal{L}^{\text{quant}}$ is the standard VQ-VAE quantization loss. Analogously, given the semantic encoder $\mathcal{E}^S$ and decoder $\mathcal{D}^S$ for the semantic 3D VQ-VAE, the semantic space is trained using the loss:

$$\mathcal{L}^{\text{sem}} = \mathcal{L}^{NLL}(S, \mathcal{D}^S(\mathcal{E}^S(S))) + \mathcal{L}^{\text{quant}}(f_S), \quad (2)$$

$$\mathcal{L}^{NLL}(S, \mathcal{D}^S(\mathcal{E}^S(S))) = -\sum_{k=1}^{c} [S]_k \log[\text{softmax}(\mathcal{D}^S(\mathcal{E}^S(S)))]_k, \quad (3)$$

where $[\cdot]_k$ denotes the k^{th} feature channel corresponding to class k, and $\mathcal{L}^{\text{quant}}$ is the standard VQ-VAE quantization

loss. The 3D latent encodings of semantics and geometry enables improved reconstruction as well as enabling localized editing based on manipulation of the semantic maps.

3.3. Factored 3D Scene Diffusion

Having obtained our factored latent semantic and geometric spaces, we can then train diffusion models, first to generate coarse semantic maps, and then to produce refined geometric synthesis. We adopt denoising diffusion probabilistic modeling (DDPM [25]) to denoise the semantic and geometric feature representations f_S and f_G from isotropic Gaussian noise in an iterative process. More specifically, DDPM takes a sample x_0 from the input data distribution $q(\mathbf{x})$ and iteratively adds small portions of Gaussian noise to obtain a sequence $x_1, x_2, \ldots, x_T$ until x_T reaches approximately an isotropic Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I})$. According to DDPM [25], the element x_t of this Markov Chain can be produced using the forward step:

$$q(x_t|x_{t-1}) \sim \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}), \tag{4}$$

where β_t is a variance schedule. During training, DDPM reverses the diffusion process and learns to predict the denoised sample x_0 from noisy x_t using a model p_θ, often represented as a neural network.

With $\alpha_t := 1 - \beta_t$, $\overline{\alpha}_t := \prod_{s=0}^{t} \alpha_s$ and $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, we can sample x_t directly from x_0:

$$x_t = \sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon. \tag{5}$$

To recover the signal instead of the added noise, we follow [11, 12] for the reverse process. In our implementation, we use the v_t parameterization as $v_t = \sqrt{\overline{\alpha}_t}\epsilon_t - \sqrt{1-\overline{\alpha}_t}x_t$.

For our text-to-semantic diffusion model, we construct a 3D variant of OpenAI LDM [17] for the main model, and use a transformer with the BERT [15] tokenizer to encode an input text query into a tokenized sequence of features. To condition the diffusion model, we apply attention where text features τ_i are treated as values and latent grids as queries and keys. Thus, the objective for the latent semantic diffusion model Ψ_S is the following:

$$\mathcal{L}_{LDM,\text{sem}} = \|\Psi_S(f_{S,t}, t, \tau_i) - v_{S,t}\|_1, \tag{6}$$

where t denotes the timestep of the diffusion process, $f_{S,t}$ is the noisy version of the feature $f_S = f_{S,0}$ and $v_{S,t} = \sqrt{\overline{\alpha}_t}\epsilon_t - \sqrt{1-\overline{\alpha}_t}f_{S,t}$ is the v-parameterization.

The semantic-to-geometry diffusion model is trained analogously with semantic maps as a condition. However, we have found that increasing awareness from local neighbourhoods is essential to capture correlations between semantic map condition and latent grid efficiently. To this aim, we modify linear layers that predict queries, keys and values as features of every separate geometric latent grid or semantic map cell. These linear layers can be viewed as convolutions with a window size of 1 and thus we employ convolutional-based attention modules with a window size of 3, where semantic maps S serve as values and latent grids as queries and keys. The second-stage diffusion model Ψ_G objective is analogous to the first-stage model:

$$\mathcal{L}_{LDM,geo} = \|\Psi_G(f_{G,t}, t, f_S) - v_{G,t}\|_2, \tag{7}$$

where $f_{G,t}$ is the noisy version of the feature $f_G = f_{G,0}$ and $v_{G,t} = \sqrt{\overline{\alpha}_t}\epsilon_t - \sqrt{1-\overline{\alpha}_t}f_{G,t}$ is the v-parameterization.

3.4. Outpainting Large-scale 3D Scenes

We train our factored diffusion models on fixed-size scene chunks; however, 3D scenes can have arbitrary spatial sizes. Thus, we must expand a generated chunk to form a full 3D scene. We generate such 3D scenes in a chunk-based sliding-window fashion, using overlaps between neighboring windows. From one or several already predicted chunks, we formulate the next chunk generation using its corresponding chunk text condition for inpainting, similar to RePaint [40]. We first outpaint semantic chunks, and then refine them to synthesize the corresponding scene geometry.

In Fig. 3, we show the step-by-step generation process for a simple example scene. The first (blue) chunk of latents is generated conditioned only on its text description. Our sliding window then moves along the direction of the arrows. The green chunk is then synthesized by inpainting, using the overlapping half of the already synthesized blue region (inpainting only the missing half). Inpainting is performed by modifying the denoising step, where instead of the classical denoising step formulation $f_{S,t-1} \sim \mathcal{N}(\tilde{\mu}_\theta(f_{S,t}; t), \Sigma_\theta(f_{S,t}; t))$ for step t, the inpainting modification is applied:

$$f^{\text{known}}_{S,t-1} \sim \mathcal{N}(\sqrt{\overline{\alpha}_t}f^{\text{known}}_S, (1-\overline{\alpha}_t)\mathbf{I}), \tag{8}$$

$$f^{\text{unknown}}_{S,t-1} \sim \mathcal{N}(\tilde{\mu}_\theta(f_{S,t}; t), \Sigma_\theta(f_{S,t}; t)), \tag{9}$$

$$f_{S,t-1} = m \odot f^{\text{known}}_{S,t-1} + (1-m) \odot f^{\text{unknown}}_{S,t-1}, \tag{10}$$

where f^{known}_S is a previously generated part of the scene, and m is a binary mask aligned with the currently generated chunk with ones denoting the known region within a chunk. Similarly, the yellow chunk is then inpainted given the already synthesized green half. The next chunk to be synthesized is highlighted in red, which shares overlap with the already synthesized blue, green, and yellow chunks. The missing purple region of the red chunk is then inpainted.

Since we use sliding windows with a step size of half of the horizontal chunk size, the unknown region to be inpainted is always either 25%, 50%, or 100% of the full chunk size in terms of number of parameters. Once the semantic map latent representation of a scene is fully outpainted, we traverse it using the same path of chunks and decode every chunk latent grid into a semantic map chunk

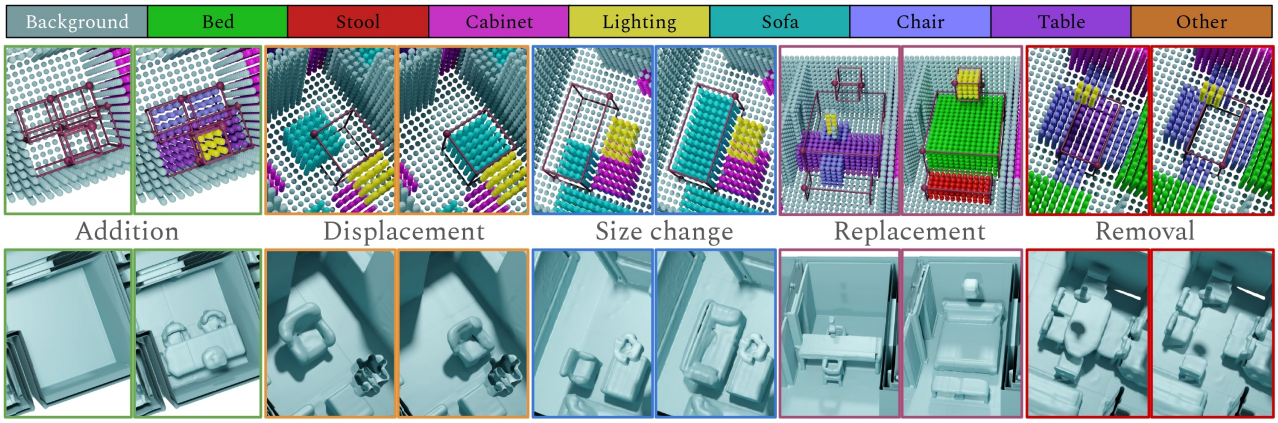

Figure 4. Scene editing. SceneFactor enables seamless localized editing through easy manipulation of the 3D semantic box map. We demonstrate the addition of objects (adding boxes), moving objects (moving an existing semantic box), changing object size (scaling an existing semantic box), replacing objects (replacing an existing object box with a new one of a different category), and removing objects (removing an existing semantic box). Note that the rest of the 3D scene remains consistent outside of the editing region.

with the VQVAE decoder $\mathcal{D}^S$. During this decoding process, the next semantic chunk always overwrites the regions that were previously decoded. The full scene geometric latent representation is outpainted analogously using the generated semantic maps as a condition. However, to obtain the full geometric scene representation we do not perform chunkwise decoding but decode the entire scene geometric latent grid using $\mathcal{D}^G$ to avoid seams between chunks.

3.5. Localized 3D Scene Editing

Crucially, our factored diffusion approach, disentangling 3D scene generation into coarse semantic synthesis followed by geometric refinement, enables various localized scene edits that can be performed by easy semantic box manipulation of the proxy semantic map representation in just a few mouse clicks. We demonstrate five example scene edits (object addition, removal, replacement, size changing, and displacement) in Fig. 4. We also choose equal resolutions of geometric chunk latents f_G and semantic chunk conditions S_k, ensuring their exact spatial alignment. This enables edits to propagate seamlessly from S_k to f_G, improving scene consistency after editing.

Edits are performed as follows, as simple, user-friendly box manipulations of the coarse semantic representation, specifying the two opposite box corners and possibly a new semantic class. For semantic grid S and corresponding geometric latent grid F_G:

- **Object addition**: is performed by adding semantic bounding boxes into an empty editing region $\mathcal{R}_S$ of the scene semantic map S. We fill only $\mathcal{R}_S$ in the grid F_G with Gaussian noise and re-generate geometry for it.
- **Object removal**: is performed by locating a 3D grid region $\mathcal{R}_S$ corresponding to the object to be removed, and deleting all semantic voxels that belong to it. The same region of the grid F_G is assigned to Gaussian noise and re-synthesized.
- **Object replacement**: is performed by replacing the 3D grid region $\mathcal{R}_S$ with a semantic box corresponding to the category of object desired as a replacement. The same region of the grid F_G is then assigned to Gaussian noise and re-synthesized.
- **Changing object size**: we first select an object in semantic map S and either increase (by adding new voxels to a box of the same category) or decrease (removing voxels by axis-aligned slices) its box size. We consider the editing region $\mathcal{R}_S$ to be the union of the original box and new box, and similarly re-assign the corresponding region of F_G to Gaussian noise for re-synthesis. Note that since we operate on a semantic rather than instance layout, size increases much larger than the likely size of an object tend to produce multiple objects to fill the size increase.
- **Moving an object**: an object is selected by selecting a box region $\mathcal{R}_S^1$ in the semantic map S; this box is then translated to the new region $\mathcal{R}_S^2$ of the same size as $\mathcal{R}_S^1$. The geometric features are analogously translated from $\mathcal{R}_S^1$ to $\mathcal{R}_S^1$ of the geometrical feature grid F_G. The initial region $\mathcal{R}_S^1$ of F_G is then filled with Gaussian noise, and both regions are re-synthesized.

4. Experimental Results

4.1. Experimental Setup

Datasets. We train and evaluate our method using a combination of the 3D-FRONT [21] and 3D-FUTURE [22] datasets. 3D-FUTURE contains >15,000 3D furniture models from 34 class categories. 3D-FRONT has 18,968 3D indoor scenes furnished with 3D-FUTURE objects. We obtain 3 million 3D crops of sizes 2.7m and 5.4m to train our VQ-VAE and diffusion models (voxel size 4.2cm). After filtering out empty or near-empty scenes, we use a train/test split of 6000/250.

We obtain two types of captions for scene chunks, where

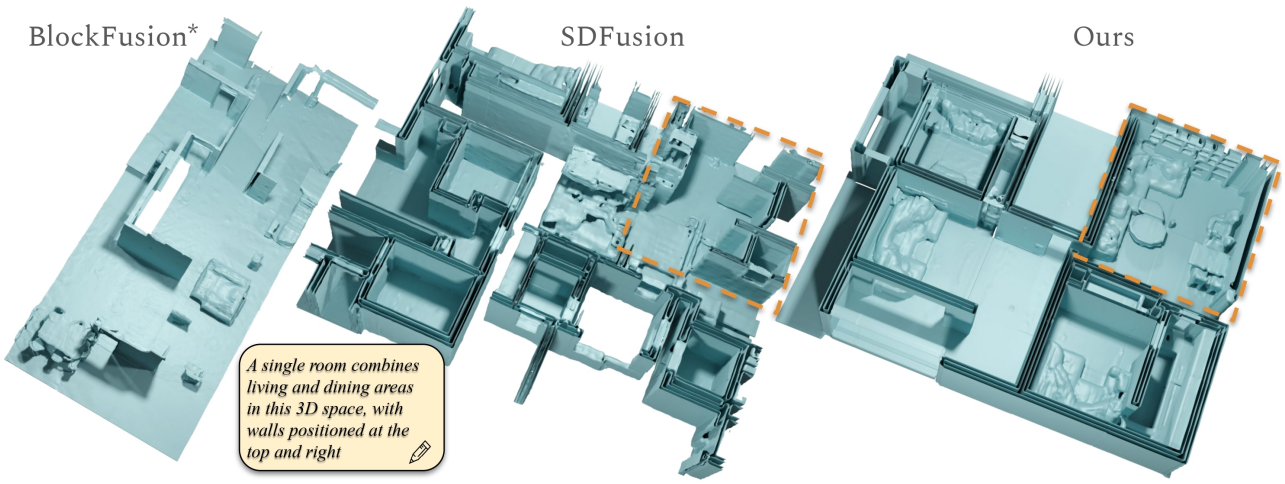

Figure 5. Qualitative comparisons to state-of-the-art diffusion-based 3D scene generative approaches BlockFusion [70], and SDFusion [11]. Our approach produces improved scene geometry and more cohesive global scene structure with consistent walls compared to baselines. *Note that results for BlockFusion are generated unconditionally.

the first set of captions is automatically generated from the 3D-FRONT object annotations in a template-based fashion, and the second set is refined from the first using Qwen1.5 [62]. For further information about data processing and caption generation, we refer to the supplemental.

Implementation Details. Our method is trained with an Adam [33] optimizer with learning rates 1e-4 and 2e-4 for the semantic and geometric VQ-VAEs. We use AdamW [39] with a learning rate 1e-5 for both semantic and geometric latent diffusion models. The semantic and geometric VQ-VAEs are trained on 2 NVIDIA A6000s each for 320k and 160k iterations ($\sim$ 50 hours) until convergence. The diffusion models are trained on 2 NVIDIA A100s each for 400k iterations ($\sim$ 100 and 150 hours, respectively).

4.2. Evaluation Metrics

We assess both generation and editing quality, in terms of geometric fidelity and adherence to text and editing inputs, evaluated for both individually generated chunks and crops of outpainted 3D scenes.

Geometric quality. We evaluate synthesized 3D scene geometry, following established evaluation metrics [70, 74], which do not take input conditions into account. Specifically, we use Minimum Matching Distance (MMD), Coverage (COV), and 1-Nearest-Neighbor-Accuracy (1-NNA). For MMD, lower is better; for COV, higher is better; for 1-NNA, 50% is the optimal. We use a Chamfer Distance (CD) distance measure for computing these metrics in 3D. Further details can be found in the supplementary.

Method	Independent chunks						Scene chunks					
	MMD ↓		COV ↑		1-NNA (0.5)		MMD ↓		COV ↑		1-NNA (0.5)	
	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD
NFD [56]	0.023	0.230	0.396	0.312	0.775	0.839	-	-	-	-	-	-
PVD [82]	0.021	0.221	0.367	0.220	0.778	0.867	-	-	-	-	-	-
SDFusion [11]	0.034	0.246	0.291	0.269	0.847	0.890	0.031	0.245	0.282	0.283	0.882	0.897
BlockFusion* [70]	0.048	0.305	0.177	0.110	0.953	0.986	0.054	0.330	0.186	0.091	0.961	0.993
Ours	**0.019**	**0.140**	**0.421**	**0.316**	**0.738**	**0.512**	**0.021**	**0.147**	**0.410**	**0.321**	**0.783**	**0.458**

Table 1. Geometric quality of synthesized 3D scene geometry as independent chunks (left) and as chunks of outpainted 3D scenes (right). SceneFactor generates scenes more reflective of ground-truth geometric distributions. *Note that BlockFusion results are generated unconditionally.

Target (Tr)	Distractor (Dis)	P(Tr)	P(Dis)	P(conf.)	P(Dis=GT) - P(Tr) ↓
Ours	SDFusion [11]	58%	42%	25%	-
Ours	Text2Room [28]	65%	35%	32%	-
SDFusion [11]	GT	33%	67%	25%	34%
Text2Room [28]	GT	38%	62%	31%	24%
Ours	GT	42%	58%	33%	**16%**

Table 2. Quality of text-guided generation using a pretrained neural listener model. Our results are preferred over that of SDFusion [11], and Text2Room [28], both in direct comparison as well as relative to ground truth.

Method	Independent chunks	Scene chunks
NFD [56]	26.59	26.59
PVD [82]	24.79	24.79
SDFusion [11]	28.01	27.70
Ours	**29.81**	**29.40**

Table 3. CLIP-Score evaluation of text-guided generation. Rendered views of chunks generated by our method better match text captions.

Consistency of synthesized geometry with text inputs. To evaluate how well synthesized geometry corresponds to input text queries, we follow the evaluation proposed by ShapeGlot [1]. A neural evaluator is trained to distinguish the target and distracting chunks, given the text description.

We also evaluate using CLIP [50] score, which reflects the consistency of generated geometry to text inputs in CLIP space. We render each chunk from 5 views (1 top, 4 side views). Since individual views may contain occluded objects, we evaluate the max CLIP score.

We also include a perceptual study in the supplemental.

Method	Independent chunks						Scene chunks					
	MMD ↓		COV ↑		1-NNA (0.5)		MMD ↓		COV ↑		1-NNA (0.5)	
	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD
w/o sem stage	0.022	0.146	0.405	0.295	0.757	0.482	0.026	0.231	0.409	0.299	0.785	0.895
w/o conv attn	0.020	0.161	**0.433**	0.312	0.752	0.514	0.024	0.233	**0.410**	**0.347**	0.869	0.896
w/o 3D latent	0.029	0.248	0.321	0.276	0.931	0.951	0.025	0.237	0.383	0.335	0.915	**0.496**
Ours	**0.019**	**0.140**	0.421	**0.316**	**0.738**	**0.512**	**0.021**	**0.147**	**0.410**	0.321	**0.783**	0.458

Table 4. Ablations. Our semantic proxy representation, 3D attention conditioning, and use of 3D latent spaces for semantics and geometry significantly improve generated scene quality.

4.3. Comparison with State of the Art

We compare with several state-of-the-art 3D diffusion-based generative methods leveraging various geometry representations: PVD [82] generates points, NFD [56] learns a latent triplane diffusion model, SDFusion [11] leverages a scalable latent grid representation for text-conditioned generation, Text2Room [28] employs RGB image synthesis to fuse observations into a scene mesh, and BlockFusion [70] uses a latent triplane diffusion model to generate large-scale scenes. We extend PVD and NFD approaches using the same BERT-based text encoding as SDFusion and ours. We apply our scene outpainting strategy for PVD, NFD, and SDFusion, but find empirically that it fails to generate coherent scenes for PVD and NFD, so we visualize SDFusion; BlockFusion is designed to produce large-scale scenes using triplane outpainting.

Tab. 1 shows a quantitative evaluation of the geometric quality of generated chunks as well as chunks sampled from generated scenes for models trained with synthetically generated captions. Our factored approach produces consistently improved geometry in comparison with baselines. PVD [82] does not decouple geometric compression and diffusion training and, using a limited number of points, which makes it unable to produce fine geometric details of scene chunks. NFD [56] struggles with the complex, diverse, non-canonicalized scene data. In addition, SDFusion [11] and BlockFusion [70] perform worse due to the lack of an intermediary spatially-structured condition.

This can also be seen in the qualitative results in Fig. 5. BlockFusion uses latent triplanes to outpaint scenes; however, the triplanes can produce misshapen objects, especially those intersecting with outpainting seams. Note that following the authors' suggestions for comparisons with BlockFusion, the results are generated unconditionally without text input, in contrast to SDFusion and our method.

Tabs. 2 and 3 evaluate the consistency of generated geometry to the input text, showing that our factored approach better adheres to input text prompts.

In contrast to state-of-the-art 3D generative methods, SceneFactor enables localized editing of generated 3D scenes, as shown in Fig. 4, maintaining scene consistency while manipulating geometry-based on easy box manipulations in the semantic domain. We include further comparisons and visualizations in the supplemental.

4.4. Ablation Studies

What is the impact of using a proxy semantic map for 3D scene generation? This helps to disentangle 3D scene generation into coarse object arrangement and refined geometric synthesis. Tab. 4 shows that performance improves with a proxy semantic generation, which helps to avoid generating floaters and incoherent object geometry or arrangements, since the semantic map provides guidance for object type and extent.

What is the effect of convolutional attention for geometric diffusion? Tab. 4 shows that attention with convolutions to extract queries, keys and values instead of linear layers and to interpret the semantic map enables far more accurate geometric synthesis than MLP-based attention, which tends to generate oversmoothed results. Convolutions with nontrivial window sizes enable better handling of correlations between a latent grid and condition due to encoding neighborhood information.

What is the impact of 3D latent grids for diffusion? We show in Tab. 4 that using a 3D latent space for semantic map generation and geometric synthesis significantly improves generation over a 1D latent space. In contrast to a 1D latent space, a 3D latent space maintains spatial correlations to the semantic structure of the scene, producing more effective output scene geometry.

Limitations. SceneFactor offers a first step towards text-guided controllable 3D indoor scene generation, though various limitations remain. For instance, while input text to our method is flexibly encoded, we train our 3D semantic layouts on closed vocabulary data, which can limit generation diversity of generated object types. Additionally, since we train on chunks and generate scenes by outpainting, room boundaries can be more difficult to control based on text input, instead requiring editing of the semantic map.

5. Conclusion

We have introduced SceneFactor, a new factored latent diffusion approach for controllable, editable 3D scene generation. By disentangling the complex 3D scene generation task into first creating a coarse, high-level structural semantic, followed by finer-grained geometric refinement, SceneFactor enables both effective text-guided 3D scene synthesis of large-scale scenes, and moreover, synthesis of editable 3D scene representations. Our coarse semantic map is structured as semantic boxes, enabling user-friendly box manipulation that can be used for various localized editing (object addition, removal, replacement, moving, altering size) of the generated 3D scenes. We believe this represents an important step towards artist-driven automated 3D content creation, through the formulation of editable 3D scene generation for content creation scenarios.

Acknowledgments

This project was supported by the ERC Starting Grant SpatialSem (101076253), the Bavarian State Ministry of Science and the Arts and coordinated by the Bavarian Research Institute for Digital Transformation (bidt), and the German Research Foundation (DFG) Grant "Learning How to Interact with Scenes through Part-Based Understanding."

References

[1] Panos Achlioptas, Judy Fan, Robert Hawkins, Noah Goodman, and Leonidas J Guibas. Shapeglot: Learning language for shape differentiation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8938–8947, 2019. 7

[2] Rio Aguina-Kang, Maxim Gumin, Do Heon Han, Stewart Morris, Seung Jean Yoo, Aditya Ganeshan, R. K. Jones, Qiuhong Anna Wei, Kailiang Fu, and Daniel Ritchie. Open-universe indoor scene generation using llm program synthesis and uncurated object databases. *ArXiv*, abs/2403.09675, 2024. 3

[3] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18208–18218, 2022. 2

[4] Sherwin Bahmani, Jeong Joon Park, Despoina Paschalidou, Xingguang Yan, Gordon Wetzstein, Leonidas Guibas, and Andrea Tagliasacchi. Cc3d: Layout-conditioned generation of compositional 3d scenes. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7137–7147, 2023. 3

[5] Omer Bar-Tal, Dolev Ofri-Amar, Rafail Fridman, Yoni Kasten, and Tali Dekel. Text2live: Text-driven layered image and video editing. In *European Conference on Computer Vision*, pages 707–723. Springer, 2022. 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. GeNVS: Generative novel view synthesis with 3D-aware diffusion models. In *arXiv*, 2023. 2

[7] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 2

[8] Aditya Chattopadhyay, Xi Zhang, David Paul Wipf, Himanshu Arora, and René Vidal. Learning graph variational autoencoders with constraints and structured priors for conditional indoor 3d scene generation. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 785–794, 2023. 3

[9] Siddhartha Chaudhuri, Daniel Ritchie, Jiajun Wu, Kai Xu, and Hao Zhang. Learning generative models of 3d structures. *Computer Graphics Forum*, 39(2):643–666, 2020. 2

[10] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2

[11] Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander G Schwing, and Liang-Yan Gui. SDFusion: Multimodal 3d shape completion, reconstruction, and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4456–4465, 2023. 2, 3, 5, 7, 8, 1, 4, 6, 9

[12] Gene Chou, Yuval Bahat, and Felix Heide. Diffusion-sdf: Conditional generative modeling of signed distance functions. 2023. 2, 5

[13] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Jordi Salvador, Kiana Ehsani, Winson Han, Eric Kolve, Ali Farhadi, Aniruddha Kembhavi, and Roozbeh Mottaghi. ProcTHOR: Large-Scale Embodied AI Using Procedural Generation. In *NeurIPS*, 2022. Outstanding Paper Award. 3

[14] Gilad Deutch, Rinon Gal, Daniel Garibi, Or Patashnik, and Daniel Cohen-Or. Turboedit: Text-based image editing using few-step diffusion models, 2024. 2

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019. 5

[16] Helisa Dhamo, Fabian Manhardt, Nassir Navab, and Federico Tombari. Graph-to-3d: End-to-end generation and manipulation of 3d scenes using scene graphs. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 3

[17] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. 2021. 5

[18] Ziya Erkoç, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Hyperdiffusion: Generating implicit neural fields with weight-space diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14300–14310, 2023. 2

[19] Chuan Fang, Xiaotao Hu, Kunming Luo, and Ping Tan. Ctrl-room: Controllable text-to-3d room meshes generation with layout constraints. *arXiv preprint arXiv:2310.03602*, 2023. 3

[20] Rafail Fridman, Amit Abecasis, Yoni Kasten, and Tali Dekel. Scenescape: Text-driven consistent scene generation. *arXiv preprint arXiv:2302.01133*, 2023. 2

[21] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10933–10942, 2021. 6, 1

[22] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3d-future: 3d furniture shape with texture. *International Journal of Computer Vision*, 129:3313–3337, 2021. 6, 1

[23] Lin Gao, Jia-Mu Sun, Kaichun Mo, Yu-Kun Lai, Leonidas Guibas, and Jie Yang. Scenehgn: Hierarchical graph networks for 3d indoor scene generation with fine-grained geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–18, 2023. 3

[24] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 5

[25] Jonathan Ho, Ajay Jain, and P. Abbeel. Denoising diffusion probabilistic models. *ArXiv*, abs/2006.11239, 2020. 5

[26] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

[27] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 1997. 4

[28] Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7909–7920, 2023. 2, 7, 8, 3, 4, 5

[29] Jingwei Huang, Hao Su, and Leonidas Guibas. Robust watertight manifold surface generation method for shapenet models. *arXiv preprint arXiv:1802.01698*, 2018. 1

[30] Xiaoliang Ju, Zhaoyang Huang, Yijin Li, Guofeng Zhang, Yu Qiao, and Hongsheng Li. Diffindscene: Diffusion-based high-quality 3d indoor scene generation. 2023. 3

[31] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions, 2023. 3

[32] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Popa Tiberiu. Clip-mesh: Generating textured meshes from text using pretrained image-text models. *SIGGRAPH Asia 2022 Conference Papers*, 2022. 2

[33] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015. 7, 4

[34] Jumin Lee, Sebin Lee, Changho Jo, Woobin Im, Juhyeong Seon, and Sung-Eui Yoon. Semcity: Semantic scene generation with triplane diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024. 3

[35] Manyi Li, Akshay Gadi Patil, Kai Xu, Siddhartha Chaudhuri, Owais Khan, Ariel Shamir, Changhe Tu, Baoquan Chen, Daniel Cohen-Or, and Hao Zhang. Grains: Generative recursive autoencoders for indoor scenes. *ACM Transactions on Graphics*, 37, 2018. 3

[36] Muheng Li, Yueqi Duan, Jie Zhou, and Jiwen Lu. Diffusion-sdf: Text-to-shape via voxelized diffusion. In *CVPR*, 2023. 2, 3

[37] Shidi Li, Miaomiao Liu, and Christian Walder. Editvae: Unsupervised parts-aware controllable 3d point cloud shape generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36:1386–1394, 2022. 3

[38] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[39] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017. 7, 5

[40] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11461–11471, 2022. 2, 5

[41] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. AutoSDF: Shape priors for 3d completion, reconstruction and generation. In *CVPR*, 2022. 2, 3

[42] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv preprint arXiv:2302.08453*, 2023. 2

[43] Kiyohiro Nakayama, Mikaela Angelina Uy, Jiahui Huang, Shi-Min Hu, Ke Li, and Leonidas Guibas. Difffacto: Controllable part-based 3d point cloud generation with cross diffusion. In *International Conference on Computer Vision (ICCV)*, 2023. 3

[44] Charlie Nash, Yaroslav Ganin, SM Ali Eslami, and Peter Battaglia. Polygen: An autoregressive generative model of 3d meshes. In *International conference on machine learning*, pages 7220–7229. PMLR, 2020. 2

[45] Yinyu Nie, Angela Dai, Xiaoguang Han, and Matthias Nießner. Learning 3d scene priors with 2d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 792–802, 2023. 3

[46] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 3, 4, 5

[47] Akshay Patil, Supriya Patil, Manyi Li, Matthew Fisher, and Manolis Savva. Advances in data-driven analysis and synthesis of 3d indoor scenes. *Computer Graphics Forum*, 43, 2023. 2

[48] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 2

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

[50] 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 *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 7

[51] Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. Xcube: Large-scale 3d generative modeling using sparse voxel hierarchies. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2024. 3

[52] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 2

[53] Rahul Sajnani, Jeroen Vanbaar, Jie Min, Kapil Katyal, and Srinath Sridhar. Geodiffuser: Geometry-based image editing with diffusion models, 2024. 2

[54] Jonas Schult, Sam Tsai, Lukas Höllein, Bichen Wu, Jialiang Wang, Chih-Yao Ma, Kunpeng Li, Xiaofang Wang, Felix Wimbauer, Zijian He, Peizhao Zhang, Bastian Leibe, Peter Vajda, and Ji Hou. Controlroom3d: Room generation using semantic proxy rooms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[55] Zifan Shi, Yujun Shen, Jiapeng Zhu, Dit-Yan Yeung, and Qifeng Chen. 3d-aware indoor scene synthesis with depth priors. 2022. 3

[56] J Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, , and Gordon Wetzstein. 3d neural field generation using triplane diffusion, 2023. 2, 7, 8, 1, 4, 5, 9

[57] Yawar Siddiqui, Antonio Alliegro, Alexey Artemov, Tatiana Tommasi, Daniele Sirigatti, Vladislav Rosov, Angela Dai, and Matthias Nießner. Meshgpt: Generating triangle meshes with decoder-only transformers. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2024. 2

[58] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. 2

[59] Liangchen Song, Liangliang Cao, Hongyu Xu, Kai Kang, Feng Tang, Junsong, Yuan, and Yang Zhao. Roomdreamer: Text-driven 3d indoor scene synthesis with coherent geometry and texture, 2023. 2

[60] Xuan Su, Jiaming Song, Chenlin Meng, and Stefano Ermon. Dual diffusion implicit bridges for image-to-image translation. In *International Conference on Learning Representations*, 2023. 2

[61] Jiapeng Tang, Yinyu Nie, Lev Markhasin, Angela Dai, Justus Thies, and Matthias Nießner. Diffuscene: Denoising diffusion models for gerative indoor scene synthesis. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2024. 3

[62] Qwen Team. Introducing qwen1.5, 2024. 7, 2, 4

[63] Konstantinos Tertikas, Despoina Paschalidou, Boxiao Pan, Jeong Joon Park, Mikaela Angelina Uy, Ioannis Emiris, Yannis Avrithis, and Leonidas Guibas. Generating part-aware editable 3d shapes without 3d supervision. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[64] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 4

[65] Kai Wang, Manolis Savva, Angel X Chang, and Daniel Ritchie. Deep convolutional priors for indoor scene synthesis. *ACM Transactions on Graphics (TOG)*, 37(4):70, 2018. 3

[66] Peng-Shuai Wang, Yang Liu, and Xin Tong. Dual octree graph networks for learning adaptive volumetric shape representations. *ACM Transactions on Graphics (TOG)*, 41:1 – 15, 2022. 1

[67] 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 preprint arXiv:2305.16213*, 2023. 2

[68] Qiuhong Anna Wei, Sijie Ding, Jeong Joon Park, Rahul Sajnani, Adrien Poulenard, Srinath Sridhar, and Leonidas Guibas. Lego-net: Learning regular rearrangements of objects in rooms. *arXiv preprint arXiv:2301.09629*, 2023. 3

[69] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Joshua B. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Neural Information Processing Systems*, 2016. 2

[70] Zhennan Wu, Yang Li, and Han Yan. Blockfusion: Expandable 3d scene generation using latent tri-plane extrapolation. https://synthical.com/article/66e48646-f127-4f88-bc78-f293758c0986, 2024. 3, 7, 8, 1, 2, 4, 5, 6, 9

[71] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. 2015. 5

[72] Han Yan, Yang Li, Zhennan Wu, Shenzhou Chen, Weixuan Sun, Taizhang Shang, Weizhe Liu, Tian Chen, Xiaqiang Dai, Chao Ma, Hongdong Li, and Pan Ji. Frankenstein: Generating semantic-compositional 3d scenes in one tri-plane. *ArXiv*, abs/2403.16210, 2024. 3

[73] Xingguang Yan, Liqiang Lin, Niloy J. Mitra, Dani Lischinski, Danny Cohen-Or, and Hui Huang. Shapeformer: Transformer-based shape completion via sparse representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3

[74] G. Yang, X. Huang, Z. Hao, M. Liu, S. Belongie, and B. Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4540–4549, Los Alamitos, CA, USA, 2019. IEEE Computer Society. 7

[75] Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, Chris Callison-Burch, Mark Yatskar, Aniruddha Kembhavi, and Christopher Clark. Holodeck: Language guided generation of 3d embodied ai environments. *arXiv preprint arXiv:2312.09067*, 2023. 3

[76] Guangyao Zhai, Evin Pınar Örnek, Shun-Cheng Wu, Yan Di, Federico Tombari, Nassir Navab, and Benjamin Busam. Commonscenes: Generating commonsense 3d indoor scenes with scene graph diffusion. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 3

[77] Jingbo Zhang, Xiaoyu Li, Ziyu Wan, Can Wang, and Jing Liao. Text2nerf: Text-driven 3d scene generation with neural radiance fields. *arXiv preprint arXiv:2305.11588*, 2023. 2

[78] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 2

[79] Song-Hai Zhang, Shao-Kui Zhang, Wei-Yu Xie, Cheng-Yang Luo, Yong-Liang Yang, and Hongbo Fu. Fast 3d indoor

scene synthesis by learning spatial relation priors of objects. *IEEE Transactions on Visualization and Computer Graphics*, 28(9):3082–3092, 2022. 3

[80] Zibo Zhao, Wen Liu, Xin Chen, Xianfang Zeng, Rui Wang, Pei Cheng, BIN FU, Tao Chen, Gang YU, and Shenghua Gao. Michelangelo: Conditional 3d shape generation based on shape-image-text aligned latent representation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 3

[81] Xin-Yang Zheng, Hao Pan, Peng-Shuai Wang, Xin Tong, Yang Liu, and Heung-Yeung Shum. Locally attentional sdf diffusion for controllable 3d shape generation. *ACM Transactions on Graphics (SIGGRAPH)*, 42(4), 2023. 3

[82] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5826–5835, 2021. 2, 7, 8, 1, 4, 5, 9

SceneFactor: Factored Latent 3D Diffusion for Controllable 3D Scene Generation

Supplementary Material

In this supplemental material, we provide details of data processing and caption generation in Section 6, show the additional qualitative and quantitative comparison to diffusion- and non-diffusion-based methods in Section 7, provide details of the evaluation metrics and the perceptual study in Section 8 and additional implementation details in Section 9.

6. Data Processing

Geometry. To make 3D-FRONT [21] data suitable for training and testing, we first combine 3D furniture and 3D scene meshes using 3D-FRONT annotation. 3D-FUTURE [22] models are preliminarily converted into high-quality watertight meshes using the Manifold [29] approach. This method can create meshes with double surfaces, so we remove all closed surfaces that lie within a mesh interior. To obtain the unsigned distance field of 3D-FRONT scenes with a resolution of 4.2 cm, we apply the virtual scanning tool mesh2sdf [66]. Preliminarily, we remove the ceiling from all 3D-FRONT scenes. In addition to the distance field, we regularly sample points with corresponding semantic labels belonging to scene layouts and furniture objects to form a semantic map of a scene with a resolution of 16.8 cm. The training chunks are obtained by randomly cropping from scene distance fields and semantic maps. We convert all test scenes into a test-suitable format by cutting the scenes into a regular grid of overlapping geometric and semantic chunks. All scene chunks are normalized to be centered at the origin and scaled to a unit cube.

Captions. To obtain captions for scene chunks, we use the 3D-FRONT object annotations to automatically generate seven types of captions. These caption types include descriptions with object counts or object lists without counts, subcategory information, and spatial relationships between objects. First, every scene annotation includes object instances of 8 categories depicted in Fig. 4. For every chunk, we add names of object categories into a caption if at least 35% of an object lies within a chunk. Here, we have two types of text captions: explicit lists of single objects as category names and aggregated lists where repeated objects are counted. Another caption type can be obtained from

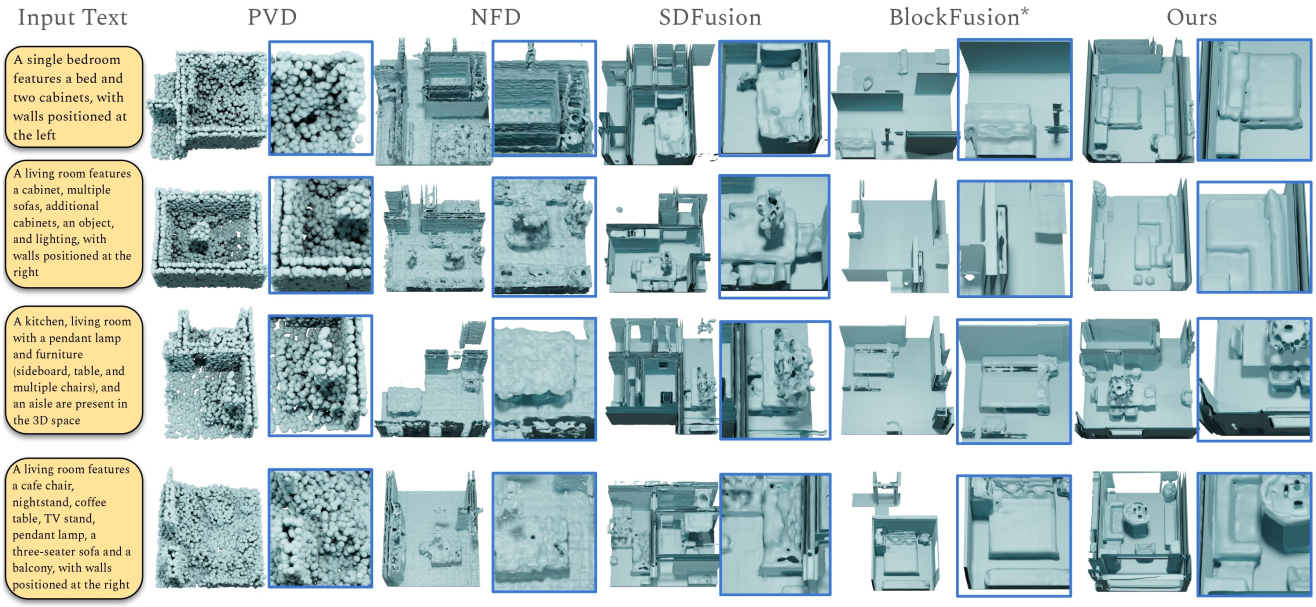

Figure 6. Qualitative comparison with state of the art on text-guided scene chunk generation using Qwen1.5 captions. In comparison with PVD [82], NFD [56], SDFusion [11], and BlockFusion [70] SceneFactor generates higher-fidelity, more coherent scene structures through our factored approach.
*Note that results for BlockFusion are generated unconditionally

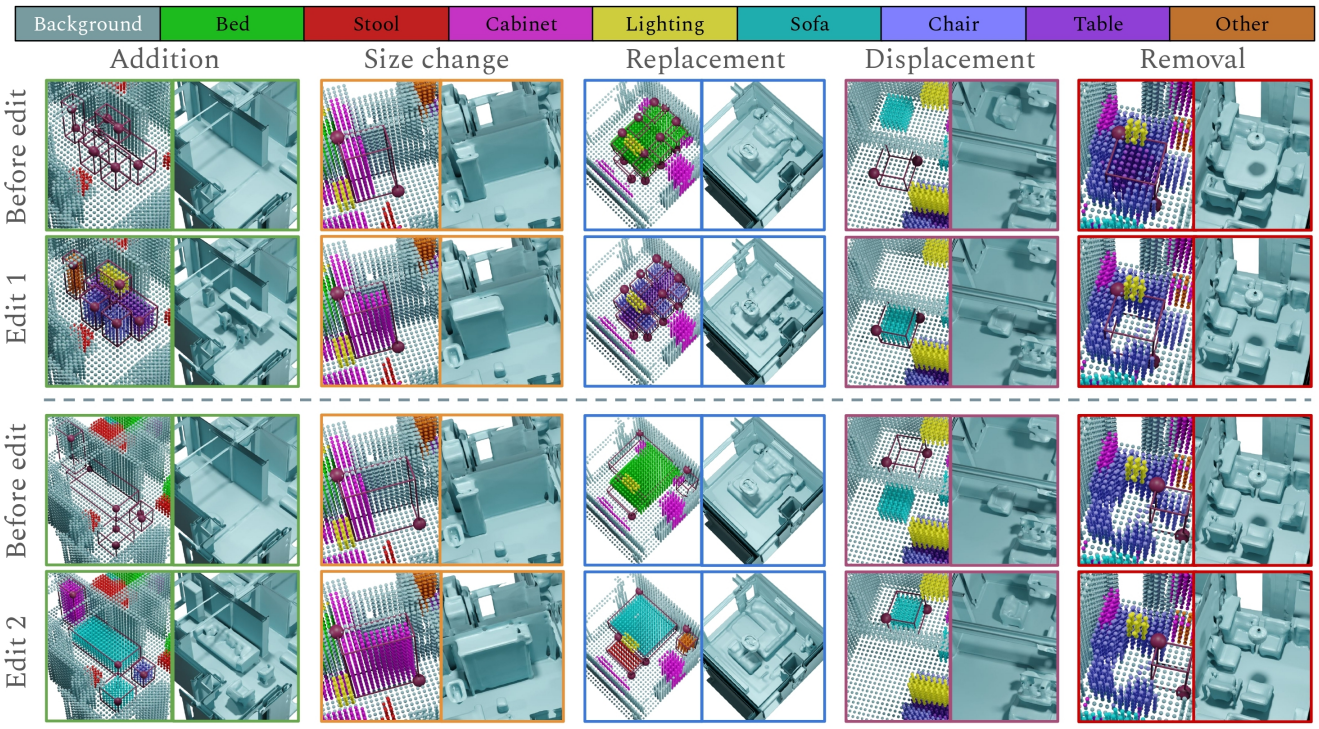

Figure 7. Additional qualitative scene editing results. Generated scenes and their corresponding semantic maps are shown in the top row, and two alternatives for each object synthesis-based edit are shown below.

the latter by adding spatial relationships between objects in a chunk. Second, using simple proximity checks based on Euclidean L_2 distance between object centers or object centers and wall points, we can identify if two or more objects form a group, stand across from each other, or stand next to a wall. For every caption, we also identify if there are walls along the borders of chunks. These three types of captions can be augmented using 33 subcategory names from 3D-FRONT annotation instead of category names. Finally, we have an extra room type caption, where for every chunk, we add room names from 3D-FRONT annotation to a caption if at least 25% of a room lies in a chunk.

LLM-Refined Captions. Finally, we train additional instances of SceneFactor, SDFusion [11], NFD [56], PVD [82] with the second set of captions – complex, natural text inputs. We utilize the large-language model Qwen1.5 [62] to refine our synthetic-looking captions using the following query: `Reformulate the following synthetic description of a 3D scene into a human-readable but concise, extremely minimalistic, and non-list format in only one sentence: <caption>`, where `<caption>` is the caption before LLM refinement.

Augmentations. During the training of the geometric and semantic VQ-VAE autoencoders and diffusion models, random 90°-fold rotation and symmetric reflection across xz- or yz-plane augmentations are applied to all train scene chunks and input latent representations.

7. Additional Results

Additional Comparison to Diffusion-based Methods. Fig. 6, 11 and 12 show additional qualitative comparisons with state-of-the-art baselines on scene chunk generation using synthetic and Qwen-refined captions. PVD [82] model uses explicit point cloud diffusion, which makes it significantly harder to generate clean and complete scenes. NFD [56] produces much cleaner scene layouts due to its signed distance field prediction. However, objects tend to lack details, with various low-level geometric artifacts due to the lack of structured latent space for generation. SDFusion [11] can generate more recognizable furniture. Nonetheless, due to direct text-to-geometry prediction and the absence of convolutional attention, SDFusion tends to generate more incoherent global structures (e.g., objects penetrating each other and inconsistent walls). Finally, BlockFusion [70] unconditional generations contain inconsistent wall structures, and triplane-based generation is unable to produce accurate furniture objects in arbitrary chunk

locations. In Tab. 9, 10, we provide the quantitative evaluation of our method and baseline approaches for the geometric quality and text-guided generation using Qwen1.5 captions as input.

Figs. 9 and 10 show additional qualitative comparisons for 3D scene generation with SDFusion [11] and BlockFusion [70]. SDFusion tends to produce more noticeable transitions between generated chunks, along with floating geometric artifacts and holes in furniture objects. Both SDFusion and BlockFusion generate significant artifacts, such as holes in the floor, due to the lack of conditioning on spatial information. BlockFusion struggles to outpaint objects from one chunk to the next chunk, which results in a significantly unnatural appearance of the generated room spaces.

Finally, we provide additional qualitative scene editing results for our method in Fig. 7. Our approach is able to produce diverse and consistent editing results for the same input scene.

Comparison to Non-diffusion-based Methods. In addition, we provide a comparison to 2D diffusion lifting-based approach Text2Room [28] and a retrieval-based method ATISS [46], for which we evaluate only independent chunks generation since these models are not applicable for large-scale scene generation.

Tab. 6 quantitatively evaluates the geometric quality of generated chunks against ATISS and Text2Room. Text2Room [28] takes significant time to generate one chunk ($\sim$ 3.5 hours); therefore, we limited the evaluation of this approach to 92 chunks. Our factored approach produces consistently improved geometry in comparison with these baselines. In Tab. 7, we also show that our approach significantly outperforms Text2Room by the CLIP score between rendered chunks and input text captions. We do not evaluate against the retrieval-based ATISS method because the CLIP score is biased towards non-generated but retrieved synthetic meshes placed on top of the floor. We found this comparison not meaningful. Instead, we evaluate our approach against ATISS using a pretrained neural listener model for the input text correspondence in Tab. 8. A neural evaluator is trained to distinguish the target chunk from a distracting chunk, given the text description. Given two chunks from different methods or one chunk from a method and another chunk from a GT set, the neural evaluator provides a confidence score for each of them based on the binary classification logits. If the absolute difference between two confidence scores ≤ 0.2, we consider the comparison to be confused. ATISS is not able to handle a large diversity of text captions and is significantly inferior to our approach in terms of text coherence.

Additional Semantic Evaluation. We provide additional analysis of our first-stage semantic map generation model, where the original latent diffusion model and the diffusion model explicitly trained with one-hot semantic maps are compared to each other. We first compute the average chunk semantic accuracy with respect to text input, where for every object class category mentioned in a caption, we check if the corresponding object has been predicted. For this metric, the latent-based model has accuracy of $\mathbf{91}\%$ against 83% for the model without latent representation. In Tab. 5, we provide further evaluation, which is based on MMD/COV/1-NNA metrics.

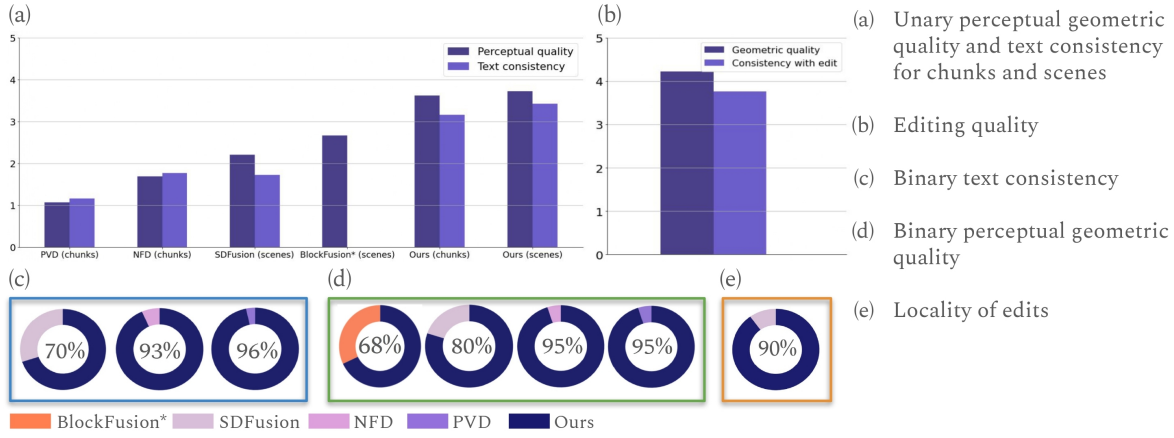

Figure 8. Perceptual study of the quality of text-guided 3D indoor scene generation and editing. (a) Unary study on perceptual geometric quality and text consistency for generated chunks and scenes. (b) Unary study on editing quality and scene consistency for SceneFactor. (c) Binary study between SceneFactor and baselines on text consistency between captions and generated chunks. (d) Binary study between SceneFactor and baselines on perceptual geometric quality of generated chunks. (e) Unary study of SceneFactor for locality of edits.
*Note that results for BlockFusion are generated unconditionally

8. Baseline Evaluation Setup

Metrics. Following the works for 3D shape generation, we use the following metrics on point clouds extracted from mesh surfaces:

$$\text{MMD}(S_g, S_r) = \frac{1}{|S_r|} \sum_{Y \in S_r} \min_{X \in S_g} D(X, Y),$$

$$\text{COV}(S_g, S_r) = \frac{|\{\text{argmin}_{Y \in S_r} D(X, Y) | X \in S_g\}|}{|S_r|},$$

$$\text{1-NNA}(S_g, S_r) = \frac{\sum_{X \in S_g} \mathbf{1}_X + \sum_{Y \in S_r} \mathbf{1}_Y}{|S_g| + |S_r|},$$

$$\mathbf{1}_X = \mathbf{1}[N_X \in S_g],\ \mathbf{1}_Y = \mathbf{1}[N_Y \in S_r],$$

where S_r and S_g are reference and generated sets of point clouds extracted from ground-truth and generated mesh surfaces, respectively, N_X is a point cloud that is closest to X in both generated and reference dataset, i.e., $N_X = \text{argmin}_{K \in S_r \cup S_g} D(X, K)$. We use Chamfer distance (CD) and Earth-mover distance (EMD) as $D(X, Y)$ to compute these metrics in 3D. To evaluate these metrics, we extract 4096 points from ground-truth and generated mesh surfaces or sample 4096 points from PVD point clouds.

We utilize the official implementations of NFD [56], PVD [82], SDFusion [11], BlockFusion [70], Text2Room [28], and ATISS [46]. For NFD and PVD, we do not implement the same or similar scene-aware generation mechanism, which inpaints missing chunks because PVD leverages the explicit point cloud representation in the diffusion model, and NFD demonstrates extremely poor results when using an inpaiting mechanism resulting in empty chunks which degrade in quality along the generation sequence. Text2Room and ATISS approaches are also inapplicable for large-scale scene generation using the outpainting mechanism. We use the same context encoding for text captions as in SceneFactor and SDFusion for NFD and PVD, while Text2Room, ATISS, and BlockFusion are designed to take text as input.

To evaluate geometric quality in Tab. 1, we normalize the ground-truth and predicted chunk meshes or point clouds into a unit cube and extract 4096 points from mesh surface or point cloud.

For the text-aware evaluation in Tab. 2, we train the neural listener model consisting of geometric encoder, text embedded, and language encoder. Geometric encoder consists of 5 ResNet blocks with GeLU activations and 2 linear layers with ReLU activations and takes uDF of geometric chunks as input. The input text is encoded using the same text encoder as in SceneFactor, but with an embedding dimension of 128. The text features are then processed using the LSTM [27] network. The resulting features are concatenated with geometric features and finally processed with a shallow MLP network with ReLU activations.

For the CLIP score evaluation in Tab. 3, we render 4 views of predicted meshes or point clouds and compute the cosine distance to text caption used for generation. We add a prefix 'a render of a 3D scene with ' to captions for CLIP score evaluation for ones not generated with Qwen1.5 [62] model.

Perceptual Study. To more effectively capture the perceptual quality of synthesized geometry, as well as adherence to text and editing inputs, we perform a perceptual study. We ask users to evaluate perceptual geometric quality as well as adherence to the text prompts, both as unary evaluation scores and binary comparisons between SceneFactor and each baseline. Perceptual geometric quality is assessed on a scale from 1 (Awful quality) to 5 (Great quality). Adherence to text input is assessed on a scale from 1 (Not matching) to 5 (Matching).

In particular, since we lack ground truth editing results as well as baselines that perform local spatial edits, we evaluate our editing performance through unary evaluation in the perceptual study. Editing results in the perceptual study are generated randomly across each possible editing operation. We ask users to assess (1) if the resulting edited scene is consistent with the given edit operation using a scale from 1 to 5; (2) the perceptual geometric quality of an edited scene using a scale from 1 to 5; and (3) if a scene remained unchanged outside of the editing region as either 1 (Yes) or 2 (No). In total, 21 participants took part in a perceptual study consisting of 53 questions per user. We provide the quantitative results of the conducted perceptual study in Fig. 8.

We developed a Django-based web application for the perceptual study. In total, we have 5 sections for our survey. For the first part, an unary study on perceptual geometric quality and text consistency for generated chunks and scenes, there are 25 questions and 5 randomly chosen scenes and chunks for every approach. Here, the user is asked to provide a score from 1 to 5 based on the perceptual geometric quality of chunks and the consistency of generation to an input text caption. In addition to chunkwise comparison, for SDFusion, BlockFusion, and our approach, there is also a unary study on scenes, where users are asked to evaluate the geometric quality of the whole scene. For SDFusion and ours, users are asked to evaluate the consistency of one scene chunk to a text caption.

9. Implementation Details

Our method is implemented using PyTorch. Semantic and geometric VQ-VAE models are trained with an Adam [33]

Method	Independent chunks					
	MMD ↓		COV ↑		1-NNA (0.5)	
	CD	EMD	CD	EMD	CD	EMD
w/o latent	0.263	0.473	0.335	0.344	0.784	0.784
Ours	**0.222**	**0.458**	**0.495**	**0.491**	**0.598**	**0.631**

Table 5. Semantic quality of synthesized 3D scene geometry as independent chunks.

optimizer with learning rates 1e-4 and 2e-4 for the semantic and geometric VQ-VAEs. We use AdamW [39] with a learning rate of 1e-5 for both semantic and geometric latent diffusion models. The semantic and geometric VQ-VAEs are trained on 2 NVIDIA A6000s each for 320k and 160k iterations ($\sim$ 50 hours) until convergence. The diffusion models are trained on 2 NVIDIA A100s each for 400k iterations ($\sim$ 100 and 150 hours, respectively).

VQ-VAE semantic and geometric models comprise 3 ResNet blocks in the encoder and 3 ResNet blocks in the decoder with bilinear upsampling layers and GeLU [24] nonlinearities. For the semantic VQ-VAE latent space, we encode semantic chunks into $(1, 4, 4, 4)$ latent grids, with only 1 feature channel using a dictionary size of 8192. Geometric chunks are encoded using the geometric VQ-VAE model into $(1, 16, 16, 16)$ latent grids, with 1 feature channel using a dictionary size of 32768.

The semantic diffusion model is trained using larger latent grids of size $(1, 8, 4, 8)$ that correspond to two twice bigger semantic chunks in both horizontal dimensions. We pad these grids with zeros to the shape of $(1, 8, 8, 8)$ to enable compression using 4 ResNet blocks in the encoder of the UNet model. The first 3 ResNet blocks combine convolutional operations with attention layers with 8 heads. To encode the context, we use the transformer-based model with BERT tokenizer and context dimension of 1280 and 77 maximum number of tokens.

Analogously, the geometric diffusion model is trained using larger latent grids of size $(1, 32, 16, 32)$ that correspond to two twice bigger geometric chunks in both horizontal dimensions. The UNet model encoder consists of 3 ResNet blocks with attention layers with 8 heads in each block. To encode the semantic context, we first encode the input semantic chunk of size $(1, 32, 16, 32)$ into one-hot representation with 10 class channels. This one-hot representation is encoded into a context feature grid of size $(128, 16, 8, 16)$ using the fully convolutional network with LeakyReLU activations [71].

Method	Independent chunks					
	MMD ↓		COV ↑		1-NNA (0.5)	
	CD	EMD	CD	EMD	CD	EMD
Text2Room	0.048	0.316	0.021	0.021	0.997	0.993
ATISS [46]	0.050	0.327	0.117	0.117	0.993	0.992
Ours	**0.019**	**0.140**	**0.421**	**0.316**	**0.738**	**0.512**

Table 6. Geometric quality of synthesized 3D scene geometry as independent chunks (left) and as chunks of outpainted 3D scenes (right).

Method	Independent chunks	Scene chunks
Text2Room [28]	24.11	24.11
Ours	**29.81**	**29.40**

Table 7. CLIP-Score evaluation of text-guided generation. Rendered views of chunks generated by our method better match text captions.

Target (Tr)	Distractor (Dis)	P(Tr)	P(Dis)	P(conf.)	P(Dis=GT) - P(Tr) ↓
Ours	ATISS [46]	66%	34%	21%	-
ATISS [46]	GT	33%	67%	22%	34%
Ours	GT	42%	58%	33%	**16%**

Table 8. Quality of text-guided generation using a pretrained neural listener model. Our results are preferred over that of SDFusion [11], ATISS [46], and Text2Room [28], both in direct comparison as well as relative to ground truth.

Method	Independent chunks						Scene chunks					
	MMD ↓		COV ↑		1-NNA (0.5)		MMD ↓		COV ↑		1-NNA (0.5)	
	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD
NFD [56]	0.023	0.225	**0.411**	**0.335**	0.744	0.814	-	-	-	-	-	-
PVD [82]	**0.021**	0.221	0.396	0.285	**0.729**	0.876	-	-	-	-	-	-
SDFusion [11]	0.031	0.240	0.331	0.277	0.835	0.898	0.035	0.253	0.313	0.265	0.874	**0.910**
BlockFusion* [70]	0.048	0.305	0.177	0.110	0.953	0.986	0.054	0.330	0.186	0.091	0.961	0.993
Ours	**0.021**	**0.165**	0.399	0.300	0.772	**0.769**	**0.026**	**0.249**	**0.365**	**0.270**	**0.839**	**0.910**

Table 9. Geometric quality of synthesized 3D scene geometry as independent chunks (left) and as chunks of outpainted 3D scenes (right) generated with Qwen1.5 captions.

Method	Independent chunks	Scene chunks
NFD [56]	21.61	21.61
PVD [82]	20.32	20.32
SDFusion [11]	23.08	23.17
Ours	**23.96**	**23.79**

Table 10. CLIP-Score evaluation of text-guided generation using Qwen1.5 captions. Rendered views of chunks generated by our method better match text captions.

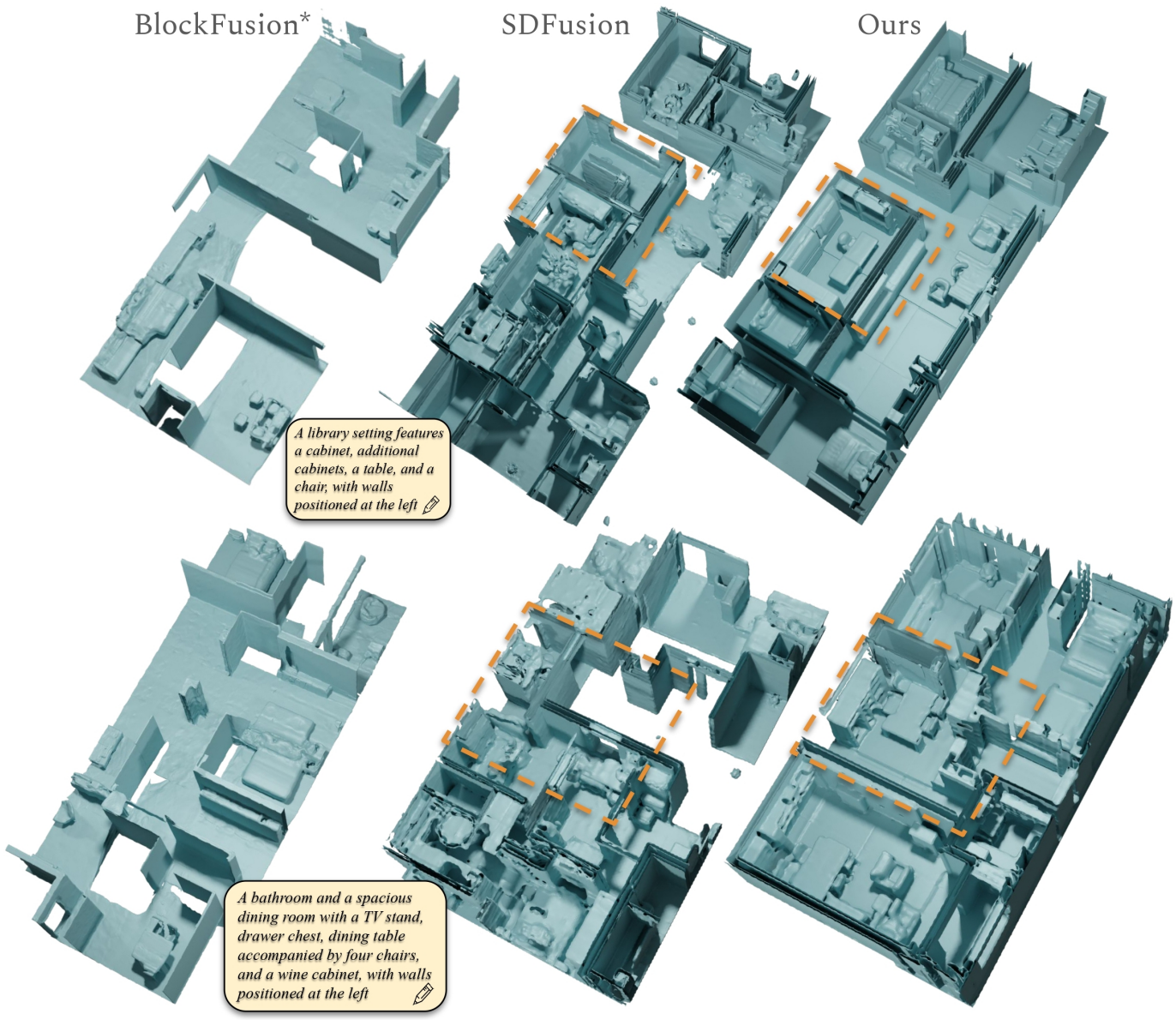

Figure 9. Additional qualitative comparisons for scene generation in comparison with SDFusion [11] and BlockFusion [70].
*Note that results for BlockFusion are generated unconditionally

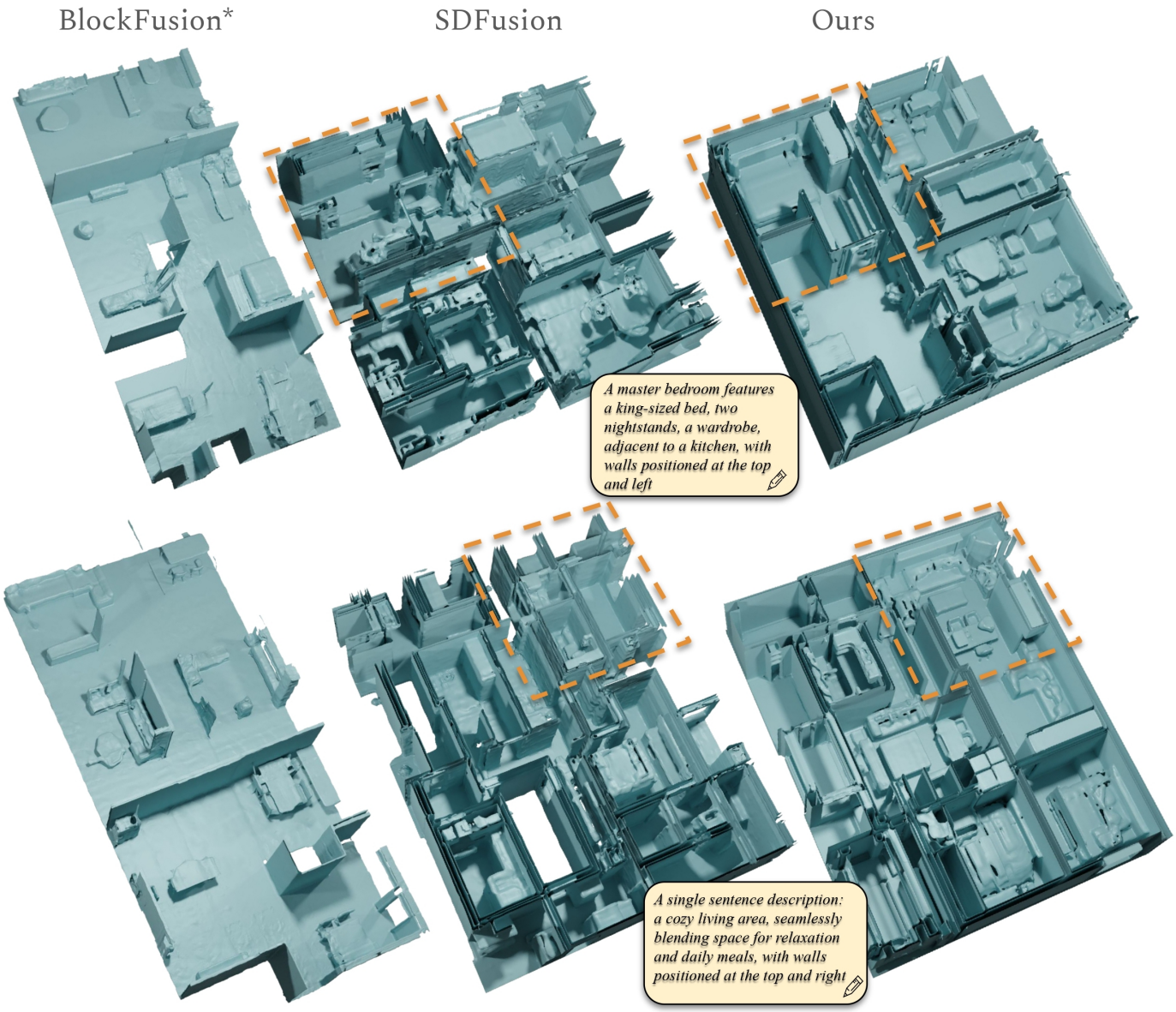

Figure 10. Additional qualitative comparisons for scene generation in comparison with SDFusion [11] and BlockFusion [70].
*Note that results for BlockFusion are generated unconditionally

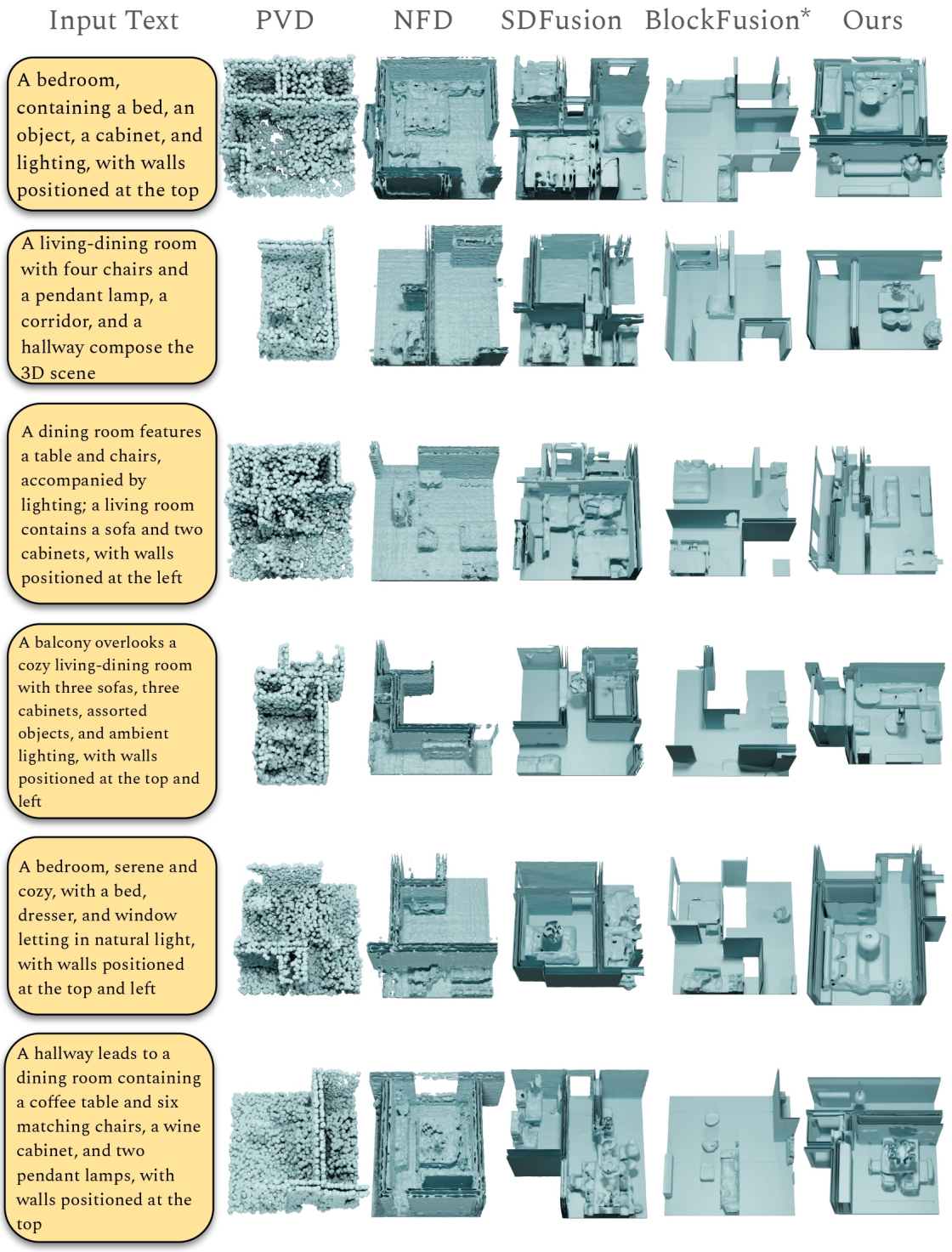

Figure 11. Additional qualitative comparisons to state-of-the-art diffusion-based 3D generative approaches PVD [82], NFD [56], SDFusion [11], and BlockFusion [70] using Qwen1.5 captions. Our approach produces sharper scene geometry and more coherent scene structure.
*Note that results for BlockFusion are generated unconditionally

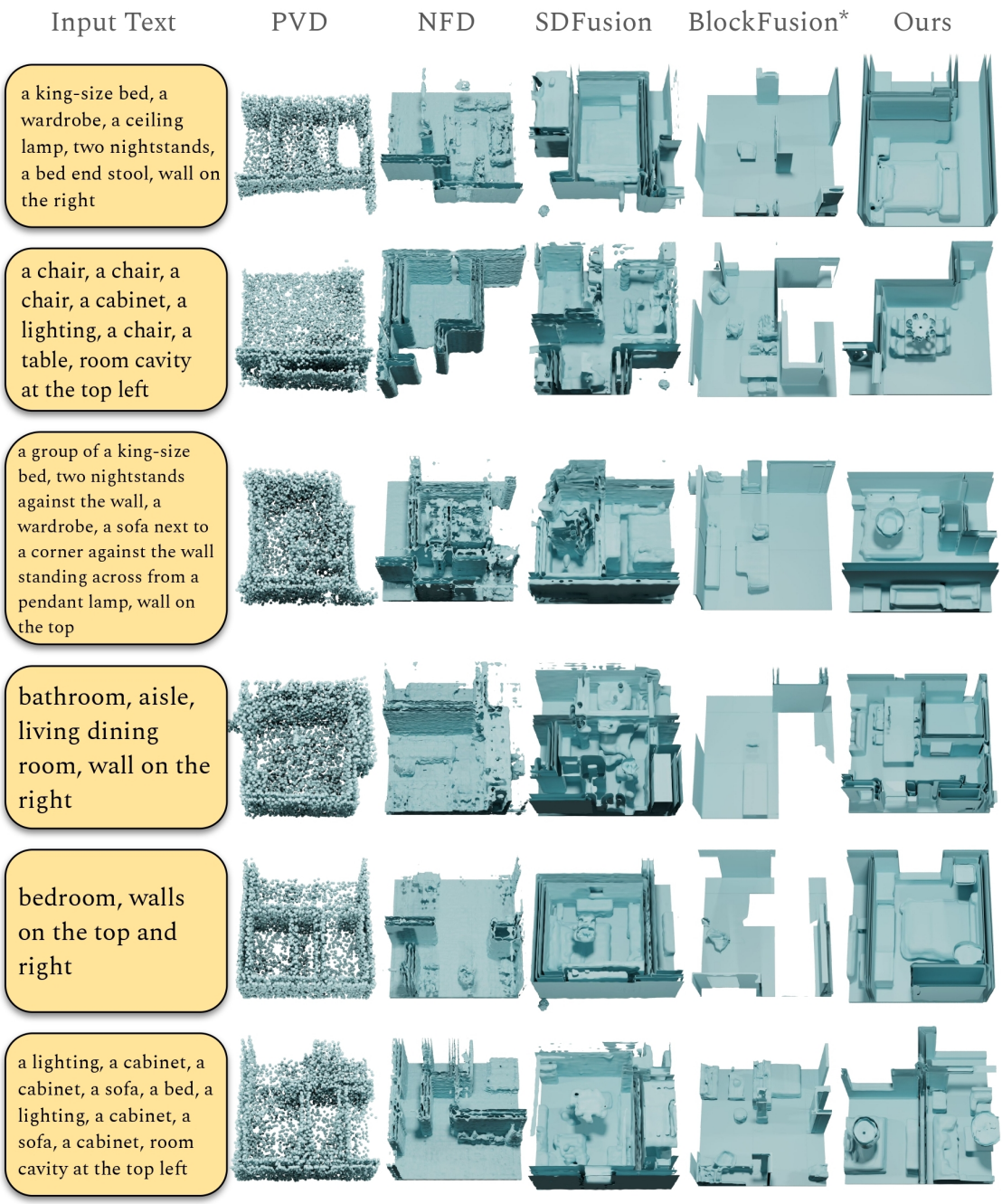

Figure 12. Additional qualitative comparisons to state-of-the-art diffusion-based 3D generative approaches PVD [82], NFD [56], SDFusion [11], and BlockFusion [70] using synthetic captions. Our approach produces sharper scene geometry and more coherent scene structure.
*Note that results for BlockFusion are generated unconditionally