VOXSCENE: VOXEL REPRESENTATION FOR 3D SCENE ARRANGEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Current data-driven layout generation techniques typically encode objects using bounding-box proxies or implicit features. While computationally efficient, such representations abstract away volumetric occupancy, obscuring object boundaries and weakening direct geometric constraints on spatial arrangement. To address these limitations, we introduce VoxScene, which shifts the 3D scene arrangement paradigm to an explicit voxel representation, enabling direct modeling of fine-grained spatial relations. Our method factors the generation process into two stages: First, we generate box-level anchors to specify the coarse layout. Second, an object-centric diffusion network sequentially synthesizes instance-labeled volumetric occupancies conditioned on these anchors. Specifically, each voxel is assigned to at most one instance, preventing inter-instance overlap at the voxel level. These generated voxels further serve as geometric queries for asset retrieval. Extensive experiments on 3D-FRONT demonstrate that our method outperforms existing methods in physical plausibility while maintaining visual quality. Furthermore, results on M3D-Shelf show that VoxScene can generate high-quality fine-grained object arrangements under complex structural constraints, where previous methods struggle.

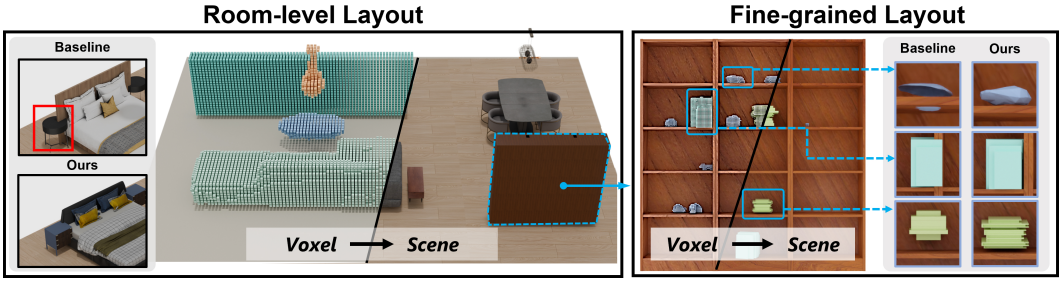

Figure 1: **VoxScene** uses explicit voxel layouts for asset retrieval, reducing object interpenetration (red box) and enabling fine-grained arrangements under complex structural constraints (blue boxes).

1 INTRODUCTION

3D scene synthesis has drawn growing interest because of its extensive applications in virtual asset creation (Maleki & Zhao, 2024; Wang et al., 2025), interior design (Liu et al., 2025a), and embodied AI (Duan et al., 2022; Liu et al., 2025b). These generative techniques significantly accelerate prototype development and reduce manual labor. With the development of deep neural networks, generative models, such as generative adversarial networks (GANs) (Nauata et al., 2020; 2021; Tang et al., 2023), autoregressive models (Vaswani et al., 2017; Esser et al., 2021), and diffusion models (Ho et al., 2020; Rombach et al., 2022), have significantly boosted layout diversity. More recently, large language and vision-language models (LLMs and VLMs) have enabled more flexible natural-language control for 3D scene generation (Feng et al., 2023; Yang et al., 2024b; Çelen et al., 2024; Sun et al., 2025b; Yang et al., 2026a; Fu et al., 2024; Yang et al., 2025), leveraging their pretrained knowledge and reasoning abilities to improve the generalization of existing frameworks.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Despite recent progress, synthesizing physically plausible scenes remains a formidable challenge, especially in dense scenes. These difficulties become more severe when generating fine-grained arrangements of small items under complex structural constraints, where existing methods often break down. Early methods simplify the layout problem by representing objects as bounding boxes (Paschalidou et al., 2021; Lin & Mu, 2024), which omit fine-grained geometry. More recent methods encode object shapes with latent features (Tang et al., 2024; Yang et al., 2024a; Liu et al., 2023), improving efficiency but making precise boundaries less accessible. Other methods use distance fields (Wei et al., 2025; Bokhovkin et al., 2025; Meng et al., 2025) to model intricate geometries. However, they hardly resolve how neighboring objects fit together. These limitations lead to severe spatial mismatches, instance entanglement, and physical collisions. Although LLM-based or VLM-based systems generate and refine layouts via multi-modal feedback or agent tool invocation (Feng et al., 2023; Sun et al., 2025b; Yang et al., 2026a; Pfaff et al., 2026), they remain bottlenecked by an absence of native 3D spatial understanding.

To overcome these limitations, we propose VoxScene (Fig. 1), a voxel diffusion framework that shifts the scene modeling paradigm from ambiguous abstractions to an explicit volumetric representation. Our pipeline generates an explicit voxel layout and then retrieves geometrically matched assets to instantiate the final scene. In our design, each voxel is assigned to at most one instance, which prevents overlap at the layout level. However, jointly generating all instances in a global grid makes it difficult to preserve instance identities and local geometric details for asset retrieval, while incurring high memory costs. Therefore, we adopt a coarse-to-fine pipeline: We start with a 1D diffusion network to generate box-level anchors that establish the coarse layout, followed by an object-centric voxel diffusion network that sequentially synthesizes each object's occupancy in its local grid, conditioned on these anchors. The generated occupancies then serve as geometric queries for asset retrieval.

Extensive experiments were conducted on a room-level dataset 3D-FRONT (red box in Fig. 1) and a shelf-level dataset M3D-Shelf (blue box in Fig. 1). Our method achieves compelling results across multiple metrics, with clear gains in physical plausibility. In particular, existing methods often struggle with fine-grained arrangements of smaller items under complex structural constraints, whereas VoxScene remains reliable in these challenging settings.

Our main contributions can be summarized as follows:

- We introduce an explicit, instance-labeled voxel representation paradigm for 3D scene arrangement. Its mutually exclusive occupancy prevents overlap at the voxel-layout level.
- We design a coarse-to-fine pipeline for 3D scene generation and an object-centric diffusion framework. By sequentially instantiating objects conditioned on prior anchors, our framework preserves instance labels and fine-grained volumetric details.
- We evaluate VoxScene across room-level and shelf-level benchmarks, demonstrating improved physical plausibility while maintaining realistic layouts.

2 Related Work

2.1 Indoor Scene Synthesis

Traditional procedural approaches (Raistrick et al., 2024; Deitke et al., 2022; Yu et al., 2011) synthesize large-scale scenes using hand-crafted rules or heuristics, but require substantial manual design. Recent LLM-based approaches (Zhou et al., 2025; Chen et al., 2026; Sun et al., 2025a; Hu et al., 2024) enable natural-language interaction and open-vocabulary control, although their coverage remains constrained by the underlying procedural rules. Large 3D scene datasets have also enabled data-driven layout generation with autoregressive models (Paschalidou et al., 2021; Wang et al., 2021; Feng et al., 2025; Bucher & Armeni, 2025), masked non-autoregressive models (Choi et al., 2026), and diffusion models (Tang et al., 2024; Hu et al., 2026; Lin & Mu, 2024; Zhai et al., 2024; 2023; Yang et al., 2026b; 2024a; Sun et al., 2026). Another line of work transfers 2D priors to 3D scenes through distillation (Zhou et al., 2024; Gao et al., 2024; Fu et al., 2024) or multi-view generation (Chen et al., 2025; Li et al., 2026; Fang et al., 2025). More recently, multimodal agent systems (Yang et al., 2026a; Pfaff et al., 2026; Xia et al., 2026) have used LLMs to plan layouts

and coordinate external tools. This enables iterative scene refinement, but repeated tool calls and physics-based optimization make generation time-consuming.

2.2 Representations in Scene Synthesis

Representations affect both physical plausibility and the geometric queries used for asset retrieval. Early methods such as ATISS (Paschalidou et al., 2021) describe objects with bounding-box attributes. While effective for coarse spatial planning, these proxies overlook fine-grained geometry. Subsequent approaches (Tang et al., 2024) encode objects into latent shape codes using pretrained models (Liu et al., 2023; Yang et al., 2018). Their compact form facilitates integration into layout networks, but compressing geometry also obscures explicit boundaries. As a result, objects placed in the generated layout can intersect after mesh retrieval. PhyScene (Yang et al., 2024a) introduces physical constraints to mitigate this problem, yet geometric mismatches between proxies and meshes can still cause collisions and boundary violations. CASAGPT (Feng et al., 2025) instead decomposes meshes into explicit cuboid primitives for autoregressive modeling to avoid object intersections. However, these approximations sacrifice curved surfaces and irregular geometric details. In contrast, VoxScene represents object occupancy with mutually exclusive voxels, preserving volumetric boundaries at the voxel level and enabling more plausible retrieval.

2.3 Voxel Diffusion for Scene Synthesis

At the object level, XCube (Ren et al., 2024), Structured 3D Latents (Xiang et al., 2025), Vox-E (Sella et al., 2023), and FullPart (Ding et al., 2025) show the value of structured volumetric representations, but extending these designs to complex scenes remains challenging. Scene-level methods such as BlockFusion (Wu et al., 2024), SceneFactor (Bokhovkin et al., 2025), and LT3SD (Meng et al., 2025) generate geometry through block-level latent representations. Although these designs scale to large environments, their outputs describe scenes or regions rather than individual object instances. This makes it difficult to retrieve and edit assets independently. Our object-centric pipeline addresses this limitation by maintaining instance labels while preserving local geometric detail.

3 Method

VoxScene generates a scene in two stages: box-level anchor generation for coarse layout planning, followed by object-centric voxel generation for detailed geometry. The resulting instance-labeled occupancies guide asset retrieval to assemble the final scene (Fig. 2).

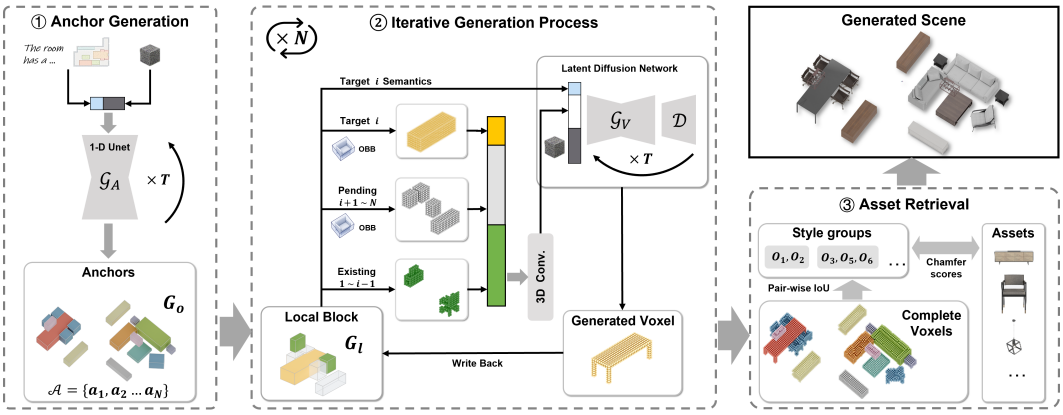

Figure 2: **Method overview.** $\mathcal{G}_A$ generates box-level anchors (Part 1). Given these anchors and the surrounding context, $\mathcal{G}_V$ generates object latents that D decodes into voxel occupancies for scene write-back (Part 2). Asset retrieval, with optional style clustering, assembles the final scene (Part 3).

3.1 DIFFUSION MODEL

Both stages follow the denoising diffusion probabilistic model (DDPM) formulation (Ho et al., 2020). For a clean representation $\mathbf{h}_0$ and noise schedule $\{\beta_t\}_{t=1}^T$, define $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$. The noisy representation at timestep t is

$$\mathbf{h}_t = \sqrt{\bar{\alpha}_t}\mathbf{h}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{1}$$

Sampling starts from Gaussian noise and reverses this process using a conditional denoiser. Given scene conditions $\mathcal{C}$, we generate the anchor set $\mathcal{A}$ followed by a global voxel grid G_o, where each voxel stores an instance ID or 0 for free space:

$$p_{\phi,\theta}(\mathcal{A}, G_o \mid \mathcal{C}) = p_\phi(\mathcal{A} \mid \mathcal{C})\, p_\theta(G_o \mid \mathcal{A}, \mathcal{C}), \tag{2}$$

where ϕ and θ parameterize the anchor denoiser $\mathcal{G}_A$ and voxel denoiser $\mathcal{G}_V$, respectively.

3.2 ANCHOR GENERATION

Formally, let $\mathcal{A} = \{\mathbf{a}_i\}_{i=1}^N$ denote the set of spatial anchors, where N is the number of target objects. Each anchor is parameterized as $\mathbf{a}_i = \{\mathbf{p}_i, \mathbf{r}_i, \mathbf{s}_i, \mathbf{y}_i\}$, encapsulating the spatial position $\mathbf{p}_i \in \mathbb{R}^3$, the continuous 2D heading orientation $\mathbf{r}_i \in \mathbb{R}^2$, the geometric size $\mathbf{s}_i \in \mathbb{R}^3$, and a semantic category vector $\mathbf{y}_i \in \{0,1\}^{N_C}$, where N_C denotes the total number of object classes.

We use a 1D U-Net as the denoiser $\mathcal{G}_A$ to process a fixed number of object slots $N_{\max}$, with unused slots assigned to an empty category. For each slot, we encode the anchor attribute groups with their respective MLPs and sum the resulting features as input to $\mathcal{G}_A$. The network then predicts each attribute group through dedicated output heads. The condition $\mathcal{C}$ can be a floorplan, text, or known objects. Floorplan masks and text are encoded by ResNet18 (He et al., 2016) and a pretrained BERT model (Devlin et al., 2019), respectively. For a partial layout containing N_{known} known objects, we retain their attributes in the first N_{known} slots and encode them with a shared MLP.

We train $\mathcal{G}_A$ using velocity prediction. Let $\mathbf{X}_0$ denote the clean first-stage representation and $\mathbf{X}_t$ its noisy counterpart obtained using Eq. (1). Given $\mathbf{X}_t$, t, and $\mathcal{C}$, the denoiser is trained to predict the velocity target

$$\mathbf{v}_t^{\mathrm{A}} = \sqrt{\bar{\alpha}_t}\boldsymbol{\epsilon} - \sqrt{1-\bar{\alpha}_t}\mathbf{X}_0. \tag{3}$$

For the denoising loss, we group each anchor's geometric attributes into $\mathbf{b}_i = [\mathbf{p}_i^\top, \mathbf{s}_i^\top, \mathbf{r}_i^\top]^\top \in \mathbb{R}^8$ and keep its category vector $\mathbf{y}_i$ separate:

$$\mathcal{L}_{\text{attr}} = \mathbb{E}_{\mathbf{X}_0,\mathcal{C},t,\boldsymbol{\epsilon}}\left[\bar{\alpha}_t \sum_{g\in\{b,y\}} \mathrm{MSE}\big([\mathcal{G}_A(\mathbf{X}_t, t, \mathcal{C})]_g, [\mathbf{v}_t^{\mathrm{A}}]_g\big)\right], \tag{4}$$

Both losses include all $N_{\max}$ slots.

3.3 OBJECT-CENTRIC VOXEL GENERATION

Object-centric Modeling. For each anchor $\mathbf{a}_i$, we perform synthesis within a local block G_l of fixed resolution K^3, which is centered at $\mathbf{p}_i$ and aligned with the heading orientation $\mathbf{r}_i$. Since instance identifiers i are arbitrarily assigned and lack cross-scene consistency, directly feeding them into the neural network would severely disrupt the learning process. To address this, we remap these absolute identifiers into a relative state $\ell \in \{0,1,2\}$, corresponding to free space ($\ell = 0$), contextual objects ($\ell = 1$), and the target object ($\ell = 2$), respectively. Once generated, the voxels classified as $\ell = 2$ are inversely transformed and written back into the global grid G_o. This operation is exclusively restricted to the remaining free space in G_o, which ensures that previously established geometries are preserved throughout the iterative process.

Latent Space Compression. To reduce the cost of diffusion on local voxel grids, we use a fully convolutional 3D Vector-Quantized Variational Autoencoder (VQ-VAE) with a spatial compression factor of 4. The local input $\mathbf{x}$ consists of ternary states $\mathbf{x}_{\text{state}}$ and a continuous distance field $\mathbf{x}_{\text{SDF}}$. An encoder E and a discrete codebook map this input to a quantized latent $\mathbf{z}_q$. A shared decoder

backbone D supports a state prediction head H_{state} and an auxiliary geometry head H_{SDF}. We train these components with

$$\begin{aligned}\mathcal{L}_{\text{VQ}} = \mathcal{L}_{\text{CE}}(\mathbf{x}_{\text{state}}, H_{\text{state}}(D(\mathbf{z}_q))) + \mathcal{L}_{\text{quant}}(E(\mathbf{x}), \mathbf{z}_q) \\ + \mathcal{L}_1(\mathbf{x}_{\text{SDF}}, H_{\text{SDF}}(D(\mathbf{z}_q))).\end{aligned} \tag{5}$$

Here, $\mathcal{L}_{\text{CE}}$ supervises the ternary state predictions, $\mathcal{L}_{\text{quant}}$ combines the codebook and commitment losses, and $\mathcal{L}_1$ supervises the distance field. We compute $\mathbf{x}_{\text{SDF}}$ from the isolated target object to provide geometric supervision. At inference, the continuous latent sampled by the diffusion model is quantized through the VQ-VAE codebook before decoding.

Anchor Conditions. For a sampled anchor order π, the object occupancies are generated sequentially:

$$p_\theta(\mathbf{v}_{\pi_1}, \ldots, \mathbf{v}_{\pi_N} \mid \mathcal{A}, \mathcal{C}, \pi) = \prod_{i=1}^{N} p_\theta(\mathbf{v}_{\pi_i} \mid \mathbf{v}_{\pi_{<i}}, \mathbf{a}_{\pi_i}, \mathcal{A}_{\pi_{>i}}, \mathcal{C}), \tag{6}$$

where $\mathbf{v}_{\pi_{<i}}$ are previously generated occupancies and $\mathcal{A}_{\pi_{>i}}$ are pending anchors. We encode generated voxels $\mathcal{V}_{\text{gen}}$, the target anchor $\mathbf{a}_i$, and pending anchors $\mathcal{A}_{\text{p}}$ as $\mathbf{c}_{\text{gen}}$, $\mathbf{c}_{\text{tar}}$, and $\mathbf{c}_{\text{pend}}$, respectively (Part 2 of Fig. 2). The complete local condition is $\mathbf{c}_i = (\mathbf{c}_{\text{gen}}, \mathbf{c}_{\text{tar}}, \mathbf{c}_{\text{pend}})$. Structural constraints such as walls and floors can also be converted to voxels $\mathbf{v}_S$ and put into $\mathcal{V}_{\text{gen}}$ as conditions. The generated context $\mathcal{V}_{\text{gen}}$ is rasterized into an N_C-channel tensor using its voxel occupancies and semantic labels. Since geometries of pending objects are unavailable, their anchors are converted to coarse Oriented Bounding Boxes (OBBs) and voxelized into another N_C-channel tensor. Overlapping OBBs are merged by a logical OR operation, producing a multi-hot semantic encoding.

For the target object condition $\mathbf{c}_{\text{tar}}$, we separate semantic and spatial signals. The category vector $\mathbf{y}_i$ specifies the object class. The spatial signal is a two-channel tensor: one channel encodes the target OBB to specify the generation region, while the other encodes an anisotropic volumetric kernel of size $\mathbf{s}_k \in \mathbb{Z}^3$ to indicate the synthesis origin. This volumetric kernel provides a spatial signal for the convolutional down-sampling network and supports the anchor shifting strategy in Section 3.4.

All spatial condition tensors ($\mathbf{c}_{\text{gen}}$, $\mathbf{c}_{\text{pend}}$, and the spatial part of $\mathbf{c}_{\text{tar}}$) are processed through a 3D convolutional down-sampling network to the VQ-VAE latent resolution and concatenated with the noisy latent features. The semantic vector $\mathbf{y}_i$ is broadcast across the latent space and concatenated as category conditioning for the target object.

Denoising Process. Let $\mathbf{z}_0$ denote its clean latent and $\mathbf{z}_t$ its noisy counterpart sampled using Eq. (1), with $\mathbf{h}_0 = \mathbf{z}_0$. The voxel denoiser $\mathcal{G}_V$ is a 3D U-Net that takes $(\mathbf{z}_t, t, \mathbf{c}_i)$ as input and predicts the velocity target

$$\mathbf{u}_t \coloneqq \sqrt{\bar{\alpha}_t}\boldsymbol{\epsilon} - \sqrt{1-\bar{\alpha}_t}\mathbf{z}_0. \tag{7}$$

The network is trained with the mean squared error

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{(\mathbf{z}_0, \mathbf{c}_i),\, t,\, \boldsymbol{\epsilon}} \left[\|\mathcal{G}_V(\mathbf{z}_t, t, \mathbf{c}_i) - \mathbf{u}_t\|_2^2 \right], \tag{8}$$

where the expectation is over training pairs $(\mathbf{z}_0, \mathbf{c}_i)$, sampled diffusion timesteps t, and Gaussian noise $\boldsymbol{\epsilon}$. The denoiser parameterizes $p_\theta(\mathbf{z}_{t-1} \mid \mathbf{z}_t, \mathbf{c}_i)$; the decoded voxel states are integrated into G_o using the write-back rule described above.

3.4 Training Policy

We train the voxel diffusion model with stochastic context masking and anchor shifting, as illustrated in Fig. 3.

Stochastic Context Masking. To empower the model to synthesize scenes in an arbitrary order, we introduce a stochastic context masking strategy during the training phase. For each surrounding object within the local data block, we independently apply a dropout probability (empirically set to $p = 0.5$) to determine its generation state. Based on this probability, a context object is randomly partitioned into either the generated set $\mathcal{V}_{\text{gen}}$ or the pending set $\mathcal{A}_{\text{p}}$. This randomized masking encourages the network to adapt to diverse context densities and arbitrary generation sequences encountered during inference.

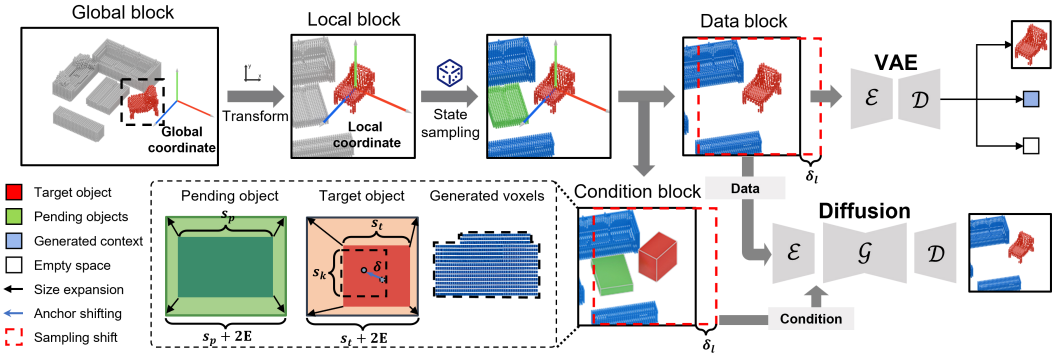

Figure 3: **Training policy**. We employ a stochastic masking policy to enable our model to generate in an arbitrary sequence. The anchor shift policy further improves the robustness and effectiveness.

Anchor shifting. First-stage anchors provide coarse spatial guidance but may contain spatial noise. To prevent the model from relying on exact anchor-object alignment as a positional shortcut, we perturb the target anchor during training by applying a random translation $\boldsymbol{\delta}$, sampled uniformly from $[-\mathbf{E}, \mathbf{E}] \cap \mathbb{Z}^3$, where $\mathbf{E}$ specifies the maximum offset along each axis in voxel units. Accordingly, we set the volumetric kernel size to $\mathbf{s}_k = 2\mathbf{E} + 1$, ensuring that the original anchor center remains covered. This allows variation in object position around the anchor. The OBB extents of the target and pending anchors, originally $\mathbf{s}_t$ and $\mathbf{s}_p$, are expanded to $\mathbf{s}_t + 2\mathbf{E}$ and $\mathbf{s}_p + 2\mathbf{E}$ respectively, preserving coverage of the original object occupancies. The expanded OBBs provide a larger region for voxel generation. Furthermore, to reduce bias toward the center of G_l, we independently shift the local grid origin by $\boldsymbol{\delta}_l$, sampled uniformly from $[-\mathbf{E}, \mathbf{E}]$. These policies give the model spatial flexibility to generate more plausible occupancies from the surrounding context.

3.5 Anisotropic Asset Retrieval

We instantiate each generated voxel object with a realistic model. All models are pre-normalized into a canonical unit coordinate system and voxelized on a fixed K_A^3 grid for asset matching, where K_A is the per-axis asset resolution. For each generated object, we extract its raw voxel occupancy and apply an inverse transformation matrix R_v to map it into the canonical unit space. We identify the optimal geometric match by computing the Soft Chamfer Score. Let $Q = \{q_i\}_{i=1}^{|Q|}$ and $C = \{c_j\}_{j=1}^{|C|}$ denote the voxel sets of the generated query and the candidate shapes, respectively. The unidirectional point-to-set distances are defined as:

$$d(q_i, C) = \min_{c_j \in C} \|q_i - c_j\|_2, \quad d(c_j, Q) = \min_{q_i \in Q} \|c_j - q_i\|_2. \tag{9}$$

To robustly evaluate the structural similarity against voxel discretization noise, the Soft Chamfer Score is formulated as a symmetric Gaussian-weighted average:

$$S(Q, C) = \frac{1}{2}\left(\frac{1}{|Q|}\sum_{q_i \in Q} e^{-\frac{d(q_i,C)^2}{2\sigma^2}} + \frac{1}{|C|}\sum_{c_j \in C} e^{-\frac{d(c_j,Q)^2}{2\sigma^2}}\right), \tag{10}$$

where σ is the Gaussian tolerance in voxel units. We select the candidate asset M_1 that yields the maximum score $S(Q, C)$. However, implausible anchors may produce low-quality voxel occupancies that poorly match the available assets. We therefore discard objects whose best matching score falls below a threshold τ_{ret}. For retained objects, the selected asset is scaled to fit the generated bounding box.

Style Clustering. To improve stylistic consistency in room scenes, we group objects of the same category with similar aligned voxel occupancies and sizes. We retrieve an asset for each object, then assign the highest-scoring retrieved asset to all members of its group (Fig. 6). The grouping criteria and thresholds are detailed in Appendices B.3 and C.2.

4 EXPERIMENT

4.1 EXPERIMENTAL SETUP

Datasets. We evaluate the superiority of VoxScene on two distinct datasets spanning different spatial scales: 3D-FRONT (Fu et al., 2021) and M3D-Shelf (Zhang et al., 2025). 3D-FRONT is a large-scale benchmark dataset for indoor scene synthesis. M3D-Shelf is a customized subset extracted from M3DLayout, concentrating on delicate object arrangement in the shelf region. We evaluate 1,000 generated scenes per 3D-FRONT room category and 200 generated scenes on M3D-Shelf.

Baselines and Model Configuration. On 3D-FRONT, we compare VoxScene with four baselines: DiffuScene (Tang et al., 2024), PhyScene (Yang et al., 2024a), Planner3D (Wei et al., 2025), and LayoutGPT (Feng et al., 2023). We could not include CASAGPT (Feng et al., 2025) since its implementation was not publicly available at the time of our experiments. In M3D-Shelf, we choose DiffuScene and SceneWeaver (Yang et al., 2026a) as our baselines. VoxScene treats shelf arrangement as a scene completion task, with the shelf provided as a known-object condition. SceneWeaver uses its complete generation pipeline to create an open-front shelf from a room prompt and then populate it with objects. For retrieval-based methods, we use 3D-FUTURE and M3D-Shelf CAD assets. VoxScene uses style clustering on 3D-FRONT and retrieves assets independently on M3D-Shelf. We report results with and without retrieval filtering on both datasets. For Planner3D, we evaluate its generated object meshes directly.

4.2 EVALUATION ON 3D-FRONT

Evaluation Metrics We employ a suite of metrics focusing on statistical realism, physical plausibility, and shape diversity. Fréchet Inception Distance (FID) (Heusel et al., 2017) assesses overall distribution quality. For collisions, we report average pairwise intersection I_p and global overlap ratio OR (both $\times 10^3$ in the tables). Regarding diversity, category-weighted diversity CD calculates the utilization rate of unique CAD assets retrieved from the dataset.

Table 1: Quantitative results on 3D-FRONT. The upper block uses text conditioning, while the lower block uses floorplan-mask conditioning. The best and second-best results for each metric within each block are marked in **bold** and underlined, respectively; – denotes an unreported result.

Methods	Bedroom				Livingroom				Diningroom			
	$FID\downarrow$	$I_p\downarrow$	$OR\downarrow$	$CD\uparrow$	$FID\downarrow$	$I_p\downarrow$	$OR\downarrow$	$CD\uparrow$	$FID\downarrow$	$I_p\downarrow$	$OR\downarrow$	$CD\uparrow$
DiffuScene	**25.37**	8.53	20.45	0.522	41.77	2.34	8.35	0.545	38.23	2.51	19.16	0.555
Planner3D	73.98	18.22	14.02	–	89.98	9.20	19.45	–	85.32	7.12	22.38	–
LayoutGPT	33.69	19.88	8.91	0.539	42.49	4.24	8.89	**0.961**	45.63	3.09	7.92	0.584
VoxScene τ_{ret}=0	27.16	7.64	9.54	**0.715**	39.32	2.12	7.78	0.918	38.09	1.88	12.57	**0.845**
VoxScene τ_{ret}=0.75	27.52	**6.15**	**6.82**	0.674	**39.06**	**1.52**	**4.79**	0.809	**37.88**	**1.32**	**4.17**	0.743
PhyScene	–	–	–	–	45.25	3.30	13.36	0.292	–	–	–	–
VoxScene τ_{ret}=0	–	–	–	–	42.72	3.42	9.76	**1.069**	–	–	–	–
VoxScene τ_{ret}=0.75	–	–	–	–	**42.29**	**2.26**	**5.60**	0.931	–	–	–	–

Results. As shown in Table 1, VoxScene achieves competitive results across all metrics in each room. Directly using its voxel representation generally yields lower collision rate than the other data-driven baselines, and filtering mechanism further reduces intersections, giving VoxScene the lowest I_p and OR within each conditioning setting. A slightly higher bedroom FID than DiffuScene's reflects a multi-objective trade-off: suppressing interpenetration can shift the distribution of top-down semantic renderings while yielding better physical metrics.

DiffuScene and PhyScene compress object shape into latent codes, which obscure volumetric boundaries. Planner3D reconstructs independent SDFs from scene-graph features, but the resulting objects can still overlap after placement. These abstractions limit the models' ability to capture geometric interactions between objects. LayoutGPT relies on an LLM to predict bounding-box layouts, but the model's limited 3D geometric understanding makes it difficult to account for actual object shapes

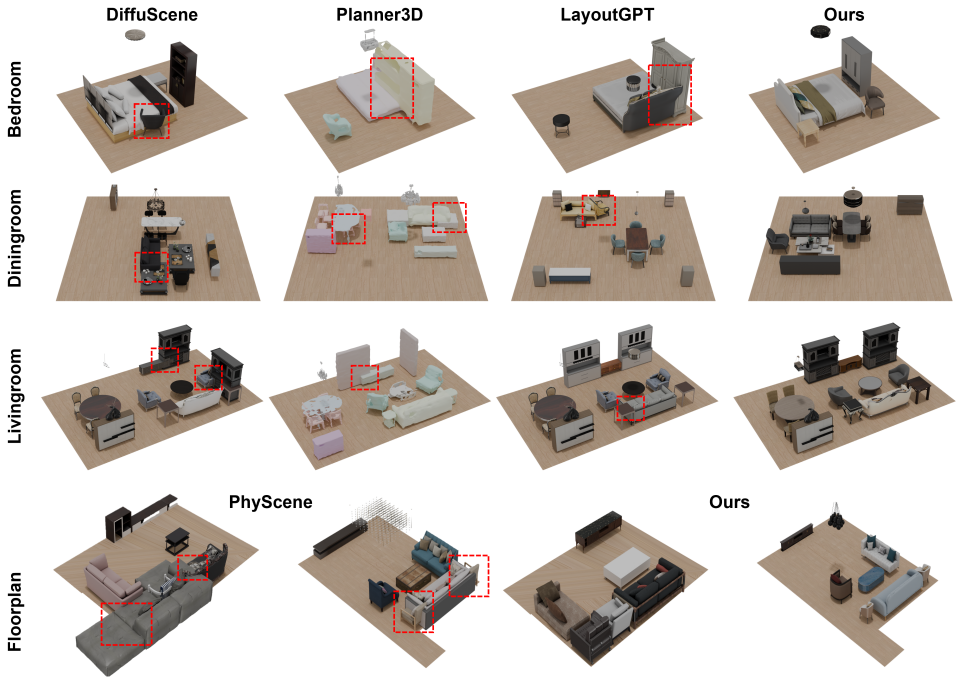

Figure 4: **Qualitative comparisons in 3D-FRONT**.

during placement. The comparisons are illustrated in Fig. 4. For shape diversity, voxel-based retrieval alleviates the mode collapse observed in latent-code retrieval, where generated queries repeatedly map to a limited subset of CAD assets.

4.3 Evaluation on M3D-Shelf

Evaluation Metrics. For physical evaluation, we measure out-of-bounds (R_o), floating (R_f), and shelf-colliding (R_s) object ratios, together with the overlap ratio OR. In addition, we use GPT-5.4 and a user study to assess realism (S_r), layout quality (S_l), and completion (S_c) on a 0–10 scale. In the user study, 12 participants rate ten randomly sampled renderings per method. Table 2 reports the physical metrics and mean scores.

Table 2: Quantitative results and user study on M3D-Shelf. The best results are marked in **bold**. - denotes an unreported score.

Methods	Physical metrics				GPT-5.4 evaluation			User study		
	$R_o \downarrow$	$R_f \downarrow$	$R_s \downarrow$	$OR \downarrow$	$S_r \uparrow$	$S_l \uparrow$	$S_c \uparrow$	$S_r \uparrow$	$S_l \uparrow$	$S_c \uparrow$
DiffuScene	0.408	0.830	0.580	30.60	4.715	3.565	**5.965**	2.76	2.54	4.07
SceneWeaver	0.398	0.214	0.415	8.60	5.854	5.000	5.177	4.79	4.51	4.13
VoxScene τ_{ret}=0	0.053	0.184	0.034	2.70	–	–	–	–	–	–
VoxScene τ_{ret}=0.75	**0.051**	**0.165**	**0.027**	**2.27**	**6.275**	**5.950**	5.900	**5.63**	**5.58**	**5.96**

Results. As demonstrated in Table 2, VoxScene outperforms these baselines in various metrics. As illustrated in Fig. 5, although DiffuScene performs reasonably on room-level layouts, it fails to model fine-grained object arrangement under complex structural constraints. Similarly, while SceneWeaver employs iterative rule-based optimization to refine arrangements and somewhat mitigates these errors, it still leaves a substantial number of unresolved collisions. In contrast, VoxScene respects the structural constraints and produces plausible arrangements.

In our user study (shown in Table 2), participants rate VoxScene highest on all three criteria. The human rankings for realism and layout quality agree with those of GPT-5.4. For completion, however, participants rate VoxScene (5.96) above SceneWeaver (4.13) and DiffuScene (4.07). This dis-

crepancy suggests that human judgments of completion also reflect the visual quality and physical plausibility of the arrangement, beyond the amount of visible shelf content.



Figure 5: **Qualitative comparisons in M3D-Shelf.** We chose the most similar scene from SceneWeaver as a comparison.

4.4 Ablation Study

Table 3: **Ablation study on Livingroom**. Lower values are better. The best result is marked in **bold**.

Settings	$FID \downarrow$	$I_p \downarrow$	$OR \downarrow$
Anchor boxes + size-based retrieval	41.76	2.91	11.17
w/o SDF head	42.69	**1.40**	**3.43**
w/o anchor shifting	41.40	1.91	5.12
w/o generated-object context	39.72	3.56	8.21
w/o pending anchor conditions	39.45	1.60	4.24
w/o style clustering	39.17	1.50	4.74
Full	**39.06**	1.52	4.79

We conduct ablation studies on the *livingroom* split of the 3D-FRONT dataset, with quantitative results summarized in Table 3. Replacing voxel generation with direct size-based CAD retrieval from anchor boxes worsens all three metrics. Within voxel generation, removing the SDF head raises FID from 39.06 to 42.69, the largest degradation among the tested ablations, despite reducing I_p and OR. Anchor shifting helps the model accommodate spatial variations in the anchors, and removing it worsens all three metrics. The context ablations further show the role of surrounding objects in spatial coordination: removing generated-object context increases all metrics. The pending anchor condition $\mathbf{c}_{\text{pend}}$ provides spatial guidance for objects yet to be generated, and omitting it can lower realism and lead to more collision pairs. At the retrieval stage, style clustering introduces a trade-off between stylistic consistency and geometric matching accuracy. Assigning a shared representative CAD model to similar objects improves visual coherence but can introduce slight geometric deviations from individual voxel-based matches, leading to small increases in collisions.

5 Conclusions

We introduced VoxScene, a voxel diffusion framework for 3D scene arrangement. Factoring the task into two stages, such explicit representation preserves object boundaries and local geometric detail, enabling more physically plausible arrangements. Experiments on room and shelf scenes demonstrate its effectiveness, particularly for fine-grained object placement under complex structural constraints, where existing methods often struggle.

Limitations and Future Work. We currently use fixed-size local blocks, so accommodating large objects incurs unnecessary computation for smaller ones. Training and asset retrieval also focus primarily on geometry, without explicit objectives for appearance or scene-level stylistic coherence. In our future work, we will explore scale-normalized generation in canonical object coordinates and incorporate appearance and style objectives. We also plan to extend VoxScene from asset retrieval to direct 3D asset synthesis.

6 AI USE STATEMENT

Generative AI tools were used to improve the clarity and readability of the manuscript, support literature retrieval and discovery, and assist with implementation, code generation, debugging, and interpretation of experimental results. The authors developed the research ideas, methodology, and experimental design and made the final technical decisions. All AI-assisted content, references, and generated code were reviewed and verified by the authors, who take full responsibility for the final content of the paper.

7 ETHICS STATEMENT

Twelve adults voluntarily participated in the user study by rating rendered scenes. We collected only their ratings, without names or other identifying information, and report the results in aggregate.

REFERENCES

Aleksey Bokhovkin, Quan Meng, Shubham Tulsiani, and Angela Dai. Scenefactor: Factored latent 3d diffusion for controllable 3d scene generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 628–639, 2025.

Martin JJ Bucher and Iro Armeni. Respace: Text-driven autoregressive 3d indoor scene synthesis and editing. *arXiv preprint arXiv:2506.02459*, 2025.

Ata Çelen, Guo Han, Konrad Schindler, Luc Van Gool, Iro Armeni, Anton Obukhov, and Xi Wang. I-design: Personalized llm interior designer. In *European Conference on Computer Vision*, pp. 217–234. Springer, 2024.

ChunTeng Chen, YiChen Hsu, YiWen Liu, WeiFang Sun, TsaiChing Ni, ChunYi Lee, Min Sun, and YuanFu Yang. Scenefoundry: Generating interactive infinite 3d worlds. *arXiv preprint arXiv:2601.05810*, 2026.

Yiwen Chen, Hieu T Nguyen, Vikram Voleti, Varun Jampani, and Huaizu Jiang. Housecrafter: Lifting floorplans to 3d scenes with 2d diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 28440–28450, 2025.

Jeongjun Choi, Yeonsoo Park, and H Jin Kim. Scenenat: Masked generative modeling for language-guided indoor scene synthesis. *arXiv preprint arXiv:2601.07218*, 2026.

Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems*, 35:5982–5994, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Lihe Ding, Shaocong Dong, Yaokun Li, Chenjian Gao, Xiao Chen, Rui Han, Yihao Kuang, Hong Zhang, Bo Huang, Zhanpeng Huang, et al. Fullpart: Generating each 3d part at full resolution. *arXiv preprint arXiv:2510.26140*, 2025.

Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(2):230–244, 2022.

Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883, 2021.

Chuan Fang, Heng Li, Yixun Liang, Jia Zheng, Yongsen Mao, Yuan Liu, Rui Tang, Zihan Zhou, and Ping Tan. Spatialgen: Layout-guided 3d indoor scene generation. *arXiv preprint arXiv:2509.14981*, 3, 2025.

Weitao Feng, Hang Zhou, Jing Liao, Li Cheng, and Wenbo Zhou. Casagpt: cuboid arrangement and scene assembly for interior design. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 29173–29182, 2025.

Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems*, 36: 18225–18250, 2023.

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*, pp. 10933–10942, 2021.

Rao Fu, Zehao Wen, Zichen Liu, and Srinath Sridhar. Anyhome: Open-vocabulary generation of structured and textured 3d homes. In *European Conference on Computer Vision*, pp. 52–70. Springer, 2024.

Gege Gao, Weiyang Liu, Anpei Chen, Andreas Geiger, and Bernhard Schölkopf. Graphdreamer: Compositional 3d scene synthesis from scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21295–21304, 2024.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in neural information processing systems*, volume 33, pp. 6840–6851, 2020.

Siyi Hu, Diego Martin Arroyo, Stephanie Debats, Fabian Manhardt, Luca Carlone, and Federico Tombari. Mixed diffusion for 3d indoor scene synthesis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1262–1272, 2026.

Ziniu Hu, Ahmet Iscen, Aashi Jain, Thomas Kipf, Yisong Yue, David A Ross, Cordelia Schmid, and Alireza Fathi. Scenecraft: An llm agent for synthesizing 3d scenes as blender code. In *Forty-first International Conference on Machine Learning*, 2024.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Sikuang Li, Chen Yang, Jiemin Fang, Taoran Yi, Jia Lu, Jiazhong Cen, Lingxi Xie, Wei Shen, and Qi Tian. Worldgrow: Generating infinite 3d world. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 6433–6441, 2026.

Chenguo Lin and Yadong Mu. Instructscene: Instruction-driven 3d indoor scene synthesis with semantic graph prior. *arXiv preprint arXiv:2402.04717*, 2024.

Jiachen Liu, Yuan Xue, Haomiao Ni, Rui Yu, Zihan Zhou, and Sharon X Huang. Computer-aided layout generation for building design: A review. *Computational Visual Media*, 2025a.

Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. Openshape: Scaling up 3d shape representation towards open-world understanding. *Advances in neural information processing systems*, 36:44860–44879, 2023.

Yang Liu, Weixing Chen, Yongjie Bai, Xiaodan Liang, Guanbin Li, Wen Gao, and Liang Lin. Aligning cyber space with physical world: A comprehensive survey on embodied ai. *IEEE/ASME Transactions on Mechatronics*, 2025b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Mahdi Farrokhi Maleki and Richard Zhao. Procedural content generation in games: A survey with insights on emerging llm integration. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 20, pp. 167–178, 2024.

Quan Meng, Lei Li, Matthias Nießner, and Angela Dai. Lt3sd: Latent trees for 3d scene diffusion. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 650–660, 2025.

Nelson Nauata, Kai-Hung Chang, Chin-Yi Cheng, Greg Mori, and Yasutaka Furukawa. House-gan: Relational generative adversarial networks for graph-constrained house layout generation. In *European Conference on Computer Vision*, pp. 162–177. Springer, 2020.

Nelson Nauata, Sepidehsadat Hosseini, Kai-Hung Chang, Hang Chu, Chin-Yi Cheng, and Yasutaka Furukawa. House-gan++: Generative adversarial layout refinement network towards intelligent computational agent for professional architects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13632–13641, 2021.

Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. *Advances in neural information processing systems*, 34:12013–12026, 2021.

Nicholas Pfaff, Thomas Cohn, Sergey Zakharov, Rick Cory, and Russ Tedrake. Scenesmith: Agentic generation of simulation-ready indoor scenes. *arXiv preprint arXiv:2602.09153*, 2026.

Alexander Raistrick, Lingjie Mei, Karhan Kayan, David Yan, Yiming Zuo, Beining Han, Hongyu Wen, Meenal Parakh, Stamatis Alexandropoulos, Lahav Lipson, et al. Infinigen indoors: Photorealistic indoor scenes using procedural generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21783–21794, 2024.

Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. Xcube: Large-scale 3d generative modeling using sparse voxel hierarchies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4209–4219, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Etai Sella, Gal Fiebelman, Peter Hedman, and Hadar Averbuch-Elor. Vox-e: Text-guided voxel editing of 3d objects, 2023.

Chunyi Sun, Junlin Han, Weijian Deng, Xinlong Wang, Zishan Qin, and Stephen Gould. 3d-gpt: Procedural 3d modeling with large language models. In *2025 International Conference on 3D Vision (3DV)*, pp. 1253–1263. IEEE, 2025a.

Fan-Yun Sun, Weiyu Liu, Siyi Gu, Dylan Lim, Goutam Bhat, Federico Tombari, Manling Li, Nick Haber, and Jiajun Wu. Layoutvlm: Differentiable optimization of 3d layout via vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 29469–29478, 2025b.

Xiaohao Sun, Divyam Goel, and Angel X Chang. Semlayoutdiff: Semantic layout generation with diffusion model for indoor scene synthesis. pp. 1545–1555, 2026.

Hao Tang, Zhenyu Zhang, Humphrey Shi, Bo Li, Ling Shao, Nicu Sebe, Radu Timofte, and Luc Van Gool. Graph transformer gans for graph-constrained house generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2173–2182, 2023.

Jiapeng Tang, Yinyu Nie, Lev Markhasin, Angela Dai, Justus Thies, and Matthias Nießner. Diffuscene: Denoising diffusion models for generative indoor scene synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20507–20518, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers. In *2021 International conference on 3D vision (3DV)*, pp. 106–115. IEEE, 2021.

Yuntao Wang, Qinnan Hu, Zhou Su, Linkang Du, Qichao Xu, and Weiwei Li. Large model empowered metaverse: State-of-the-art, challenges and opportunities. *IEEE Network*, 2025.

Yao Wei, Martin Renqiang Min, George Vosselman, Li Erran Li, and Michael Ying Yang. Planner3d: Llm-enhanced graph prior meets 3d indoor scene explicit regularization. *IEEE transactions on pattern analysis and machine intelligence*, 2025.

Zhennan Wu, Yang Li, Han Yan, Taizhang Shang, Weixuan Sun, Senbo Wang, Ruikai Cui, Weizhe Liu, Hiroyuki Sato, Hongdong Li, et al. Blockfusion: Expandable 3d scene generation using latent tri-plane extrapolation. *ACM Transactions on Graphics (ToG)*, 43(4):1–17, 2024.

Hongchi Xia, Xuan Li, Zhaoshuo Li, Qianli Ma, Jiashu Xu, Ming-Yu Liu, Yin Cui, Tsung-Yi Lin, Wei-Chiu Ma, Shenlong Wang, et al. Sage: Scalable agentic 3d scene generation for embodied ai. *arXiv preprint arXiv:2602.10116*, 2026.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 21469–21480, 2025.

Yandan Yang, Baoxiong Jia, Peiyuan Zhi, and Siyuan Huang. Physcene: Physically interactable 3d scene synthesis for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16262–16272, 2024a.

Yandan Yang, Baoxiong Jia, Shujie Zhang, and Siyuan Huang. Sceneweaver: All-in-one 3d scene synthesis with an extensible and self-reflective agent. *Advances in neural information processing systems*, 38:140319–140351, 2026a.

Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 206–215, 2018.

Yixuan Yang, Zhen Luo, Tongsheng Ding, Junru Lu, Mingqi Gao, Jinyu Yang, Victor Sanchez, and Feng Zheng. Llm-driven indoor scene layout generation via scaled human-aligned data synthesis and multi-stage preference optimization. *arXiv e-prints*, pp. arXiv–2506, 2025.

Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, et al. Holodeck: Language guided generation of 3d embodied ai environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16227–16237, 2024b.

Zhifei Yang, Guangyao Zhai, Keyang Lu, YuYang Yin, Chao Zhang, Zhen Xiao, Jieyi Long, Nassir Navab, and Yikai Wang. Flowscene: Style-consistent indoor scene generation with multimodal graph rectified flow. *arXiv preprint arXiv:2603.19598*, 2026b.

Lap-Fai Yu, Sai Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F Chan, and Stanley J Osher. Make it home: Automatic optimization of furniture arrangement. *ACM Trans. Graph.*, 30 (4):86, 2011.

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. *Advances in Neural Information Processing Systems*, 36:30026–30038, 2023.

Guangyao Zhai, Evin Pınar Örnek, Dave Zhenyu Chen, Ruotong Liao, Yan Di, Nassir Navab, Federico Tombari, and Benjamin Busam. Echoscene: Indoor scene generation via information echo over scene graph diffusion. In *European Conference on Computer Vision*, pp. 167–184. Springer, 2024.

Yiheng Zhang, Zhuojiang Cai, Mingdao Wang, Meitong Guo, Tianxiao Li, Li Lin, and Yuwang Wang. M3dlayout: A multi-source dataset of 3d indoor layouts and structured descriptions for 3d generation. *arXiv preprint arXiv:2509.23728*, 2025.

Mengqi Zhou, Yuxi Wang, Jun Hou, Shougao Zhang, Yiwei Li, Chuanchen Luo, Junran Peng, and Zhaoxiang Zhang. Scenex: Procedural controllable large-scale scene generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 10806–10814, 2025.

Xiaoyu Zhou, Xingjian Ran, Yajiao Xiong, Jinlin He, Zhiwei Lin, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Gala3d: Towards text-to-3d complex scene generation via layout-guided generative gaussian splatting. *arXiv preprint arXiv:2402.07207*, 2024.

A Data Processing

A.1 Voxelization

Given a 3D scene $R = \{M_1, M_2, ..., M_N, S\}$, composed of N object meshes and a fixed architectural structure S, we first discretize the continuous scene into a global instance-label grid $G_o \in \mathbb{N}^{D\times W\times H}$ with voxel size s_v, where each voxel stores an instance ID and 0 denotes free space. To prevent slender geometries from degrading or fracturing during this discrete sampling, we determine voxel occupancies using a surface-band threshold of $b = 0.5$ in voxel units. Subsequently, a 3D morphological hole-filling algorithm is applied to solidify the internal cavities of the objects. Through this process, each object M_i is converted to a corresponding dense volumetric occupancy grid O_i, which explicitly encodes its corresponding semantic category $\mathbf{y}_i$. Crucially, raw 3D assets often contain inherent interpenetrations, and the voxelization process itself can introduce quantization errors, occasionally causing a single voxel to be claimed by multiple instances. To resolve these spatial conflicts, we introduce a size-based priority heuristic: if a voxel's occupancy is contested, it is exclusively assigned to the object with the larger bounding volume. This rigorously enforces a mutually exclusive assignment constraint $O_i \cap O_j = \emptyset$, ensuring that any single voxel belongs to at most one specific instance, and is explicitly assigned a label $(\mathbf{y}_i, i)$ to denote its semantic category and instance identifier.

For each object O_i, we establish a local voxel grid G_l^i with a fixed resolution of K^3 centered at its origin, maintaining the identical physical voxel size s_v of the global space G_o. Considering that forcing the network to model arbitrary orientations imposes an immense learning burden and severely compromises fine details, we explicitly canonicalize each local volume by applying the inverse of the object's spatial rotation. To guarantee global contextual consistency, we populate the local grid G_l^i by directly sampling the semantic and instance tuples from the global grid G_o via nearest-neighbor interpolation. Consequently, each canonicalized local block encapsulates both the target object and its surrounding context information.

B Experiment Details

B.1 Dataset

3D-FRONT. To ensure a fair and rigorous comparison, we strictly adhere to the data filtering protocols established by DiffuScene (Tang et al., 2024) in the 3D-FRONT experiment. The curated dataset comprises 4,041 bedrooms, 900 dining rooms, and 813 living rooms for training and testing. For evaluation, we randomly sample text and floorplan conditions from the test scenes to generate 1,000 scenes per room category.

M3D-Shelf. M3DLayout is a high-quality indoor scene layout dataset that notably features the arrangement of delicate objects. To evaluate our model's capability in fine-grained spatial reasoning, we curate a customized subset denoted as M3D-Shelf. To build this subset, we exclusively extract the cabinet/shelf structures alongside their supported items. Before voxelization, we convert every shelf and contained object from the Blender/Z-up frame into our project Y-up coordinate system. For each asset, we fit the mesh to the local axis-aligned bounding box provided by M3DLayout, apply the layout `matrix_world`, and then center the resulting shelf scene. This curated subset comprises 490 shelf environments, with each containing 3 to 20 objects. We partition the data into training and testing splits using a standard 80%/20% ratio. For evaluation, we randomly sample known-shelf conditions from the test scenes to generate 200 scenes.

In M3D-Shelf, the shelf body is treated as a structural context rather than as a generated furniture target. Concretely, the shelf body is assigned the semantic name *largeshelf*, voxelized into the structure occupancy, and provided to both object-centric conditioning and downstream evaluation. The contained objects are the generation targets. Corresponding semantic categories and rendering colors are listed in Table 4.

B.2 Implementation Details

Anchor generation details. We pad each scene to $N_{\max}$ object slots, setting $N_{\max} = 12$ for Bedroom and $N_{\max} = 21$ for Livingroom, Diningroom, and M3D-Shelf. Unused slots carry the

empty category and zero geometric attributes. Before diffusion, positions and sizes are clipped to the bounds of the corresponding dataset and linearly mapped to $[-1, 1]$. We encode heading as $(\cos\theta, \sin\theta)$ and map the one-hot category indicators from $\{0, 1\}$ to $\{-1, 1\}$. We randomly permute occupied object slots during training. For M3D-Shelf completion, we keep the known shelf in the first slot and randomly permute the remaining objects.

The 1D U-Net uses a base feature width of 512, width multipliers $(1, 1, 1, 1)$, and 128-dimensional learned slot embeddings. We train it with $T = 1000$ diffusion steps and a linear noise schedule with β_t ranging from 10^{-4} to 0.02. The batch size is 128. On 3D-FRONT, we train the anchor models to convergence: 10^6 iterations for Bedroom and Diningroom, and 6×10^5 for Livingroom. At inference, we discard slots whose empty-category output is nonnegative and assign each remaining slot its highest-scoring object class. In scene completion, we preserve the known objects by replacing their slots at each reverse step with their observed attributes corrupted to the corresponding noise level, then restore their original attributes after sampling.

Voxel Generation details. In our experiment, we configure different voxel sizes for each dataset: $0.0375m$ for 3D-FRONT and a finer $0.01m$ for M3D-Shelf, considering the varying spatial scales of our target domains. Across both experiments, the local bounding grid G_l uses $K = 64$ voxels per axis, and CAD assets are voxelized at $K_A = 64$ per axis for retrieval. We use a Soft Chamfer tolerance of $\sigma = 1.5$ voxels. Retrieval filtering uses $\tau_{\text{ret}} = 0.75$; we report both filtered and unfiltered results for 3D-FRONT and M3D-Shelf. The style clustering strategy is applied only to 3D-FRONT, with an IoU threshold of $t_{\text{IoU}} = 0.50$ and a maximum axis-wise size ratio of $t_s = 1.10$. We analyze these choices in Appendix C.2. Regarding the spatial perturbation strategy, the maximum translation ranges for the random shift range $\mathbf{E}$ are empirically set to $[4, 0, 4]$ and $[6, 6, 6]$ voxels along the three spatial axes for the two datasets, respectively.

The VQ-VAE takes four input channels and has a base width of 16 channels. Two stride-2 downsampling stages map each 64^3 local block to a 16^3 latent grid. Its codebook contains 32,768 entries with an embedding dimension of one. The voxel diffusion model uses $T = 1000$ noise steps for training. At inference, we randomly order the target-object anchors, generate their occupancies sequentially, and sample each object with 100 DDIM steps ($\eta = 1.0$).

All experiments are conducted on two NVIDIA RTX 4090 GPUs. The VQ-VAE is trained for 300 epochs using the standard Adam optimizer (Kingma & Ba, 2014) with a constant learning rate of 2×10^{-4}. Subsequently, the diffusion models are optimized via AdamW (Loshchilov & Hutter, 2017) with a learning rate of 1×10^{-5} to ensure stable convergence and robust regularization. Specifically, to accommodate the distinct geometric complexities of the two tasks, the diffusion models undergo 450 training epochs for the 3D-FRONT dataset and 500 epochs for the fine-grained M3D-Shelf dataset.

B.3 Style Clustering

For instances with the same semantic label, we build a graph whose nodes are generated objects. Two instances are connected if their aligned voxel occupancy IoU exceeds t_{IoU} and their maximum axis-wise size ratio does not exceed t_s. Each connected component defines a style cluster. We retrieve an asset for each member, choose the member with the highest Soft Chamfer Score as the representative, and assign its top-ranked CAD asset to all objects in the cluster. We apply this procedure only to 3D-FRONT. Threshold settings and sensitivity results are reported in Appendix C.2.

B.4 Baseline

In the 3D-FRONT experiment, all comparative evaluations are conducted using the official open-source repositories of the respective baselines. For DiffuScene and PhyScene, we directly deploy their publicly released pre-trained weights. Because pre-trained weights were unavailable for Planner3D, we trained the model from scratch for 180 epochs across all room categories, strictly adhering to the configurations specified in their official documentation. Notably, since the official PhyScene repository only offers a checkpoint for the living rooms without the training code, our comparative analysis against this specific baseline is confined to the living room domain.

In the M3D-Shelf experiment, we follow the official setting for DiffuScene, training the autoencoder for 1000 epochs and the diffusion model for 35000 epochs. For SceneWeaver, we used five room-

Table 4: M3D-Shelf semantic categories and their corresponding evaluation colors. HEX values and visual swatches are provided for reference.

Semantic category	HEX & Visual Color
largeshelf	#7A8285
bookcolumn	#FABD2E
bookstack	#F29433
bottle	#00A1B8
bowl	#D17AC7
can	#8C94A3
cup	#3B7DF2
foodbag	#DB6E57
foodbox	#E6B24C
hardware	#333842
jar	#2EAD5C
natureshelftrinkets	#B28040
pan	#61616B
plate	#F2EBC7
pot	#A87052
wineglass	#6BCCEB

Table 5: Prompts and invocation protocol for the SceneWeaver open-shelf test.

Office: Design a compact office with one open-front shelf. The shelf must stay visibly open, with no doors, drawers, or closed cabinet panels. Make the shelf look naturally used by arranging appropriate small items inside its shelves.
Kitchen: Design a compact kitchen with one open-front shelf. The shelf must stay visibly open, with no doors, drawers, or closed cabinet panels. Make the shelf look naturally used by arranging appropriate small items inside its shelves.
Living room: Design a compact living room with one open-front shelf. The shelf must stay visibly open, with no doors, drawers, or closed cabinet panels. Make the shelf look naturally used by arranging appropriate small items inside its shelves.
Bedroom: Design a compact bedroom with one open-front shelf. The shelf must stay visibly open, with no doors, drawers, or closed cabinet panels. Make the shelf look naturally used by arranging appropriate small items inside its shelves.
Bathroom: Design a compact bathroom with one open-front shelf. The shelf must stay visibly open, with no doors, drawers, or closed cabinet panels. Make the shelf look naturally used by arranging appropriate small items inside its shelves.
Invocation: For each trial, SceneWeaver first initializes a compact room using the sampled user prompt and room type, then reloads the scene and adds only small objects inside the existing shelf, using the shelf as the parent relation target.

conditioned prompts, each specifying a compact indoor scene with a single open-front shelf. The prompts share the same structural constraint: the shelf must remain visibly open and must not be converted into a closed cabinet with doors, drawers, or solid panels (Table 5). During testing, we sample prompts from this set and pass the selected text to SceneWeaver to meet the user demand.

Unless otherwise specified, repeated SceneWeaver trials are produced by sampling from these five prompts with replacement under the benchmark seed. No prompt is manually adjusted for a particular generated scene.

B.5 Evaluation Metric Calculation Details

All geometric metrics are computed after the retrieval stage using the placed CAD meshes. For each evaluated object mesh, we first build a surface voxel grid with pitch s_e, then apply 3D hole filling to obtain a solid occupied voxel set V_i. We use $s_e = 0.02\,\mathrm{m}$ for 3D-FRONT and $s_e = 0.012\,\mathrm{m}$ for M3D-Shelf.

Fréchet Inception Distance (FID). Following previous works, we render top-down orthographic images at a resolution of 256×256, coloring instantiated objects according to their semantic category. Let (μ_r, Σ_r) and (μ_g, Σ_g) be the Gaussian statistics of Inception features from real and generated renderings. We compute

$$\mathrm{FID} = \|\mu_r - \mu_g\|_2^2 + \mathrm{Tr}\left(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\right). \tag{11}$$

Average Pairwise Intersection (I_p). For a scene with N objects, let $\mathcal{P} = \{(i,j) \mid 1 \le i < j \le N\}$ be all object pairs, and let V_i be the solid occupied voxel set of object i. The pairwise intersection severity is

$$s_{ij} = \max\left(\frac{|V_i \cap V_j|}{|V_i|}, \frac{|V_i \cap V_j|}{|V_j|}\right). \tag{12}$$

Then I_p is calculated as:

$$I_p = \frac{1}{|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} s_{ij}. \tag{13}$$

Global Overlap Ratio (OR). We measure scene-level overlap as

$$\mathrm{OR} = \frac{\sum_{(i,j)\in\mathcal{P}} |V_i \cap V_j|}{\sum_{i=1}^{N} |V_i| - \sum_{(i,j)\in\mathcal{P}} |V_i \cap V_j| + \epsilon}, \tag{14}$$

where $\epsilon > 0$ ensures numerical stability.

Category-weighted Diversity (CD). Let N_c^{ref} be the number of reference objects in category c, and let U_c^{ref} and U_c^{gen} be the sets of distinct CAD IDs in the reference and generated scenes. For the categories $\mathcal{C}$ present in the reference set, we compute

$$\mathrm{CD} = \sum_{c\in\mathcal{C}} \frac{N_c^{\mathrm{ref}}}{\sum_{k\in\mathcal{C}} N_k^{\mathrm{ref}}} \frac{|U_c^{\mathrm{gen}}|}{|U_c^{\mathrm{ref}}|}. \tag{15}$$

Repeated uses of the same CAD ID count once; a category absent from the generated scenes contributes zero.

M3D-Shelf spatial validity. Let Ω_S be the valid volume of the aligned shelf. An object is exempt from the out-of-bounds count if it protrudes only through the front opening at $z_{\max}$ and the projection of its center lies within the opening, allowing a one-voxel ($0.012\,\mathrm{m}$) margin. We denote this exception by b_i and compute the out-of-bounds rate as

$$R_o = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[V_i \setminus \Omega_S \neq \emptyset \ \wedge\ \neg b_i]. \tag{16}$$

The shelf-collision rate is

$$R_s = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[\mathcal{T}_S \cap B_i^- \neq \emptyset\right], \tag{17}$$

where $\mathcal{T}_S$ is the set of shelf mesh triangles and B_i^- is the object's local axis-aligned bounding box (AABB), inset by 0.01 normalized units ($0.012\,\mathrm{m}$) on each face before the intersection test.

The floating rate is

$$R_f = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[h_i^{\mathrm{sup}} \text{ is undefined } \vee\ h_i^{\mathrm{bottom}} - h_i^{\mathrm{sup}} > \delta\right], \tag{18}$$

where h_i^{bottom} is the object bottom height and h_i^{sup} is the support height under the object center. We set $\delta = 3s_e = 0.036\,\mathrm{m}$ for M3D-Shelf. OR uses the same object-object formulas above, computed only over contained objects.

LLM assessment. We render one front-facing semantic-color view for each M3D-Shelf scene (following Table 4) and provide the image, semantic color mapping, and object metadata to the evaluator. The LLM assigns integer scores $g_{n,k} \in \{0, \ldots, 10\}$ for realism, layout, and completion.

The evaluation prompt is shown in Table 6, for which we made minimum modification from SceneWeaver to adapt it to our shelf-level tasks.

Table 6: Prompt and scoring protocol for the M3D-Shelf LLM assessment.

Task: You are given a single {`llm_view`} render of an extracted open-front shelf/cabinet. Evaluate how well the objects are placed inside or on the shelf. This is a cabinet-level evaluation, not a whole-room evaluation; ignore the rest of the room.
Rendering: The image uses semantic colors rather than realistic textures. Ignore texture, lighting, and image style. Use geometry, object categories, positions, sizes, and visible placement quality.
Strict scoring: For each criterion, assign an integer score from 0 to 10 and provide a brief justification. Score 10 means the result fully meets the criterion, score 5 indicates noticeable flaws, and score 0 indicates that the aspect is absent, wrong, or contradicts the request. Critical issues such as floating objects, objects outside the shelf, hidden objects behind closed panels, severe collisions, implausible categories, or missing shelf contents should significantly lower the relevant score.
Realism: Assess whether the shelf/cabinet placement is physically plausible. Good results show a visibly open shelf/cabinet with plausible shelf objects, believable object sizes, and a naturally used arrangement. Bad results include closed or visually nonsensical shelves, implausible categories, severe scale errors, floating objects, or deeply embedded objects.
Layout: Assess whether objects are geometrically arranged correctly inside or on the shelf. Good results have objects supported by shelf surfaces, inside the shelf footprint, upright or reasonably oriented, not severely colliding, not outside the cabinet, and not blocking the front opening. Bad results contain floating objects, board penetration, heavy collisions, outside placement, buried objects, wrong scale/orientation, or objects invisible from the front.
Completion: Assess whether the result is sufficiently populated as an open-front shelf/cabinet. Good results contain an appropriate number and variety of small objects and look intentionally arranged. Bad results are empty, nearly empty, sparse, irrelevant, or do not meaningfully populate the shelf. Do not require whole-room furniture or room-level functionality.
Metadata: The prompt includes four JSON blocks: target shelf/cabinet metadata, contained/on-shelf object metadata, object semantic color mapping, and camera/render metadata. The metadata and image must be used together; objects should be counted carefully and checked for visibility, support, scale, placement, and severe collision.
Coordinate conventions: The metadata is in a right-handed local coordinate system with meter units. The Y axis is up. The shelf front/opening in the rendered frame is {`opening_direction`} ({`opening_face_name`}). Object local location is the center, size is given as `[x_half, y_half, z_half]`, and AABBs are local-frame bounds for quick checking.
Return format: Return only a JSON object with three keys: `realism`, `layout`, and `completion`. Each key contains an integer `grade` and a short `comment`. If the image and metadata conflict, briefly mention the conflict in the relevant comment.

C Other Results

C.1 Effect of Free-Space Write-Back

We examine free-space write-back on 100 generated scenes per room type in 3D-FRONT with text conditions. For this comparison, CAD retrieval uses the raw decoded object occupancies before write-back excludes voxels already occupied by earlier objects. Relative to the standard write-back setting, the post-retrieval OR increases by 0.048 in the 10^3-scaled units used for 3D-FRONT. The small change indicates that free-space write-back contributes only a limited part of the overlap reduction in this experiment.

C.2 Sensitivity to Style-Clustering Thresholds

We evaluate the sensitivity of style clustering on the Livingroom and Diningroom splits of 3D-FRONT by varying the IoU threshold t_{IoU} and the maximum axis-wise size ratio t_s around our default setting $(0.50, 1.10)$. Each sweep varies one threshold while keeping the other at its default and all remaining settings fixed. A size ratio limit of 1.10 corresponds to a 10% size tolerance. Table 7 reports nearby settings.

Style clustering assigns a representative CAD asset to multiple objects, so the retrieved meshes may depart from their individual voxel occupancies and cause more collisions. This trend appears when

Table 7: Sensitivity to style-clustering thresholds around the default setting. The best value for each metric within each sweep is bold. Lower values are better for all metrics. I_p and OR are scaled by 10^3.

Value	Livingroom			Diningroom		
	$FID \downarrow$	$I_p \downarrow$	$OR \downarrow$	$FID \downarrow$	$I_p \downarrow$	$OR \downarrow$
IoU threshold t_{IoU} (fixed $t_s = 1.10$)						
0.40	39.074	1.528	4.844	37.859	**1.317**	4.175
0.45	39.071	1.529	4.849	37.893	**1.317**	4.171
0.50	**39.065**	1.523	4.792	37.876	1.318	4.168
0.55	39.072	1.518	4.778	**37.842**	1.321	4.163
0.60	39.102	**1.516**	**4.770**	37.868	**1.317**	**4.145**
Size ratio limit t_s (fixed $t_{\text{IoU}} = 0.50$)						
1.00	39.168	**1.505**	**4.737**	37.931	**1.302**	**4.110**
1.05	39.052	1.516	4.771	37.903	1.317	4.142
1.10	39.065	1.523	4.792	37.876	1.318	4.168
1.15	38.991	1.533	4.819	37.318	1.334	4.227
1.20	**38.990**	1.535	4.815	**37.253**	1.340	4.247

the size-ratio limit is relaxed: increasing t_s from 1.10 to 1.20 improves FID in both room types but raises I_p and OR. In the IoU sweep, FID varies by less than 0.06 and is lowest at intermediate thresholds. In the end, we choose $(t_{\text{IoU}}, t_s) = (0.50, 1.10)$ to balance scene distribution quality against physical collisions.

C.3 Sensitivity to the Object-Filtering Threshold

We evaluate the object-filtering threshold τ_{ret} on the Livingroom and Diningroom splits of 3D-FRONT. Table 8 reports nearby thresholds together with the fraction of filtered objects.

Table 8: Sensitivity to the retrieval filtering threshold τ_{ret} on 3D-FRONT. I_p and OR are scaled by 10^3. Filtered denotes the percentage of generated objects discarded.

Room	τ_{ret}	$FID \downarrow$	$I_p \downarrow$	$OR \downarrow$	Filtered (%)
Livingroom	0.65	39.289	1.759	6.853	1.91
	0.70	39.245	1.644	5.897	3.42
	0.75	39.065	1.523	4.792	6.79
	0.80	39.163	1.406	3.111	15.28
Diningroom	0.65	38.102	1.471	5.743	2.29
	0.70	37.958	1.343	4.774	3.88
	0.75	37.876	1.318	4.168	7.81
	0.80	38.375	1.265	2.758	17.54

As τ_{ret} increases from 0.65 to 0.75, FID decreases in both rooms, as do I_p and OR despite the reduction in object count. This joint improvement suggests that moderate filtering removes implausible voxel occupancies. At 0.80, the filtered fractions more than double to 15.28% and 17.54%, respectively. Collisions continue to decrease, but FID rises in both rooms, indicating that more aggressive filtering harms scene distribution quality. We therefore choose 0.75 in our experiments.

C.4 Memory Consumption

We report our memory values in the training stage in Table 9.

Table 9: Training resource usage of the M3D-Shelf VQ-VAE and object-centric diffusion models (measured on two NVIDIA RTX 4090 GPUs).

Model (Setting)	Batch	Peak Mem.	Time/Step
VQ-VAE *(64^3 local blocks)*	4	8.1 GiB	0.12 s
Diffusion *(16^3 latents + online)*	4	12.8 GiB	0.45 s

C.5 GENERATION EFFICIENCY

The runtime of our pipeline is reported in Table 10. All VoxScene timings use one NVIDIA RTX 4090 GPU and one CPU thread, with the pipeline stages run sequentially. For comparison, Table 11 reports the average wall-clock time over 200 successful SceneWeaver open-shelf runs. In SceneWeaver, initialization creates the compact room and open shelf; the timed shelf-population stage reloads the scene and places objects inside the shelf.

For VoxScene, the measured end-to-end runtime is **67.80** seconds per scene, with **8.09** objects on average. SceneWeaver takes **612.14** seconds per scene to populate a shelf with **5.49** retained objects on average. Its solve-and-cleanup stage takes **226.06** seconds, including **123.66** seconds for pre-solve asset population and **63.07** seconds for the shelf-placement solver core. On this benchmark, VoxScene populates shelves with more objects on average in about one-ninth of the time taken by the agent-based SceneWeaver pipeline.

Table 10: Average runtime of VoxScene on 200 M3D-Shelf scenes containing 1,617 objects.

Pipeline Stage	Time / Scene	Time / Object
Anchor Generation	24.40 s	3.018 s
Voxel Diffusion Generation	23.15 s	2.863 s
CAD Retrieval	20.25 s	2.505 s
End-to-end	**67.80 s**	**8.386 s**

Note: Per-object averages are amortized over the 1,617 generated objects.

Table 11: Average SceneWeaver runtime for shelf population, amortized over the 5.49 objects retained per completed scene.

Pipeline Stage	Time / Scene	Time / Object
GPT object planning	60.39 s	11.00 s
Load initialized scene	48.69 s	8.87 s
Graph action	10.48 s	1.91 s
Solve and cleanup	226.06 s	41.18 s
Pre-solve asset population	123.66 s	22.52 s
Shelf-placement solver core	63.07 s	11.49 s
Record scene	253.99 s	46.26 s
Other	12.53 s	2.28 s
Shelf-population total	**612.14 s**	**111.50 s**

Note: The indented timings are included in solve and cleanup.

C.6 Qualitative Ablations

Figures 6 and 7 show qualitative results for style clustering and anchor shifting, respectively.

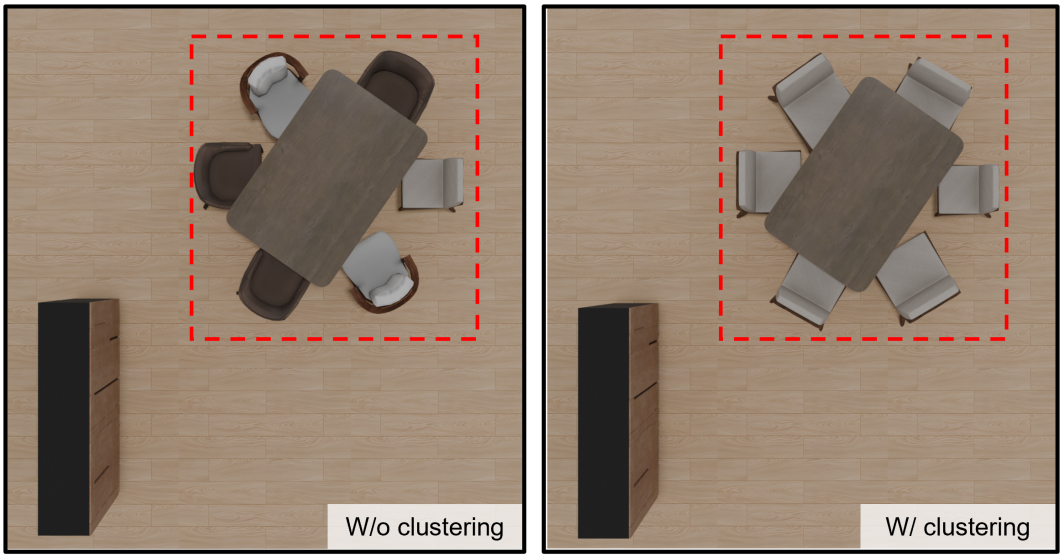

Figure 6: **Effect of style clustering**.

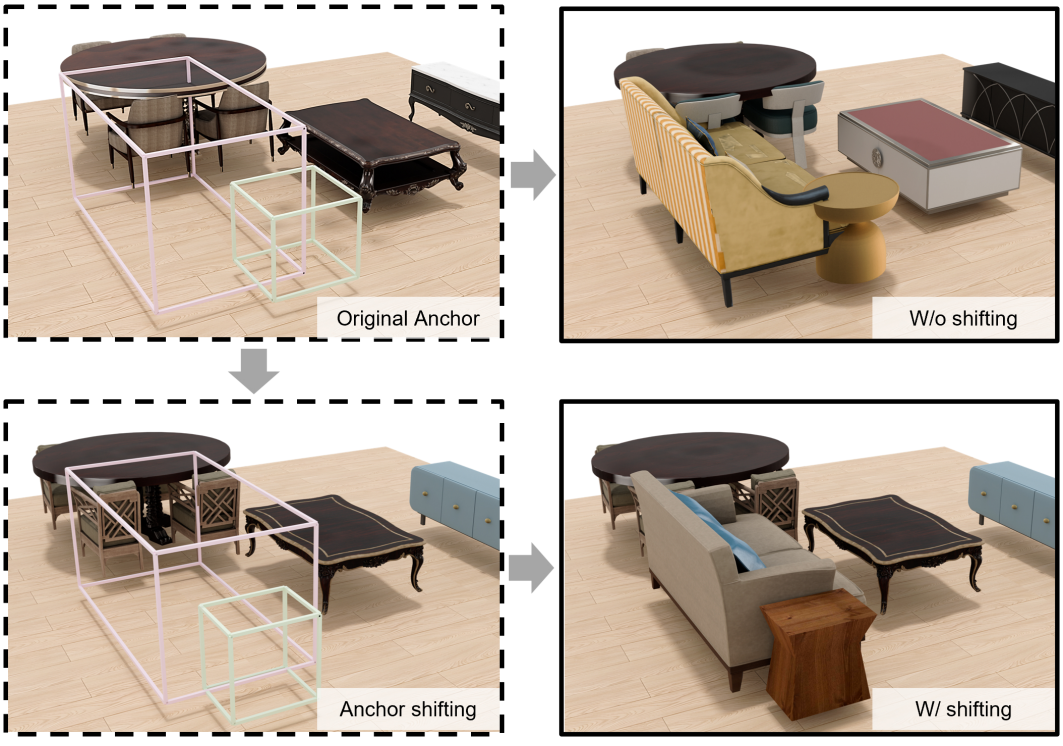

Figure 7: **Effect of anchor shifting.** The left column shows anchor conditions; the right column shows generated scenes without (top) and with (bottom) anchor shifting.

D APPLICATION

D.1 USER EDITING

VoxScene supports user-directed scene editing through box-level anchors. As shown in Fig. 8, the room examples demonstrate object addition, removal, and size changes, while the shelf examples show displacement and replacement within a confined structure.

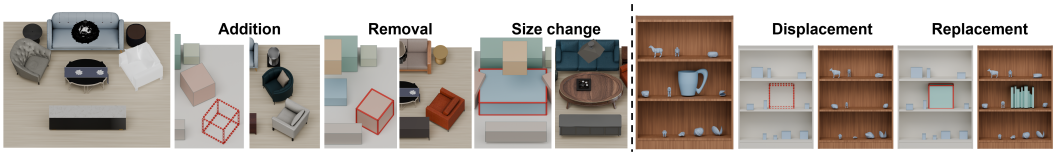

Figure 8: **User-directed scene editing.** Room-level examples (left) show object addition, removal, and size changes; shelf-level examples (right) show displacement and replacement. Red boxes highlight the user edits.

D.2 DOWNSTREAM ROBOTIC DATA AUGMENTATION

We evaluate generated layouts as training data for cluttered-shelf retrieval with a Franka arm. All methods use the same M3DLayout shelf geometries. Source-only retains ground-truth arrangements, while DiffuScene and VoxScene use their generated layouts.

For each method, we fine-tune ACT from the same initialization using $N{=}800$ CuRobo demonstrations, then evaluate retrieval success and collision rate on the same dense OOD shelves. A retrieval succeeds when the policy grasps and extracts the target without collision. We measure the collision rate from ACT rollouts on the shared OOD shelves, rather than from object overlap in the generated training scenes. Disturbance describes the shared OOD shelf geometry and therefore has the same value for all methods. Figure 9 shows qualitative examples; Table 12 reports quantitative results.

Table 12: Cluttered-shelf target retrieval under matched demonstration budget ($N{=}800$). Higher is better for yield/success; lower is better for collision. Disturbance is a shared-OOD geometry statistic (identical across methods).

Training data	Yield ↑	Success ↑	Collision ↓	Disturbance ↓
Source only	38.4%	72.3%	5.3%	0.006
DiffuScene	56.5%	74.0%	**2.7%**	0.006
VoxScene	**66.5%**	**78.8%**	**2.7%**	0.006

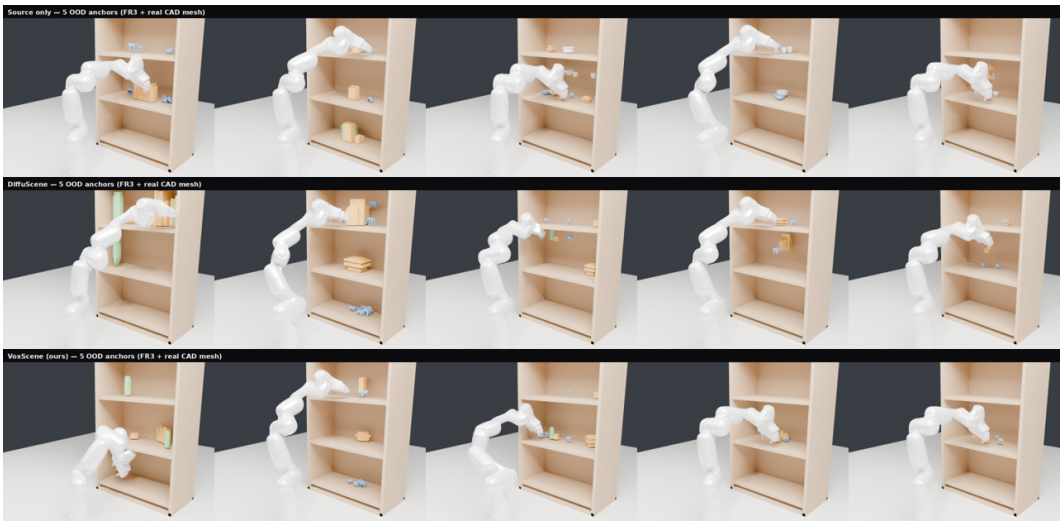

Figure 9: Shared OOD shelf anchors under three training-data sources (Source, DiffuScene, VoxScene), rendered with Franka and real CAD meshes. Rows share the evaluation setting; columns are different anchors.

E SUPPLEMENTARY VISUALIZATIONS

E.1 ADDITIONAL QUALITATIVE COMPARISONS

Figures 10 and 11 provide additional qualitative comparisons on 3D-FRONT and M3D-Shelf, respectively.

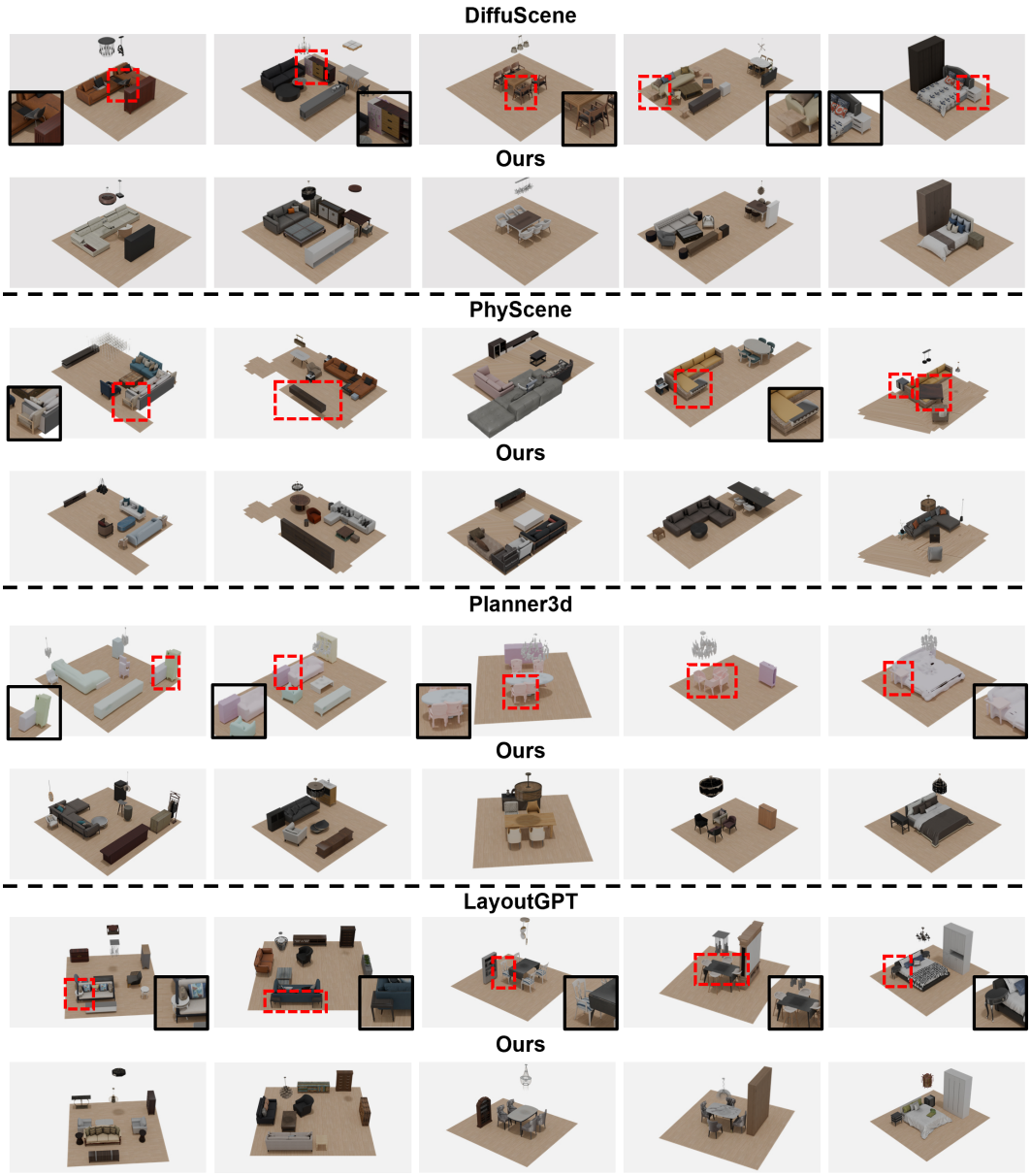

Figure 10: **Additional qualitative comparisons on 3D-FRONT.** Each pair of rows compares a baseline with VoxScene. Red dashed boxes highlight geometric conflicts, with close-up views shown in black-bordered insets.



Figure 11: **Additional qualitative comparisons on M3D-Shelf.** The rows show SceneWeaver, DiffuScene, and VoxScene, respectively, across five shelf examples.

E.2 Additional Qualitative Results

Figure 12 shows additional generated voxels and retrieved assets. Further room-level and shelf-level results are shown in Figs. 13 and 14.

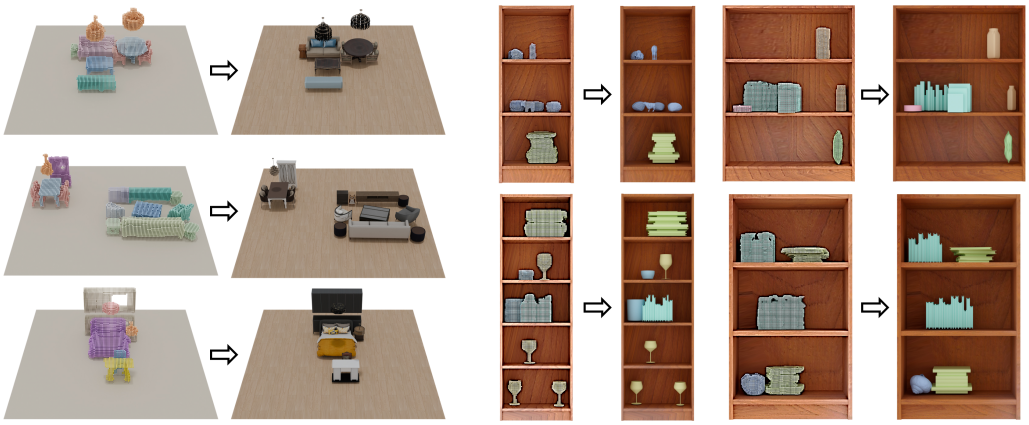

Figure 12: **More results of our voxel generation and retrieved assets.**

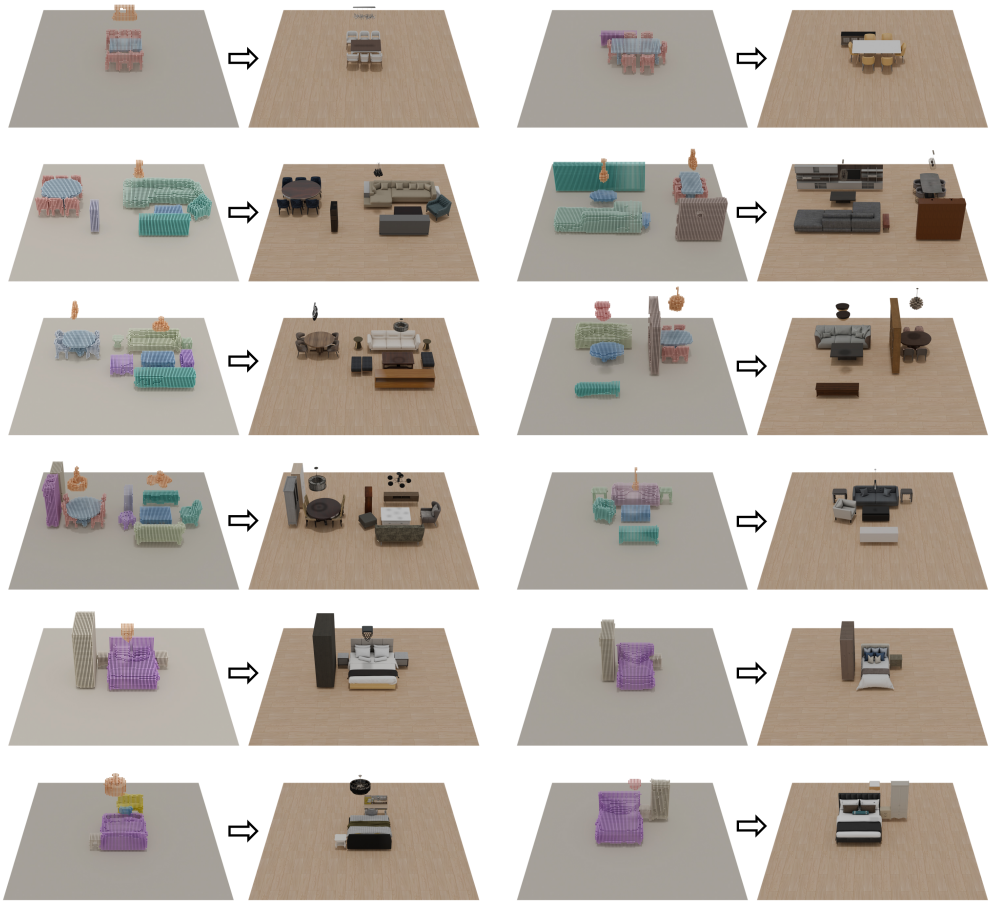

Figure 13: More results on 3D-FRONT.

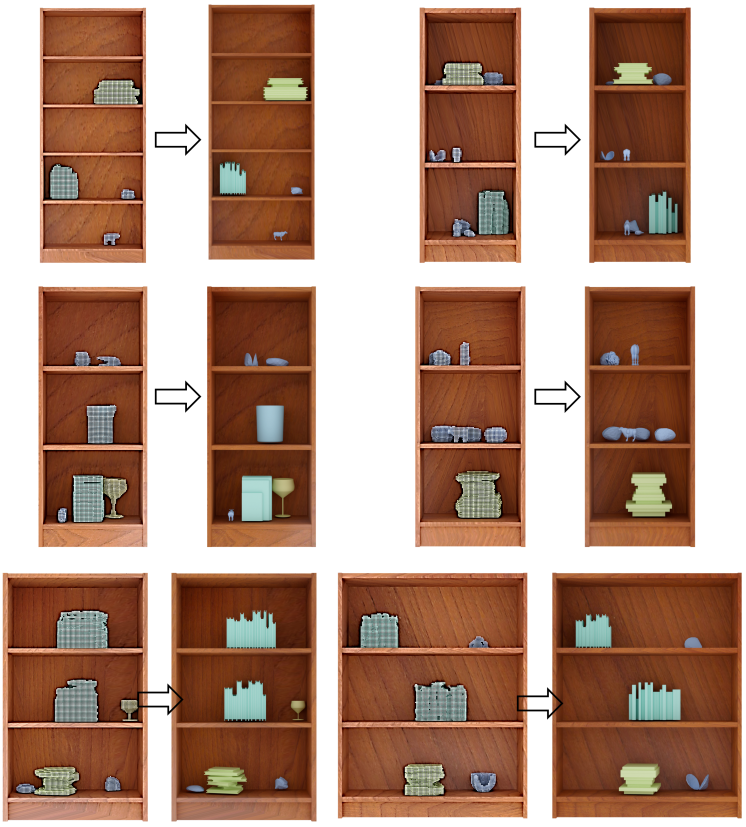

Figure 14: More results on M3D-Shelf.