PROSCENE: PROGRESSIVE 3D INDOOR SCENE GENERATION WITH SPATIAL-VALIDITY GUIDANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Indoor scene synthesis methods commonly generate a layout where objects are represented by oriented 3D bounding boxes, followed by mesh retrieval or generation based on box parameters to complete the scene. This often leads to output with spatial violations, including collisions and out-of-bounds errors. Because boxes are too coarse to represent complex shapes or interleaving objects, true violations cannot be reliably distinguished from valid configurations where bounding boxes overlap but the meshes do not, limiting accurate guidance toward spatial validity. We therefore propose PROSCENE, a retrieval-free multi-stage model that progressively generates 3D scenes through a box→cuboid→mesh hierarchy. PROSCENE first produces a coarse box layout in stage 1, then employs a coupled layout-cuboid diffusion model in stage 2 to jointly refine the layout and generate object cuboids, with bidirectional feature exchange allowing both representations to co-evolve during denoising. We further apply cross-scale spatial-validity guidance, using boxes for efficient spatial error assessment and cuboids for more accurate evaluation. The resulting violation energies are used to guide the output toward spatial plausibility. Finally, stage 3 conditions an object generator on the predicted cuboids and semantic information to synthesize detailed meshes. This hierarchy provides an explicit structural bridge between coarse scene arrangement and detailed object geometry, enabling spatial-violation-aware scene generation.

1 INTRODUCTION

Realistic 3D indoor scenes are important for a wide range of applications, including VR/AR, embodied AI, interior design, and game development Liu et al. (2025); Duan et al. (2022); Zhao et al. (2021). However, creating high-quality scenes manually requires substantial time and modeling expertise, motivating research on automatic scene synthesis. To support downstream applications, a generated 3D scene should contain separate object meshes that enable object-level interactions. Existing methods typically address this problem by first predicting a scene layout that determines the objects and their placements, followed by instantiating the geometry of each object. The layout prediction stage can be formulated as an optimization problem with hand-crafted design constraints Merrell et al. (2011), or learned directly from 3D data using deep learning Paschalidou et al. (2021); Tang et al. (2024); Hu et al. (2026); Feng et al. (2023; 2025), typically followed by object retrieval to construct the complete scene.

However, generating spatially valid scenes remains challenging for existing methods. Specifically, synthesized scenes can still contain spatial violations, where objects collide with each other or extend beyond the room boundary, limiting their usability in simulation and downstream applications such as VR/AR. These violations mainly stem from the discrepancy between the object representation used during layout generation and final shapes placed in the scene. In particular, most methods represent objects as oriented 3D boxes and generate the scene layout by predicting their box parameters, while the final scene is composed of detailed meshes retrieved based on those predictions.

Since a single 3D box provides only a coarse approximation of object geometry, box layouts cannot accurately represent scenes with complex shapes, such as L-shaped sofas, or interleaved objects, such as dining tables and chairs. In these cases, boxes from different objects may overlap even when their corresponding meshes do not, leading to false-positive collisions, as illustrated in Fig. 1 (left). An alternative is to use explicit voxel grids to represent object geometry more accurately Mao et al.

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.

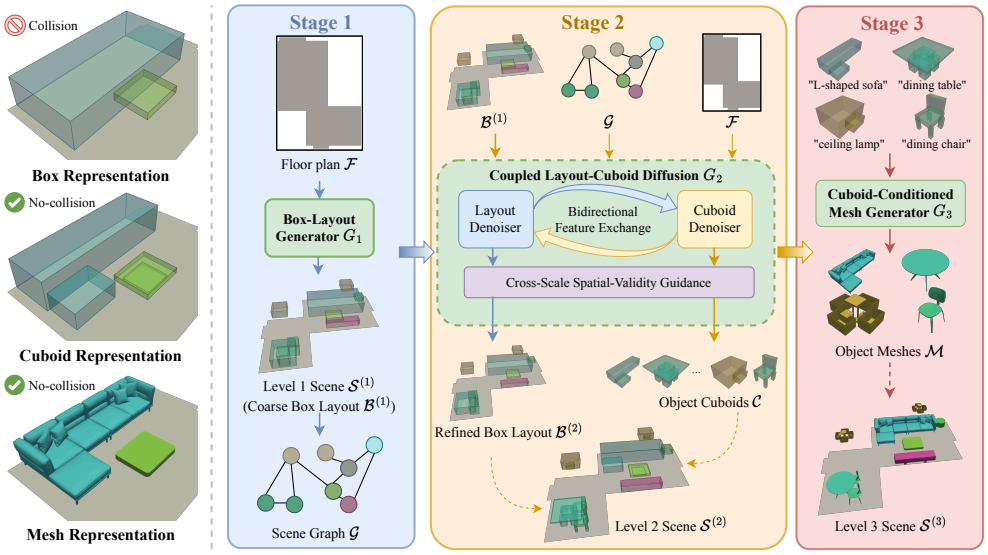

Figure 1: Left: 3D boxes struggle to capture complex shapes or interleaved objects, leading to false-positive collisions, whereas cuboids provide a more accurate yet compact representation. Right: ProScene uses cuboids as a structural bridge between coarse boxes and detailed meshes, progressively generating scenes from box to cuboid to mesh with cross-scale spatial-validity guidance.

(2026), but their memory and computation costs grow cubically with spatial resolution. Representing an object as a set of cuboids Feng et al. (2025) provides a useful middle ground, capturing more complex geometric structure than a single box while remaining compact and efficient. As illustrated in Fig. 1 (left), cuboids can better approximate object parts and represent interleaved objects, making them a suitable intermediate representation between coarse boxes and detailed meshes.

Moreover, existing spatial-validity guidance based on box overlap and out-of-bounds errors Yang et al. (2024) can provide inaccurate training signals due to these false-positive violations. This makes it difficult for the model to distinguish true geometric conflicts from valid configurations with overlapping boxes. In such cases, retrieving meshes based only on box attributes, as commonly done in existing methods Paschalidou et al. (2021); Tang et al. (2024); Hu et al. (2026), provides limited geometric constraints for selecting object shapes that fit together without collision. Furthermore, retrieval requires database access during generation and restricts the available geometry to that database, making it harder to find suitable meshes for valid overlapping-box configurations.

To address these limitations, we propose ProScene, a multi-stage method for retrieval-free 3D scene synthesis. ProScene connects box and mesh representations through cuboids and progressively generates scenes at increasing granularity levels. Specifically, ProScene first predicts a box-based scene layout. Next, a coupled layout-cuboid diffusion module simultaneously refines the layout and denoises a cuboid set for each object. The layout and cuboid denoisers are synchronized with bidirectional feature exchange, allowing the evolving layout and cuboid structures to guide each other at every denoising step. We further introduce cross-scale spatial-validity guidance to enforce spatial constraints at this stage. In particular, we first detect potential violations at the box level and then assess them more accurately using the cuboid structures, guiding both the layout and cuboids away from spatial errors. Finally, the predicted cuboid set of each object conditions a TRELLIS-based mesh generator Xiang et al. (2025). This leverages TRELLIS's ability to generate high-quality meshes while guiding the generated geometry to follow the scene-aware cuboid structure, reducing spatial violations in the final scene. In summary, we introduce:

- **Progressive scene generation.** We represent objects in a scene using a three-level box→cuboid→mesh hierarchy and generate these representations in that order of increasing granularity. We use cuboids as an intermediate structural bridge between boxes and meshes, enabling retrieval-free generation with greater control over object geometry.
- **Coupled layout-cuboid diffusion.** We jointly evolve layout corrections and cuboid predictions through bidirectional feature exchange, allowing emerging object structures to refine the layout while the updated scene context guides cuboid denoising.

- **Cross-scale spatial-validity guidance.** We assess spatial violations at the box scale and refine the assessment at the cuboid level, producing violation energies that guide the model toward spatially valid outputs during both training and inference.

2 Related Work

In this section, we summarize previous works that lay the foundation for our proposed method. For more detailed comparisons, please refer to the appendix.

Indoor Scene Generation. 3D indoor scene generation has received growing attention with recent advances in deep learning and the availability of large-scale 3D datasets Fu et al. (2021a); Chang et al. (2017); Zheng et al. (2020). Previous works synthesize scenes unconditionally Tang et al. (2024) or condition generation on scene graphs Zhai et al. (2024; 2023), multi-view images Chen et al. (2025); Li et al. (2026); Fang et al. (2026), or natural language Lin & Mu (2024); Hu et al. (2024). Other methods Paschalidou et al. (2021); Feng et al. (2025); Hu et al. (2026) use a 2D floor plan to constrain the generated layout to room geometry, making them practical for applications such as gaming and virtual reality. However, these approaches rely on mesh retrieval, which requires access to an object database at inference time and limits scene diversity. In contrast, our ProScene progressively generates objects from 3D boxes to cuboids and finally meshes, avoiding retrieval while preserving geometric fidelity and providing control over spatial validity.

Shape Abstraction in Scene Generation. The shape abstraction used in a scene generator strongly affects its occupancy accuracy and the fidelity of the final objects. Most indoor layout methods Paschalidou et al. (2021); Tang et al. (2024); Yang et al. (2024) represent each object using an oriented 3D box. This compact parameterization is effective for global spatial planning, but provides only a coarse approximation of object geometry. Some approaches augment boxes with learned implicit shape codes Tang et al. (2024), which improve geometric expressiveness but lack explicit surface or boundary information for spatial-violation control. VoxScene Mao et al. (2026) instead introduces voxel occupancy, providing finer spatial information at the cost of cubic memory and computation. Meanwhile, CASAGPT Feng et al. (2025) decomposes objects into cuboid primitives, providing a compact representation that captures useful geometry for object retrieval. Motivated by this trade-off, we use cuboids as an intermediate abstraction between boxes and meshes. Our ProScene combines the efficiency of boxes, the explicit structure of cuboids, and the high fidelity of meshes, progressively generating scenes in this order of granularity.

Spatial Validity in 3D Scene Generation. Spatial plausibility has been studied in scene-conditioned pose estimation Hassan et al. (2019) and motion generation Wang et al. (2021), but incorporating such constraints directly into 3D scene generation remains challenging. PhyScene Yang et al. (2024) introduces collision avoidance and room-boundary objectives defined over object level boxes as guidance for diffusion-based scene generation. While effective, box-based error estimates remain coarse, and inaccurate. CASAGPT Feng et al. (2025) uses cuboid primitives to identify object intersections, but relies on collision-aware filtering and rejection rather than directly guiding the model to avoid violations during generation. In our proposed ProScene, we introduce cross-scale spatial-validity guidance that steers the model toward spatially valid scenes. We first perform a coarse assessment at the box scale to identify potential violations, which are then re-evaluated using more accurate cuboid geometry. Our guidance is applied during both training and inference, further improving the plausibility of generated scenes.

3 Methodology

Overview. ProScene is a three-stage scene synthesis method that progressively moves from a coarse box layout to compact object structures and finally to detailed object meshes, as illustrated in Fig. 1 (right). We represent a scene $\mathcal{S}$ as a set of N objects at three levels of granularity: $\mathcal{S}^{(1)} = \mathcal{B}^{(1)}$, $\mathcal{S}^{(2)} = (\mathcal{B}^{(2)}, \mathcal{C})$, $\mathcal{S}^{(3)} = (\mathcal{B}^{(2)}, \mathcal{M})$, where $\mathcal{B}^{(1)} = \{b_i^{(1)}\}_{i=1}^N$, $\mathcal{B}^{(2)} = \{b_i^{(2)}\}_{i=1}^N$, $\mathcal{C} = \{\mathcal{C}_i\}_{i=1}^N$ and $\mathcal{M} = \{m_i\}_{i=1}^N$ denote the sets of coarse and refined boxes, object cuboids, and final meshes.

Given a floor plan $\mathcal{F}$, stage 1 (Sec. 3.1) generates the coarse layout $\mathcal{B}^{(1)}$ where each object is represented as an oriented 3D box, which is used to derive a scene graph $\mathcal{G}$ to capture scene context and spatial violations. Then stage 2 (Sec. 3.2) jointly refines the box layout and generates the cuboid set

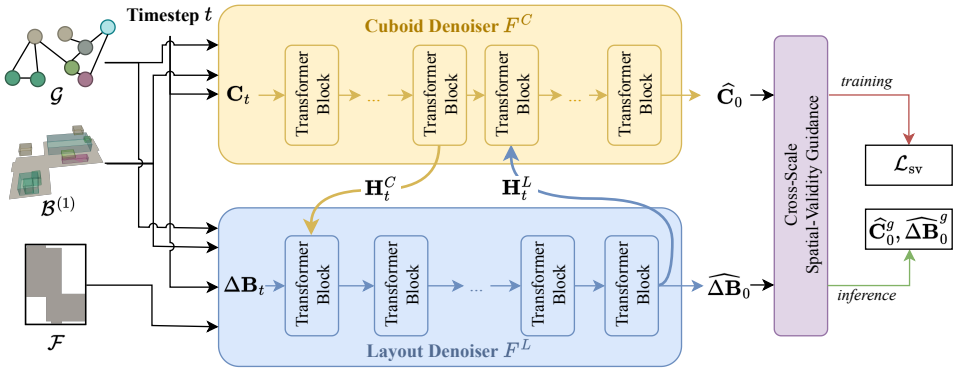

Figure 2: Coupled layout-cuboid diffusion jointly evolves layout corrections and object cuboids. At each denoising step, early cuboid blocks provide object structural features to the layout denoiser, while the resulting scene-aware layout feature conditions the remaining cuboid blocks. Cross-scale spatial-validity guidance is applied during both training and inference to reduce spatial violations.

for each object with a coupled layout-cuboid diffusion module that allows the two representations to exchange information at every denoising step. With cross-scale spatial-validity guidance, the refined layout $\mathcal{B}^{(2)}$ and object cuboids $\mathcal{C}$ are adjusted to avoid collision and out-of-bounds errors, improving the scene's spatial validity. Finally, conditioned on each object's cuboids and semantic information, stage 3 (Sec. 3.3) generates a detailed mesh and places it into the boxes, resulting in a fully generated scene $\mathcal{S}^{(3)}$ without relying on an object database and retrieval.

3.1 Stage 1: Box Layout Generation

Object i is represented as a box $b_i^{(l)} = (y_i, \mathbf{p}_i^{(l)}, \mathbf{s}_i^{(l)}, \theta_i^{(l)})$ at level $l \in \{1, 2\}$, where $y_i \in \{1, \ldots, N_{cls}\}$, $\mathbf{p}_i^{(l)}, \mathbf{s}_i^{(l)} \in \mathbb{R}^3$, and $\theta_i^{(l)} \in \mathbb{R}$ denote its semantic category, location, size, and rotation angle around the vertical axis. Given a floor plan $\mathcal{F}$, a scene generator G_1 produces the coarse box layout with N objects $\mathcal{B}^{(1)} = \{b_i^{(1)}\}_{i=1}^N$ in the first stage.

This box representation is commonly used in previous works Tang et al. (2024); Hu et al. (2026); Paschalidou et al. (2021) because it provides a simple and efficient scene parameterization. However, a single oriented box is too coarse to accurately reflect collisions of the underlying object geometry, making spatial constraints difficult to enforce at the box level. As a result, $\mathcal{B}^{(1)}$ may contain spatial violations that can propagate to the final scene when the predicted box attributes are used directly for mesh retrieval or generation. Instead, we use this coarse layout as input to the next stage of ProScene, where it is refined while a finer object representation is generated.

We also construct a scene graph $\mathcal{G} = G_{\text{graph}}(\mathcal{B}^{(1)}, \mathcal{F})$ from the box layout and floor plan to provide object relationships and scene geometry for the next stage. The i-th node in $\mathcal{G}$ corresponds to box $b_i^{(1)}$ and its edges are determined by $\mathcal{N}^{\text{graph}}(i) = \text{kNN}(i) \cup \left\{k : \text{Intersect}\left(b_i^{(1)}, b_k^{(1)}\right)\right\}$, which contains its k nearest neighbors and any other objects that intersect with it.

The initial i-th node feature is constructed from its box attributes and two coarse spatial-violation cues: accumulated collision severity with other objects and boundary violation with respect to the floor plan $\mathcal{F}$. For each edge, the features encode the relative displacement, Euclidean distance, yaw, and directional relations between the two objects. The scene graph $\mathcal{G}$ is then encoded by a graph neural network with edge-aware multi-head attention layers. The edge features modulate both the attention weights and messages exchanged between neighboring nodes, producing a contextual feature $\mathbf{g}_i$ for each object to condition stage 2 generation. This provides the network with both coarse box information and spatial-violation cues, facilitating layout correction in the next stage.

3.2 Stage 2: Layout-Cuboid Diffusion

In addition to the box representation, each object is further decomposed into a set of K cuboids $\mathcal{C}_i = \{c_{ij}\}_{j=1}^K$, where each cuboid c_{ij} is parameterized by its center, size, and existence state.

The goal of stage 2 is to jointly refine the coarse layout $\mathcal{B}^{(1)}$ by predicting a layout correction that transforms it into $\mathcal{B}^{(2)}$, while simultaneously generating the cuboid sets for all objects in the scene.

As illustrated in Fig. 2, we introduce a coupled layout-cuboid diffusion model G_2, consisting of a layout denoiser and a cuboid denoiser that operate at the same denoising timestep t and share the same noise schedule $\bar{\alpha}_t$. Let $\Delta\mathbf{B}_0$ denote the clean layout correction and $\mathbf{C}_0$ denote the clean cuboid state for all objects, their forward diffusion processes are defined as:

$$\Delta\mathbf{B}_t = \alpha_t\Delta\mathbf{B}_0 + \sigma_t\boldsymbol{\epsilon}^L, \quad \mathbf{C}_t = \alpha_t\mathbf{C}_0 + \sigma_t\boldsymbol{\epsilon}^C, \quad \text{where } \alpha_t = \sqrt{\bar{\alpha}_t}, \quad \sigma_t = \sqrt{1-\bar{\alpha}_t}, \quad \boldsymbol{\epsilon}^L, \boldsymbol{\epsilon}^C \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{1}$$

Given the noisy states $(\Delta\mathbf{B}_t, \mathbf{C}_t)$, the two denoisers jointly estimate their corresponding clean states while exchanging intermediate features at every denoising step. Specifically, the early cuboid blocks produce object-level structural features that condition the layout denoiser, while the resulting cuboid-aware layout feature is fed back to the remaining cuboid blocks. This bidirectional interaction allows the box layout and cuboid structure to co-evolve throughout the reverse diffusion process. We next describe the two denoisers and their interaction in detail.

3.2.1 Layout Denoiser

The layout denoiser corrects the coarse layout $\mathcal{B}^{(1)}$ into the refined $\mathcal{B}^{(2)}$ to improve the scene spatial plausibility. Rather than predicting $\mathcal{B}^{(2)}$ directly, we let the denoiser predict the clean layout correction $\Delta\mathbf{B}_0 = [\Delta\mathbf{b}_i]_{i=1}^N \in \mathbb{R}^{N\times 8}$, where the correction of object i is:

$$\Delta\mathbf{b}_i = [\mathbf{p}_i^{(2)} - \mathbf{p}_i^{(1)}, \log(\mathbf{s}_i^{(2)}/\mathbf{s}_i^{(1)}), \sin(\theta_i^{(2)} - \theta_i^{(1)}), \cos(\theta_i^{(2)} - \theta_i^{(1)})]. \tag{2}$$

Given the noisy $\Delta\mathbf{B}_t$, the layout denoiser is trained to estimate the corresponding clean correction $\Delta\mathbf{B}_0$, which is used to refine $\mathcal{B}^{(1)}$ to $\mathcal{B}^{(2)}$ by inverting Eq. 2 (details provided in the appendix).

We adopt velocity prediction for the layout denoiser. At step t, the velocity target is defined as $\mathbf{v}_t^L = \alpha_t\boldsymbol{\epsilon}^L - \sigma_t\Delta\mathbf{B}_0$, where α_t, σ_t, and $\boldsymbol{\epsilon}^L$ are defined in Eq. 1. Given the predicted velocity $\widehat{\mathbf{v}}_t^L$, the clean layout correction and corresponding noise estimate can be computed as:

$$\widehat{\Delta\mathbf{B}}_0 = \alpha_t\Delta\mathbf{B}_t - \sigma_t\widehat{\mathbf{v}}_t^L, \qquad \widehat{\boldsymbol{\epsilon}}_t^L = \sigma_t\Delta\mathbf{B}_t + \alpha_t\widehat{\mathbf{v}}_t^L. \tag{3}$$

The layout denoiser is trained with the following objective:

$$\mathcal{L}_{\text{diff}}^L = \mathbb{E}\left[w_L(t)\left\|\widehat{\mathbf{v}}_t^L - \mathbf{v}_t^L\right\|_F^2\right], \tag{4}$$

where $w_L(t) = \min(\text{SNR}_t, \gamma_L)/(\text{SNR}_t + 1)$ is the Min-SNR weight Hang et al. (2023), with $\text{SNR}_t = \bar{\alpha}_t/(1-\bar{\alpha}_t)$ and γ_L denoting the clipping threshold.

3.2.2 Cuboid Denoiser

Each object i is represented by K cuboid slots with clean state $\mathbf{C}_{i,0} \in \mathbb{R}^{K\times 7}$. The j-th cuboid slot is parameterized as $\mathbf{c}_{ij} = [\mathbf{x}_{ij}, \mathbf{d}_{ij}, e_{ij}] \in \mathbb{R}^7$, where $\mathbf{x}_{ij}, \mathbf{d}_{ij} \in \mathbb{R}^3$ denote the object-local center and size, and $e_{ij} \in \{-1, +1\}$ denotes the existence state.

Given the noisy cuboid state $\mathbf{C}_{i,t}$ of object i, the cuboid denoiser predicts the corresponding Gaussian noise $\widehat{\boldsymbol{\epsilon}}_{i,t}^C$. The clean cuboid state is then recovered as $\widehat{\mathbf{C}}_{i,0} = (\mathbf{C}_{i,t} - \sigma_t\widehat{\boldsymbol{\epsilon}}_{i,t}^C)/\alpha_t$.

The same cuboid denoiser is applied to every object and is trained with the following objective:

$$\mathcal{L}_{\text{diff}}^C = \mathbb{E}\left[w_C(t)\left\|\widehat{\boldsymbol{\epsilon}}_t^C - \boldsymbol{\epsilon}^C\right\|_F^2\right], \tag{5}$$

with $\widehat{\boldsymbol{\epsilon}}_t^C = [\widehat{\boldsymbol{\epsilon}}_{i,t}^C]_{i=1}^N$ and $w_C(t)$ denoting the corresponding Min-SNR weight Hang et al. (2023).

3.2.3 Bidirectional Layout-Cuboid Evolution

A key design of our layout-cuboid diffusion is the bidirectional interaction between the two denoisers, allowing the layout and cuboid representations to co-evolve at each denoising step while being aware of the current state of each other.

At timestep t, the noisy cuboid state $\mathbf{C}_{i,t}$ of object i is passed through the first M Transformer blocks of the cuboid denoiser. The resulting cuboid hidden features are average-pooled and transformed into an object-level structural feature:

$$\widetilde{\mathbf{H}}^C_{i,t} = F^C_{1:M}\left(\mathbf{C}_{i,t}; t, y_i, b_i^{(1)}, \mathbf{g}_i\right), \qquad \mathbf{h}^C_{i,t} = \psi^C\left(\mathrm{AvgPool}\left(\widetilde{\mathbf{H}}^C_{i,t}\right)\right). \tag{6}$$

The feature $\mathbf{h}^C_{i,t}$ summarizes the current cuboid structure of object i and is used as an additional condition for the layout denoiser at the same timestep. Let $\mathbf{H}^C_t = [\mathbf{h}^C_{i,t}]_{i=1}^N$ denote the structural features of all objects. The layout denoiser then computes:

$$\left(\widehat{\mathbf{v}}^L_t, \mathbf{H}^L_t\right) = F^L\left(\Delta\mathbf{B}_t; t, \mathcal{B}^{(1)}, \mathcal{F}, \mathcal{G}, \mathbf{H}^C_t\right), \tag{7}$$

where $\mathbf{H}^L_t = [\mathbf{h}^L_{i,t}]_{i=1}^N$ stacks the layout features of all objects. The feature $\mathbf{h}^L_{i,t}$ for object i is then used as an additional condition for the remaining Transformer blocks of the cuboid denoiser:

$$\left(\widehat{\boldsymbol{\epsilon}}^C_{i,t}, \mathbf{H}^C_{i,t}\right) = F^C_{M+1:}\left(\widetilde{\mathbf{H}}^C_{i,t}; t, y_i, b_i^{(1)}, \mathbf{g}_i, \mathbf{h}^L_{i,t}\right). \tag{8}$$

Thus, at each step, the emerging object structure informs the scene layout, while the resulting cuboid-aware layout feature immediately guides the remaining cuboid denoising. Combined with cross-scale spatial-validity guidance, this reciprocal interaction enables the box and cuboid states to co-evolve toward spatial validity.

3.2.4 Cross-Scale Spatial-Validity Guidance

The explicit box-cuboid hierarchy enables spatial-validity guidance at complementary scales. Given the refined layout $\mathcal{B}^{(2)}$ and cuboid structures $\mathcal{C}$, we first detect coarse collisions, out-of-bounds violations, and implausible object sizes at the box scale. For objects with detected violations, we then evaluate the finer cuboid geometry to obtain more accurate collision and out-of-bounds energies.

Box-Scale Spatial-Violation Guidance Energies. For each refined box $b_i^{(2)} = (y_i, \mathbf{p}_i^{(2)}, \mathbf{s}_i^{(2)}, \theta_i^{(2)})$, we define the box-level collision energy as $E_i^{\mathrm{box,col}} = \sum_{k\neq i} \mathrm{Pen}_{\mathrm{SAT}}\left(b_i^{(2)}, b_k^{(2)}\right)$, where $\mathrm{Pen}_{\mathrm{SAT}}(\cdot,\cdot)$ denotes the penetration depth between two yaw-oriented boxes computed using the separating-axis theorem (SAT). Specifically, the boxes are projected onto the candidate separating axes, and the minimum positive overlap determines the penetration depth.

We additionally define the box-level out-of-bounds and size-prior energies as:

$$E_i^{\mathrm{box,oob}} = \frac{1}{|\mathcal{Q}_i|}\sum_{\mathbf{q}\in\mathcal{Q}_i}[d_{\mathcal{F}}(\mathbf{q})]_+, \qquad E_i^{\mathrm{size}} = \left\|[\boldsymbol{\ell}_{y_i} - \log\mathbf{s}_i^{(2)}]_+\right\|_1 + \left\|[\log\mathbf{s}_i^{(2)} - \mathbf{u}_{y_i}]_+\right\|_1, \tag{9}$$

where $\mathcal{Q}_i$ contains sampled points on the floor-plane projection of the oriented box, $d_{\mathcal{F}}(\mathbf{q})$ denotes the signed distance to the valid room region, with positive values indicating out-of-bounds points, and $[x]_+ = \max(x, 0)$; the vectors $\boldsymbol{\ell}_{y_i}$ and $\mathbf{u}_{y_i}$ denote category-specific lower and upper log-size bounds estimated from the training set. More details are provided in the appendix.

Cuboid-Scale Spatial-Violation Guidance Energies. Box-level energies provide efficient violation detection but may overestimate their severity because bounding boxes only coarsely approximate object geometry. We therefore refine violation assessments using the predicted cuboids. Specifically, for cuboid $c_{ij} = (\mathbf{x}_{ij}, \mathbf{d}_{ij}, e_{ij})$ of object i, the collision and out-of-bounds energies are:

$$E_{ij}^{\mathrm{cub,col}} = \sum_{k\in\mathcal{N}_i^{\mathrm{col}}}\sum_{\ell=1}^{K}\pi_{k\ell}\,\mathrm{Pen}_{\mathrm{SAT}}\left(c_{ij}, c_{k\ell}\right), \qquad E_{ij}^{\mathrm{cub,oob}} = \mathbf{1}\left[E_i^{\mathrm{box,oob}} > 0\right]\frac{1}{|\mathcal{Q}_{ij}|}\sum_{\mathbf{q}\in\mathcal{Q}_{ij}}[d_{\mathcal{F}}(\mathbf{q})]_+. \tag{10}$$

where $\mathcal{N}_i^{\mathrm{col}}$ contains objects whose boxes have positive penetration with $b_i^{(2)}$, $\mathcal{Q}_{ij}$ contains sampled points on the floor-plane projection of cuboid c_{ij}, and $\pi_{kl} = \mathrm{Sigmoid}(\widehat{e}_{kl})$ is the existence weight of cuboid c_{kl}. The energy terms of the K cuboids are then summed using their existence weights to obtain $E_i^{\mathrm{cub,col}}$ and $E_i^{\mathrm{cub,oob}}$ for object i.

Spatial-Violation-Guided Training. During training, we regularize the predicted clean states using the energies defined above. Specifically, we define the spatial-violation regularization loss as:

$$\mathcal{L}_{\text{sv}} = \lambda_{\text{size}}\mathcal{L}_{\text{size}} + \lambda_{\text{box}}\mathcal{L}_{\text{box}} + \lambda_{\text{cub}}\mathcal{L}_{\text{cub}}, \tag{11}$$

where λ terms control the relative contributions of the spatial-validity objectives:

$$\mathcal{L}_{\text{size}} = \frac{1}{N}\sum E_i^{\text{size}}, \mathcal{L}_{\text{box}} = \frac{1}{N}\sum[(E_i^{\text{box,col}})^2 + (E_i^{\text{box,oob}})^2], \mathcal{L}_{\text{cub}} = \frac{1}{N}\sum[(E_i^{\text{cub,col}})^2 + (E_i^{\text{cub,oob}})^2]. \tag{12}$$

Because clean-state predictions are unreliable at high noise levels, spatial-violation regularization is applied only for timesteps below a threshold T_{sv}. The total training objective in stage 2 is:

$$\mathcal{L} = \mathcal{L}_{\text{diff}}^{L} + \mathcal{L}_{\text{diff}}^{C} + \mathbf{1}[t \le T_{\text{sv}}]\,\mathcal{L}_{\text{sv}}. \tag{13}$$

Inference Late-Sampling Guidance. Spatial-violation-guided training encourages the denoiser to produce spatially plausible predictions, but small violations may remain during inference. We therefore use the violation energies as guidance during the late reverse-diffusion steps. We define the scene spatial-violation energy as:

$$E_{\text{sv}}^{g} = \lambda_{\text{size}}^{g}E^{\text{size}} + \lambda_{\text{box}}^{g}(E^{\text{box,col}} + E^{\text{box,oob}}) + \lambda_{\text{cub}}^{g}(E^{\text{cub,col}} + E^{\text{cub,oob}}) \tag{14}$$

where λ^g terms control the component weights, and each scene-level energy is obtained by averaging its corresponding object-level energies. E_{sv}^{g} is then used to obtain guided clean-state estimates:

$$\widehat{\Delta\mathbf{B}}_0^{g} = \widehat{\Delta\mathbf{B}}_0 - \eta_B\nabla_{\widehat{\Delta\mathbf{B}}_0}E_{\text{sv}}^{g}, \qquad \widehat{\mathbf{C}}_0^{g} = \widehat{\mathbf{C}}_0 - \eta_C\nabla_{\widehat{\mathbf{C}}_0}E_{\text{sv}}^{g}, \tag{15}$$

where η_B and η_C are the layout and cuboid guidance step sizes. From the final clean states, we construct the level 2 scene representation $\mathcal{S}^{(2)} = (\mathcal{B}^{(2)}, \mathcal{C})$, which is passed to stage 3 for detailed mesh generation.

3.3 Stage 3: Mesh Generation

In this stage, we generate a detailed mesh for each object using a generator G_3 based on TRELLIS Xiang et al. (2025). Specifically, for the i-th object, its cuboid structure $\mathcal{C}_i$ is voxelized and encoded by a 3D network F_{voxel} while its semantic category y_i is encoded using CLIP Radford et al. (2021) as : $\mathbf{f}_i^{3D} = F_{\text{voxel}}(\mathcal{C}_i)$ and $\mathbf{f}_i^{\text{CLIP}} = \text{CLIP}(y_i)$. These features jointly condition G_3 to generate the object mesh from noise:

$$m_i = G_3(\boldsymbol{\epsilon}_i^{M}; \mathbf{f}_i^{3D}, \mathbf{f}_i^{\text{CLIP}}). \tag{16}$$

This is applied independently to all objects to obtain $\mathcal{M} = \{m_i\}_{i=1}^{N}$, which are placed according to $\mathcal{B}^{(2)}$ to form the complete scene without relying on an object database or retrieval.

Moreover, cuboids condition provides direct structural control over the generated geometry. Since the cuboids have already been constrained by spatial regularization, they guide the final meshes to better preserve the intended spatial configuration and reduce violations in the completed scene.

4 Experiments

Datasets. We conduct experiments on the large-scale 3D indoor scene dataset 3D-FRONT Fu et al. (2021a). 3D-FRONT is a synthetic dataset containing 6,813 houses and 14,629 rooms, which are furnished with high-quality 3D objects from the 3D-FUTURE dataset Fu et al. (2021b). Following previous works Paschalidou et al. (2021); Tang et al. (2024); Hu et al. (2026), we consider three room categories for training and evaluation: 4,041 bedrooms, 900 dining rooms, and 813 living rooms. For each room type, we use 80% of the scenes for training and the remaining 20% for testing.

Evaluation Metrics. We evaluate the generated scenes in terms of visual realism and spatial validity. For visual realism, we use Fréchet Inception Distance (FID) Heusel et al. (2017) to measure the distributional similarity between generated and ground-truth scenes. For spatial validity, we adopt the Voxelization-Based Loss (VBL) introduced in Bucher & Armeni (2025). VBL is the sum of the Out-of-Bounds Loss (OOB) and Mesh Boundary Loss (MBL), which measure the volume (m^3) of objects outside the room boundary and the overlapping volume between object pairs.

Methods	Bedroom				Livingroom				Diningroom			
	FID	*MBL*	*OOB*	*VBL*	*FID*	*MBL*	*OOB*	*VBL*	*FID*	*MBL*	*OOB*	*VBL*
ATISS	**87.30**	0.076	0.232	0.308	46.86	0.126	0.303	0.429	46.38	0.086	0.260	0.346
ProScene (w/ ATISS)	87.53	**0.058**	**0.166**	**0.224**	**40.58**	**0.076**	**0.139**	**0.215**	**41.88**	**0.065**	**0.111**	**0.176**
DiffuScene	87.94	0.041	0.184	0.225	**57.72**	0.057	0.688	0.745	**50.75**	0.047	0.465	0.512
ProScene (w/ DiffuScene)	**87.18**	**0.033**	**0.136**	**0.169**	58.01	**0.034**	**0.622**	**0.656**	51.48	**0.035**	**0.411**	**0.446**
MiDiffusion	87.79	0.068	0.111	0.179	42.62	0.081	0.165	0.246	**42.66**	0.087	0.115	0.202
ProScene (w/ MiDiffusion)	**87.74**	**0.059**	**0.062**	**0.121**	**42.30**	**0.055**	**0.084**	**0.139**	43.47	**0.067**	**0.058**	**0.125**

Table 1: Quantitative comparison on 3D-FRONT dataset. Lower values are better for all metrics.

Baselines. To evaluate the compatibility of our method with different box layout generators, we integrate three representative scene-layout models into ProScene: ATISS Paschalidou et al. (2021), DiffuScene Tang et al. (2024), and MiDiffusion Hu et al. (2026). Each baseline is used as the stage 1 generator to produce the coarse box layout, which is subsequently refined and completed by the later stages of ProScene. Additional experimental details are provided in the appendix.

4.1 3D Indoor Scene Generation

We report the results of the baselines and our proposed method in Tab. 1. For the baselines, scenes are synthesized by retrieving 3D-FUTURE object meshes for the predicted box layouts following their original settings, while ProScene generates object meshes directly with our proposed three-stage process. We can see that ProScene consistently reduces both collision and out-of-bounds violations across all room types and three box-layout generators. This demonstrates the advantage of our hierarchical box-cuboid-mesh representation over using boxes alone for plausible scene generation. Notably, despite fully generating the object meshes, ProScene achieves comparable or better realism than the retrieval-based baselines, as reflected by the FID scores. This suggests that the reduction in spatial violations comes from improved scene structures rather than simply generating degraded object geometry to avoid collisions.

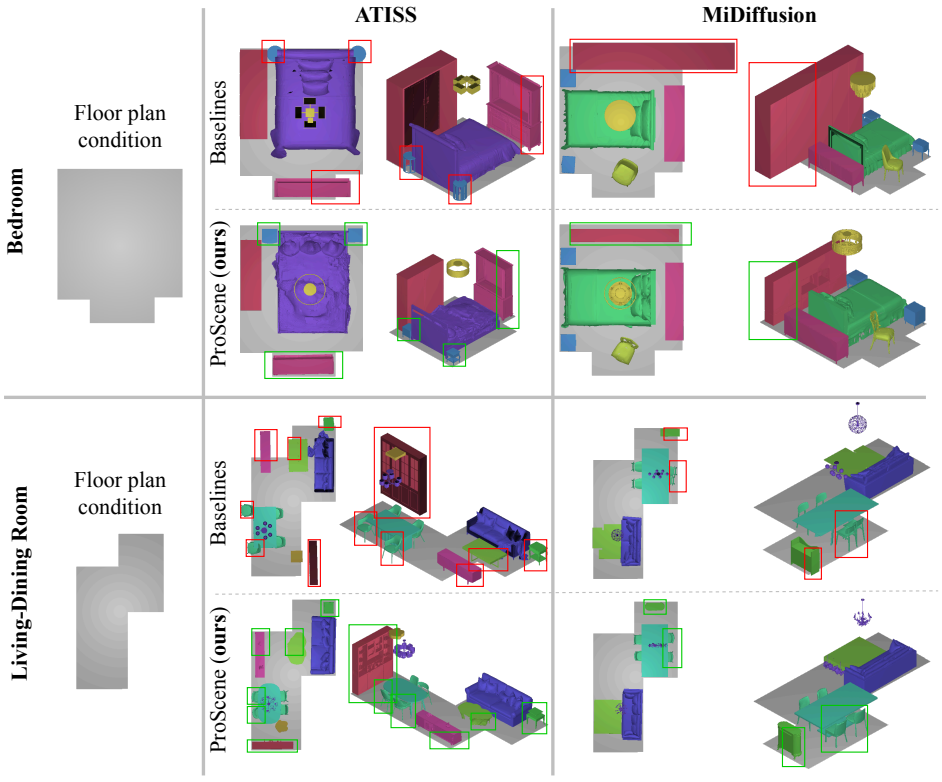

Figure 3: Qualitative comparison of ATISS, MiDiffusion, and ProScene. For each room type, the first row shows baseline outputs from two views, and the second shows the corresponding scenes by our model. Red boxes highlight collision and out-of-bounds violations of the baselines, while green boxes indicate regions corrected by ProScene. The input floor plan is shown on the left.

Qualitative Comparison. Fig. 3 compares scenes generated by PROSCENE with two box-layout generators, ATISS and MiDiffusion. Both baselines are susceptible to spatial violations, as highlighted by the red boxes. With cross-scale spatial-validity guidance, PROSCENE refines object positions, sizes, and orientations while generating cuboid structures that better satisfy the spatial constraints, ultimately correcting these violations as highlighted by the green boxes. Additional examples are provided in the appendix.

4.2 ABLATION STUDY

	FID	*MBL*	*OOB*	*VBL*
Retrieve w/ boxes	87.30	0.076	0.232	0.308
Retrieve w/ cuboids	**87.18**	**0.068**	**0.213**	**0.281**
Generate w/ boxes	88.35	0.073	0.220	0.293
Generate w/ cuboids	**87.53**	**0.058**	**0.166**	**0.224**

Table 2: Ablation of the cuboid representation.

	FID	*MBL*	*OOB*	*VBL*
No interaction	87.91	0.076	0.200	0.276
Cuboid $\rightarrow$ layout	87.92	0.073	0.170	0.243
Layout $\rightarrow$ cuboid	87.65	0.070	0.189	0.259
Cuboid $\leftrightarrow$ layout	**87.53**	**0.058**	**0.166**	**0.224**

Table 3: Ablation of layout-cuboid interaction.

We conduct ablation studies to evaluate the contributions of different components in our design. All experiments use ATISS as the box-layout generator in stage 1 and report results on bedrooms.

Hierarchical Representation. We study the effect of using cuboids as the intermediate representation between boxes and meshes. We compare mesh retrieval using box sizes or cuboid structures, and mesh generation conditioned on boxes or cuboids. As shown in Tab. 2, using cuboids consistently reduces collision and out-of-bounds errors for both retrieval and generation, demonstrating the benefit of the additional structural information provided by the cuboid representation.

Bidirectional Layout-Cuboid Evolution. Tab. 3 compares different layout-cuboid interactions. Allowing information exchange between the two consistently improves spatial plausibility. Specifically, cuboid-to-layout interaction helps reduce out-of-bounds violations, while layout-to-cuboid interaction reduces collision errors. The bidirectional exchange in PROSCENE combines both benefits and achieves the best performance.

	Size prior	Box guidance	Cuboid guidance	Late sampling	MBL	OOB	VBL
(i)	-	-	-	-	0.070	0.186	0.256
(ii)	✓	-	-	-	0.069	0.180	0.249
(iii)	✓	✓	-	-	0.066	0.176	0.242
(iv)	✓	-	✓	-	0.062	0.174	0.236
(v)	✓	✓	✓	-	0.060	0.170	0.230
(vi)	✓	✓	✓	✓	**0.058**	**0.166**	**0.224**

Table 4: Ablation of cross-scale spatial-validity guidance.

Cross-Scale Spatial-Validity Guidance. We evaluate the contribution of each spatial-validity guidance component in Tab. 4. Adding the size prior in setting (ii) gives a small improvement. Both collision and out-of-bounds energies on box and cuboid levels further reduce spatial violations, with cuboid-level guidance providing a larger gain due to its more accurate error detection, as shown in (iii) and (iv). Combining all violation energies further improves the results in (v), while our design with late-sampling guidance achieves the lowest overall violations.

5 CONCLUSION

In this paper, we present PROSCENE, a retrieval-free multi-stage method for indoor scene generation based on a box→cuboid→mesh representation hierarchy. Our coupled layout-cuboid diffusion jointly refines the scene layout and generates object cuboid structures through bidirectional feature exchange. PROSCENE further introduces cross-scale spatial-validity guidance that combines efficient box-level violation evaluation with finer cuboid-level assessment during training and late-stage sampling. The generated cuboids then condition detailed mesh synthesis, enabling complete scene generation without object retrieval. Experiments on the 3D-FRONT dataset show that PROSCENE improves spatial validity while maintaining competitive scene realism.

AI USE STATEMENT

Generative AI tools were used in this work to assist with developing conceptual frameworks, proposing and refining hypotheses, providing feedback on research methodology and experimental design, and supporting method implementation. They were also used to suggest experimental parameters, create and edit software code, brainstorm research ideas, identify research gaps, search for information and relevant literature, suggest paper structure and titles, draft portions of the manuscript, and improve the readability of the writing.

Generative AI tools were not used to formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in writing proofs, translate content, clean or reformat datasets, perform qualitative or thematic data analysis, or interpret experimental results. The generation of synthetic datasets was not applicable to this work.

All AI-assisted outputs were reviewed and verified by the authors. Research ideas suggested by generative AI were critically assessed before use; AI-generated or AI-modified code was inspected, tested, and verified for correctness by the first author; suggested literature was independently checked for relevance and accuracy; and AI-assisted outlines and manuscript text were reviewed and revised by the authors. The authors take full responsibility for the final content of this work, including all text, claims, code, and other artifacts produced with the assistance of generative AI.

REFERENCES

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

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017.

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.

Dasith de Silva Edirimuni and Ajmal Saeed Mian. Class-partitioned vq-vae and latent flow matching for point cloud scene generation. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence and Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence and Sixteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'26/IAAI'26/EAAI'26. AAAI Press, 2026. ISBN 978-1-57735-906-7. doi: 10.1609/aaai.v40i5.37352. URL https://doi.org/10.1609/aaai.v40i5.37352.

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.

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. In *2026 International Conference on 3D Vision (3DV)*, pp. 1392–1402. IEEE, 2026.

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, 2021a.

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 (12):3313–3337, 2021b.

Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via min-snr weighting strategy. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7407–7417. IEEE, 2023.

Mohamed Hassan, Vasileios Choutas, Dimitrios Tzionas, and Michael J Black. Resolving 3d human pose ambiguities with 3d scene constraints. 2019.

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.

Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7875–7886. IEEE, 2023.

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.

Gwanhyeong Koo, Hyunsu Kim, Youngji Kim, Taejae Lee, Siwoo Lim, Sunjae Yoon, Suyong Yeon, and Chang D. Yoo. Inspace: Structure-aware 3d indoor scene generation from a single 360deg $image$. *arXiv preprint arXiv:2607.03990*, 2026.

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. In *International Conference on Learning Representations*, volume 2024, pp. 25687–25718, 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*, 2025.

Haotian Mao, Yuhan Huang, Jiatao Lin, Yang Zhao, Hui Wang, Yiheng Zhang, Yuwang Wang, Chenliang Zhou, Yan Zhang, Fangcheng Zhong, et al. Voxscene: Anchor-conditioned voxel diffusion for indoor scene arrangement. *arXiv preprint arXiv:2605.17102*, 2026.

Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. Interactive furniture layout using interior design guidelines. *ACM Transactions on Graphics*, 2011.

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.

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

Nicholas Pfaff, Thomas Cohn, Sergey Zakharov, Rick Cory, and Russ Tedrake. Scenesmith: Agentic generation of simulation-ready indoor scenes. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=WwS8CTpUA6.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

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.

Jiashun Wang, Huazhe Xu, Jingwei Xu, Sifei Liu, and Xiaolong Wang. Synthesizing long-term 3d human motion and interaction in 3d scenes. 2021.

Zhennan Wu, Yang Li, Han Yan, Taizhang Shang, Weixuan Sun, Senbo Wang, Ruikai Cui, Weizhe Liu, Hiroyuki Sato, Hongdong Li, and Pan Ji. BlockFusion: Expandable 3D scene generation using latent tri-plane extrapolation. *ACM Transactions on Graphics (TOG)*, 2024.

Hongchi Xia, Xuan Li, Zhaoshuo Li, Qianli Ma, Jiashu Xu, Ming-Yu Liu, Yin Cui, Tsung-Yi Lin, Wei-Chiu Ma, Shenlong Wang, Shuran Song, and Fangyin Wei. Sage: Scalable agentic 3d scene generation for embodied ai. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21469–21480. IEEE, 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, 2024.

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, 2026.

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.

Yizhou Zhao, Kaixiang Lin, Zhiwei Jia, Qiaozi Gao, Govind Thattai, Jesse Thomason, and Gaurav S Sukhatme. Luminous: Indoor scene generation for embodied ai challenges. *arXiv preprint arXiv:2111.05527*, 2021.

Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling. In *European Conference on Computer Vision*, pp. 519–535. Springer, 2020.

A APPENDIX

A.1 COMPARISON WITH RELATED WORKS

Other 3D Indoor Scene Generation. A separate line of work generates a scene as a single continuous representation, such as a NeRF, a 3D Gaussian splat, or one fused mesh, rather than a set of separable object meshes Wu et al. (2024); Li et al. (2026); Höllein et al. (2023). Because the output does not distinguish individual objects, these methods do not support object-level interaction, which is required for downstream applications such as embodied simulation and VR/AR object manipulation. In contrast, PROSCENE generates a separate mesh per object, enabling such interactions.

Existing methods also differ in their input conditions. Scene-graph-conditioned methods Zhai et al. (2023; 2024) generate a scene from a graph but impose no explicit room-boundary constraint. Multi-view-image-conditioned methods Chen et al. (2025); Fang et al. (2026) lift 2D diffusion priors into 3D, typically producing NeRF- or 3DGS-style output focused on visual fidelity rather than explicit geometric constraints. Natural-language-conditioned methods Lin & Mu (2024); Feng et al. (2023); Hu et al. (2024) are intuitive to specify but difficult to ground precisely within a room of irregular shape. Other conditions include a single image Koo et al. (2026), which is geometrically underdetermined outside its field of view, and real-to-sim reconstruction, which reproduces an existing physical room rather than synthesizing a novel populated layout. In contrast, PROSCENE conditions on a floor plan: a simple, directly specified input that applies to rooms of arbitrary shape rather than only the rectangular rooms assumed by unconditioned or graph-conditioned methods, and has the potential to populate a full house-level layout for embodied-agent or VR/AR training.

Floor-Plan-Conditioned Scene Generation. Most floor-plan-conditioned methods Paschalidou et al. (2021); Tang et al. (2024); Hu et al. (2026); Yang et al. (2024); Feng et al. (2025) follow a layout-generation-then-retrieval pipeline. The layout is represented as a set of 3D object boxes with box attributes alone Paschalidou et al. (2021); Hu et al. (2026), boxes augmented with a latent shape feature Tang et al. (2024); Yang et al. (2024), or cuboid sets Feng et al. (2025). Object meshes are then retrieved by matching box size Paschalidou et al. (2021); Hu et al. (2026) or latent-feature similarity Tang et al. (2024); Yang et al. (2024). Box-based retrieval provides limited geometric constraints for selecting a mesh that avoids intersection; it also requires database access at generation time and limits output geometry to the available assets. A related retrieval-free approach, CPVQ-VAE de Silva Edirimuni & Mian (2026), instead generates a box layout and decodes each object independently via a class-partitioned VQ-VAE codebook, but likewise imposes no explicit geometric constraint on the decoded objects. In contrast, PROSCENE generates the scene from boxes to cuboids to meshes without retrieval, with spatial-validity guidance applied at both box and cuboid scales.

Prior Spatially Constrained Methods. PhyScene Yang et al. (2024) guides collision avoidance via the IoU of colliding box pairs, and room-boundary compliance by treating the room boundary as a set of infinitely thick wall boxes and computing object-wall box IoU. CASAGPT Feng et al. (2025) instead uses rejection sampling based on cuboid-IoU collisions detected during fine-tuning, rather than guiding the generative model directly, and does not constrain out-of-bounds placement. VoxScene Mao et al. (2026) generates a voxel occupancy grid for each object and controls mesh retrieval via Soft Chamfer similarity; although voxels within the generated grid are mutually exclusive, the retrieved mesh only approximates this exclusive structure and can still produce collisions, and VoxScene imposes no explicit room-boundary constraint.

In PROSCENE, we apply a category-specific size prior at the box scale and collision and out-of-bounds energies at both box and cuboid scales to improve violation-assessment accuracy. Rather than the coarse box-IoU used in prior guidance, we use the separating-axis theorem for collision and a differentiable signed distance to the room boundary for out-of-bounds evaluation. These energies are applied during both training and inference to steer generation toward spatial validity.

Recent scene generation methods based on multimodal LLM agentic systems Yang et al. (2026); Pfaff et al. (2026); Xia et al. (2026) rely on external tools such as physics engines to ensure physical plausibility. Such methods involve substantially more components, including LLM planners, asset generators, and physics simulators, and are significantly more computationally expensive, so we exclude them from direct comparison, following the standalone-method baselines used in this paper.

A.2 Training Across Stages

The three stages of ProScene are trained separately. We use ATISS Paschalidou et al. (2021), DiffuScene Tang et al. (2024), or MiDiffusion Hu et al. (2026) as the stage 1 box-layout generator, whose outputs are provided to the stage 2 layout-cuboid model. Stage 3 is trained separately using aligned cuboid and mesh representations. This modular design allows existing box-layout generators and the pretrained mesh generator to be integrated without end-to-end retraining.

A.3 Stage 2 Implementation Details

Training Data. We construct paired training examples by perturbing ground-truth layouts to obtain coarse input layouts while retaining the original layouts as refinement targets. Object identities and semantic categories are preserved, providing explicit correspondence between input and target objects. The perturbations are applied to object translation, log-size, and yaw.

For the cuboid branch, each object is represented using $K = 12$ cuboid slots. Cuboid geometry is expressed in the object-local coordinate frame and normalized by the longest dimension of its object box. Cuboid slots are randomly permuted during training.

Network Architecture. The layout denoiser is implemented as a DiT-style Transformer Peebles & Xie (2023) with width 512, eight attention heads, and six object self-attention blocks. Floorplan information is encoded using a lightweight convolutional encoder together with 64 sampled boundary tokens, which are incorporated through boundary cross-attention.

The cuboid denoiser is a permutation-equivariant Transformer with eight blocks, width 512, and eight attention heads. It operates independently on each object's cuboid set and uses no cuboid-slot positional embeddings. The first $M = 4$ blocks produce the structural feature passed to the layout denoiser, while the remaining four blocks are conditioned on the resulting layout feature. The scene graph connects each object to its $k = 4$ nearest neighbors based on center-to-center Euclidean distance and augments these edges with intersecting box pairs. It is encoded by two edge-aware attention layers. Timestep conditioning follows the adaLN-Zero design of DiT Peebles & Xie (2023).

Refined Layout Recovery. For object i, let

$$\Delta \mathbf{p}_i = \mathbf{p}_i^{(2)} - \mathbf{p}_i^{(1)}, \qquad \Delta \log \mathbf{s}_i = \log\left(\mathbf{s}_i^{(2)} / \mathbf{s}_i^{(1)}\right), \qquad \Delta\theta_i = \theta_i^{(2)} - \theta_i^{(1)}.$$

Then

$$\Delta \mathbf{b}_i = [\Delta \mathbf{p}_i, \Delta \log \mathbf{s}_i, \sin(\Delta\theta_i), \cos(\Delta\theta_i)].$$

Accordingly, the refined box is recovered from the predicted correction as

$$\mathbf{p}_i^{(2)} = \mathbf{p}_i^{(1)} + \widehat{\Delta \mathbf{p}}_i, \quad \mathbf{s}_i^{(2)} = \mathbf{s}_i^{(1)} \odot \exp\left(\widehat{\Delta \log \mathbf{s}}_i\right), \quad \theta_i^{(2)} = \theta_i^{(1)} + \operatorname{atan2}\left(\widehat{\sin(\Delta\theta_i)}, \widehat{\cos(\Delta\theta_i)}\right). \tag{17}$$

Training and Sampling. We first pretrain the layout and cuboid branches independently. During independent pretraining, the cross-branch conditioning paths are disabled, so each denoiser is optimized using only its branch-specific inputs; bidirectional conditioning is enabled during joint adaptation. The pretrained branches initialize the coupled model, after which the layout network, scene graph, and bidirectional interaction modules are jointly optimized. The stage 2 networks are trained with AdamW optimizer. The cuboid branch uses a learning rate of 8×10^{-5} and batch size 256, while the layout branch uses 1.2×10^{-4} with batch size 128. During joint adaptation, we use learning rates of 5×10^{-7} for the layout network and 5×10^{-6} for the graph and interaction modules.

Both diffusion branches use $T = 1000$ steps with a cosine noise schedule. At inference, we use deterministic DDIM sampling Song et al. (2021) with 64 steps. Late-sampling guidance is applied during the final two sampling steps.

The main training and guidance hyperparameters for stage 2 are summarized in Table 5. As shown in Table 5, we use a smaller cuboid guidance step size η_C than the layout step size η_B. Because cuboid guidance also acts on the predicted existence states, an overly large update can reduce collision

Table 5: Stage 2 training and guidance hyperparameters.

Parameter	Value
γ_L	5
$\lambda_{\text{size}}, \lambda_{\text{box}}, \lambda_{\text{cub}}$	1, 10, 0.2
T_{sv}	499
$\lambda^g_{\text{size}}, \lambda^g_{\text{box}}, \lambda^g_{\text{cub}}$	0.2, 10, 2
η_B, η_C	0.10, 0.002
DDIM steps	64
Guided sampling steps	last 2

energy by suppressing cuboids rather than correcting their geometry. The smaller η_C therefore encourages more conservative cuboid updates.

A.4 Stage 3 Implementation Details

We adapt TRELLIS Xiang et al. (2025) by conditioning its sparse-structure generation stage on the predicted cuboid representation. For each object, the union of its cuboids is voxelized on a 64^3 grid and encoded using the TRELLIS sparse-structure encoder. This implements the cuboid encoding denoted by F_{voxel} in the main text. The resulting cuboid features are combined with CLIP ViT-L/14 Radford et al. (2021) category features and provided to the sparse-structure denoiser through cross-attention. We train the cuboid conditioner and sparse-structure denoiser while keeping the sparse-structure encoder and decoder, CLIP encoder, pretrained structured-latent (SLAT) generator, and mesh decoder frozen.

At inference, we use 50 Euler steps for sparse-structure generation and 25 steps for SLAT generation. Each generated mesh is centered in its local coordinate frame, uniformly scaled to match the corresponding refined box along the source mesh's longest axis, and transformed using the predicted yaw and translation to obtain the final scene.

A.5 Evaluation Details

For each stage 1 generator and room category, the baseline and ProScene use the same generated layouts, floor plans, semantic categories, and object identities. For FID evaluation, generated and ground-truth scenes are rendered as 256×256 top-down orthographic images using a shared semantic color palette. We use orthographic half-extents of $3.1\,\text{m}$ for bedrooms and $6.1\,\text{m}$ for living and dining rooms. Collision and out-of-bounds metrics are evaluated on the assembled object meshes using a voxel size of $0.05\,\text{m}$ following ReSpace Bucher & Armeni (2025).

A.6 Spatial-Validity Energy Details

SAT-Based Collision Energy. For the upright boxes used in our scene representation, the separating-axis theorem reduces to four horizontal axes and the shared vertical axis. For box i with center $\mathbf{p}_i$, full size $\mathbf{s}_i$, and yaw θ_i, its horizontal directions are

$$\mathbf{u}_i = (\cos\theta_i, \sin\theta_i), \qquad \mathbf{v}_i = (-\sin\theta_i, \cos\theta_i). \tag{18}$$

For a horizontal unit axis $\mathbf{a}$, the projected half-width is

$$R_i(\mathbf{a}) = \frac{s_{ix}}{2}|\mathbf{a}^\top \mathbf{u}_i| + \frac{s_{iz}}{2}|\mathbf{a}^\top \mathbf{v}_i|. \tag{19}$$

For a pair of boxes (i, k), let $\mathcal{A}_{ik} = \{\mathbf{u}_i, \mathbf{v}_i, \mathbf{u}_k, \mathbf{v}_k\}$. The overlap along horizontal axis $\mathbf{a}$ is

$$d_{ik}(\mathbf{a}) = R_i(\mathbf{a}) + R_k(\mathbf{a}) - \left|\mathbf{a}^\top(\mathbf{p}_{k,xz} - \mathbf{p}_{i,xz})\right|, \tag{20}$$

and the vertical overlap is $d^y_{ik} = (s_{iy} + s_{ky})/2 - |p_{iy} - p_{ky}|$. We compute the SAT penetration used in the main text as

$$\text{Pen}_{\text{SAT}}(b_i, b_k) = \left[\min\left(\min_{\mathbf{a}\in\mathcal{A}_{ik}} d_{ik}(\mathbf{a}), d^y_{ik}\right)\right]_+. \tag{21}$$

If the boxes are separated along any candidate axis, the minimum overlap is non-positive and the energy becomes zero. Otherwise, the minimum overlap gives their SAT penetration.

Cuboid-Scale Collision Evaluation. The cuboids $c_{ij} = (\mathbf{x}_{ij}, \mathbf{d}_{ij}, e_{ij})$ in the main text are represented in the local coordinate frame of object i. Before evaluating $\mathrm{Pen}_{\mathrm{SAT}}(c_{ij}, c_{k\ell})$ in Eq. 10, both cuboids are transformed to the scene frame using their refined parent boxes.

For object i, let $L_i = \|\mathbf{s}_i^{(2)}\|_\infty$. The scene-space geometry of cuboid c_{ij} is

$$\mathbf{p}_{ij} = \mathbf{p}_i^{(2)} + R_y(\theta_i^{(2)})\,(L_i \mathbf{x}_{ij})\,, \qquad \mathbf{s}_{ij} = L_i \mathbf{d}_{ij}, \qquad \theta_{ij} = \theta_i^{(2)}, \tag{22}$$

where $R_y(\theta)$ denotes rotation about the vertical axis. The SAT penetration in Eq. 10 is evaluated using these scene-space parameters.

Only object pairs whose parent boxes have positive SAT penetration are evaluated at the cuboid scale, corresponding to $\mathcal{N}_i^{\mathrm{col}}$ in the main text. Cuboid pairs belonging to the same object are excluded. The resulting pairwise penetrations are weighted by the predicted cuboid existence probabilities as defined in Eq. 10.

Boundary and Size Energies. For the out-of-bounds energies in Eqs. 9 and 10, $\mathcal{Q}_i$ and $\mathcal{Q}_{ij}$ are constructed as 3×3 grids over the corresponding box and cuboid footprints, respectively. Each grid contains the four corners, four edge midpoints, and the center. The signed-distance field $d_{\mathcal{F}}$ is computed from the binary floor-plan mask using Euclidean distance transforms of the valid and invalid regions, with positive values outside the valid room region. The signed distance $d_{\mathcal{F}}$ is sampled using bilinear interpolation to preserve gradients with respect to the predicted geometry.

The category-specific bounds $\boldsymbol{\ell}_{y_i}$ and $\mathbf{u}_{y_i}$ in Eq. 9 are defined by the 1st and 99th percentiles of the training-set log sizes. Global bounds are used for categories with fewer than eight training examples.

A.7 Extra Visualization

We provide additional qualitative comparisons between the baseline methods and ProScene in Fig. 4. Across different layouts, ProScene consistently reduces spatial violations by adjusting object box sizes, positions, and orientations. This reduces collisions and out-of-bounds placements and can also reposition objects that obstruct usable parts of the room, as illustrated by the second bedroom example. These results further demonstrate the ability of ProScene to improve the spatial validity of generated scenes.

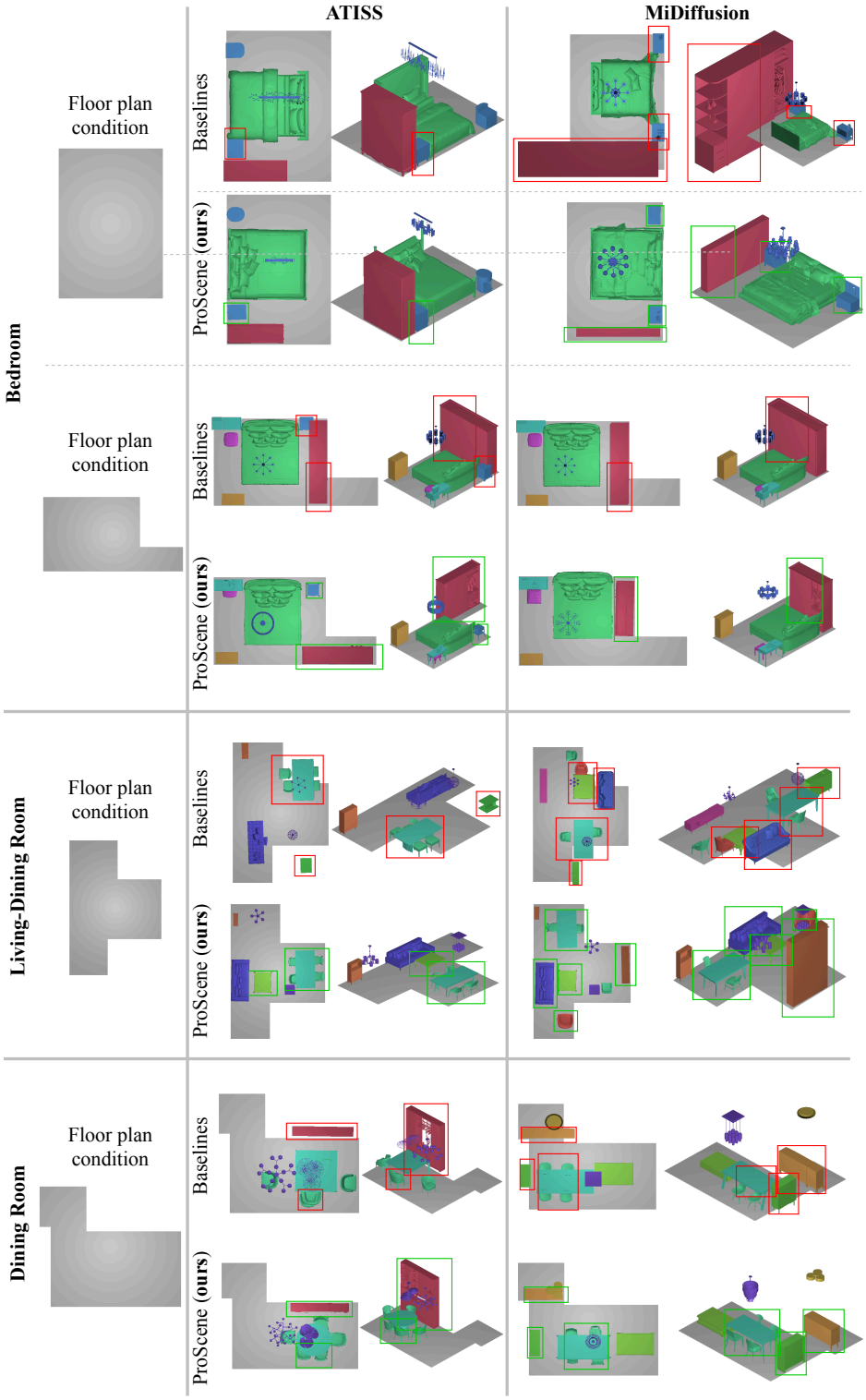

Figure 4: Additional qualitative comparison of ATISS, MiDiffusion, and PROSCENE. For each room type, the first row shows baseline outputs from two views, while the second row shows the corresponding outputs of PROSCENE. Red boxes highlight collision and out-of-bounds violations in the baseline scenes, while green boxes indicate the corresponding regions corrected by PROSCENE. The input floor plan is shown on the left.