LEARNING GEOMETRIC CORRECTIONS FOR PHYSICALLY PLAUSIBLE 3D INDOOR SCENE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

3D indoor scene generation provides a foundation for virtual reality, interactive gaming, robot simulation, and embodied AI, requiring environments that are both diverse and physically sensible. Although diffusion models have advanced layout synthesis, generated scenes may still contain collisions, boundary violations, scale inconsistencies, and obstructed paths. Physics-guided methods often rely on predefined constraints or search-based refinement, making it difficult to adapt correction decisions to evolving scene states, while jointly correcting position, size, and orientation can cause interference among attribute updates and introduce new spatial violations when resolving local conflicts. We present a framework that learns geometric corrections for physically plausible 3D indoor scene generation. It builds a multi-relational semantic–geometric scene graph, fuses semantic and direction-aware geometric context, and predicts gated corrections for position, size, and orientation through factor-aware branches. Counterfactual perturbations provide supervision, and the predicted corrections are injected into diffusion posterior updates during denoising. Extensive experiments demonstrate that our method achieves leading layout fidelity and collision reduction through state-adaptive, factor-aware geometric correction, jointly advancing generation quality and physical plausibility. Our code is available at `https://anonymous.4open.science/status/LGC-1B50`.

1 INTRODUCTION

Generating realistic indoor 3D scenes is a core problem for 3D content creation, virtual reality, simulation, and embodied intelligence Tam et al. (2026); Tang et al. (2024); Yue et al. (2025); Sun et al. (2025); Xia et al. (2026). A useful scene generator must not only place semantically appropriate objects, but also ensure physically feasible and functionally plausible layouts, avoiding collisions, implausible object scales, and blocked paths while preserving plausible support and functional relations Yang et al. (2024a); Fang et al. (2025); Li et al. (2026); Jiang et al. (2026).

Recent LLM/VLM and agentic frameworks have substantially improved language understanding, spatial reasoning, and feedback-driven refinement in 3D scene generation. Related research has progressed from language-driven layout planning Feng et al. (2023); Yang et al. (2024b) to numerical spatial reasoning Ran et al. (2026), native 3D layout modeling Wan et al. (2026); Feng et al. (2026), and scene generation and editing with hierarchical optimization or visual–physical feedback Sun et al. (2025); Jiang et al. (2026); Xia et al. (2026). However, when precise low-level geometric adjustments to the current layout are required, these frameworks typically delegate the concrete modifications to continuous optimization, search, or tool execution. This motivates investigating how scene-level assessment can inform the direct prediction of object-wise geometric corrections during generation.

Another line of work incorporates object relations and physical constraints into layout generation and optimization Wang et al. (2019); Para et al. (2021); Wu et al. (2026). Early methods rely on given relations to drive scene generation Tang et al. (2024); Zhai et al. (2023), as illustrated in Fig. 1(a). Recent physics-guided diffusion methods further introduce physical information, including collisions, boundaries, accessibility and spatial relations, into the sampling process Yang et al. (2024a); Fang et al. (2025); Christopher et al. (2024); Li et al. (2026), as illustrated in Fig. 1(b). For methods that apply physical correction after substantial layout structure has been established, resolving

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.

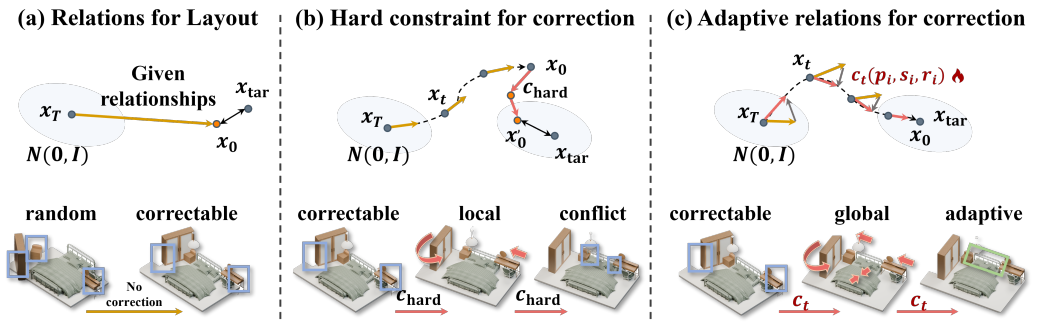

Figure 1: Illustration of (a) relation-based scene generation, (b) hard correction using explicit rules, and (c) our scene-conditioned correction framework. Our framework uses the current state x_t to predict attribute-specific geometric updates during diffusion denoising.

violations can require modifying already formed object relations. When corrections are computed using explicitly formulated rules or objectives, their effectiveness depends on the constraints and interactions captured by these formulations. If relevant interactions are not adequately accounted for, correcting one relation may introduce conflicts elsewhere. These considerations motivate learning continuous geometric corrections from the global scene state and applying them during denoising, so that refinement can adapt to the evolving layout.

Developing such a correction mechanism for diffusion-based indoor scene generation requires addressing two closely related issues. First, the model must perceive local physical conflicts while considering object relations, room boundaries, doors, and windows across the scene to reduce the risk of introducing violations elsewhere. Second, it must use the evolving scene state to determine adjustments to position, size, and orientation during denoising, before conflicts accumulate into a layout requiring substantial final correction. Updates to position, size, and orientation have different geometric effects, but all depend on the surrounding layout. This motivates using a shared scene representation to inform separate predictions for each attribute.

To address these challenges, we propose a geometric correction framework for diffusion-based indoor 3D scene generation. At each guidance timestep, a dynamic semantic–geometric representation captures object interactions and room context from the denoised layout. Using this representation, factor-specific branches determine whether and how to correct each object's position, size, and orientation. As shown in Fig. 1(c), the predicted corrections are adapted to the current layout and incorporated into diffusion posterior updates to refine intermediate layouts during denoising. Experiments show consistent improvements in distributional fidelity and collision reduction over existing methods. Ablation and scheduling analyses validate the effectiveness of joint semantic–geometric encoding, factor-aware correction, and early-stage guidance. Our main contributions are:

- We introduce a dynamic semantic–geometric state encoder that captures object interactions, directional spatial relations, and room-boundary constraints.
- We propose a factor-aware correction predictor trained on counterfactual samples constructed by perturbing valid layouts, which uses factor-specific scene context to determine whether and how to adjust object position, size, and orientation during diffusion denoising.
- Experiments demonstrate consistent improvements in distributional fidelity and collision reduction. Ablation and scheduling studies validate the benefits of joint semantic–geometric encoding, factor-aware correction, and early-stage guidance.

2 Related Work

2.1 Constraint-Driven 3D Indoor Scene Generation

Early approaches generated furniture layouts with manually designed interior rules and optimization objectives Merrell et al. (2011). Subsequent methods progressively moved from relation-based

planning and constraint optimization to directly incorporating learned object relations and structural priors into generative models, improving distribution modeling and relational consistency Wang et al. (2019); Para et al. (2021); Wei et al. (2023); Zhai et al. (2023); Yue et al. (2025); Wu et al. (2026). DiffuScene Tang et al. (2024) uses diffusion to generate realistic and diverse layouts, while MiDiffusion Hu et al. (2026) introduces multimodal conditions for more controllable synthesis.

Recent studies introduce physical constraints directly into generation and sampling. PhyScene Yang et al. (2024a) incorporates collision, room-boundary, and accessibility guidance, while related methods further explore scene priors and constrained sampling Fang et al. (2025); Christopher et al. (2024). SPREAD Li et al. (2026) uses geometry-aware representations and differentiable objectives for collisions, spatial relations, and gravity, while Global Graph-Validated Optimization Huang et al. (2026) enhances multi-object consistency through scene-graph validation and global–local optimization. Together, these methods highlight the value of relational modeling and physical constraints, but their corrections still rely largely on predefined objectives, projection rules, or search and optimization. In contrast, we learn object-level and attribute-level continuous geometric corrections from the current relational state.

2.2 Large-Model-Driven Scene Generation and Optimization

Large language models and vision-language models have expanded semantic planning and spatial reasoning for open-vocabulary 3D scene generation. Related methods explore language-model-based layout generation, complex 3D environment construction, and direct numerical spatial reasoning Feng et al. (2023); Yang et al. (2024b); Ran et al. (2026). LayoutVLM Sun et al. (2025) converts VLM-predicted spatial relations into differentiable objectives for continuous optimization, improving semantic consistency and physical plausibility.

Recent work explores direct 3D spatial modeling. Related methods investigate native 3D geometric reasoning and autoregressive layout generation Wan et al. (2026); Feng et al. (2026), while HOG-Layout Jiang et al. (2026) combines hierarchical representations, vision-language reasoning, and explicit optimization for scene generation and editing. These studies advance generative 3D spatial modeling, whereas we learn fine-grained continuous corrections conditioned on the current physical state during diffusion generation.

Another line of work uses feedback, critics, and tool calls for iterative scene generation, assessment, and correction Hu et al. (2024); Yang et al. (2026); Xia et al. (2026), while SceneOrchestra He et al. (2026) learns complete tool-use trajectories for more efficient agentic synthesis. These systems support scene-level assessment and self-correction, but fine-grained geometry is mainly adjusted through tool execution, program modification, or external optimization. In contrast, we directly model object-level and attribute-level correction requirements from the current diffusion state and predict continuous geometric updates.

3 Method

In **Section** 3.1, semantic and geometric graphs are dynamically constructed from the current denoised layout estimate. In **Section** 3.2, an adaptive correction model is developed to determine whether the position, size, and orientation of each furniture instance require adjustment and to predict the corresponding corrections. In **Section** 3.3, the predicted corrections are applied to the current layout, which is then used to continue the subsequent reverse diffusion process.

3.1 Dynamic Semantic–Geometric State Encoding

At each guidance timestep t, we compute the denoised layout encoding $\hat{X}_0^{(t)}$ from the intermediate denoising state $\mathbf{x}_t$ Hu et al. (2026). We then extract the components corresponding to furniture category, position, size, and orientation to obtain the physical layout $U^{(t)} = \{(\mathbf{c}_i, \mathbf{p}_i, \mathbf{s}_i, \mathbf{r}_i)\}_{i=1}^{N_t}$, and N_t is the number of valid furniture instances. Combining this layout with the boundary-opening information $\mathcal{O}$ (doors and windows) and the room boundaries $\mathcal{B}$, we obtain the scene state $\mathcal{S}^{(t)} = \{U^{(t)}, \mathcal{O}, \mathcal{B}\}$. To account for both the functional roles of objects and their current spatial relations during layout correction, we dynamically construct a semantic graph $\mathcal{G}_{\text{sem}}^{(t)}$, a geometric graph $\mathcal{G}_{\text{geo}}^{(t)}$, and a set of furniture–wall-segment relations $\mathcal{R}_{\text{bnd}}^{(t)}$ from the scene state $\mathcal{S}^{(t)}$.

Algorithm 1: Semantic–Geometric Scene Representation

Require: Scene state $\mathcal{S}^{(t)} = \{U^{(t)}, \mathcal{O}, \mathcal{B}\}$

Semantic graph $\mathcal{G}_{\text{sem}}^{(t)}$:

$\text{Node}_{\text{sem}}^{t} = \mathcal{V}_f^{(t)} \cup \mathcal{V}_o \cup \{V_s\}$, $\text{NodeAttr}_i = [\mathbf{c}_i, \mathbf{p}_i, \mathbf{s}_i, \mathbf{r}_i]$

$\mathcal{E}_{\text{sem}}^{(t)} = \{\text{radius, projection overlap, furniture–opening, scene-membership}\}$

$\text{Edge}_{\text{sem}}^{t} = \{(\text{type} \in \mathcal{E}_{\text{sem}}^{(t)}, \text{relative geometry})\}$

Geometric graph $\mathcal{G}_{\text{geo}}^{(t)}$:

$\text{Node}_{\text{geo}}^{t} = \mathcal{V}_f^{(t)} \cup \mathcal{V}_o$, $\text{NodeAttr}_i = (\mathbf{s}_i^{(t)}, \mathbf{v}_i^{(t)})$

$e_{ij}^{(t)} \mapsto (\mathbf{s}_{ij}^{(t)}, \mathbf{v}_{ij}^{(t)})$, $(e_{ij}^{(t)} \in \text{Edge}_{\text{sem}}^{t}, i, j \in \text{Node}_{\text{geo}}^{t})$

$\text{Edge}_{\text{geo}}^{t} = \{(\mathbf{s}_{ij}^{(t)}, \mathbf{v}_{ij}^{(t)})\}_{(i,j)}$

Furniture–wall relations:

$\mathcal{R}_{\text{bnd}}^{(t)} = \{r_{i\ell}^{(t)} \mid (i, w_\ell) \in \mathcal{V}_f^{(t)} \times \mathcal{W}\}$, $\mathcal{W} = \text{Segments}(\mathcal{B})$

$r_{i\ell}^{(t)} = (\mathbf{s}_{i\ell}^{(t)}, \mathbf{v}_{i\ell}^{(t)}, \text{segment length})$

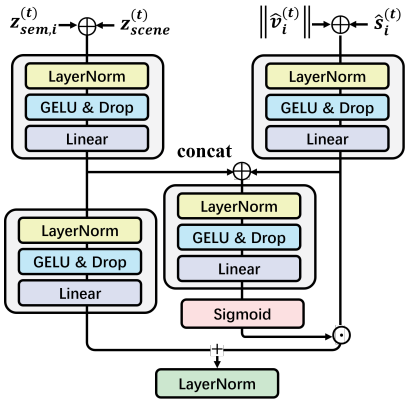

Figure 2: Gated semantic–geometric feature fusion. $\odot$ denotes element-wise multiplication.

Dynamic graph and boundary construction. At each guidance step t, we construct two complementary graphs to capture semantic context and pairwise geometric relations. The semantic graph $\mathcal{G}_{\text{sem}}^{(t)}$ contains furniture nodes $\mathcal{V}_f^{(t)}$, door and window nodes $\mathcal{V}_o$, and a virtual scene node V_s connected to all physical objects to aggregate and distribute scene-level context. In contrast, the geometric graph $\mathcal{G}_{\text{geo}}^{(t)}$ contains only physical-object nodes $\mathcal{V}_f^{(t)} \cup \mathcal{V}_o$, with node and edge features represented as scalar–vector pairs $(\mathbf{s}_i^{(t)}, \mathbf{v}_i^{(t)})$ and $(\mathbf{s}_{ij}^{(t)}, \mathbf{v}_{ij}^{(t)})$. Scalar features describe object properties and the types and strengths of geometric relationships, while vector features retain spatial directions. Together, they enable message passing to account for both the severity and direction of proximity conflicts, supporting direction-aware geometric correction. To explicitly model room boundaries, we additionally construct a set of furniture–wall-segment relations $\mathcal{R}_{\text{bnd}}^{(t)} = \{r_{i\ell}^{(t)}\}$, where $r_{i\ell}^{(t)}$ denotes the relation between furniture instance i and the ℓ-th valid wall segment. These boundary relations provide explicit boundary information for geometric reasoning. Construction details are provided in Algorithm 1 and Appendix A.

Semantic and geometric encoders. To incorporate inter-object relations and scene context into node representations, we encode the semantic and geometric graphs using graph neural networks, extracting semantic and geometric features, respectively:

$$\left(\{z_{\text{sem},i}^{(t)}\}, z_{\text{scene}}^{(t)}\right) = G_{\text{sem}}\left(A_{\text{sem}}\left(\mathcal{G}_{\text{sem}}^{(t)}\right)\right), \quad \left\{\left(\hat{\mathbf{s}}_i^{(t)}, \hat{\mathbf{v}}_i^{(t)}\right)\right\} = G_{\text{geo}}\left(\mathcal{G}_{\text{geo}}^{(t)}, \mathcal{R}_{\text{bnd}}^{(t)}\right), \quad i \in \mathcal{V}_f^{(t)} \cup \mathcal{V}_o. \tag{1}$$

Here, G_{sem} is a pretrained five-layer GraphTripleConvNet, and G_{geo} is a three-layer GVP message-passing network. The input adapter A_{sem} projects graph features to match the input dimensions of G_{sem}. The outputs $z_{\text{sem},i}^{(t)}$ and $z_{\text{scene}}^{(t)}$ denote physical-node semantic features and scene-level semantic features aggregated by the virtual scene node, respectively. The outputs $\hat{\mathbf{s}}_i^{(t)}$ and $\hat{\mathbf{v}}_i^{(t)}$ are node-level scalar and vector representations aggregated from neighboring-object relations and furniture–wall-segment constraints, encoding metric and directional geometric information for subsequent fusion and geometric correction.

Semantic-geometric fusion. As illustrated in Fig. 2, the semantic branch combines object-level features with scene-level context. The geometric branch combines scalar features $\hat{\mathbf{s}}_i^{(t)}$ with vector norms $\|\hat{\mathbf{v}}_i^{(t)}\|$. Both branches are projected into a shared feature space. A sigmoid gate uses both representations to modulate the geometric features. The gated features are added to a transformed semantic representation, followed by layer normalization. This fusion $\mathbf{f}_i^{(t)}$ retains semantic context while adapting the contribution of geometric information to the current layout, supporting subsequent geometric correction.

3.2 FACTOR-AWARE CORRECTION PREDICTION

The position, size, and orientation of a furniture instance jointly determine its spatial occupancy and its relations with the surrounding environment. Because a geometric inconsistency may be corrected by modifying multiple factors, a unified predictor can entangle the distinct behaviors required for position, size, and orientation correction. We therefore use factor-specific branches as an inductive bias to accommodate the distinct geometric effects of position, size, and orientation, while retaining shared scene context for coordinated correction.

For each factor $k \in \{P, S, R\}$, where P, S, and R denote position, size, and orientation, respectively, we concatenate the fused furniture representation $\mathbf{f}_i^{(t)}$, the magnitude of the geometric vector feature $\|\hat{\mathbf{v}}_i^{(t)}\|$, the semantic representation $\mathbf{z}_{\text{sem},i}^{(t)}$, and a learnable factor embedding $\mathbf{e}_k$. A factor-specific MLP block with a concatenative skip connection maps this descriptor to a contribution logit $\alpha_i^{k,(t)}$. This can be interpreted as an attention-style, factor-conditioned weighting mechanism, where the factor embedding provides a query-like condition, the concatenated furniture descriptor provides the features used to compute the attention weights, and the fused furniture representation serves as the value feature. The logits are normalized across the furniture instances in the current scene:

$$\hat{\alpha}_i^{k,(t)} = \frac{\exp\left(\alpha_i^{k,(t)}\right)}{\sum_{j \in \mathcal{V}_f^{(t)}} \exp\left(\alpha_j^{k,(t)}\right)}, \qquad i \in \mathcal{V}_f^{(t)}. \tag{2}$$

The normalized weight $\hat{\alpha}_i^{k,(t)}$ represents the relative contribution of furniture instance i to the scene context associated with factor k. Since the three branches use independent parameters, they can assign different contribution weights according to the relational cues relevant to position, size, and orientation correction. The factor-specific scene context vector is then computed as

$$\mathbf{c}_{\text{scene}}^{k,(t)} = \sum\nolimits_{i \in \mathcal{V}_f^{(t)}} \hat{\alpha}_i^{k,(t)} \mathbf{f}_i^{(t)}. \tag{3}$$

Given the factor-specific scene context $\mathbf{c}_{\text{scene}}^{k,(t)}$, we combine it with the instance-level geometric feature $\hat{\mathbf{v}}_i^{(t)}$ and fused furniture representation $\mathbf{f}_i^{(t)}$, and feed the resulting feature into a factor-specific MLP. Two parallel linear heads predict an unconstrained candidate correction $\Delta\mathbf{u}_i^{k,(t)}$ and a scalar gating logit $l_i^{k,(t)}$. The latter is passed through a sigmoid activation and then binarized using a threshold determined during training, yielding a binary gate $g_i^{k,(t)} \in \{0, 1\}$. The final correction is computed as

$$\Delta\hat{\mathbf{u}}_i^{k,(t)} = g_i^{k,(t)} \Delta\mathbf{u}_i^{k,(t)}. \tag{4}$$

For the position factor, we additionally aggregate the vector states to construct a position-specific scene context that preserves spatial directional information:

$$\mathbf{v}_{\text{scene}}^{P,(t)} = \sum\nolimits_{i \in \mathcal{V}_f^{(t)}} \hat{\alpha}_i^{P,(t)} \hat{\mathbf{v}}_i^{(t)}. \tag{5}$$

Alongside the candidate correction and gating heads, a third parallel linear head takes the same hidden representation from the position-specific MLP. Its output is multiplied by $\mathbf{v}_{\text{scene}}^{P,(t)}$ to produce an additional correction, which is then added to $\Delta\hat{\mathbf{u}}_i^{P,(t)}$, with the updated result denoted by the same symbol. The dataset construction and model training procedures are described in the Appendix B and Appendix C.

3.3 DENOISING-TIME CORRECTION AND POSTERIOR UPDATE

During reverse diffusion, we introduce correction guidance at appropriate timesteps $t \in \mathcal{T}_g$. At each timestep $t \in \mathcal{T}_g$, we obtain the denoised layout estimate $\hat{X}_0^{(t)}$ from the current state $\mathbf{x}_t$ and construct its semantic–geometric graph representation as described in Section 3.1. The factor-aware correction model then predicts the position, size, and orientation corrections for each furniture instance i. We concatenate the final corrections from the three branches into a unified correction vector in physical layout space:

$$\Delta\mathbf{u}_{\text{phys},i}^{(t)} = \text{Concat}\left(\Delta\hat{\mathbf{u}}_i^{P,(t)}, \Delta\hat{\mathbf{u}}_i^{S,(t)}, \Delta\hat{\mathbf{u}}_i^{R,(t)}\right), \tag{6}$$

The correction model is trained on our constructed counterfactual dataset, whose correction magnitudes may differ from those required during reverse diffusion. To account for this scale mismatch, we first clip and normalize the predicted correction:

$$\boldsymbol{\rho}_i^{(t)} = \frac{\mathrm{clip}\Big(\Delta \mathbf{u}_{\mathrm{phys},i}^{(t)}, -\boldsymbol{\gamma}, \boldsymbol{\gamma}\Big)}{\boldsymbol{\gamma}}, \tag{7}$$

where $\boldsymbol{\gamma}$ is a positive threshold vector derived from counterfactual correction statistics, matching the dimensionality of the predicted correction. Element-wise clipping limits correction magnitudes, and element-wise division by $\boldsymbol{\gamma}$ yields normalized ratios in $[-1, 1]$.

We then rescale these ratios according to the current furniture states:

$$\Delta \tilde{\mathbf{u}}_{\mathrm{phys},i}^{(t)} = \lambda \boldsymbol{\rho}_i^{(t)} \odot \left(\mathbf{u}_{\max}^{(t)} - \mathbf{u}_{\min}^{(t)} \right), \tag{8}$$

where λ denotes the correction strength, $\mathbf{u}_{\max}^{(t)}$ and $\mathbf{u}_{\min}^{(t)}$ denote the component-wise maximum and minimum physical layout values across furniture instances in the current scene, respectively, and $\odot$ denotes element-wise multiplication. This rescaling adapts the correction magnitudes to the value ranges of the current layout while preserving their signs.

We add the rescaled corrections $\Delta \tilde{\mathbf{u}}_{\mathrm{phys},i}^{(t)}$ to the corresponding position, size, and orientation components of $\hat{X}_0^{(t)}$, yielding the corrected clean layout estimate $\tilde{X}_0^{(t)}$. We then use this estimate in the DDPM posterior update:

$$\mathbf{x}_{t-1} = \frac{\sqrt{\bar{\alpha}_{t-1}}\,\beta_t}{1-\bar{\alpha}_t} \tilde{X}_0^{(t)} + \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \mathbf{x}_t + \sqrt{\tilde{\beta}_t}\,\boldsymbol{\epsilon}, \tag{9}$$

where β_t is the noise variance, $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. The posterior variance is $\tilde{\beta}_t = \beta_t(1-\bar{\alpha}_{t-1})/(1-\bar{\alpha}_t)$, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is standard Gaussian noise. For $t = 1$, $\tilde{\beta}_1 = 0$, and hence no additional noise is added in the final denoising step.

4 Experiments

4.1 Experimental Setup

We conduct experiments on the 3D-FRONT Fu et al. (2021a) benchmark dataset, whose indoor scenes are furnished with high-quality furniture models from 3D-FUTURE Fu et al. (2021b). Following the construction procedure described in Appendix C, we generate counterfactual samples by perturbing valid layouts. The bedroom, dining room, and living room datasets contain 125,468/22,045, 32,715/5,625 and 30,680/5,110, training/validation samples, respectively.

Evaluation metrics. Rendering-based visual and semantic fidelity is evaluated using FID, KID, SCA, and CKL. All metrics use shared 256×256 orthographic top-down renderings with category-colored furniture and disabled textures. SCA uses a fixed room-wise 500/500 train/test split to prevent the real and generated versions of the same room from crossing the split. Physical plausibility is measured by Col-pair, Col-obj, Col-scene, OOB-obj, and Overflow. A positive IoU between oriented 3D bounding boxes denotes a collision; the first three metrics report the fractions of colliding pairs, involved objects, and affected scenes, respectively. OOB-obj is the fraction of objects with more than 2% of their footprint outside the floor region, while Overflow is the outside footprint area divided by the total furniture footprint area.

Implementation details. We train separate geometric-correction predictors for bedrooms, living rooms, and dining rooms using the corresponding room-type counterfactual data, while keeping the pretrained diffusion backbone frozen. During sampling, correction is applied only at timesteps $t \in \mathcal{T}_g$ and modifies object positions, sizes, and orientations, without directly changing category or object-feature channels. All experiments were conducted on a server equipped with eight NVIDIA RTX 3090 GPUs. Training details and model selection are in Appendix B.

4.2 Comparison with Existing Methods

We compare our method with ATISS Paschalidou et al. (2021), DiffuScene Tang et al. (2024), MiDiffusion Hu et al. (2026), PhyScene Yang et al. (2024a), and GPT-6 Astra. For the GPT-6 Astra

Table 1: Generation quality across rooms of three types. KID and CKL are scaled by 10^3 and 10^2, respectively. **Bold** and underlined values denote the best and second-best results within each metric.

Method	Bedroom				Dining Room				Living Room			
	FID ↓	KID ↓	SCA	CKL ↓	FID ↓	KID ↓	SCA	CKL ↓	FID ↓	KID ↓	SCA	CKL ↓
ATISS	49.58	8.52	78.79	1.15	37.53	7.73	80.03	0.92	32.82	11.59	79.23	0.36
DiffuScene	47.92	8.69	77.44	**1.04**	37.62	6.63	79.44	0.76	31.58	**10.89**	78.66	**0.33**
MiDiffusion	47.69	8.28	68.81	1.73	33.40	6.09	**66.12**	0.65	31.90	11.76	76.98	0.47
PhyScene	43.41	7.97	67.14	1.37	33.54	**5.20**	66.76	0.74	31.47	11.68	**75.94**	0.44
GPT-6 Astra	48.27	8.42	73.65	1.29	36.98	6.51	77.10	0.97	31.88	11.29	76.52	0.59
Ours	**42.79**	**7.67**	**65.92**	1.52	**31.45**	5.22	67.24	**0.63**	**31.17**	11.33	77.72	0.45

Table 2: Physical plausibility across rooms of three types. **Bold** and underlined values denote the best and second-best results within each metric.

Method	Bedroom					Dining Room					Living Room				
	Col-pair ↓	Col-obj ↓	Col-scene ↓	OOB-obj ↓	Overflow ↓	Col-pair ↓	Col-obj ↓	Col-scene ↓	OOB-obj ↓	Overflow ↓	Col-pair ↓	Col-obj ↓	Col-scene ↓	OOB-obj ↓	Overflow ↓
ATISS	15.65	50.97	86.34	40.00	5.30	17.50	51.05	89.40	41.08	5.72	17.46	53.08	87.97	43.97	5.79
DiffuScene	14.75	50.86	83.33	**34.20**	4.25	15.16	51.18	85.80	35.29	4.53	15.08	52.17	85.87	35.91	**4.38**
MiDiffusion	14.25	50.32	82.12	35.73	4.32	14.95	50.42	85.04	**34.98**	4.53	15.02	50.87	86.03	35.69	4.42
PhyScene	14.23	49.03	80.42	34.51	4.22	14.85	50.57	84.08	35.13	4.44	14.92	50.07	84.95	**35.08**	4.40
GPT-6 Astra	16.83	52.47	85.69	37.08	4.54	17.34	52.49	86.63	35.24	4.91	17.50	53.17	87.55	39.82	4.82
Ours	**13.83**	**47.42**	**78.67**	35.67	**4.21**	**14.50**	**48.04**	**83.13**	35.69	**4.37**	**14.85**	**48.57**	**82.86**	35.15	4.43

baseline, we provide the room outline and a textual description as conditioning inputs and instruct the model to directly operate Blender to construct and render the scene. Table 1 reports generation quality, whereas Table 2 reports physical plausibility across the three room categories.

Generation quality. Our method achieves the lowest FID for all three room categories, with relative reductions of 10.7%, 16.4%, and 1.3% over DiffuScene for bedrooms, dining rooms, and living rooms, respectively. It also obtains the best KID and SCA for bedrooms, as well as the best CKL and second-best KID for dining rooms. Compared with GPT-6 Astra, our method achieves lower FID across all three room categories and outperforms it on all four generation-quality metrics in dining rooms, demonstrating stronger alignment with the reference layout distribution. The qualitative comparison results in Figure 3 complement these findings, showing more coherent furniture groupings and better spatial organization in the ours examples. The gains remain metric-dependent: DiffuScene achieves better CKL for bedrooms and living rooms, while competing methods obtain better SCA for dining and living rooms.

Physical plausibility. Our method achieves the lowest Col-pair, Col-obj, and Col-scene values for all three room categories, outperforming both unguided DiffuScene and the guided PhyScene baseline. Relative to DiffuScene, Col-obj and Col-scene decrease by 3.14–3.60 and 2.67–4.66 percentage points, respectively. Compared with GPT-6 Astra, the corresponding reductions are 4.45–5.05 and 3.50–7.02 percentage points, further demonstrating the advantage of our method in collision control. These improvements consistently hold at the pair, object, and scene levels. Figure 3 provides complementary visual evidence, with our examples exhibiting fewer apparent spatial conflicts and more appropriate relative placement of neighboring objects. Our method also obtains the lowest Overflow for bedrooms and dining rooms. The boundary results are less uniform: OOB-obj remains higher than PhyScene across the three room categories, and living-room Overflow is higher than DiffuScene. Overall, the strongest and most consistent physical gain is collision reduction, while improvements in boundary violations depend on the room category and metric.

4.3 Ablation of Scene Encoding and Factor-Aware Correction

We ablate the semantic encoder, geometric encoder, and factor-aware correction module to assess their contributions to physical plausibility. Encoder-removal variants are trained with the corresponding reduced architectures, rather than constructed by masking inputs to the full model at in-

Table 3: Ablation of scene encoding and correction formulation. Direct 7D denotes direct prediction of the seven correction components. **Bold** values indicate the best result for each metric.

Configuration		Physical plausibility				
Scene encoding	Correction	Col-pair ↓	Col-obj ↓	Col-scene ↓	OOB-obj ↓	Overflow ↓
Semantic-only	Direct 7D	16.84	51.32	87.80	39.51	4.73
Semantic-only	Factor-aware	16.67	50.92	86.00	38.85	4.65
Geometric-only	Direct 7D	16.74	52.43	86.64	37.47	4.72
Geometric-only	Factor-aware	16.43	51.47	86.81	37.36	4.57
Semantic + geometric	Direct 7D	16.38	51.43	83.26	**35.36**	4.46
Semantic + geometric	Factor-aware	**13.83**	**47.42**	**78.67**	35.67	**4.21**

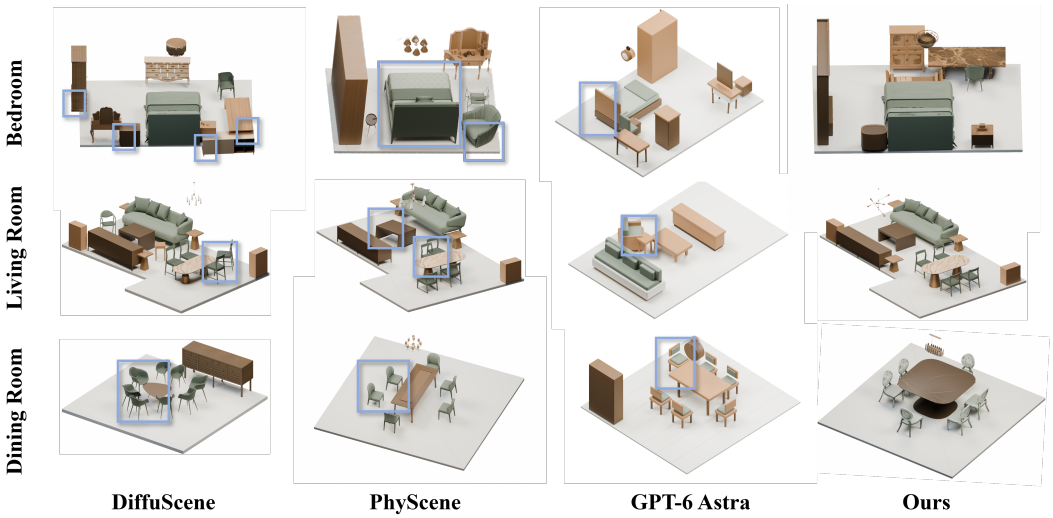

Figure 3: Quantitative comparison results. All results are rendered using a unified pipeline to facilitate visual comparison. The framed regions denote collisions, size errors, position errors, etc. The viewpoint is rotated to show more details.

ference time. Their factor-aware modules are subsequently trained on the corresponding direct-correction backbones, while the full-encoder variants reuse existing checkpoints.

Contribution of scene encoding. Table 3 reports the physical plausibility metrics for all variants.With factor-aware correction fixed, we compare semantic-only, geometric-only, and joint semantic–geometric encoding.Removing either encoder increases Col-pair, Col-obj, and Col-scene relative to the full model, indicating that semantic information and geometric relations provide complementary signals for collision correction.With factor-aware correction fixed, joint semantic–geometric encoding also achieves the lowest OOB-obj and Overflow, demonstrating the benefit of combining semantic and geometric context for boundary compliance.

Contribution of factor-aware correction. We further compare direct 7D correction with factor-aware correction under the same encoder configuration. Factor-aware correction reduces Col-pair and Col-obj across all three encoder configurations, with the largest gains observed under the full semantic–geometric representation: Col-pair decreases from 16.38 to 13.83, Col-obj from 51.43 to 47.42, and Col-scene from 83.26 to 78.67, corresponding to reductions of 2.55, 4.01, and 4.59 percentage points, respectively. In contrast, improvements are smaller under single-encoder configurations, and Col-scene increases slightly with geometric-only encoding. This pattern indicates that factor-aware correction can more effectively utilize the joint semantic–geometric representation. Under the full encoder configuration, Overflow decreases while OOB-obj increases, indicating that collision reduction does not necessarily lead to a lower incidence of boundary violations.

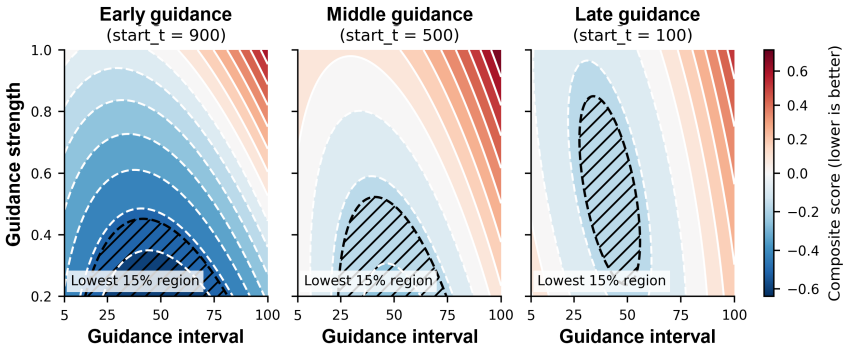

Figure 4: Fitted composite-score surfaces for early ($t_{\text{start}} = 900$), middle (500), and late (100) guidance onset. Lower scores indicate better performance. White lines show contours; hatching marks regions below the 15th percentile in each panel.

Overall, the ablation results show that joint semantic–geometric encoding and factor-aware correction work best together. The gains are primarily reflected in collision reduction, while boundary-violation metrics exhibit a trade-off.

4.4 Effect of Guidance Scheduling

We evaluate the joint effects of guidance onset, interval, and strength on the same 1134 evaluation rooms. We test 36 configurations with $t_{\text{start}} \in \{100, 500, 900\}$, $\Delta t \in \{5, 10, 50, 100\}$, and $\lambda \in \{0.2, 0.6, 1.0\}$. Guidance is applied from t_{start} to $t = 0$, with the final step clamped to zero when necessary. To summarize the trade-offs among collision, boundary, and category-distribution metrics, each metric is standardized across the 36 configurations as $z_m = (m - \mu_m)/\sigma_m$, where μ_m and σ_m are its mean and population standard deviation. We then compute

$$S = \frac{1}{3}\left[\frac{z_{\text{Col-pair}} + z_{\text{Col-obj}} + z_{\text{Col-scene}}}{3} + \frac{z_{\text{OOB-obj}} + z_{\text{Overflow}}}{2} + z_{\text{CKL}}\right]. \tag{10}$$

The three metric groups receive equal weight, and lower values of S indicate better overall performance. We fit a joint quadratic response surface with pairwise interaction terms to characterize the parameter combinations across different guidance onsets.

Figure 4 shows that early onset favors lower correction strengths and moderate intervals. As the onset becomes later, the favorable region shifts toward moderate strengths and becomes more concentrated along the interval axis, indicating that the effects of interval and strength depend on the guidance onset. Early guidance can correct geometric errors before they accumulate during denoising. An intermediate interval allows the diffusion model to establish a global layout between corrections, whereas overly sparse guidance weakens the correction effect. Large λ values may also over-adjust the layout in a single step.

The mean fitted composite scores for early, middle, and late onset are -0.551, -0.187, and -0.154, respectively. We therefore select the interpolated configuration $t_{\text{start}} = 900$, $\Delta t = 40$, and $\lambda = 0.3$ for the main experiments, combining early error correction, sufficiently frequent guidance, and moderate correction strength.

5 Conclusion and Limitations

We introduced a scene-conditioned framework that learns object-wise geometric corrections from semantic and directional information during diffusion denoising. Experiments demonstrate improved distributional fidelity and consistent collision reduction across the evaluated room categories, supporting learned state-dependent correction for indoor scene generation. Controlled perturbations provide practical supervision but may not fully represent errors arising during denoising. Future work will investigate supervision from denoising trajectories, constraint-aware refinement, and more comprehensive physical evaluation.

AI USE STATEMENT

In this study, we used generative artificial intelligence (AI) tools to assist with language editing and grammar correction and to improve the clarity, flow, and readability of the paper. During project development, we also used generative AI tools for limited coding assistance, debugging, and refinement of implementation details. In addition, GPT-6 Astra was used as a comparison baseline to operate Blender to construct and render 3D indoor scenes from room outlines and textual descriptions, as described in the experimental section. AI tools were not used to fabricate experimental results or data, conduct core scientific analyses, or make final judgments regarding the research methods, interpretation of results, or conclusions. Other disclosure items unrelated to these uses are not applicable to this study. All AI-assisted content was carefully reviewed, revised, and verified by the authors. The core research ideas, technical contributions, experimental design, and scientific claims presented in this paper were independently developed and established by the authors. All final decisions concerning the study design, implementation, experiments, analyses, and presentation of results were made by the authors. We take full responsibility for the final content of this work.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. We did not recruit human participants, collect new personal data, or conduct experiments involving human or animal subjects. Our experiments use publicly available 3D-FRONT scenes and 3D-FUTURE furniture assets in accordance with their applicable terms of use. We generate counterfactual layouts through synthetic perturbations of furniture position, size, and orientation; the GPT-6 Astra baseline generates comparison scenes from room outlines and textual descriptions. No personally identifiable information was intentionally accessed or processed, and our experiments do not involve personally identifiable information or raise privacy concerns.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide the implementation of our method in an anonymous repository: `https://anonymous.4open.science/status/LGC-1B50`. The main paper and Appendices A–C describe the model architecture, semantic–geometric graph construction, dataset preparation, counterfactual perturbation generation, source-scene-level data splits, staged training procedures, loss functions, guidance schedules, and evaluation protocols. These details support replication of our experiments on diffusion-based 3D indoor scene generation and geometric correction, including baseline comparisons, ablation studies, and guidance-scheduling analyses.

REFERENCES

Jacob K Christopher, Stephen Baek, and Ferdinando Fioretto. Constrained synthesis with projected diffusion models, 2024. *URL https://arxiv. org/abs/2402.03559*, 2024.

Shaoheng Fang, Haitao Yang, Raymond Mooney, and Qixing Huang. Text-guided interactive scene synthesis with scene prior guidance. In *Computer Graphics Forum*, volume 44, pp. e70039. Wiley Online Library, 2025.

Haoran Feng, Yifan Niu, Zehuan Huang, Yang-Tian Sun, Chunchao Guo, Yuxin Peng, and Lu Sheng. Repurposing 3d generative model for autoregressive layout generation. *arXiv preprint arXiv:2604.16299*, 2026.

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 *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10913–10922. IEEE, 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.

Yun He, Kelin Yu, and Matthias Zwicker. Sceneorchestra: Efficient agentic 3d scene synthesis via full tool-call trajectory generation. *arXiv preprint arXiv:2604.19907*, 2026.

Siyi Hu, Diego Martin Arroyo, Stephanie Debats, Fabian Manhardt, Luca Carlone, and Federico Tombari. Mixed diffusion for 3d indoor scene synthesis. In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1262–1272. IEEE, 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 *ICML*, pp. 19252–19282, 2024.

Jialu Huang, Yingxuan You, Fei Wang, and Zheng Dang. Global graph-validated optimization for vlm-based 3d indoor scene generation. *arXiv preprint arXiv:2608.03064*, 2026.

Haiyan Jiang, Deyu Zhang, Dongdong Weng, Weitao Song, and Henry Been-Lirn Duh. Hog-layout: Hierarchical 3d scene generation, optimization and editing via vision-language models. *arXiv preprint arXiv:2604.10772*, 2026.

Minzhang Li, Kuixiang Shao, Xuebing Li, Yuyang Jiao, Yinuo Bai, Hengan Zhou, Sixian Shen, Jiayuan Gu, and Jingyi Yu. Spread: Spatial-physical reasoning via geometry aware diffusion. *arXiv preprint arXiv:2603.27573*, 2026.

Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. Interactive furniture layout using interior design guidelines. *ACM transactions on graphics (TOG)*, 30(4):1–10, 2011.

Wamiq Para, Paul Guerrero, Tom Kelly, Leonidas J Guibas, and Peter Wonka. Generative layout modeling using constraint graphs. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6690–6700, 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.

Xingjian Ran, Yixuan Li, Linning Xu, Mulin Yu, and Bo Dai. Direct numerical layout generation for 3d indoor scene synthesis via spatial reasoning. *Advances in Neural Information Processing Systems*, 38:125055–125081, 2026.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29469–29478. IEEE, 2025.

Hou In Ivan Tam, Hou In Derek Pun, Austin T. Wang, Xiaohao Sun, Qirui Wu, Han-Hung Lee, Angel X. Chang, and Manolis Savva. Survey on compositional 3D indoor scene generation. *Computer Graphics Forum*, pp. e70591, 2026.

Jiapeng Tang, Yinyu Nie, Lev Markhasin, Angela Dai, Justus Thies, and Matthias Nießner. Diffuscene: Denoising diffusion models for generative indoor scene synthesis. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20507–20518. IEEE, 2024.

Cheng Wan, Yongsen Mao, Wenzheng Wu, Yuxuan Xie, Chucheng Xiang, Runze Wang, Xiang Zhang, Zhongyuan Liu, Rushi Dai, and Yuan Liu. Nala: A 3d native llm layout agent for high-quality 3d scene generation. In *European Conference on Computer Vision*, pp. 285–301. Springer, 2026.

Kai Wang, Yu-An Lin, Ben Weissmann, Manolis Savva, Angel X Chang, and Daniel Ritchie. Planit: Planning and instantiating indoor scenes with relation graph and spatial prior networks. *ACM Transactions on Graphics (TOG)*, 38(4):1–15, 2019.

Qiuhong Anna Wei, Sijie Ding, Jeong Joon Park, Rahul Sajnani, Adrien Poulenard, Srinath Sridhar, and Leonidas Guibas. Lego-net: Learning regular rearrangements of objects in rooms. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19037–19047. IEEE, 2023.

Yingrui Wu, You-Kang Kong, Mingyang Zhao, Weize Quan, Dongmin Yan, and Yang Liu. Caslayout: Cascaded 3d layout diffusion for indoor scene synthesis with implicit relation modeling. *ACM Transactions on Graphics (TOG)*, 45(4):1–16, 2026.

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.

Yandan Yang, Baoxiong Jia, Peiyuan Zhi, and Siyuan Huang. Physcene: Physically interactable 3d scene synthesis for embodied ai. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16262–16272. IEEE, 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, 2026.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16277–16287. IEEE, 2024b.

Liang Yue, Shao-Kui Zhang, Lin Yuan, Yi-Tao Chen, Zirui Zhou, and Song-Hai Zhang. Synthesizing 3d scenes via diffusion model that incorporates indoor scene characteristics. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 9385–9394, 2025.

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.

A Construction Details of Geometric and Semantic Graphs

We use consistent coordinate, dimension, and orientation conventions to compute features for the semantic graph $\mathcal{G}_{\text{sem}}^{(t)}$, the geometric graph $\mathcal{G}_{\text{geo}}^{(t)}$, and the furniture–wall-segment relations $\mathcal{R}_{\text{bnd}}^{(t)}$. All quantities that depend on furniture states are computed from the current physical layout $U^{(t)}$.

Coordinates and dimensions. The room floor-plan centroid is used as the origin. The y-axis is vertical, and the x–z plane is horizontal. We transform room-boundary vertices by subtracting the floor-plan centroid from their world coordinates. For doors and windows, we first map the local center to world coordinates and then apply the same translation; their vertical coordinates are unchanged. The entries of $\mathbf{s}_i$ are ordered as width, depth, and height. Semantic node attributes use half dimensions, $\mathbf{s}_i/2$, while geometric and furniture–wall-segment features use the full dimensions $\mathbf{s}_i$.

Orientation and local coordinates. The orientation $\mathbf{r}_i$ is the rotation about the vertical axis. The width and depth axes of furniture i, expressed in the room frame, are

$$[\cos \mathbf{r}_i,\ 0,\ \sin \mathbf{r}_i], \qquad [-\sin \mathbf{r}_i,\ 0,\ \cos \mathbf{r}_i],$$

respectively. We obtain local coordinates by taking the dot product of a relative horizontal displacement with these two axes. The sign gives the direction and the magnitude gives the distance along the corresponding axis. We use the same convention for object–object and furniture–wall-segment relations. Horizontal vectors are stored as three-dimensional vectors with a zero vertical component. The semantic graph stores orientation as $(\cos \mathbf{r}_i, \sin \mathbf{r}_i)$.

Normalization. Horizontal positions, distances, widths, depths, and wall-segment lengths in the geometric features are normalized by the room's horizontal diagonal length, computed as the square root of the squared room width and depth. Vertical positions, displacements, and heights are normalized by the room height. Both scales are lower-bounded by 10^{-8} meters. Unit directions, orientation sine and cosine values, and dimension ratios are not normalized further. Raw semantic positions and dimensions remain in meters.

A.1 Semantic Graph Nodes and Edges

At timestep t, the semantic graph $\mathcal{G}_{\text{sem}}^{(t)}$ contains furniture nodes $\mathcal{V}_f^{(t)}$, door and window nodes $\mathcal{V}_o$, and a virtual scene node V_s. Physical edges encode radius, projection overlap, and furniture–opening relations. Scene-membership edges connect every physical node to V_s.

Node attributes $\mathrm{NodeAttr}_i$. For each physical node, the raw attribute used by Algorithm 1 is

$$\big[\mathbf{c}_i,\ \mathbf{p}_i,\ \cos \mathbf{r}_i,\ \sin \mathbf{r}_i,\ \mathbf{s}_i/2\big].$$

Here, $\mathbf{c}_i$ is a one-hot category vector. The remaining entries are the three-dimensional center position, the orientation sine and cosine, and the half dimensions in width, depth, and height order. Door and window nodes use the same format with their own category vectors. The virtual scene node is initialized with a zero placeholder; the input adapter A_{sem} replaces it with a learnable scene-node representation.

Edge construction $\mathcal{E}_{\text{sem}}^{(t)}$. The relation types and their construction rules are listed below.

Edge type	Type ID	Construction rule
Radius	0	Connect two distinct furniture nodes when their horizontal center distance is at most 1.6 meters.
Scene-membership	1	Connect every furniture, door, and window node to V_s.
Projection-overlap	2	Connect two distinct furniture nodes when their horizontal axis-aligned boxes have positive IoU.
Furniture–opening	3	Connect a furniture node to a door or window when their horizontal center distance is at most 3.0 meters.

Each connection is stored in both directions. If a furniture pair satisfies more than one rule, we retain one directed edge for each relation type. No physical edges are created between opening nodes.

Horizontal projection and overlap. For overlap computation, we approximate each object's horizontal projection by an axis-aligned rectangle centered at the horizontal coordinates of $\mathbf{p}_i$. Its half extents are the width and depth entries of $\mathbf{s}_i/2$; the object orientation is not applied. We clamp each intersection length to zero, multiply the two lengths to obtain the intersection area, and divide by the union area to obtain the 2D IoU. If the union area is at most 10^{-8} square meters, the IoU is set to zero.

For projection-overlap edges, the overlap feature is this 2D IoU. For radius and furniture–opening edges, we multiply it by the ratio of the smaller object height to the larger object height. The ratio is set to zero when the larger half height is at most 10^{-8} meters. This height weighting is a 2D proxy: it does not use the vertical center positions and is not an exact 3D bounding-box IoU.

Edge attributes $\mathrm{Edge}^t_{\mathrm{sem}}$**.** For a directed edge from node j to node i, the seven edge attributes are:

Entries	Definition
1–3	The displacement $\mathbf{p}_i - \mathbf{p}_j$, ordered as x, y, z and expressed in meters.
4–5	$\cos(\mathbf{r}_i - \mathbf{r}_j)$ and $\sin(\mathbf{r}_i - \mathbf{r}_j)$ for radius and furniture–opening edges; the fixed pair $(1, 0)$ for projection-overlap edges.
6	The overlap value defined above.
7	The relation-type ID.

For the reverse physical edge, the displacement and the sine term change sign; the cosine, overlap, and relation-type ID remain unchanged. Scene-membership edges use zeros in the first six entries and type ID 1 in the last entry, in both directions. Furniture node attributes, state-dependent connectivity, and state-dependent edge attributes are recomputed at every guidance timestep.

A.2 Geometric Graph Nodes and Edges $\mathcal{G}^{(t)}_{\mathrm{geo}}$

The geometric graph $\mathcal{G}^{(t)}_{\mathrm{geo}}$ retains semantic edges whose endpoints are both physical nodes. It removes the virtual scene node and its scene-membership edges while preserving the directions and relation types of the remaining edges. Node and edge geometric features are recomputed from $U^{(t)}$. The semantic overlap value is not used as a geometric edge feature.

Node scalar features. The node scalar features combine object attributes, room scales, and semantic representations. Their components are summarized below; positions, distances, and dimensions follow the normalization in Section A.1.

Feature group	Definition and computation
Category and node type	Linear projection of the category one-hot vector and a learnable embedding of the node type: furniture, door, or window.
Vertical position and dimensions	The normalized vertical center and the full width, depth, and height.
Horizontal position	The normalized horizontal distance from the object center to the room origin.
Room scales	The reciprocals of the room's horizontal diagonal length and height.
Height-degeneracy indicator	One if the object's full height is nonpositive, and zero otherwise.
Node semantics	Layer-normalized $z^{(t)}_{\mathrm{sem},i}$ followed by a linear projection.
Scene semantics	Layer-normalized $z^{(t)}_{\mathrm{scene}}$ followed by a linear projection.

Node vector features. We initialize $\mathbf{v}^{(t)}_i$ with the normalized horizontal position of the object center and the width and depth axes defined in Section A.1. These three vectors are expressed in the room frame and have zero vertical components. The resulting scalar–vector representation is passed through the GVP input module before geometric message passing.

Edge scalar features. For a physical edge $e^{(t)}_{ij}$, node j is the source and node i is the receiving node. Messages therefore propagate along $j \to i$, with $\mathrm{edge_index} = [j, i]$. We use $\mathbf{p}_j - \mathbf{p}_i$ as the

geometric displacement, namely the position of the source relative to the receiving node.The raw scalar features are:

Feature group	Definition and computation
Local relative position	Project the horizontal component of $\mathbf{p}_j - \mathbf{p}_i$ onto the width and depth axes of node i.
Vertical difference and horizontal distance	Use the vertical coordinate difference and horizontal center distance.
Dimensions of both nodes	Include the full width, depth, and height of the receiving node and its neighbor.
Relative dimensions	Use the elementwise logarithm of the neighbor-to-receiving-node ratios after adding 10^{-3} meters to each dimension.
Relative orientation	Use $\sin(\mathbf{r}_j - \mathbf{r}_i)$ and $\cos(\mathbf{r}_j - \mathbf{r}_i)$.

The geometric part has 15 scalar features. We concatenate these normalized values with embeddings of the edge type and the node types at both endpoints. All physical edge types use the actual relative orientation, including projection-overlap edges.

Edge vector features. The raw edge vector feature $\mathbf{v}_{ij}^{(t)}$ has two channels. The first is the horizontal unit direction obtained by normalizing $\mathbf{p}_j - \mathbf{p}_i$. If its horizontal components are (x, z), the second channel is the perpendicular direction $(-z, x)$. Both channels are expressed in the room frame, have zero vertical components, and are not normalized by the room scale. When the horizontal center distance is at most 10^{-6} meters, both channels are set to zero. Reverse edges recompute the two channels using their own source and receiving nodes. The GVP edge module jointly encodes the scalar and vector edge features.

A.3 Furniture–Wall-Segment Relations

Wall-segment construction. Given the room boundary $\mathcal{B}$, we project the floor mesh onto the horizontal $x - z$ plane. We merge vertices with coincident horizontal coordinates and retain an undirected mesh edge when it occurs in exactly one floor face. The retained edges are chained into rings, and adjacent collinear edges are merged. The largest ring is oriented counterclockwise and interior rings clockwise. The resulting directed segments form

$$\mathcal{W} = \mathrm{Segments}(\mathcal{B}).$$

For each $w_\ell \in \mathcal{W}$, the vector from its start point to its end point defines the unit tangent, and its norm is the segment length. We choose the horizontal quarter-turn of the tangent that points into the room as the unit inward normal. Degenerate segments are discarded. If the floor mesh does not yield a reliable boundary ring, we use a rectangular fallback only when the ratio between the mesh floor area and the corresponding rectangular footprint area is at least 0.99; otherwise, the scene's boundary relations are marked unusable. Wall segments are not graph nodes. They enter the geometric branch only through $\mathcal{R}_{\mathrm{bnd}}^{(t)}$.

Relation coverage and closest-point computation. For the current layout $U^{(t)}$, we create one relation for every furniture node and every valid wall segment in the same room:

$$\mathcal{R}_{\mathrm{bnd}}^{(t)} = \left\{ r_{i\ell}^{(t)} \mid (i, w_\ell) \in \mathcal{V}_f^{(t)} \times \mathcal{W} \right\},$$

where

$$r_{i\ell}^{(t)} = \left(\mathbf{s}_{i\ell}^{(t)}, \mathbf{v}_{i\ell}^{(t)}, \text{segment length} \right).$$

We do not apply a distance cutoff. Doors and windows remain physical nodes in the geometric graph, but they are not targets of boundary relations. Wall geometry is fixed within a room; furniture-dependent quantities are recomputed at every guidance step.

For each furniture–wall-segment pair, we project the furniture center onto the finite segment and clamp the projection parameter to the interval between the endpoints. A projection outside the segment therefore maps to the nearer endpoint. The distance from the furniture center to this closest point is used as the wall-distance feature. For a degenerate segment, we use its start point.

Scalar features. We express the segment endpoints and closest point relative to the horizontal furniture center, then project them onto the local width and depth axes from Section A.1. Horizontal positions, distances, and lengths are normalized by the room's horizontal diagonal; furniture height is normalized by the room height. The 15 scalar features are:

Feature group	Definition
Segment positions	Local width–depth coordinates of the segment start, end, and closest point.
Distance and segment length	Horizontal distance to the closest point and the wall-segment length.
Furniture dimensions	Full width, depth, and height of the furniture instance.
Wall tangent	Projections of the wall tangent onto the local width and depth axes.
Inward normal	Projections of the inward normal onto the local width and depth axes.

We use the normalized segment length as the aggregation weight. This makes the weighted average independent of how a wall is subdivided into collinear segments. The target node type is furniture for every relation in $\mathcal{R}^{(t)}_{\mathrm{bnd}}$.

Vector features and boundary messages. Each relation contains three horizontal vectors:

1. the unit direction from the furniture center to the closest point;
2. the unit tangent of the wall segment;
3. the unit inward normal.

If the furniture center coincides with the closest point, the first vector is set to zero. The vectors remain in the room frame. The local projections of the tangent and inward normal are included in the scalar features above.

At each GVP layer, the relation encoding and the target furniture state are used to compute a boundary message. We also include the projection of the furniture vector state onto the inward normal. For furniture node i, the messages from all associated wall segments are weighted by segment length and normalized by the total segment length. In the dual-aggregation variant, segment length is also included in the attention weights. The aggregated boundary message is added to the messages from physical furniture–object edges before updating the furniture state. Nodes without valid wall relations receive a zero boundary message.

B Model Details and Training Process

We train a separate correction model for each room category using the counterfactual layout dataset described in Appendix C. Training has three stages: semantic adaptation, geometric representation pretraining, and factor branch learning. The pretrained diffusion model is kept frozen in all three stages.

Semantic adaptation. We keep the parameters of the pretrained semantic encoder G_{sem} fixed and train only the input adapter A_{sem}. To provide supervision for this adaptation, we aggregate the node-level semantic features and the scene representation, and use the result to predict the physical-conflict labels of each scene with an auxiliary multi-label classifier. Let $\hat{\mathbf{y}}_{\mathcal{S}}$ and $\mathbf{y}_{\mathcal{S}}$ denote the predicted conflict probabilities and ground-truth labels for scene $\mathcal{S}$, respectively. The objective is

$$\mathcal{L}_{\mathrm{sem}} = \mathbb{E}_{\mathcal{S}}[\mathrm{BCE}_w(\hat{\mathbf{y}}_{\mathcal{S}}, \mathbf{y}_{\mathcal{S}})]\,, \tag{11}$$

The auxiliary task maps features from the current layout into the feature space expected by the pretrained graph network. Once this stage is complete, we keep A_{sem} and G_{sem} and remove the auxiliary classifier.

Geometric representation pretraining. Next, we freeze the semantic encoder and train the geometric encoder G_{geo} together with the semantic–geometric fusion module. An auxiliary seven-dimensional readout head maps the fused representation $\mathbf{f}_i$ to position, size, and orientation corrections. Its targets are the physical corrections needed to recover the reference layout from the current state. We clip these targets componentwise and normalize them by their corresponding scales.

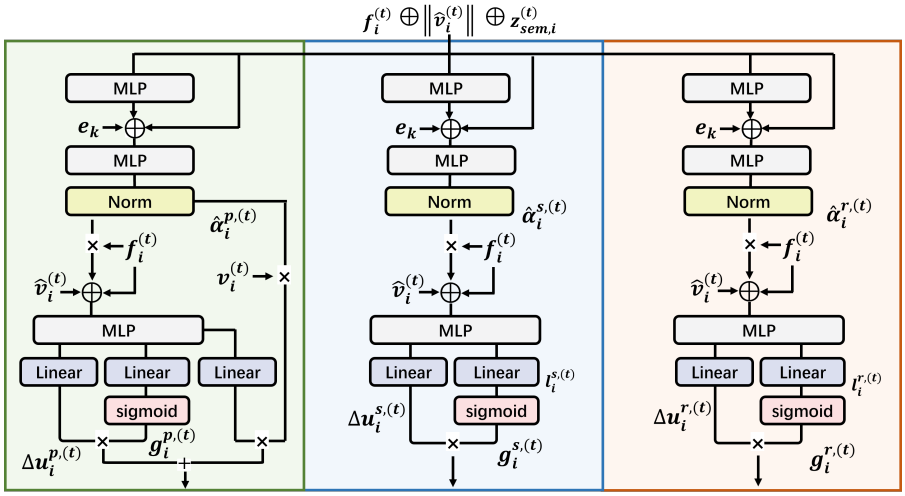

Figure 5: Factor-specific correction branches for position (left), size (middle), and orientation (right). Each branch predicts a candidate correction and a gating logit using factor-conditioned scene context and instance features.

The training objective is

$$\mathcal{L}_{\text{geo}} = \mathcal{L}_{\text{reg}} + \eta_{\text{dir}}\mathcal{L}_{\text{dir}} + \eta_{\text{mag}}\mathcal{L}_{\text{mag}} + \eta_{\text{zero}}\mathcal{L}_{\text{zero}}, \tag{12}$$

where $\mathcal{L}_{\text{reg}}$ is the SmoothL1 loss on the correction vectors, $\mathcal{L}_{\text{dir}}$ is the cosine distance between predicted and target directions, and $\mathcal{L}_{\text{mag}}$ is the SmoothL1 loss between their magnitudes. The term $\mathcal{L}_{\text{zero}}$ penalizes the ℓ_1 norm of predicted corrections and therefore discourages changes to instances that do not require them. The coefficients η_{dir}, η_{mag}, and η_{zero} are the corresponding loss weights.

We compute all four terms in normalized space. The first three terms supervise modified furniture instances, whereas $\mathcal{L}_{\text{zero}}$ is applied to unmodified instances. We first average each loss within a scene and then average across scenes, so scenes with more furniture do not receive greater weight during optimization. After pretraining, we discard the auxiliary seven-dimensional readout head and pass the learned fused features to the three factor branches.

Factor branch learning. In the final stage, we freeze the semantic encoder, geometric encoder, and fusion module, and jointly train the position, size, and orientation branches shown in Fig. 5. For each factor $k \in \{P, S, R\}$, we use the gating logit l_i^k and candidate correction $\Delta\mathbf{u}_i^k$ defined in the main text, and denote the target correction by $\Delta\mathbf{u}_i^{k,*}$.

The factor label $m_i^k \in \{0, 1\}$ is computed from the ground-truth correction as described in Appendix C. Because the labels are defined independently for the three factors, more than one factor can be active for the same furniture instance. The objective is

$$\mathcal{L}_{\text{factor}} = \sum_{k\in\{P,S,R\}} \left[\mathbb{E}_i \ \text{BCE}\big(\sigma(l_i^k), m_i^k\big) + \mathbb{E}_{i:m_i^k=1}\, \ell_{\text{reg}}\Big(\Delta\mathbf{u}_i^k, \Delta\mathbf{u}_i^{k,*}\Big)\right], \tag{13}$$

where σ is the sigmoid function and ℓ_{reg} is the SmoothL1 loss after predictions and targets have been normalized by the same componentwise scales.

The first expectation averages the selection loss over all furniture instances. The second averages the correction loss only over instances for which factor k is active. Thus, every instance contributes to selection supervision, while regression is applied only when a correction is required. We do not apply the predicted binary gates during training; the regression branch therefore continues to receive supervision even when the selection branch makes an error. All three branches are optimized jointly with the objective above.

C Details of the Counterfactual Perturbation Scene Dataset

C.1 Original Scenes

We build the correction dataset from preprocessed 3D-FRONT scenes. The original furniture layout of each scene serves as the recovery target. The initial collection contains 4,041 bedroom, 900 dining-room, and 813 living-room scenes. During perturbation, the room structure remains fixed; only the geometry of selected furniture instances is changed.

Before generating perturbations, we filter scenes with substantial physical violations. We sum four scores in $[0, 1]$: collision, boundary violation, traffic obstruction, and floating. Lower values indicate fewer violations. Collision measures relative penetration between furniture boxes. Boundary violation combines the fraction of footprint corners outside the room with their distance from the boundary. Traffic obstruction measures path elongation and insufficient clearance, while floating measures floor clearance relative to object height for furniture expected to rest on the floor. For collision, boundary, and floating, we use the maximum score over eligible pairs or objects. When a traffic score is missing, we replace it with the mean of the available scores in the same room category. We retain scenes satisfying $s_{\text{phys}} \leq 0.60$ and containing at least two perturbable furniture instances. This filtering leaves 1,988 bedroom, 608 dining-room, and 545 living-room scenes.

C.2 Perturbation Generation

We generate perturbed layouts by translating, rotating, and changing the dimensions of selected furniture. These operations target collisions, boundary violations, traffic obstruction, size deviations, and floating. For example, we translate furniture toward nearby objects, room boundaries, or circulation areas; resize it by scaling or stretching; and lift it above the floor. We apply these operations separately or combine them on the same object to produce samples with multiple correction factors.

We balance generation attempts across perturbation groups involving position, size, rotation, and their combinations. Four severity levels control both the perturbation magnitude and the target number of affected objects. Within one sample, all selected objects share the same combination of perturbed attributes. Each original scene receives 16 generation attempts. We discard failed attempts and samples that do not satisfy the prescribed attribute and magnitude constraints.

To supervise iterative correction at different residual magnitudes, we interpolate between each perturbed layout and its original layout. The interpolation produces states with progressively smaller residuals and samples more densely near the target. We interpolate positions and dimensions linearly, and interpolate orientations along the shortest angular difference. The resulting intermediate states supplement the initial perturbations with small corrections near the reference layout, so the model sees both large adjustments and fine refinements during training.

C.3 Supervision Labels

The labels describe how to restore the current perturbed layout to its original state. They therefore represent the required correction rather than the perturbation operator that produced the sample. For each furniture instance, we compute an inverse correction vector $g = [\Delta x, \Delta y, \Delta z, \Delta w, \Delta d, \Delta h, \Delta r]$ from the difference between its perturbed and original states. The first three components correct position, the next three correct the full dimensions along the three axes, and the last component corrects rotation using the shortest angular difference. An unperturbed furniture instance has a zero correction target. When several operations affect the same instance, we accumulate their corresponding corrections so that the label describes the final geometric state.

For each furniture instance, we assign three binary labels for position, size, and rotation. After normalizing the correction components by their respective physical scales, we set $m_P = 1$ when any position component $(\Delta x, \Delta y, \Delta z)$ has absolute value greater than a small threshold ϵ. We set $m_S = 1$ when any size component $(\Delta w, \Delta d, \Delta h)$ exceeds ϵ, and set $m_R = 1$ when $|\Delta r| > \epsilon$. Each label is zero otherwise. The binary labels identify the active correction factors, while the continuous targets specify their magnitudes.

During training, we first normalize the seven correction components by their respective physical scales. Each factor then uses two branches: a selection branch predicts whether the factor is active

Table 4: Statistics of the generated correction dataset. Intermediate states are obtained by interpolating between the initial perturbed layouts and their original layouts.

Room type	Original scenes	Initial perturbations	Intermediate states	Total samples
Bedroom	1,988	29,503	118,010	147,513
Dining room	608	8,433	29,907	38,340
Living room	545	7,858	27,932	35,790

for a furniture instance, and a magnitude regression branch predicts the continuous correction when its label is 1. We sum the losses for position, size, and rotation and do not apply hard gating during training. This keeps the regression branch supervised even when the selection branch is incorrect, and avoids forcing inactive factors to regress toward an arbitrary prescribed value. Thus, each furniture instance is supervised by three factor-selection labels and their corresponding conditional correction targets. The representation supports instances that require several factors at once as well as instances that require no correction.

C.4 Dataset Statistics and Splits

We generate 221,643 samples across the three room categories, including initial perturbed layouts and their intermediate states. Table 4 reports the complete dataset. When room-boundary features are used, scenes whose floor-plan boundaries cannot be reconstructed reliably are excluded from training.

We split data at the source-scene level using a fixed random split. For each room category, 85% of scenes are assigned to training and 15% to validation; The resulting training/validation scene counts are 1,690/298 for bedrooms, 517/91 for dining rooms, and 464/81 for living rooms. All initial perturbations and intermediate states derived from one source scene remain in the same subset. Consequently, the training and validation sets do not share source scenes within a room category, and validation measures performance on layouts derived from held-out scenes.

C.5 Furniture–Wall-Segment Relations

We construct furniture–wall-segment relations as described in Appendix A. These relations are used only for scenes whose floor-plan boundaries can be reconstructed reliably.

The relation set contains every furniture–wall-segment pair and does not use an additional distance threshold. Door and window nodes are not targets of these relations. For each pair, we compute the closest point on the finite wall segment and use its distance from the furniture center as the wall-distance feature. We then express the segment endpoints and the closest point in the furniture's local width–depth coordinate system.