Appendix

Synthesis of Interactive and Expansive Apartment Environments

A. Discussion

We provide a critical analysis of the current limitations of our framework and discuss the broader societal implications of our work below.

A.1. Limitation

Despite the demonstrated efficacy of our proposed method in generating functionally viable and apartment-scale environments, several limitations remain to be addressed.

Inference Latency. A primary constraint is the computational cost associated with the multi-stage pipeline. While the LLM-guided floor plan generation is relatively efficient, the core diffusion-based furniture population relies on iterative denoising steps. Coupled with the gradient calculations required for our novel constraints, the inference time for a full apartment scale is considerable. This currently precludes the system from real-time generation applications.

Heuristic Approximation of Articulation. Our Articulated Object Collision Constraint relies on a heuristic expansion of bounding boxes to approximate the kinematic workspace of objects. While robust for standard furniture morphologies found in GAPartNet, this axis-aligned expansion simplifies the complex, potentially non-linear trajectories of certain articulated parts. Consequently, for highly complex mechanisms or multi-jointed objects, the collision avoidance might be overly conservative or, in rare cases, insufficient.

Dataset Bias and Generalization. The stylistic and semantic diversity of our generated scenes is inherently bounded by the underlying training data, specifically 3D-FRONT and GAPartNet. While these datasets are extensive, they may not fully encompass the architectural styles of different cultures or historical periods. As with all learning-based generative models, the system may exhibit biases present in the dataset, potentially favoring modern, Western-style interior layouts over others.

A.2. Social Impact

The primary societal contribution of this work lies in its potential to accelerate the development of embodied AI and service robotics. By automating the synthesis of large-scale, functionally sound training environments, our proposed method significantly reduces the reliance on costly and labor-intensive real-world data collection. This democratization of high-quality simulation data can foster innovation in domestic robotics, potentially leading to deploying intelligent agents capable of assisting the elderly or individuals with disabilities in their daily lives.

From an environmental perspective, while the training of large diffusion models incurs a carbon footprint, the ability to train robots in simulation (Sim-to-Real) drastically reduces the energy consumption, material waste, and physical risks associated with trial-and-error learning in the physical world. Regarding ethical considerations, unlike generative models for faces or media, the generation of indoor scene layouts carries a relatively low risk of malicious misuse. However, we advocate for continued awareness regarding the cultural biases embedded in synthetic datasets to ensure the inclusivity of future AI technologies.

B. Preliminaries

Our generative framework is built upon Denoising Diffusion Probabilistic Models (DDPMs) [3]. DDPMs are a class of latent variable models designed to learn a data distribution $p(\mathbf{x})$ by reversing a gradual noising process. In this section, we briefly review the mathematical formulation of DDPMs, including the forward diffusion process, the reverse denoising process, and the training objective.

B.1. Denoising Diffusion Probabilistic Models

Forward Diffusion Process. The forward process, also known as the diffusion process, is a fixed Markov chain that gradually adds Gaussian noise to the data $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ over a sequence of timesteps $t = 1, \ldots, T$. The transition probability at each step is defined as:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}), \quad (1)$$

where $\{\beta_t \in (0,1)\}_{t=1}^T$ is a pre-defined variance schedule. As $T \to \infty$, the data $\mathbf{x}_T$ approaches an isotropic Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$. A key property of this process is that we can sample $\mathbf{x}_t$ at any arbitrary timestep t directly from $\mathbf{x}_0$ in closed form:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}), \quad (2)$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$. This allows us to express $\mathbf{x}_t$ as a linear combination of the original data and noise:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \quad (3)$$

Reverse Denoising Process. The goal of the generative model is to reverse this process, sampling from $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ to reconstruct the data. Since the exact posterior is intractable, we approximate it using a learnable Markov chain with parameterized Gaussian transitions:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)), \quad (4)$$

starting from $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The mean $\boldsymbol{\mu}_\theta$ and covariance $\boldsymbol{\Sigma}_\theta$ are predicted by neural networks. Following [3], we set $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2\mathbf{I}$, where σ_t^2 is set to β_t or $\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$. The mean is parameterized to predict the noise $\boldsymbol{\epsilon}$ added to $\mathbf{x}_0$:

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right). \quad (5)$$

Optimization Objective. The model is trained by optimizing the variational lower bound on the negative log-likelihood. Ho et al. [3] demonstrated that a simplified objective yields better sample quality. This simplified loss calculates the mean squared error between the true noise $\boldsymbol{\epsilon}$ and the predicted noise $\boldsymbol{\epsilon}_\theta$:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|^2\right], \quad (6)$$

where t is uniformly sampled from $\{1, \ldots, T\}$. In our framework, we adapt this backbone to generate 3D scene layouts by conditioning the denoiser on floor plan constraints.

C. Implementation Details

Experimental Setting. We evaluate our method on the 3D-FRONT dataset, employing the official train/test splits to ensure consistency with prior work. For articulation-aware generation, we augment the object assets using the GAPartNet dataset, which provides part-level annotations. To verify the robustness of our method, all baselines are retrained on this identical data subset. We generate 1,000 scenes for each experimental condition to compute reliable metrics.

Evaluation Metrics. To quantitatively evaluate the quality, diversity, and semantic coherence of our generated scenes, we employ a comprehensive suite of metrics. *Standard Perceptual & Semantic Metrics:*

- **Fréchet Inception Distance (FID) [2]:** Measures the distributional distance between deep features of generated (μ_g, Σ_g) and real (μ_r, Σ_r) scene renderings:

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}) \quad (7)$$

- **Kernel Inception Distance (KID) [1]:** An unbiased estimator of the Maximum Mean Discrepancy (MMD) between feature representations, suitable for smaller sample sizes:

$$\begin{aligned}\text{KID} =& \text{MMD}^2(P_r, P_g) \\ =& \mathbb{E}[k(x, x')] + \mathbb{E}[k(y, y')] - 2\mathbb{E}[k(x, y)]\end{aligned} \quad (8)$$

- **Scene-Class Alignment (SCA) [4]:** Evaluates semantic consistency by calculating the classification accuracy of a pre-trained scene classifier C on generated layouts x_{gen}:

$$\text{SCA} = \mathbb{E}_{x_{gen}}[\mathbb{I}(C(x_{gen}) = y_{label})] \quad (9)$$

- **Category KL divergence (CKL) [4]:** Measures the divergence between the object category distribution of the generated set (P_g) and the ground truth (P_r):

$$\text{CKL} = D_{KL}(P_g||P_r) = \sum_i P_g(i) \log \frac{P_g(i)}{P_r(i)} \tag{10}$$

Proposed Controllability Metrics (Ours):

- **LLM-Guided Layout Metric:** Evaluates the structural and semantic fidelity of the generated floor plan graph against the ground truth by assessing node matching, edge connectivity, and constraint satisfaction.
- **Object Quantity Control Metric:** Defines the success rate of generating scenes that contain the exact target number of objects specified by the user.
- **Articulation Collision Ratio:** Measures the percentage of articulated objects (e.g., cabinets) that are functionally obstructed by other objects when in their open/extended state.
- **Walkable Area Controllability:** Calculates the success rate of generating scenes where the ratio of unobstructed walkable floor area meets or exceeds a specified threshold.

C.1. Compare Model Settings

We benchmark against three state-of-the-art baselines:

- **ATISS:** An autoregressive transformer model that places objects sequentially. We use the official implementation, retraining it on our dataset split for fair comparison.
- **DiffuScene:** A diffusion-based model that generates scene layouts in parallel. This represents the current state-of-the-art in unconditional layout generation.
- **PhyScene:** A recent physics-aware generative model. We compare against PhyScene to highlight the advantages of our specific articulated object constraints.

All models receive the same floor plan input during the conditional generation tasks.

C.2. Training Details

The model is trained using the Adam optimizer with a learning rate of 2×10^{-4} and weight decay of 0.0. We employ a step learning rate schedule with a step size of 20,000 and a decay factor of 0.5. Training runs for 130,000 epochs with a batch size of 128. The gradient norm is clipped at 10. Table 1 summarizes the complete hyperparameter configuration.

Table 1. Detailed hyperparameter settings for training the diffusion model.

Configuration	Value
Optimizer	Adam
Base Learning Rate	2×10^{-4}
Weight Decay	0.0
Batch Size	128
Max Gradient Norm	10
Learning Rate Schedule	Step Decay
LR Step Size	20,000
LR Decay Factor (γ)	0.5
Total Epochs	130,000

C.3. Computing Resource Configuration

All model training and evaluation were conducted on a computing node equipped with a single **NVIDIA 3090 GPU (24GB VRAM)** and an **Intel Core i9-12900K**. Under this configuration, training the core diffusion model takes approximately 1500 hours. During inference, generating a complete apartment-scale scene (3 rooms) with full constraint guidance takes approximately 300 seconds per scene.

D. Additional Experiments

D.1. LLM Controllability

To rigorously evaluate the fidelity of our LLM-based parameter space guidance, we designed a comprehensive benchmark consisting of 20 distinct natural language prompts.

Overall Performance. We define a generation as successful only if the final layout strictly adheres

to all constraints specified in the prompt, as shown in Table 2.

Table 2. Summary of LLM Controllability Experiments.

Metric	Value
Total Test Cases	20
Average Score	96.5%

Component Analysis. To understand the specific challenges in layout control, we decompose the performance into three sub-metrics. The score is composed of weights in the Table 3.

- **Room Presence:** Existence of required room types.
- **Adjacency:** Correct connectivity between rooms.
- **Constraints:** Geometric or functional requirements.

Table 3. Weights indicate the relative importance assigned to each component in the overall score.

Component	Avg. Score	Weight
Room Presence	98.9%	50%
Adjacency	92.3%	30%
Constraints	95.4%	20%

Detailed Test Results We provide a granular breakdown of each test case in Table 4. In these cases, the system maintains high scores on connectivity and functional constraints, ensuring the generated layouts remain usable.

E. Render Results

We present an extensive qualitative evaluation of the 3D indoor layouts generated by our proposed method. While quantitative metrics provide numerical evidence of our model's performance, visual inspection is equally crucial for assessing the perceptual quality, spatial coherence, and practical usability of the synthesized scenes. To this end, we provide a comprehensive gallery of results across a diverse range of scene categories, demonstrating the robustness of our approach in handling complex room geometries.

To guarantee the highest visual fidelity, all visualizations were produced using the **Blender** creation suite, leveraging its advanced **Cycles** rendering engine. Cycles is a production-grade, physically-based path tracer that excels at simulating the intricate interactions of light transport. Unlike real-time rasterization engines, Cycles calculates global illumination, multi-bounce indirect lighting, and accurate soft shadows, which are essential for verifying that objects are properly grounded and not floating. Furthermore, we utilized high-resolution Physically Based Rendering textures and materials to enhance the realism of the furniture, allowing for a rigorous assessment of the layout's aesthetic quality.

The rendering pipeline imposes significant computational demands, particularly when processing scenes with high-poly assets and complex lighting setups. Consequently, all rendering tasks were executed on a dedicated workstation equipped with an **Intel Core i3-14100F CPU** and an **NVIDIA RTX 5090 GPU**. The massive 32GB VRAM of the RTX 5090 proved instrumental in loading large-scale scene data and high-resolution textures without memory bottlenecks, while the CPU efficiently managed scene graph traversal and asset loading.

These visualizations highlight the model's capability to handle complex spatial arrangements naturally. We observe that the generated objects are physically plausible, exhibiting proper orientations and avoiding inter-object collisions, as shown in Figure 1 and Figure 2. Collectively, these qualitative results validate that our method not only adheres to rigid geometric constraints but also produces aesthetically pleasing and functionally realistic environments suitable for practical design applications.

Table 4. Complete Test Results for LLM Controllability. This table details the performance of 20 distinct test prompts.

Test ID	Score	Time(s)	Room Presence	Adjacency	Constraints
two_bedroom_apt_01	100.0%	262.2	100%	100%	100%
open_plan_loft_01	100.0%	274.7	100%	100%	100%
family_home_01	100.0%	276.7	100%	100%	100%
master_suite_home_01	100.0%	265.3	100%	100%	100%
four_bedroom_house_01	100.0%	270.8	100%	100%	100%
guest_suite_home_01	100.0%	282.2	100%	100%	100%
dual_master_suite_01	100.0%	269.5	100%	100%	100%
balcony_apartment_01	100.0%	285.2	100%	100%	100%
multigenerational_home_01	100.0%	289.7	100%	100%	100%
basic_studio_01	100.0%	259.0	100%	100%	100%
one_bedroom_apt_01	100.0%	270.0	100%	100%	100%
compact_efficiency_01	100.0%	269.7	100%	100%	100%
student_apartment_01	100.0%	266.0	100%	100%	100%
single_floor_accessible_01	100.0%	262.5	100%	100%	100%
entertainment_home_01	95.8%	274.9	100%	100%	79.2%
work_from_home_01	90.0%	263.0	80.0%	100%	100%
luxury_penthouse_01	88.8%	270.8	87.5%	100%	75.0%
three_bed_townhouse_01	85.0%	268.3	100%	50%	100%
separated_zones_01	85.0%	285.7	100%	50.0%	100%
home_office_layout_01	62.5%	267.8	25.0%	100%	100%

Figure 1. **Generated 3D Layouts.** Representative visualization of scenes generated by our proposed framework. (Part 1).

Figure 2. **Generated 3D Layouts.** Representative visualization of scenes generated by our proposed framework. (Part 2).

References

[1] Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans, 2021. 2

[2] 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, 2018. 2

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. 1, 2

[4] Jiapeng Tang, Yinyu Nie, Lev Markhasin, Angela Dai, Justus Thies, and Matthias Nießner. Diffuscene: Denoising diffusion models for generative indoor scene synthesis, 2024. 2, 3