iARCS: Iterative Agentic RL for Controllable 3D Scene Generation

Saugat Adhikari[1]* Ashok Prasad Neupane[1]* Pramish Paudel[3]* Ajad Chhatkuli[2,3]
Danda Pani Paudel[2,3]

[1]Pulchowk Campus, IOE, Tribhuvan University, Lalitpur, Nepal
[2]NAAMII, Kathmandu, Nepal
[3]INSAIT, Sofia University "St. Kliment Ohridski", Bulgaria
*Equal contribution

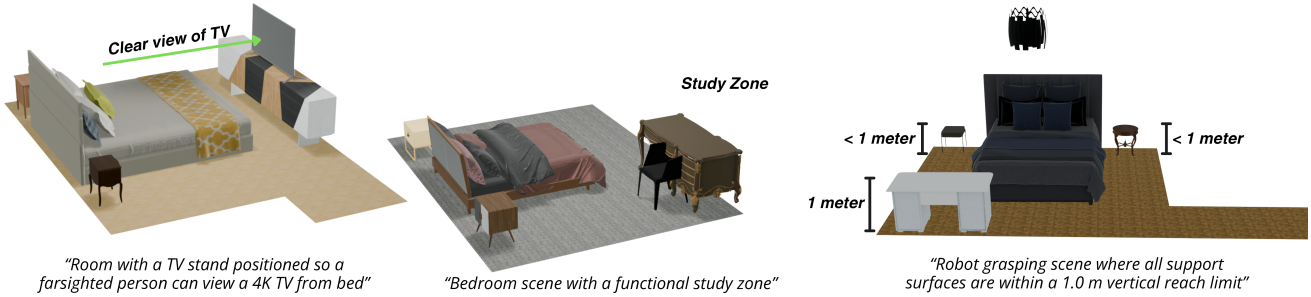

Figure 1. Qualitative results of iARCS under different reward-function specifications. Across tasks, the generated scenes follow the intended reward-driven constraints (e.g., spatial arrangement and functional objectives) while preserving overall scene quality, including physical plausibility and walkable structure.

Abstract

Synthetic 3D scene generation is increasingly used as a data source for computer vision and embodied AI, but existing generators often optimize perceptual realism without reliably satisfying task-critical functional constraints. This mismatch limits the usefulness of synthetic data for downstream training, where accessibility, traversability, and spatial rule compliance are often essential. We present iARCS, an iterative agentic reinforcement learning framework that adapts a pretrained scene generator to natural-language task requirements. iARCS uses a two-stage strategy: universal-reward pretraining to improve physical plausibility and layout quality, followed by task-specific fine-tuning with LLM-generated reward programs that are iteratively refined from training feedback. Experiments show improved constraint fidelity on walkability, reachability, and clearance-focused tasks, strong task-conditioned generalization, and competitive scene diversity. We further show that data generated by iARCS improves a base generator, supporting its value as a practical synthetic data generation tool rather than only a controllable scene editing method.

1. Introduction

Synthetic 3D indoor scene generation is increasingly important for computer vision and embodied AI, where large-scale and diverse training data are essential but expensive to collect and annotate in the real world [5, 10, 18, 31, 33]. Recent generative models [13, 16, 20, 24, 30] have significantly improved perceptual realism and distribution matching in scene generation [15, 23, 34]. Yet, many downstream applications require scenes that satisfy *functional* constraints beyond semantic plausibility [31, 33, 38], such as accessibility-aware clearance, controllable traversability for robot testing, and explicit spatial rules over object size, count, and relative placement.

This mismatch creates a practical data bottleneck: benchmark datasets and unconstrained generators rarely provide enough task-critical edge cases [10, 15, 23, 34, 38].

Consequently, generated training data can appear realistic while remaining weakly aligned with downstream functional utility.

Current approaches for controllable 3D scene synthesis fall into three lines: procedural generation [1, 4–6, 26, 28, 36, 39], LLM-augmented construction [21, 40] and data-driven generative modeling [15, 19, 23, 34, 41]. Procedural methods provide scalable variation [5, 28], but their hand-crafted rules make fine-grained task semantics hard to specify and verify consistently. LLM-based pipelines improve prompt-level flexibility by converting language into scene graphs or editing constraints [19, 39], yet reliability still depends on LLM scene composition ability, with weak guarantees under complex multi-constraint settings [19, 39]. Data-driven models such as ATISS [23], DiffuScene [34], MiDiffusion [15], and PhyScene [38] match training distributions and improve plausibility, but still struggle to satisfy user-defined functional constraints.

Text conditioning alone is usually insufficient for strict functional scene requirements: language-guided pipelines improve instruction following, but they do not reliably enforce precise geometric constraints and typically lack explicit verification during generation [19, 39]. Reward-guided optimization improves controllability, yet existing practice still depends on manually engineered objectives and substantial reward-design effort, which limits clean scaling to arbitrary new constraints [2, 22, 25, 38].

We formulate synthetic scene generation as the post-training adaptation of a pretrained scene prior to satisfy user-defined functional constraints while preserving realism and diversity. This view aligns with recent post-training and RL-based control of generative models, which optimize task-specific objectives on top of pretrained diffusion priors rather than rebuilding generators from scratch [2, 25]. In order to realize this goal, we propose a two-stage agentic reinforcement learning (RL) framework for post-training scene generators called iARCS. In **Stage 1**, we optimize universal rewards (e.g., physical plausibility and design priors) to strengthen baseline scene quality, and in **Stage 2**, an LLM agent translates user prompts into task-specific reward programs and iteratively refines them using training feedback. We fine-tune the diffusion generator with Denoising Diffusion Policy Optimization (DDPO) [2] to optimize non-differentiable functional objectives (e.g., walkability, reachability, and clearance) while regularizing against distribution collapse.

This design addresses both failure modes: it retains the realism and diversity of learned scene distributions while enabling explicit enforcement of functional constraints through reward-driven adaptation. Empirically, the two-stage optimization schedule is substantially more effective than single-stage training: first optimizing universal rewards, then fine-tuning with task-specific and universal rewards jointly, yields stronger physical and functional performance. As summarized in Fig. 2, we find that iARCS functions as an effective synthetic-data engine: training MiDiffusion with iARCS-generated synthetic data improves a strong state-of-the-art indoor scene generator while preserving diversity and realism.

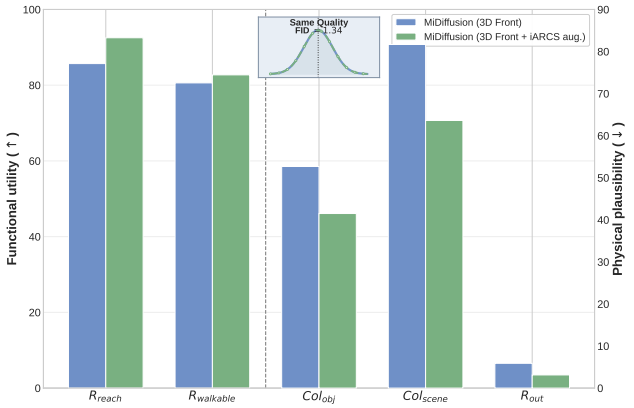

Figure 2. **MiDiffusion trained with iARCS augmentation vs. MiDiffusion trained on 3D-FRONT only.** We compare MiDiffusion trained on 3D-FRONT against MiDiffusion trained on 3D-FRONT + iARCS-generated synthetic data. The augmented setting improves functional utility (R_{reach}, $R_{walkable}$) and physical plausibility (Col_{obj}, Col_{scene}, R_{out}) across all reported metrics, while matched FID (1.34) indicates preserved generative quality and diversity.

Our contributions are as follows:

- We propose iARCS, an iterative agentic RL framework for controllable 3D scene generation from natural-language constraints.
- We introduce a two-stage training strategy (universal-reward pretraining + task-specific fine-tuning) that improves physical plausibility and functional utility.
- We show strong task-conditioned generalization and that iARCS-generated data improves a base generator while maintaining competitive diversity.

2. Related Works

Generative Priors for 3D Scenes. Recent approaches to 3D indoor scene synthesis leverage deep generative models to capture complex geometric and semantic distributions. Methods such as ATISS [23, 37] utilize autoregressive transformers to sequence object placements, while subsequent works [15, 34, 38] employ diffusion models to map latent noise to structured layouts S. These architectures typically learn a distribution $p(S|F)$ conditioned on a floor boundary F. However, downstream applications often require scenes to satisfy specific functional or spatial rules C [1, 3, 8, 19, 38, 39]. Sampling from the resulting constrained posterior $p(S|F, C)$ is non-trivial because these

functional dependencies are rarely represented in the training data [3, 6, 8, 37, 41]. Naive approaches like rejection sampling from the prior $p(S|F)$ are computationally inefficient and often fail to produce diverse scenes [7, 25]. Consequently, explicit guidance or optimization is required to shift the generative prior toward these narrow, task-critical regions of the scene space [2, 12, 29, 38].

Guidance and Constraint Satisfaction in Diffusion. To enforce specific constraints during generation, prior literature has explored various inference-time and train-time guidance mechanisms. At inference time, Classifier-Free Guidance (CFG) [12] is standard for aligning outputs with text embeddings, while methods like PhyScene [38] apply training-time and test-time gradient guidance utilizing differentiable guidance functions to enforce physical rules; recent text-driven layout/scene synthesis methods further highlight the need for stronger controllability under natural-language constraints [1, 19, 39]. Other approaches [25, 29] rely on inference-time scaling; for instance, Steerable Scene Generation [25] employs Monte Carlo Tree Search (MCTS) to navigate the generative trajectory toward valid states. However, inference-time guidance faces fundamental limitations: gradient-based methods strictly require all user-defined constraints to be differentiable, and search-based methods incur prohibitive latency. In order to bypass these issues, train-time optimization via reinforcement learning has emerged as an alternative, utilizing policy optimization [2, 7, 25, 42] to fine-tune the diffusion model directly toward specific, non-differentiable objectives.

Agentic RL Formulation. Applying reinforcement learning to diffusion models requires robust, dense reward signals. Traditional constraint-satisfaction pipelines rely heavily on hand-crafted, mathematically rigid reward functions, which demand significant engineering effort and fail to scale across arbitrary, user-specified spatial rules. Recently, Eureka [22] used a Large Language Model (LLM) as an automated reward generator within an agentic loop, writing executable reward programs that surpass human-engineered functions. While Eureka [22] focuses on iterative curriculum learning for robotic control policies in physical simulators, iARCS adapts this paradigm to the generative space. By transitioning from rigid manual objectives to dynamic, LLM-generated reward programs, our two-stage RL framework enables the post-training adaptation of diffusion priors to complex, non-differentiable functional constraints [2, 22].

3. Method

Our method is a two-stage framework. In Stage 1, we use reinforcement learning (RL) to correct biases in a diffusion model trained on the base data distribution by optimizing for physical plausibility (collision avoidance and in-bound placement) and functional utility (walkability and reachability). In Stage 2, given a user prompt, an LLM module generates task-specific reward functions on top of Stage 1 constraints, and the model is further optimized through iterative reward reflection. In our experiments, we use Gemini[35] model as LLM function generator and perform reward reflections every 10 epochs of RL training loop.

3.1. Problem Formulation

Scene Representation. Following prior work [15, 23], we define a 3D scene as an unordered set of N objects, $S = \{o_1, o_2, \ldots, o_N\}$. Each object o_i is parameterized by its continuous geometric attributes and discrete semantic category:

$$o_i = (t_i, s_i, \cos(\theta_i), \sin(\theta_i), c_i) \tag{1}$$

where $t_i \in \mathbb{R}^3$ is the centroid translation, $s_i \in \mathbb{R}^3$ represents the 3D dimensions (width, height, depth), $\theta_i \in [0, 2\pi)$ is the orientation around the vertical axis, and $c_i \in \mathbb{R}^K$ is the one-hot encoded object category for K classes.

Let F denote a given floor boundary represented as image feature [23] and C be a user-specified functional constraint expressed in natural language. We treat the 3D scene generation process as a policy $\pi_\theta(x|F)$ parameterized by θ, which generates a scene S conditioned on the floor plan F. Our primary objective is to find the optimal model parameters θ' that maximize the expected composite reward:

$$\max_\theta \mathbb{E}_{x\sim\pi_\theta(x|F)}[R_{\text{total}}(S, C, F)] \tag{2}$$

where the total reward R_{total} evaluates the fundamental physical plausibility and functionality of the generated scene and its adherence to the user's high-level functional constraint C.

3.2. Diffusion Formulation and Policy Optimization

We model scene generation with a conditional diffusion process [13, 30]. Given clean scene parameters x_0 and condition F, the forward noising process is

$$q(x_t \mid x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I), \quad t = 1, \ldots, T, \tag{3}$$

and the learned reverse policy π_θ denoises

$$\pi_\theta(x_{t-1} \mid x_t, F) = \mathcal{N}(\mu_\theta(x_t, t, F), \Sigma_t). \tag{4}$$

To adapt this pretrained prior to functional constraints, we treat denoising as a finite-horizon Markov Decision Process and optimize with DDPO [2]. For terminal reward $R_{\text{total}}(x_0, C, F)$, the objective is

$$\max_\theta \; J(\theta) = \mathbb{E}_{x_{0:T}\sim\pi_\theta(\cdot|F)}\left[R_{\text{total}}(x_0, C, F)\right], \tag{5}$$

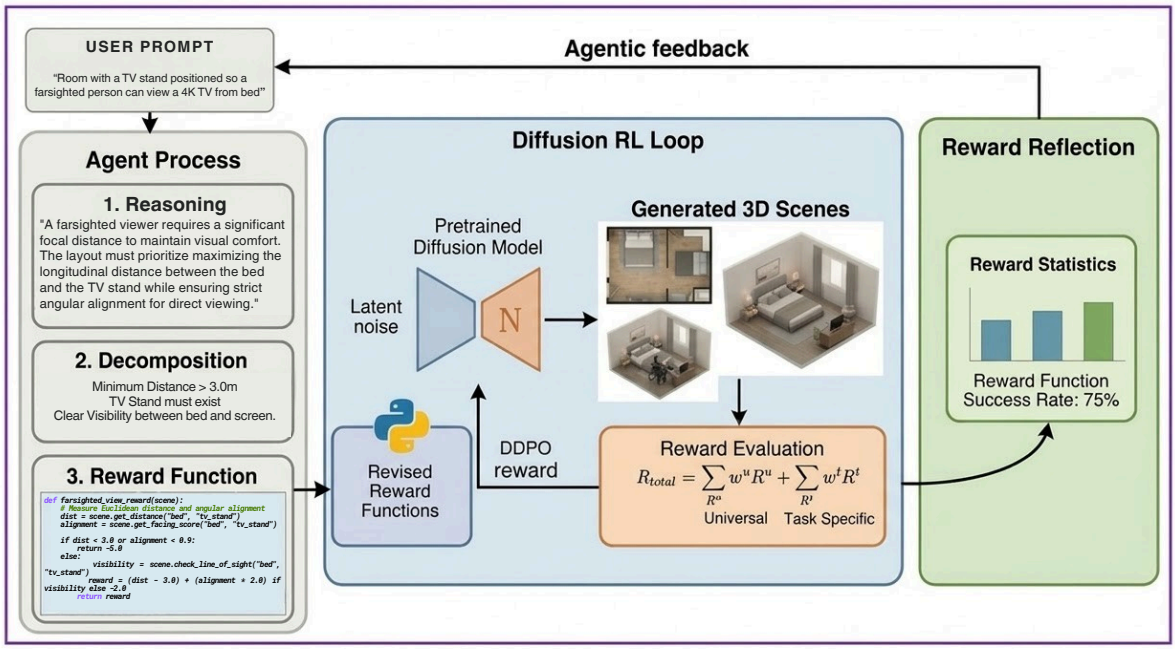

Figure 3. **iARCS overview: agentic reward synthesis with diffusion RL and reflection.** Given a user prompt, an LLM agent performs (1) reasoning, (2) constraint decomposition, and (3) executable reward-program generation. The generated task reward is combined with universal rewards to form a composite objective, which is optimized by DDPO in the diffusion RL loop to fine-tune a pretrained scene generator. A reward-reflection module monitors reward statistics from generated scenes, revises reward code/weights, and feeds updates back to the agent, enabling iterative improvement in constraint satisfaction while preserving scene realism and diversity.

with policy-gradient updates on denoising transitions. Optimizing eq. (5) requires careful regularization in order to preserve the pretrained model prior while improving constraint satisfaction without losing scene realism.

3.3. Framework Overview

In order to achieve the objective in eq. (2) with the formulation in eq. (5), without suffering from catastrophic forgetting or structural degradation, we propose the iARCS framework, as illustrated in Fig. 3. The framework operates in two distinct optimization stages:

- **Stage 1: Base Model Refinement with Universal Rewards.** Pretrained generative models often internalize dataset flaws such as object collisions. To rectify this, we first optimize the base policy π_θ using a set of universal, rule-based rewards (R^u). These enforce basic physical laws and structural integrity.
- **Stage 2: Functional Adaptation via Agentic Feedback.** Once a physically plausible baseline is established, we introduce the user constraint C. As shown in Fig. 3, an LLM agent translates the natural language prompt into an executable reward program (R^t). The model is then optimized using the composite reward R_{total}, supported by an iterative reward reflection loop to prevent reward hacking and ensure stable learning.

3.4. Agentic Reward Synthesis

Illustrated in the **Agent Process** module in our pipeline in Fig. 3, a Large Language Model (LLM) agent translates the user prompt into an executable reward program, denoted as R^t, following recent language-driven scene reasoning and programmatic control paradigms [3, 8, 22, 39]. This synthesis follows a rigorous three-step procedure:

1. **Reasoning:** The agent identifies critical, restrictive constraints and spatial dependencies implied by the prompt. For example, given the prompt "narrow bedroom with 0.2m walking path", the agent reasons: *"A 0.2m walking path is a critical, restrictive constraint. Furniture must be pushed against the walls to ensure a continuous 0.2m wide aisle from the door to the bed."*
2. **Decomposition:** Abstract constraints are broken down into atomic, measurable geometric checks. Following the previous example, this decomposes into: *Walking Space = 0.2m (Check), Furniture Alignment (against the walls) and Path Clearance (continuous aisle).*
3. **Reward Function:** The agent out-

puts executable Python code (e.g., `def robot_3DFRONT_walkability_reward(scene): ...`) that computes a scalar reward based on the generated scene and the decomposed metrics.

3.5. Optimization via Denoising Diffusion Policy Optimization

In order to optimize the pretrained diffusion model against the synthesized rewards, we execute the **Diffusion RL Loop**. We treat the diffusion sampling process as a multi-step Markov Decision Process (MDP) and employ Denoising Diffusion Policy Optimization (DDPO) [2] to update the model weights.

During the Reward Evaluation phase, the total composite reward R_{total} for each generated scene is computed as a weighted sum of the Universal (R^u) and Task-Specific (R^t) rewards:

$$R_{\text{total}} = \sum_{R^u} w^u R^u + \sum_{R^t} w^t R^t \tag{6}$$

The weights w^u and w^t are chosen by the LLM and adjusted through reward reflection, described next in Sec 3.6. By utilizing RL, we successfully optimize the generator for complex, non-differentiable geometric constraints that standard gradient-based guidance methods cannot handle.

3.6. Reward Reflection and Iterative Refinement

A key challenge in reward-guided generation is the potential for "reward hacking" or the formulation of objectives that are too sparse for the current model state. iARCS addresses this through the **Reward Reflection** module.

After a fixed number of RL iterations, the LLM inspects the **Reward Statistics** and **top-down projection images** of representative sampled scenes. The LLM may either:

- **Redefine:** Debug and simplify the revised reward functions if the underlying logic is flawed or too hard for current policy.
- **Curriculum Generation:** Decompose the reward into a simpler, intermediate objective to "warm up" the model before introducing the full constraint.

This iterative feedback loop ensures the generator progressively learns to satisfy high-level user intent without collapsing the diversity of the scene prior. The full method is summarized in Algorithm 1.

Algorithm 1: DDPO Training for 3D Scene Generation

```
for epoch = 1 to num_epochs do
    trajectories ← [];
    x_T ~ N(0, I);
    for t = T to 1 do
        ε_pred ← model(x_t, F, t);
        x_{t-1}, log p_old ← DDIM_step(ε_pred, x_t, t);
        trajectories.append((x_t, log p_old, t));
        x_t ← x_{t-1};
    x_0 ← x_t;
    r_u ← R_universal(x_0, F);
    r_t ← R_task(x_0, F);
    r_total ← w_u · r_u + w_t · r_t;
    A ← normalize(r_total);
    for inner_epoch = 1 to num_inner_epochs do
        B ← sample_random_timesteps(trajectories);
        for i ∈ B do
            x_t, log p_old, t ← trajectories[i];
            ε_new ← model(x_t, F, t);
            _, log p_new ← DDIM_step(ε_new, x_t, t);
            ρ ← exp(log p_new − log p_old);
            L ← −E[ρ · A];
            Update model parameters via ∇_θ L;
```

4. Experiments

To demonstrate our framework's viability as a scalable synthetic data generator, our experiments focus on three core questions: (1) Can our method improve physical and design constraint satisfaction while preserving the diversity of the original data distribution? (2) Does MiDiffusion trained with 3D-FRONT + our generated data outperform MiDiffusion trained with only 3D-FRONT? (3) How well does the agentic framework handle specific, task-oriented constraints?

4.1. Experimental Setup

Dataset and Base Model. We utilize 3D-FRONT [10], a widely used synthetic dataset containing 6,813 professionally designed indoor scenes. We employ the standard train/test split established in prior work [15, 23, 34]. Our base generator is a continuous domain-only MiDiffusion model [15] pretrained on 3D-FRONT [9] following the MiDiffusion architecture [15]. The generative model operates in object-parameter space, as explained in the Method section, retrieving canonical CAD models from 3D-FUTURE [11] to construct the final meshes. In all our experiments, we use bedroom scenes from the 3D-FRONT [9] dataset.

RL Finetuning. To fine-tune the pretrained diffusion model via reinforcement learning (RL), we employ Low-Rank Adaptation (LoRA) [14] with a rank (r) of 16 and a scaling factor (α) of 16. For sample rollouts during training, we use 20 DDIM steps [30] with $\eta = 1.0$. The model weights are subsequently updated using Denoising Diffusion Policy Optimization (DDPO) with importance sampling [2]. We use the Adam optimizer [17] with a learning rate of 1×10^{-5}. We use early stopping to find an optimal operating point that balances the trade-off between diversity, quality, and rewards.

Rewards. The first is a set of universally applicable rewards that improve overall physical plausibility and functional utility. These rewards are manually designed targeting collision avoidance, boundary adherence, object density regularization, and functional accessibility. Full implementation details are provided in the supplementary material. The second type is task-specific rewards generated from user prompts by our agentic pipeline. The prompts used to generate executable reward functions are also included in the supplementary material.

Baselines. We consider two state-of-the-art floor plan-conditioned scene synthesis methods as baselines: (1) ATISS [23], an autoregressive transformer for indoor scene synthesis that models sequential object placement from data distributions, and (2) MiDiffusion [15], a continuous-only variant of a mixed discrete-continuous diffusion model designed to synthesize plausible 3D indoor scenes.

Evaluation Metrics. To evaluate both realism and functional quality, we follow prior scene-generation work [15, 23, 34, 38] and report distribution metrics on 1,080 synthesized scenes: Fréchet Inception Distance (FID↓) and Scene Classification Accuracy (SCA). Together, these metrics quantify perceptual fidelity, dataset coverage, and diversity relative to the target distribution. We use CLIP-FID because our inputs are top-down projections of 3D scenes; accordingly, we compute features using CLIP embeddings [27] instead of InceptionNet class predictions [32].

In addition to distribution quality, we evaluate physical plausibility and scene functionality directly in 3D. We compute object-collision rate $Col_{obj} \downarrow$ (percentage of objects colliding with any other object) and scene-collision rate $Col_{scene} \downarrow$ (percentage of scenes containing at least one collision).

To measure floor-plan compliance, we report out-of-bound placement rate $R_{out} \downarrow$, defined as the fraction of objects placed outside room boundaries. For functional utility, we report reachable-object ratio $R_{reach}(\%) \uparrow$ from sampled valid starting locations, and walkability score $R_{walkable} \uparrow$, defined as the ratio between the area of largest connected walkable region and the total walkable area.

Our evaluation protocol prioritizes both task adherence and physical realism. We filter the test set to include only 3D-FRONT scenes that satisfy individual task constraints and apply a non-penetration constraint (threshold = -0.25) to eliminate physically inconsistent layouts. While FID is computed against the entire 3D-FRONT dataset to measure global distribution alignment, task-specific metrics are derived from a pool of 1,080 generated samples per task. The final sample sizes for the filtered datasets are provided in Table 3.

4.2. Scene Synthesis

We report quantitative results in Tab. 1. Compared with baseline methods, our approach achieves stronger physical plausibility and functional utility. In particular, our method consistently improves physical plausibility metrics and attains higher $R_{walkable}$ than the original data distribution.

Our FID remains competitive, although it is slightly lower than some baselines. We attribute this behavior to quality issues in the raw 3D-FRONT distribution (e.g., object collisions and out-of-bound placements), which can bias realism-focused scores. As reported in Tab. 1, the 3D-FRONT dataset itself contains poor-quality scenes with frequent collisions and functional issues. This observation highlights a broader limitation of relying only on FID and SCA when evaluating function-critical scene generation. Qualitative comparisons in Fig. 4 show that iARCS produces layouts with fewer collisions, better boundary adherence, and improved walkable free space than ATISS and MiDiffusion.

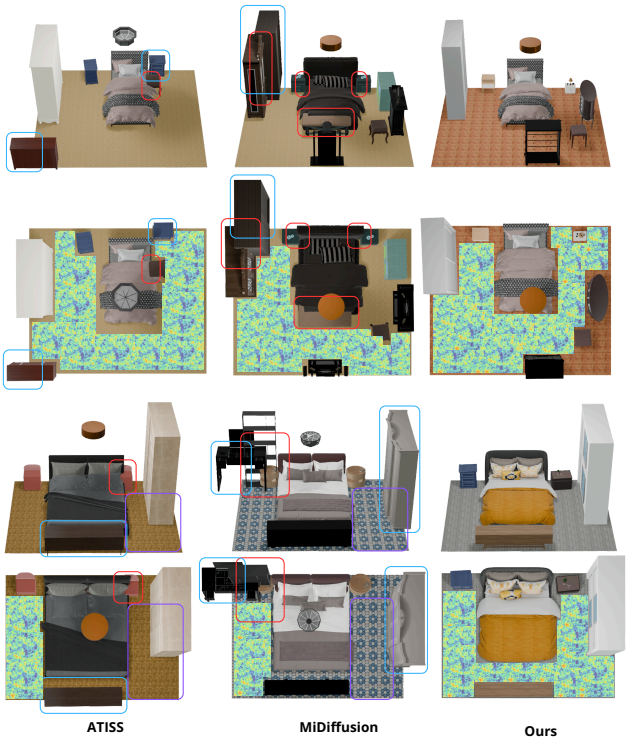

Figure 4. **Qualitative comparison of generated scenes from ATISS, MiDiffusion, and iARCS (ours).** iARCS produces more physically plausible and functionally usable layouts, with fewer object collisions, better boundary adherence, and clearer walkable regions.The red, blue, and purple boxes highlight collisions between objects, objects outside the floor plan, and unreachable areas respectively. (Better viewed zoomed.)

Table 1. Quantitative comparison of indoor scene synthesis methods on physical plausibility, functional utility, and distribution-level quality. Our method (iARCS) achieves the best physical/functional performance while maintaining competitive realism and diversity.

Method	Physical Plausibility			Functional Utility		Diversity	
	$Col_{obj}\downarrow$	$Col_{scene}\downarrow$	$R_{out}\downarrow$	$R_{reach}(\%)\uparrow$	$R_{walkable}\uparrow$	FID↓	SCA
Ground Truth (GT)	42.00%	72.04%	5.79%	82.80%	0.841	-	-
ATISS	54.12%	85.96%	7.32%	83.57%	0.780	1.39	67.23%
MiDiffusion	52.67%	81.67%	5.89%	85.7%	0.806	**1.34**	**65.99%**
iARCS (Ours)	**40.45%**	**64.63%**	**3.04%**	**87.82%**	**0.861**	1.60	68.51%

4.3. Data Augmentation Improves Base Generator

Tab. 2 compares MiDiffusion trained on the 3D-FRONT dataset with the same model trained on 3D-FRONT augmented using synthetic data generated by iARCS. Training with iARCS-augmented data yields clear improvements in both physical plausibility and functional utility while maintaining similar diversity. Notably, these gains are achieved without any external data or additional guidance, showing that the method can improve the generator through self-augmentation alone. We generate 4,000 synthetic scenes using the training-set floor layouts F from the 3D-FRONT [9] dataset and fine-tune all model weights with a learning rate of 1×10^{-5} for 200 epochs. The qualitative results in Fig. 5 are consistent with Table 2: training with iARCS-generated synthetic data improves layout feasibility and functional accessibility while preserving overall scene realism.

Table 2. Comparison of MiDiffusion trained on 3D-FRONT versus MiDiffusion trained on 3D-FRONT augmented with iARCS-generated data. The augmented setting yields consistent gains in physical plausibility and functional utility while maintaining similar diversity metrics (FID and SCA).

Method	Physical Plausibility			Functional Utility		Diversity	
	$Col_{obj}\downarrow$	$Col_{scene}\downarrow$	$R_{out}\downarrow$	$R_{reach}(\%)\uparrow$	$R_{walkable}\uparrow$	FID↓	SCA
MiDiffusion (3D-FRONT)	52.67%	81.67%	5.89%	85.7%	0.806	1.34	66.00%
MiDiffusion (3D-FRONT + iARCS aug.)	**41.49%**	**63.61%**	**3.12%**	**92.52%**	**0.8272**	1.34	66.10%

4.4. Task-Specific Constraint Generalization

Tab. 3 reports quantitative results on complex, task-specific optimization settings. Here, 3D-FRONT* denotes the subset of 3D-FRONT scenes that satisfy each task constraint. Lower FID and competitive SCA relative to 3D-FRONT* indicate that iARCS generalizes beyond the filtered subset and generates more diverse scenes while respecting task-specific constraints. This demonstrates its utility for constraint-aware synthetic data generation and augmentation.

iARCS leverages LLM generalization to interpret complex tasks, decompose them into optimizable constraints, and synthesize executable reward functions for RL-based refinement. Fig. 6 provides a qualitative example of how iARCS follows diverse task-specific reward specifications, yielding scene layouts that satisfy intended spatial and functional constraints while maintaining plausible global structure.

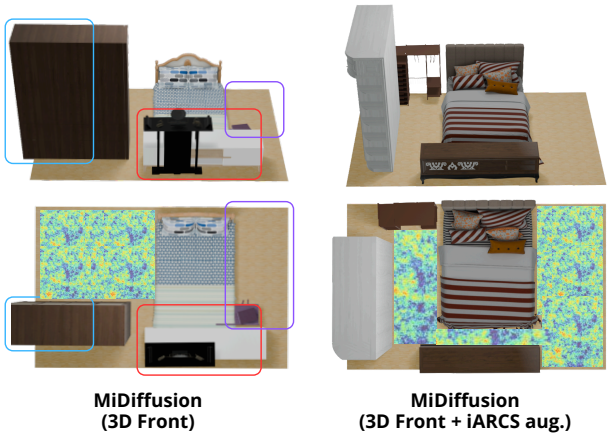

Figure 5. **Qualitative effect of synthetic-data augmentation on MiDiffusion.** We compare MiDiffusion trained on 3D-FRONT only versus MiDiffusion trained with 3D-FRONT + iARCS-generated synthetic data (Table 2). The augmented model produces layouts with improved physical plausibility and functional utility while preserving scene realism. The red, blue, and purple boxes highlight collisions between objects, objects outside the floor plan, and unreachable areas respectively. (Better viewed zoomed.)

4.5. Ablations

We ablate the effectiveness of our two-stage training pipeline in Tab. 4. Here, the task is *"Scene with TV and bed at a 3 m distance"*, and training is stopped after reaching the same reward level for a fair comparison. Directly optimizing task-specific constraints with physical plausibility, functional utility, and task-specific rewards yields worse results than our two-stage strategy: first training with universal rewards (physical plausibility + functional utility), then jointly fine-tuning with task-specific rewards.

Table 3. **Task-wise constrained generation results.** Compared with constraint-satisfying 3D-FRONT subsets (3D-FRONT*) for each task, iARCS achieves consistently lower FID while maintaining strong physical and functional quality. This indicates greater diversity under constraints and highlights iARCS as an effective engine for synthetic scene generation and data augmentation.

Task	Method	Diversity		Physical Plausibility			Functional Utility	
		FID↓	SCA	Col_{obj} ↓	Col_{scene} ↓	R_{out} ↓	$R_{reach}(\%)$ ↑	$R_{walkable}$ ↑
1. "Robot grasping scene where all support surfaces are within a 1.0 m vertical reach limit"	3D-FRONT* (229/4041)	4.36	97.34%	50.68%	83.72%	**2.79%**	**82.26%**	**0.786**
	iARCS (Ours)	**3.144**	**77.92%**	**49.29%**	**83.61%**	5.84%	80.37%	0.744
2. "Room with a TV stand positioned so a farsighted person can view a 4K TV from bed"	3D-FRONT* (246/4041)	4.706	97.02%	38.79%	70.04%	**1.71%**	86.38%	0.810
	iARCS (Ours)	**2.905**	**85.70%**	**36.37%**	**67.50%**	3.45%	**86.77%**	**0.790**
3. "A bedroom scene with a functional study zone"	3D-FRONT* (42/4041)	14.49	97.58%	**56.86%**	**88.67%**	**3.33%**	**75.17%**	0.728
	iARCS (Ours)	**2.629**	**77.97%**	59.37%	91.93%	4.59%	74.75%	**0.738**

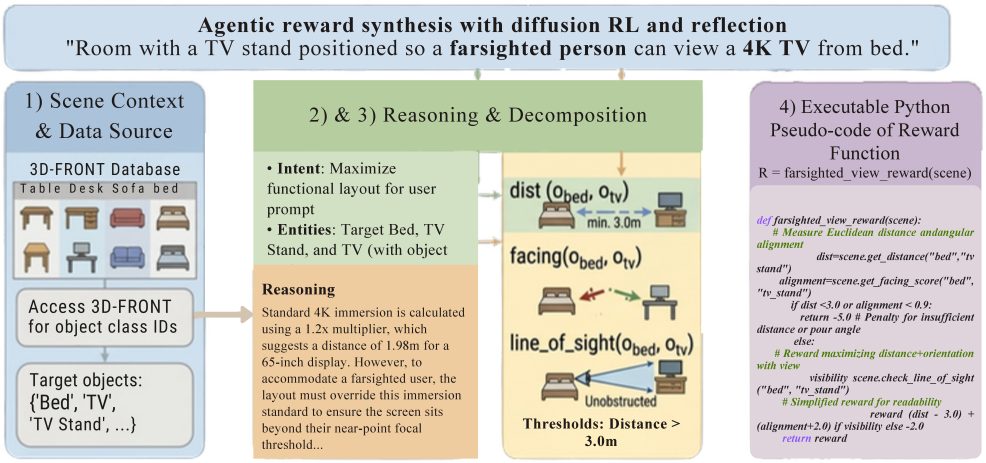

Figure 6. **Example of agentic reward synthesis for a task-specific prompt.** Given the prompt "Room with a TV stand positioned so a farsighted person can view a 4K TV from bed," the LLM pipeline (1) retrieves scene context from 3D-FRONT objects, (2) performs intent-level reasoning and constraint decomposition, (3) derives measurable geometric checks (e.g., viewing distance and orientation constraints), and (4) outputs executable reward code used in the diffusion RL loop. This example illustrates how natural-language requirements are translated into optimizable reward functions for constraint-aware scene generation.

Table 4. **Ablation on single-stage vs. two-stage training for a task-specific constraint.** Task: "Scene with TV and bed at a 3 m distance." **Single-stage** jointly optimizes task, physical, and functional rewards from scratch. **Two-stage** first optimizes physical+functional rewards, then fine-tunes with task+physical+functional rewards. Under the same reward budget, two-stage training consistently outperforms single-stage training across all reported metrics.

Method	Physical Plausibility			Functional Utility		Diversity	
	Col_{obj} ↓	Col_{scene} ↓	R_{out} ↓	$R_{reach}(\%)$ ↑	$R_{walkable}$ ↑	FID↓	SCA
Single-stage	62.17%	95.83%	16.07%	64.66%	0.679	1.96	78.69%
Two-stage	**38.79%**	**70.04%**	**5.84%**	**80.37%**	**0.744**	**1.75**	**72.38%**

4.6. Limitations

Our method has several limitations. First, reward quality depends on LLM-generated task decomposition and reward-code synthesis; ambiguous prompts can produce suboptimal or incomplete constraints, which may affect optimization reliability. Second, although our two-stage training improves physical plausibility and functional utility, it introduces additional compute cost due to iterative RL fine-tuning and repeated reward evaluation.

5. Conclusion

We presented iARCS, an iterative agentic RL framework for controllable 3D indoor scene generation that combines

LLM-based task decomposition, executable reward synthesis, and staged policy refinement. Experiments show that iARCS improves physical plausibility and functional utility while maintaining competitive distribution quality, and that iARCS-generated data can further improve a base scene generator through self-augmentation. We also demonstrated strong task-conditioned generation, where iARCS achieves lower FID than constraint-satisfying dataset subsets, indicating improved diversity under user-specified constraints.

For future work, we aim to extend generalization beyond 3D-FRONT-style indoor distributions to broader domains, including physically grounded 4D generation tasks.

References

[1] Zixuan Bian, Ruohan Ren, Yue Yang, and Chris Callison-Burch. Holodeck 2.0: Vision-language-guided 3d world generation with editing, 2025. 2, 3

[2] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning, 2023. 2, 3, 5

[3] Bernhard Bucher et al. Respace: Iterative scene generation with retrieval-augmented spatial constraints. *arXiv preprint arXiv:2501.05484*, 2025. 2, 3, 4

[4] Angel Chang, Will Monroe, Manolis Savva, Christopher Potts, and Christopher D. Manning. Text to 3d scene generation with rich lexical grounding, 2015. 2

[5] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Jordi Salvador, Kiana Ehsani, Winson Han, Eric Kolve, Ali Farhadi, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation, 2022. 1, 2

[6] Helisa Dhamo, Fabian Manhardt, Nassir Navab, and Federico Tombari. Graph-to-3d: End-to-end generation and manipulation of 3d scenes using scene graphs, 2021. 2, 3

[7] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models, 2023. 3

[8] Weitao Feng, Hang Zhou, Jing Liao, Li Cheng, and Wenbo Zhou. Layoutgpt: Compositional visual planning and generation with large language models. *arXiv preprint arXiv:2303.09499*, 2023. 2, 3, 4

[9] Huan Fu, Bowen Cai, Lin Gao, Lingxiao Zhang, Jiaming Wang Cao Li, Zengqi Xun, Chengyue Sun, Rongfei Jia, Binqiang Zhao, and Hao Zhang. 3d-front: 3d furnished rooms with layouts and semantics, 2021. 5, 7

[10] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10933–10942, 2021. 1, 5

[11] 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, 2021. 5

[12] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. 3

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1, 3

[14] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. 5

[15] Siyi Hu, Diego Martin Arroyo, Stephanie Debats, Fabian Manhardt, Luca Carlone, and Federico Tombari. Mixed diffusion for 3d indoor scene synthesis, 2024. 1, 2, 3, 5, 6

[16] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 5

[18] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. In *arXiv preprint arXiv:1712.05474*, 2017. 1

[19] Chenguo Lin and Yadong Mu. Instructscene: Instruction-driven 3d indoor scene synthesis with semantic graph prior, 2024. 2, 3

[20] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023. 1

[21] Gabrielle Littlefair, Niladri Shekhar Dutt, and Niloy J. Mitra. Flairgpt: Repurposing llms for interior designs. *Computer Graphics Forum*, 44(2), 2025. 2

[22] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models, 2024. 2, 3, 4

[23] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. Atiss: Autoregressive transformers for indoor scene synthesis, 2021. 1, 2, 3, 5, 6

[24] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 1

[25] Nicholas Pfaff, Hongkai Dai, Sergey Zakharov, Shun Iwase, and Russ Tedrake. Steerable scene generation with post training and inference-time search, 2025. 2, 3

[26] Siyuan Qi, Yixin Zhu, Siyuan Huang, Chenfanfu Jiang, and Song-Chun Zhu. Human-centric indoor scene synthesis using stochastic grammar, 2018. 2

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 6

[28] Alexander Raistrick, Lingjie Mei, Karhan Kayan, David Yan, Yiming Zuo, Beining Han, Hongyu Wen, Meenal Parakh, Stamatis Alexandropoulos, Lahav Lipson, Zeyu Ma, and Jia Deng. Infinigen indoors: Photorealistic indoor scenes using procedural generation, 2024. 2

[29] Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models, 2025. 3

[30] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021. 1, 3, 5

[31] Sanjana Srivastava, Aishwarya Padmakumar, Theophile Gervet, Dustin Schwenk, Shubham Mahi, Tejas Gokhale, Jaya Dharanipragada, Ilija Radosavovic, Roberto Mart'in-Mart'in, Li Fei-Fei, Silvio Savarese, Hyowon Gweon,

Juan Carlos Niebles, Yuke Zhu, and Siddharth Karamcheti. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. *arXiv preprint arXiv:2403.09227*, 2024. 1

[32] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014. 6

[33] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, Jack Turner, Nathan Maestre, Mustafa Mukadam, Oleksandr Maksymets, Li Anqi, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems*, 34:251–266, 2021. 1

[34] Jiapeng Tang, Yinyu Nie, Lev Markhasin, Angela Dai, Justus Thies, and Matthias Nießner. Diffuscene: Denoising diffusion models for generative indoor scene synthesis, 2024. 1, 2, 5, 6

[35] Gemini Team. Gemini: A family of highly capable multimodal models, 2025. 3

[36] Weiqi Wang, Zihang Zhao, Ziyuan Jiao, Yixin Zhu, Song-Chun Zhu, and Hangxin Liu. Rearrange indoor scenes for human-robot co-activity, 2023. 2

[37] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers, 2021. 2, 3

[38] Yandan Yang, Baoxiong Jia, Peiyuan Zhi, and Siyuan Huang. Physcene: Physically interactable 3d scene synthesis for embodied ai, 2024. 1, 2, 3, 6

[39] Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, Chris Callison-Burch, Mark Yatskar, Aniruddha Kembhavi, and Christopher Clark. Holodeck: Language guided generation of 3d embodied ai environments, 2024. 2, 3, 4

[40] Yixuan Yang, Zhen Luo, Tongsheng Ding, Junru Lu, Mingqi Gao, Jinyu Yang, Victor Sanchez, and Feng Zheng. Optiscene: Llm-driven indoor scene layout generation via scaled human-aligned data synthesis and multi-stage preference optimization, 2025. 2

[41] 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, 2023. 2, 3

[42] Yinan Zhang, Eric Tzeng, Yilun Du, and Dmitry Kislyuk. Large-scale reinforcement learning for diffusion models, 2024. 3