I-OCTOSCENE: MULTI-AGENT 3D SCENE SYNTHESIS WITH MULTIMODAL INTERACTIVE INSTRUCTIONS

Anonymous authors
Paper under double-blind review



Figure 1: Overview of the multi-agent 3D scene synthesis with multimodal interactive instructions.

ABSTRACT

3D Scene Synthesis (3DSS) has emerged as a key enabler of embodied AI, providing diverse and controllable 3D environments for training and evaluation. Although recent approaches have achieved notable progress in 3DSS, they are still constrained by closed and monolithic paradigms, limiting synthesis personalization and quality. To address these issues, we propose I-OctoScene, an Interactive Multi-Agent 3DSS framework that generates 3D indoor scenes through multi-agent collaboration guided by progressive external interactions. Specifically, I-OctoScene introduces a Progressive Multimodal Interactive Refinement module (PMIR) to support the interpretation and integration of external feedback, enabling controllable scene generation aligned with personalized user requirements. In addition, an Intent-Aligned Tool Coordination mechanism (IATC) leverages Adaptive Tooling Skills (ATSs) to systematically orchestrate tool usage, facilitating complex external intent resolution and coordinated tool collaboration, ultimately improving synthesis quality. Extensive quantitative and visualization experiments across diverse room-type settings demonstrate the superiority of our proposed method.

1 INTRODUCTION

3D Scene Synthesis (3DSS) (Qi et al., 2018; Wei et al., 2023; Mildenhall et al., 2021) aims to leverage diverse 3D assets to carefully construct realistic 3D environments, serving as a critical enabler for rapid content creation in virtual games and metaverse applications. With the rapid advancement of Embodied Intelligence (EI) (Yang et al., 2025; Feng et al., 2026b), there is a fast-growing demand for employing 3DSS to precisely and controllably model actionable worlds tailored to EI agent training (Feng et al., 2025). However, the joint requirements of physical plausibility and realism

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.

render 3DSS a time-consuming and labor-intensive challenge, demanding interdisciplinary expertise across 3D vision, computer graphics, and interior design.

To achieve automated 3DSS, early rule-based methods (Raistrick et al., 2024) leverage fixed layout templates and physical rules to select assets and arrange scenes programmatically. Although they enforce physical plausibility, their rigid design choices limit the diversity and realism of generated scenes. To tackle this, generation-based methods (Tang et al., 2024; Zhai et al., 2024) transfer successful paradigms of 2D image generation (*e.g.*, GANs and Diffusion Models) to 3DSS. In addition, several approaches (Wang et al., 2024; Sun et al., 2025; Fu et al., 2024) leverage Vision-Language Models (VLMs) to incorporate rich 2D priors, enhancing visual realism and variability. However, their limited 3D reasoning ability and inherent hallucination still compromise the physical plausibility of generated scenes. To address these limitations, agent-based methods (Yang et al., 2026) have recently emerged. They exploit LLM-based agentic frameworks (Lu et al., 2025a; Zhou et al., 2023) and dynamically invoke layout rules, physical constraints, and generative models as tools, enabling a reason–act–react paradigm for 3DSS to enhance the physical plausibility and realism.

However, existing methods remain limited to **centralized single-agent orchestration**, suffering from coupling bias that adversely affects process involvement and task decomposition. On one hand, most pipelines are monolithic, requiring users to fully specify all design requirements before scene synthesis, only **producing scenes in a one-shot manner without user involvement**. This prevents users from progressively controlling asset selection and placement during the synthesis process, making it difficult to achieve satisfactory results. On the other hand, the multi-role design of a single agent leads to ambiguous task decomposition and role assignment, introducing inconsistencies between design intent and execution actions. This consequently results in **misinterpretation of the design goals and uncoordinated tool invocation**, ultimately leading to suboptimal synthesis quality.

To address this, we propose an Interactive Multi-Agent 3DSS framework, namely **I-OctoScene**, which enables multi-round interactions with external users and progressively generates 3D scenes through multi-agent collaborative task decomposition, as shown in Figure 2. Specifically, I-OctoScene employs a set of specialized agents, including *Art Director*, *Interior Designer*, *Engineer*, and *Quality Inspector*, forming an agentic 3DSS closed loop that covers management, planning, design, execution, and verification. To incorporate external user control and requirements while coordinating agent tool execution, we further propose a Progressive Multimodal Interactive Refinement module (PMIR) and an Intent-Aligned Tool Coordination mechanism (IATC). More specifically, PMIR introduces an *External Instructor* that supports multi-round and multimodal (*i.e.*, linguistic, control, and visual) interactions between external users and the synthesis framework, enabling iterative scene refinement. To better accommodate external instructions, PMIR further evolves the *Art Director* into an *Art Consultant* capable of interpreting and integrating interactive feedback, thereby enhancing scene controllability and alignment with personalized user requirements. In addition, inspired by object-oriented programming, IATC introduces Adaptive Tooling Skills (ATSs), which define an empirical abstract orchestration for tool execution. Under specific design intents, *Engineer* can invoke appropriate ATSs according to their respective Skill Manuals and systematically organize tool usage based on the contexts of the scene synthesis process. Benefiting from this design, such a structured yet flexible orchestration mechanism ensures coordinated and effective tool execution, providing a comprehensive measure of synthesis performance and interaction compliance. To fill the gap in process-oriented evaluation for interactive 3DSS methods, we introduce an Interaction Evaluation Benchmark. It provides a comprehensive assessment across scene quality, interaction diversity, per-round intent adherence, and final query fulfillment, evaluating the effectiveness of both scene synthesis and interactions. In summary, our contributions are as follows:

- We propose a novel Interactive Multi-Agent 3D Scene Synthesis Framework (I-OctoScene), which supports multi-round external interactions and progressively generates 3D scenes through collaborative multi-agent tool orchestration.
- A Progressive Multimodal Interactive Refinement module (PMIR) and an Intent-Aligned Tool Coordination mechanism (IATC) are presented to bridge external requirements with the coordinated execution of agent tools.
- An Interaction Evaluation Benchmark is introduced to provide a new perspective for evaluating 3DSS in terms of scene quality and interaction compliance.
- We conduct extensive 3DSS comparison experiments across various scenes. Our LLM-based and human-based I-OctoScene demonstrate superior performance in 3DSS evaluations.

2 Related Works

3D Scene Synthesis. 3D Scene Synthesis (3DSS) aims to construct plausible layouts and physically consistent 3D environments. Existing 3DSS approaches can be categorized into three subgroups: **1)** Rule-based methods (Deitke et al., 2022; Raistrick et al., 2024; Gao et al., 2024) rely on predefined layouts and physical constraints to populate scenes procedurally. However, such rigid and sequential execution significantly limits the scene diversity. To address this issue, **2)** generation-based methods (Yang et al., 2024a; Bai et al., 2025) model and generate scene layouts and object arrangements in a data-driven manner. With the advancement of LLMs, recent methods (Sun et al., 2025; Fu et al., 2024) further leverage rich world and 2D prior knowledge to produce more complex layouts and enable open-vocabulary object placement. Despite the improved diversity, the inherently stochastic nature of generation introduces challenges in controllability (Lu et al., 2025a). **3)** Agent-based methods (Yang et al., 2026) exploit the reasoning and planning capabilities of LLMs, treating the aforementioned models as tools and invoking them iteratively. This paradigm enables a better balance between diversity and controllability in 3DSS.

LLM-driven Agentic Systems. As LLMs scale and architectures advance, they increasingly exhibit emergent reasoning and planning capabilities, enabling them to serve as a central *brain* that orchestrates specialized tools and models for complex tasks. Early LLM-based agents (Yao et al., 2022; Schick et al., 2023) primarily focused on simple task decomposition and tool use, whereas recent agents (Shen et al., 2023; Wang et al., 2023) support dynamic decision-making over multi-step workflows. These advances have further unlocked the potential of 3D vision tasks that require both perception and reasoning. Recent approaches (Xu et al., 2024; Mi et al., 2025) enhance spatial understanding by integrating 3D perception and localization models, while others (Ling et al., 2026) incorporate multimodal inputs and planning modules for more precise 3D reasoning. Additionally, LLMs have been used to generate executable action sequences (Zhang et al., 2025; Yu et al., 2025), enabling agents to navigate, manipulate objects, and perform fine-grained relational arrangements in indoor environments.

3 Method: I-OctoScene

Given an initially designed query q_0, agentic 3D Scene Synthesis (3DSS) (Yang et al., 2026; Pfaff et al., 2026) requires a model $\mathcal{M}$ to generate a 3D scene s_0 based on the query, leveraging a tool set $\mathcal{T}$ and asset repositories $\mathcal{A}$, and to obtain its corresponding 2D room rendering r_0 and asset layout parameters l_0. After n rounds of iterative design refinement, the process yields a sequence of synthesized 3D scenes $\mathcal{S} = \{s_i\}_{i=1}^{n}$, their corresponding 2D renderings with grid annotations $\tilde{\mathcal{R}} = \{\tilde{r}_i\}_{i=1}^{n}$, and layout parameters $\mathcal{L} = \{l_i\}_{i=1}^{n}$. At each round, the annotated rendering $\tilde{r}_i$ and layout parameters l_i of the generated scene are fed into an LLM for evaluation, yielding 3DSS evaluation grades $\mathcal{G} = \{g_i\}_{i=1}^{n}$ to assess scene quality. To address this complex task, we propose an Interactive Multi-Agent 3DSS framework, termed **I-OctoScene**, which orchestrates multiple LLM-driven agents to collaboratively construct 3D scenes under progressive external guidance, as illustrated in Figure 2.

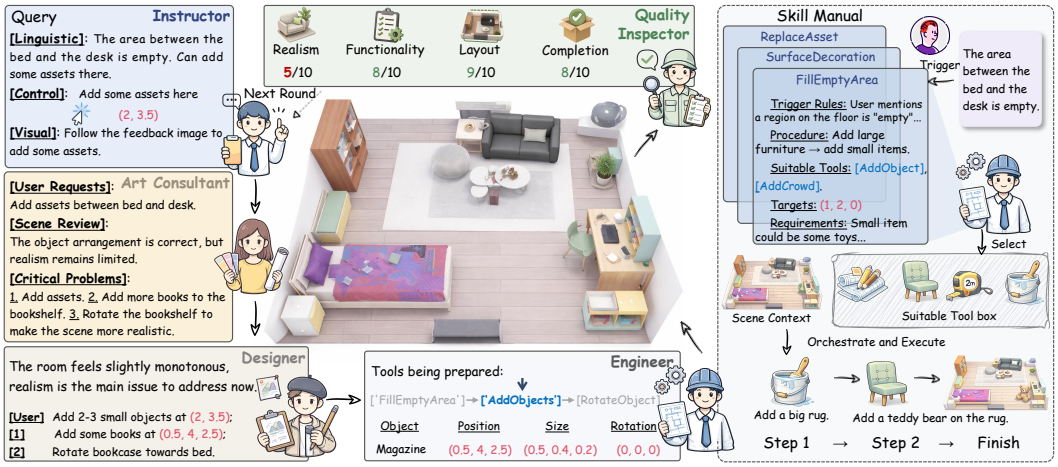
3.1 Multi-Agent 3D Scene Synthesis Framework

Existing agentic 3DSS solutions still rely on a single monolithic agent, resulting in coupling bias in task decomposition and inherent performance bottlenecks. To address these limitations, we propose a Multi-Agent 3DSS framework that coordinates multiple specialized agents to facilitate more comprehensive and effective scene synthesis. Specifically, we define the agent set as:

$$\mathcal{M} = \{\mathcal{M}_\alpha, \mathcal{M}_\beta, \mathcal{M}_\gamma, \mathcal{M}_\delta\}, \tag{1}$$

where $\mathcal{M}_\alpha$, $\mathcal{M}_\beta$, $\mathcal{M}_\gamma$, and $\mathcal{M}_\delta$ denote the *Art Director*, *Interior Designer*, *Engineer*, and *Quality Inspector* agents, respectively. Taking the i-th iteration round as an example, the four agents in our framework can be introduced as follows:

Art Director (*Issue Identification*): *Art Director* $\mathcal{M}_\alpha$ is responsible for analyzing the current visual state and last-round evaluation results of the synthesized scenes. More specifically, it leverages evaluation metrics and scores of the last round g_{i-1}, asset layout parameters l_{i-1}, and top-down renderings with grid annotations $\tilde{r}_{i-1}$ to determine the primary directions for the next round. Formally,



(a) Multi-agent 3DSS Framework with PMIR (b) IATC Illustration

Figure 2: Pipeline of I-OctoScene. Specifically, the left shows the Multi-agent 3DSS framework with PMIR: *Instructor* provides interactive requirements, *Art Consultant* interprets interactions and identifies critical issues, *Designer* designs to refine the scene, *Engineer* executes tools and ATSs for scene enhancement, and *Quality Inspector* evaluates the scene and guides the next round. The right illustrates the ATS execution, where complex instructions trigger ATS invocation and the *Engineer* follows the Skill Manual to execute tools systematically.

it generates high-level refinement guidance $h_i = \mathcal{M}_\alpha(g_{i-1}, \tilde{r}_{i-1}, l_{i-1})$, where h_i denotes the refinement guidance for the current interaction round.

Interior Designer (*Plan Formulation*): Acting as a strategic planner, *Interior Designer* $\mathcal{M}_\beta$ attempts to formulate the high-level guidance into a concrete scene design plan. Following the guidance, it determines which objects at which locations should be added, moved, rotated, or removed. A key constraint of this agent is to maintain semantic coherence (*e.g.*, ensuring consistency with the designated room type) and asset compatibility, producing a step-by-step operating plan for scene refinement and asset adjustment: $p_i = \mathcal{M}_\beta(h_i, l_{i-1})$, where $p_i = \{p_i^j\}_{j=1}^m$ represents the ordered manipulation plan consisting of m actions.

Engineer (*Plan Execution*): Serving as the technical executor, *Engineer* $\mathcal{M}_\gamma$ consumes the *Designer*'s operating plan and translates it into concrete and executable actions f_i^j, where $j \in [1, m]$. Grounded by a comprehensively defined toolset (Yang et al., 2026) (*e.g.*, AddObject, UpdateLayout), the *Engineer* dynamically generates parameterized tool calls $f_i^j = \mathcal{M}_\gamma(p_i^j, s_i^{j-1}, \mathcal{T}, \mathcal{A})$. It denotes the j-th tool action in the i-th round, which invokes appropriate tools and assets from $\mathcal{T}$ and $\mathcal{A}$ to modify the scene s_i^{j-1}, namely the intermediate scene after the $(j-1)$-th action in this i-th round. The *Engineer* is uniquely capable of chaining multiple tools in a single round, enabling iterative and logical manipulation of the synthesized scene to materialize the design plan, formulated as:

$$s_i = s_{i-1} \oplus \bigcup_{j=1}^{m} f_i^j, \tag{2}$$

where $\oplus$ denotes the integration of scene updates within the current iteration, while $\bigcup$ represents the aggregation of all tool-induced modifications.

Quality Inspector (*Result Verification*): *Quality Inspector* $\mathcal{M}_\delta$ rigorously evaluates the results of scene synthesis. It leverages annotated scene rendering $\tilde{r}_i$ and asset layout parameters l_i of s_i to compute quantitative measures (*i.e.*, the number of objects, out-of-boundary objects, and collided objects) and aesthetic grades $g_i = \mathcal{M}_\delta(\tilde{r}_i, l_i)$, including realism, functionality, correctness, and completeness.

At this point, the four agents operate in a coordinated manner to iteratively construct scenes. Specifically, the *Art Director* first identifies the core issues in the current synthesized scene (with the first round focusing on scene initialization) and generates high-level guidance. Based on this guidance,

the *Interior Designer* formulates a concrete operating plan, which is then executed by the *Engineer* through a series of tool executions. The resulting scene is subsequently evaluated by the *Quality Inspector*, completing one iteration. This process continues for n rounds or until the *Art Director* determines that the scene is sufficiently refined.

3.2 Progressive Multimodal Interactive Refinement

Although the multi-agent 3DSS framework alleviates the coupling bias of single-agent methods, its inflexible closed-loop pipeline can still lead to overconfidence and monotonous scene generation, making it prone to rigid and suboptimal solutions. To address this issue, we propose a Progressive Multimodal Interactive Refinement module (PMIR), which enables richer perception and interaction with external users for scene synthesis. Specifically, inspired by interactive learning (Chen et al., 2024), PMIR introduces an *External Instructor* into the isolated synthesis process. Based on the synthesized scene result from the previous round, *Instructor* can progressively provide instructive feedback to I-OctoScene. In contrast to the internal *Quality Inspector*, it goes beyond scene quality evaluation by providing an interface for external users to express evolving requirements and real preferences, which are promptly reflected in the synthesized scene. To achieve this, PMIR supports multimodal composed instructions for scene refinement. To be specific:

- **Linguistic Instruction:** *External Instructor* can freely express its requirements for scene synthesis using natural language. Specifically, the linguistic instructions may refer to any objects in the scene or their absolute coordinates (*e.g.*, (x, y, z)) or relative positions (*e.g.*, left, right, between), and directly specify the actions (*e.g.*, AddObject, RemoveObject) to be performed on them.
- **Control Instruction:** The *External Instructor* is derived from control instructions (*e.g.*, clicking, selecting, and referring) represented as structured text. Combined with natural language instructions, it enables rapid and flexible refinement of selected regions or objects.
- **Visual Instruction:** *External Instructor* can directly annotate grid-labeled top-down renderings of synthesized scenes using brush strokes such as circles, rectangles, or arrows to indicate target regions or objects that require modification. Specifically, *External Instructor* can circle a region and use linguistic instructions to specify adding particular objects within that area. It can also mark or highlight a target object and use linguistic instructions to request its relocation or removal. Additionally, it supports drawing arrows on objects to indicate arbitrary movement adjustments in conjunction with linguistic instructions.

To better handle complex external instructions, the *Art Director* $\mathcal{M}_\alpha$ is extended into a VLM, namely *Art Consultant* $\tilde{\mathcal{M}}_\alpha$. In addition to leveraging the previous-round grades g_{i-1}, layouts l_{i-1}, and renderings $\tilde{r}_{i-1}$, it interprets and incorporates linguistic, control, or visual instructions $\mathcal{E}_i$ to identify key scene issues for subsequent planning:

$$h_i = \tilde{\mathcal{M}}_\alpha(g_{i-1}, \tilde{r}_{i-1}, l_{i-1}, \mathcal{E}_i). \tag{3}$$

Specifically, when control instructions are provided, control keys (*e.g.*, clicks, box selections, and arrow indicators) together with their positional values are converted into a control-position text for execution. When visual instructions are available, they are fed into the *Art Consultant*, enabling the agent to identify annotated regions in the rendering and localize them using the associated visual coordinate information. Finally, the agent generates a 3DSS guideline for the *Interior Designer* by prioritizing external interactive requirements, based on the linguistic instruction, as well as annotations and positional information derived from the control and visual instructions.

3.3 Intent-Aligned Tool Coordination

By leveraging external instructions in PMIR, I-OctoScene achieves better alignment with user requirements. However, complex intents of *Instructor* and *Designer* often lead to incomplete, redundant, or even conflicting tool invocations, resulting in a suboptimal synthesis process. To address this, we draw inspiration from studies on hierarchical planning in human cognition (Correa et al., 2025; Tomov et al., 2020). These studies indicate that cognitively mature humans pursue abstract goals through structured subplans rather than step-wise greedy decisions, motivating a similar paradigm for agentic tool orchestration. For complex intents that cannot be addressed through single-step tool invocation, I-OctoScene leverages Adaptive Tooling Skills (ATSs), which encapsulate

predefined abstract procedures that guide coordinated tool orchestration to accomplish these intents efficiently, as illustrated in Figure 3.

More specifically, to enable *Engineer* agent to understand ATSs and clearly determine when and how to invoke them, we first define a Skill Manual (*i.e.*, metadata) for each ATS. To be specific:

- **Trigger Rules:** Similar to the skill, an ATS is triggered when relevant needs or semantically similar cues are mentioned. For example, if the *Instructor* comments that an area is relatively empty, this can trigger the FILLEMPTYAREA to address the issue.
- **Procedure:** It constitutes pre-declared empirical steps. Specifically, it defines the key executable steps $\mathcal{P} = \{p_1, \cdots, p_i\}_i^{n_p}$ required to address a complex intent, while leaving the concrete tool actions unspecified. For example, when a scene is identified as partially empty by *Instructor*, FILLEMPTYAREA defines: Add Large Furniture $p_1 \rightarrow$ Add Small Items p_2. With empirical procedural guidance, *Engineer* can avoid hastily populating a region with an arbitrary assortment of haphazard assets, preventing physical conflicts and visual inconsistency within the scene.
- **Suitable Tools:** It provides a predefined set of tools for each step p_i in the procedure, from which *Engineer* can choose. This ensures tool relevance and reduces the risk of misuse. For example, in the Add Small Items step, only specific tools are available. *Engineer* then selects appropriate tools based on the scene context and the current skill state to add appropriate objects.
- **Targets:** It specifies the exact positions of the target region or specific object within the intent. For example, a designer may identify an area as relatively empty, and the specific location of that target region is then filled into this field.
- **Other Requirements:** It specifies other additional requirements within the intent. For example, the *Instructor* may require that the target area be filled and that a round table be placed first.

ATS is inspired by object-oriented programming, where the Skill Manual serves as an abstract class that defines the fundamental specifications of an ATS. This design enables users to easily create customized ATSs by specifying trigger rules, abstract procedures, and suitable tools. Once triggered, the *Engineer* instantiates the ATS based on the current room type and layout context, and follows its structured procedure to determine execution objectives and invoke appropriate tools.

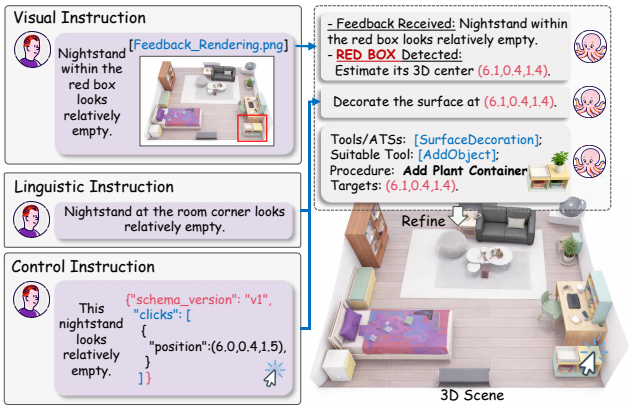


Figure 3: Illustration of I-OctoScene invoking ATSs to handle complex multimodal instructions.

Benefiting from this, high-level design guidelines or complex intents can be effectively translated into coherent tool actions. The predefined procedure facilitates structured tool orchestration, reducing execution conflicts and omissions. Furthermore, unlike existing approaches (Yang et al., 2026), which are limited to recognizing atomic tool actions, our method allows ambiguous design intents to invoke ATSs for targeted resolution, thereby improving both interpretability and coordination.

4 INTERACTION EVALUATION BENCHMARK

Existing 3DSS evaluation benchmarks (Yang et al., 2026; Çelen et al., 2024) primarily assess the physical plausibility and functional validity of the synthesized scenes. However, they fail to provide a comprehensive evaluation of scene quality, user intent understanding, and fulfillment of external interactive feedback. To address this limitation, we propose an *Interaction Evaluation Benchmark* to comprehensively and progressively evaluate interactive agentic 3DSS methods. To be specific, it consists of the following four components:

- **Quality of the synthesized scene** (*Rating 4/10*). Following the existing 3DSS benchmark (Yang et al., 2026), an LLM is used to evaluate the scene from a top-down view in terms of realism, functionality, layout, and completion. Specifically, this includes measuring asset placement, such

Table 1: Comparison of physics and visual quality with common bedroom and living room settings.

Type	Bedroom								Livingroom							
Method	Physics			Quality					Physics			Quality				
	#Obj	#OB	#CN	Real.	Func.	Lay.	Comp.	Int.	#Obj	#OB	#CN	Real.	Func.	Lay.	Comp.	Int.
ATISS	3.9	0.5	0.6	7.4	7.1	6.6	4.2	–	7.8	0.1	0.7	5.8	5.3	6.4	3.7	–
Holodeck	32.2	0	38.5	8.6	9.1	7.8	6.2	–	23.0	0	5.3	8.9	9.3	7.6	8.1	–
PhyScene	3.3	0.1	0.3	5.7	6.3	5.4	4.0	–	8.0	0	0.7	5.2	5.3	5.1	3.3	–
DiffuScene	3.5	0.1	1.1	6.5	7.0	6.7	3.6	–	6.9	0.5	1.2	5.5	4.9	5.2	3.5	–
LayoutGPT	5.4	1.0	1.3	7.5	8.1	6.7	4.2	–	8.4	1.1	2.8	6.4	5.8	5.2	3.6	–
I-Design	9.6	0	0	8.6	9.3	7.6	6.1	–	9.7	0	0	8.4	8.9	7.7	5.9	–
SceneWeaver	13.5	0	0	8.7	9.2	7.3	7.4	–	18.5	0	0	8.8	9.1	7.4	7.5	–
Ours	20.3	0	0	8.9	9.8	7.5	7.8	8.4	25.4	0	0	9.0	9.8	7.6	8.0	8.4
Ours†	21.7	0	0	9.0	9.9	8.1	8.4	8.9	20.7	0	0	8.9	9.9	8.0	8.0	9.1

as positions, orientations, and scales. Furthermore, it assesses whether the arranged assets meet functional requirements, logical consistency, and livability within the specified room types.

- **Diversity of external interaction** (*Rating 2/10*). This evaluates the ability to respond to external instructions. Specifically, it assesses the extent of interaction support provided by a method, including composed multimodal interactions, linguistic-only interactions, and the ability to handle progressive interactions.
- **Per-round intent adherence** (*Rating 2/10*). In each round, an expert LLM is used to determine whether the generated top-down scene satisfies the user's instruction for that round.
- **Final user query fulfillment** (*Rating 2/10*). An expert LLM is also used to summarize all user requirements and evaluate whether they are satisfied based on the final generated top-down scene.

5 Experiments

Synthesis settings and baselines. We conduct quantitative evaluations of I-OctoScene in comparison with existing baselines under two main synthesis settings. Specifically, the first focuses on common room types that are widely used in existing 3DSS works, *i.e.*, bedroom and living room. The second considers open-vocabulary scene generation, following (Yang et al., 2026; Sun et al., 2025), *e.g.*, children room, gym, *etc*. For the common setting, we compare our method with a range of 3D scene synthesis approaches, including ATISS (Paschalidou et al., 2021), PhyScene (Yang et al., 2024a), DiffuScene (Tang et al., 2024), *etc*. For the open-vocabulary setting, we further evaluate against additional baselines, including LayoutGPT (Feng et al., 2023), Holodeck (Yang et al., 2024b), I-Design (Çelen et al., 2024), and SceneWeaver (Yang et al., 2026). We consider two variants of I-OctoScene, differing in the implementation of the external *Instructor*: **1)** a VLM-based *Instructor* (*i.e.*, Ours) and **2)** a human-in-the-loop *Instructor* (*i.e.*, Ours†).

Evaluation benchmarks. To comprehensively evaluate the effectiveness of existing methods, we follow Yang et al. (2026) and report the average number of objects (*#Obj*), out-of-boundary objects (*#OB*), and collided objects (*#CN*) to measure the physical realism of the synthesized scenes. In addition, we assess visual realism (*Real.*), functionality (*Func.*), layout correctness (*Lay.*), and scene completeness (*Comp.*), along with our proposed interaction evaluation benchmark (*Int.*), to evaluate visual quality and the ability to satisfy user queries. More experiments, detailed experimental settings, and implementation details are provided in our Appendix.

5.1 Comparison with the State-of-the-Art

Tables 1 and 2 present comparisons of the synthesis performance for common and open-vocabulary room settings. These results could yield the following observations: **1)** Current generation-based methods (*e.g.*, LayoutGPT), constrained by the inherent limitations of the training data, produce scenes containing only a limited number of assets, resulting in low overall completeness. In contrast, LLM-based methods (*e.g.*, Holodeck) improve the richness of scenes, but physical collisions among objects increase. **2)** Although agent-based methods achieve relatively better physical plausibility, their rigid overall agent design and lack of progressive interaction with external feedback lead to noticeable limitations in visual quality and query satisfaction ability. **3)** Whether interacting with an external VLM or humans, our I-OctoScene exhibits no physical collision issues and achieves state-of-the-art visual quality performance. This demonstrates the superiority of multi-agent independent orchestration and the incorporation of multi-round interaction in 3DSS. **4)** Our I-OctoScene achieves

Table 2: Comparison of physics and visual quality scores with open-vocabulary room settings.

Metric	#Obj	#OB	#CN	Real.	Func.	Lay.	Comp.	Int.	#Obj	#OB	#CN	Real.	Func.	Lay.	Comp.	Int.	#Obj	#OB	#CN	Real.	Func.	Lay.	Comp.	Int.
Method	Bathroom								Children Room								Gym							
LayoutGPT	7.7	1.3	1	8.3	9.3	7.7	6	-	7.3	1	0.7	6.3	8	6	4	-	6.7	0.7	0	6.7	5.7	3.7	3.7	-
Holodeck	12	0	1.7	7.7	6.7	7	5.3	-	13.7	0	2	7.5	7.5	6.5	5.5	-	20.3	0	5.3	9.7	9.3	6.5	6	-
I-Design	9.7	0	0	7.4	7.2	7.4	5.4	-	11.3	0	0	7.8	8.3	6.8	5.5	-	12	0	0.8	8.2	8.4	7	5.2	-
SceneWeaver	11.7	0	0	8.3	9.3	8	7	-	18.3	0	0	8.7	9.7	7.7	6.7	-	23.7	0	0	8	8	6	5.7	-
Ours	12.3	0	0	8.7	9	8	7	8.3	21	0	0	9	9.7	7.7	6.7	8.3	36.3	0	0	8	10	8	8	8.4
Ours†	13	0	0	9	9.7	8	7.3	8.7	21.7	0	0	9	10	8	8	9.5	32.7	0	0	8.3	10	8.3	8.3	9.5
Method	Kitchen								Meeting Room								Office							
LayoutGPT	7.7	1.3	1.3	5.7	6.3	4.7	3.7	-	7.3	1	0.7	4	3	5.3	2	-	7.3	0.3	0	6.7	7.7	6.3	4	-
Holodeck	20	0	1.3	7.3	6.3	6.3	4.3	-	27	0	0.3	9	10	8	7	-	27	0	4.7	7	6.3	4.3	4	-
I-Design	11.7	0	0	6.5	6.8	5.3	3.5	-	18.7	5.3	0	6	4.5	5.8	4.3	-	11.7	0	0	8	9	6.8	5.4	-
SceneWeaver	25.33	0	0	6.3	5	6.7	6.7	-	26	0	0	8.3	9	7.6	6.3	-	35.7	0	0	8	8	7	6.7	-
Ours	31.0	0	0	8.3	9.0	7.3	7.3	8.2	37.3	0	0	8.3	9.0	6.7	6.7	8.1	36.3	0	0	9	9.3	7.7	8.3	8.4
Ours†	28.7	0	0	8.7	8.7	7.3	8.0	8.6	35.0	0	0	8.3	9.0	7.7	7.3	9.2	38.7	0	0	9.0	10	8.0	9.0	9.6
Method	Restaurant								Waiting Room								Avg. Rating							
LayoutGPT	7.0	0.3	1.7	3.3	2.3	4.7	2.0	-	6.3	0	0.3	6.7	5.7	6.0	4.0	-	7.2	0.7	0.7	6.0	6.0	5.6	3.7	-
Holodeck	35.0	0	12.3	5.3	4.3	4.3	3.7	-	24.0	0	3.7	8.3	9.3	6.7	5.7	-	22.4	0.0	3.9	7.7	7.5	6.2	5.2	-
I-Design	27.7	0	0	6.2	5.2	5.2	4.0	-	10.7	0	0	6.6	6.4	5.8	4.2	-	14.2	0.7	0.1	7.1	7.0	6.3	4.7	-
SceneWeaver	47.3	0	0	7.7	6.3	7.3	5.7	-	24.0	0	0	8.3	9.3	7.7	6.7	-	26.5	0.0	0.0	8.0	8.1	7.3	6.4	-
Ours	48.7	0	0	7.7	6.7	7.7	5.7	7.8	22.7	0	0	8.7	9.3	8.0	7.0	8.3	30.7	0.0	0.0	8.5	9.0	7.6	7.1	8.2
Ours†	56.3	0	0	8.0	7.6	8.0	6.3	8.7	25.7	0	0	8.7	9.7	8.3	7.7	9.4	31.5	0.0	0.0	8.6	9.4	8.0	7.7	9.2

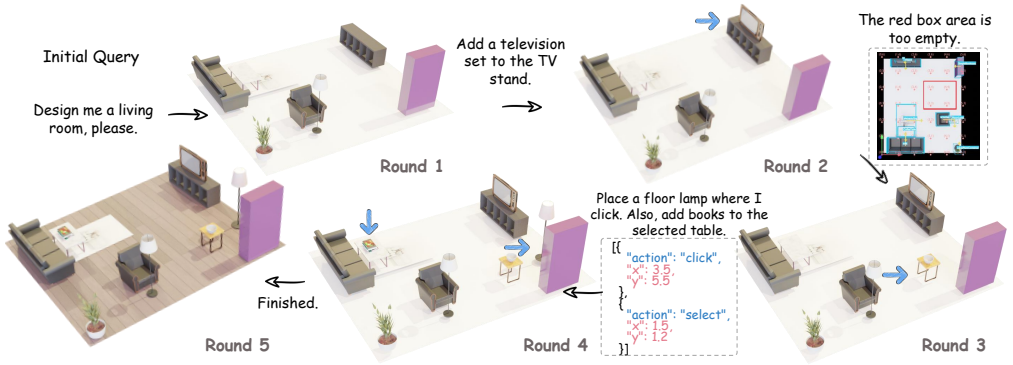


Figure 4: Case study of the scene synthesis process using our I-OctoScene in a living room setting.

superior performance on the interaction evaluation benchmark. This indicates that its diverse and effective interaction mechanisms progressively satisfy external queries over multiple rounds, thereby improving scene quality and alignment with user requirements.

5.2 Ablation Study

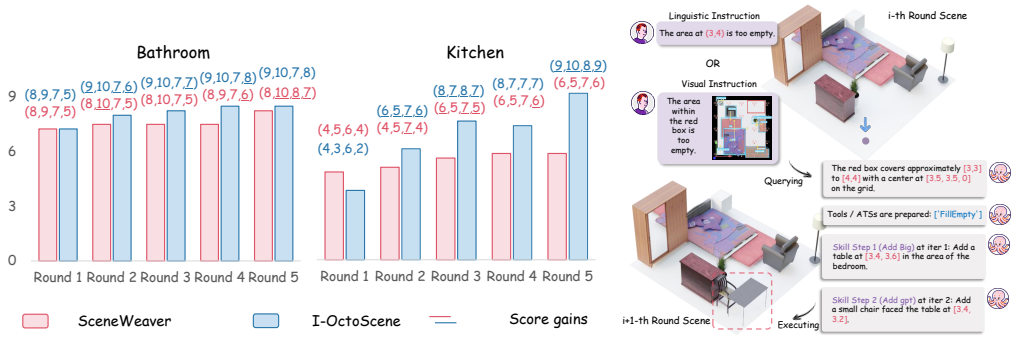
Table 3: Ablation studies with *Bedroom* setting. *w/o* stands for not used. Results are averaged over 3 runs.

Method	Physics			Quality				
	#Obj	#OB	#CN	Real.	Func.	Lay.	Comp.	Int.
w/o Multi-Agent	8.3	0	0	7.3	8.0	6.0	4.7	7.9
w/o PMIR	14.7	0	0	8.0	9.3	7.0	6.3	-
w/o IATC	14.0	0	0	8.0	9.0	6.0	6.0	8.9
w/o User	24.3	0	0	8.7	9.7	7.0	8.0	8.3
Ours	21.0	0	0	9.0	9.7	8.0	8.0	9.5

We conduct an ablation study to evaluate the contribution of each proposed component. We first replace the multi-agent collaboration with the monolithic framework (*i.e.*, *w/o* Multi-Agent). Additionally, we ablate the external interactions and ATSs in PMIR and IATC (*i.e.*, *w/o* PMIR and *w/o* IATC). Furthermore, we present results based on both LLMs and humans. The results in Table 3 lead to the following observation: **1)** I-OctoScene with/without any component will improve/degrade 3DSS quality, demonstrating the effectiveness of the components. **2)** After degrading I-OctoScene into a monolithic framework, both the number of generated objects and the scene quality decrease substantially, indicating that a single agent struggles to coordinate the diverse subtasks involved in 3DSS. **3)** With the involvement of humans, the synthesis quality is improved more easily, demonstrating the importance of external experiential suggestions for 3DSS applications.

5.3 Synthesis Process Analysis

To gain deeper insights into our I-OctoScene, we conduct extensive analysis of the scene synthesis process. Specifically, we first present a representative synthesis case to illustrate the multi-round



(a) Round-wise Score Bar Chart (b) ATS Illustration

Figure 5: (a) quantifies the multi-round scores of our I-OctoScene under two room settings across different difficulty levels, while (b) visualizes the effectiveness of ATS during the 3DSS.

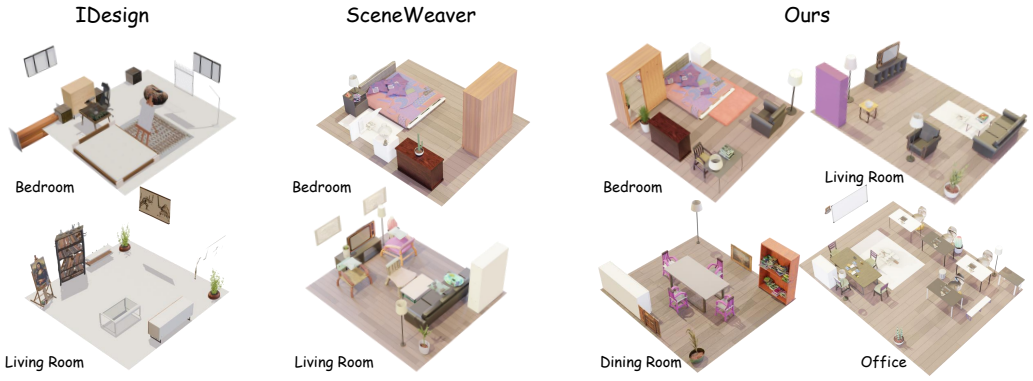


Figure 6: Case illustrations of 3DSS by I-OctoScene, compared with IDesign and SceneWeaver.

interaction of our framework, as shown in Figure 4. In addition, we further compare the round-wise scores of multi-round agentic SceneWeaver and our I-OctoScene across open-vocabulary bathroom and kitchen settings with varying difficulty, as depicted in Figure 5 (a). The results demonstrate that I-OctoScene can effectively exploit external instructions to refine the scene iteratively, thereby achieving better visual quality and 3DSS performance. In particular, under poor initialization in the kitchen setting, SceneWeaver exhibits limited improvement within a few rounds, whereas I-OctoScene can rapidly and comprehensively improve the scene within only 1-2 rounds.

5.4 Visualization Analysis

In this section, we first visualize the invocation workflows and execution results of the ATSs in IATC to provide deeper insights into their roles in scene synthesis, as illustrated in Figure 5 (b). In addition, we visualize and compare the 3D scenes generated by I-OctoScene against I-Design and SceneWeaver, as shown in Figure 6. From these results, we can draw that I-OctoScene enables more organized and rational tool utilization, producing more diverse and practical object placements and layouts than existing methods, leading to higher-quality scene synthesis.

6 Conclusion

In this paper, we propose an Interactive Multi-Agent 3D Scene Synthesis (3DSS) framework, termed I-OctoScene, which progressively generates 3D scenes through collaborative tooling among multiple agents. To be specific, I-OctoScene incorporates a Progressive Multimodal Interactive Refinement module (PMIR) to interpret and integrate external interactive feedback, enhancing scene planning and user alignment. Moreover, the Intent-Aligned Tool Coordination mechanism (IATC) leverages Adaptive Tooling Skills (ATSs) for systematic tool orchestration, facilitating action coordination and complex intent resolution throughout the 3DSS process. Extensive quantitative and visualization experiments across diverse room settings demonstrate the superiority of our proposed method.

7 USAGE OF LARGE LANGUAGE MODELS IN PAPER WRITING

We declare that ChatGPT and MiMo were used solely as general-purpose assistive tools for generating several icons, checking grammar, and polishing the wording of the manuscript drafts. They were not used for retrieval and discovery (*e.g.*, finding related work), research ideation, methodological design, data collection, analysis, or writing substantive content that could qualify as a contribution. The authors take full responsibility for the contents of the paper, including any text influenced by these tools, and no LLM is listed as an author.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Tongyuan Bai, Wangyuanfan Bai, Dong Chen, Tieru Wu, Manyi Li, and Rui Ma. Freescene: Mixed graph diffusion for 3d scene synthesis from free prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5893–5903, 2025.

Mu Cai, Haotian Liu, Siva Karthik Mustikovela, Gregory P Meyer, Yuning Chai, Dennis Park, and Yong Jae Lee. Vip-llava: Making large multimodal models understand arbitrary visual prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12914–12923, 2024.

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

Sijin Chen, Xin Chen, Chi Zhang, Mingsheng Li, Gang Yu, Hao Fei, Hongyuan Zhu, Jiayuan Fan, and Tao Chen. Ll3da: Visual interactive instruction tuning for omni-3d understanding reasoning and planning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 26428–26438, 2024.

David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996.

Carlos G Correa, Sophia Sanborn, Mark K Ho, Frederick Callaway, Nathaniel D Daw, and Thomas L Griffiths. Exploring the hierarchical structure of human plans via program generation. *Cognition*, 255:105990, 2025.

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

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13142–13153, 2023.

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.

Yanglin Feng, Yang Qin, Dezhong Peng, Hongyuan Zhu, Xi Peng, and Peng Hu. Pointcloud-text matching: Benchmark dataset and baseline. *IEEE Transactions on Multimedia*, 2025.

Yanglin Feng, Yongxiang Li, Yuan Sun, Yang Qin, Dezhong Peng, and Peng Hu. Interactive cross-modal learning for text-3d scene retrieval. *Advances in Neural Information Processing Systems*, 38:97972–97992, 2026a.

Yanglin Feng, Hongyuan Zhu, Dezhong Peng, Xi Peng, Xiaomin Song, and Peng Hu. Robust cross-modal alignment learning for cross-scene spatial reasoning and grounding. *Advances in Neural Information Processing Systems*, 38:49055–49075, 2026b.

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.

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

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

Lu Ling, Chen-Hsuan Lin, Tsung-Yi Lin, Yifan Ding, Yu Zeng, Yichen Sheng, Yunhao Ge, Ming-Yu Liu, Aniket Bera, and Max Li. Scenethesis: A language and vision agentic framework for 3d scene generation. In *International Conference on Learning Representations*, volume 2026, pp. 136596–136629, 2026.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023a.

Honglin Liu, Peng Hu, Changqing Zhang, Yunfan Li, and Xi Peng. Interactive deep clustering via value mining. *Advances in Neural Information Processing Systems*, 37:42369–42387, 2024.

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

Pan Lu, Bowen Chen, Sheng Liu, Rahul Thapa, Joseph Boen, and James Zou. Octotools: An agentic framework with extensible tools for complex reasoning. *arXiv preprint arXiv:2502.11271*, 2025a.

Yiding Lu, Mouxing Yang, Dezhong Peng, Peng Hu, Yijie Lin, and Xi Peng. Llava-reid: Selective multi-image questioner for interactive person re-identification. In *International Conference on Machine Learning*, pp. 40868–40887. PMLR, 2025b.

Boyu Mi, Hanqing Wang, Tai Wang, Yilun Chen, and Jiangmiao Pang. Language-to-space programming for training-free 3d visual grounding. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3844–3864, 2025.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Ryoya Nara, Yu-Chieh Lin, Yuji Nozawa, Youyang Ng, Goh Itoh, Osamu Torii, and Yusuke Matsui. Revisiting relevance feedback for clip-based interactive image retrieval. In *European Conference on Computer Vision*, pp. 1–16. Springer, 2024.

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

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

Siyuan Qi, Yixin Zhu, Siyuan Huang, Chenfanfu Jiang, and Song-Chun Zhu. Human-centric indoor scene synthesis using stochastic grammar. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5899–5908, 2018.

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

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36:38154–38180, 2023.

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

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

Momchil S Tomov, Samyukta Yagati, Agni Kumar, Wanqian Yang, and Samuel J Gershman. Discovery of hierarchical representations for efficient planning. *PLoS computational biology*, 16(4): e1007594, 2020.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 2609–2634, 2023.

Yian Wang, Xiaowen Qiu, Jiageng Liu, Zhehuan Chen, Jiting Cai, Yufei Wang, Tsun-Hsuan Wang, Zhou Xian, and Chuang Gan. Architect: Generating vivid and interactive 3d scenes with hierarchical 2d inpainting. *Advances in Neural Information Processing Systems*, 37:67575–67603, 2024.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19037–19047, 2023.

Runsen Xu, Zhiwei Huang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. Vlm-grounder: A vlm agent for zero-shot 3d visual grounding. *arXiv preprint arXiv:2410.13860*, 2024.

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

Yandan Yang, Baoxiong Jia, Shujie Zhang, and Siyuan Huang. Sceneweaver: All-in-one 3d scene synthesis with an extensible and self-reflective agent. *Advances in neural information processing systems*, 38:140319–140351, 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16227–16237, 2024b.

Yuncong Yang, Han Yang, Jiachen Zhou, Peihao Chen, Hongxin Zhang, Yilun Du, and Chuang Gan. 3d-mem: 3d scene memory for embodied exploration and reasoning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 17294–17303, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

Huangyue Yu, Baoxiong Jia, Yixin Chen, Yandan Yang, Puhao Li, Rongpeng Su, Jiaxin Li, Qing Li, Wei Liang, Song-Chun Zhu, et al. Metascenes: Towards automated replica creation for real-world 3d scans. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 1667–1679, 2025.

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

Wenqi Zhang, Mengna Wang, Gangao Liu, Xu Huixin, Yiwei Jiang, Yongliang Shen, Guiyang Hou, Zhe Zheng, Hang Zhang, Xin Li, et al. Embodied-reasoner: Synergizing visual search, reasoning, and action for embodied interactive tasks. *arXiv preprint arXiv:2503.21696*, 2025.

Wangchunshu Zhou, Yuchen Eleanor Jiang, Long Li, Jialong Wu, Tiannan Wang, Shi Qiu, Jintian Zhang, Jing Chen, Ruipu Wu, Shuai Wang, et al. Agents: An open-source framework for autonomous language agents. *arXiv preprint arXiv:2309.07870*, 2023.

Supplementary Appendix

I-OctoScene: Multi-Agent 3D Scene Synthesis with Multimodal Interactive Instructions

In this appendix, we provide supplementary information and discussions on related work, method details, proposed benchmark details, evaluation details, and additional experiments, which are presented in Appendix A, Appendix B, Appendix C, Appendix D, and Appendix E, respectively.

Table 4: Organization of our Supplementary Appendix.

Appendix Section	Description	Pages
Appendix A	Supplementary discussion on related work.	14
Appendix B	Additional details of the proposed method.	14–21
Appendix C	Complete description of the proposed Interaction Evaluation Benchmark.	21–23
Appendix D	Evaluation settings and implementation details.	23
Appendix E	Additional experiments.	23–33
Appendix F	Ethics Statement.	34
Appendix G	Reproducibility Statement.	34

A Additional Related Work

Interactive Learning. In contrast to conventional learning paradigms, interactive learning (Cohn et al., 1996) focuses on progressively improving model behavior through continual interactions. Early studies (Nara et al., 2024; Liu et al., 2024) primarily relied on simple, rigid forms of feedback, such as preference selection or relevance scoring. In recent years, research efforts (Lu et al., 2025b; Feng et al., 2026a) have shifted toward more flexible interaction paradigms based on free-form natural language dialogue. Benefiting from the advancement of LLMs (Bai et al., 2023; Achiam et al., 2023), these paradigms significantly improve interaction effectiveness and gradually expand to encompass more diverse forms and modalities. 3DSS, as a complex and highly structured generation task, inherently requires interaction to refine both design and construction. Although several existing works (Yang et al., 2026; 2024b) incorporate internal iteration, they are limited in that user external guidance cannot be dynamically updated as the scene evolves. Furthermore, the lack of more intuitive feedback mechanisms (Liu et al., 2023a; Chen et al., 2024; Cai et al., 2024) (*e.g.*, visual and composed guidance) still poses a controllability bottleneck for current 3DSS approaches.

B Additional Method Details

In this section, we provide additional methodological details of I-OctoScene that could not be fully covered in the main paper due to space limitations. Specifically, we first summarize the overall I-OctoScene pipeline and present it in the form of an algorithmic workflow. Furthermore, following (Yang et al., 2026), we provide the representation of 3D scenes used within the LLM during the scene synthesis process. In addition, we provide detailed prompts for each agent in I-OctoScene. Furthermore, we introduce the sources of the 3D assets adopted in I-OctoScene. Finally, we describe in detail each tool and Adaptive Tooling Skill (ATS) used in our framework.

B.1 IMPLEMENTATION DETAILS OF I-OCTOSCENE

Algorithm 1 Main Pipeline of I-OctoScene

Input: Query q_0, asset repository $\mathcal{A}$, tool set $\mathcal{T}$, ATS set $\mathcal{K}$, and maximum iteration number n.
Output: Refined 3D scene s_n.

```
Generate the initial scene s_0 from q_0.
Render s_0 and obtain (r_0, l_0).
Evaluate s_0 and obtain g_0.
for i = 1 to n do
    // Progressive Multimodal Interactive Refinement (PMIR).
    Collect multimodal instructions I_i from the Instructor
    Art Consultant analyzes (g_{i-1}, l_{i-1}, r̃_{i-1}, I_i) and generates guidance h_i.
    Interior Designer formulates a refinement plan p_i.
    Engineer determines whether an ATS in K should be triggered.
    // Intent-Aligned Tool Coordination (IATC).
    if ATS is triggered then
        Instantiate the corresponding ATS and generate a tool sequence.
    else
        Generate tool actions directly from p_i.
    end if
    Execute tool actions using T and A to obtain s_i.
    Render s_i and obtain (r_i, l_i).
    Quality Inspector evaluates the scene and obtains g_i.
    if g_i satisfies the stopping criterion then
        break
    end if
end for
return s_i
```

I-OctoScene Algorithm Pipeline. The proposed framework adopts a collaborative multi-agent interaction paradigm for iterative scene refinement, as shown in Algorithm 1. In each round, the *External Instructor* first provides multimodal feedback, including linguistic instructions, control interactions, and visual annotations, to express evolving user preferences. These instructions, together with the current scene state and previous evaluation results, are analyzed by the *Art Director* (*Art Consultant*), which identifies key scene issues and generates high-level refinement objectives.

Based on these objectives, the *Interior Designer* formulates a concrete editing plan, which is subsequently executed by the Engineer through a sequence of tool invocations and asset manipulations. The resulting scene is then assessed by the *Quality Inspector* from both geometric and aesthetic perspectives, with severe degradations triggering an automatic rollback mechanism. Through this closed-loop interaction process, the framework progressively improves scene quality while dynamically adapting to external user intentions.

Scene Representation Metadata. Following (Yang et al., 2026), we represent the 3D scene at each step using a combination of structured 3D layout JSON data and a 2D top-down rendering. The details are illustrated in Figure 7. More specifically, we present a top-down rendering of the scene in Blender, which aligns the visual representation with the coordinate-based layout. To enhance spatial understanding, we annotate the image with the x, y, and z coordinate axes at the origin and 2D projection coordinates on the (x, y) plane to emphasize spatial positions. Each object is additionally marked with a 3D bounding box and its semantic category label to facilitate object recognition by the agent.

```
{
    "objects": {
        "5348940_BedFactory": {
            "location": [
                1.0,
                1.11,
                0.14
            ],
            "rotation": [
                -0.0,
                0.0,
                0.0
            ],
            "size": [
                1.6,
                0.8,
                0.6
            ],
            "parent": [
                [
                    "newroom_0-0",
                    "onfloor"
                ],
                [
                    "newroom_0-0",
                    "against_wall"
                ]
            ]
        },
        ...
        ...
}
```

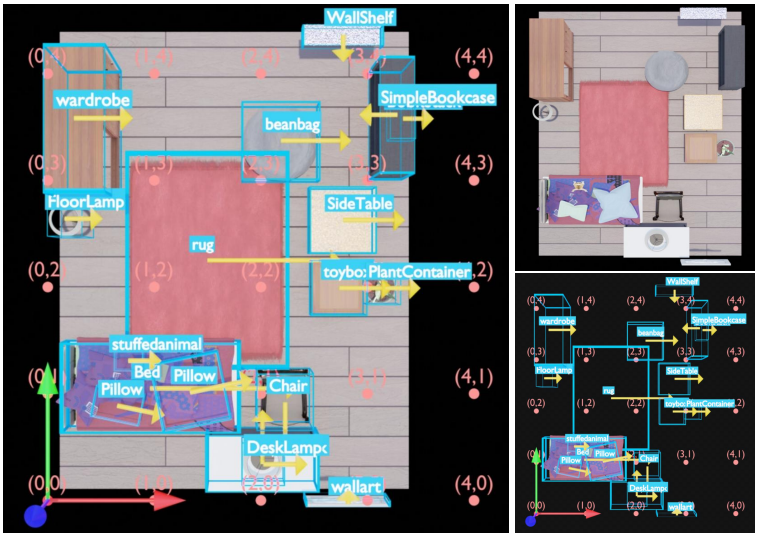



Figure 7: Illustration of the 3D scene representation adopted in our proposed I-OctoScene following (Yang et al., 2026), which combines structured 3D layout JSON data with a 2D top-down rendering.

To compensate for the limited spatial reasoning capabilities of VLMs, particularly in object orientation, we annotate each bounding box with a directional arrow indicating the object's front-facing direction. On the right side, the layout encodes each object's semantic category as the key, with its location, rotation, and size represented as values. We also record relational information for each object, including its parent object and the corresponding relationship type.

B.2 Agentic Prompt Details

Our multi-agent indoor scene generation framework adopts a shared-context memory mechanism in which all agents access the same global scene state, user instructions, and interaction history. The following section summarizes the prompts used by each agent in the pipeline.

Shared Context / Memory Prompt. Before each agent invocation, the framework constructs a shared context containing the global user demand, scene state, and optional user interaction feedback.

Global Demand:

```
Design a modern-style bedroom.
```

Scene Information State:

```
Room Type: {roomtype}
Room Size: {roomsize}
Layout (Objects): {layout_str}
Structure: {structure}
Existing object types you MUST prefer reusing
(to avoid generation timeout): {existing_obj_names}
```

User Explicit Feedback (Optional):

```
USER EXPLICIT FEEDBACK FOR THIS STEP:
{feedback_content}
```

Prompt

***Art Consultant* Agent.** The *Art Consultant* Agent analyzes the rendered scene, spatial arrangement, evaluation metrics, and external feedback to identify geometric inconsistencies, empty regions, and layout deficiencies.

System Prompt:

```
You are an Art Consultant Agent.  Analyze the current scene, spatial
layout, evaluation metrics, and external feedback.  Identify geometry
issues and emptiness, then suggest improvements.
RULES: Focus purely on indoor furniture and layout.  DO NOT suggest
adding doors/windows.
```

Prompt

User Prompt:

```
{User Explicit Feedback}
Based on the current visual environment and feedback, provide a concise
reflection on what needs to be fixed.  Pay special attention to the
User Explicit Feedback (if any).
```

Prompt

The *Art Consultant* Agent additionally maintains the previous two rounds of reflection history as memory context.

***Designer* Agent.** The *Designer* Agent converts the reflection results and user feedback into a precise design plan describing object-level modifications.

System Prompt:

```
You are a Designer Agent.  Create a precise, step-by-step design plan
specifying WHICH objects to add, move, rotate, or remove.
RULES:
Never add doors/windows.
Focus solely on furniture and decor.
ABSOLUTE PRIORITY: Execute 'User' feedback and exact JSON commands
BEFORE generating any other items.
CRITICAL: Strictly match objects to the current room type.
ASSET COMPATIBILITY: Use broad, general categories for object names
instead of overly specific items to ensure successful retrieval from
Objaverse.
```

Prompt

User Prompt:

```
{User Explicit Feedback}
Reflector Analysis:
{reflection_content}
Generate your precise Design Plan to guide the Engineer.  (REMEMBER:
Implement the USER EXPLICIT FEEDBACK first!)
```

Prompt

The *Designer* Agent stores the previous two rounds of generated design plans as memory history.

***Engineer* Agent.** The *Engineer* Agent translates the textual design plan into executable tool calls for scene editing and asset manipulation.

System Prompt:

```
{Base System Prompt defining tool specifications}
You are an Engineer Agent.  You have access to tools.  Execute the
Design Plan strictly by generating necessary tool calls.
You can call MULTIPLE tools in a single response to execute different
steps sequentially.
CRITICAL: Keep object descriptions clear and general (e.g., 'toys'
instead of 'teddy bear', 'plant' instead of 'monstera') when passing
arguments to tools to ensure successful asset retrieval from Objaverse.
WARNING: IF there is explicit User Feedback or JSON interaction data
from the user, YOU MUST prioritize using tools to fulfill that feedback
over any generic layout improvements.
```

Prompt

User Prompt:

```
Based on user needs, current scene status, and feedback:
1.  **Review Execution & Feedback**:
- Analyze the result of the last step.
- **Crucial**:  Detailedly review any "External User Feedback" or
"Evaluation Results" provided in the chat history.
- If there is specific external feedback, it implies the previous
result was not satisfactory.  You MUST prioritize fixing the issues
mentioned in the feedback.
- Identify the lowest GPT score in the evaluation.  What is the
specific problem?
2.  **Problem Identification**:
- Determine if previous problems were solved.
- Look for any bright red markings (e.g.  red boxes, arrows) in the
feedback image.  Before planning, FIRST describe their exact locations
based on the coordinate system mapping.
- Identify the most critical current problem (e.g., collisions, missing
objects, bad layout, or specific user requests).
- **User Input Overrides All**:  If the User provided JSON interaction
data or explicit text feedback, YOU MUST EXECUTE IT FIRST AND FOREMOST.
Ignore other layout additions until user commands are completely
fulfilled.
3.  **Tool Selection Strategy**:
- List applicable tools with confidence scores (0-1).
- **Visual Instructions**:  If an annotated image (with arrows, boxes,
or text) is provided, you **MUST** align your actions with these visual
cues.
- An arrow pointing to a location implies adding an object there or
moving something to there.
- A box or circle around an object often implies removing it or
modifying it.
- Use the provided 'layout' coordinates to map these visual markings to
specific object instances or 3D positions.
- **Self-Correction**:  If a tool failed previously or received
negative feedback, lower its score or change parameters.
- **Multi-Tool Execution**:  You are allowed to use **multiple tools**
in this step if they address different aspects of the problem (e.g.,
adding an object AND moving another).
- Ensure tool calls are logical and do not conflict.
4.  **Action**:
- Generate tool calls to solve the identified problems.
- If the scene is good and no further improvements are needed, use
valid `Terminate` tool.
Clearly explain your reasoning, specifically how you are addressing the
latest feedback.
```

Prompt

***Quality Inspector* Agent / Simulated LLM Feedback.** To simulate iterative user interaction, the framework optionally employs a *Quality Inspector* Agent that critiques the generated scene and proposes further refinement suggestions.

System Prompt:

```
You are given a top-down room render image and the corresponding layout
of each object.
Your task is to evaluate how well they align with the user's
preferences across the four criteria listed below.
For each criterion, assign a score from 0 to 10, and provide a brief
justification for your rating.
Scoring must be strict.  If any critical issue is found (such as
missing key objects, obvious layout errors, or unrealistic elements),
the score should be significantly lowered, even if other aspects are
fine.
**Score Guidelines**:
- Score 10:  Fully meets or exceeds expectations; no major improvements
needed.
- Score 5:  Partially meets expectations; some obvious flaws exist that
limit usefulness or quality.
- Score 0:  Completely fails to meet expectations; the aspect is absent,
wrong, or contradicts user needs.
**Evaluation Criteria**:
1.  **Realism**:  How realistic the room appears.  *Ignore texture,
lighting, and doors.*
- **Good (8-10)**:  The layout (position, rotation, and size) is
believable, and common daily objects make the room feel lived-in.  Rich
of daily furniture and objects.
- **Bad (0-3)**:  Unusual objects or strange placements make the room
unrealistic.
- **Note**:  If object types or combinations defy real-world logic
(e.g., bathtubs in bedrooms), score should be below 5.
2.  **Functionality**:  How well the room supports the intended
activities (e.g., sleeping, working).
- **Good (8-10)**:  Contains the necessary furniture and setup for the
specified function.
- **Bad (0-3)**:  Missing key objects or contains mismatched furniture
(e.g., no bed in a bedroom).
- **Note**:  Even one missing critical item should lower the score
below 6.
3.  **Layout**:  Whether the furniture is arranged logically in a good
pose and aligns with the user's preferences.
- **Good (8-10)**:  Each object is in **reasonable size**, neatly
placed, objects of the same category are well aligned, relationships
are reasonable (e.g., chairs face desks), sufficient space exists for
walking, and **orientations must** be correct.
- **Bad (0-3)**:  Floating objects, crowded floor, **abnormal size**,
objects with collision, incorrect **orientation**, or large items
placed oddly (e.g., sofa not against the wall).  Large empty space.
Blocker in front of furniture.
- **Note**:  If the room has layout issues that affect use, it should
not score above 5.
4.  **Completion**:  How complete and finished the room feels.
- **Good (8-10)**:  All necessary large and small items are present.
Has rich details.  Each shelf is full of objects (>5) inside.  Each
supporter (e.g.  table, desk, and shelf) has small objects on it.
Empty area is less than 50%.  The room feels done.
- **Bad (0-3)**:  Room is sparse or empty, lacks decor or key elements.
- **Note**:  If more than 30% of the room is blank or lacks detail,
score under 5.
```

Prompt

B.3 Available Assets

This paper adopts general-purpose, open-source 3D furniture assets, including Objaverse (Deitke et al., 2023), OpenShape (Liu et al., 2023b), 3D-FUTURE (Fu et al., 2021), and Infinigen (Raistrick et al., 2024). Specifically:

- **Objaverse** (Deitke et al., 2023): Objaverse is a large-scale open-source 3D asset dataset containing over 800,000 annotated 3D objects collected from diverse online repositories. It provides rich metadata, multiple mesh formats, and broad category coverage, making it widely used for 3D generative modeling, robotics simulation, neural rendering, and text-to-3D research.
- **OpenShape** (Liu et al., 2023b): OpenShape is an open-world 3D representation learning framework that aligns point clouds, images, and text into a unified embedding space through large-scale multimodal contrastive learning. By leveraging multiple 3D datasets and CLIP-aligned supervision, OpenShape enables strong zero-shot 3D understanding and cross-modal retrieval capabilities.
- **3D-FUTURE** (Fu et al., 2021): 3D-FUTURE is a large-scale repository of high-quality textured 3D furniture assets designed for indoor scene understanding and 3D reconstruction research. The dataset contains numerous detailed furniture 3D models with realistic textures and professionally designed indoor layouts, supporting tasks such as segmentation, pose estimation, retrieval, and texture recovery.
- **Infinigen** (Raistrick et al., 2024): Infinigen is an open-source procedural 3D scene generator for computer vision and graphics research. Built on Blender, it generates diverse photorealistic environments entirely from randomized mathematical rules without relying on external assets or generative AI, enabling scalable synthesis of high-quality 3D training data and automatic annotations.

B.4 Existing Tools and ATSs

Following the agentic design of I-OctoScene (Yang et al., 2026), we integrate 9 specialized tools into our framework to facilitate task execution and decision-making. The functionality of each tool is briefly described below.

- InitGPT: Uses GPT (Achiam et al., 2023) to generate an initial scene layout based on user requirements and supports arbitrary room types. Its strengths are strong alignment with user intent, rich details, and high flexibility. However, the generated layouts may be less realistic than those produced by data-driven approaches.
- InitPhyScene: Uses a network trained on the 3D-FRONT dataset (Fu et al., 2021) to generate an initial indoor scene, supporting living rooms, bedrooms, and dining rooms. The generated scenes are typically well-organized and contain high-quality assets, but they offer limited layout diversity and fewer details, often requiring subsequent editing tools to satisfy specific user requirements.
- AddObject: Uses GPT (Achiam et al., 2023) to add new objects to the current scene, including large furniture, small items placed on supporting surfaces, objects inside containers, and wall decorations. It cannot place objects on ceilings, and the precision of object orientation is limited.
- AddCrowd: Uses GPT (Achiam et al., 2023) for dense placement of small objects, enabling batch insertion of items on supporting surfaces or inside containers. It is effective for creating cluttered or densely populated environments, but placement accuracy is relatively low, execution can be slow, and existing objects may be affected.
- RemoveObject: Uses GPT (Achiam et al., 2023) to remove redundant, inappropriate, or problematic objects from a scene and applies to all room types. It excels at precisely removing specific objects to resolve collisions and overcrowding issues, but it cannot add or replace objects.
- AddRelation: Adds explicit spatial relationships between objects based on the current layout, such as placing objects against walls, facing each other, or stacking them vertically. This helps create more organized and coherent scenes. While effective at establishing logical object relationships, it cannot correct underlying layout or positioning issues.
- UpdateRotation: Uses GPT (Achiam et al., 2023) to adjust object orientations to improve spatial arrangement and functionality across all room types. It can correct improper facing directions and enhance functional coherence, but it only modifies rotations and does not move, add, or remove objects.
- UpdateLayout: Uses GPT (Achiam et al., 2023) to modify object positions and scales, making it suitable for major layout refinements in any room type. It is effective at adjusting the placement of specific objects, but cannot introduce new objects and may overlook existing spatial relationship constraints.
- UpdateSize: Uses GPT (Achiam et al., 2023) to adjust object dimensions, primarily for rescaling items on supporting surfaces whose proportions appear unrealistic. It can effectively correct object sizes to better satisfy functional requirements, but it cannot alter the overall dimensions of the room itself.

In the Intent-Aligned Tool Coordination mechanism (IATC), we present three commonly used ATSs:

- FILLEMPTYAREA: Triggered when a scene or region is identified as overly sparse or underutilized, this ATS is designed to efficiently and systematically populate the vacant space. It mainly consists of two steps: Add Large Furniture $p_1 \rightarrow$ Add Small Items p_2. The available tools are: $p_1 \in \{\text{ADDOBJECT}\}$ and $p_2 \in \{\text{ADDOBJECT}, \text{ADDCROWD}\}$.
- REPLACEASSET: Triggered when a piece of furniture in the scene does not satisfy the desired style, appearance, or category requirements, this ATS is adopted to replace the asset with a more suitable alternative. It mainly consists of two steps: Remove Furniture $p_1 \rightarrow$ Add New Furniture p_2. The available tools are: $p_1 \in \{\text{REMOVEOBJECT}\}$ and $p_2 \in \{\text{ADDOBJECT}\}$.
- SURFACEDECORATION: Triggered when the surface of a large furniture item appears overly empty and requires additional decorative objects, this ATS is adopted to reasonably enrich and decorate the vacant surface area. It consists of a single step: Add Small Crowd p_1. The available tool is: $p_1 \in \{\text{ADDCROWD}\}$.

These ATSs correspond to the tools most commonly used in existing 3DSS systems and are integrated into I-OctoScene. The framework is readily extensible, allowing users to incorporate additional ATSs tailored to their requirements, thereby supporting more flexible and customized tool execution workflows.

C INTERACTION EVALUATION BENCHMARK DETAILS

Due to space limitations, we only provide an introduction to the proposed *Interaction Evaluation Benchmark* in our main paper. In this section, we present the complete scoring details of the benchmark. Specifically, our *Interaction Evaluation Benchmark* consists of four evaluation dimensions: *Quality of the synthesized scene*, *Diversity of external interaction*, *Per-round intent adherence*, and *Final user query fulfillment*, as illustrated in Figure 8.

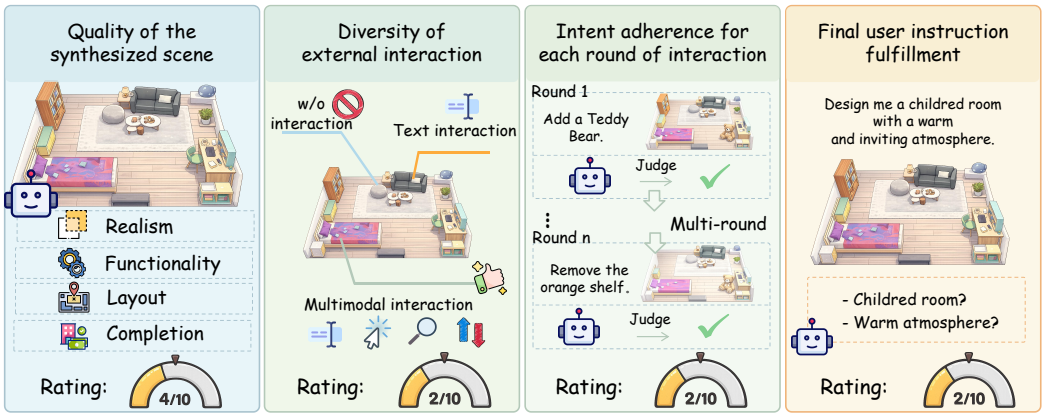


Figure 8: Overview of the four evaluation dimensions in our Interaction Evaluation Benchmark.

To be specific:

- **Quality of the synthesized scene**. Following the evaluation protocol of 3DSS (Yang et al., 2026), an expert LLM evaluates the generated top-down scene from four aspects: visual realism (*Real.*), functionality (*Func.*), layout correctness (*Lay.*), and scene completeness (*Comp.*). Each aspect is scored on a scale from 0 to 10. The final score is computed as:

$$\text{Quality Score} = \frac{\textit{Real.} + \textit{Func.} + \textit{Lay.} + \textit{Comp.}}{4} \times \frac{4}{10},$$

  resulting in a maximum of 4 points.
- **Diversity of external interaction**. This metric evaluates the capability of a method to accept external instructions during the generation process. A score of 2 points is assigned if the method supports multimodal interaction (*e.g.*, text and image inputs) during scene generation. A score of 1

point is assigned if the method only supports textual interaction. A score of 0 points is assigned if no progressive interaction is supported during generation.

- **Per-round intent adherence**. For each interaction round, an expert LLM determines whether the generated top-down scene satisfies the user instruction issued in that round. A score of 2 points is assigned if the instruction is substantially satisfied, 1 point if the method responds to the instruction but only partially fulfills it, and 0 points if the instruction is not addressed.
- **Final user query fulfillment**. After the final interaction round, an expert LLM summarizes all user requirements throughout the session and evaluates whether they are reflected in the final generated top-down scene. A score of 2 points is assigned if the overall requirements are substantially satisfied, 1 point if the method responds to the requests but only partially fulfills them, and 0 points if the requirements are not addressed.

More specifically, their prompts are as follows:

1) Quality of the synthesized scene: See the agent prompts above.

2) Diversity of external interaction: This score is determined by the interaction mode without requiring LLM evaluation: *none* receives 0 points, single-modal *LLM-simulated* feedback receives 1 point, and *real user* feedback receives 2 points.

3) Per-round intent adherence: For each round where external feedback is provided, GPT-4.1 evaluates whether the agent's action faithfully addressed the user's instruction for that specific round.

```
You are an evaluator.  Look at the provided top-down render of a room.
The user's instruction for THIS SPECIFIC ROUND was:  "{instruction}"
IMPORTANT CONTEXT:
 • The instruction might reference "red boxes", "red arrows", or other
   markings.
 • These markings were present on an input image provided to the
   previous agent.
 • The image you are looking at is the final result image!  ALL RED
   MARKINGS HAVE BEEN REMOVED!
 • DO NOT penalize the scene or give a score of 0 just because you
   cannot see the red markings in this output image!
 • Your job is to evaluate if the ACTUAL UNDERLYING ACTION was
   performed (e.g., was an object added, moved, or removed).  Be
   very lenient and flexible!  If a new object is present vaguely
   corresponding to the request, assume it correct.
Score from 0 to 2 (integer or float):
 • 2:  Fully satisfied or attempt is clearly visible.
 • 1:  Partially satisfied.
 • 0:  Completely ignored or completely failed.
Return pure JSON: {"score": 2, "comment": "short comment"}
```

Prompt

4) Final user query fulfillment: We first summarize all user demands and round-level interactions into a concise global description, then ask GPT-4.1 to judge whether the final scene satisfies the overall user intent.

Step 1 -- Demand Summarization:
Summarize the following user demands and interactions into 1-2 macroscopic, high-level sentences. Focus on the overall style, purpose, and main functions of the room. Ignore overly specific details, coordinate positions, or iterative fine-tuning steps.
Interactions: {combined_queries}

Step 2 -- Global Satisfaction Evaluation:
You are an evaluator. Look at the provided top-down render of a room.
The user's GLOBAL OVERALL DEMAND is: "{global_demand_summary}"
Evaluate how perfectly the final scene satisfies this overall demand.
Be lenient. Score from 0 to 2 (integer or float):

- 2: Basically totally meets the demand.
- 1: Partially meets the demand.
- 0: Fails to meet the demand.

Return pure JSON: {"score": 2, "comment": "short comment"}

Prompt

D Supplementary Evaluation Details

In this section, we provide additional details regarding the implementation and experimental settings used in the evaluation. Specifically, we first present supplementary implementation details of the proposed method and the evaluation.

The implementation of our models and tools (*e.g.*, asset retrieval, Stable Diffusion) is conducted on GeForce RTX 4090 GPUs, and all other LLM/VLM calls are made through the OpenAI API. Specifically, for fair comparison, following prior work (Yang et al., 2026), all agents and LLM-based modules and evaluation benchmarks are powered by OpenAI's GPT-4.1, using the snapshot version gpt-4.1-2025-04-14 accessed via the OpenAI API. In addition, our I-OctoScene adopts the OpenManus platform to implement the agentic framework, with particular emphasis on tool extensibility and the reason–act–react paradigm for 3DSS. The tool implementation follows and extends prior work (Yang et al., 2026). For detailed descriptions of the tools and ATSs, please refer to Appendix B.4.

For evaluation, under both common and open-vocabulary settings, we retrieve assets from Objaverse (Deitke et al., 2023) using OpenShape (Liu et al., 2023b) text embeddings. In the closed-vocabulary setting, depending on user requirements, we additionally leverage existing 3D-FUTURE (Fu et al., 2021) assets or employ the Infinigen generator (Raistrick et al., 2024) to produce standard assets, following (Çelen et al., 2024). Notably, following Sun et al. (2025), we employ GPT-4 to evaluate these benchmarks by providing it with the query and top-down scene renderings as input. Specifically, each participant is given a randomized questionnaire of rendered scenes with no identifying information disclosed, and is asked to score them according to the benchmark criteria and indicate their preferred ranking. To ensure a fair comparison of agentic methods by controlling for the effects of assets and tool availability, both SceneWeaver (Yang et al., 2026) and I-OctoScene were reproduced under the following unified settings. The results of the remaining baselines are directly adopted from (Yang et al., 2026). Following the evaluation protocol of SceneWeaver (Yang et al., 2026), we perform 10 independent synthesis runs for common room categories (*e.g.*, bedrooms and living rooms) and report the average results. For more open-ended room categories, such as gyms, waiting rooms, and restaurants, we perform 3 runs and report the mean performance. All ablation experiments are also repeated 3 times, with average results reported.

E More Additional Experiments

In this section, we provide additional experiments to conduct a more comprehensive and in-depth investigation of our I-OctoScene. Specifically, we first conduct a human evaluation and analyze the time and token costs associated with I-OctoScene. We then quantify the scene scoring process and present additional instruction-response cases along with supplementary visualizations. Finally, we

Table 5: Human comparative evaluation in terms of visual realism (Real.), functionality (Func.), layout correctness (Lay.), scene completeness (Comp.), and overall preference ranking (Rank).

Method	Bedroom					Living Room				
	Real. ↑	Func. ↑	Lay. ↑	Comp. ↑	Rank ↓	Real. ↑	Func. ↑	Lay. ↑	Comp. ↑	Rank ↓
IDesign	5.7	6.7	6.0	6.5	3.0	6.4	6.2	6.6	6.3	2.9
SceneWeaver	8.0	8.1	8.0	7.2	2.0	7.6	7.4	6.1	7.8	1.9
Ours	**8.9**	**9.0**	**8.6**	**8.1**	**1.0**	**8.2**	**8.9**	**7.8**	**8.5**	**1.2**

further perform open-vocabulary case studies to gain a deeper understanding of the open-vocabulary capabilities of our I-OctoScene.

E.1 Human Comparative Analysis

To ensure generality, we further conduct a human evaluation, in which physical realism, visual quality, and query satisfaction are comprehensively assessed via scoring and ranking by 10 users. Specifically, the users followed the same evaluation criteria as the LLM agent, with scores assigned on a 10-point scale. The ranking was recorded on a scale from 1 to 3, where a smaller value indicates a higher rank. More details of the questionnaire are provided in Figure 9.

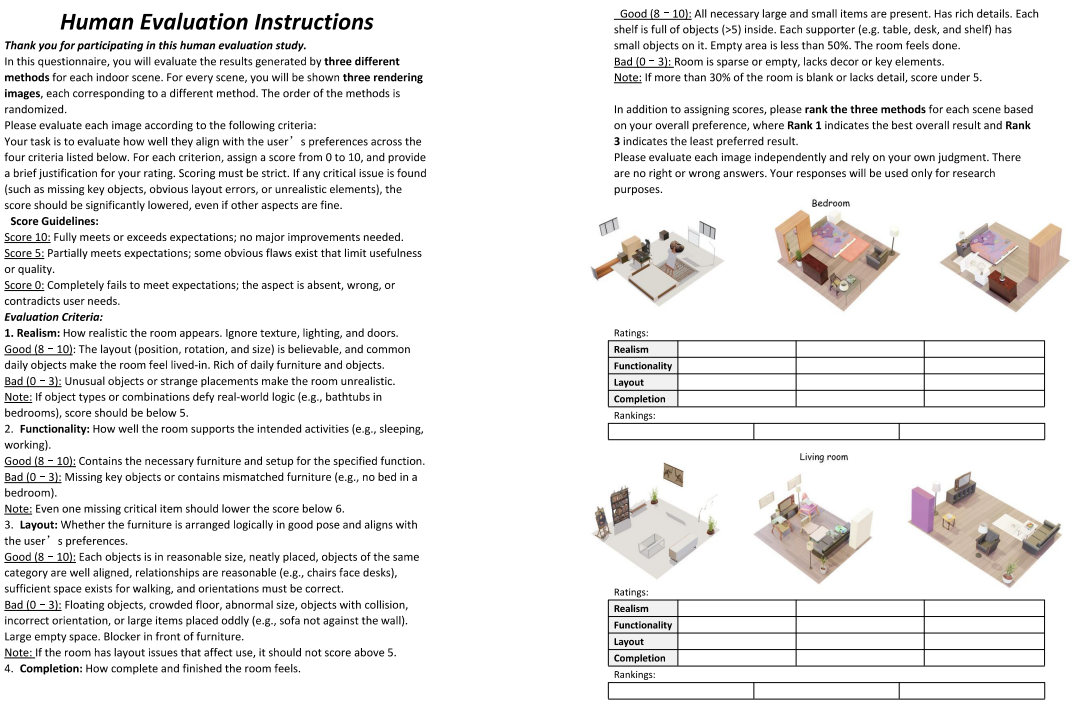


Figure 9: Questionnaire used in the human comparative analysis.

More specifically, Table 5 depicts the average human scoring and ranking for two representative room types: bedroom and living room. From the results, we can observe that:

- **Broad Human Recognition:** Beyond achieving superior performance on LLM-based benchmarks, our I-OctoScene also attains the highest human evaluation score, with an average of 8.65 in the bedroom setting, outperforming the existing agentic baseline (*i.e.*, SceneWeaver) by nearly 1 point. This demonstrates the effectiveness of our multi-agent framework and coordinated tool orchestration in 3DSS.
- **Stronger Human Preference:** Our I-OctoScene consistently achieves 1*st* rank in human ranking. It outperforms other baselines across both room types, demonstrating that the introduced PMIR can progressively interact and adapt to users' inherent preferences for different scenes.

E.2 Cost Analysis

In this section, we investigate the computational cost of I-OctoScene, further demonstrating its practicality for real-world applications. On the one hand, when access to an LLM API is available, our framework requires only a machine with minimal GPU resources to accomplish 3D scene design and synthesis (when the Stable Diffusion Tool is not used). To evaluate its efficiency, we compare the time and token consumption across different synthesized stages (with Art Director, Interior Designer, Engineer, and Quality Inspector), including consultation, design, tool engineering, as well as total cost. The results are summarized in Table 6. On the other hand, we report the approximate number of tokens consumed by I-OctoScene in a single interaction round, as shown in Table 7.

Table 6: Approximate time consumption of different development stages per round of our I-OctoScene.

Stage	Consultation	Design	Tool Engineering	Total
Initial Round	2 s	5 s	5 min	13 s + 5 min
Add Round	3 s	6 s	4–15 min	20 s + 4–15 min
Update Round	5 s	20 s	4 min	25 s + 4 min

Table 7: Approximate token consumption ($\times$ 1 k) per round of our I-OctoScene.

Instruction Type	Consultation	Execution	Evaluation
Visual Instruction	7	10–50	6
System/Linguistic Instruction	4	10–50	6

From the experimental results, we make the following observations. **1)** Compared with human designers, who typically spend several hours conceptualizing a scene and additional hours using professional software for scene design and synthesis, I-OctoScene is able to generate a rich and instruction-compliant 3D scene in approximately one hour, with each interaction round taking no more than 10 minutes on average and most rounds being completed within 5–8 minutes. **2)** I-OctoScene requires only approximately 100–300k tokens to complete the design of a single scene. Based on our implementation using GPT-4.1 (gpt-4.1-2025-04-14) accessed via the OpenAI API, whose pricing is \$2 per million input tokens and \$8 per million output tokens, the corresponding API cost is approximately \$0.4–\$1.2 USD per scene. Compared with the labor, hardware, and software expenses associated with manual scene design, I-OctoScene offers substantial cost savings while maintaining high-quality scene generation.

E.3 Quantification of Scene Scoring Process

In this section, we provide a more in-depth analysis of the multi-round iterative design process of our I-OctoScene. Specifically, we quantitatively evaluate the first four rounds of the three-stage 3DSS process for two room categories, namely bedrooms and living rooms, in terms of four metrics: visual realism (*Real.*), functionality (*Func.*), layout correctness (*Lay.*), and scene completeness (*Comp.*). In addition, we visualize the top-down view of a representative multi-round bedroom design process. The corresponding results are presented in Figures 10 and 11 and table 8, respectively.

The experimental results reveal a clear trend: nearly all evaluation metrics improve consistently across successive design iterations, demonstrating the effectiveness of our multi-agent and multi-round framework. More specifically, the weaker aspects in each round (*i.e.*, the dimensions with lower scores) show noticeable improvement in the subsequent round. In particular, scene completeness shows substantial gains in the second or third round as additional assets are progressively introduced into the scene. Moreover, the newly added assets are arranged in a coherent and well-structured manner, improving scene completeness while preserving functionality and layout consistency.

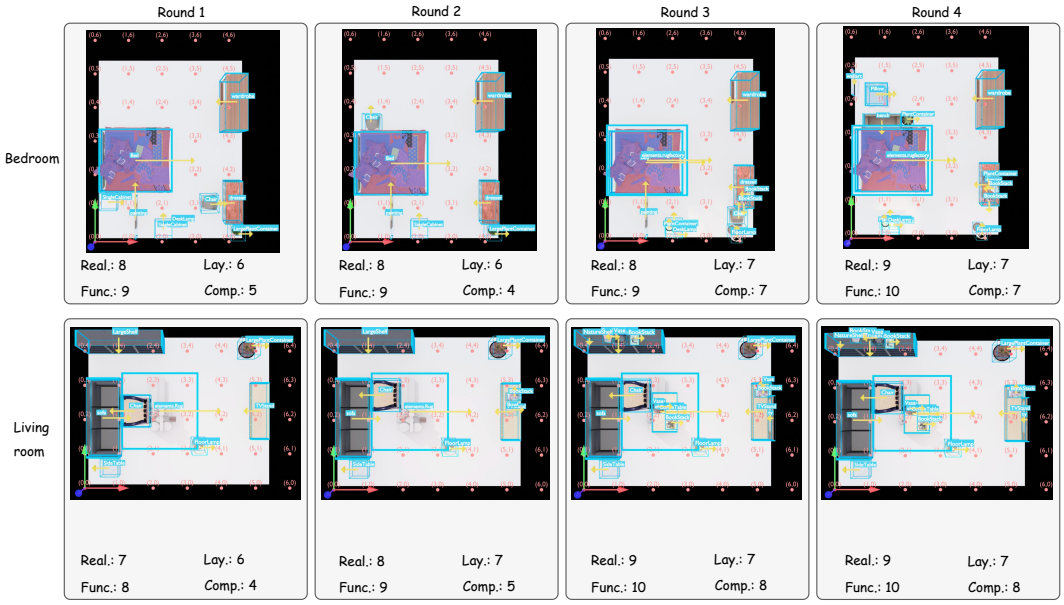


Figure 10: Top-down view visualizations of the multi-round iterative process for common room type settings (bedroom and living room), together with changes in visual realism (Real.), functionality (Func.), layout correctness (Lay.), and scene completeness (Comp.) scores.

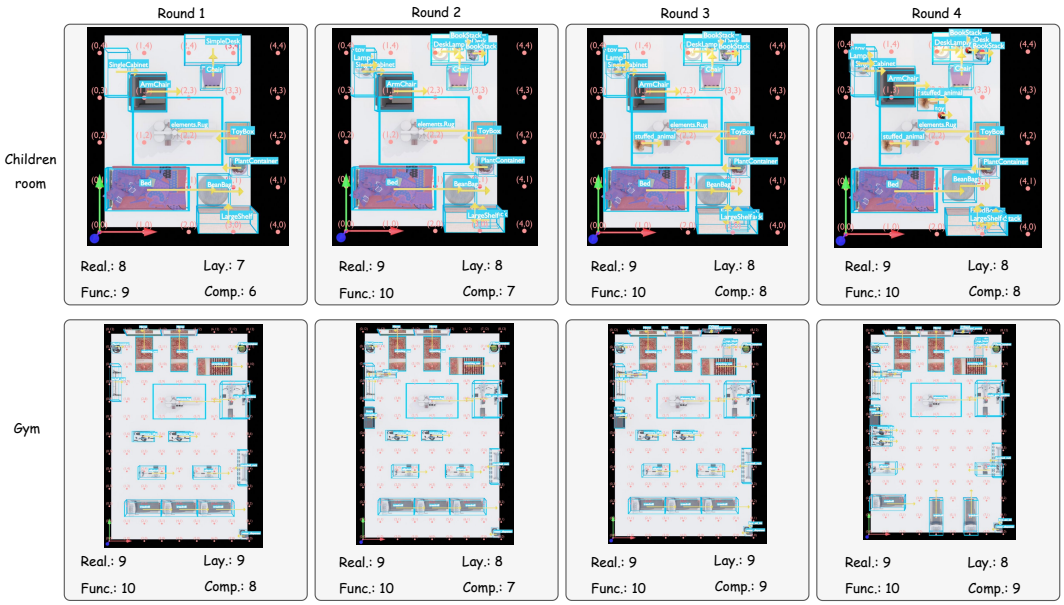


Figure 11: Top-down view visualizations of the multi-round iterative process for open-vocabulary room type settings (children room and gym), together with changes in visual realism (Real.), functionality (Func.), layout correctness (Lay.), and scene completeness (Comp.) scores.

Table 8: Scores of multi-round iterations on visual realism (*Real.*), functionality (*Func.*), layout correctness (*Lay.*), and scene completeness (*Comp.*). Specifically, the upper half of the table presents results with an LLM serving as the *External Instructor*, while the lower half presents results with a human user serving as the *External Instructor*. Δ denotes the score change between two consecutive rounds; negative changes are highlighted in red, and positive changes in green.

	Bedroom				Living Room			
	Real.	Func.	Lay.	Comp.	Real.	Func.	Lay.	Comp.
w/ LLM:								
Round 1	6.3	7.7	6.7	4.0	7.0	8.3	7.0	4.0
Δ	+2.0	+1.0	-0.3	+2.0	+1.3	+0.7	+0.3	+2.0
Round 2	8.3	8.7	6.3	6.0	8.3	9.0	7.3	6.0
Δ	-0.3	+0.3	0.0	+0.3	0.0	+0.3	-1.0	+0.7
Round 3	8.0	9.0	6.3	6.3	8.3	9.3	6.3	6.7
Δ	+0.7	+0.7	+1.0	+0.7	+0.7	+0.3	+0.7	+1.3
Round 4	8.7	9.7	7.3	7.0	9.0	9.7	7.0	8.0
w/ User:								
Round 1	7.7	9.0	7.3	5.0	7.0	8.0	6.3	4.0
Δ	+0.3	+0.0	-0.7	+2.0	+0.7	+0.7	+1.3	+0.7
Round 2	8.0	9.0	6.7	7.0	7.7	8.7	7.7	4.7
Δ	+0.7	+0.3	+1.0	+0.3	+0.7	+0.3	-0.3	+2.0
Round 3	8.7	9.3	7.7	7.3	8.3	9.0	7.3	6.7
Δ	+0.3	+0.7	+0.3	+1.0	+0.7	+0.5	+0.7	+0.3
Round 4	9.0	10.0	8.0	8.3	9.0	9.5	8.0	7.0

E.4 Instruction Response Case

In this section, we present a series of complex requirements and visual instructions proposed by external users, together with the corresponding responses and follow-up actions generated by I-OctoScene, as shown in Figures 12 and 13. Compared with the relatively simple instruction-following scenarios presented in the main paper, these examples are intentionally designed to cover more challenging cases involving multiple intent points, heterogeneous task requirements, and multimodal inputs, thereby providing a more comprehensive evaluation of the practical capabilities of I-OctoScene.

From the experimental results, we observe that:

1) Beyond the relatively simple external instructions discussed earlier, I-OctoScene is capable of comprehensively identifying all intent points contained in complex user requirements and invoking appropriate tools or skills (ATSs) to address them in a coordinated manner. Rather than treating a complicated request as a single monolithic task, I-OctoScene can automatically decompose it into multiple sub-goals, establish the dependencies among them, and execute the corresponding ATSs in a logical sequence. This demonstrates not only its ability to understand complex instructions but also its capability to perform multi-step reasoning and planning, enabling it to generate responses that remain consistent with the overall user intention while satisfying individual sub-requirements.

2) When the external interaction is provided in the form of visual images, I-OctoScene can effectively interpret the visual guidance before performing the required scene modifications. Specifically, it is able to accurately detect the red bounding boxes contained in the input images, estimate their approximate spatial locations, and leverage these visual cues to identify the target regions for subsequent editing. Based on the semantic instructions associated with the visual annotations, I-OctoScene further refines the scene by performing the required modifications while preserving the consistency and coherence of the remaining content. These results indicate that the proposed framework is capable of jointly reasoning over textual and visual information, allowing it to seamlessly integrate multimodal instructions into a unified pipeline.

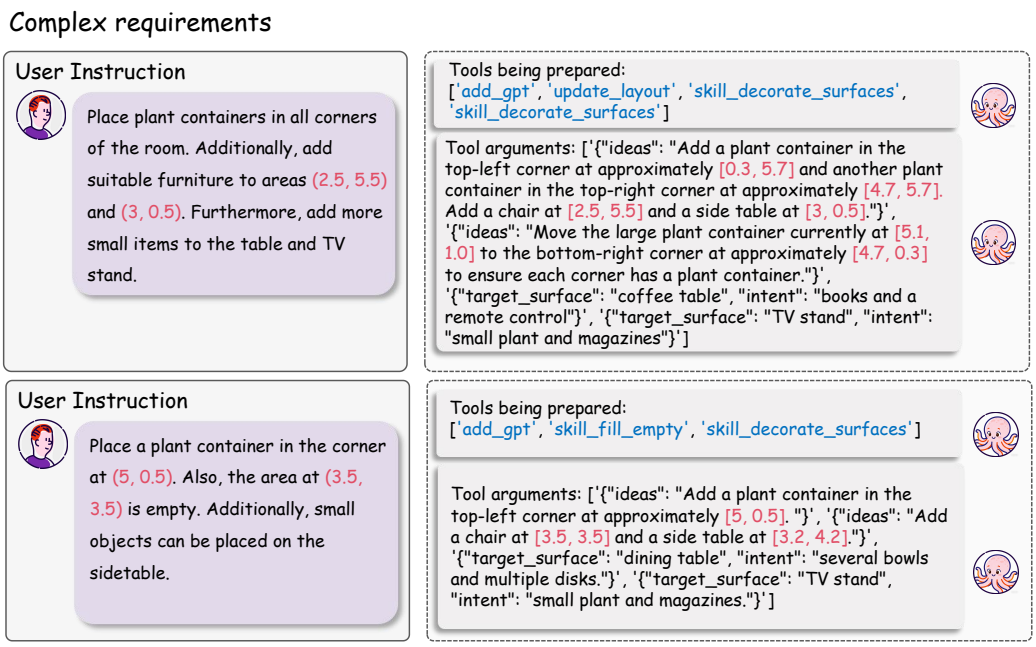


Figure 12: Illustration of I-OctoScene's actions when handling complex requirements. This includes the tools it intends to invoke and the corresponding tool arguments. The purple boxes represent the user's instructions, while the gray boxes represent I-OctoScene's action outputs.

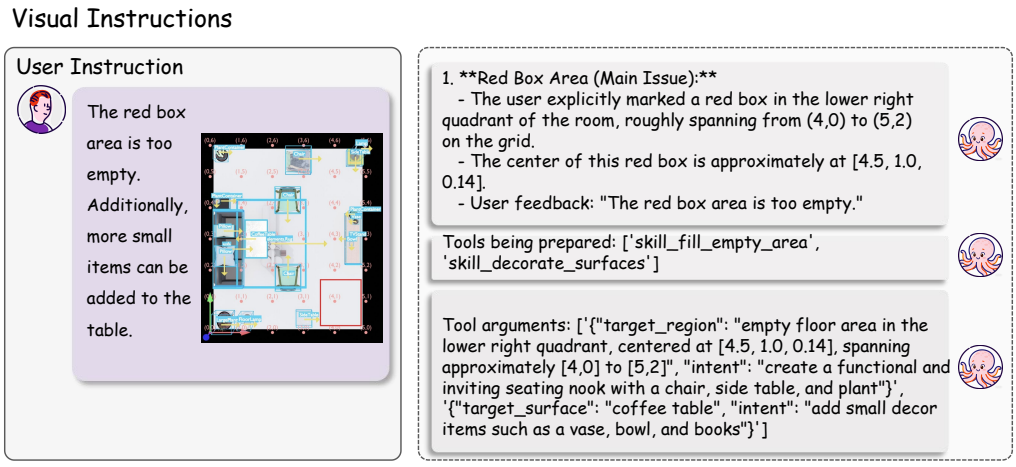
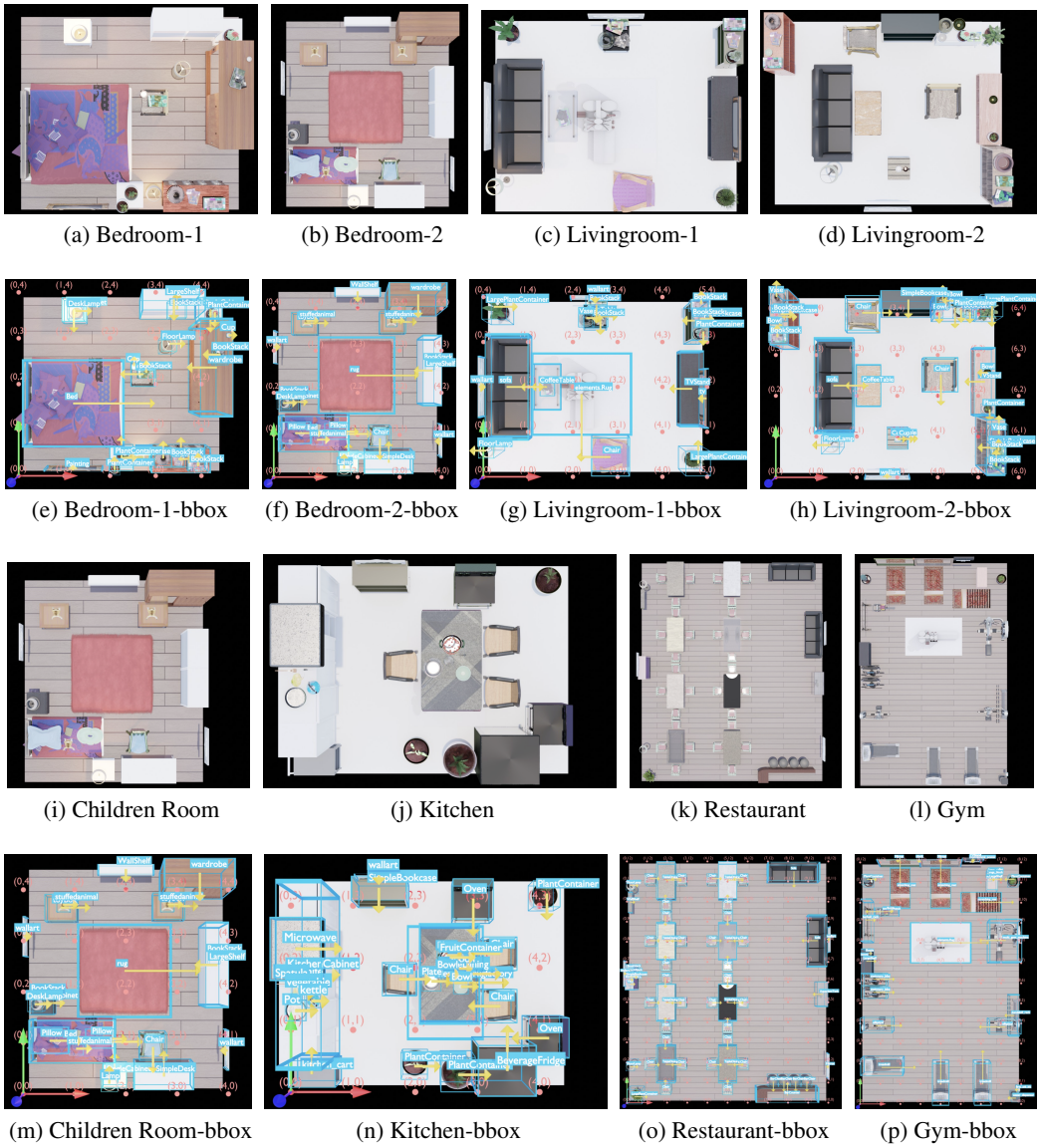


Figure 13: Illustration of I-OctoScene's actions when handling visual instructions. This includes the tools it intends to invoke and the corresponding tool arguments. The purple boxes represent the user's instructions, while the gray boxes represent I-OctoScene's action outputs.

E.5 Supplementary Visualizations

Top-Down Scene Visualization. To provide a more comprehensive and general understanding of the 3D scene synthesis performance, we visualize the top-down views of several synthesized scenes (including both common and open-vocabulary scenes) during the generation process, as well as their top-down views annotated with bounding boxes, as depicted in Figure 14. From these visualizations, we can observe that:

Blender Project Visualization. In this section, to provide a more comprehensive understanding of the 3D scene visualization process, we present several visual elements from the Blender rendering



(a) Bedroom-1 (b) Bedroom-2 (c) Livingroom-1 (d) Livingroom-2

(e) Bedroom-1-bbox (f) Bedroom-2-bbox (g) Livingroom-1-bbox (h) Livingroom-2-bbox

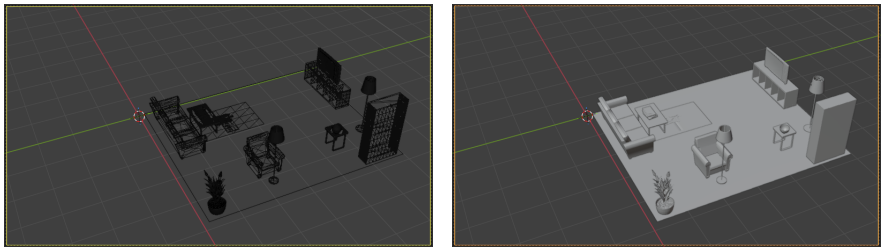
(i) Children Room (j) Kitchen (k) Restaurant (l) Gym

(m) Children Room-bbox (n) Kitchen-bbox (o) Restaurant-bbox (p) Gym-bbox

Figure 14: Top-view visualization of 3DSS. Specifically, the first and third rows show top-down views of the rooms, and the second and fourth rows present the corresponding top-down views with object bounding box annotations.

pipeline, including the project preview interface and the world-environment lighting backgrounds (*i.e.*, *Interior* and *Forest*). More detailed technical procedures and implementation details will be fully released once the double-blind review process is completed and the codebase is made publicly available. Specifically, we first showcase the wireframe, entity, material, and rendering views of the scenes within the Blender project, as illustrated in Figure 15. In addition, we present the two world-environment lighting settings adopted during visualization, as shown in Figure 16.

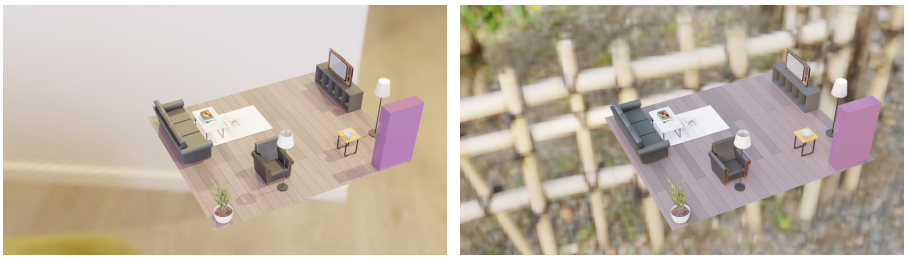
More Scene Case Visualization. In this section, we provide additional 3D scene synthesis (3DSS) cases to offer a more comprehensive and in-depth demonstration of I-OctoScene, as shown in Figures 17 and 18. Specifically, these cases cover a wider variety of scene categories and layouts. From the visualized results, we make the following observations. **1)** Although I-OctoScene achieves the highest quantitative scores, the generated scenes also exhibit realistic and naturally cluttered arrangements of small everyday objects, which contribute to greater scene authenticity. **2)** The small



(a) Wireframe (b) Entity

(c) Material (d) Rendering

Figure 15: The four Blender 5.0 viewport display modes (*i.e.*, *Wireframe*, *Solid*, *Material Preview*, and *Rendered*) are shown from left to right.



(a) Interior (b) Forest

Figure 16: The two world-environment lighting backgrounds provided by Blender 5.0 and used in our paper (*i.e.*, *Interior* and *Forest*) are better suited for indoor 3D scene design and synthesis.

objects and furniture introduced by I-OctoScene are consistently aligned with the specified room type settings, without exhibiting unreasonable or semantically inconsistent object placements. **3)** Across all cases, scenes generated by I-OctoScene rarely suffer from physical implausibility, such as object overlaps, boundary violations, or other spatial inconsistencies.

More Visualizations of the Scene Synthesis Process. Due to space limitations, we only present a single scene synthesis process case in our main paper. In this section, we provide additional scene synthesis process cases to offer a more comprehensive reference, as illustrated in Figures 19 and 20. Specifically, the showcased cases involve diverse interactions and the use of multiple tools, providing a more complete understanding of the operational workflow of I-OctoScene. These examples further demonstrate how I-OctoScene leverages agentic interactions and tool-assisted reasoning to progressively refine scene layouts and satisfy user requirements throughout the synthesis process.

E.6 Other Open-Vocabulary Case Analysis

In this section, we present additional and more challenging open-vocabulary 3D indoor scene synthesis cases. Specifically, we first visualize two scene settings that have not appeared in previous studies: a bathroom and a dancing hall. For each scene, we show the 3D scene from the initial first



Figure 17: Additional 3DSS case visualizations of a children's room and a gym. The corresponding initial queries are: *Design me a children room.* and *Design me a gym.*

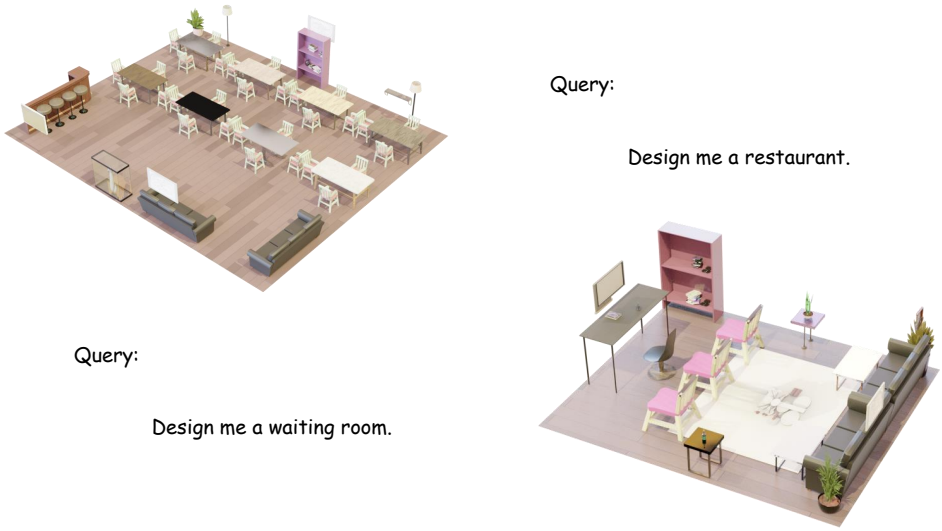


Figure 18: Additional 3DSS case visualizations of a restaurant and a waiting room. The corresponding initial queries are: *Design me a restaurant.* and *Design me a waiting room.*

iteration and the final fifth iteration, and the top-down views from the final iteration, as illustrated in Figures 21 and 22. Furthermore, we visualize several scene synthesis cases that initially involve complex requirements, as shown in Figures 23 and 24.

From these results, we draw the following three key observations: **1)** Regardless of the scene requirements, I-OctoScene consistently demonstrates strong 3D scene synthesis (3DSS) capabilities. **2)** I-OctoScene exhibits remarkable open-vocabulary generalization ability. Even for previously unseen scene categories, such as bathrooms and dancing halls, it can generate plausible layouts and introduce numerous scene-specific objects. **3)** I-OctoScene is capable of understanding complex 3D scene synthesis requirements and effectively preserves the original user intentions throughout the agentic iterative optimization process, without losing or drifting from the specified constraints.

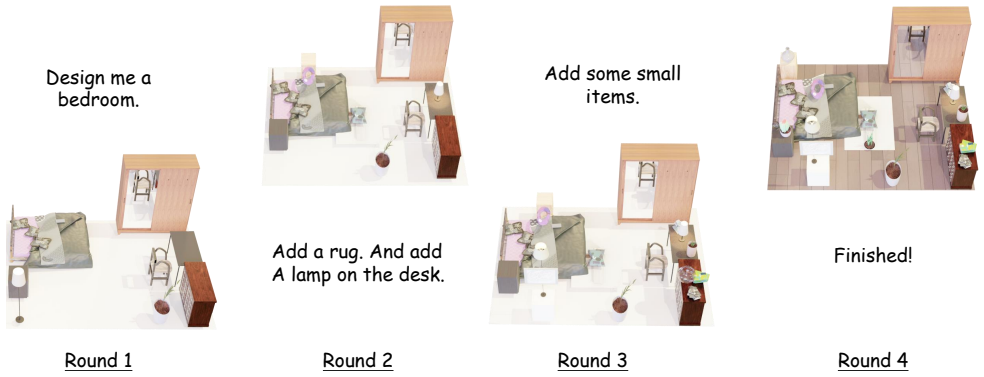


Figure 19: The synthesis process of our I-OctoScene in a bedroom environment. This synthesis was guided entirely by linguistic instructions, without the use of visual instructions as references.

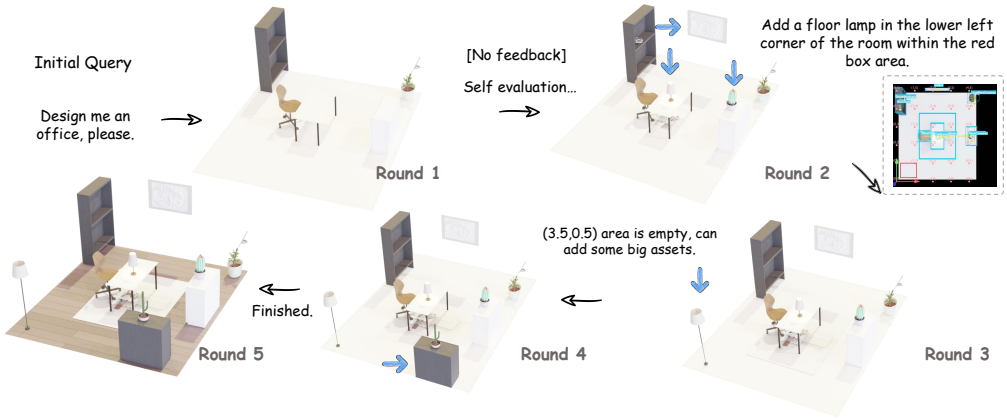


Figure 20: The synthesis process of our I-OctoScene in an office environment. Both linguistic and visual instructions were provided as references. In the first round, no external interactive feedback was available; I-OctoScene autonomously conducted scene design and refinement through self-evaluation.

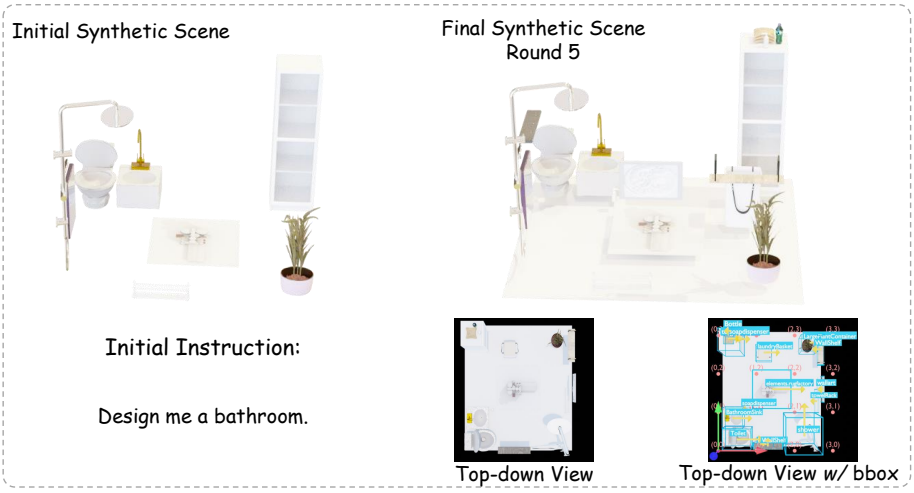


Figure 21: Open-vocabulary 3DSS case in bathroom setting.

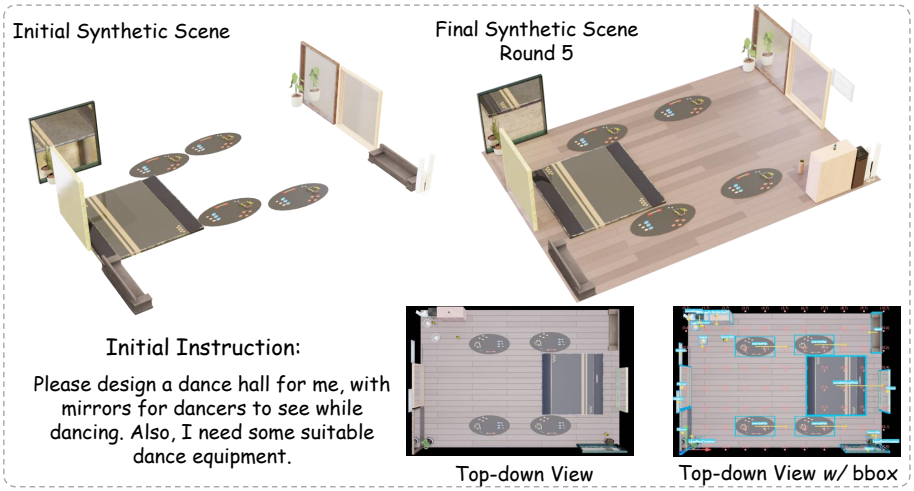


Figure 22: Open-vocabulary 3DSS case in dancing hall room setting.

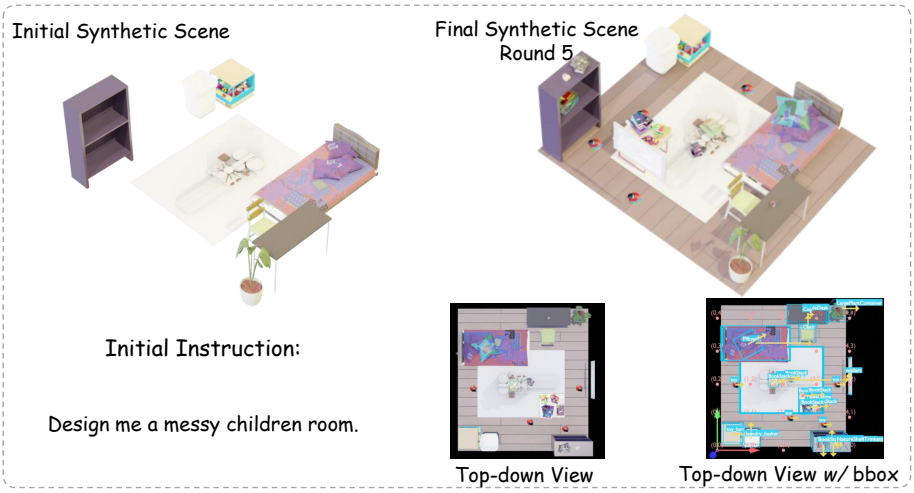


Figure 23: Open-vocabulary 3DSS case in children room room setting with a complex query.

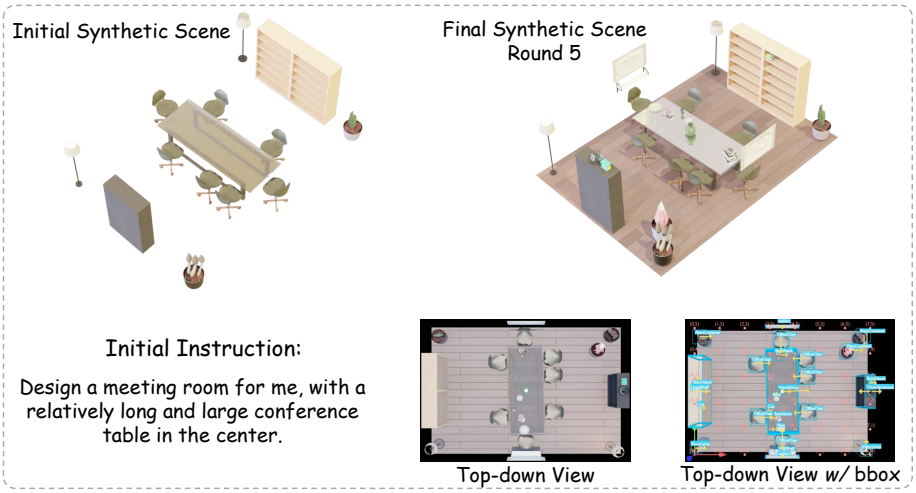


Figure 24: Open-vocabulary 3DSS case in meeting room setting with complex query.

F ETHICS STATEMENT

This work adheres to the ICLR 2027 Code of Ethics. We study 3D scene synthesis through multi-agent collaboration and evaluate it on existing benchmarks, without conducting human-subject studies or animal experiments.

G REPRODUCIBILITY STATEMENT

We follow the ICLR 2027 reproducibility guidelines. Implementation and evaluation details of our I-OctoScene are provided in *Experiment* Section of the main paper and our *Appendix*. We will release our code upon acceptance.