GALA: Geometry-Aware Language Model for Controllable Object Arrangement

ANONYMOUS AUTHOR(S)
SUBMISSION ID: 1948

Fig. 1. Left: input objects and fine-grained placement instructions. Middle: plausible layouts that follow the instructions. Right: zoom-in views show how GALA goes beyond oriented bounding boxes by perceiving geometry to produce correct support, containment, and collision-free placements.

Reasoning about fine-grained placement constraints, such as support and containment, is essential for realistic 3D scene creation. It remains a significant challenge for existing methods that rely on simplified object representations like oriented bounding box (OBB), which are blind to the detailed geometry. To address this limitation, we introduce a geometry-aware language model with explicit point cloud perception for controllable object placement. Our model performs autoregressively by predicting the 6D pose of one target object at a time conditioned on the scene context and placement instruction. We utilize a two-stage training strategy: the first stage aligns geometric and textual modalities through pretraining tasks, and the second stage fine-tunes the model for placement. To facilitate this task, we also curate a multi-source scene dataset covering diverse room types and fine-grained furniture arrangements, which will be open-sourced for community benefit. Experiments on our benchmark show our model significantly outperforms baselines without explicit geometric reasoning. We further develop a multi-stage plan–place–verify agent pipeline that uses the model for text-driven scene generation.

CCS Concepts: • **Computing methodologies** → **Artificial intelligence**; **Natural language processing**; **Computer graphics**; **Scene understanding**.

Additional Key Words and Phrases: object placement, multimodal model, point cloud, scene layout

ACM Reference Format:
Anonymous Author(s). 2026. GALA: Geometry-Aware Language Model for Controllable Object Arrangement. *ACM Trans. Graph.* 1, 1 (June 2026), 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM 1557-7368/2026/6-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Generating realistic 3D scenes is important in game development, film production, architectural design, and embodied AI training, where virtual environments must be both physically plausible and semantically meaningful. Scene construction can be viewed as an iterative process of single-object placement: each step inserts a new object conditioned on the scene built so far. This single-object placement problem is the focus of this work. Placing a new object requires understanding both high-level semantic constraints, such as user instructions and object relationships, and low-level geometric constraints, such as collision avoidance, support, and containment.

However, most existing methods place objects only at the level of oriented bounding box (OBB): each object is represented by its center, size, and rotation. This abstraction is sufficient for coarse arrangements, but it struggles with the geometric constraints in realistic scenes, as shown in Fig. 1. Three basic failure modes are common in practice. First, a chair placed under a desk must clear the desktop with its seat while avoiding the table legs. Second, small objects like books are often placed inside shelves whose internal cavities are invisible to the OBB. Third, cushions placed on curved

surfaces, such as sofas, require reasoning about support heights that vary across the contact region. In all three cases, OBB-only methods cannot encode the fine-grained constraints, leading to either collisions (in OBB space), floating, or misalignments with the underlying geometry.

We view this limitation as primarily a geometry-awareness problem rather than an algorithmic one. A natural solution is to bring geometric information into the placement problem through a 3D representation of the objects and the scene. Among possible 3D representations, point clouds offer a favorable trade-off: they are lightweight, easy to obtain, and uniform across asset libraries, and recent work on 3D multimodal LLMs [Hong et al. 2023; Mao et al. 2026; Xu et al. 2024] shows that LLMs can consume point clouds directly. Our core hypothesis is simple: LLM-based placement models augmented with point clouds can handle fine-grained arrangements beyond OBB reasoning.

Building on this hypothesis, we introduce a geometry-aware placement model. The model is a multimodal LLM built on a strong LLM backbone, extended with a point cloud encoder and a projector that maps geometry features into the LLM embedding space. Given an existing scene, an object to be placed, and a placement instruction, the model predicts the 6D pose of the target object. The instruction specifies the intended relation and makes object insertion controllable. To make the point cloud tokens readable to the LLM, we adopt a two-stage training recipe: geometric pretraining aligns the projector, and task fine-tuning adapts the full model for object placement. Existing scene datasets, such as 3D-FRONT [Fu et al. 2021], contain noisy scenes and provide limited examples of fine-grained placement. We therefore curate a multi-source scene dataset from 3D-FRONT, Imaginarium [Zhu et al. 2025], SpatialLM [Mao et al. 2026], and an artist-crafted set of 300 scenes, followed by filtering noisy layouts and adding fine-grained placement examples. Finally, to make our model easier to use and support automated scene generation, we develop a multi-stage agent pipeline that converts textual user requirements into complete 3D scenes through object planning, instruction generation, and geometry-aware placement.

We summarize our contributions as follows.

- We introduce a geometry-aware language model for controllable object pose prediction conditioned on placement instruction and point cloud, enabling fine-grained placement beyond OBB-based reasoning.
- We curate a multi-source scene dataset, combining public scenes and artist-crafted scenes. The dataset is physically and visually verified, contains fine-grained placement samples with complex spatial relations, and will be open-sourced for community benefit.
- We design an agent system for natural language scene creation with geometric reasoning, paving the way for automatic scene generation.

2 Related Work

2.1 Data-driven Scene Synthesis

Early learning-based approaches represent an indoor scene as a set of oriented bounding boxes and learn the joint distribution of object classes and box parameters, including convolutional or recursive-prior methods [Li et al. 2019; Wang et al. 2019, 2018], autoregressive models [Bucher and Armeni 2025; Feng et al. 2026; Paschalidou et al. 2021], and diffusion-based models [Tang et al. 2024]. Other lines of work model scenes as graphs and predict pairwise relations [Gao et al. 2023; Lin and Mu 2024; Zhai et al. 2024, 2023; Zhou et al. 2019]. These methods reason at the OBB level, so they cannot express the geometric detail needed to slide a chair under a table or place a book on a specific shelf inside a cabinet. They also tie generation to the scene dataset distribution, which limits generalization to unseen room types.

2.2 LLM- and VLM-based Scene Layout

A recent line of work treats LLMs and VLMs as priors over indoor scenes. LayoutGPT [Feng et al. 2023] prompts an LLM with in-context examples to output object positions and sizes from a scene description, while LayoutVLM [Sun et al. 2025] leverages the visual abilities of modern VLMs to generate scenes from text. Holodeck [Yang et al. 2024] decomposes scene generation into LLM-driven planning, asset retrieval, and constraint-based placement. Because these model outputs are still coarse layouts, several methods add explicit optimization or solvers after the language-model stage. [Gumin et al. 2025; Wu et al. 2026] generate indoor scenes through LLM-generated programs and layout optimization, and Co-Layout [Xiang et al. 2026] couples an LLM-driven agent with grid-based integer programming to jointly optimize room layouts and furniture placements. FirePlace [Huang et al. 2025a] similarly uses VLM-generated constraints and a solver for object placement. Closer to our setting, PlaceIt3D [Abdelreheem et al. 2025] trains a 3D LLM for language-guided placement on reconstructed scanned scenes, but predicts a placement mask rather than a numeric 6D pose. These methods show that pretrained foundation models carry useful priors over spatial common sense, but their geometric reasoning is mostly externalized into boxes, constraints, rules, solvers, or discrete placement masks. As a result, fine-grained placements that depend on object shape are not handled through native 3D geometric perception.

2.3 3D Multimodal Large Language Models

Recent 3D multimodal LLMs extend language models from image-text reasoning to point-cloud and scene-level understanding. Early systems such as 3D-LLM [Hong et al. 2023] and PointLLM [Xu et al. 2024] show that LLMs can describe 3D content, answer questions, and follow instructions when supplied with 3D inputs. This direction has since expanded to more structured 3D tasks: 3D-LLaVA [Deng et al. 2025] studies general 3D instruction following, SegPoint [He et al. 2024] and Reason3D [Huang et al. 2025b] use LLMs for point-level segmentation from referring or reasoning queries, and SpatialLM [Mao et al. 2026] targets structured indoor scene understanding. Recent 3D VLM work further studies spatial reasoning itself, such as SpatialStack [Zhang et al. 2026b], which focuses on point-language reasoning for 3D spatial relations. These works make 3D geometry accessible to foundation models, but their outputs are mostly language responses, masks, or structured scene descriptions

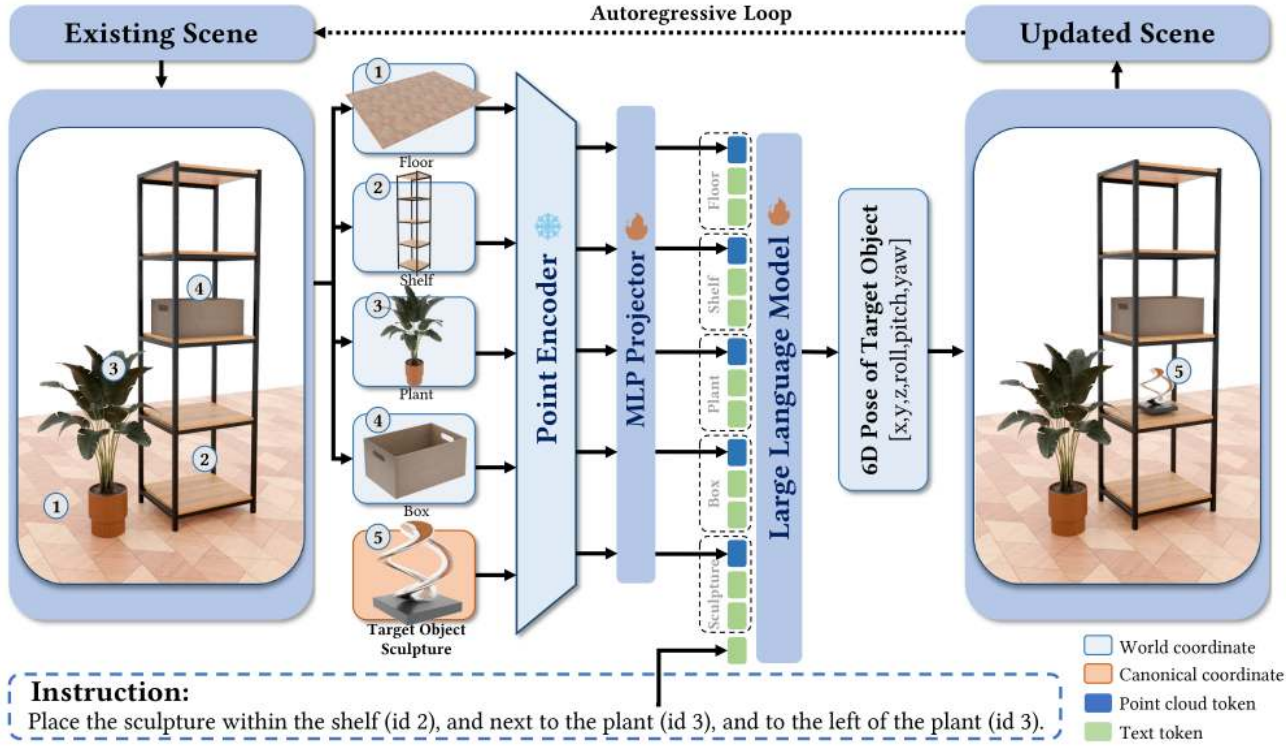

Fig. 2. Overview of the geometry-aware placement model. Given the scene boundary, the already-placed objects (with poses, sizes, semantic categories, and point clouds), the new object (with size, semantic category, and point cloud), and a placement instruction, the multimodal LLM outputs a numeric 6D pose. The model performs placement autoregressively, adding each predicted object to the scene context before predicting the next one.

rather than precise poses for inserting a new object into an existing scene. Our work uses the 3D multimodal formulation for object placement, where the model must map scene geometry, target object geometry, and a placement instruction to a numeric 6D pose.

3 Method

Problem Formulation. We study a single-object placement task in a bounded scene region. The scene region is described by a 2D footprint polygon $\mathcal{B} \subset \mathbb{R}^2$. A scene consists of a set of existing object instances $\boldsymbol{O}_{\text{exist}} = \{O_i\}_{i=1}^{N}$. Each existing object O_i is associated with an identifier i, a semantic category string c_i, an axis-aligned size $\boldsymbol{s}_i = (w_i, d_i, h_i)$, and a point cloud P_i. Once placed, it has a 6D pose $\boldsymbol{p}_i = (x_i, y_i, z_i, \phi_i, \theta_i, \psi_i)$, where (x_i, y_i, z_i) denotes the translation, and $(\phi_i, \theta_i, \psi_i)$ denotes the roll, pitch, and yaw angles, respectively. Here we use a world frame with $+X$ pointing right, $+Y$ pointing forward, and $+Z$ pointing up. By convention, an object faces the $-Y$ direction when $\psi_i = 0$, and positive yaw values are defined counterclockwise. The placement input consists of $\boldsymbol{O}_{\text{exist}}$, one additional object O_{N+1} to be inserted into the scene, and a language instruction $\boldsymbol{I}$. The inserted object is specified by its category c_{N+1}, size $\boldsymbol{s}_{N+1}$, and point cloud P_{N+1}, while the instruction $\boldsymbol{I}$ encodes the desired spatial relation to the scene and reference objects, for example *"place the table against the west wall, in front of the sofa (id 0), and to the left of the wardrobe (id 3)"*. We want to train a predictor f_θ to output the 6D pose of the inserted object,

$$\boldsymbol{p}_{N+1} = f_\theta(\mathcal{B}, \boldsymbol{O}_{\text{exist}}, c_{N+1}, \boldsymbol{s}_{N+1}, P_{N+1}, \boldsymbol{I}). \quad (1)$$

3.1 Geometry-Aware Multimodal Architecture

As illustrated in Fig. 2, we instantiate f_θ as a multimodal LLM whose text backbone is a pretrained causal LLM and whose geometric branch is a 3D point cloud encoder $\mathcal{E}$ followed by a projector g.

Point cloud perception module. To achieve precise geometric perception of 3D scenes, we employ Utonia [Zhang et al. 2026a], a state-of-the-art, self-supervised point transformer model. Its pretraining on diverse 3D data is crucial, as it mitigates the need for domain-specific priors, allowing the model to capture scene-level scale and fine-grained object geometry. The encoder $\mathcal{E}$ takes a point cloud $P \in \mathbb{R}^{N \times C}$ with N input points and C channels (coordinates, color, and normals) as input, and returns a feature matrix $\mathcal{E}(P) \in \mathbb{R}^{M \times d_e}$, where M is the number of output point features, typically with $M < N$, and d_e is the feature dimension.

To avoid feature conflation from a single scene-level point cloud, we encode each object's geometry independently. This gives the model separate point features for each instance.

Geometric-to-text projection. We use a projector to align geometry features with the text embedding space. The projector $g : \mathbb{R}^{d_e} \to \mathbb{R}^{d_h}$ is parameterized by a two-layer MLP with GELU activation,

where d_h is the LLM hidden dimension. For the point cloud $P^{(i)}$ of i-th object O_i, the coupled encoder-projector architecture generates a block of geometric token embeddings,

$$\mathbf{H}^{(i)} = g(\mathcal{E}(P^{(i)})) = \left(\boldsymbol{h}_1^{(i)}, \ldots, \boldsymbol{h}_{M_i}^{(i)}\right), \quad \boldsymbol{h}_m^{(i)} \in \mathbb{R}^{d_h}. \tag{2}$$

Interleaved multimodal context. We interleave the projected point-cloud tokens with the text tokens according to their positions in the prompt. For a prompt containing K point clouds, the first $K-1$ clouds encode the existing scene context in the world coordinate, including the floor and reference objects, while the last one describes the target object's intrinsic geometry in its canonical coordinate.

Let $\mathbf{H}^{(\text{tar})} = \mathbf{H}^{(K)}$ denote the embedding block of the target object. Let $\mathbf{u}_0, \ldots, \mathbf{u}_K$ denote the contiguous text-token subsequences before, between, and after the inserted point-cloud token blocks, and let $\mathbf{U}_j = \text{Emb}(\mathbf{u}_j)$ denote their text embedding blocks. We denote the mixed context embedding sequence as

$$\mathbf{X}_{\text{ctx}} = \left[\mathbf{U}_0; \mathbf{H}^{(1)}; \mathbf{U}_1; \cdots; \mathbf{H}^{(K-1)}; \mathbf{U}_{K-1}; \mathbf{H}^{(\text{tar})}; \mathbf{U}_K\right]. \tag{3}$$

The causal LLM processes this mixed sequence with standard self-attention to perceive both the existing scene context and the target-object geometry before autoregressively decoding the target pose.

3.2 Dataset Construction

To train the geometry-aware placement model, we construct a high-quality scene dataset containing object semantics, point-level geometry, spatial relations, and 6D poses. The dataset combines four complementary sources: two public datasets, 3D-FRONT [Fu et al. 2021] and Imaginarium [Zhu et al. 2025]; a SpatialLM-PCG dataset generated from SpatialLM [Mao et al. 2026] scene annotations through procedural asset replacement and small-object enrichment; and an artist-crafted scene collection. The PCG dataset increases the frequency and diversity of complex object relations, with construction details provided in the supplemental material. As summarized in Fig. 3, the dataset covers a wide range of scene complexities. SpatialLM-PCG and artist-crafted scenes provide many placement instructions and increase the number of support and containment relations.

Dataset filtering. As shown in Fig. 7, all sources are processed by a unified filtering pipeline before training. Physics simulation and geometric validation correct or remove floating objects, severe interpenetrations, and out-of-bound placements. We then render each candidate scene and use a VLM to score placement plausibility, scene coherence, visual quality, style consistency, and semantic reasonableness. We discard samples failing these checks to reduce noisy supervision.

Scene data structuring. We then convert each retained scene into single-step autoregressive placement samples. For each object, we extract its category, size, location, rotation, and point cloud, and derive spatial relations from oriented bounding boxes in the world coordinate frame. We build a floor-rooted scene tree $\mathcal{T}$ from relations defining placement dependencies: *abs* edges attach floor-supported objects to room-level location cues, while *on*/*within*/*under* edges attach objects to their supporting, containing, or overhead parents. A breadth-first traversal maps this tree to the linear placement order

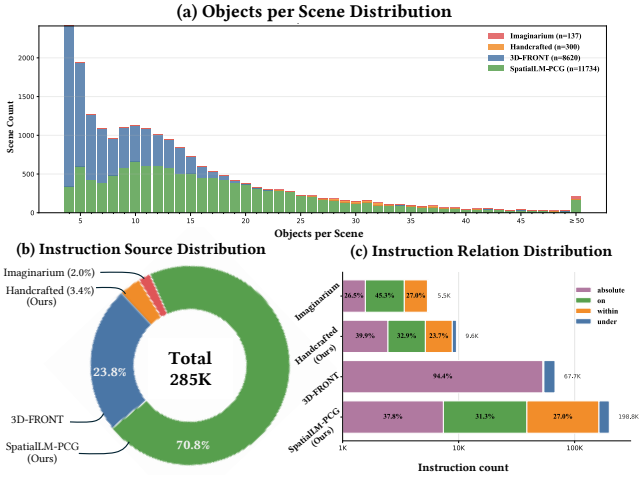

Fig. 3. Dataset statistics. (a) Distribution of object counts per scene across data sources. (b) Proportion of placement instructions generated from each source. (c) Distribution of instruction relation types, including *absolute*, *on*, *within*, and *under*.

$\boldsymbol{\pi} = \text{BFS}(\mathcal{T}) = (\pi_1, \ldots, \pi_N)$. For each target object in this order, the instruction is formed from its tree edge and augmented with nearby *next to* relations to provide local distance, direction, and facing context, such as left/right and front/behind, relative to already placed objects. The target 6D pose serves as the supervision signal.

3.3 Two-Stage Training Strategies

Multimodal LLMs commonly separate modality alignment from downstream task fine-tuning: a lightweight projector is first trained to map visual or geometric features into the LLM token space, and the aligned model is then adapted to task-specific supervision [Deng et al. 2025; Liu et al. 2023; Xu et al. 2024]. We follow this recipe because our point encoder, projector, and LLM operate in different feature spaces. We first align the geometric and language modalities through auxiliary alignment tasks, and then fine-tune the model on autoregressive placement.

Training objective. Both stages use the standard causal language modeling objective. For a training sample with context $\mathbf{X}_{\text{ctx}}$, let $\mathbf{t} = (t_1, \ldots, t_L)$ denote the complete task-specific target token sequence. With teacher forcing, we optimize the negative log-likelihood of the target tokens,

$$\mathcal{L} = -\sum_{\ell=1}^{L} \log p(t_\ell \mid \mathbf{X}_{\text{ctx}}, t_{<\ell}). \tag{4}$$

Stage 1: geometric–language alignment. We first construct auxiliary alignment samples from the scene dataset. Each sample pairs the floor point cloud or a set of object point clouds with a textual question answerable only from geometry. The questions cover boundary descriptions, object captions, object centers, sizes, rotations, and pairwise distances or displacements. In this stage, the point cloud encoder and the LLM backbone are frozen, and only the projector

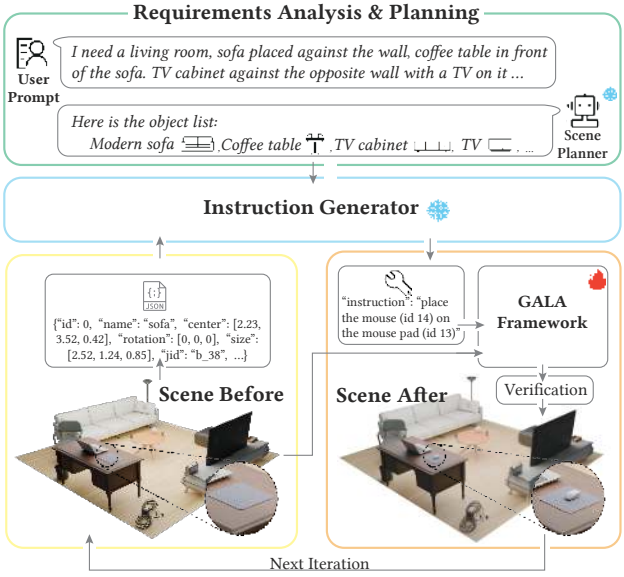

Fig. 4. Agent pipeline for automated scene generation, where a planner turns a user prompt into object-level steps, GALA places each object, and a verifier feeds back failures for correction.

is trained, making the projected point features compatible with the language model.

Stage 2: autoregressive placement training. The aligned model is then fine-tuned on autoregressive placement samples. Following the placement order $\boldsymbol{\pi}$, the model conditions at step n on the ground-truth poses of $O_{\pi_1}, \ldots, O_{\pi_{n-1}}$ and uses the pose of O_{π_n} as the supervision target. We serialize each 6D pose $\boldsymbol{p} = (x, y, z, \phi, \theta, \psi)$ into a fixed-format textual sequence and tokenize it with the LLM tokenizer. The point encoder is kept frozen, while the projector and LLM are jointly trained to combine the textual instruction with geometric context and predict a precise pose. At inference time, placement proceeds autoregressively: at each step, the model conditions on previously predicted poses, generates a textual pose sequence, parses it into numeric pose parameters, updates the scene with the newly placed object, and advances to the next one.

3.4 Application: Automated Scene Generation

For text-driven scene generation, we design a multi-stage agent pipeline that wraps GALA in a plan-place-verify loop, as shown in Fig. 4. The planner analyzes the user prompt, maintains the object list and scene state, and asks an instruction generator to describe the next placement step. GALA predicts the pose of the target object, after which a verifier checks the updated scene and sends any failure back to the instruction generator for the next iteration. Gallery results from this agent pipeline are shown in Fig. 10.

4 Experiments

4.1 Experimental Settings

Datasets. We use the two-stage datasets produced by the construction pipeline in Section 3.2. To avoid leakage, we split scenes into train, validation, and test sets with stratified sampling over the source collections at an 18:1:1 ratio. All evaluations are conducted on the test set.

Evaluation metrics. We evaluate placements along two complementary axes: *instruction following* and *geometric sensitivity*. Jointly, we deem a placement *valid* if its pose is well-formed, satisfies all specified relations, and is collision-free and non-floating. We report two success rates over this criterion: *Instruction Valid Rate*, the overall validity across all test samples, and *Within Valid Rate*, the validity on containment ("within") samples that serves as our primary geometry-sensitive metric since such relations demand local geometric reasoning. We further report the *Floating Rate* and the bottom-surface *Support Ratio* to gauge physical support from the floor or existing objects. Detailed metric definitions are provided in the supplemental material.

Implementation details. We implement our model using LLaMA-Factory [Zheng et al. 2024], with Qwen3-0.6B [Yang et al. 2025] as the LLM backbone and Utonia [Zhang et al. 2026a] as the point encoder, for approximately 0.7B total parameters. Training follows the two-stage recipe in Section 3.3. In Stage 1, we train only the projector for 3 epochs with a global batch size of 256 and a learning rate of 1×10^{-3}. In Stage 2, we fine-tune the LLM and projector for 4 epochs while keeping the point encoder frozen, using a global batch size of 128, an LLM learning rate of 2×10^{-5}, a projector learning rate of 5×10^{-4}, and cosine scheduling with 5% warmup. We train models in bfloat16 on 8 NVIDIA H20 GPUs for about 7 hours.

4.2 Comparisons with State-of-the-Art Methods

Quantitative Comparisons. We compare external baselines using a single-object insertion protocol, where each method places one target object in a fixed scene. ATISS [Paschalidou et al. 2021] and ReSpace [Bucher and Armeni 2025] serve as OBB-based references conditioned on object category. Neither method accepts a specified placement instruction as input, so they are evaluated with only physical metrics. For instruction conditioned baselines, we use a controlled LayoutVLM [Sun et al. 2025] adaptation: existing objects are fixed, only the target object is optimized, structured relations are mapped to LayoutVLM constraints. We also evaluate GPT-OSS-120B [Agarwal et al. 2025] as a zero-shot large parameter model baseline. For ATISS and ReSpace, which predict both pose and size, we scale the target point cloud by the predicted size before computing physical metrics. Table 1 summarizes the quantitative results.

Our model achieves the strongest instruction-conditioned performance, with clear gains in both within validity and overall instruction validity over LayoutVLM and GPT-OSS-120B. It also maintains competitive physical plausibility, with lower floating rate and higher support ratio. ATISS obtains a lower floating rate and a higher support ratio because it tends to place objects on the floor instead of making fine-grained placements.

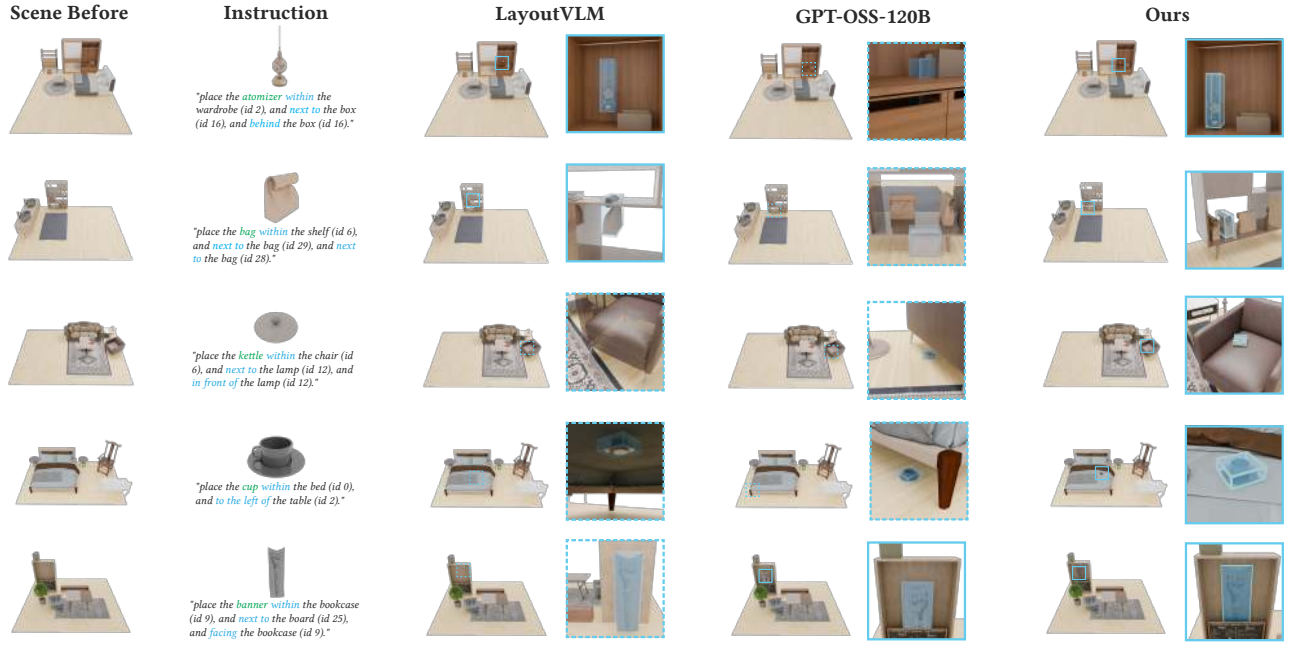

Fig. 5. Qualitative comparisons of instruction-conditioned single-object placement results across instruction-aware baselines and our method.

Table 1. Quantitative comparisons for single-object inference on the held-out test split. Cat., Instr., and PC denote category, instruction, and point cloud, respectively. Dashes indicate instruction-dependent metrics that are not applicable.

Method	Input	Within. ↑	Instruction. ↑	Float. ↓	Support. ↑
ATISS	OBB+Cat.	–	–	**0.93%**	**99.20%**
ReSpace	OBB+Cat.	–	–	5.79%	94.20%
LayoutVLM	OBB+Instr.	10.96%	37.71%	17.05%	82.91%
GPT-OSS-120B	OBB+Instr.	22.53%	42.29%	7.64%	93.54%
Ours (0.7B)	PC+Instr.	**65.62%**	**78.59%**	4.63%	96.03%

Table 2. User study scores, ranging from 1 to 5.

Method	Single-object placement		Whole-scene result	
	Physical. ↑	Instruction. ↑	Physical. ↑	Aesthetics. ↑
ATISS		–	1.9	1.9
ReSpace		–	2.7	2.9
LayoutVLM	1.9	2.0	3.0	3.2
GPT-OSS-120B	2.7	2.4	2.5	2.5
Ours (0.7B)	**4.5**	**4.5**	**3.9**	**4.1**

Qualitative Comparisons. Figure 5 illustrates LayoutVLM, GPT-OSS-120B, and our method under the same single-object placement instructions. LayoutVLM and GPT-OSS-120B respond to the instruction more directly, yet their predictions often miss fine-grained geometric constraints such as the target support surface, local free space, or containment region. In contrast, our model uses point cloud to place the target object in geometrically valid regions.

Figure 8 extends this qualitative comparison to whole-scene autoregressive inference. Across multi-step scene construction, our model produces more plausible and coherent layouts. Additional inference results produced by our model on held-out test scenes are shown in Fig. 9.

User Study. Table 2 presents the overall results of the user study. We conduct a user study with 30 volunteers to evaluate single-object and whole-scene comparison results. Each volunteer rates the displayed results on a five-point Likert scale, where 1 is the lowest quality and 5 is the highest quality. For single-object placement results, the volunteers score physical plausibility and instruction following. For whole-scene results, they score physical plausibility and visual aesthetics. Since ATISS and ReSpace do not accept placement instructions, their single-object evaluations are omitted.

4.3 Ablation Studies

We isolate the contribution of point cloud input, Stage 1 alignment pretraining, and high-quality training scenes through controlled ablations on the same test dataset. The high-quality scenes include the Imaginarium and artist-crafted subsets.

Effect of point cloud input. The instruction valid rate remains comparable across variants, indicating point cloud input does not substantially improve generic instruction-conditioned validity. Removing point cloud input weakens the model most directly on the within valid metric, where instruction following must be grounded in local containment geometry. The same ablation also increases floating failures and reduces bottom-surface support, suggesting text-only context is insufficient for reliably grounding objects on

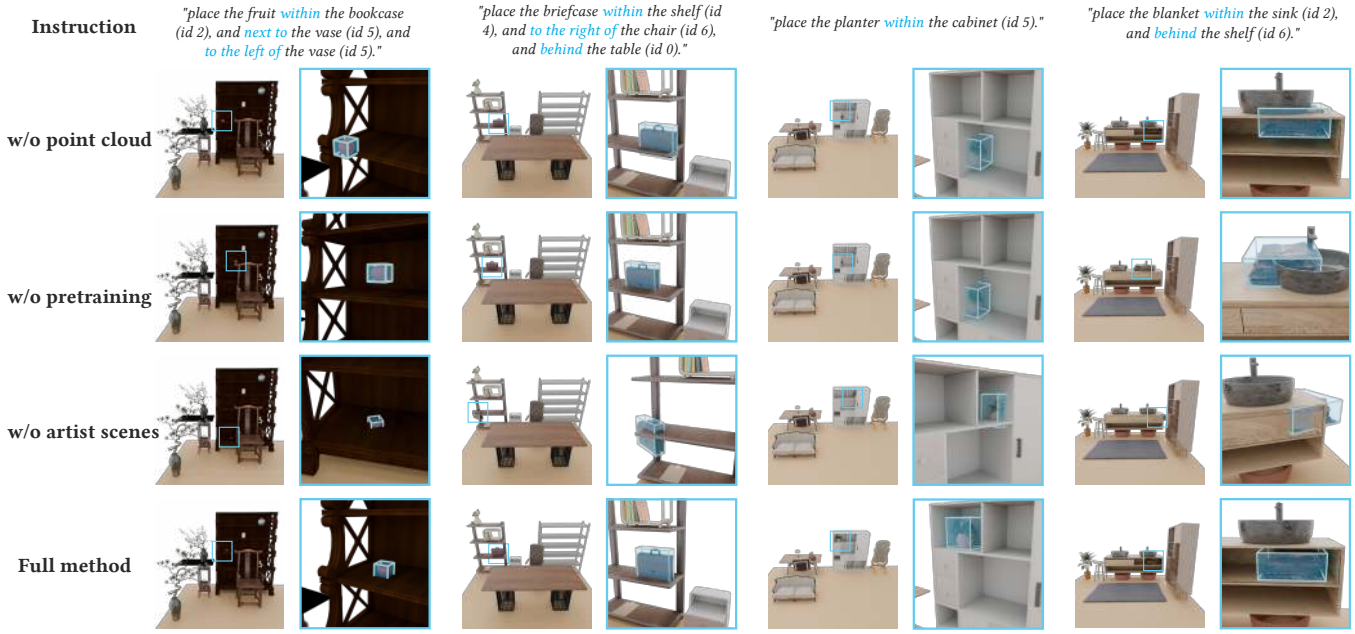

Fig. 6. Qualitative ablation results on fine-grained placement. Each column shows one instruction, the predicted scene, and a zoomed crop around the target placement. Blue boxes mark inserted target objects.

Table 3. Ablations on the held-out test split.

Method	Within. ↑	Instruction. ↑	Float. ↓	Support. ↑
w/o point cloud	62.40%	**78.81%**	5.19%	95.35%
w/o pretraining	62.69%	78.37%	5.20%	95.42%
w/o artist scenes	60.32%	76.69%	5.02%	95.88%
Full method	**65.62%**	78.59%	**4.63%**	**96.03%**

available support surfaces. Figure 6 corroborates these numbers: without point cloud input, the predicted objects frequently miss the referenced container, e.g., the fruit is placed outside the bookcase (column 1) even though the textual relation is parsed correctly.

Effect of alignment pretraining. Removing Stage 1 alignment pretraining leads to a similar degradation, even though the model still receives point cloud input during placement fine-tuning. This result suggests the geometric branch benefits from language alignment before the downstream placement task. Together, these two ablations indicate explicit scene geometry and alignment pretraining are complementary.

Effect of high-quality scenes. Removing high-quality training scenes including the Imaginarium and artist-crafted scenes produces the largest drop in within valid rate, while the instruction valid rate decreases more moderately. This result suggests high-quality scenes primarily improve fine-grained geometric grounding for containment relations rather than generic instruction validity. The floating rate and support ratio also degrade, indicating these high-quality scenes provide useful supervision for stable object placement. The qualitative cases in Fig. 6 mirror this trend: without high-quality scenes, the placed objects exhibit clear collisions or misalignment with the referenced anchors, e.g., the blanket intersects the rim of the sink (column 4) and the briefcase overlaps the shelf frame (column 2) instead of resting inside it, whereas our full method produces collision-free placements that consistently satisfy the specified relations. A stratified analysis in the supplementary material further shows this drop is concentrated on the high-quality test subset.

5 Conclusions

We propose a geometry-aware 3D object placement and scene-editing framework. By augmenting LLMs with point cloud inputs and a two-stage training recipe, our method predicts context-conditioned 6D poses. Our approach captures fine-grained spatial constraints that are often missed by OBB-only methods. We also integrate it seamlessly into interactive agents for automated scene generation. Furthermore, we will release a dataset of high-fidelity scenes with verified, complex placements to facilitate future research.

Limitations. Despite its advantages, our approach exhibits certain limitations. It currently relies on offline-sampled clean meshes, suffers from high computational overhead during autoregressive decoding for scenes with numerous objects, and its instruction quality is heavily dependent on an independent planner.

Future Work. Future research will extend this framework to handle noisy real-world scanned scenes, incorporate richer hierarchical structures (e.g., containment spaces and placeable surfaces), and support diverse interactive editing operations.

References

Ahmed Abdelreheem, Filippo Aleotti, Jamie Watson, Zawar Qureshi, Abdelrahman Eldesokey, Peter Wonka, Gabriel Brostow, Sara Vicente, and Guillermo Garcia-Hernando. 2025. PlaceIt3D: Language-guided object placement in real 3D scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 6645–6655.

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925* (2025).

Martin JJ Bucher and Iro Armeni. 2025. Respace: Text-driven 3d scene synthesis and editing with preference alignment. *arXiv e-prints* (2025), arXiv–2506.

Jiajun Deng, Tianyu He, Li Jiang, Tianyu Wang, Feras Dayoub, and Ian Reid. 2025. 3d-llava: Towards generalist 3d lmms with omni superpoint transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 3772–3782.

Haoran Feng, Yifan Niu, Zehuan Huang, Yang-Tian Sun, Chunchao Guo, Yuxin Peng, and Lu Sheng. 2026. Repurposing 3D Generative Model for Autoregressive Layout Generation. *arXiv preprint arXiv:2604.16299* (2026).

Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. 2023. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems* 36 (2023), 18225–18250.

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

Lin Gao, Jia-Mu Sun, Kaichun Mo, Yu-Kun Lai, Leonidas J Guibas, and Jie Yang. 2023. Scenehgn: Hierarchical graph networks for 3d indoor scene generation with fine-grained geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 7 (2023), 8902–8919.

Maxim Gumin, Do Heon Han, Seung Jean Yoo, Aditya Ganeshan, R Kenny Jones, Kailiang Fu, Rio Aguina-Kang, Stewart Morris, and Daniel Ritchie. 2025. Procedural Scene Programs for Open-Universe Scene Generation: LLM-Free Error Correction via Program Search. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*. 1–11.

Shuting He, Henghui Ding, Xudong Jiang, and Bihan Wen. 2024. Segpoint: Segment any point cloud via large language model. In *European Conference on Computer Vision*. Springer, 349–367.

Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 2023. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems* 36 (2023), 20482–20494.

Ian Huang, Yanan Bao, Karen Truong, Howard Zhou, Cordelia Schmid, Leonidas Guibas, and Alireza Fathi. 2025a. Fireplace: Geometric refinements of llm common sense reasoning for 3d object placement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. Computer Vision Foundation, Nashville, TN, USA, 13466–13476.

Kuan-Chih Huang, Xiangtai Li, Lu Qi, Shuicheng Yan, and Ming-Hsuan Yang. 2025b. Reason3d: Searching and reasoning 3d segmentation via large language model. In *2025 International Conference on 3D Vision (3DV)*. IEEE, IEEE, 1177–1186.

Manyi Li, Akshay Gadi Patil, Kai Xu, Siddhartha Chaudhuri, Owais Khan, Ariel Shamir, Changhe Tu, Baoquan Chen, Daniel Cohen-Or, and Hao Zhang. 2019. Grains: Generative recursive autoencoders for indoor scenes. *ACM Transactions on Graphics (TOG)* 38, 2 (2019), 1–16.

Chenguo Lin and Yadong Mu. 2024. Instructscene: Instruction-driven 3d indoor scene synthesis with semantic graph prior. *arXiv preprint arXiv:2402.04717* (2024).

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Advances in Neural Information Processing Systems* 36 (2023), 34892–34916.

Yongsen Mao, Junhao Zhong, Chuan Fang, Jia Zheng, Rui Tang, Hao Zhu, Ping Tan, and Zihan Zhou. 2026. Spatiallm: Training large language models for structured indoor modeling. *Advances in Neural Information Processing Systems* 38 (2026), 45165–45195.

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

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

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

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

Kai Wang, Manolis Savva, Angel X Chang, and Daniel Ritchie. 2018. Deep convolutional priors for indoor scene synthesis. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–14.

Wenzheng Wu, Chucheng Xiang, Zhi Lin, Yirui Guan, Ruchao Bao, Zhongyuan Liu, Ziqi Wang, and Ligang Liu. 2026. Scene Layout via conceptual design. *Computers & Graphics* (2026), 104553.

Chucheng Xiang, Ruchao Bao, Biyin Feng, Wenzheng Wu, Zhongyuan Liu, Yirui Guan, and Ligang Liu. 2026. Co-Layout: LLM-driven Co-optimization for Interior Layout. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 40. 14371–14379.

Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. 2024. Pointllm: Empowering large language models to understand point clouds. In *European Conference on Computer Vision*. Springer, 131–147.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388* (2025).

Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, et al. 2024. Holodeck: Language guided generation of 3d embodied ai environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16227–16237.

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

Guangyao Zhai, Evin Pınar Örnek, Shun-Cheng Wu, Yan Di, Federico Tombari, Nassir Navab, and Benjamin Busam. 2023. Commonscenes: Generating commonsense 3d indoor scenes with scene graph diffusion. *Advances in Neural Information Processing Systems* 36 (2023), 30026–30038.

Jiang Zhang, Shijie Zhou, Bangya Liu, Achuta Kadambi, and Zhiwen Fan. 2026b. SpatialStack: Layered Geometry-Language Fusion for 3D VLM Spatial Reasoning. *arXiv preprint arXiv:2603.27437* (2026).

Yujia Zhang, Xiaoyang Wu, Yunhan Yang, Xianzhe Fan, Han Li, Yuechen Zhang, Zehao Huang, Naiyan Wang, and Hengshuang Zhao. 2026a. Utonia: Toward One Encoder for All Point Clouds. *arXiv preprint arXiv:2603.03283* (2026).

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 3: system demonstrations)*. 400–410.

Yang Zhou, Zachary While, and Evangelos Kalogerakis. 2019. Scenegraphnet: Neural message passing for 3d indoor scene augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7384–7392.

Xiaoming Zhu, Xu Huang, Qinghongbing Xie, Zhi Deng, Junsheng Yu, Yirui Guan, Zhongyuan Liu, Lin Zhu, Qijun Zhao, Ligang Liu, et al. 2025. Imaginarium: Vision-guided High-Quality 3D Scene Layout Generation. *ACM Transactions on Graphics (TOG)* 44, 6 (2025), 1–24.

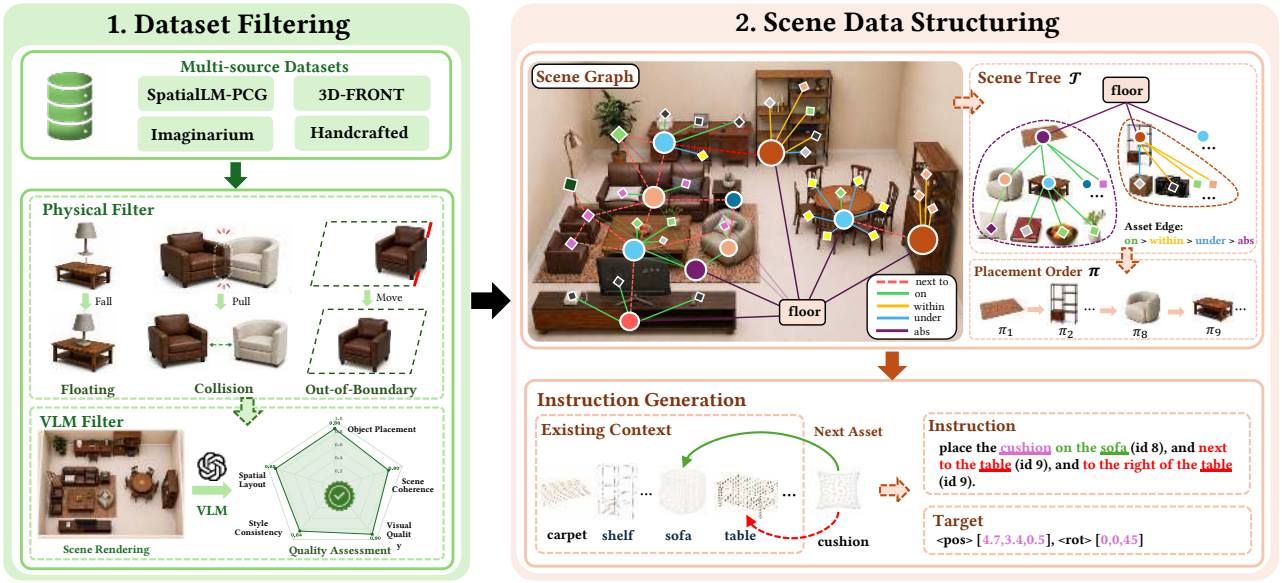

Fig. 7. Dataset construction pipeline. Multi-source scenes are first cleaned by physical plausibility checks and VLM-based quality assessment. Each retained scene is parsed into a scene graph containing absolute room-location edges (*abs*), dependency edges (*on*, *within*, *under*), and proximity edges (*next to*). We form a floor-rooted scene tree $\mathcal{T}$ by selecting dependency edges with priority *on* > *within* > *under* > *abs*. We obtain the placement order $\boldsymbol{\pi} = \mathrm{BFS}(\mathcal{T})$ by traversing from the floor root, sorting nodes at the same depth by edge priority and footprint area. For each placement step, the tree edge and nearby proximity relations are verbalized into an instruction paired with per-object point clouds and the target 6D pose.

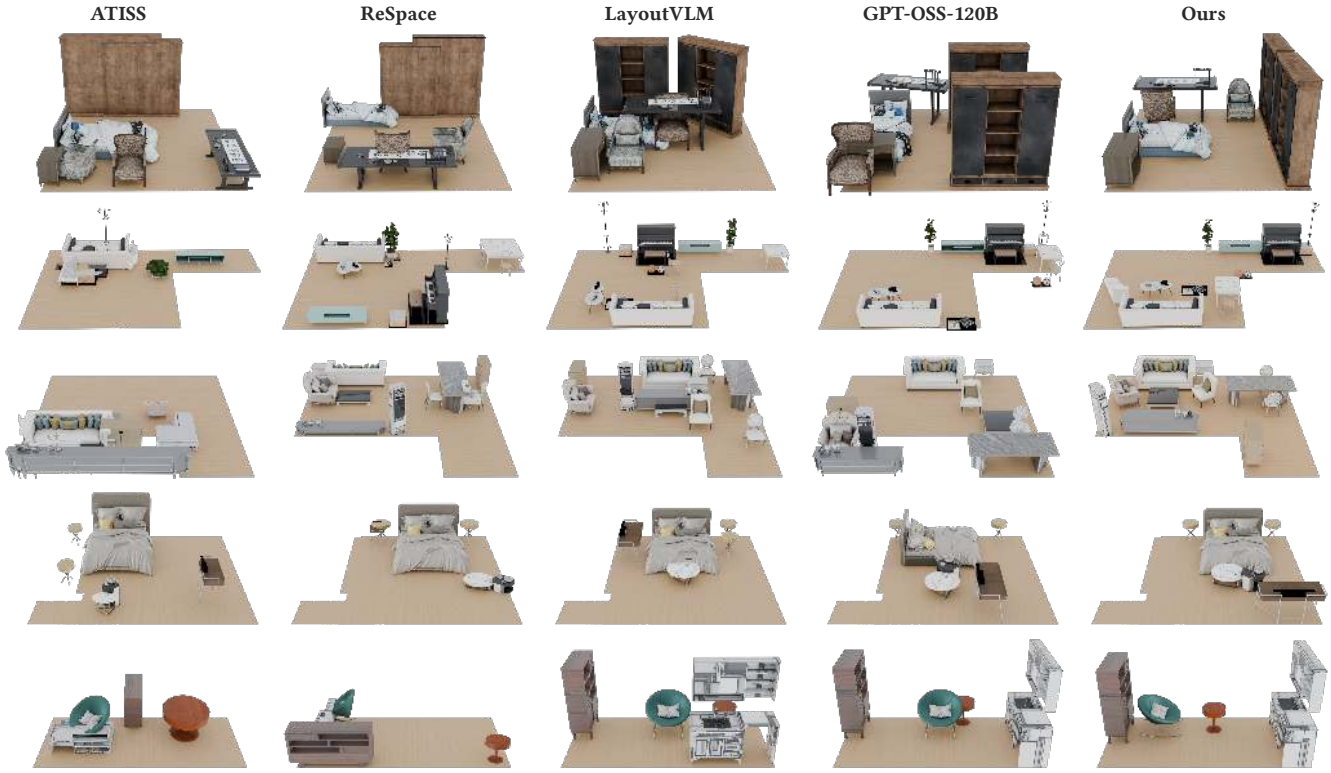

Fig. 8. Qualitative comparison with baselines on whole-scene autoregressive inference results.

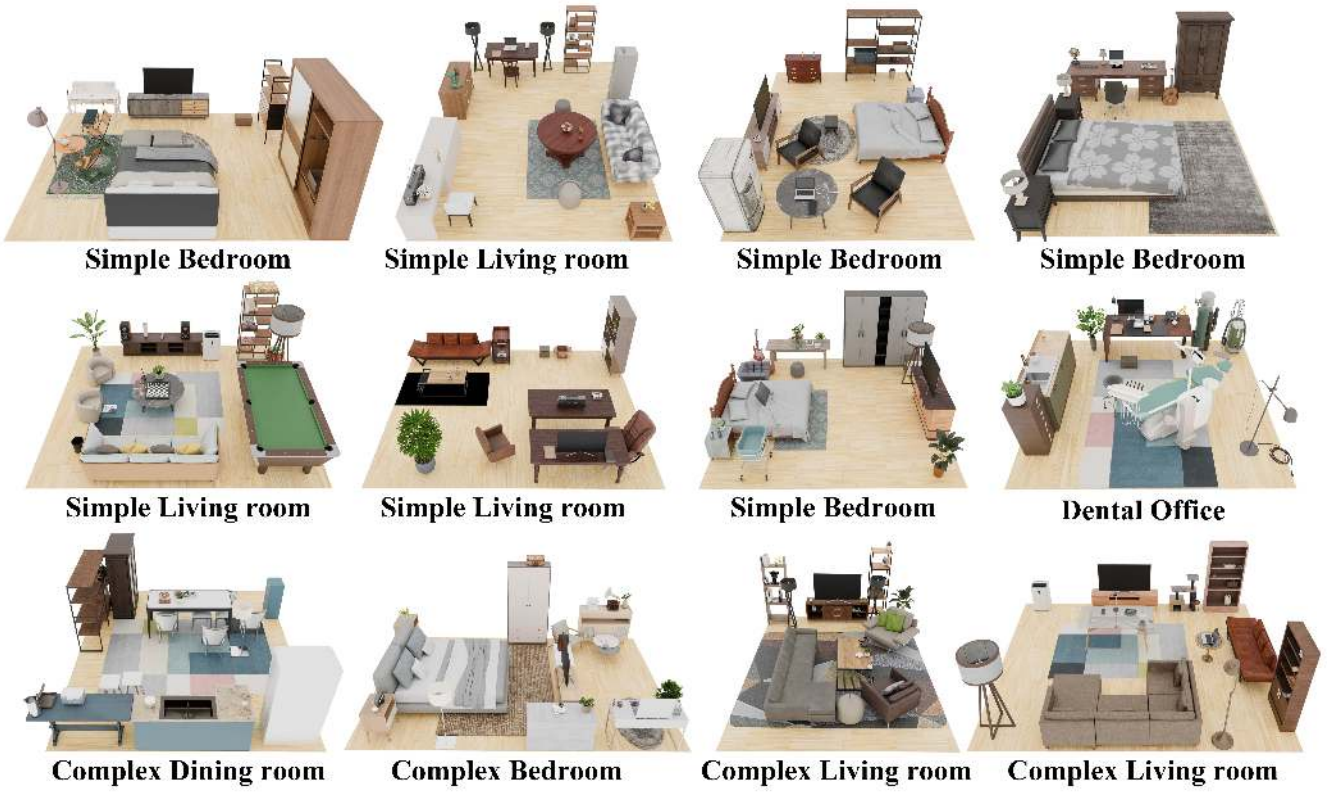

Fig. 9. More autoregressive inference results of our model on held-out test scenes. Starting from scene-level object sets and placement instructions, our model sequentially predicts object poses and produces complete arrangements across different room types and layout complexity.

Fig. 10. Gallery results from the automated scene generation agent. Given text prompts, the planner decomposes each request into object-level placement steps, GALA predicts poses for the target objects, and the verifier supports iterative correction to produce complete 3D scene arrangements.