HomeWorld: A Unified Floorplan-to-Furnished Framework for Generating Controllable, Densely Interactive Whole-Home Scenes

Wenbo Li[1,2,*] Xiaoliang Ju[1,2,*,†] Zipeng Qin[1,2,*] Rongyao Fang[2] Hongsheng Li[2,1,3]

[1]Ace Robotics [2]CUHK MMLab [3]Shenzhen Loop Area Institute

Abstract

Indoor scene generation is crucial for robot simulation and modern interior design. However, complex layouts together with scarce 3D scene data make learning-based generation challenging. Existing methods often rely on hand-crafted rules or focus on isolated sub-tasks (e.g., floorplan synthesis or single-room furnishing), producing whole-home scenes that lack global coherence, realism, and simulation readiness. To mitigate these limitations, we propose a unified hierarchical framework that decomposes indoor scene synthesis into controllable stages. First, we curate a large-scale dataset of 300K real residential floorplans to train a large language model for whole-home floorplan generation. With detailed descriptions and a K-D tree–based representation, our method enables fine-grained, controllable whole-home floorplan generation. Building upon the generated whole-home floorplan, we leverage image generation models to draft furniture layouts from multi-level roaming viewpoints, and then generate the layouts of small manipulable objects on different supporting surfaces (e.g. cabinets, desks, and dining tables) for embodied AI simulation. During furniture & object layout generation, a VLM-based refiner iteratively corrects furniture & object placement, and a 3D generative model enables flexible replacement of individual assets. We further attach basic physical attributes and simple surface texture and lighting setups to complete the pipeline for embodied AI use. Experiments and user studies demonstrate that our pipeline produces indoor spaces with greater layout diversity and stronger 3D design appeal, outperforming prior methods on both quantitative and qualitative metrics. Finally, alongside our generation pipeline, we will release the floorplan dataset and 5K fully furnished scenes to the community.

Correspondence: Xiaoliang Ju, akira@link.cuhk.edu.hk; Hongsheng Li, hsli@ee.cuhk.edu.hk

Project Page: https://https://kairos-homeworld.github.io

*Equal contribution. †Project lead.

Contents

1 Introduction

Indoor scene generation is critical for simulation in embodied AI and for modern interior design workflows. As embodied learning and generalist policies advance, agents are expected to navigate, interact, and complete increasingly complex tasks, raising the bar for virtual environments. The target is therefore shifting from isolated, single-room scenes to complete, multi-room homes with globally coherent structure and interactive surroundings. Meeting this demand calls for whole-home scene generation that is not only visually realistic, but also functionally plausible and physically feasible for simulation.

However, scaling to such whole-home generation is fundamentally constrained by data: large-scale, high-fidelity 3D residential scenes are still scarce in public datasets. Existing resources are dominated either by rule-based synthetic datasets [1–3], whose layouts and object distributions are biased by hand-designed priors, or by reconstruction-based datasets [4, 5], which often suffer from incomplete geometry and fragmental surfaces that make reliable layout extraction and asset grounding difficult. As a result, both the quantity and the quality of available 3D data are insufficient to effectively train learning-based generative models for diverse, realistic, and physically feasible multi-room homes.

Existing generative methods struggle under such data regime. Approaches that learn directly in 3D are typically limited by the quantity and fidelity of training scenes, and are computationally expensive, making it difficult to efficiently generate large-scale, whole-home environments [6, 7]. Meanwhile, rule-based pipelines like [1] can guarantee basic validity, but tend to produce repetitive layouts and biased object distributions. Another line of work exploits 2D priors from image generation model to lift room imagery to 3D such as [8]. However, without a strong 3D prior to anchor the generation, it is difficult to preserve geometric consistency when scaling to whole-home scenes. In practice, the lifted 3D outputs are often incomplete and fragmented, exhibiting artifacts commonly seen in reconstruction-based 3D data.

To overcome the data bottleneck while meeting the requirements of whole-home, interactive simulation, we propose a unified hierarchical generation framework that combines strong, scalable priors with explicit 3D grounding. Instead of learning whole-home 3D synthesis end-to-end from scarce 3D datasets, we first learn to generate globally coherent 2D floorplan, and then progressively instantiate it into a physically feasible 3D environment with furniture and manipulable objects. A key observation is that even 2D floorplan data—despite being easier to obtain than full 3D scenes—is still relatively scarce in large scale and often lacks clean, standardized representations for learning. We therefore curate and annotate a large dataset of 300K real residential floorplans, and use it to train an LLM for fine-grained, controllable floorplan generation with caption supervision and a K-D tree–based representation.

Conditioned on the generated floorplan, we instantiate the 3D home with a view-roaming, prior-guided synthesis stage. We leverage the diversity and commonsense plausibility of foundation image generation models to propose new objects from roaming camera views, while using 3D constraints derived from the floorplan (e.g., room boundaries, walls, and free space) to anchor the proposals and enforce cross-view 3D consistency. To make the process stable and semantically structured, we adopt a hierarchical roaming policy: we first place large, function-defining furniture under a top-down view (e.g., beds and sofas), and then descend to egocentric views to progressively enrich the scene with smaller items. Finally, to better support embodied AI simulation, we distribute manipulable objects throughout the home. Throughout this process, an VLM-based recursive refiner corrects implausible placements and constraint violations, and a 3D generative module enables flexible asset replacement to increase object diversity and rendering variation without breaking scene coherence.

In summary, our contributions are two-fold:

- **A whole-home indoor scene generation pipeline.** We propose a unified hierarchical pipeline for controllable whole-home generation under limited 3D data. It (i) trains LLMs to generate precise and controllable floorplans via a K-D tree representation and detailed captions; (ii) leverages foundation image models as strong 2D priors to propose objects through hierarchical view roaming while maintaining 3D consistency and physical adherence; (iii) employs an VLM-based recursive refiner to correct construction errors; and (iv) supports flexible 3D asset replacement via a 3D generative module to increase object diversity and rendering variation.
- **A large-scale whole-home indoor scene dataset.** We curate and will publicly release a dataset of **300K** annotated real-world residential floorplans, together with **5K** fully furnished, high-quality whole-home 3D scene samples, providing a new dataset that directly mitigate the data scarcity bottleneck in whole-home layout generation. We plan to continuously expand this dataset in subsequent releases.

2 Related Work

Indoor scene dataset. We summarize representative datasets in table 1 and group them by data source, scope, simulator readiness, and curation, together with key scale statistics. Early real-world floorplan dataset such as LI-FULL [9], RPLAN [10], MSD [11], and ResPlan [12] provide large-scale residential layouts, but they are typically limited to 2D floorplans and therefore are not directly usable for embodied simulation. In contrast, reconstruction-based datasets (e.g., ScanNet [4] and Matterport3D [5]) offer real 3D scans with semantic annotations; however, these scans often cover partial spaces and generally do not provide instance-separated, manipulable assets required by interactive simulators, making them less "sim-ready" in our definition.

To bridge the gap, several designed or synthetic datasets provide furnished 3D indoor scenes. 3D-FRONT [13] contains thousands of curated residential scenes with explicit object instances and renderable assets, and is commonly used for 3D scene synthesis and rendering. Structured3D [2] programmatically generates large-scale indoor scenes with ground-truth structure, but it is not primarily designed for interactive manipulation. More recent "collection" datasets, including InternScenes [14] and SceneVerse [15], aggregate heterogeneous sources to scale up scene diversity; while they provide many furnished scenes, simulator readiness and manipulability vary across subsets (hence marked as partial). ProcTHOR-10K [1] procedurally generates fully 3D houses that are directly usable in simulation; although it does not report per-scene manipulable-object counts, it provides an interactable asset library of 1633 instances.

Overall, existing datasets trade off between scale, realism, and direct usability for embodied simulation. Our dataset targets this gap by combining large-scale real floorplans with generated object assets to produce sim-ready, furnished 3D houses with hybrid curation.

3D indoor scene generation. Prior work on 3D indoor scene generation can be broadly grouped into the following categories. 1) Procedural rule-based generation constructs scenes by executing pre-defined grammars, priors, or handcrafted constraints (e.g., [1, 21]), which can yield controllable outputs but is often bottlenecked by a finite rule set and asset library—resulting in limited diversity. 2) Direct 3D-domain generation like [6, 7, 25, 26] naturally satisfies geometric constraints, but scaling to whole-home scenes remains challenging due to computational cost and the limited availability of diverse 3D training data. 3) Compositional generation that first predicts a layout and then fills

Table 1 Indoor Scene Resources Comparison. Rec. denotes reconstruction-based real-world datasets. **S.**/**H.** denote the number of furnished **Scenes** (individual, often room-level areas) and **Homes** (unified, with multiple rooms).**Sim-ready** denotes whether a dataset provides fully 3D scenes that are directly instantiable in a simulator/rendering engine and supports object-level manipulation. **MObj.** reports manipulable objects per scene on average when available. **n/r** means not reported and "–" means not applicable. **Collection** indicates datasets aggregated from multiple sources.

Resource	Source / Training Data	Scope	Sim-ready	Curation	Data Scale / Scalability		
					Floorplan	Furnished S./H.	MObj.
PART A: STATIC / RELEASED DATASET							
LI-FULL [9]	Real	Home	✗	Raw	5M	-	-
RPLAN [10]	Real	Home	✗	Manual	80K	-	-
MSD [11]	Real	Home	✗	Manual	5.3K	-	-
ResPlan [12]	Real	Home	✗	Manual	17K	-	-
ScanNet [4]	Real(Rec.)	Partial	✗	Manual	-	1.5K S.	n/r
MatterPort3D [5]	Real(Rec.)	Home	✗	Manual	90	90 H.	n/r
3D-FRONT [13]	Designed	Home	✓	Manual	6.8K	6.8K H.	n/r
Structured3D [2]	Designed	Home	✗	Programmatic	3.5K	21K S.	n/r
InternScenes [14]	Collection	Mixed	Partial	Hybrid	-	48K S.	8
SceneVerse [15]	Collection	Mixed	Partial	Hybrid	-	68K S.	n/r
ProcTHOR-10K [1]	Synthetic(Rules)	Home	✓	Programmatic	10K	10K H.	n/r*
Ours (Released)	FP.(Real) + Obj(Gen.)	Home	✓	Hybrid	300K	5K H.	>15
PART B: GENERATION PIPELINE							
Floorplan-LLaMA [16]	Collection	Home	✗	LLM	✓	✗	✗
Floorplan-Diffusion [17]	Real	Home	✗	Diffusion	✓	✗	✗
LayoutGPT [18]	None	Partial	Partial	LLM	✗	✓	✗
Holodeck [19]	Collection	Home	✓	LLM	✓	✓	✗
LayoutVLM [20]	Designed	Partial	Partial	VLM	✗	✓	✗
ProcTHOR [1]	Synthetic(Rules)	Home	✓	Programmatic	✓	✓	✗
Infinigen [21]	Synthetic(Rules)	Home	✓	Programmatic	✓	✓	✗
PhyScene [22]	Designed	Home	✓	Diffusion	✓	✓	✗
ChOrD [23]	Designed	Home	✓	Diffusion	✓	✓	✗
EmbodiedGen [24]	Collection	Mixed	✓	Diffusion	✗	✓	✗
Ours (Pipeline)	Real floorplan only	Home	✓	MLLM	✓	✓	✓

* ProcTHOR-10K does not report manipulable-object counts per scene; it provides an asset library containing 1633 interactive instances.

it with 3D assets has therefore become a common paradigm. [19, 27, 28] use LLMs to generate whole-home floorplans and asset layouts, but remains at a relatively preliminary stage. Since native LLMs have limited spatial understanding, the resulting layouts are also limited in terms of plausibility and diversity. More methods focus on room-level scene, such as [18, 20, 29–31]. While these methods work well for composing a full home from individually generated rooms, indoor environments exhibit strong global coherence (e.g., functional adjacencies and consistent style/scale). As a result, approaches limited to local context often struggle to maintain whole-home consistency. 4) Another promising direction is "roaming" or incremental whole-scene creation via 2D-lifting approaches such as [8, 32, 33]; however, because these methods often lack strong 3D constraints, they tend to suffer from poor geometric consistency and are hard to deploy in simulation.

Our pipeline adopts a compositional approach that integrates roaming-based techniques while mitigating their limitations. We enforce explicit 3D constraints and employ a reflective loop for iterative correction, ensuring strong 3D consistency throughout generation. In addition, most prior

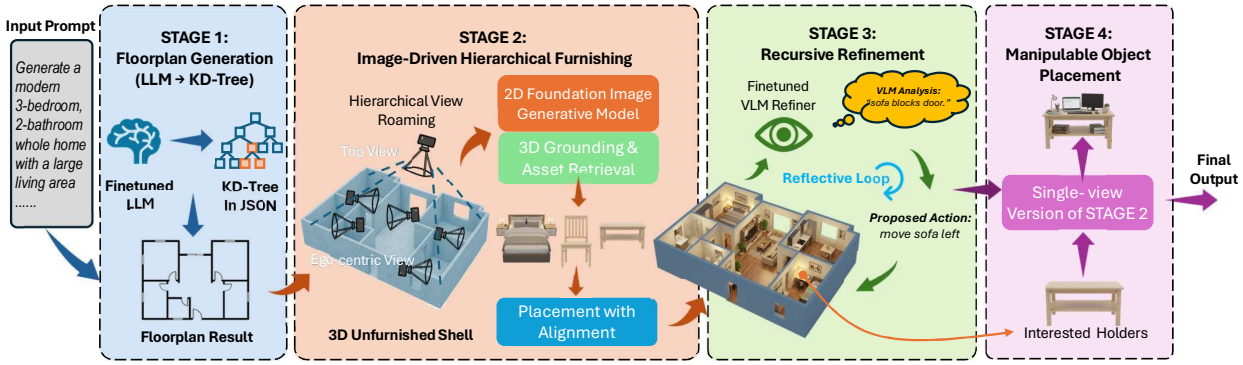


Figure 1 Overview of the whole-home indoor scene generation pipeline. Our system turns a text prompt into a complete indoor scene in four stages: 1) build a large-scale floorplan dataset and train a prompt-conditioned generator; 2) given the floorplan, create an unfurnished 3D shell and progressively place furniture via hierarchical roaming with grounding/retrieval; 3) improve the scene using a finetuned VLM-based refiner to fix violations in a reflective loop; 4) add manipulable objects in feasible surface tops (e.g. dining tables, desks, cabinets, etc.).

work does not explicitly target simulation-ready environments with plentiful manipulable objects for embodied AI. In contrast, we generate complete homes hierarchically, progressing from a floorplan, to a furniture layout, and finally to manipulable small objects, producing coherent, fully populated, simulation-ready scenes.

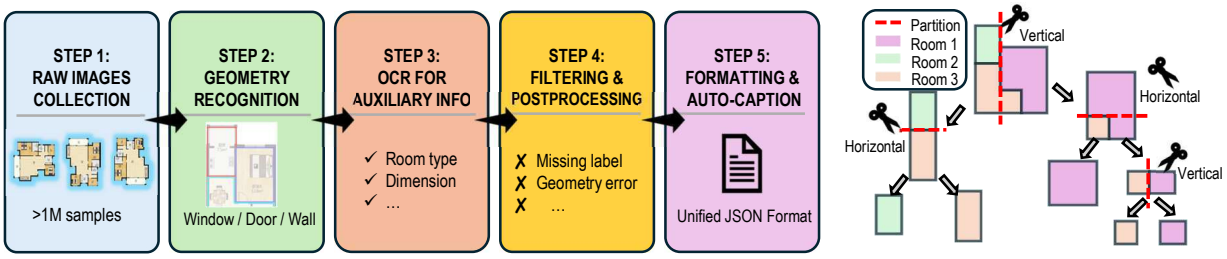
3 Methodology

As illustrated in figure 1, our system enables users to describe a desired home layout with a natural-language prompt and automatically synthesizes a complete indoor scene. The workflow can be divided into four stages. First, we describe how we build a real-world large-scale floorplan dataset and train a prompt-conditioned floorplan generator. Second, conditioned on the floorplan, we introduce a hierarchical roaming strategy to populate the scene progressively. Before starting, we first build 3D unfurnished shell as the explicit 3D constraint for the roaming. Then the system prioritizes the placement of large, room-defining furniture, and then adds more objects, using grounding and asset retrieval to ensure geometric and semantic consistency. Third, we employ a recursive refinement loop based on a finetuned VLM, which inspects the rendered scene to detect violations (e.g., collisions or blocked pathways), and then acts as a refiner agent that proposes and executes corrective actions. Finally, we use the similar strategy as Stage 2 to add manipulable objects in user-specified areas of interest.

3.1 Floorplan Dataset Curation and LLM-based Floorplan Generation

Current floorplan datasets are limited in scale, complexity, diversity, and realism [10–12]. Yet for applications such as robot navigation, a large-scale and high-fidelity dataset is essential to enable robust generalization to real-world environments. In this section, we develop an automatic procedure for extracting vectorized data from carefully-curated large-scale real-world floorplan images from professional architectural portfolios. We also devise a K-D tree representation for effective training of floorplan generation model.

Floorplan dataset curation. We construct a large-scale floorplan dataset in five steps (figure 2(a)),



a) Floorplan dataset construction pipeline. b) Floorplan KD-Tree.

Figure 2 **(a)** Floorplan dataset construction pipeline and **(b)** the KD-tree representation for floorplan used in our generator.

which transforms raw floorplan images into structured representations with three key attributes: (1) the positions of architectural elements, such as windows and doors; (2) room geometries and spatial layouts; and (3) semantic room-type labels.

In step 1, we collect 1.084 million floorplan images from an online real-estate repository. These images contain common architectural elements, including doors, windows, walls, room labels, and dimension annotations. In step 2, we train a detection model on 2K manually annotated images to identify doors, windows, walls, and dimension chains. The detected bounding boxes are converted into centerlines, and nearby endpoints are merged to reconstruct the vectorized floorplan structure.

In step 3 and 4, we apply OCR to extract room labels and dimensional annotations, followed by quality filtering to discard noisy or unreliable samples. This yields validated floorplans with reliable geometric and semantic information.

In step 5, we further derive structural annotations such as room connectivity from shared doors and adjacent walls, and assign unique identifiers to all entities, including rooms, doors, windows, and walls. Based on the resulting geometric, semantic, and topological information, we automatically generate detailed textual captions for each floorplan.

These captions are used to construct training data for the LLM-based generator. Specifically, we randomly sample and combine caption components, such as room counts, boundary information, positional constraints, connectivity relations, and attachment requirements, to form diverse natural-language prompts, while the corresponding floorplan representation serves as the target output. This procedure allows each floorplan to generate multiple prompt–target pairs with varying levels of difficulty. The final dataset contains 314K validated floorplans, all fully vectorized and paired with detailed structural captions.

LLM-based floorplan generator with K-D tree representation. We formulate floorplan generation as a text-to-structure prediction task. The input to the LLM is a natural-language description of the design requirements, including the apartment boundary, room program, positional constraints, adjacency relations, and placement requirements for openings or attachments. The output is not direct geometric primitives such as room polygons; instead, the model predicts a structured `json` file that explicitly encodes the floorplan as a binary K-D tree, together with attachment information.

In this representation, the interior space is recursively partitioned into axis-aligned subregions by

alternating vertical and horizontal cuts (Fig. 2(b)). Each internal node stores the split axis and split coordinate, while each leaf node corresponds to a spatial cell and is assigned semantic information such as room identity and room type. Attachments, including doors, windows, and related elements, are represented separately in the same output `json`. Because the entire layout is serialized as a hierarchical text structure, it is naturally compatible with the autoregressive generation process of LLMs.

Compared with directly predicting polygon coordinates and openings, this K-D tree-based output space is much more structured and easier to constrain. In particular, it avoids many common geometric errors, such as polygon overlap, and produces layouts that can be deterministically converted back into conventional floorplan representations for downstream use. Since generic pretrained LLMs do not possess sufficient domain knowledge for this task, we finetune the model on our floorplan dataset. Ablation results that validate the effectiveness of this representation are provided in section 4.3.

3.2 Image-driven Hierarchical Furnishing

We adopt a 2D-lifting approach realized through hierarchical indoor roaming, primarily because existing 3D furniture-layout datasets are insufficient in both quantity and quality as aforementioned. We leverage the strong semantic priors of 2D foundation generative models to enable open-vocabulary object placement with improved semantic plausibility. Our system then grounds the generated results into precise 3D bounding boxes, parameterized by semantic label, physical dimension and pose. The roaming consists of two stages: *top-down roaming* initializes coarse object placement at the scene level, while *ego-centric roaming* focuses on local regions to refine object geometry.

Unfurnished shell as 3D constraints for roaming. Before the roaming starts, we first instantiate an empty, unfurnished shell in Blender with basic lighting and material settings. This shell is used to impose explicit 3D constraints during roaming, which is especially important for irregular, non-rectangular room layouts as shown in the teaser image. With the consistent geometry, it reduces perspective-induced artifacts and 3D inconsistencies caused by fully free roaming (e.g., drifting geometry in Text2Room [8]). In addition, the lighting-driven shading provides a natural prior for the subsequent 2D generation stage, leading to more coherent appearance and smoother view-to-view transitions.

Global layout generation via top-down view. The key idea is to exploit a structure-aware top-down view as an intermediate representation, since major furniture placement is largely governed by global constraints: room boundary, doors, windows, and inter-room connectivity. Given the empty shell, we render a top-down view image I_{top}. We then annotate I_{top} with doors and windows, and for each door we also mark the adjacent room, because these cues strongly influence the furniture layout. Conditioned on these structural priors, we apply an image inpainting model to I_{top} to synthesize a furnished top view $\hat{I}_{\text{top}}$, leveraging 2D foundation priors for more realistic, prompt-controllable furniture arrangements. After a lightweight validation step to discard infeasible generations, we use a VLM for open-vocabulary category recognition and SAM-3 [34] for instance masks and 2D boxes. This initial layout of major furniture will be further refined in the following stage.

Detail refinement via ego-centric view. While the global top-down view provides a reliable initialization for major furniture, it is inherently limited by occlusion and scale ambiguity, making it difficult to recover small objects and wall-attached items (e.g., bins, hooks, racks, and upper/lower cabinets in kitchens). We therefore introduce an ego-centric roaming stage to enrich the scene with fine-grained details.

Viewpoint selection. To cover the entire room with limited renders, we compute a compact set of viewpoints using a heatmap-based viewpoint selection strategy. Concretely, we discretize the floor plane into a grid and sample candidate camera poses along the room boundary and interior, subject to a fixed field-of-view and a safe distance to walls. Each candidate view contributes visibility to a subset of grid cells; we accumulate these contributions into a coverage heatmap. We then greedily select viewpoints by repeatedly choosing the view that covers the currently least-observed regions (equivalently, maximally reduces the uncovered heat), updating the heatmap after each selection.

Ego-centric synthesis and 3D grounding. For each viewpoint k, we render an ego-centric image I_k and apply inpainting to synthesize additional secondary objects, producing $\hat{I}_k$. We then use SAM-3 [34] to extract instance masks for newly introduced objects and employ SAM-3D [35] to reconstruct their 3D geometry. We apply a geometric alignment module to integrate these newly reconstructed 3D assets into the existing room environment. By moving the viewpoint, this iterative ego-centric roaming process provides 360-degree coverage, resulting in a densely populated and highly realistic 3D indoor environment.

3.3 Recursive layout refinement

Since both 2D synthesis and 3D grounding are inherently error-prone, the generated layouts may contain physical, semantic, and structural violations, such as blocked doorways, boundary-crossing objects, object collisions, and floating objects. To address these issues, we introduce a lightweight VLM-based refinement module that performs closed-loop verification and correction for each candidate room layout, thereby improving the robustness of the proposed generation pipeline. We begin with the iterative refinement formulation, since it defines the sequential correction task that the refiner is trained to solve, and then describe the construction of the corresponding fine-tuning dataset.

Iterative refinement formulation. For each candidate room layout, we render a top-view image and pair it with the corresponding structured 3D layout description, which encodes object categories, positions, sizes, rotations, and room-level architectural elements. These multimodal observations are provided to the VLM refiner, which evaluates the layout under physical plausibility, semantic consistency, and structural constraints. Rather than treating refinement as a one-shot validation problem, we formulate it as a sequential decision-making process: at each iteration, the refiner observes the current layout state and predicts a structured corrective action that identifies the target object and specifies the required edit, such as translation, rotation, or a combined transformation. The predicted action is then deterministically applied to the 3D bounding-box layout representation, after which the updated layout is re-rendered and re-checked. This iterative loop continues until the layout passes validation or a predefined iteration limit is reached.

Fine-tuning data construction. Following the above sequential refinement formulation, we construct a dedicated supervision corpus to train the VLM refiner for reliable multi-step layout correction. The dataset is designed to expose the model to invalid intermediate layouts and teach it to predict the next corrective action that best improves the scene. It is built from three complementary components: *corrupted layout construction*, *oracle-labeled repair actions*, and *model-in-the-loop samples*.

- *Corrupted Layout Construction.* Starting from clean room layouts that satisfy basic physical, semantic, and structural constraints, we inject controlled perturbations to synthesize corrupted states. These perturbations cover common failure modes observed in the generation pipeline, including boundary violations, object collisions, doorway blocking, implausible rotations, scale inconsistencies, and category-level errors. Since each perturbation step is explicitly recorded,

the resulting clean-to-corrupted trajectories naturally provide reverse supervision for iterative layout repair.

- *Oracle-labeled Repair Actions*. For each corrupted state, we further generate high-quality action labels using an oracle action generation pipeline. The oracle first proposes a set of candidate corrections, such as translation, rotation, combined transformation, and displacement along the minimum separation direction. Each candidate is then evaluated by a verifier according to residual errors, newly introduced violations, movement cost, and semantic plausibility. The highest-scoring candidate is selected as the ground-truth next action for the current layout state.

- *Model-in-the-loop Samples*. To improve robustness under closed-loop inference, we additionally collect model-in-the-loop samples. After training an initial refiner, we roll it out on corrupted layouts and collect the intermediate states produced by the model during iterative correction. These states are relabeled by the oracle and added back to the training set, allowing the model to learn from its own failure cases and reducing the distribution gap between supervised training and deployment. Each training sample consists of a top-view rendering, the corresponding structured 3D layout description, the current error state, optional action history, and the oracle-labeled correction action. This dataset construction enables the refiner to learn stable multi-step correction behavior under physical, semantic, and structural constraints.

3.4 Manipulable Object Placement

Although the top-down and ego-centric roaming stages recover coherent layouts for large furniture, many supporting surfaces, such as tables, desks, countertops, shelves, and nightstands, may still remain unnaturally empty. This reduces both visual realism and functional richness, since small manipulable objects provide not only clutter cues for everyday environments but also potential interaction targets for embodied tasks. To address this issue, we introduce a surface-centric object placement strategy that populates supporting furniture with plausible small objects.

Surface Object Synthesis. Given a generated room, we first identify furniture items that can support small objects according to their semantic categories and geometric properties, such as the existence of a horizontal surface and sufficient available area. For each selected furniture item, we construct a local surface canvas conditioned on the room type, furniture category, and the target support region. An inpainting model is then used to synthesize realistic surface-level object arrangements by leveraging learned priors on object co-occurrence and spatial organization. For example, a desk may be populated with books, a laptop, a lamp, and stationery, while a kitchen countertop may contain bowls, bottles, utensils, or small appliances.

Physical Attribute Assignment. To make synthesized manipulable objects physically meaningful and simulation-ready, we use PhysX-Anything [36] to infer object-level and part-level physical attributes. For each object, PhysX-Anything [36] predicts its semantic category, dimensions, part decomposition, and material properties, including density, Young's modulus, and Poisson's ratio. In our pipeline, the predicted density and mesh volume are used to estimate approximate object mass, while part-level material properties support the assignment of component-wise simulation parameters. These attributes are further used during layout recovery and validation to assess support validity, object stability, collision consistency, and plausible interaction behavior, improving the physical grounding of generated small-object arrangements for downstream simulation and embodied interaction.

3D Layout Recovery and Filtering. The synthesized surface image is subsequently parsed by a VLM into a set of manipulable objects, including their semantic categories, approximate scales, relative positions, and orientations. These predictions are first represented in the local coordinate system of the supporting furniture, and are then transformed into the global room coordinate system according to the furniture pose and physical dimensions. This formulation ensures that small objects remain spatially aligned with their host furniture even when the furniture is rotated or placed at different locations in the room.

Before inserting these objects into the 3D scene, we perform lightweight physical filtering to improve plausibility. Specifically, we remove or adjust objects that exceed the support boundary, produce severe inter-object collisions, float above the supporting surface, or violate category-level support constraints. We also preserve the explicit support relation between each small object and its host furniture, which facilitates downstream scene graph construction, task generation, and embodied interaction. As a result, the generated rooms are enriched with realistic and functionally meaningful object-level details while maintaining basic physical consistency.

3.5 Shell Texture Generation and Basic Lighting

Surface Texturing. For the indoor shell, including walls, floors, and ceilings, textures are obtained from two sources. When suitable assets are available in the 3D-FRONT [13] repository, we directly reuse those materials. Otherwise, the textures are synthesized using a 2D image generation model and then applied to the corresponding surfaces. Since this component is not a core contribution of our work, we adopt this simple strategy to enrich the appearance diversity of indoor shells without introducing a dedicated material generation pipeline.

Lighting Setup. For indoor lighting, we adopt rule-based generation methods. A primary ceiling light is placed near the center of each room to provide the main illumination, while auxiliary and decorative lights are configured using predefined heuristics based on the room layout. These rules determine the placement, color temperature, and relative brightness of secondary light sources. The final renderable parameters are assigned from predefined templates to ensure stable and consistent rendering.

4 Experiments

Figure 3 Overview of the whole-home scene generation.

Figure 3 provides an intuitive overview of our whole-home scene generation pipeline. Given a natural-language prompt, our method progressively generates a floorplan, instantiates an unfurnished

3D shell, produces a furnished 3D scene, and finally enriches the environment with manipulable objects. Following this pipeline, our experiments are organized into three parts. First, we evaluate floorplan generation to demonstrate the effectiveness of our proposed K-D tree-based representation on our dataset. Second, we report results on floorplan-conditioned whole-home generation. Third, we conduct a comprehensive ablation study. Both quantitative and qualitative evaluations show that our method outperforms existing approaches.

4.1 Floor Plan Generation

Experiment setup. We split our dataset into 281K training samples, 15K validation samples, and 15K test samples. We finetune Qwen3-4B-Instruct [37] using supervised finetuning with LoRA [38] on our training corpus.

We formulate floorplan synthesis as a constrained instruction-following task. The prompt specifies following factors of the target floorplan: a) dimensions; b) room count and types; c) door/window placement; d) a structural output format requiring a K-D tree with axis-aligned (horizontal or vertical) partitions; e) spatial constraints regarding room adjacencies and connections.

Evaluation metrics. We introduce two quantitative metrics to assess the generated floorplans: 1) *graph validity*, the fraction of generated floorplans whose connectivity graph is fully connected, measuring basic topological navigability; 2) *graph diversity*: the number of distinct connectivity graph structures generated for a fixed room count, measuring topological variety. We report typed diversity (room types included) and untyped diversity (topology only).

Table 2 Quantitative comparison of floorplan generation. Our model generates more accurate and diverse room topologies than the selected baseline.

	Graph Validity	Graph Diversity	
		Typed	Untyped
Procthor-10K [1]	72.6%	6.2	2.3
Ours	**87.7%**	**22.4**	**6.9**

Quantitative results. We randomly sample 500 floorplans from our finetuned Qwen3 model and compare them against generated samples from Procthor-10K[1]. For graph validity, our model achieves a 15.1% improvement over Procthor-10K (see table 2 column 1), demonstrating its ability to learn accurate opening placements (e.g., doors) that yield fully connected floorplan topology. For both typed and untyped graph diversity, we compute the numbers of distinct connectivity graphs for samples containing one to ten rooms - a range representative of typical residential layouts - and report the average. As shown in table 2 columns 2 and 3, our model consistently outperforms the baseline across on the graph diversity, suggesting that our model successfully learns diverse, real-world strategies for constructing room connection topologies from the training data.

Qualitative results. We present a qualitative comparison of floorplans generated by our finetuned Qwen3-4B [37] model against samples from the Procthor [1] baseline in figure 4. Examples of structural artifacts include *unreasonable room semnatics*, where doors connect two room types in an implausible manner, such as bedroom–bedroom or livingroom–livingroom connections. *Wrong entrance door placement* refers to positioning the apartment's primary entrance outside the livingroom area. On the other hand, our samples demonstrate superior structural validity and design quality. The baseline method exhibits limited ability to recognize appropriate opening placements, such as *missing window* instances and door placements that disrupt the connectivity topology of the

floorplan.

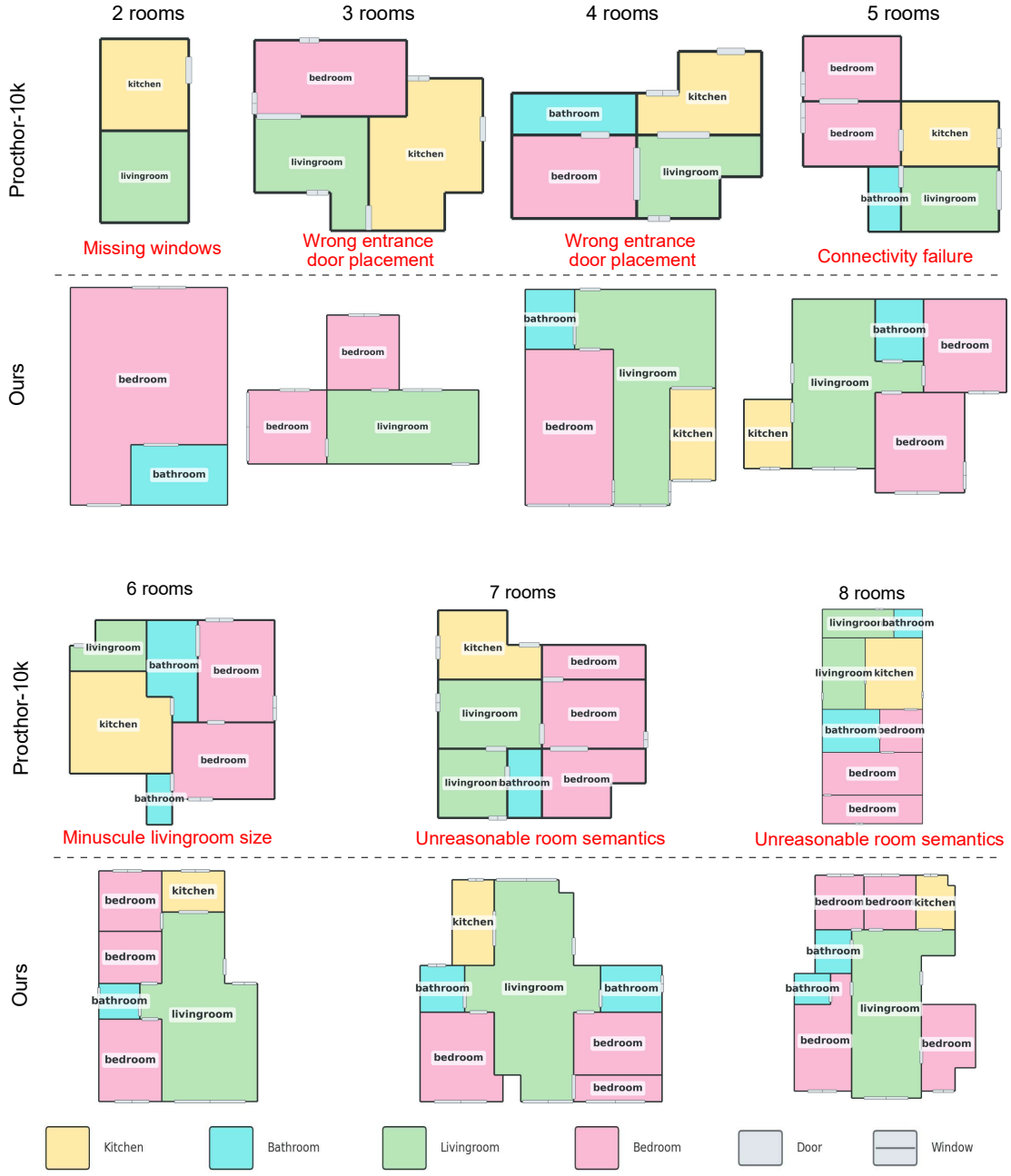


Figure 4 Samples from generated floorplan samples. Zoom-in for better view. We compare our generated floorplans with the baseline across the simpler setting (2-5 rooms) and a more complex setting (6-8 rooms). Our trained model consistently produces more feasible and plausible floorplans compared to that of the baseline method.

User study. We conduct a human evaluation study with 24 participants to assess the perceived quality of generated floorplans. We choose the recently proposed FloorPlan-LLaMa [16] and Floorplan-Diffusion [17] as our compared methods. Participants are presented with three generated floorplans from the three methods and asked to rank them along two dimensions: a) *Reasonability* for practical feasibility, realistic room flows, and constructibility; and b) *Richness* for appropriate

Figure 5 Comparison of whole-home generation.

organizational richness without unnecessary complication to support multiple daily uses. To ensure consistent interpretation, all floorplans are rendered using a standardized color palette encoding room types and openings. Rankings are converted to numerical scores: 3, 2, and 1 points for first, second, and third place, respectively. These scores are aggregated across participants and averaged as shown in table 3. Our generated results consistently achieve the highest quality across both evaluation dimensions.

Table 3 User study of floorplan generation. Our generated results achieve superior quality on both *Reasonability* and *Richness* dimensions.

	Reasonability	Richness	Average
FloorPlan-LLaMA [16]	1.85	2.00	1.93
Floorplan-Diffusion [17]	2.13	1.81	1.97
Ours	**2.20**	**2.34**	**2.27**

4.2 Furniture Layout Generation

Experiment setup. We exploit Blender to perform all 3D template initialization and perspective rendering in our automated 3D scene generation pipeline. For 2D image inpainting in both top-down and ego-centric view roaming, we use Gemini 3.0 Image Pro Preview (aka Nano Banana Pro). For instance-level detection and dense pixel-wise segmentation, we employ SAM3 [34]. We then use SAM 3D(Object) [35] to obtain the initial object geometry and pose.

We evaluate our method against the following baselines: LayoutGPT [18], Holodeck [19], and LayoutVLM [20], representing existing methods for open-universe layout generation.

Evaluation Metrics. We evaluate generated 3D layouts along two dimensions: **physical plausibility** and **distributional properties**. For physical plausibility, we report the *Collision Ratio* (CR) and the *Out-of-Boundary Ratio* (OOB). CR quantifies severe inter-object penetrations (excluding semantically valid contacts), and OOB measures the fraction of object footprints outside the room boundary. For distributional properties, we report *Volume Density* (VD) and *Footprint Object Density* (FOD), where VD is total 3D box volume normalized by floor area and FOD is the object count normalized by the union area of all furniture footprints, reflecting how densely objects occupy the floor plan.

Qualitative results.We visualize predicted layouts in both whole-home and single functional room settings. Fig. 5 compares whole-home results; among the evaluated baselines, only Holodeck [19] and our method support whole-home generation. For single-room generation, we provide additional visual comparisons for the living room/bedroom in Fig. 6, and kitchen / bathroom in Fig. 7, respectively. Note that LayoutGPT [18] is limited to living-room and bedroom generation, and thus is excluded from kitchen and bathroom comparisons.

Quantitative results. To evaluate controllable room-layout generation across common residential spaces, we generate four room categories: living room, bedroom, bathroom, and kitchen, with 25 samples per method per category. As shown in Tab. 4, our approach consistently outperforms prior methods on the metrics defined above. These quantitative gains align with qualitative observations: our generated layouts exhibit fewer severe overlaps and yield more diverse, natural spatial arrangements.

Table 4 Quantitative comparison of layout generation. Our generated results achieve superior quality.

	Physical Plausibility		Density & Diversity	
Method	CR↓	OOB↓	VD↑	FOD↑
LayoutGPT [18]	0.05	0.01	0.22	1.99
Holodeck [19]	0.07	0.02	0.22	2.10
LayoutVLM [20]	0.20	0.01	0.27	2.15
Ours	**0.05**	**0.01**	**0.35**	**4.16**

User Study. Additionally, we conducted a user study participated by 30 users. For each method, we sampled 10 results spanning both the whole-home setting and four single-room categories: living room, bedroom, bathroom, and kitchen.

Participants were shown each sample generated by 4 methods, and asked to rank them from best to worst along three dimensions: (a) *Reasonability* for practical feasibility, realistic room flows, and constructibility; (b) *Aesthetics* for visual legibility, balance, and design coherence; and (c) *Complexity* for appropriate organizational richness without unnecessary complication. As reported in Tab. 5, our method is preferred by roughly all of participants across all questions and scene categories.

Figure 6 Comparison of generated functional rooms: livingroom and bedroom.

4.3 Ablation Study

Ablation on Floorplan Generation(*w/o K-D tree representation*). Empirically, when using the canonical representation, 9.2% of 500 generated floorplans exhibit overlapping room polygons, and 2.0% contain holes within apartment outlines. Furthermore, the percentage of floorplans with disconnected rooms rises to 16.2% under the canonical representation, compared to 12.3% with our K-D tree formulation-a notable degradation in the reliability and structural coherence of floorplan generation.

Ablation on Layout Generation. We conduct an ablation study by disabling one module at a time while keeping all remaining components, prompts, and the evaluation protocol unchanged. We compare four variants against the full system (**Ours**) in Tab. 6, reporting physical plausibility (**CR**↓, **OOB**↓) and distributional properties (**VD**↑, **FOD**↑).

Table 5 User study of layout generation. Our generated results achieve superior quality on *Reasonability*, *Aesthetics* and *Complexity*.

	Reasonability	Aesthetics	Complexity	Average
LayutGPT [18]	0.202	0.184	0.153	0.180
Holodeck [19]	0.395	0.412	0.452	0.420
LayoutVLM [20]	0.252	0.260	0.283	0.265
Ours	0.807	0.827	0.797	0.811

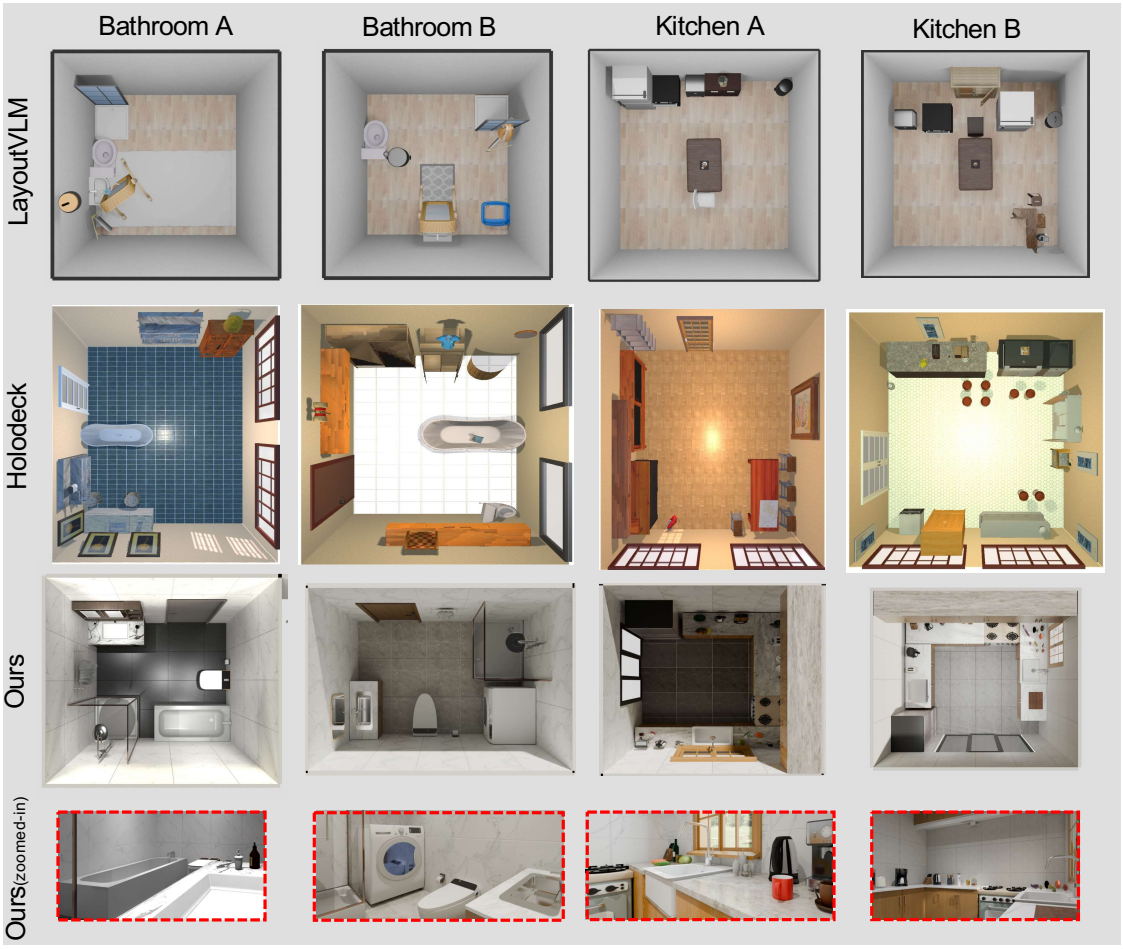


Figure 7 Comparison of generated functional rooms: bathroom and kitchen.

1) *w/o top-down view roaming.* We remove the structure-aware top-down stage (top-view inpainting and the top-down 2D-to-3D initialization) and rely on the remaining pipeline. This leads to worse reduces scene density (**VD** drops from **0.35** to **0.22**), indicating that the top-down stage provides crucial global structure priors for both feasibility and coarse furniture placement.

2) *w/o ego-centric view roaming.* We disable ego-centric roaming and generate layouts only from the top-down initialization. The plausibility slightly degrades, while **VD** decreases (**0.35** $\rightarrow$ **0.20**), showing that ego-centric roaming is important for enriching furniture-level content.

3) *w/o VLM refiner.* We turn off the recursive refinement loop (no multi-view consistency checking or iterative box corrections). This significantly harms physical plausibility (**CR**/**OOB** rises to **0.20**/**0.05**), demonstrating that the refiner is critical for correcting structural and physical violations produced by synthesis and grounding.

4) *w/o manipulable object placement.* We remove manipulable surface-object placement while keeping the rest of the pipeline unchanged. Physical plausibility remains on par with **Ours**, but **FOD** drops sharply (**4.16** $\rightarrow$ **1.82**) and **VD** slightly decreases (**0.35** $\rightarrow$ **0.33**), indicating that object placement mainly increases the number of small, manipulable objects without sacrificing feasibility.

Overall, the top-down stage and ego-centric stage improves global feasibility and coarse density,

Table 6 Ablation study on the Layout Generation.

Method	Physical Plausibility		Density & Diversity	
	CR↓	OOB↓	VD↑	FOD↑
w/o Top-down View Roaming	0.09	0.02	0.22	2.6
w/o Ego-centric View Roaming	0.07	0.02	0.20	3.44
w/o VLM–LLM Refiner	0.20	0.05	0.33	3.02
w/o Manipulable Object Placement	0.05	0.01	0.33	1.82
Ours	0.05	0.01	0.35	4.16

object placement substantially increase object-level richness (especially FOD), and the VLM-based refiner is essential for ensuring physical plausibility.

5 Conclusion

In this work, we present HomeWorld, a unified hierarchical framework for controllable whole-home scene generation from natural-language prompts. First, we synthesize fine-grained residential floorplans using an LLM trained on a large-scale real-world floorplan corpus with a structured K-D tree representation. We then progressively instantiate the generated layout into a furnished 3D home through hierarchical view roaming, recursive refinement, and surface-centric manipulable object placement. This unified design connects floorplan generation with downstream 3D scene realization, enabling the creation of complete whole-home environments rather than isolated rooms.

To address the scarcity of large-scale 3D residential training data, we combine scalable 2D generative priors with explicit 3D grounding and iterative correction. The resulting pipeline supports coherent furniture arrangement, realistic placement of manipulable small objects, and basic scene completion with physical attributes, texture assignment, and lighting setup for downstream embodied AI simulation. Experiments and user studies show that our framework produces more valid and diverse floorplans, as well as more plausible and richer 3D layouts, than prior methods.

Alongside the generation framework, we will release a dataset of 300K annotated real-world residential floorplans together with 5K high-quality furnished whole-home scenes. We hope this combination of model and data can support future research on controllable indoor scene generation and simulation-ready embodied AI environments.

References

[1] 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.

[2] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling. In European Conference on Computer Vision, pages 519–535. Springer, 2020.

[3] Manolis Savva, Angel X Chang, Alexey Dosovitskiy, Thomas Funkhouser, and Vladlen Koltun. Minos: Multimodal indoor simulator for navigation in complex environments. arXiv preprint arXiv:1712.03931, 2017.

[4] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 5828–5839, 2017.

[5] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. arXiv preprint arXiv:1709.06158, 2017.

[6] Zhennan Wu, Yang Li, Han Yan, Taizhang Shang, Weixuan Sun, Senbo Wang, Ruikai Cui, Weizhe Liu, Hiroyuki Sato, Hongdong Li, et al. Blockfusion: Expandable 3d scene generation using latent tri-plane extrapolation. ACM Transactions on Graphics (ToG), 43(4):1–17, 2024.

[7] Xiaoliang Ju, Zhaoyang Huang, Yijin Li, Guofeng Zhang, Yu Qiao, and Hongsheng Li. Diffindscene: Diffusion-based high-quality 3d indoor scene generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4526–4535, 2024.

[8] Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 7909–7920, 2023.

[9] LIFULL Co., Ltd. and National Institute of Informatics (NII). LIFULL HOME'S High Resolution Floor Plan Image Data. https://www.nii.ac.jp/dsc/idr/en/lifull/1.html, 2017. Accessed: 2026-02-28.

[10] Wenming Wu, Xiao-Ming Fu, Rui Tang, Yuhan Wang, Yu-Hao Qi, and Ligang Liu. Data-driven interior plan generation for residential buildings. ACM Transactions on Graphics (SIGGRAPH Asia), 38(6), 2019.

[11] Casper Van Engelenburg, Fatemeh Mostafavi, Emanuel Kuhn, Yuntae Jeon, Michael Franzen, Matthias Standfest, Jan van Gemert, and Seyran Khademi. Msd: A benchmark dataset for floor plan generation of building complexes. In European Conference on Computer Vision, pages 60–75. Springer, 2024.

[12] Mohamed Abouagour and Eleftherios Garyfallidis. Resplan: A large-scale vector-graph dataset of 17,000 residential floor plans. arXiv preprint arXiv:2508.14006, 2025.

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

[14] Weipeng Zhong, Peizhou Cao, Yichen Jin, Li Luo, Wenzhe Cai, Jingli Lin, Hanqing Wang, Zhaoyang Lyu, Tai Wang, Bo Dai, et al. Internscenes: A large-scale simulatable indoor scene dataset with realistic layouts. arXiv preprint arXiv:2509.10813, 2025.

[15] Baoxiong Jia, Yixin Chen, Huangyue Yu, Yan Wang, Xuesong Niu, Tengyu Liu, Qing Li, and Siyuan Huang. Sceneverse: Scaling 3d vision-language learning for grounded scene understanding. In European Conference on Computer Vision, pages 289–310. Springer, 2024.

[16] Jun Yin, Pengyu Zeng, Haoyuan Sun, Yuqin Dai, Han Zheng, Miao Zhang, Yachao Zhang, and Shuai Lu. Floorplan-llama: Aligning architects' feedback and domain knowledge in architectural floor plan generation. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6640–6662, 2025.

[17] Minyang Xu, Yunzhong Lou, Xiang Gao, and Xiangdong Zhou. Floorplan-diffusion: Automatic floor plan generation via pre-trained large latent diffusion model. In Proceedings of the 2025 International Conference on Multimedia Retrieval, pages 1617–1625, 2025.

[18] 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.

[19] 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, pages 16227–16237, 2024.

[20] 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, pages 29469–29478, 2025.

[21] 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, pages 21783–21794, 2024.

[22] 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, pages 16262–16272, 2024.

[23] Chong Su, Yingbin Fu, Zheyuan Hu, Jing Yang, Param Hanji, Shaojun Wang, Xuan Zhao, Cengiz Öztireli, and Fangcheng Zhong. Chord: Generation of collision-free, house-scale, and organized digital twins for 3d indoor scenes with controllable floor plans and optimal layouts. arXiv preprint arXiv:2503.11958, 2025.

[24] Xinjie Wang, Liu Liu, Yu Cao, Ruiqi Wu, Wenkang Qin, Dehui Wang, Wei Sui, and Zhizhong Su. Embodiedgen: Towards a generative 3d world engine for embodied intelligence. arXiv preprint arXiv:2506.10600, 2025.

[25] Aleksey Bokhovkin, Quan Meng, Shubham Tulsiani, and Angela Dai. Scenefactor: Factored latent 3d diffusion for controllable 3d scene generation. In Proceedings of the Computer Vision and Pattern Recognition Conference, pages 628–639, 2025.

[26] Zehuan Huang, Yuan-Chen Guo, Xingqiao An, Yunhan Yang, Yangguang Li, Zi-Xin Zou, Ding Liang, Xihui Liu, Yan-Pei Cao, and Lu Sheng. Midi: Multi-instance diffusion for single image to 3d scene generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 23646–23657, 2025.

[27] 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, pages 217–234. Springer, 2024.

[28] Xinhang Liu, Chi-Keung Tang, and Yu-Wing Tai. Worldcraft: Photo-realistic 3d world creation and customization via llm agents. arXiv preprint arXiv:2502.15601, 2025.

[29] 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, pages 20507–20518, 2024.

[30] Martin JJ Bucher and Iro Armeni. Respace: Text-driven 3d indoor scene synthesis and editing with preference alignment. arXiv preprint arXiv:2506.02459, 2025.

[31] Yixuan Yang, Junru Lu, Zixiang Zhao, Zhen Luo, James JQ Yu, Victor Sanchez, and Feng Zheng. Llplace: The 3d indoor scene layout generation and editing via large language model. arXiv preprint arXiv:2406.03866, 2024.

[32] Jaeyoung Chung, Suyoung Lee, Hyeongjin Nam, Jaerin Lee, and Kyoung Mu Lee. Luciddreamer: Domain-free generation of 3d gaussian splatting scenes. arXiv preprint arXiv:2311.13384, 2023.

[33] Yiwen Chen, Hieu T Nguyen, Vikram Voleti, Varun Jampani, and Huaizu Jiang. Housecrafter: Lifting floorplans to 3d scenes with 2d diffusion models. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 28440–28450, 2025.

[34] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. arXiv preprint arXiv:2511.16719, 2025.

[35] Xingyu Chen, Fu-Jen Chu, Pierre Gleize, Kevin J Liang, Alexander Sax, Hao Tang, Weiyao Wang, Michelle Guo, Thibaut Hardin, Xiang Li, et al. Sam 3d: 3dfy anything in images. arXiv preprint arXiv:2511.16624, 2025.

[36] Ziang Cao, Fangzhou Hong, Zhaoxi Chen, Liang Pan, and Ziwei Liu. Physx-anything: Simulation-ready physical 3d assets from single image. arXiv preprint arXiv:2511.13648, 2025.

[37] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. arXiv preprint arXiv:2505.09388, 2025.

[38] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. Iclr, 1(2):3, 2022.