This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

VULCAN: Tool-Augmented Multi Agents for Iterative 3D Object Arrangement

Zhengfei Kuang[1]*, Rui Lin [2], Long Zhao[2], Gordon Wetzstein[1], Saining Xie[2,3], Sanghyun Woo[2]
[1]Stanford University [2]Google [3]New York University
{zhengfei,gordonwz}@stanford.edu {linrui,shwoo}@google.com
gary.zhao9012@gmail.com saining.xie@nyu.edu
vulcan-3d.github.io

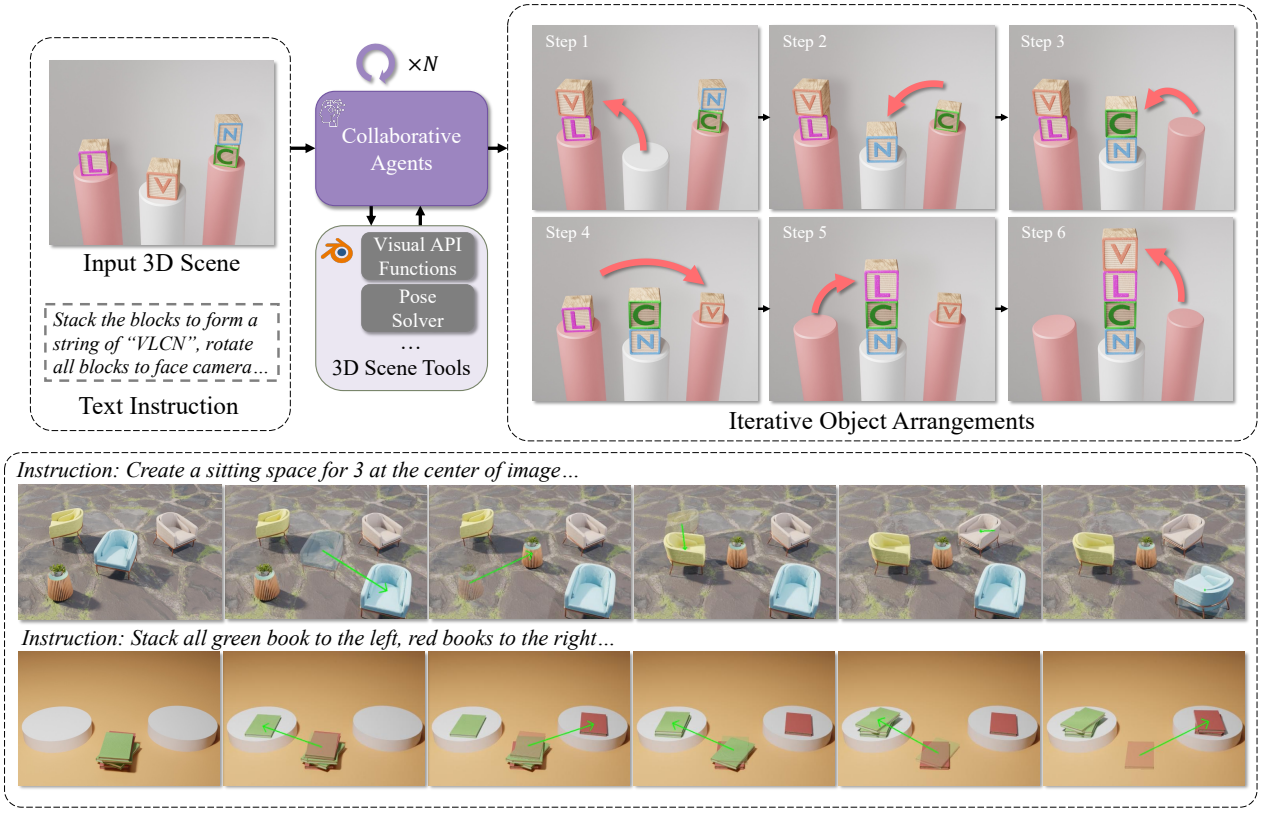


Figure 1. **VULCAN** plans and executes multiple actions for complex object arrangement tasks given input image and user prompt.

Abstract

Despite the remarkable progress of Multimodal Large Language Models (MLLMs) in 2D vision-language tasks, their application to complex 3D scene manipulation remains underexplored. In this paper, we bridge this critical gap by tackling three key challenges in 3D object arrangement task using MLLMs. First, to address the weak visual grounding of MLLMs, which struggle to link programmatic edits with precise 3D outcomes, we introduce an MCP-based API. This shifts the interaction from brittle raw code manipulation to more robust, function-level updates. Second, we augment the MLLM's 3D scene understanding with a suite of specialized visual tools to analyze scene state, gather spatial information, and validate action outcomes. This perceptual feedback loop is critical for closing the gap between language-based updates and precise 3D-aware manipulation. Third, to manage the iterative, error-prone up-

*This work was done when Zhengfei Kuang was an intern at Google.

dates, we propose a collaborative multi-agent framework with designated roles for planning, execution, and verification. This decomposition allows the system to robustly handle multi-step instructions and recover from intermediate errors. We demonstrate the effectiveness of our approach on a diverse set of 25 complex object arrangement tasks, where it significantly outperforms existing baselines.

1. Introduction

Humans routinely rearrange objects in their environment, instinctively decomposing complex, multi-object tasks into sequential steps. For instance, moving a dining table naturally requires first clearing its surface and moving surrounding chairs. This fundamental ability to perform multi-step planning, grounded in deep spatial reasoning and a commonsense understanding of the physical world, is essential for executing complex arrangement tasks.

Recent Multimodal Large Language Models (MLLMs) [5, 34, 50, 56] have enhanced human-like reasoning, enabling prior works [14, 15, 21, 26, 27, 61] to explore 3D object arrangement, which involves moving, rotating, or inserting objects to create a plausible layout. However, a key limitation of these approaches is their formulation of arrangement as a single-step process: starting from an initial scene, the agent performs one comprehensive edit to reach the goal state. The standard pipeline simply involves MLLM agents analyzing the 3D scene to generate this single edit.

In this work, we introduce a fundamentally more flexible paradigm. The core of our approach is the ability to elastically decompose tasks. Our model is not confined to a single edit; instead it dynamically assesses the user's instruction and the scene's complexity. For simple requests, it can generate a single-step solution, similar to the capabilities of prior work. For complex tasks, it decomposes the instruction into a multi-step plan of sequential operations. This ability to decide whether to perform a single edit or long-horizon sequence of actions—a capability that was by design infeasible for previous approaches—is what allows our model to mirror the human-level reasoning and planning. Given a user instruction, whether abstract (e.g., "Create a dining space") or detailed (e.g., "Move the book to the shelf, then move the table"), the agent can now formulate a coherent, step-by-step plan and reliably execute each action in sequence to achieve the specified goal.

The fundamental requirement for this advanced task is a pipeline that operates with high-fidelity and robustness at every single step. In iterative arrangement, unlike single-step tasks, errors in either scene analysis or execution can propagate and compound, quickly derailing the entire multi-step process. Prior single-step approaches have employed various modalities for MLLM-based scene analysis, from raw visual representations [26, 27] to structured scene descriptions [24, 67]. However, these methods strike a poor balance: they either overburden the MLLM with complex raw 3D data it cannot natively parse or provide simplified information that is insufficient for robust spatial reasoning.

This is the gap that recent MLLM tool calling [28, 36, 62] and the development of the Model Context Protocol [23] have begun to address. These breakthroughs demonstrate a new way where MLLMs can be equipped to handle complex tasks [21, 46] that often lie beyond their core, encoded knowledge. This approach aids the MLLM by allowing to interact with external, use-defined application programming interfaces (APIs). Inspired by these innovations, we address both the analysis and execution challenges. For interactive analysis, we equip our agents with a powerful visual API toolset. This allows them to actively query the environment, infer scene layouts from observations, and determine appropriate arrangement configurations without being overburdened. For reliable execution, we develop an advanced collision-free solver. This component ensures physical plausibility by translating the agent's high-level, symbolic intent into precise, valid numerical object poses.

The next critical challenge is scaling our tool-augmented capabilities to a multi-step scenario. This transition demands a sophisticated reasoning process: the agent must plan several moves ahead, ensuring each intermediate arrangement is not only physically plausible but also serves as a valid precondition for subsequent actions. Simultaneously managing high-level, long-horizon strategy and low-level, step-by-step execution is a significant burden for a single agent. We therefore separate these tasks by employing specialized agents with distinct responsibilities. One agent focuses on global planning across the entire task trajectory, while others handle the per-step execution and evaluation within the 3D scene. This division is implemented in our collaborative multi-agent pipeline.

Another critical challenge is the combinatorial search space. Multi-step arrangement is inherently complex; the search space grows exponentially with each step, as every placement decision branches into numerous possible future states. This creates a tree of possibilities that quickly becomes intractable. To manage this complexity, we propose an effective search algorithm that strategically backtracks from unpromising paths, efficiently pruning the search tree and avoiding the trap of exhaustive enumeration.

We propose **VULCAN**, a tailored agentic pipeline that leverages state-of-the-art MLLMs to generate accurate and plausible sequences of arrangements based on user instructions. We summarize our key contribution as:

- **A tool-augmented MLLM** that integrates MCP-based visual tools with a constraint-based solver for efficient 3D scene analysis and physically-plausible arrangement;
- **A collaborative multi-agent framework** that strategi-

cally separates responsibilities for global planning, step-by-step execution, and intermediate-state examination;
- **An adaptive backtracking search algorithm** that efficiently navigates the exponential search space to find a valid path to the goal state.
- **A new 3D object arrangment benchmark** designed to evaluate challenging 3D-aware object arrangements, on which our model significantly outperforms all the previous baselines.

2. Related Works

3D Object Manipulation and Arrangement The task of 3D object manipulation and arrangement is fundamental to both computer vision and robotics, as it enables agents to understand, interact with, and plausibly reorganize complex environments. Conventional 3D object arrangement methods typically employ large 3D datasets [11, 13, 32, 33, 43, 44, 60] to build data-driven models [16, 17, 52–55, 58]. However, these models are not guided by the common-sense reasoning of large language models, which limits their capability for general-purpose applications. Following recent developments in MLLMs [50, 56], recent works [15, 24, 49, 57, 61, 67] have begun to leverage the reasoning capabilities of MLLMs to address object arrangement in a way that aligns with common sense. Since the standard MLLMs are not natively trained on 3D scene data at scale, these works enable the MLLM to access various intermediate representations for 3D scene analysis, such as textual scene description files [12, 67], code scripts [26, 49], or rendered images [14, 26, 27]. Among these, FirePlace [27] and ScanEdit [14] also apply dedicated solvers with geometric constraints to ensure physically plausible results. However, all these works are limited to single-step operations. They lack the capability to handle complex multi-step scenarios where objects must be arranged sequentially with careful consideration of intermediate states. Our work directly addresses this gap by introducing an iterative framework that can decompose complex instructions into a coherent sequence of actions.

MLLMs for 3D The development of MLLMs extends large language models [50, 51, 56] to other modalities, including audio [18], image [5, 34], and video [7]. Leveraging their powerful cross-modal reasoning capabilities, numerous works have developed systems to tackle various vision-language-based 3D tasks, such as geometry generation [9, 38, 40, 47], scene understanding [6, 25, 39, 63, 65], and appearance editing [41, 48]. Compared to methods that directly apply hard-coded algorithms [20, 30, 31, 35, 42, 64] or deep learning models on task-specific datasets [22, 45, 59, 66], these MLLM-based approaches demonstrate better generalization and produce more semantically plausible results. While benchmarks like BlenderGym [19] were introduced to evaluate these 3D operations, both these existing works and their benchmarks were not designed for complex object arrangement. They provide limited support for the spatial reasoning required by an iterative, multi-object scenario. In contrast, our framework is uniquely designed for this long-horizon arrangement task, and to properly validate this new, complex capability, we also introduce a novel benchmark including 25 distinct object manipulation scenarios specifically designed to thoroughly test multi-step planning and reasoning capability.

3. Problem Formulation

We formulate our problem as follows: Given an image I rendered from a fixed camera C in an underlying 3D scene S, and a textual instruction T (which may range from high-level goals to detailed per-step instructions), the model must output a sequence of single-object arrangements that accomplish the user's objective while maintaining physical plausibility throughout all intermediate states.

We define three key objectives for these arrangements:
- **Collision-Free**: After each edit, the moved object should not collide with other objects in the scene.
- **Floating-Free**: After each edit, every object in the scene (except for those originally floating) should be placed solidly on a surface.
- **High Semantic Quality**: The arrangement should be semantically plausible and align well with the user's intent.

Unlike existing work that focuses on single-object insertion or offline multi-object placement, our problem addresses sequential, multi-step movements. At each step, the model must comprehend the relationships between all subtasks and objects to devise a globally coherent and actionable plan, then execute it reliably to prevent unacceptable cumulative errors. Additionally, the model must effectively search across different solution branches to identify valid paths to the goal, and when necessary, recover from dead ends caused by previous missteps.

To address these challenges, we introduce several key innovations detailed in the following sections. Section 4 presents our collaborative MLLM agents for planning, placement, and evaluation; Section 5 describes our object arrangement framework, including API-based visual tools and a constraint-based collision-free solver; and Section 4 introduces our adaptive backtracking search algorithm for discovering valid multi-step plans.

4. Multi-Agent Framework

Requiring a single MLLM to handle all 3D arrangement aspects introduces excessive processing burden and context length limitations. This problem is intensified in multi-step settings that demand global awareness. We therefore adopt a multi-agent approach with three specialized

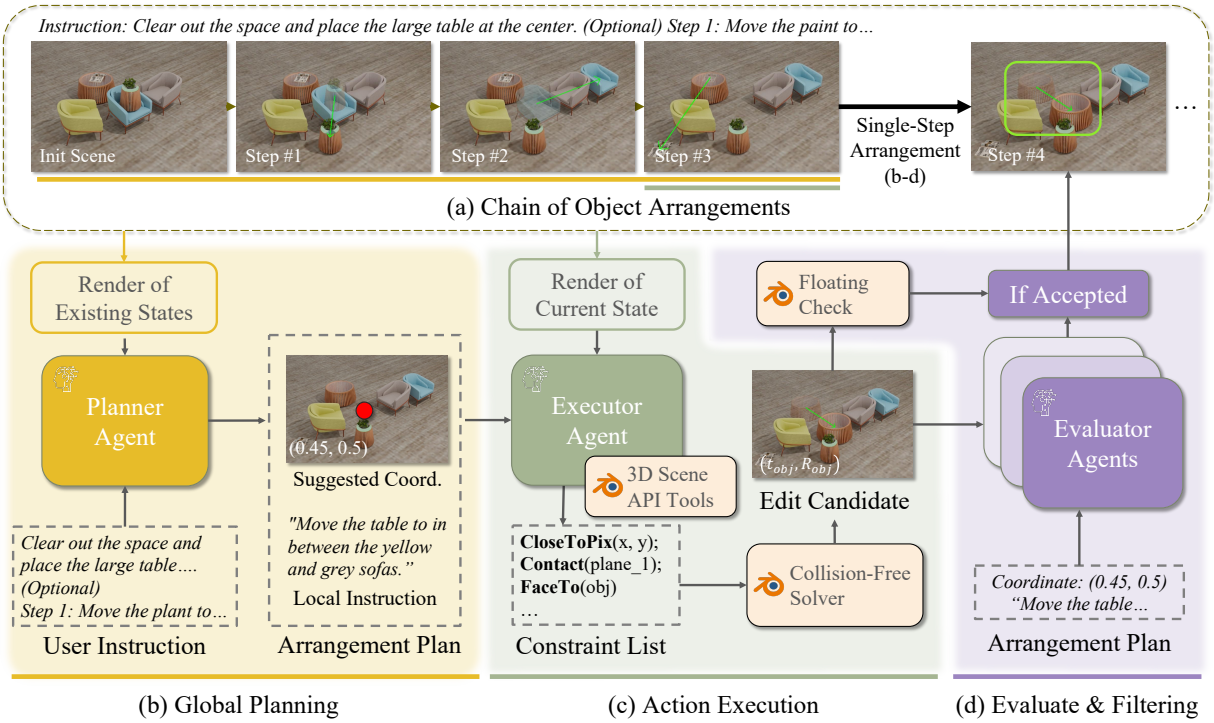


Figure 2. **VULCAN Overview.** (a) Our method solves a long-horizon task through an iterative multi-agent process. (b) The *Planner* agent examines the global context (the user instruction and all previous rendered states) to formulate a concrete plan for the current movement. (c) The *Executor* implements this single-step plan in the 3D scene using API tools and solvers. (d) A set of *Evaluators* and an automatic floating check assess the execution quality. The entire "Plan-Execute-Evaluate" loop repeats until the *Planner* validates that the final arrangement fulfills the original user instruction.

MLLMs [27, 48, 49]. Our pipeline (Fig, 2) consists of three agents per step: the ***Planner*** extracts a concrete arrangement plan, the ***Executor*** operates the 3D scene using our framework, and the ***Evaluator*** validates the result. To effectively distribute the workload, this architecture has two key designs: First, the *Planner* receives global context (i.e., information from other steps), while the *Executor* and *Evaluator* operate only within the local scope of the current step. Second, only the *Executor* interacts directly with the 3D scene; the other two agents observe rendered images.

Planning with Global Context The *Planner* generates an executable action plan based on the user's instruction and a visual history, which consists of sequential renderings from previous steps. It analyzes this visual timeline to understand scene evolution and associates past states with the user's goal to determine the most appropriate next action. Critically, the *Planner* outputs both a textual instruction and an approximate target placement as a normalized 2D pixel coordinate $\boldsymbol{c} = (x, y)$. We found this combination of language and spatial grounding to generate a less ambiguous plan.

Tool-Assisted Action Execution To address the challenges of direct MLLM execution, which demands simultaneous instruction grounding, 3D modeling, and physical plausibility checks, our *Executor* defers these complexities to dedicated external modules. Specifically, we equip the MLLM with a suite of MCP-based APIs for interactive tool usage. The *Executor* leverages these tools to project the *Planner*'s instruction and 2D coordinate c into the 3D environment. The general execution pipeline proceeds as follows: First, it performs visual selection by querying the scene to localize a target object and receptacles. Next, it translates intended spatial relationships into a set of geometric constraints [14, 27]. Finally, the *Executor* invokes a optimization-based solver to compute an optimal, collision-free pose that guarantees physical validity. Details of our visual selection, constraint formulation, and solver optimization are presented in Sec. 5.

Evaluation and Filtering The final agent, the *Evaluator*, performs a visual check on the arrangement, examining its plausibility and consistency with the *Planner*'s instruction. Given the resulting image from the *Executor*, the agent provides one of five categorical ratings: terrible, bad, fair, good, or excellent. Due to the inherent uncertainty in current MLLMs, the *Evaluator* can occasionally produce hallucinated evaluations (e.g., giving an "excellent"

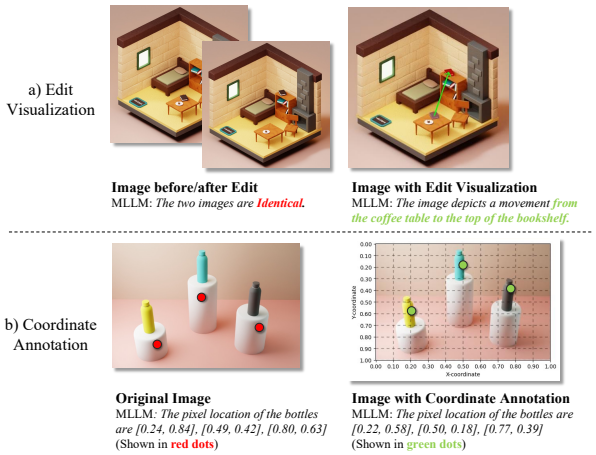


Figure 3. **Impact of visual annotation.** (a) Edit visualization improves the MLLM's spatial arrangement recognition; (b) Coordinate annotations enhance its object localization accuracy.

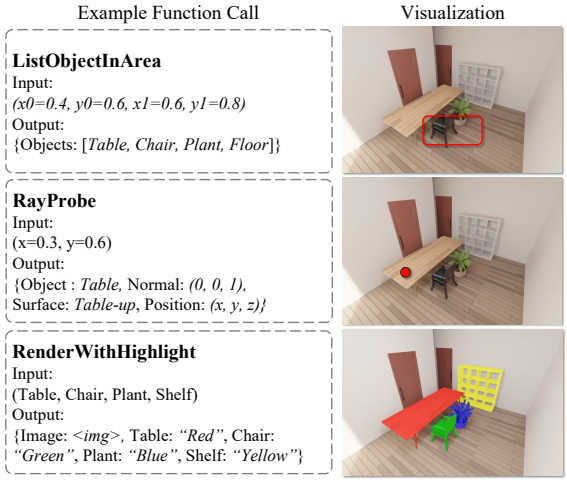


Figure 4. **Visual Probing Tools.** For each API, we present an example function call format, along with a result visualization.

rating to a completely incorrect placement). To mitigate this, we employ consensus-based filtering. This process involves polling multiple *Evaluator* agents and calculating an average judgment score, mapped from -2 (terrible) to +2 (excellent). A solution is only accepted if this consensus score is positive. Finally, in addition to the MLLM-based evaluation, we employ a rule-based check to detect physically implausible placements, such as floating objects. This deterministic check complements the agent's learned judgment by catching obvious physical errors that a model might overlook. Any movement that violates these physical constraints is automatically rejected.

Visual Annotation To enhance agent reasoning, we augment input images with two features (Fig. 3). First, to provide precise spatial grounding, we add pixel coordinate labels and a dashed grid in the normalized pixel space. This grid helps the *Planner* to generate accurate target coordinates and for the *Executor* to ground its tools. Second, we visualize the completed action with an arrow connecting the object's original location to its target. This clearly displays the start and end states, providing context for the *Planner* as it reviews the visual history and enabling the *Evaluator* to effectively assess the most recent edit.

Adaptive Backtracking During multi-step execution, errors from previous steps can render subsequent steps infeasible, leading to dead ends. For example, poor placement of initial objects on a table may leave insufficient space for later ones. To overcome this, we introduce an adaptive backtracking search algorithm, inspired by classic methods [10, 29], to efficiently recover from these errors. Our algorithm maintains an *anchor* step, which serves as the restart position for failed attempts. This anchor is adaptively reset: it moves to half the current depth when an attempt fails, or it advances when the action sequence reaches a new maximum length from the whole run (e.g. successfully placed three bottles on the table, while previous attempts placed at most two).

5. Multi-Tool Library

As described in Sec. 4, the *Executor* agent's primary role is to translate the *Planner*'s 2D-based instruction into a physically valid 3D action. To bridge this 2D-to-3D gap, the *Executor* is equipped with a visual tool library that defines a clear, step-by-step workflow:

1. **Visual Probing:** The agent queries the 3D scene to "see" and identify the 3D objects and planes.
2. **Constraint Formulation:** The agent assembles a placement plan as a list of geometric constraints.
3. **Optimization:** The constraints are passed to a solver, which computes the optimal valid 3D pose.

Visual Probing We equip the agent with three visual probing tools, to interactively analyze the 3D scene layout.

- **ListObjectsInArea**: Returns object names within or partially overlapping a specified image area. This helps the MLLM identify objects visible in that region.
- **RayProbe**: Casts a ray from the camera at a given pixel coordinate and returns information about the first hit point. This includes the 3D location, the name of the hit object, and the name and normal of the hit plane.
- **RenderWithHighlight**: Renders an instance segmentation image with specified objects highlighted. This function helps the MLLM confirm the correlation between object names and their visual representations.

During a typical *Executor* run, we observe that the agent calls **ListObjectsInArea** and **RayProbe** to retrieve the names of relevant objects and planes, including the object to be edited and any reference objects. If the name

alone is ambiguous (e.g., multiple objects named "Book"), the agent will then call **RENDERWITHHIGHLIGHT** to visually discern the correct target object from the others.

Constraint Formulation After identifying the necessary 3D elements, the agent assembles a placement plan as a list of constraints. The supported constraints are:

- **CLOSETOPIX**(obj, x, y): The object obj should be located close to the pixel coordinate (x, y) in camera view.
- **CONTACT**(obj, dir, p): The plane of the object obj's bounding box facing dir should contact the plane p.
- **NOOVERHANG**(obj, dir, p): The plane of the object's bounding box facing dir should be fully inside the convex hull of plane p.
- **DISTANCE**$(obj, obj_2, dist)$: The Euclidean distance between obj and obj_2 should be close to $dist$.
- **FACETO**$(obj, obj_2/p)$: The facing direction of object obj should point towards the center of obj_2, or align with the normal direction of plane p.
- **ROTATE**$(obj, degree)$: The object should rotate counterclockwise by $degree$.

Together, these constraints form a comprehensive vocabulary that ensures strong geometric plausibility and is applicable to a wide variety of 3D object arrangement tasks.

Optimization Given that the target pixel c and constraints from previous steps already define a small search space, we adopt an efficient, sampling-based approach. Inspired by preconditioning techniques in numerical analysis [4, 8], we first randomly perturb the target pixel coordinate c into a batch of variants. Next, we apply the solver to each variant, generating a set of candidate poses distributed around the original target. The solver employs an AdamW [37] optimizer to minimize constraint losses. Following optimization, we validate each solved candidate by recalculating its constraint losses in the original pixel coordinate space, and remove any candidates whose error exceeds the threshold τ. Finally, a collision detector filters these candidates, and we select the valid, collision-free pose with the lowest constraint error. This method, detailed in Alg. 1, efficiently ensures a non-colliding solution.

Algorithm 1: Constraint-based Pose Solver

Parameter: pixel coordinate variance σ_{pix}, loss threshold τ
Input: Suggested pixel coordinate $\boldsymbol{c}$, Constraint list $D = \{d_{1,\dots,l}\}$
Output: The target object pose $T_{obj} = (t_{obj}, R_{obj})$

$\boldsymbol{c}'_{1,\dots,n} \leftarrow \text{Sample}(\mathcal{N}(\boldsymbol{c}, \sigma_{pix}))$;
$T_{1,\dots,n} \leftarrow \text{Optimize}(D, \boldsymbol{c}'_{1,\dots,n})$;
/* Pre-filtering with constraint errors */
$T^{pre}_{1,\dots,m} \leftarrow \text{FilterByError}(T_{1,\dots,n}, D, \boldsymbol{c}, \tau)$;
for T_i *in* $\text{SortByError}(T^{pre}_{1,\dots,m})$ **do**
  if *not* $\text{CollisionDetected}(T_i)$ **then**
    return T_i;
end
return "Failed";

6. Experiments

We conduct all experiments using Gemini-2.5-pro, a state-of-the-art MLLM. Our object arrangement framework, including the constraint-based solver and API visual functions, is implemented using Blender's Python extension packages integrated with the Blender-MCP library [3]. For each arrangement step, we employ 3 *Evaluator* agents and execute 4 parallel attempts, selecting the valid solution with the highest average evaluation score. If no solution is found in all attempts, the step is marked as failed and the system proceeds with backtracking.

6.1. Dataset and Metrics

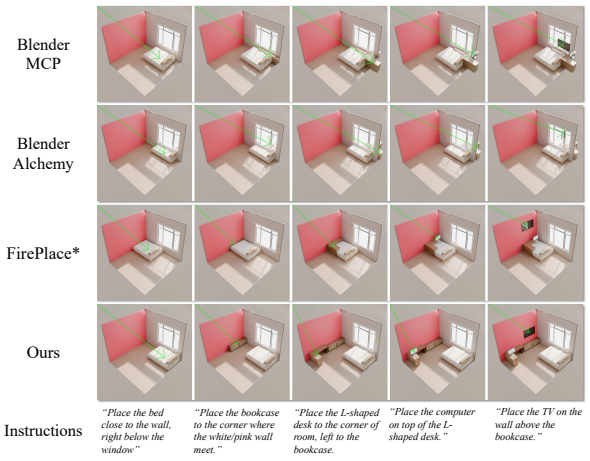
We evaluate our model's ability to produce plausible and precise multi-step arrangements through comprehensive benchmark comparisons against baseline approaches. Our evaluation dataset comprises 25 carefully curated scenes sourced from BlenderKit [2], InfiniGen [43, 44], and BlenderGym [19], encompassing 111 unit tasks in total. To ensure fair comparison, all baseline methods receive identical reference per-step instructions and execute the same sequence of operations. The task design deliberately introduces interdependencies where subsequent steps can be affected by preceding operations without proper execution.

We assess performance at every step using the following metrics, averaged across all tasks:

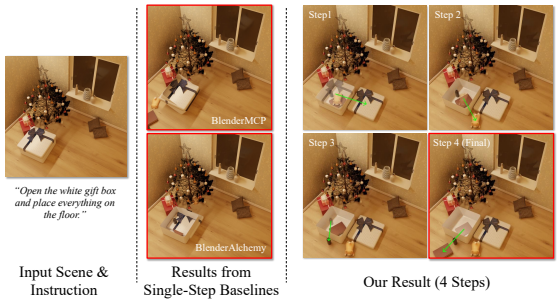
- **Collision Rate**: The proportion of steps that result in a collision between the moved object and other elements.
- **Floating Rate**: The proportion of steps that cause any object in the scene to become unsupported (i.e., "floating").
- **Plausibility**: The physical realism of the arrangement, assessed by an MLLM, scored from 0 (poor) to 4 (great).
- **Consistency**:The alignment between the user's instructions and the resulting arrangement, assessed by an MLLM and scored from 0 to 4.

6.2. Baselines Comparisons

We compare **VULCAN** against three recent MLLM-based baselines reflecting different components of our system: **BlenderAlchemy** [26] uses an executor-evaluator MLLM loop for script-based Blender editing. While it shares our multi-agent structure, it lacks our interactive visual tools and constraint-based solver (we provide object poses as input to its script representation). **Fireplace** [27] is a state-of-the-art constraint-based MLLM solver for object placement,



(a) Benchmark Comparison (Per-Step Instruction)



(b) Comparison with General Instruction

Figure 5. **Qualitative Comparisons.** We compare results using two types of guidance: detailed per-step instructions (*Top*) and general, abstract instructions (*Bottom*). *Top*: When given per-step instructions, our method outperforms the baselines. *Bottom*: Using general instructions, our multi-step approach produces reasonable intermediate states and yields a significantly better final state (marked in red) compared to single-step baselines.

Method	Coll.%↓	Fl.%↓	Plausibility ↑	Consistency↑
Blender-MCP	0.459	0.774	3.348	2.973
BlenderAlchemy	0.631	0.676	3.368	2.770
FirePlace*	0.513	0.225	3.515	3.135
Ours	**0.000**	**0.000**	**3.796**	**3.592**

Table 1. **Quantitative Comparisons.** We compare our method against baselines using several metrics: the average rate of collision (*Coll.%*), the rate of floating (*Fl.%*), va isual plausibility score (*Plausibility*), and a consistency score between the text instruction and edited image (*Consistency*).

s

analogous to our Executor. We reimplemented it for our setting, but as a standalone solver, it lacks our broader multi-agent framework (Planner and Evaluator) for global context and multi-step reasoning. Finally, **Blender-MCP** [3] serves as a simpler, tool-only baseline interacting with 3D scenes via basic callable functions (e.g., GETSCENEINFO, RENDERIMAGE, EXECUTECODE). Configured with its default toolset, it lacks specialized agent roles, a constraint solver, and multi-step capabilities.

Method	Win % / Tie %		
	Const.	Plaus.	Phys.
Ours vs. BlenderAlchemy [26]	**62.1**/22.7	**65.9**/24.7	**70.0**/20.0
Ours vs. Blender-MCP [3]	**60.5**/25.0	**59.0**/25.8	**62.9**/26.1
Ours vs. FirePlace* [27]	**58.8**/30.2	**54.4**/35.0	**58.0**/31.5

Table 2. **Results on Human Study.** We conduct a human study comparing our model against baselines across three criteria: Instruction Following (*Const.*), Visual Plausibility (*Plaus.*), and Physical Plausibility (*Phys.*). Our model is significantly preferred by users across all aspects. Fireplace was re-implemented with task-specific adaptations as its code is not publicly available.

As Tab. 1 demonstrates, **VULCAN** substantially outperforms all baselines. Driven by our collision-free solver and filtering techniques, our approach precisely eliminates collisions and floating artifacts while generating more realistic, accurate movements. Qualitatively (Fig. 5), **VULCAN** succeeds where baselines struggle. In the first benchmark example, BlenderAlchemy and Blender-MCP fail completely without effective tools, and Fireplace suffers from cascading collisions after initially moving the bed. In contrast, our method achieves artifact-free results. The second example further illustrates that our multi-step approach successfully handles complex tasks that cause single-step baselines to fail.

6.3. Human Evaluation

Beyond MLLM evaluations, we conducted a 30-subject human study comparing our model against the baselines. Participants viewed paired results and selected the output that was more physically correct, visually plausible, and consistent with the given instructions. As shown in Tab. 2, users overwhelmingly prefer our method, yielding a substantial win rate over all competitors.

6.4. Ablation Study

We validate our architectural components through ablation experiments against five variants: **Model w/o Multi Tool Library** replaces the constraint-based solver and visual API with Blender-MCP [3] functions. **Model w/o Backtracking** disables adaptive backtracking, accepting the best available solution regardless of artifacts (solver allows collisions). **Model w/o MCP Tools** removes all MCP [23] tools (including Blender-MCP [3]), requiring the model to directly generate raw Blender [1] Python scripts. **Model w/o Planner Coordinates** variant removes coordinate guidance from the *Planner* agent. **Model w/ Single Agent** replaces our collaborative multi-agent system with a single MLLM agent.

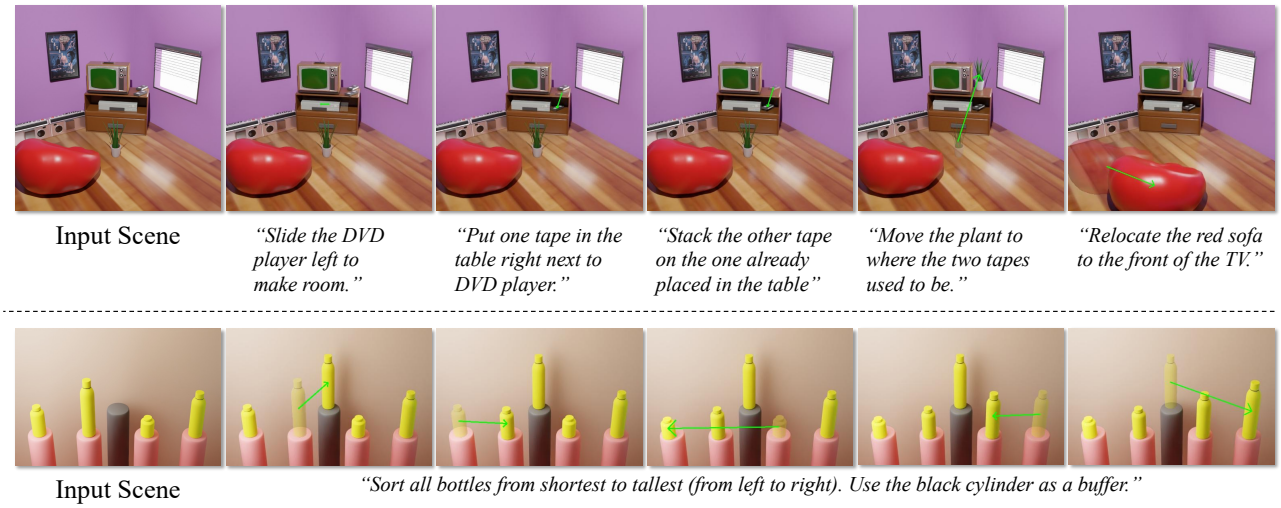


Figure 6. **More Visual Results.** Our model tackles complex arrangement tasks with per-step prompts (*top*) or general instructions (*bottom*).

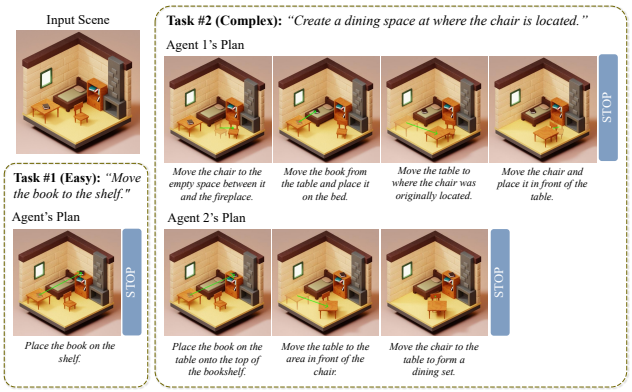


Figure 7. **Adaptive Planning.** Our model plans adaptively for different tasks and can generate diverse plans for the same objective. The plan output is shown below the corresponding edited image.

As shown in Tab. 3, the full model achieves optimal performance across all metrics. Removing the solver and visual tools substantially degrades plausibility and quality, though our multi-agent design and backtracking mechanism keep performance above the Blender-MCP baseline. Without backtracking, the model occasionally accepts suboptimal solutions, resulting in rare collision and floating artifacts. Finally, the single-agent variant maintains zero physical artifacts but suffers in visual quality and textual consistency, confirming that our multi-agent decomposition is vital for preserving reasoning quality in complex scenarios.

6.5. Visual Results

We present additional qualitative results demonstrating our method's performance across various tasks. Fig. 6 illustrates our model's ability to handle complex scenarios requiring robust placement pipelines and effective global reasoning across multiple steps. Fig. 7 further demonstrates the model's flexibility in generating diverse plans: It adapts its planning to different tasks and produces diverse solutions for identical objectives across runs.

Method	Coll.%↓	Fl.%↓	Plaus.↑	Const.↑
w/o Multi-Tool Library	0.495	0.711	3.484	3.103
w/o Backtracking	0.036	0.054	3.703	3.549
w/o MCP Tools	0.603	0.738	3.357	3.067
w/o Planner's Coordinates	**0.000**	**0.000**	3.772	3.526
Single Agent	**0.000**	**0.000**	3.623	3.328
Ours	**0.000**	**0.000**	**3.796**	**3.592**

Table 3. **Ablation Study Results.** Our complete model achieves substantially better performance than any ablated variant.

7. Conclusion

We propose **VULCAN**, a robust pipeline for multi-step, text-to-3D object arrangement. By integrating novel visual APIs, a collision-free solver, collaborative agents, and adaptive backtracking, our model effectively handles complex tasks and substantially outperforms state-of-the-art baselines. Currently, to manage MLLM context lengths and maintain reasoning performance, our system relies on single-camera views. This restricts performance on tasks requiring unseen perspectives, such as placing a chair behind a wall. However, as long-context MLLMs evolve, our pipeline can be readily extended to multi-view scenarios.

7.1. Acknowledgement

This work was supported by computational resources provided by Google Cloud. We thank Alireza Fathi, Yanan Bao, and Ian Huang for their assistance in reproducing the results from Fireplace, as well as Jiacheng Chen for his technical support with Blender scripting.

References

[1] Blender. https://www.blender.org/, . 7

[2] Blenderkit. https://www.blenderkit.com/, . 6

[3] Blender-mcp. https://github.com/ahujasid/blender-mcp?tab=readme-ov-file, . 6, 7

[4] Owe Axelsson. *Iterative Solution Methods*. Cambridge University Press, 1994. 6

[5] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 2, 3

[6] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14455–14465, 2024. 3

[7] Guo Chen, Yin-Dong Zheng, Jiahao Wang, Jilan Xu, Yifei Huang, Junting Pan, Yi Wang, Yali Wang, Yu Qiao, Tong Lu, et al. Videollm: Modeling video sequence with large language models. *arXiv preprint arXiv:2305.13292*, 2023. 3

[8] Ke Chen. *Matrix Preconditioning Techniques and Applications*. Cambridge University Press, 2005. 6

[9] Tianrun Chen, Chunan Yu, Yuanqi Hu, Jing Li, Tao Xu, Runlong Cao, Lanyun Zhu, Ying Zang, Yong Zhang, Zejian Li, et al. Img2cad: Conditioned 3-d cad model generation from single image with structured visual geometry. *IEEE Transactions on Industrial Informatics*, 2025. 3

[10] Dah-Ming Chiu and Raj Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN systems*, 17(1):1–14, 1989. 5

[11] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 3

[12] Fernanda De La Torre, Cathy Mengying Fang, Han Huang, Andrzej Banburski-Fahey, Judith Amores Fernandez, and Jaron Lanier. Llmr: Real-time prompting of interactive worlds using large language models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pages 1–22, 2024. 3

[13] 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. 3

[14] Mohamed el amine Boudjoghra, Ivan Laptev, and Angela Dai. Scanedit: Hierarchically-guided functional 3d scan editing, 2025. 2, 3, 4

[15] 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. *arXiv preprint arXiv:2305.15393*, 2023. 2, 3

[16] Matthew Fisher, Daniel Ritchie, Manolis Savva, Thomas Funkhouser, and Pat Hanrahan. Example-based synthesis of 3d object arrangements. In *ACM SIGGRAPH Asia 2012 papers*, 2012. 3

[17] Matthew Fisher, Manolis Savva, Yangyan Li, Pat Hanrahan, and Matthias Nießner. Activity-centric scene synthesis for functional 3d scene modeling. *ACM Transactions on Graphics (TOG)*, 34(6):1–13, 2015. 3

[18] Sreyan Ghosh, Sonal Kumar, Ashish Seth, Chandra Kiran Reddy Evuru, Utkarsh Tyagi, S Sakshi, Oriol Nieto, Ramani Duraiswami, and Dinesh Manocha. Gama: A large audio-language model with advanced audio understanding and complex reasoning abilities. *arXiv preprint arXiv:2406.11768*, 2024. 3

[19] Yunqi Gu, Ian Huang, Jihyeon Je, Guandao Yang, and Leonidas Guibas. Blendergym: Benchmarking foundational model systems for graphics editing. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18574–18583, 2025. 3, 6

[20] Ayaan Haque, Matthew Tancik, Alexei Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 3

[21] Shun-ichiro Hayashi, Daichi Mukunoki, Tetsuya Hoshino, Satoshi Ohshima, and Takahiro Katagiri. 3dify: a framework for procedural 3d-cg generation assisted by llms using mcp and rag. *arXiv preprint arXiv:2510.04536*, 2025. 2

[22] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023. 3

[23] Xinyi Hou, Yanjie Zhao, Shenao Wang, and Haoyu Wang. Model context protocol (mcp): Landscape, security threats, and future research directions. *arXiv preprint arXiv:2503.23278*, 2025. 2, 7

[24] Ziniu Hu, Ahmet Iscen, Aashi Jain, Thomas Kipf, Yisong Yue, David A Ross, Cordelia Schmid, and Alireza Fathi. Scenecraft: An llm agent for synthesizing 3d scenes as blender code. In *Forty-first International Conference on Machine Learning*, 2024. 2, 3

[25] Haifeng Huang, Yilun Chen, Zehan Wang, Rongjie Huang, Runsen Xu, Tai Wang, Luping Liu, Xize Cheng, Yang Zhao, Jiangmiao Pang, et al. Chat-scene: Bridging 3d scene and large language models with object identifiers. *Advances in Neural Information Processing Systems*, 37: 113991–114017, 2024. 3

[26] Ian Huang, Guandao Yang, and Leonidas Guibas. Blenderalchemy: Editing 3d graphics with vision-language models. In *European Conference on Computer Vision*, pages 297–314. Springer, 2024. 2, 3, 6, 7

[27] Ian Huang, Yanan Bao, Karen Truong, Howard Zhou, Cordelia Schmid, Leonidas Guibas, and Alireza Fathi. Fireplace: Geometric refinements of llm common sense reasoning for 3d object placement. *arXiv preprint arXiv:2503.04919*, 2025. 2, 3, 4, 6, 7

[28] Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. An

LLM compiler for parallel function calling. In *Proceedings of the 41st International Conference on Machine Learning*, pages 24370–24391. PMLR, 2024. 2

[29] Donald E Knuth, James H Morris, Jr, and Vaughan R Pratt. Fast pattern matching in strings. *SIAM journal on computing*, 6(2):323–350, 1977. 5

[30] Zhengfei Kuang, Kyle Olszewski, Menglei Chai, Zeng Huang, Panos Achlioptas, and Sergey Tulyakov. Neroic: Neural rendering of objects from online image collections. *ACM Transactions on Graphics (TOG)*, 41(4):1–12, 2022. 3

[31] Zhengfei Kuang, Fujun Luan, Sai Bi, Zhixin Shu, Gordon Wetzstein, and Kalyan Sunkavalli. Palettenerf: Palette-based appearance editing of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20691–20700, 2023. 3

[32] Jiaman Li, Jiajun Wu, and C Karen Liu. Object motion guided human motion synthesis. *ACM Transactions on Graphics (TOG)*, 42(6):1–11, 2023. 3

[33] Jiaman Li, Alexander Clegg, Roozbeh Mottaghi, Jiajun Wu, Xavier Puig, and C Karen Liu. Controllable human-object interaction synthesis. In *European Conference on Computer Vision*, pages 54–72. Springer, 2024. 3

[34] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023. 2, 3

[35] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5773–5783, 2021. 3

[36] Weiwen Liu, Xu Huang, Xingshan Zeng, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong Wang, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu, Xinzhi Wang, Yong Liu, Yasheng Wang, Duyu Tang, Dandan Tu, Lifeng Shang, Xin Jiang, Ruiming Tang, Defu Lian, Qun Liu, and Enhong Chen. Toolace: Winning the points of llm function calling. *ArXiv*, abs/2409.00920, 2024. 2

[37] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017. 6

[38] Sining Lu, Guan Chen, Nam Anh Dinh, Itai Lang, Ari Holtzman, and Rana Hanocka. Ll3m: Large language 3d modelers, 2025. 3

[39] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Videogpt+: Integrating image and video encoders for enhanced video understanding. *arxiv*, 2024. 3

[40] Kiyohiro Nakayama, Jan Ackermann, Timur Levent Kesdogan, Yang Zheng, Maria Korosteleva, Olga Sorkine-Hornung, Leonidas J Guibas, Guandao Yang, and Gordon Wetzstein. Aipparel: A multimodal foundation model for digital garments. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 8138–8149, 2025. 3

[41] Francesco Palandra, Andrea Sanchietti, Daniele Baieri, and Emanuele Rodola. Gsedit: Efficient text-guided editing of 3d objects via gaussian splatting. *arXiv preprint arXiv:2403.05154*, 2024. 3

[42] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 3

[43] Alexander Raistrick, Lahav Lipson, Zeyu Ma, Lingjie Mei, Mingzhe Wang, Yiming Zuo, Karhan Kayan, Hongyu Wen, Beining Han, Yihan Wang, Alejandro Newell, Hei Law, Ankit Goyal, Kaiyu Yang, and Jia Deng. Infinite photorealistic worlds using procedural generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12630–12641, 2023. 3, 6

[44] 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. 3, 6

[45] Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. Xcube: Large-scale 3d generative modeling using sparse voxel hierarchies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4209–4219, 2024. 3

[46] Haozhan Shen, Kangjia Zhao, Tiancheng Zhao, Ruochen Xu, Zilun Zhang, Mingwei Zhu, and Jianwei Yin. ZoomEye: Enhancing multimodal LLMs with human-like zooming capabilities through tree-based image exploration. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 6613–6629, Suzhou, China, 2025. Association for Computational Linguistics. 2

[47] Yawar Siddiqui, Antonio Alliegro, Alexey Artemov, Tatiana Tommasi, Daniele Sirigatti, Vladislav Rosov, Angela Dai, and Matthias Nießner. Meshgpt: Generating triangle meshes with decoder-only transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19615–19625, 2024. 3

[48] Chunyi Sun, Junlin Han, Weijian Deng, Xinlong Wang, Zishan Qin, and Stephen Gould. 3d-gpt: Procedural 3d modeling with large language models. In *2025 International Conference on 3D Vision (3DV)*, pages 1253–1263. IEEE, 2025. 3, 4

[49] 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. 3, 4

[50] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 2, 3

[51] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 3

[52] Kai Wang, Manolis Savva, Angel X Chang, and Daniel Ritchie. Deep convolutional priors for indoor scene syn-

thesis. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 3

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

[54] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers. In *2021 International Conference on 3D Vision (3DV)*, pages 106–115. IEEE, 2021.

[55] Qiuhong Anna Wei, Sijie Ding, Jeong Joon Park, Rahul Sajnani, Adrien Poulenard, Srinath Sridhar, and Leonidas Guibas. Lego-net: Learning regular rearrangements of objects in rooms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19037–19047, 2023. 3

[56] Philip Welsby and Bernard MY Cheung. Chatgpt, 2023. 2, 3

[57] Zhen Wu, Jiaman Li, Pei Xu, and C Karen Liu. Human-object interaction from human-level instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11176–11186, 2025. 3

[58] Kun Xu, Kang Chen, Hongbo Fu, Wei-Lun Sun, and Shi-Min Hu. Sketch2scene: Sketch-based co-retrieval and co-placement of 3d models. *ACM Transactions on Graphics (TOG)*, 32(4):1–15, 2013. 3

[59] Yinghao Xu, Menglei Chai, Zifan Shi, Sida Peng, Ivan Skorokhodov, Aliaksandr Siarohin, Ceyuan Yang, Yujun Shen, Hsin-Ying Lee, Bolei Zhou, et al. Discoscene: Spatially disentangled generative radiance fields for controllable 3d-aware scene synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4402–4412, 2023. 3

[60] Jianing Yang, Xuweiyi Chen, Nikhil Madaan, Madhavan Iyengar, Shengyi Qian, David F Fouhey, and Joyce Chai. 3d-grand: A million-scale dataset for 3d-llms with better grounding and less hallucination. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29501–29512, 2025. 3

[61] 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. 2, 3

[62] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023. 2

[63] Hanxun Yu, Wentong Li, Song Wang, Junbo Chen, and Jianke Zhu. Inst3d-lmm: Instance-aware 3d scene understanding with multi-modal instruction tuning, 2025. 3

[64] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. Nerf-editing: geometry editing of neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18353–18364, 2022. 3

[65] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023. 3

[66] Longwen Zhang, Qiwei Qiu, Hongyang Lin, Qixuan Zhang, Cheng Shi, Wei Yang, Ye Shi, Sibei Yang, Lan Xu, and Jingyi Yu. Dreamface: Progressive generation of animatable 3d faces under text guidance. *arXiv preprint arXiv:2304.03117*, 2023. 3

[67] Yunzhi Zhang, Zizhang Li, Matt Zhou, Shangzhe Wu, and Jiajun Wu. The scene language: Representing scenes with programs, words, and embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24625–24634, 2025. 2, 3