TabletopGen: Instance-Level Interactive 3D Tabletop Scene Generation from Text or Single Image

Ziqian Wang[1,3,2*] Yonghao He[2*†] Licheng Yang[1,3] Wei Zou[1,3] Hongxuan Ma[3]
Liu Liu[2] Wei Sui[2] Yuxin Guo[1,3] Hu Su[3✉]

[1]School of Artificial Intelligence, University of Chinese Academy of Sciences
[2]D-Robotics
[3]State Key Laboratory of Multimodal Artificial Intelligence Systems (MAIS), Institute of Automation, Chinese Academy of Sciences

[*]**Equal Contribution** [†]**Project Leader** [✉]**Corresponding Author**

Figure 1. We present **TabletopGen**, a training-free, fully automatic unified framework that generates instance-level interactive 3D tabletop scenes. As shown on the left, TabletopGen can generate visually realistic, detail-rich, plausibly arranged, and collision-free 3D scenes from either text or a single image input. As shown on the right, our framework can produce a wide variety of tabletop scenes, spanning different shapes, styles, and functional categories.

Abstract

*Generating high-fidelity, physically interactive 3D simulated tabletop scenes is essential for embodied AI—especially for robotic manipulation policy learning and data synthesis. However, current text- or image-driven 3D scene generation methods mainly focus on large-scale scenes, struggling to capture the high-density layouts and complex spatial relations that characterize tabletop scenes. To address these challenges, we propose **TabletopGen**, a training-free, fully automatic framework that generates diverse, instance-level interactive 3D tabletop scenes. TabletopGen accepts a reference image as input, which can be synthesized by a text-to-image model to enhance scene diversity. We then perform instance segmentation and completion on the reference to obtain per-instance images. Each instance is reconstructed into a 3D model followed by canonical coordinate alignment. The aligned 3D models then undergo pose and scale estimation before being assembled into a collision-free, simulation-ready tabletop scene. A key component of our framework is a novel pose and scale alignment approach that decouples the complex spatial reasoning into two stages: a Differentiable Rotation Optimizer for precise rotation recovery and a Top-view Spatial Alignment mechanism for robust translation and scale estimation, enabling accurate 3D reconstruction from 2D reference. Extensive experiments and user studies show that TabletopGen achieves state-of-the-art performance, markedly surpassing existing methods in visual fidelity, layout accuracy, and physical plausibility, capable of generating realistic tabletop scenes with rich stylistic and spatial diversity. Our code will be publicly available.*

Project Page: https://d-robotics-ai-lab.github.io/TabletopGen-Page/

1. Introduction

3D scene generation has broad applications such as embodied AI [12, 36, 61], VR/AR [19, 22, 41], gaming [48, 53, 59], and digital twins [11, 35, 50]. With the rapid progress of embodied AI, 3D scenes are required not only to be photorealistic but also instance-level and physically interactive to support the training of robot policies in simulation[32, 44, 52, 56]. In current embodied AI research, a primary goal is to enable robots to execute complex object manipulation, tabletops are the "*last meter*" of this environment: it is the fundamental stage for most fine-grained interactions and the primary setting for complex robotic manipulation tasks[25, 66]. Therefore, automated, large-scale generation of high-fidelity, interactive tabletop scenes is a critical component for advancing embodied manipulation policy learning.

A natural question arises: what tabletop scenes are truly suitable for embodied manipulation in simulation and training? Based on current embodied AI demands [4, 25, 61], we distill three core criteria for simulation-ready tabletop scenes: **(1) Interactive High-Quality Instances:** each object is an independent, geometrically complete 3D model, enabling fine-grained actions on any single instance; **(2) Functionally-Semantic Layout:** objects are arranged according to function and commonsense (e.g., mouse to the side of the keyboard), not just randomly stacked; **(3) Precise and Physically Plausible Spatial Relations:** each instance's pose is accurate, ensuring the scene is collision-free, with no interpenetration or floating.

With these criteria, we conducted a comprehensive evaluation of representative methods and found severe shortcomings. (1) Retrieval-based methods [6, 11, 25, 58] fail on instance quality and diversity. By retrieving instances from fixed 3D asset libraries [13, 14, 18, 32], their content is restricted to a finite set of assets that often fail to precisely match the target scene. (2) Text-driven methods [2, 6, 19, 22, 49, 58] struggle to capture functional small-object layouts on tabletops. These methods rely on LLM-based planning and predefined spatial constraints for large-scale indoor scene arrangements, which do not adapt well to high-density, semantically constrained tabletop layouts. (3) Single-image reconstruction methods [3, 8, 28, 33, 36] suffer from incomplete instances and pose/scale mismatch. Limited by severe single-view occlusions, they lack a robust way to recover each instance's complete appearance and precise pose. Consequently, their reconstructions often exhibit incorrect object counts, incomplete or distorted geometries, and unstable layouts.

To address the above challenges, we propose **TabletopGen**: a unified, training-free framework that automatically generates instance-level interactive 3D tabletop scenes from either text or a single image, producing visually realistic, semantically plausible, and collision-free layouts. To obtain more realistic, precise layout information and leverage the diversity of text-to-image models, we first convert the text input to a reference image; image input is used directly. Subsequently, we perform segmentation and multi-modal generative completion—rather than standard inpainting—to restore 2D instances from occluded or blurred regions, and then generate high-quality models via image-to-3D. This per-instance generation design ensures aligned object counts and appearance with high geometric flexibility. To precisely estimate the instance's rotation, translation, and scale, we propose **a novel Pose and Scale Alignment approach** that solves the problem in two sequential phases: A Differentiable Rotation Optimizer (DRO) estimates rotations; a Top-view Spatial Alignment (TSA) mechanism that infers globally consistent translations and scales via a top-view image and commonsense size priors. Finally, we assemble all 3D instances and attach physical properties in a simulator to obtain physically plausible 3D scenes.

In summary, our contributions are threefold:

- We propose TabletopGen, a fully automated unified framework that, via instance-wise 3D reconstruction and two-stage pose–scale alignment, generates diverse, instance-level, physically interactive 3D tabletop scenes from either text or a single image.
- We introduce a novel 3D pose and scale alignment method that robustly recovers the rotation, translation, and scale of all instances from a 2D reference, enabling collision-free scene assembly.
- Extensive experiments and a large-scale user study demonstrate that TabletopGen achieves state-of-the-art performance, markedly surpassing existing methods in terms of visual fidelity, layout accuracy, and physical plausibility. We will release our source code, positioning TabletopGen as a robust tool that can be integrated into the tabletop stage of larger scene generation pipelines.

2. Related Works

Generating photorealistic and diverse 3D scenes has been a long-standing pursuit in computer vision [12, 40, 51, 53, 55]. The advent of diffusion models [10, 26] and large language models (LLMs) [5, 67] has markedly accelerated the progress. Current research follows two main directions: text-driven generation methods using LLMs for layout reasoning, and image-driven reconstruction methods recovering geometry and spatial relations from a single image. Given our focus on high-fidelity, interactive tabletop scenes for embodied AI, we review these two lines and, in addition, discuss datasets specifically curated for tabletop scenes.

2.1. Text to 3D Scene Generation with LLMs.

Text-driven methods leverage LLMs for semantic and spatial reasoning, typically following two paths. The first uses LLMs to directly generate 3D layouts [16, 38, 57], which

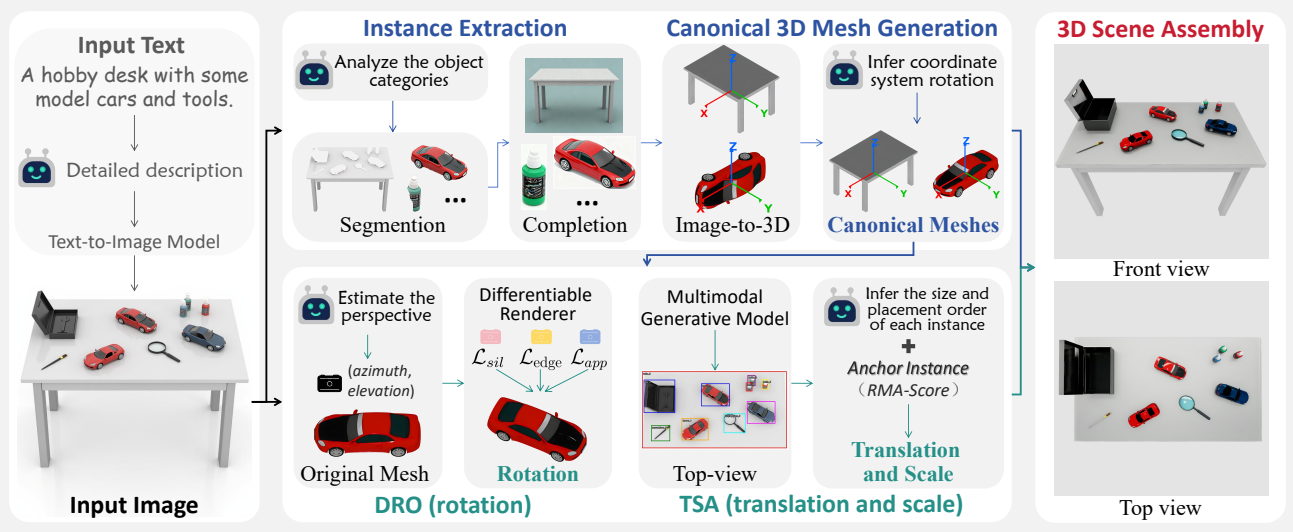


Figure 2. **Overview of our TabletopGen Framework.** Our framework accepts either text (which is first converted into a reference image) or a single image. Starting from the image, we proceed in four stages: (1) **Instance Extraction** performs category analysis, segmentation, and completion to obtain clean, high-resolution per-instance images. (2) **Canonical Model Generation** uses Image-to-3D and MLLM-based alignment to create a 3D model with canonical coordinate system for each instance. (3) Our core **Pose and Scale Alignment** stage recovers the spatial layout. The **DRO (Differentiable Rotation Optimizer)** estimates rotation by optimizing a tri-modal loss, while the **TSA (Top-view Spatial Alignment)** mechanism synthesizes a top-view image and, together with MLLM reasoning, selects an anchor instance via our RMA-Score to infer each instance's translation and scale. (4) **3D Scene Assembly** stage combines all instance models with their poses and scales in a simulator to produce the final collision-free, interactive 3D tabletop scene.

can easily produce physically implausible results. A second, more common path employs LLMs to generate high-level scene graphs or spatial constraints [2, 6, 19, 21, 33, 34, 49, 58], which are then optimized for physical feasibility. Both approaches, however, are often limited by their reliance on retrieval from fixed 3D libraries, restricting stylistic consistency and geometric flexibility. Moreover, they typically focus on large-scale indoor scenes and fail to capture the high-density, functionally layouts of tabletops.

2.2. Scene Reconstruction from A Single Image

Single-image scene reconstruction needs to tackle image alignment, occlusion, and complex spatial relations. Retrieval-based methods [11, 20, 23] match instances to 3D libraries, but are limited by asset diversity and geometric discrepancies. Diffusion-based methods [7, 28, 62] synthesize the scene at once to produce a result consistent with the input view, but they often suffer from multi-view inconsistency and incomplete geometry for occluded objects. Recent compositional generative methods [3, 22, 24, 27, 36, 60] improve flexibility by reconstructing and aligning each instance to the input image. However, under a single-view constraint, they cannot reliably recover pose for occluded or hidden parts, leading to interpenetration, floating, and other physically implausible layouts. Our work addresses this bottleneck via a novel pose and scale alignment approach.

2.3. Tabletop Scenes Dataset and Generation

Compared with room-scale datasets [15, 17, 18, 64, 68], dedicated tabletop resources are scarce. LVDiffusor [63] uses large vision–language models to synthesize plausible 2D tabletop images without 3D spatial information. TO-Scene [54] offers a large 3D tabletop dataset but is primarily manually constructed and lacks the capability for automated data generation. The recent work MesaTask [25] introduces Task-to-Scene generation and a manually-refined dataset; however, it focuses on placing objects on a rectangular plane, does not model the full table geometry, and relies on asset retrieval. In contrast, we automatically generate style-consistent tables of diverse shapes together with tabletop objects, producing complete, simulation-ready tabletop scenes for embodied AI.

3. Method

TabletopGen can generate instance-level, interactive tabletop scenes with accurate spatial relations from text or a single image. Specifically, given a text description T or an image I, our goal is to generate a tabletop scene $S = \{o_i\}_{i=1}^{n}$ composed of n object instances. Each instance o_i is defined by its 3D model m_i, scale $s_i \in \mathbb{R}^3$, translation $t_i \in \mathbb{R}^3$, and rotation $r_i \in \mathbb{R}$ around the vertical axis.

As shown in Fig. 2, our framework achieves this goal through a multi-stage architecture. Following prior works [22, 69], we argue that images provide more realistic and

precise layout information than text. For a text input T, an LLM expands it into a detailed prompt, specifying scene style, object set, and their functional layout. This prompt is then passed to a text-to-image model to synthesize a visually realistic and plausibly arranged reference image I_{ref}. Image inputs serve directly as I_{ref}. Given I_{ref}, the 3D tabletop scene is generated in the following four stages.

3.1. Instance Extraction from 2D Image

The first stage is to extract complete and high-quality 2D instances from I_{ref} for subsequent per-instance 3D generation. We first employ a Multimodal Large Language Model (MLLM) to analyze the scene and infer the object categories L_i, enabling open-vocabulary instance recognition that is not restricted by any predefined label set. This step allows TabletopGen to handle arbitrary tabletop objects.

Next, we use GroundedSAM-v2 [45] to detect each instance and obtain its segmentation mask M_i. However, the tabletop's high object density and heavy occlusion often yield masks with holes; additionally, many small-object boundaries are blurred or fragmented. To address these, instead of traditional inpainting methods, we utilize an advanced multimodal generative model for completion. Given I_{ref}, M_i, and category label L_i, the model redraws a high-resolution, visually consistent instance image. This step effectively restores occluded or blurred parts, providing a clear and complete input for the next stage.

3.2. Canonical 3D Model Generation

In the second stage, each completed 2D instance is generated into a high-quality 3D model m_i' using an image-to-3D diffusion model. This per-instance generation provides substantial flexibility in object appearance and geometry, freeing TabletopGen from the limitations of fixed 3D asset libraries. However, m_i' is typically defined in an arbitrary local coordinate system. To address this, we perform **canonical coordinate alignment**. We utilize an MLLM to analyze the model's upright orientation by combining visual cues from I_{ref} and semantic priors. We then apply the corrective rotation to align each model's local vertical axis with the tabletop world up, yielding a properly oriented, physically placeable 3D model m_i.

3.3. Pose and Scale Alignment

The third and most critical stage is to estimate each instance's **rotation** r_i, **translation** t_i, and **scale** s_i so that all canonical 3D models m_i can be assembled into a coherent, physically plausible scene. We propose a novel approach that decouples the problem into two sequential phases.

Differentiable Rotation Optimizer (DRO). In this phase, we estimate r_i for each instance. The MLLM first predicts an initial camera azimuth and elevation from I_{ref}. From this view, we render each m_i using a differentiable renderer and obtain a textured image $I_{render}(r_i)$, a soft silhouette $\hat{S}(r_i)$, and a soft edge map $\hat{E}(r_i)$ that derived from $\hat{S}(r_i)$ using a Sobel filter. We then define a **tri-modal matching loss** $\mathcal{L}_{rot}$ to align the render with the target instance $I_{instance}$ and its mask S:

$$\mathcal{L}_{rot}(r_i) = \lambda_s \mathcal{L}_{sil}(r_i) + \lambda_e \mathcal{L}_{edge}(r_i) + \lambda_a \mathcal{L}_{app}(r_i) \quad (1)$$

where λ_s, λ_e and λ_a are weighting coefficients. The first term, $\mathcal{L}_{sil}$, is a soft IoU loss for shape consistency:

$$\mathcal{L}_{sil}(r_i) = 1 - \frac{\sum(\hat{S}(r_i) \cdot S)}{\sum(S + \hat{S}(r_i) - S \cdot \hat{S}(r_i))} \quad (2)$$

The second, $\mathcal{L}_{edge}(r_i)$, is a one-sided Chamfer loss that matches contours. It penalizes the distance from $\hat{E}(r_i)$ to the nearest ground-truth edge. A distance transform D_S is pre-computed from Canny edges of the target mask S. The loss is the weighted average distance over all pixels x:

$$\mathcal{L}_{edge}(r_i) = \frac{\sum_x D_S(x) \cdot \hat{E}(x, r_i)}{\sum_x \hat{E}(x, r_i)} \quad (3)$$

Finally, $\mathcal{L}_{app}$ is a perceptual feature loss that ensures appearance similarity with DINOv2 [39] features Φ:

$$\mathcal{L}_{app}(r_i) = ||\Phi(I_{render}(r_i)) - \Phi(I_{instance})||_2^2 \quad (4)$$

We find the optimal rotation $\hat{r}_i$ by minimizing this loss via gradient descent: $\hat{r}_i = \arg\min_{r_i} \mathcal{L}_{rot}(r_i)$. This process can also optionally refine the camera pose.

Top-View Spatial Alignment (TSA). We then estimate t_i and s_i using our TSA mechanism to resolves the single-view scale ambiguity. We leverage a multimodal generative model to synthesize a top-view image I_{top} from I_{ref}. For each instance, we detect its 2D bounding box and query the MLLM for its commonsense physical size to compute:

$$A_{px} = w_{img} \cdot h_{img}, \; \varepsilon_{ratio} = |\log r_{phys} - \log r_{img}| \quad (5)$$

where A_{px} is the pixel area, and ε_{ratio} compares the aspect ratio r_{img} (from I_{top}) with the physical aspect ratio, which is obtained by projecting the physical size onto the XY-plane after applying r_i. We then define a **Ratio-Matched and Area-Weighted Score (RMA-Score)** to select a reliable scaling anchor:

$$\text{RMA}(i) = \frac{A_{px}(i)}{1 + (\varepsilon_{ratio}/\tau)^2} \quad (6)$$

where τ is a tolerance hyperparameter. The score favors instances with large area A_{px}, providing stability, and low aspect-ratio error ε_{ratio}, ensuring geometric consistency.

Table 1. **Quantitative comparison of different methods**. Our method TabletopGen achieves the best overall performance across all metrics, showing substantially superior results in Visual & Perceptual quality, GPT-4o Evaluation, and near-zero Collision Rates compared to all baselines.

Method	Visual & Perceptual			GPT Evaluation					Collision Rate(%)	
	LPIPS↓	DINOv2↑	CLIP↑	VF↑	IA↑	PP↑	Avg.↑	OR↓	Col_O↓	Col_S↓
ACDC	0.5124	0.3775	0.6696	2.38	1.90	2.57	2.28	3.55	8.23	67.95
Gen3DSR	0.4891	0.5602	0.8636	2.92	4.17	3.32	3.47	3.02	16.88	85.90
MIDI	0.4559	0.7070	0.8867	4.22	4.48	4.30	4.33	2.32	17.39	98.72
Ours	**0.4483**	**0.8383**	**0.9077**	**6.06**	**6.30**	**6.22**	**6.19**	**1.08**	**0.42**	**7.69**

The instance with the highest RMA-Score is selected as an anchor. From this anchor, we compute a global scaling factor α (meters/pixel) to convert all instance's bounding boxes from pixel to world coordinates, yielding s_i and the (x,y) components of t_i. Finally, the MLLM analyzes stacking order in I_{ref} to determine the z-coordinate of t_i, ensuring no vertical collisions.

3.4. 3D Scene Assembly

After collecting all rotation r_i, translation t_i, and scale s_i information, we proceed to the final stage: assembling a complete and physically interactive 3D scene. We import the canonical 3D models m_i into a physics-enabled simulator, Isaac Sim [37], and apply their respective (r_i, t_i, s_i) transforms. To ensure realistic physical interactions, each instance is assigned a collision property generated through convex decomposition, and gravity and friction parameters are enabled. This process converts our visually reconstructed instances into a simulation-ready 3D tabletop scene that can be directly used for downstream embodied AI tasks.

4. Experiments

4.1. Setup

Implementation details. We use ChatGPT [1] as the MLLM [1, 9, 46] for all vision-language reasoning tasks and Seedream [47], a multimodal image generation model, for synthesizing all 2D images. Per-instance 3D models are generated using a pretrained image-to-3D diffusion model, Hunyuan3D-3.0 [31]. In DRO, we render with Pytorch3D [43] and run a coarse search over the [0°, 360°) range at 5° step size to select the 8 candidate angles with the lowest loss at first. Each candidate is then refined using the Adam optimizer [30] with a learning rate of 3×10^{-2} for 140 steps. The loss weights are set to $\lambda_s = 0.5$, $\lambda_e = 0.5$, and $\lambda_a = 2.0$. In TSA, we set the tolerance hyperparameter $\tau = 0.25$ for the RMA-Score computation. The prompts, step-wise intermediate outputs, and efficiency analysis are provided in the supplementary materials.

Baselines. Since our tabletop scenes are primarily generated from 2D reference images, we compare our method with state-of-the-art single-image 3D scene reconstruction methods, including the retrieval-based method ACDC [11], the compositional generation method Gen3DSR [3], and the diffusion-based method MIDI [28]. To further illustrate our text-to-scene capability, we also conduct a qualitative visual comparison with the recent text-driven tabletop generation method MesaTask [25], as text-based generation lacks consistent reference images for quantitative evaluation.

Metrics. We comprehensively evaluate the generated scenes following existing scene generation methods [22, 24, 25, 27, 34, 36, 60], from three complementary aspects. **(1) Visual–perceptual quality.** We report LPIPS [65], DINOv2 [39], and CLIP [42] to measure perceptual similarity, visual consistency, and semantic alignment to the reference image. **(2) Physical plausibility.** We compute the proportion of pairs of colliding objects, Col_C, and the percentage of scenes containing collisions, Col_S, to measure the reasonableness of the generated scene. **(3) Multi-dimensional GPT evaluation.** We employ GPT-4o [29] to evaluate Visual Fidelity (VF), Image Alignment (IA), and Physical Plausibility (PP) on a 1–7 scale and directly provide an Overall Ranking (OR) across methods.

4.2. Comparisons

We construct a diverse test set containing 78 samples to evaluate the robustness and generalization capability of our method. The test set spans multiple table shapes—square, round, and triangular. For functional categories, we refer to the categorization used in MesaTask [25] and recommendations from ChatGPT, covering office tables, dining tables, workbenches, crafting tables, and a variety of other tabletop types ranging from realistic to stylized or fictional designs. By comparing TabletopGen with different baseline methods under identical inputs, we demonstrate its significant superiority in generating diverse tabletop scenes.

Quantitative Evaluation. Table 1 reports that our method, TabletopGen, achieves the best performance across all metrics in the three evaluation aspects, clearly outperforming all baselines. For **Visual & Perceptual quality**, TabletopGen attains the best LPIPS, DINOv2, and CLIP

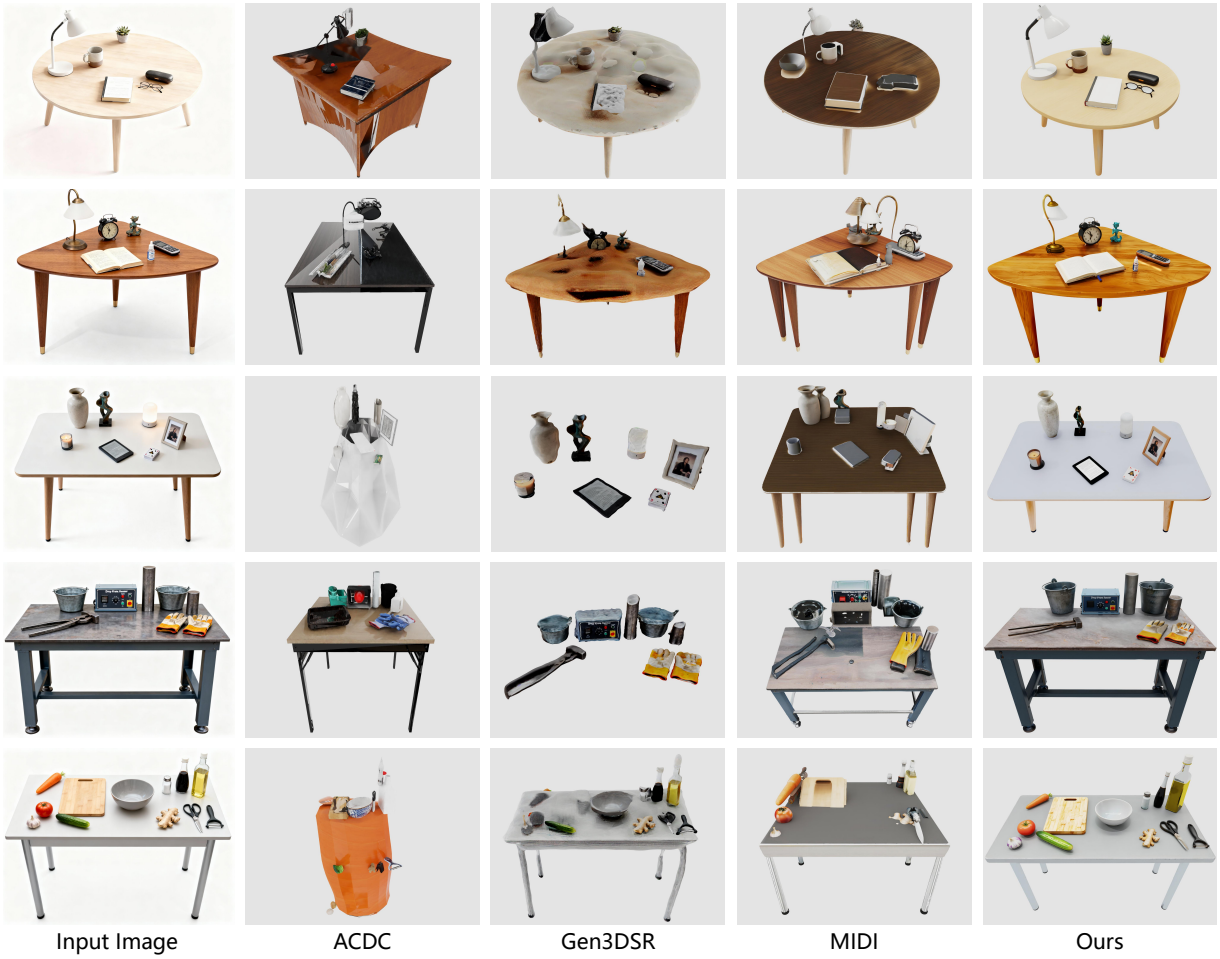


Figure 3. **Qualitative comparison under the same input images**. Our method, TabletopGen, consistently outperforms all baselines. TabletopGen demonstrates strong adaptability across diverse tabletop types, delivering more realistic appearances, finer instance models, more coherent object counts and layouts, and collision-free placement.

scores. We attribute this to our "instance-first, then alignment" design, which first ensures high-fidelity per-instance reconstruction and subsequently achieves precise pose and scale alignment. Under the **GPT Evaluation**, TabletopGen achieves the highest average score (6.19)—a 43% improvement over the second-best MIDI and the top Overall Ranking (OR). This demonstrates that our results are perceived as more realistic and semantically plausible. The **Collision Rate** shows the most significant performance gap. As previously analyzed, existing 3D scene generation methods struggle to capture the high-density tabletop layouts with complex spatial relations. Their Col_O typically lies around 8%–17%, and the scene-level Col_S reaches 67%–98%. In contrast, TabletopGen enables precise recovery of the layouts through its DRO and TSA stages. Our Col_O is near zero (0.42%), and Col_S remains only 7.69%. This overwhelming advantage demonstrates the robustness and practical value of our method for producing collision-free, interactive tabletop scenes.

Table 2. **User study results.** Our method TabletopGen attains the highest mean scores on Visual Fidelity (VF), Image Alignment (IA), and Physical Plausibility (PP), and the best overall preference (OP), being selected in 83.13% of cases.

Method	VF↑	IA↑	PP↑	Avg.↑	OP(%)
ACDC	3.00	2.46	2.87	2.78	2.38
Gen3DSR	2.95	3.25	3.19	3.13	2.88
MIDI	3.82	3.57	3.32	3.57	11.61
Ours	**5.62**	**5.50**	**5.56**	**5.56**	**83.13**

User Study. We further conducted a comprehensive user study to assess perceptual quality and human preference, involving 128 participants. Each participant was randomly assigned 8 scenes; to ensure fairness, the display order of results from different methods was fully randomized. Participants rated the generated scenes on three criteria—Visual Fidelity (VF), Image Alignment (IA), and Physical Plausi-

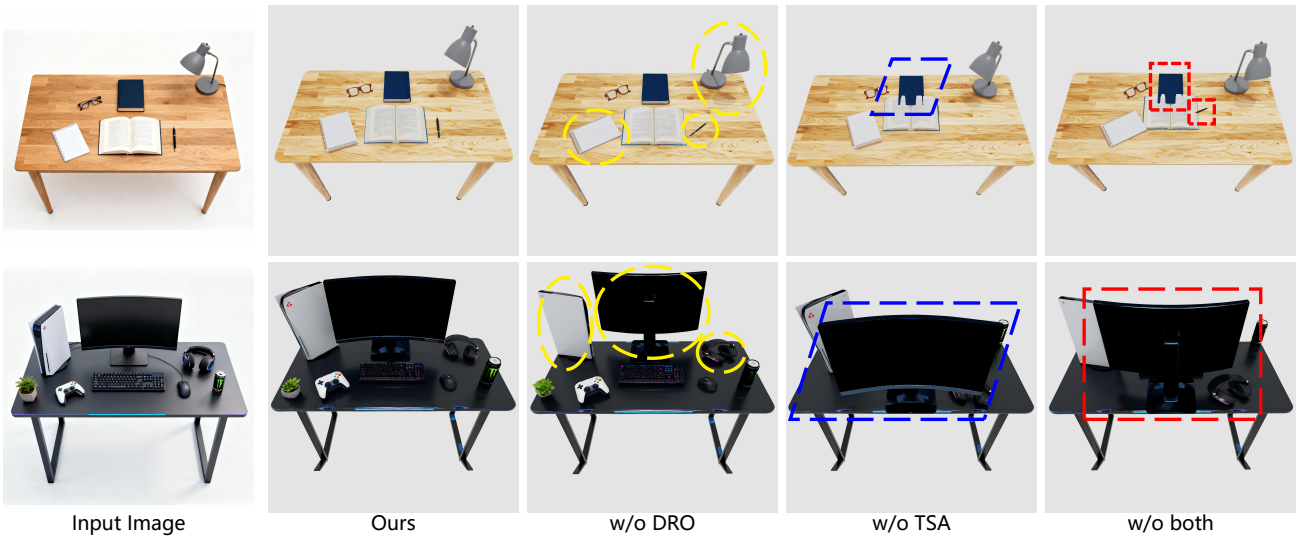


Figure 4. **Qualitative ablation study on pose and scale alignment components.** Compared to our full model (Ours), removing DRO yields incorrect instance rotations (yellow circles), removing TSA causes misplacements (blue parallelograms), and removing both amplifies these errors, often leading to severe occlusions and collisions (red rectangles).

bility (PP)—using a 7-point scale. As shown in Table 2, our method significantly outperforms all baselines on every criterion. Our average score of 5.56 was substantially higher than the second-best method (3.57). Most importantly, when asked for an Overall Preference (OP), our results were selected in 83.13% of cases, confirming that our generated scenes appear more realistic and are more persuasive to human evaluators.

Qualitative Evaluation. Fig. 3 compares our method with **single-image reconstruction** baselines under the same inputs. Thanks to our precise per-instance 3D reconstruction and the robust pose and scale alignment approach, TabletopGen produces consistent object counts and categories, finer geometry with unified style, semantically coherent layouts, and collision-free placement. In contrast, the retrieval-based method (e.g., ACDC) is limited by its fixed asset library, failing to match specific object styles and shapes. Generative reconstruction methods (e.g., Gen3DSR and MIDI) struggle with occlusions, leading to incomplete instance generation, object interpenetration, and layout inaccuracies. This finding aligns with the high collision rates reported in Table 1. Overall, TabletopGen delivers more realistic, interaction-ready tabletops with diverse types.

Furthermore, we demonstrate our **text-to-scene** capability by comparing TabletopGen against the recent tabletop generation method, MesaTask, in Fig. 5. MesaTask retrieves assets and places them on a fixed plane, limiting diversity and fidelity and occasionally causing minor collisions(e.g., the remote control and the can in the bottom row). In contrast, TabletopGen generates the entire scene, including the stylistically consistent table, delivering more realistic visual appearances, richer instance counts, more reasonable arrangements, and collision-free layouts.

Figure 5. **Qualitative comparison under the same input texts.** Compared with the recent tabletop generation method MesaTask, TabletopGen generates more complete and realistic scenes, including stylistically consistent tables, more detailed instance models, and more semantically and physically reasonable layouts.

4.3. Ablations

We ablate the two key stages, **DRO** and **TSA**, of the pose and scale alignment approach by replacing them with a naive MLLM-only baseline: when a stage is removed, we replace its specialized function by directly prompting Chat-

Table 3. **Quantitative ablation study on pose and scale alignment components.** Our full model (DRO+TSA) attains the best visual & perceptual scores and the lowest collision rates; removing either component degrades performance, and removing both leads to large increases in collisions.

Method	Visual & Perceptual			Collision Rate (%)	
	LPIPS↓	DINOv2↑	CLIP↑	Col_O↓	Col_S↓
Ours (full)	**0.4483**	**0.8383**	**0.9077**	**0.42**	**7.69**
w/o DRO	0.4523	0.8261	0.9012	1.27	16.67
w/o TSA	0.4799	0.8041	0.8954	5.50	61.54
w/o both	0.4811	0.7897	0.8922	5.41	62.82

GPT to estimate the parameters from the reference image. As reported in Table 3, removing either stage degrades visual & perceptual scores and sharply increases collisions, while removing both leads to the largest failure. Fig. 4 visually confirms these findings. The "w/o DRO" baseline fails to recover correct rotations, while the "w/o TSA" baseline produces implausible placements and scale drift. The "w/o both" compounds these errors, causing the final scene to suffer from severe collisions (e.g., the books interpenetrating) and occlusions (e.g., the monitor moving forward and blocking other objects). In contrast, our full model's result (Ours) closely matches the input image and presents a collision-free layout. This demonstrates that both DRO and TSA are essential for achieving a coherent and physically plausible scene reconstruction.

4.4. Extensions

3D Scenes from Real-World Images. As shown in Fig. 6, we compare TabletopGen with MIDI, the second-best method in our comparisons, on real-world photos. TabletopGen preserves instance counts and style, recovers accurate layouts, and yields collision-free, simulation-ready scenes. This real-to-sim path helps to bridge the sim-to-real gap for embodied manipulation.

Scene Editing. Thanks to per-instance generation and alignment, TabletopGen supports modular edits: objects can be swapped without reprocessing the whole scene. As shown in Fig. 7, we replace the ukulele with a banjo by generating its canonical model (Sec. 3.2) and inserting it using the estimated pose and scale, while the global layout remains unchanged.

We also demonstrate the applicability of our generated scenes for robotic manipulation tasks in the supplementary materials.

5. Conclusion

In this paper, we present **TabletopGen**, a training-free, fully automatic framework that turns text or a single image into instance-level interactive 3D tabletop scenes. By decoupling 2D-to-3D reasoning into a Differentiable Rotation Optimizer (DRO) for precise rotation recovery and a Top-view Spatial Alignment (TSA) mechanism for robust translation and metric scale inference, TabletopGen reliably reconstructs high-density and collision-free layouts. Extensive experiments and a large-scale user study demonstrate that TabletopGen achieves state-of-the-art performance with strong generalization over diverse tabletop

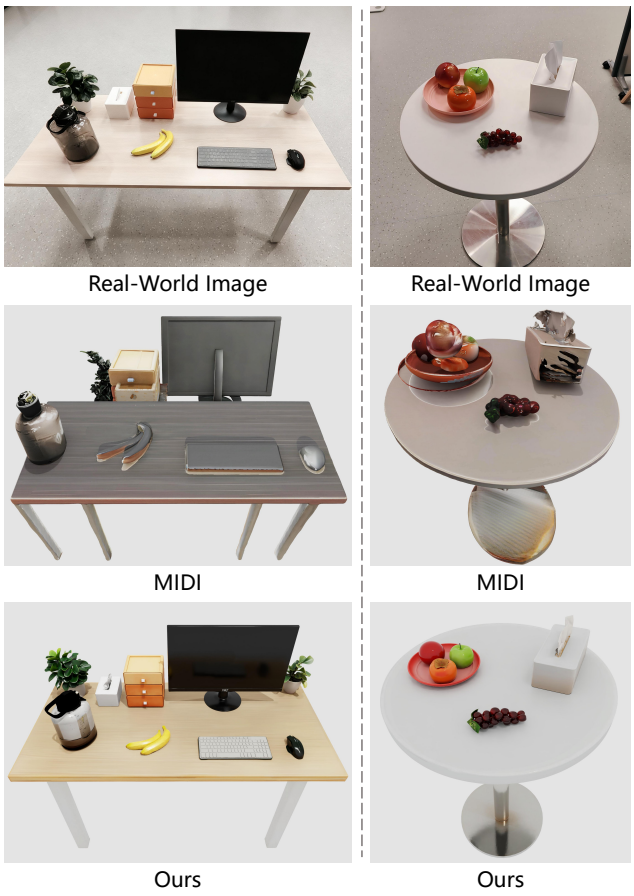


Figure 6. **3D Scenes from Real-World Images.** Compared with MIDI, our method robustly reconstructs real-world photos into 3D scenes with significantly higher geometric fidelity and a more plausible, collision-free layout.

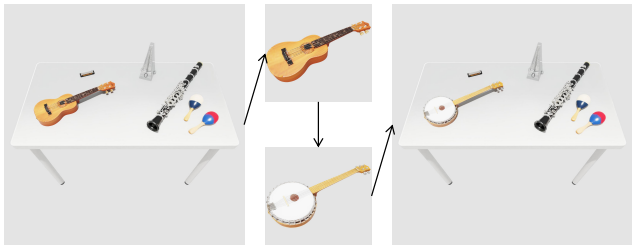


Figure 7. **Scene Editing.** Our compositional framework allows for individual instances, such as the ukulele, to be easily replaced with a new object, like the banjo, while retaining the original layout.

types. Moreover, TabletopGen shows excellent capabilities of real-world image reconstruction and scene editing.

Future work could generalize our framework to more complex support surfaces, such as multi-level shelves and high-density indoor floors, and generate finer and more structurally intricate assets. We believe that TabletopGen can provide more diverse tabletops for embodied AI simulation and robot policy learning.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 5

[2] Rio Aguina-Kang, Maxim Gumin, Do Heon Han, Stewart Morris, Seung Jean Yoo, Aditya Ganeshan, R Kenny Jones, Qiuhong Anna Wei, Kailiang Fu, and Daniel Ritchie. Open-universe indoor scene generation using llm program synthesis and uncurated object databases. *arXiv preprint arXiv:2403.09675*, 2024. 2, 3

[3] Andreea Ardelean, Mert Özer, and Bernhard Egger. Gen3dsr: Generalizable 3d scene reconstruction via divide and conquer from a single view. In *2025 International Conference on 3D Vision (3DV)*, pages 616–626. IEEE, 2025. 2, 3, 5

[4] Shuanghao Bai, Wenxuan Song, Jiayi Chen, Yuheng Ji, Zhide Zhong, Jin Yang, Han Zhao, Wanqi Zhou, Wei Zhao, Zhe Li, et al. Towards a unified understanding of robot manipulation: A comprehensive survey. *arXiv preprint arXiv:2510.10903*, 2025. 2

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, pages 1877–1901. Curran Associates, Inc., 2020. 2

[6] 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. 2, 3

[7] 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. 3

[8] 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. 2

[9] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 5

[10] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(9):10850–10869, 2023. 2

[11] Tianyuan Dai, Josiah Wong, Yunfan Jiang, Chen Wang, Cem Gokmen, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. Automated creation of digital cousins for robust policy learning. In *Conference on Robot Learning (CoRL)*, 2024. 2, 3, 5

[12] 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. In *Advances in Neural Information Processing Systems*, pages 5982–5994. Curran Associates, Inc., 2022. 2

[13] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects. In *Advances in Neural Information Processing Systems*, pages 35799–35813. Curran Associates, Inc., 2023. 2

[14] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13142–13153, 2023. 2

[15] Chuan Fang, Heng Li, Yixun Liang, Jia Zheng, Yongsen Mao, Yuan Liu, Rui Tang, Zihan Zhou, and Ping Tan. Spatialgen: Layout-guided 3d indoor scene generation. *arXiv preprint arXiv:2509.14981*, 2025. 3

[16] Weixi Feng, Wanrong Zhu, Tsu-Jui Fu, Varun Jampani, Arjun Akula, Xuehai He, S Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models. In *Advances in Neural Information Processing Systems*, pages 18225–18250. Curran Associates, Inc., 2023. 2

[17] 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. 3

[18] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3d-future: 3d furniture shape with texture. *International Journal of Computer Vision*, 129(12):3313–3337, 2021. 2, 3

[19] Rao Fu, Zehao Wen, Zichen Liu, and Srinath Sridhar. Anyhome: Open-vocabulary generation of structured and textured 3d homes. In *European Conference on Computer Vision*, pages 52–70. Springer, 2024. 2, 3

[20] Daoyi Gao, Dávid Rozenberszki, Stefan Leutenegger, and Angela Dai. Diffcad: Weakly-supervised probabilistic cad model retrieval and alignment from an rgb image. *ACM Transactions on Graphics (TOG)*, 43(4):1–15, 2024. 3

[21] Gege Gao, Weiyang Liu, Anpei Chen, Andreas Geiger, and Bernhard Schölkopf. Graphdreamer: Compositional 3d scene synthesis from scene graphs. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[22] Zeqi Gu, Yin Cui, Zhaoshuo Li, Fangyin Wei, Yunhao Ge, Jinwei Gu, Ming-Yu Liu, Abe Davis, and Yifan Ding. Artiscene: Language-driven artistic 3d scene generation through image intermediary. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2891–2901, 2025. 2, 3, 5

[23] Can Gümeli, Angela Dai, and Matthias Nießner. Roca: Robust cad model retrieval and alignment from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4022–4031, 2022. 3

[24] Haonan Han, Rui Yang, Huan Liao, Jiankai Xing, Zunnan Xu, Xiaoming Yu, Junwei Zha, Xiu Li, and Wanhua Li. Reparo: Compositional 3d assets generation with differentiable 3d layout alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 25367–25377, 2025. 3, 5

[25] Jinkun Hao, Naifu Liang, Zhen Luo, Xudong Xu, Weipeng Zhong, Ran Yi, Yichen Jin, Zhaoyang Lyu, Feng Zheng, Lizhuang Ma, et al. Mesatask: Towards task-driven tabletop scene generation via 3d spatial reasoning. *arXiv preprint arXiv:2509.22281*, 2025. 2, 3, 5

[26] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851. Curran Associates, Inc., 2020. 2

[27] Yujia Hu, Songhua Liu, Xingyi Yang, and Xinchao Wang. Flash sculptor: Modular 3d worlds from objects. *arXiv preprint arXiv:2504.06178*, 2025. 3, 5

[28] 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 Computer Vision and Pattern Recognition Conference*, pages 23646–23657, 2025. 2, 3, 5

[29] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. 5

[30] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[31] Zeqiang Lai, Yunfei Zhao, Haolin Liu, Zibo Zhao, Qingxiang Lin, Huiwen Shi, Xianghui Yang, Mingxin Yang, Shuhui Yang, Yifei Feng, et al. Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details. *arXiv preprint arXiv:2506.16504*, 2025. 5

[32] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pages 80–93. PMLR, 2023. 2

[33] Haoran Li, Haolin Shi, Wenli Zhang, Wenjun Wu, Yong Liao, Lin Wang, Lik-hang Lee, and Peng Yuan Zhou. Dreamscene: 3d gaussian-based text-to-3d scene generation via formation pattern sampling. In *European Conference on Computer Vision*, pages 214–230. Springer, 2024. 2, 3

[34] Lu Ling, Chen-Hsuan Lin, Tsung-Yi Lin, Yifan Ding, Yu Zeng, Yichen Sheng, Yunhao Ge, Ming-Yu Liu, Aniket Bera, and Zhaoshuo Li. Scenethesis: A language and vision agentic framework for 3d scene generation. *arXiv preprint arXiv:2505.02836*, 2025. 3, 5

[35] Andrew Melnik, Benjamin Alt, Giang Nguyen, Artur Wilkowski, Maciej Stefańczyk, Qirui Wu, Sinan Harms, Helge Rhodin, Manolis Savva, and Michael Beetz. Digital twin generation from visual data: A survey. *arXiv preprint arXiv:2504.13159*, 2025. 2

[36] Yanxu Meng, Haoning Wu, Ya Zhang, and Weidi Xie. Scenegen: Single-image 3d scene generation in one feedforward pass. *arXiv preprint arXiv:2508.15769*, 2025. 2, 3, 5

[37] NVIDIA. Isaac Sim 4.5.0, 2025. 5

[38] Başak Melis Öcal, Maxim Tatarchenko, Sezer Karaoğlu, and Theo Gevers. Sceneteller: Language-to-3d scene generation. In *European Conference on Computer Vision*, pages 362–378. Springer, 2024. 2

[39] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 4, 5

[40] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. In *Advances in Neural Information Processing Systems*, pages 12013–12026. Curran Associates, Inc., 2021. 2

[41] Akshay Gadi Patil, Supriya Gadi Patil, Manyi Li, Matthew Fisher, Manolis Savva, and Hao Zhang. Advances in data-driven analysis and synthesis of 3d indoor scenes. In *Computer Graphics Forum*, page e14927. Wiley Online Library, 2024. 2

[42] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 5

[43] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020. 5

[44] Pengzhen Ren, Min Li, Zhen Luo, Xinshuai Song, Ziwei Chen, Weijia Liufu, Yixuan Yang, Hao Zheng, Rongtao Xu, Zitong Huang, et al. Infiniteworld: A unified scalable simulation framework for general visual-language robot interaction. *arXiv preprint arXiv:2412.05789*, 2024. 2

[45] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng

Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024. 4

[46] ByteDance Seed, Jiaze Chen, Tiantian Fan, Xin Liu, Lingjun Liu, Zhiqi Lin, Mingxuan Wang, Chengyi Wang, Xiangpeng Wei, Wenyuan Xu, et al. Seed1. 5-thinking: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*, 2025. 5

[47] Team Seedream, Yunpeng Chen, Yu Gao, Lixue Gong, Meng Guo, Qiushan Guo, Zhiyao Guo, Xiaoxia Hou, Weilin Huang, Yixuan Huang, et al. Seedream 4.0: Toward next-generation multimodal image generation. *arXiv preprint arXiv:2509.20427*, 2025. 5

[48] Tanya Short and Tarn Adams. *Procedural generation in game design*. CRC Press, 2017. 2

[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. 2, 3

[50] Marcel Torne, Anthony Simeonov, Zechu Li, April Chan, Tao Chen, Abhishek Gupta, and Pulkit Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. *arXiv preprint arXiv:2403.03949*, 2024. 2

[51] 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. 2

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

[53] Beichen Wen, Haozhe Xie, Zhaoxi Chen, Fangzhou Hong, and Ziwei Liu. 3d scene generation: A survey. *arXiv preprint arXiv:2505.05474*, 2025. 2

[54] Mutian Xu, Pei Chen, Haolin Liu, and Xiaoguang Han. Toscene: A large-scale dataset for understanding 3d tabletop scenes. In *European conference on computer vision*, pages 340–356. Springer, 2022. 3

[55] Haitao Yang, Zaiwei Zhang, Siming Yan, Haibin Huang, Chongyang Ma, Yi Zheng, Chandrajit Bajaj, and Qixing Huang. Scene synthesis via uncertainty-driven attribute synchronization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5630–5640, 2021. 2

[56] 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. 2

[57] 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. 2

[58] 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

[59] Georgios N Yannakakis and Julian Togelius. *Artificial intelligence and games*. Springer, 2018. 2

[60] Kaixin Yao, Longwen Zhang, Xinhao Yan, Yan Zeng, Qixuan Zhang, Lan Xu, Wei Yang, Jiayuan Gu, and Jingyi Yu. Cast: Component-aligned 3d scene reconstruction from an rgb image. *ACM Transactions on Graphics (TOG)*, 44(4): 1–19, 2025. 3, 5

[61] Huangyue Yu, Baoxiong Jia, Yixin Chen, Yandan Yang, Puhao Li, Rongpeng Su, Jiaxin Li, Qing Li, Wei Liang, Song-Chun Zhu, et al. Metascenes: Towards automated replica creation for real-world 3d scans. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1667–1679, 2025. 2

[62] Hong-Xing Yu, Haoyi Duan, Charles Herrmann, William T Freeman, and Jiajun Wu. Wonderworld: Interactive 3d scene generation from a single image. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5916–5926, 2025. 3

[63] Yiming Zeng, Mingdong Wu, Long Yang, Jiyao Zhang, Hao Ding, Hui Cheng, and Hao Dong. Lvdiffusor: Distilling functional rearrangement priors from large models into diffusor. *IEEE Robotics and Automation Letters*, 2024. 3

[64] Guangyao Zhai, Evin Pı nar Örnek, Shun-Cheng Wu, Yan Di, Federico Tombari, Nassir Navab, and Benjamin Busam. Commonscenes: Generating commonsense 3d indoor scenes with scene graph diffusion. In *Advances in Neural Information Processing Systems*, pages 30026–30038. Curran Associates, Inc., 2023. 3

[65] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 5

[66] Shengqiang Zhang, Philipp Wicke, Lütfi Kerem Şenel, Luis Figueredo, Abdeldjallil Naceri, Sami Haddadin, Barbara Plank, and Hinrich Schütze. Lohoravens: A long-horizon language-conditioned benchmark for robotic tabletop manipulation. *arXiv preprint arXiv:2310.12020*, 2023. 2

[67] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2), 2023. 2

[68] 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. 3

[69] Junwei Zhou, Xueting Li, Lu Qi, and Ming-Hsuan Yang. Layout-your-3d: Controllable and precise 3d generation with 2d blueprint. *arXiv preprint arXiv:2410.15391*, 2024. 3

Supplementary Material

6. Details of TabletopGen

6.1. Stage-wise Intermediate Outputs

TabletopGen is a multi-stage 3D scene generation framework that decomposes the complex task of converting text or a single image into a 3D scene into several more manageable sub-stages, including instance extraction, canonical 3D model generation, pose and scale alignment, and final 3D scene assembly. To enhance transparency and reproducibility, we present stage-wise key intermediate results from the inputs to the final outputs.

We use a text input as an illustrative example. As shown in Fig. 8, for a short text input by the user (e.g., "A hobby desk with some model cars and tools."), we first use ChatGPT to expand it into a detailed scene description. This description is then passed into the image generation model Seedream to synthesize a visually realistic and plausibly arranged reference image I_{ref}, on which all subsequent generation stages are processed.

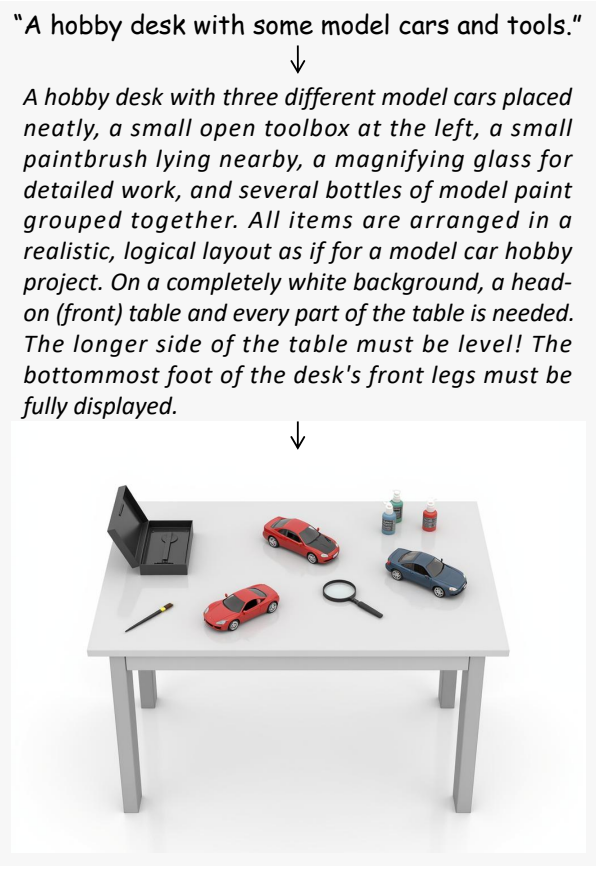


Figure 8. **Text-input preprocessing pipeline.** For a short text (top), we first use an LLM to expand it into a detailed, layout-aware scene description (middle). The expanded description is then fed into a text-to-image model to synthesize the reference image (bottom), which conditions all subsequent stages.

Instance Extraction. Beginning with I_{ref}, ChatGPT first analyzes the image to identify all object categories L_i (e.g., "table", "toy car", "magnifying glass"). These categories then guide GroundedSAM-v2 to generate instance segmentation masks M_i. However, as shown in Fig. 9, the masks often contain holes and blurry boundaries. To address this, we use Seedream for per-instance generative completion, producing high-resolution, clean instance images.

Canonical 3D Model Generation. From the completed instance images, we reconstruct high-quality 3D models using an image-to-3D model, Hunyuan3D-3.0. Subsequently, each initial model undergoes canonical coordinate alignment. We utilize ChatGPT to analyze whether its z-axis represents the correct upright orientation. If it does not, a corrective rotation is applied to align the model's local z-axis with the world coordinate system. Fig. 10 shows the output of this stage: all instance renderings in the canonical coordinates.

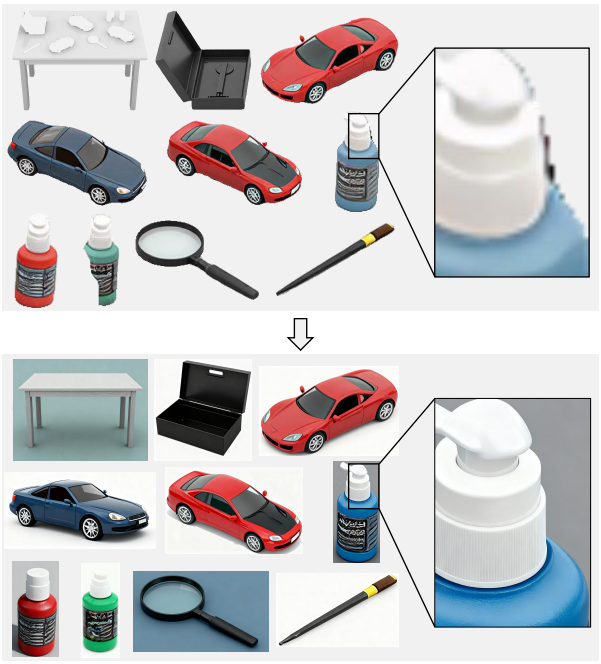


Figure 9. **Completion of segmentation masks.** We apply generative completion to refine the raw segmentation outputs: instance masks with holes and blurry boundaries (top) are redrawn into complete, high-resolution, clean-edged instance images (bottom).

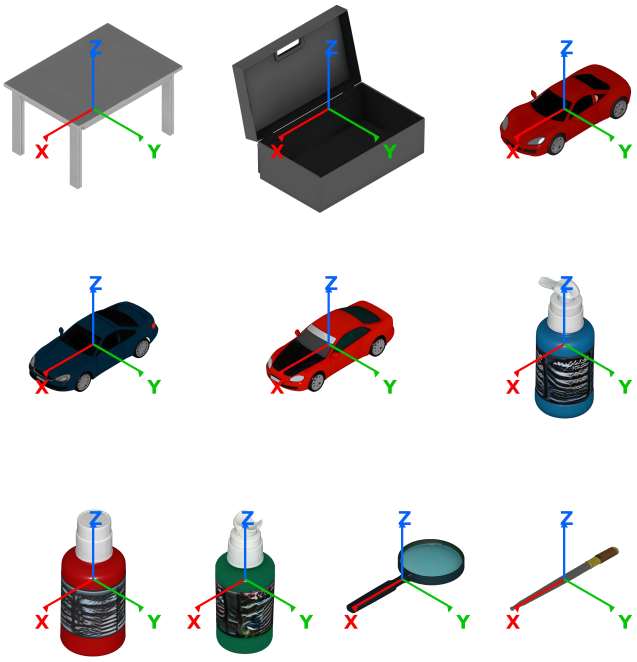


Figure 10. **Canonicalized instance 3D models.** After image-to-3D and coordinate alignment, each object's local vertical axis (+Z) is aligned with the tabletop world up, enabling direct placement.

Differentiable Rotation Optimizer. After obtaining all 3D instances, we proceed to the Pose and Scale Alignment stage. This stage begins with the Differentiable Rotation Optimizer (DRO), which estimates the precise rotation r_i for each instance in the scene. As described in Sec. 3.3, ChatGPT first predicts an initial camera perspective (azimuth, elevation) from the reference image I_{ref}. For our "hobby desk" example, this is (azimuth = 0°, elevation = 60°). Given this perspective, we differentiably render each instance and optimize its rotation r_i by minimizing the tri-modal loss $\mathcal{L}_{rot}$.

For supervision, we use the original instance crop with its segmentation mask, together with its silhouette and edge maps as reference images; all references are tightly cropped and resized to 256×256. The rendered projections are cropped with the same tight bounds and resized to the same resolution to ensure loss consistency.

Fig. 11 shows the final match for instance *toy car_1*, where the rendered silhouette, edges, and appearance maps closely agree with the corresponding reference images. Fig. 12 shows the final JSON output of the rotation estimation.

Top-View Spatial Alignment. The Top-View Spatial Alignment (TSA) estimates per-instance translation (t_i) and scale (s_i). We first synthesize a top-view image of the scene with Seedream and detect 2D bounding boxes for all instances in that view, as shown in Fig. 13. In parallel, ChatGPT provides commonsense physical sizes for each object. Subsequently, we identify the scene's anchor object (excluding the table) via the RMA-Score; in this example, it is *box_9*. Using this anchor, we rescale objects whose commonsense size deviates significantly from the bounding box size. We found experimentally that typically only the table requires scaling, as the MLLM tends to overestimate its size.

After scaling, we define a coordinate system with the table's center as the origin, the horizontal-left direction as the positive x-axis, and the forward direction as the positive y-axis. The xy-translation for each instance is then calculated

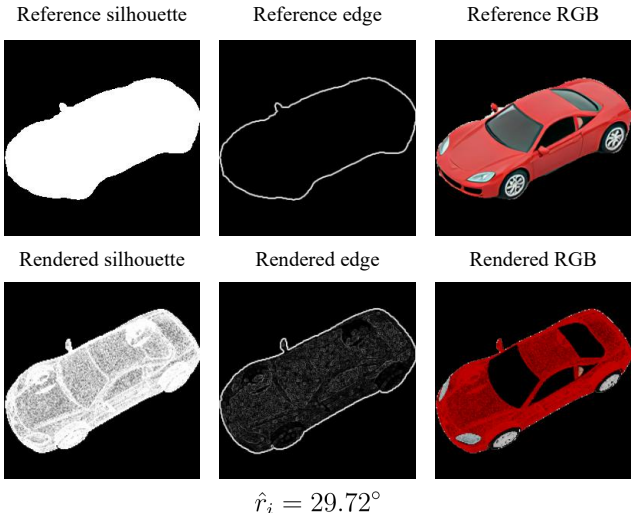


Figure 11. **Results of the DRO for a "toy car" instance**. The top row displays the reference silhouette, edge, and RGB maps. The bottom row shows the corresponding outputs from the differentiable renderer at the final estimated rotation ($\hat{r}_i = 29.72°$). The high degree of consistency across all three modalities validates the effectiveness of our tri-modal loss in accurately recovering the instance's rotation.


```
{
    "table_0": 0.00,
    "toy car_1": 29.72,
    "bottle_2": 295.00,
    "magnifying glass_3": 145.00,
    "bottle_4": 0.00,
    "toy car_5": 159.99,
    "bottle_6": 180.00,
    "toy car_7": 155.00,
    "screwdriver_8": 35.04,
    "box_9": 85.08
}
```


Figure 12. **DRO outputs (JSON).** Estimated per-instance rotation r_i (Euler *yaw*) about the scene z-axis, reported in degrees (°).

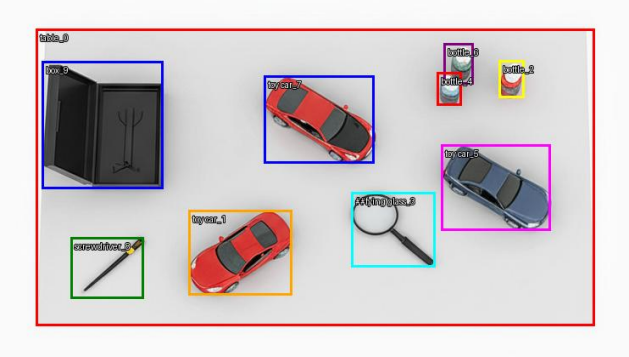


Figure 13. **Top-view synthesis and bounding box detection.** A top-view of the tabletop is synthesized with a multimodal image generation model, and 2D bounding boxes for all instances are detected on it. These boxes are used to compute the RMA-Score for anchor selection and to derive translation and scale for TSA.


```
{
    "anchor_object": "box_9",
    "objects": {
        "table_0": {
            "scale": [104.55, 58.44, 64],
            "translation": [0.0, 0.0, 32.0]
        },
        "toy car_1": {
            "scale": [18.00,  8.00,  6.00],
            "translation": [13.23, 14.45, 67.00]
        },
        // ... remaining instances omitted for brevity
        "box_9": {
            "scale": [25.00, 20.00, 20.00],
            "translation": [39.94, -9.58, 74.0]
        }
    }
}
```


Figure 14. **TSA outputs (JSON).** Estimated per-instance scale and translation, reported in centimeters (cm). Scales are listed as $[x, y, z]$ and translations as $[x, y, z]$ in the tabletop frame (origin at the table center; $+x$ left, $+y$ forward, $+z$ up). The anchor object selected by RMA-Score is shown at the top.

using the center point of its bounding box. To handle stacking situations (e.g., a pen on a book), we also use ChatGPT to analyze the stacking order of instances in the scene to determine the z-translation. Fig. 14 shows the final JSON output of the translation and scale estimation.

3D Scene Assembly. Finally, we place all canonical models using the estimated (r_i, t_i, s_i) and export a 3D scene. When loading into NVIDIA Isaac Sim, we set the objects as rigid-dynamic bodies with gravity enabled. For collisions, we generate per-object shapes via convex decomposition and attach them to the meshes. This yields an interaction-ready scene with physically plausible contacts that can be used directly for simulation and manipulation.

6.2. Efficiency Analysis

We evaluate the efficiency of our framework on a single NVIDIA GeForce RTX 4090 GPU (24GB VRAM). Taking a representative scene containing 9 object instances as an example, the entire process—from the input reference image to obtaining all canonical 3D models and their corresponding spatial parameters (r_i, t_i, s_i) for assembly—takes approximately 32 minutes. The time breakdown for each stage is as follows:

- Instance Extraction: Takes $\approx$ 240s in total. This includes $\approx$ 30s for scene detection and segmentation, and $\approx$ 23s per instance for multimodal generative completion.
- Canonical 3D Model Generation: Takes $\approx$ 830s in total. For the image-to-3D generation step, we implemented a parallelism of 3 (processing 3 instances concurrently), taking $\approx$ 180s per batch. The canonical coordinate alignment takes $\approx$ 32s per instance.
- Pose and Scale Alignment: The DRO stage consumes $\approx$ 810s ($\approx$ 90s per instance), while the TSA stage takes $\approx$ 50s in total.

It is worth noting that in our current experimental setup, with the exception of the image-to-3D generation step, all other instance-level processes are executed sequentially. Given that the processing of each instance is independent, the total inference time has the potential to be significantly reduced by implementing full parallel processing in future optimizations.

7. Robotic Manipulation Demonstrations

We utilize the NVIDIA Isaac Sim 4.5.0 environment to place the generated tabletops into an indoor scene, and introduce a Franka Emika Panda 7-DoF arm with a parallel gripper into the scene. This widely-used, standard industrial arm is tasked with executing a series of complex pick-and-place maneuvers to demonstrate the physical viability and suitability of our scenes for embodied manipulation tasks.

We selected two common tabletop environments, a kitchen table and a coffee table, for demonstration:

- Kitchen Task: The arm was tasked with executing a multi-step food preparation sequence: placing the cucumber into the bowl, and subsequently moving the carrot and salt shaker onto the cutting board.
- Coffee Table Task: The task required the arm to swap the positions of the cup and the apple.

The accompanying supplementary video, *"ManipulationTaskDemo.mp4"*, illustrates the successful execution of both tasks without spurious penetrations or unstable contacts. This demonstrates that TabletopGen produces interaction-ready tabletop scenes whose instance models

Input Image | Best-view of MIDI | Best-view of Ours

(a) **Best views selected by LPIPS↓.** For each method, we render 160 views and pick the one with the lowest LPIPS↓ to the input image.

MIDI	(70°, 1.5×) 0.6414	(50°, 1.5×) 0.6191	(30°, 1.5×) 0.5498	**(20°, 1.8×) 0.4489**	(20°, 2.5×) 0.5495
Ours	(70°, 1.5×) 0.6125	(50°, 1.5×) 0.5324	**(30°, 1.5×) 0.3573**	(20°, 1.8×) 0.5030	(20°, 2.5×) 0.5876

(b) **LPIPS across different views.** Each image is annotated by (elevation°, distance-multiplier) and its LPIPS↓. **Bold** denotes the best LPIPS↓ / best view for that method. Red dashed rectangles highlight the visual bias where visually plausible shots can hide physically invalid layouts (e.g., floating objects).

Figure 15. **LPIPS-based view selection under the camera sweep protocol.** We sweep the elevation from 90° (top) to 0° (front) in 10° steps and vary the camera distance from 1.0× to 2.5× the scene radius (160 renders per scene). While this protocol facilitates a relatively fair and meaningful comparison, the visual bias inherent in a single view allows physically implausible phenomena to be masked under the "best view," leading to overestimated scores. Conversely, our scenes maintain structural integrity and a plausible layout across all perspectives. This indicates that the actual gap between our method and baselines may be larger than the metric score suggests.

and physical properties are directly usable for embodied AI environments and robot manipulation policy learning.

8. Details of Experiments

8.1. Visual & Perceptual Quality Details

Camera Sweep Protocol. Due to the unknown camera perspective of arbitrary input images, we established a standardized camera sweep trajectory for all methods to ensure a fair and meaningful comparison. We generate a total of 160 distinct views per scene.

The sweep covers the vertical range from a top-down view (90° elevation) to a front view (0° elevation) in 10° steps. Concurrently, the camera distance from the scene center is varied in 16 incremental steps, ranging from 1.0× to 2.5× the scene radius. For each generated rendering, we calculate the LPIPS/DINOv2/CLIP scores against the original input image. The reported score of each method on each metric is the best view (the view most closely matching the input image), and the final metric scores are averaged over all scenes.

Discussion on Fairness. However, this "best-view selection" process tends to slightly favor certain baselines. As shown in Fig. 15b, when MIDI generates floating objects or models exceeding the desktop boundary, its best perspec-

tive ($20°, 1.8\times$) introduces a visual bias that masks these implausible layouts, thus allowing it to still obtain comparatively good LPIPS/DINOv2/CLIP scores. Conversely, our scenes are guaranteed to remain collision-free and physically grounded at any perspective, ensuring they are unaffected by this visual bias. Consequently, the true visual and perceptual advantage of TabletopGen over baselines is likely larger than what the reported scores alone suggest.

8.2. User Study Details

We conducted a comprehensive human evaluation to compare TabletopGen with baseline methods on tabletop scene quality. In total, 128 unpaid volunteers participated; responses were anonymous and used solely for research.

As shown in Fig. 17, the survey began with a brief description of goals and instructions. Each participant was then assigned 8 scenes at random. For each scene, the interface presented the input image and the four generated results (one per method) in a randomized order to mitigate position bias. To accommodate different audiences, every prompt and criterion was provided in both English and Chinese.

For each method, participants rated three criteria on a 1–7 scale (1=very poor, 4=average, 7=excellent) (Fig. 18):

- **Visual Fidelity (VF):** overall visual quality and realism of the scene;
- **Image Alignment (IA):** consistency with the input image in style, counts, categories, and layout;
- **Physical Plausibility (PP):** physical reasonableness (no floating/penetration, stable support, etc.).

After scoring the four results, participants selected an Overall Preference (OP) among the four images for that scene (Fig. 19).

The mean completion time was 716 seconds. We discarded submissions less than 200 seconds, yielding 126 valid questionnaires. We report per-method means for VF/IA/PP and the OP selection counts and percentages.

9. More Results

9.1. Stylized Tabletop Generation

Since TabletopGen does not depend on a fixed asset library and generates each instance from the reference image, it is naturally style-agnostic. Beyond photorealistic scenes, we can generate 3D tabletop scenes from stylized inputs. Fig. 16 shows a hand-drawn, cartoon-style office table: TabletopGen reconstructs a unified 3D scene that preserves the object semantics and layout while maintaining the input style. This diversity highlights TabletopGen's applicability to VR/AR prototyping and game content creation.

Input Image

TabletopGen Result

Figure 16. **Stylized tabletop generation from a cartoon input.** TabletopGen can transfer a cartoon-style 2D input into a 3D scene. The generated scene maintains a consistent style and a plausible layout.

9.2. Visualizations and Interaction Demos

To provide a more detailed and intuitive demonstration of the visual fidelity and interactive capability of our generated scenes, the supplementary video, "*SceneShowcase.mp4*", presents a 360° orbital demonstration of multiple scenes. The video showcases the scenes' structural integrity via white model visualization (geometry) and their final full texture rendering. Furthermore, we actively demonstrate the instance-level interactive property by executing movement on individual objects within each scene. This confirms the physical viability and independence of every generated scene, highlighting the application potential of TabletopGen in physics-enabled simulation environments.

10. Prompts

The complete prompt templates used across all stages of TabletopGen are detailed in Figs. 20-25.

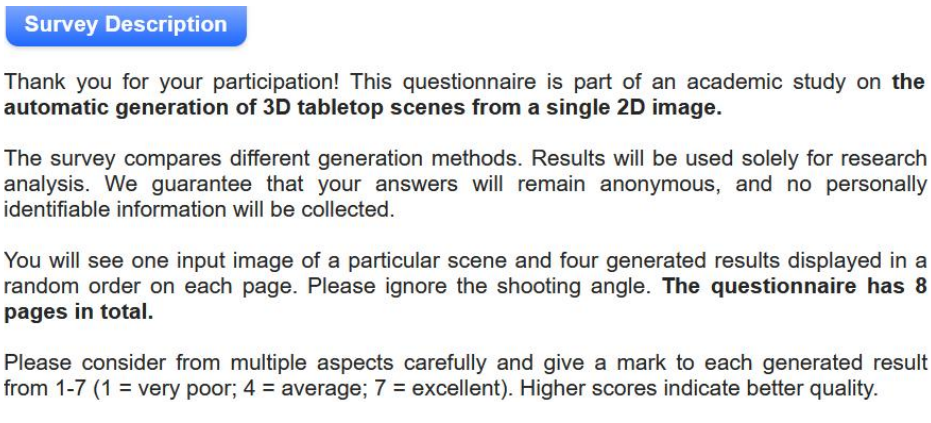


Figure 17. **Survey description.** Overview of the study goals and instructions.

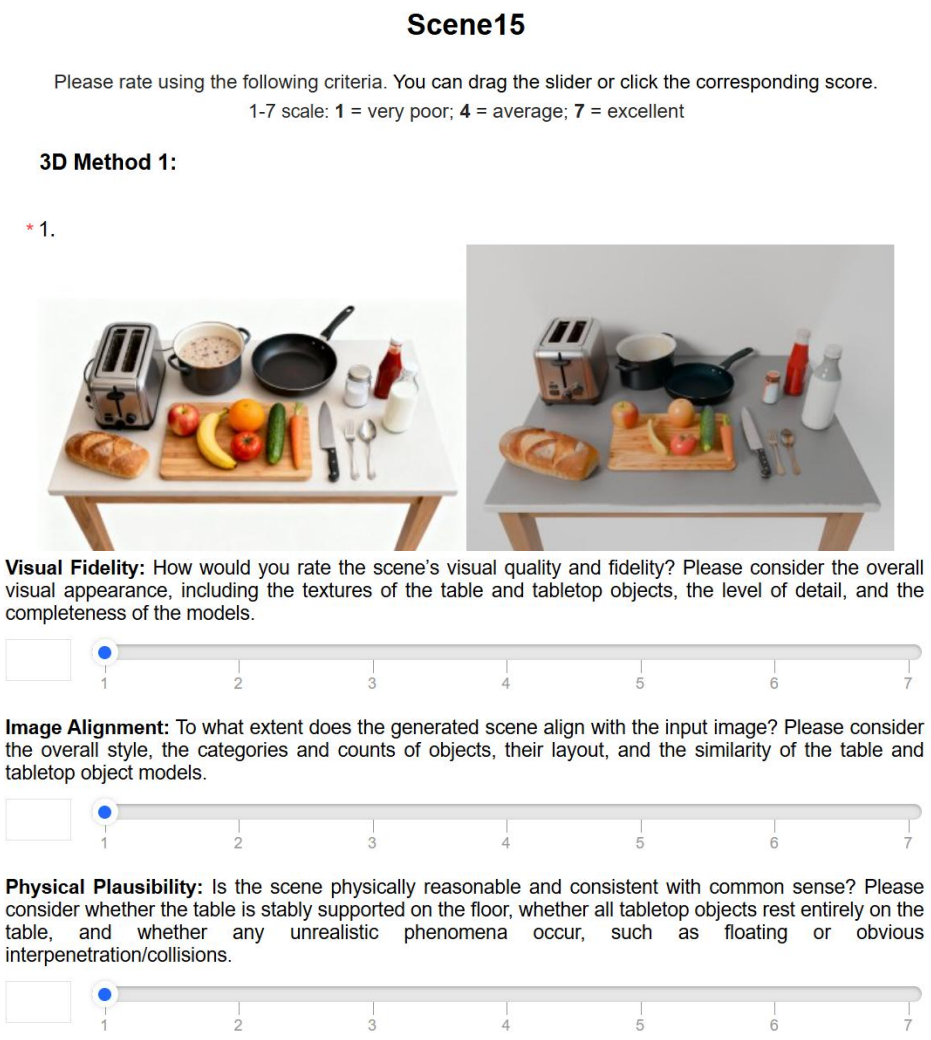
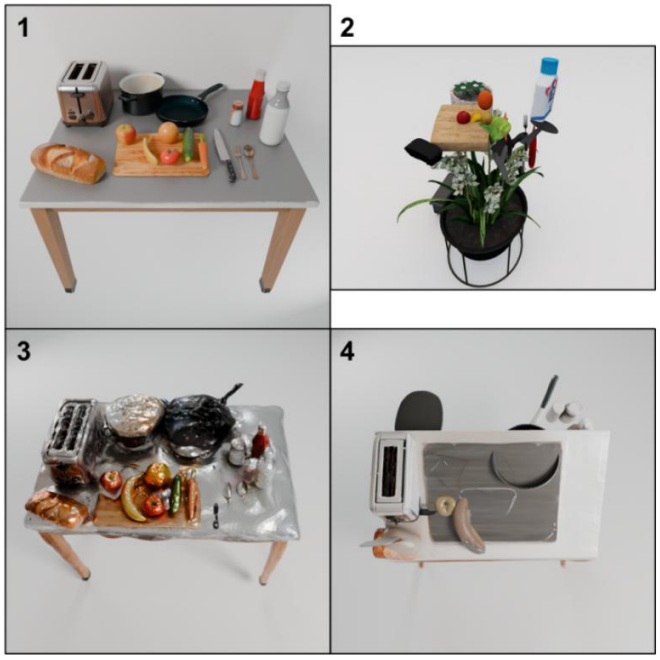


Figure 18. **Per-scene rating page.** For each scene, participants rate Visual Fidelity, Image Alignment, and Physical Plausibility (1–7) given the input image and one method's result. Shown here is the scoring UI for a single method, the other three methods use the same interface.



* 5. **Overall Preference:** Overall, which scene do you prefer?

1 2 3 4

Figure 19. **Overall preference.** After scoring four methods, participants choose a single overall favorite among the four randomized results.

```
You are a 3D scene layout prompt designer. Your task is to Infer the possible items based on the scene
description and layout the items.

**Rules**:
- First of all, a complete and head-on(front) table is needed. Every parts of the table must be very easy to be
seen, especially any of the table foot!
- The angle more than 50 degrees and less than 60 degrees above the table.
- "completely white background", " a head-on(front) table and every parts of the table is needed" and "The
longer side of the table must be level! The bottommost foot of the desk's front legs must be fully displayed"
must be added in end of the prompt.
- Make the layout of objects more natural, ensure semantic rationality, and conform to real-world logic. When
two or more items are semantically related, arrange them according to real-world logic. For example, place the
pen to the right of the notebook, the mouse to the right of the keyboard, and the laptop in front of the
keyboard. Avoid lining up all objects side by side.
- Make sure that all possible objects are on the table.And all items on the table can be clearly seen.
- Keep in mind that the full view of the table and all items can be seen completely.
- Make sure there are only table and all objects on the table in the picture, without chairs etc.
- Avoid items that are too thin, such as a piece of paper. Avoid the same thing. Avoid a single fragile label
sticker.
- If no specific items are specified, the number of items should not exceed seven.

Only output a short final prompt.

The scene description is :"{text_input}".
```

Figure 20. Prompt for expanding a short text input into a detailed scene description.

```
Please redraw the "{object_name}" object in the first reference image and generate a high-definition image of
it. Its proportions, shape, texture, and other appearance features must remain unchanged. If it is obscured or
overlapped by other objects(usually the transparent mask in the first reference image is the obstructing
object), remove the obstructing objects and draw only the "{object_name}" itself. Fill the background with a
solid color.
```

Figure 21. Prompt for multimodal generative completion.

```
You are an image analysis expert. Please analyze the size of each object in the picture and give the
corresponding coordinate axis direction at the same time. The specific tasks are:

1. **Object size estimation**
    - Match each object with the given list of objects and Figure 2. Each bounding box in Figure 2 corresponds
to a single object (Note: Don't be misled by the plural names in the object list. Each serial number corresponds
to only one item.).
    - Estimate the size of its 3D bounding box according to the display state of the object: [length, width,
height] (unit: cm).
    - The size must strictly correspond to the state of the diagram. For example, if there is an open laptop in
the diagram, the height should be the height after opening rather than the thickness in the closed state.

2. **Axis Definition & Placement State**
    - Give the direction description and placement state of the three axes of the object [x, y, z] (i.e.
[length, width, height]). For example, a book has two placement states: 'lay flat' and 'stand up'. In the lay
flat state, x is the length of the book when it is laid flat, y is the width of the book when it is laid flat,
and z is the thickness when it is laid flat.
    - Do not include any other reference objects in the direction description. It is required that the
coordinate system can be aligned only by the object itself.
    - The xy plane of all objects should be parallel to the ground/table, and the z axis should be the
height/thickness of the displayed position.

3. **Realistic Size Reference**
    Base size estimates on common real-world proportions (e.g., smartphone ≈ 15cm long, coffee cup ≈ 10cm tall,
open laptop screen height ≈ 20cm).

Output in JSON format (no additional explanations). Include the object name (from the object list), size, and
axis description.

**Output example**
{{ "book_1": {{"size":[20,10,2], "axis": {{ "placement":"lay flat", "x":"length of the book when it is lying
flat", "y":"width of the book when it is lying flat", "z":"thickness of the book when it is lying flat"}} }},
"object_2": ... }}

The object list in the figure is: {object_list}.
```

Figure 22. Prompt for analyzing the commonsense size and canonical axis definition of each instance.

```
You are a 3D model orientation reviewer, responsible for comparing the rendered model with the given description
of the actual placement direction of the object, judging and correcting the model coordinate system orientation.

Input:
- A top view of the rendered 3D model, with the local X (red), Y (green), and Z (blue) axes superimposed.
- A description of the actual placement of the object, describing the placement state and xyz orientation of the
object, and can also be used as a reference with size.

Please follow the steps below to determine whether the object needs to be rotated:
1. **Z-axis direction**: Determine whether the placement state of the rendered model matches the description. If
not, rotate along the x or y axis. For example, a book is described as lying flat, but it is standing up in the
rendered model (the z axis points to the height of the book rather than the thickness), so it needs to be
rotated.
2. **X and Y direction**: Determine whether the direction of the x and y axes in the rendered model is
consistent with the length/width of the description. If not, rotate along the z axis. You need to refer to both
the x and y sizes for analysis, and **must** ensure that the rotation is correct.
3. If the rotation was performed in the first step, the second step needs to determine the direction of the new
x and y axes after the rotation. Therefore, there may be two rotations.
4. Output ["rotation axis", angle], where the angle is a multiple of ±90.

Output example (strict list):
["x", 90,"z",-90]
If no rotation is required, output: []

The object description list is: {object_info}.
```

Figure 23. Prompt for canonicalizing the 3D model coordinate system of a single instance.

```
Generate **a perfectly orthographic top-down shot of the table**, as if taken by a camera directly above at 90
degrees with no perspective distortion.

The tabletop should be a flat rectangle fully obscuring its legs.

**Keep all items exactly in their current positions and orientations. Do not add new objects. Do not change the
aspect ratio of the table. **

The background around the table should be a solid neutral color.
```

Figure 24. Prompt for synthesizing the top-view image.

```
You are a layout designer. Given a list of objects, please output the placement order of each object in the
image. The main object in the scene (such as a table) is ranked 1. Objects placed on top of it are determined by
their stacking position. Generally, larger objects placed on top of the main object are placed first, with
smaller objects placed later. Also, if object A is partially or completely placed above object B, A should be
placed after B, and object B needs to be added to the output of object A.

Please analyze the image carefully and determine the logical placement order based on:
1. The main object (table) should be placed first (order 1)
2. Larger objects on the table should be placed before smaller objects
3. If one object is on top of another, the top object should have a higher order number
4. Consider the natural stacking and layering in the scene
5. **Analyze object overlap in scenes without bounding boxes**, and then map bounding box numbers to object
names. Don't judge overlap directly from bounding boxes, as the bounding boxes will be larger and prone to false
overlap.

Please output the placement order dictionary in JSON format, for example:
{{
    "pen_0": {{
        "order": 5,
        "on_top_of": "book_1"
    }},
    "book_1": {{
        "order": 2,
        "on_top_of": "table_2"
    }},
    "table_2": {{
        "order": 1,
        "on_top_of": null
    }}
}}

The object list is: {object_list}.
```

Figure 25. Prompt for analyzing instance stacking order.