

Supplementary Material for Steerable Scene Generation with Post Training and Inference-Time Search

Nicholas Pfaff¹, Hongkai Dai², Sergey Zakharov², Shun Iwase^{2,3}, Russ Tedrake^{1,2}

¹Massachusetts Institute of Technology, ²Toyota Research Institute, ³Carnegie Mellon University

Website with videos, code, data, and model weights: <https://steerable-scene-generation.github.io>

Contents

A	Procedurally Generated Datasets	3
A.1	Procedural Data Generation	3
A.2	Text Prompt Generation	5
B	Implementation Details	5
B.1	Model Architecture	5
B.2	Model Training	5
B.3	Cotraining Recipe	6
B.4	Physical Feasibility Post Processing	6
B.5	Post Training with Reinforcement Learning	7
B.6	Conditional Generation	7
B.6.1	Text-Conditioned Generation	7
B.6.2	Inpainting	8
B.7	Inference-Time Search via MCTS	8
B.7.1	Mask Generator	8
B.7.2	Reward Function	8
B.7.3	Relation to Gradient-Based Guidance	8
B.8	Comparison of Steering Methods	9
C	Metric Definitions and Evaluation Details	9
D	Additional Results	9
D.1	Unconditional Generation	9
D.2	Post Training with Reinforcement Learning	18
D.3	Conditional Generation	18
D.4	Inference-Time Search	29

D.5	Can We Generate Novel Scenes?	29
E	Ablations	30
E.1	Training Objective	30
E.2	Rotation Representation	30
E.3	Training Precision	31
E.4	Post Processing Pipeline	37

A Procedurally Generated Datasets

Table 1: Dataset statistics for each scene type: number of unique object assets, minimum, maximum, and mean number of objects per scene, and total number of scenes. The final row reports combined statistics across all datasets and corresponds to the dataset used for cotraining across all scene types.

Dataset	# Object Assets	Min Objects	Max Objects	Mean Objects	Total Scenes
Breakfast Table (Low-Clutter)	14	4	26	8.87	6,679,981
Breakfast Table (High-Clutter)	15	4	46	20.96	2,655,407
Dimsum Table	7	7	34	18.14	13,042,071
Living Room Shelf	18	4	23	5.93	9,087,513
Pantry Shelf	14	3	61	27.98	6,171,940
Restaurant (High-Clutter)	27	33	125	60.54	1,195,751
Restaurant (Low-Clutter)	27	9	107	35.08	1,797,170
<i>All Datasets Combined</i>	46	3	125	17.17	44,790,942

We generate a large-scale SE(3) scene dataset to address the lack of existing resources with full 6-DOF object annotations. Prior scene generative models primarily relied on the 3D-FRONT dataset [1], which provides SE(2) room layouts for 18,968 scenes and includes 13,151 unique furniture object assets. Most prior work [2, 3, 4] focused on the bedroom, dining, and living room subsets, where scenes contain between 3 and 21 objects. After preprocessing, these subsets typically yield 4,041 bedroom, 900 dining room, and 813 living room scenes.

In contrast, our dataset includes full SE(3) object poses and dramatically increases the number and variety of sampled arrangements. We generate a total of 44,790,942 scenes across five scene types, with the largest (*Dimsum Table*) containing over 13 million scenes. While our dataset uses only 46 unique object assets (fewer than 3D-FRONT), it achieves substantial variation through randomized object counts and physically diverse placements.

Table 1 summarizes key statistics, and Figure 1 shows object count histograms for each dataset.

A.1 Procedural Data Generation

We adopt the probabilistic context-free scene grammars from Izatt and Tedrake [5] to generate our datasets. These grammars define procedural recipes for constructing scenes as sequences of object types and production rules, and implicitly encode a tree structure in which nodes correspond to objects and edges represent recursive expansion rules.

Sampling begins from a root node (e.g., a table), which stochastically spawns child nodes based on the grammar. For example, a table might generate between one and P place settings, where P is drawn from a discrete distribution and each object pose is sampled from a continuous distribution (e.g., uniform or Gaussian). These place settings can recursively spawn sub-objects such as plates or cutlery, which may in turn generate additional items (e.g., food) until terminal nodes are reached.

As context-free grammars cannot enforce cross-branch constraints (e.g., non-penetration between unrelated objects), Izatt and Tedrake [5] introduced constraints such as physical feasibility via rejection sampling and Hamiltonian Monte Carlo.

We implement one grammar per scene type using the original codebase. However, sampling can be slow for complex scenes, particularly high-clutter environments like our *Restaurant (High-Clutter)* dataset, where many proposed scenes are rejected due to constraint violations. To scale up generation, we parallelize sampling across 192 CPUs on AWS EC2 M7A Metal 48xlarge instances. In the worst case, we achieved a throughput of approximately 60,000 valid scenes per day per instance. More efficient sampling may be possible with improved grammar design or a dedicated distributed sampling system. Once trained, our generative model supports scene sampling at significantly higher throughput than the procedural generation process.

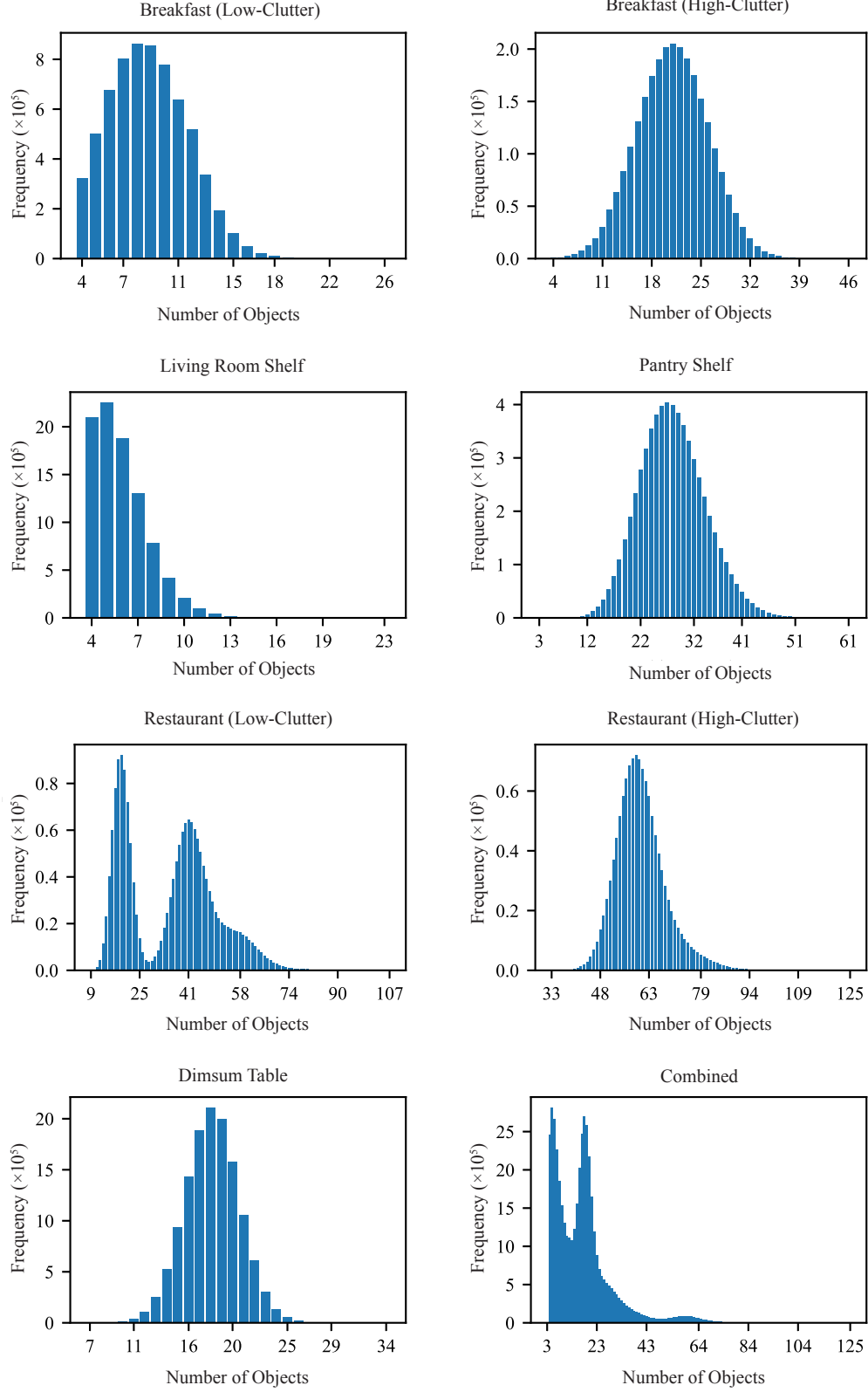


Figure 1: **Dataset statistics.** The object count statistics for each dataset and the combined dataset. The first x-label on each histogram represents the minimum object number, and the last one represents the maximum object number. Note the long tails of the distributions.

A.2 Text Prompt Generation

We generate textual annotations for our procedurally created scenes using a rule-based system, similar to the one used in [3], that maps scene content into natural language descriptions. Each scene is annotated with one or more sentences drawn from three annotation types: object count, object names, and spatial relationships.

Object Count. This annotation describes the total number of non-empty objects in the scene, e.g., “A scene with 32 objects.”

Object Names. We extract object names from the scene and generate a descriptive sentence listing either all objects or a sampled subset, e.g., “A scene with a plate, two bowls, and some other objects.” In room-scale scenes, we prioritize large objects such as tables or shelves when sampling subsets. We apply heuristics for article selection and pluralization (e.g., “a bowl” vs. “an apple”), and express counts using word forms for numbers up to ten (e.g., “two bowls”) and numerals for larger quantities (e.g., “12 bowls”).

Spatial Relationships. We extend the object name sentence with spatial relationships derived from the 3D positions of mentioned objects. These include topological (e.g., “on top of”), directional (e.g., “to the left of”), and containment-based (e.g., “inside”) relations. Only pairs within a distance threshold are considered, and relationships are expressed only between named objects. For shelf and room scenes, we apply scene-specific rules, such as identifying shelf tiers or prioritizing furniture. To disambiguate repeated object types, we insert ordinal labels where necessary (e.g., “the second apple”).

To encourage diversity in annotations, we duplicate each dataset scene three times and generate a different annotation type for each copy.

B Implementation Details

In this section, we go over some of the key implementation details.

B.1 Model Architecture

We adopt a Flux-style architecture [6] using its image branch without positional encoding to preserve object order equivariance (see Figure 2). Our denoising network takes as input the discrete component c_t and continuous components \mathbf{R}_t and \mathbf{p}_t at timestep t , and predicts the corresponding outputs c_{t-1} , \mathbf{R}_{t-1} , and \mathbf{p}_{t-1} for the next step.

The discrete and continuous inputs are embedded separately and added together following Hu et al. [4]. The result is fed into double stream transformer blocks that independently process scene and text features using separate weights, merging during the attention operation. After processing by double stream blocks, the modalities are concatenated along the token dimension and passed through single stream transformer blocks. The final outputs are projected via separate multilayer perceptrons (MLPs) into discrete and continuous components.

We use a frozen BERT-Base encoder [7] for text features, and sinusoidal embeddings for the timestep, which are injected via modulation layers.

B.2 Model Training

We train all models using exponential moving average (EMA) updates [8] and an AdamW optimizer [9] with a learning rate of $2.0\text{e-}4$ and weight decay of $1.0\text{e-}3$. We apply a cosine learning rate schedule with linear warmup over the first 5000 iterations. Training uses a linear diffusion noise schedule with 1000 timesteps for both training and inference. To enable classifier-free guidance (CFG), we drop the conditional input with 10% probability during training.

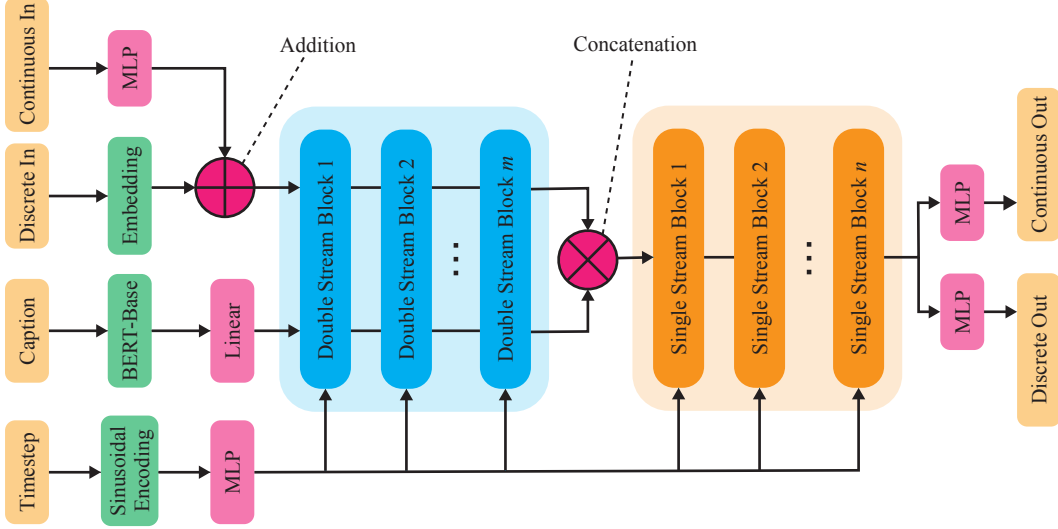


Figure 2: **Denoising network architecture.** Our model follows a Flux-style design [6], operating over mixed discrete and continuous scene representations. Discrete and continuous inputs are separately embedded and added, then passed through double stream transformer blocks that separately process scene and text features before merging at the attention step. After concatenation along the token dimension, single stream transformer blocks further process the fused representation. Final outputs are decoded into discrete and continuous components for the next timestep.

Our denoising network utilizes 5 double-stream and 10 single-stream Transformer blocks, resulting in a total of 88.3 million parameters. Text prompts are encoded using 110 BERT [7] tokens, corresponding to the longest caption across our datasets. All models are trained for 200 thousand iterations with a batch size of 256 per GPU.

Training is performed on 8 NVIDIA A100 GPUs. Most experiments use 40GB A100s, while the Restaurant and cotraining experiments use 80GB A100s. We use full precision during final training to accurately distinguish large translation values in scene poses, which low-precision formats cannot reliably separate. bfloat16 was used during development for faster experimentation and works well for non-room-level datasets with small translation ranges. We provide a training precision ablation in Section E. Overall training times range from approximately 1.06 days for the Dimsum Table dataset to 1.65 days for cotraining across all datasets.

B.3 Cotraining Recipe

During cotraining, we sample batches such that each scene type occurs with equal probability, ensuring balanced exposure to all subdatasets. We maintain a separate infinite iterable stream for each subdataset, using random offsets and buffered shuffling to enhance diversity during sampling. At each step, a subdataset is selected uniformly at random, and a scene is drawn from its corresponding iterator.

We apply this cotraining setup both for shelf cotraining, where the model is jointly trained on the Living Room Shelf and Pantry Shelf datasets, and for full cotraining across all five scene types.

B.4 Physical Feasibility Post Processing

To enforce non-penetration, we apply a projection step that adjusts object translations while keeping orientations fixed. We solve the following nonlinear optimization problem:

$$\min_{\mathbf{p}} \|\mathbf{p} - \mathbf{p}_0\|_2^2 \quad \text{s.t.} \quad d(i, j) \geq 0, \quad \forall i, j \in \{1, \dots, N\}, \quad (1)$$

where \mathbf{p} and \mathbf{p}_0 denote the optimized and original object translations, respectively, and $d(i, j)$ is the signed distance between objects i and j as computed by a collision checker.

We solve this optimization using the SNOPT solver [10]. In some complex scenes, particularly in the Restaurant dataset, SNOPT may fail to converge due to the presence of many complicated nonlinear non-convex constraints, where the objects are in contact without penetration. Finding solutions satisfying such constraints can be challenging for general-purpose nonlinear solvers, especially when using tight numerical tolerances.

To assess the effectiveness of our post processing, we conduct a qualitative ablation comparing scenes generated with (1) no post processing, (2) simulation-only, and (3) projection followed by simulation (see Section E). We find that combining projection and simulation yields the most physically plausible scenes, especially in cluttered environments.

B.5 Post Training with Reinforcement Learning

We fine-tune our continuous diffusion model using DDPO [11], treating the denoising process as a multi-step decision-making problem. We adopt the score function variant (DDPO_{SF}), which uses the REINFORCE estimator without a learned value function or importance sampling.

Trajectory generation. We use a DDIM [12] scheduler with 100–150 steps to generate full denoising trajectories. Gradients are computed either across all timesteps or a uniformly sampled subset, following the strategy in [13]. To maintain consistency between training and inference, we use the same scheduler and number of steps at both stages. Using mismatched settings (e.g., 200 DDIM or 1000 DDPM steps for inference after post training) still improves over the pretrained model but underperforms compared to the configuration used during post training.

Rewards. We use task-specific rewards evaluated only at the final denoised scene ($t = 0$). All results in this paper use object count as the reward signal. Advantages are computed by normalizing the per-batch rewards using the batch mean and standard deviation. To ensure stability in distributed settings, we synchronize these statistics across all workers before computing advantages.

Loss. As a proof-of-concept, we use the standard REINFORCE objective to compute the RL loss:

$$\mathcal{L}_{\text{RL}} = -\mathbb{E} \left[\sum_t \log p(x_t | x_{t+1}) \cdot A \right], \quad (2)$$

where A denotes the advantage. We apply an additional DDPM [14] loss term, weighted by a tunable coefficient λ_{DDPM} , to stabilize optimization [13]. We find values of $\lambda_{\text{DDPM}} \in [100, 200]$ to be effective.

Training is conducted with a batch size of 32 per GPU across 8 NVIDIA A100 80GB GPUs.

Extension to Mixed Discrete-Continuous Models. The DDPO framework can be extended to the discrete D3PM setting [15], and thus also to mixed discrete-continuous diffusion models. In this case, one would need to compute gradients of the form $\frac{\partial \log p_\theta(x_t | x_{t+1})}{\partial \theta}$, which are defined for both DDPM and D3PM formulations. The resulting RL objective would then be the sum of REINFORCE losses over both the continuous (DDPM) and discrete (D3PM) components. A key challenge in this direction is developing few-step samplers for D3PMs to make optimization practical.

B.6 Conditional Generation

B.6.1 Text-Conditioned Generation

We support text-conditioned generation by encoding prompts using a frozen BERT-Base encoder [7]. The resulting embeddings are injected into the conditional input branch of our Flux-style architecture. We use 110 tokens per prompt, corresponding to the maximum prompt length across all scene types in our datasets.

To enable both conditional and unconditional generation within a single model, we randomly mask the conditioning input with 10% probability during training. This allows classifier-free guidance (CFG) [16] at inference time, where predictions are computed using a weighted combination of conditional and unconditional outputs:

$$\hat{x} = (1 + w) \cdot \hat{x}_{\text{cond}} - w \cdot \hat{x}_{\text{uncond}}, \quad (3)$$

where w is the guidance weight, and \hat{x}_{cond} and \hat{x}_{uncond} are the model predictions under conditional and unconditional contexts, respectively. A weight of $w = -1$ corresponds to unconditional sampling, $w = 0$ yields conditional sampling without guidance, and $w > 0$ applies classifier-free guidance during sampling. We apply CFG to both discrete and continuous components.

B.6.2 Inpainting

We perform structured inpainting by masking a subset of objects or object attributes and generating the missing content while preserving the unmasked parts of the scene. Inpainting operates natively over our mixed discrete-continuous representation: masked components are initialized with noise and updated via reverse diffusion, while unmasked components are clamped to their original values throughout sampling.

This approach enables flexible editing operations, such as object rearrangement (resampling poses while keeping categories fixed) and scene completion (synthesizing both categories and poses for empty object slots).

B.7 Inference-Time Search via MCTS

Our inference-time search framework incrementally inpaints masked regions of a scene, guided by a task-specific reward. In this work, we use the number of physically feasible objects as a proof of concept. However, the framework is modular and supports alternative objectives by substituting the mask generator and reward function components.

B.7.1 Mask Generator

Objects are included in the inpainting mask if they are deemed *invalid*, meaning they violate either (1) non-penetration or (2) static equilibrium constraints.

Penetration is detected using Drake’s [17] signed distance checker, and all objects involved in penetrating pairs are masked. Static equilibrium is evaluated by simulating the scene for 0.1s in Drake. An object is deemed unstable if its translation or rotation exceeds predefined thresholds. To avoid instability and spurious support relationships, penetrating objects are excluded from the simulation.

Some objects are predefined as *welded*, such as tables in tabletop scenes and shelves in shelf scenes. These objects serve as supporting surfaces in the absence of a floor and are excluded from masking. Masking welded objects would cause all other objects to fall during simulation, severely degrading search efficiency.

B.7.2 Reward Function

The reward is defined as the number of *physically feasible* objects in the scene—those that are both non-penetrating and statically stable according to the same criteria used in the mask generator. Since each object’s feasibility is evaluated as a binary outcome, the total reward is an integer count of feasible objects.

B.7.3 Relation to Gradient-Based Guidance

In addition to MCTS, diffusion models can also be steered at inference time using gradient-based guidance, which relies on differentiable reward functions and typically incurs lower compute cost. In contrast, MCTS supports arbitrary non-differentiable objectives, making it more broadly applicable within our framework.

B.8 Comparison of Steering Methods

Our framework supports multiple complementary steering strategies:

- **RL post-training** optimizes arbitrary objectives with fast inference but requires additional training.
- **Conditional generation** requires conditioning labels during training and introduces modest extra training cost (e.g., encoding text), while incurring negligible overhead at inference.
- **Inpainting** requires no retraining but is limited to the training distribution; it is useful for rearrangement or adding distractors, and also serves as a building block for MCTS.
- **MCTS** enables flexible optimization with arbitrary objectives without retraining, at the cost of slower inference.

These strategies can be composed, for example, a conditional model can be refined with RL and further steered with MCTS, similar in spirit to modern LLM pipelines.

C Metric Definitions and Evaluation Details

Prior work on floor-aligned scenes with SE(2) poses typically evaluates generative models using top-down renderings and image-based metrics such as Fréchet Inception Distance (FID) and classifier accuracy (CA, in %) [2, 3, 4, 18]. However, these approaches are suboptimal for SE(3) scenes, where a single view may not capture the full scene state due to occlusions. For example, in a cluttered bin, no camera can reveal objects underneath the top layer.

Despite these challenges, image-based metrics remain practical proxies. Due to the lack of pre-trained feature extractors for full SE(3) representations, we reuse SE(2)-based metrics, which provide sufficient sensitivity to distinguish between generative methods. Accordingly, we adopt FID and CA as primary metrics. CA is computed by training a binary classifier to distinguish between real and generated semantic renderings. A CA of 50% indicates indistinguishable distributions, while a CA near 100% indicates clear separability. We average CA over 10 runs with random train/test splits to reduce variance. To ensure metric sensitivity across datasets, we tune classifier strength (i.e., training iterations) per dataset; this tuning is fixed across methods for fair comparison. Scenes are rendered from a manually selected informative viewpoint at 640×480 resolution using semantic colors assigned by object class as in [3]. Figure 3 shows example semantic renderings used for CA and FID evaluation.

In addition to FID and CA, we report three further metrics. We compute KL divergence between object category distributions following prior work [2]; although all models achieve low KL values on our datasets, we include it for completeness. For conditional generation, we report average prompt-following accuracy (APF), computed for two automatically verifiable prompt types: (1) number of objects, and (2) inclusion of specified object categories. For unconditional generation, we report the median total penetration (MTP) measured in centimeters before applying physical feasibility post processing; after projection, MTP is guaranteed to be zero. All metrics are computed using 5,000 training scenes and 5,000 generated scenes. For conditional generation, prompts are extracted from the training scenes and reused during synthesis.

D Additional Results

D.1 Unconditional Generation

We provide additional quantitative results for unconditional generation across all five datasets. Table 2 reports results for the Pantry Shelf and Breakfast Table (Low-Clutter) datasets, Table 3 for the Dimsum Table and Breakfast Table (High-Clutter), and Table 4 for the Restaurant (Low-Clutter) dataset.

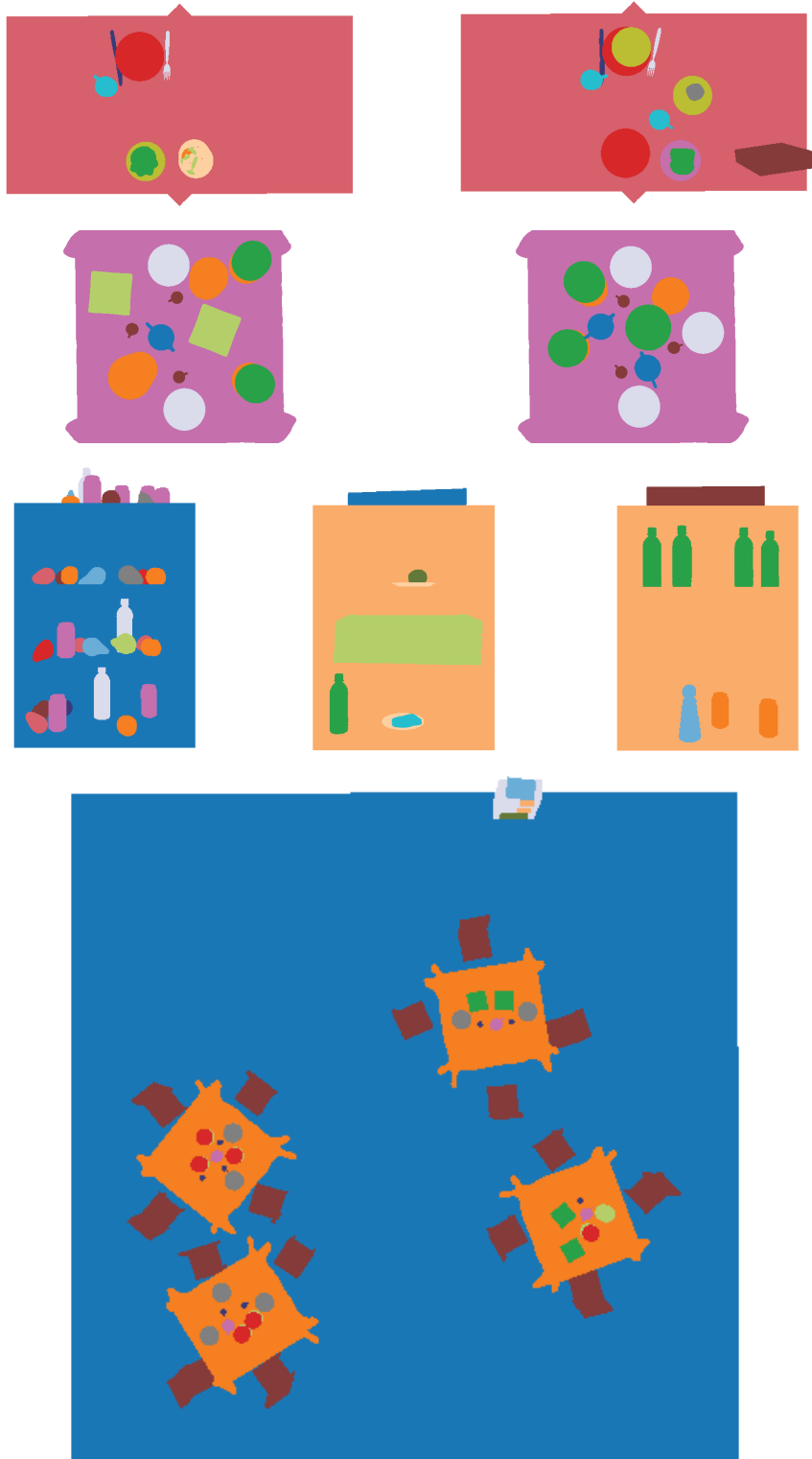


Figure 3: Semantic renderings used to compute image-based metrics such as classifier accuracy (CA) and Fréchet Inception Distance (FID). Rows correspond to the Breakfast Table, Dimsum Table, Pantry and Living Room Shelves, and Restaurant datasets, respectively.

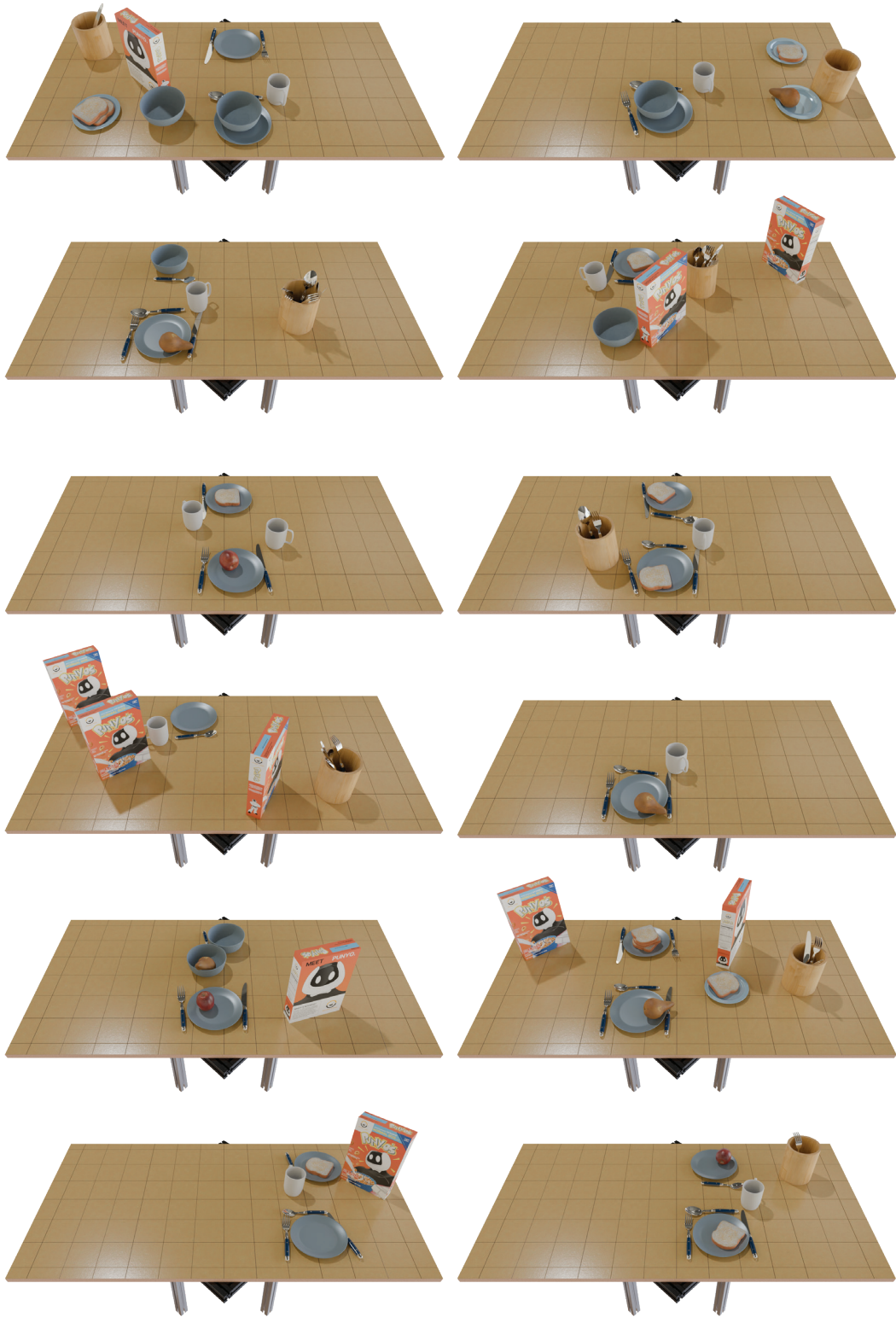


Figure 4: Unconditional generation results for a model trained on the Breakfast Table (Low-Clutter Variant) dataset.

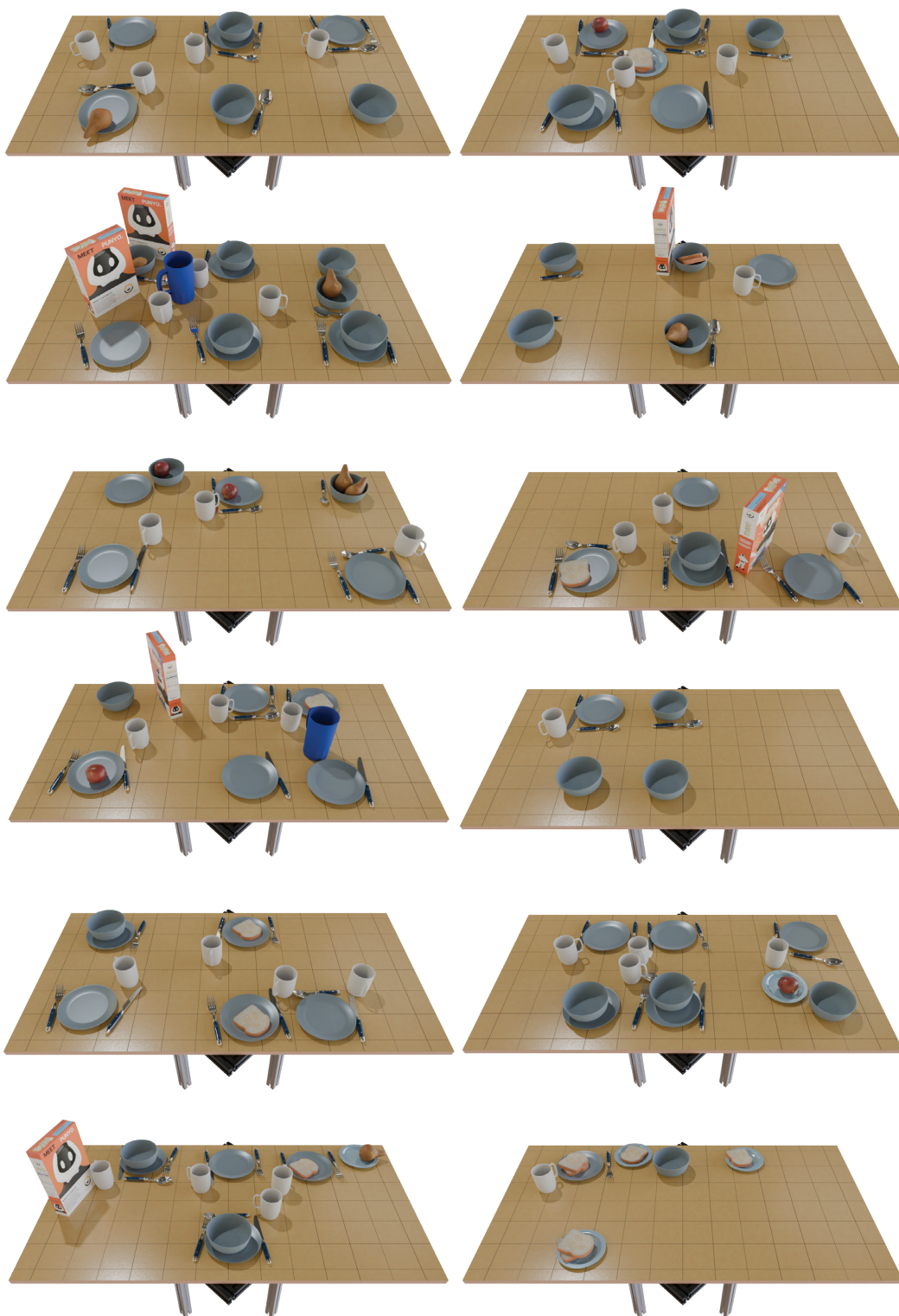


Figure 5: Unconditional generation results for a model trained on the Breakfast Table (High-Clutter Variant) dataset.



Figure 6: Unconditional generation results for a model trained on the Dimsum Table dataset.



Figure 7: Unconditional generation results for a model trained on the Living Room Shelf dataset.



Figure 8: Unconditional generation results for a model trained on the Pantry Shelf dataset.

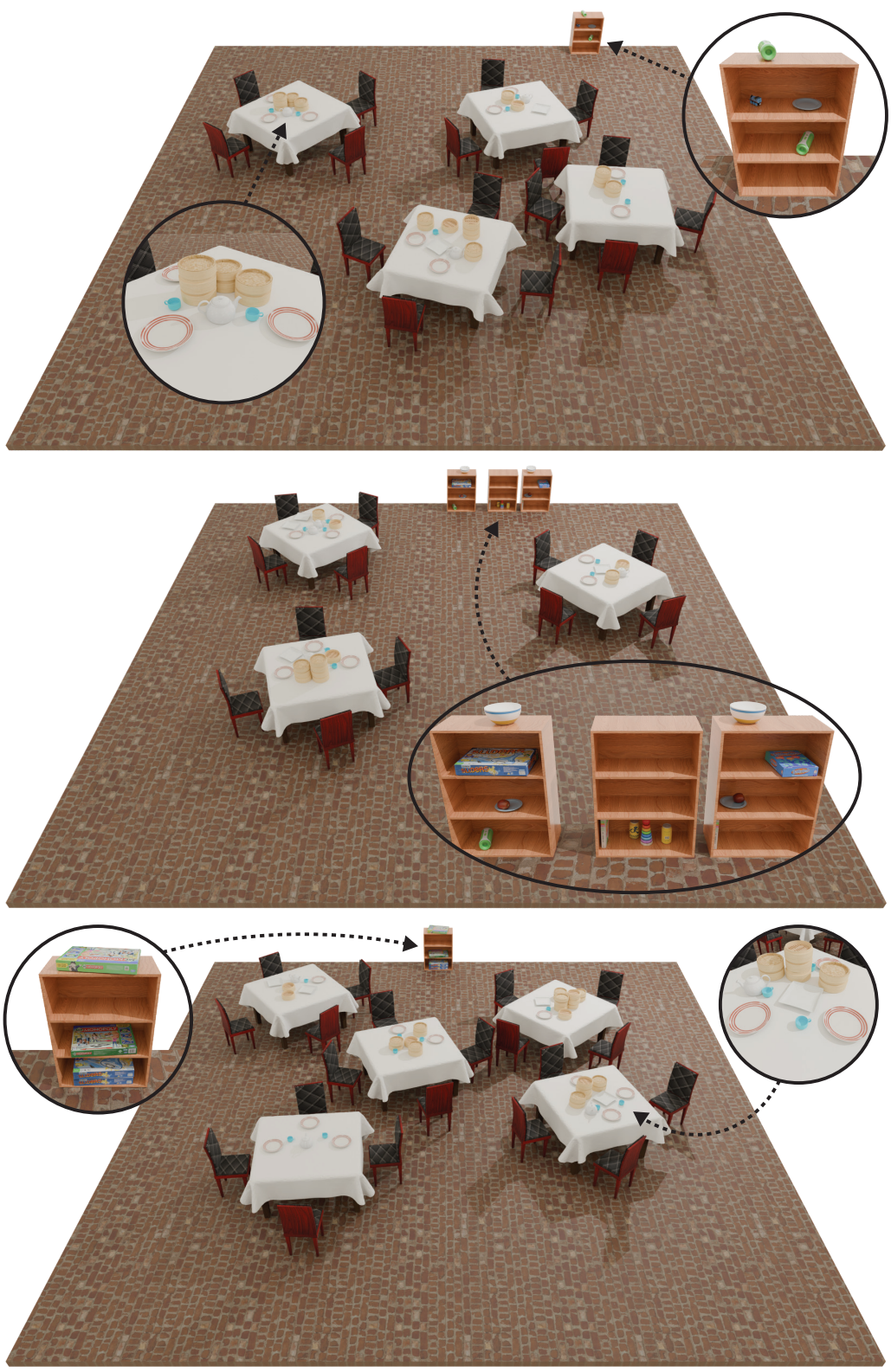


Figure 9: Unconditional generation results for a model trained on the Restaurant (High-Clutter) dataset.



Figure 10: Unconditional generation results from a model trained jointly on all our scenes with equal batch mixing ratios across all five scene types.

Table 2: Unconditional generation results on the Pantry Shelf and Breakfast Table (Low-Clutter) datasets. * indicates that we adjusted the methods for compatibility with our scene representation.

Method	Pantry Shelf				Breakfast Table (Low-Clutter Variant)			
	CA (50 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓	CA (25 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓
DiffuScene* [3]	84.38 \pm 1.79	1.99	1.97	9.32	68.32 \pm 8.48	2.64	2.59	5.94
MiDiffusion* [4]	85.42 \pm 0.98	1.18	1.90	5.57	65.34 \pm 7.39	1.79	2.43	3.81
Ours	84.04 \pm 0.58	1.45	1.89	4.92	57.85 \pm 4.98	1.67	2.39	3.50

Table 3: Unconditional generation results on the Dimsum Table and Breakfast Table (High-Clutter) datasets. * indicates that we adjusted the methods for compatibility with our scene representation.

Method	Dimsum Table				Breakfast Table (High-Clutter Variant)			
	CA (100 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓	CA (50 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓
DiffuScene* [3]	83.56 \pm 8.14	0.39	0.95	0.18	81.71 \pm 5.00	1.57	1.86	6.97
MiDiffusion* [4]	78.69 \pm 10.25	0.57	0.96	0.19	78.86 \pm 4.89	0.84	1.943	7.31
Ours	57.66 \pm 4.70	0.63	0.89	0.20	69.82 \pm 5.37	0.81	1.88	6.97

Our model achieves the lowest classifier accuracy (CA) across all datasets, indicating strong alignment with dataset distributions. We also obtain the lowest FID on all datasets except Breakfast Table (High-Clutter), where our score is competitive and within a small margin of the best. In terms of physical realism, we achieve the lowest mean total penetration (MTP) scores on all datasets except Dimsum Table, where our score is again comparable to the best. While we do not always achieve the lowest KL divergence, all methods exhibit low KL values across datasets, suggesting it is less discriminative as an evaluation metric in this setting.

We also show qualitative unconditional generation results. Figure 4 shows samples from a model trained on the Breakfast Table (Low-Clutter) dataset, Figure 5 from Breakfast Table (High-Clutter), Figure 6 from Dimsum Table, Figure 7 from Living Room Shelf, Figure 8 from Pantry Shelf, and Figure 9 from Restaurant (High-Clutter). These visualizations provide an overview of the types of scenes in our datasets and the outputs produced by our generative model. Finally, Figure 10 shows samples from a model trained jointly on all scene types, as described in Section B.3, demonstrating that the model remains effective under multi-distribution training.

D.2 Post Training with Reinforcement Learning

We show additional unconditional generation results after post training with reinforcement learning using an object count reward. Figure 11 displays samples from a model trained on the Dimsum Table dataset, where RL fine-tuning leads to tall steamer tray stacks that were rarely seen during pretraining. Figure 12 shows samples from the Living Room Shelf dataset, where we also increase the maximum object capacity by 20 prior to post training. The model adapts to fill this expanded capacity, generating denser shelf scenes. These results further support the conclusion that RL post training can effectively steer generative models toward task-aligned objectives without retraining from scratch.

D.3 Conditional Generation

We provide additional results for conditional generation across all datasets. Quantitative results are reported in Table 5 for the Dimsum Table and Breakfast Table (Low-Clutter) datasets, Table 6 for the Living Room Shelf and Restaurant (High-Clutter) datasets, and Table 7 for the Restaurant (Low-Clutter) dataset.

Our method achieves the lowest classifier accuracy (CA) across all datasets, indicating the strongest alignment between generated scenes and input prompts. We also obtain the highest prompt-following accuracy (APF) and the lowest FID across all datasets, along with competitive KL scores. These results suggest that our model effectively balances prompt adherence and scene realism.

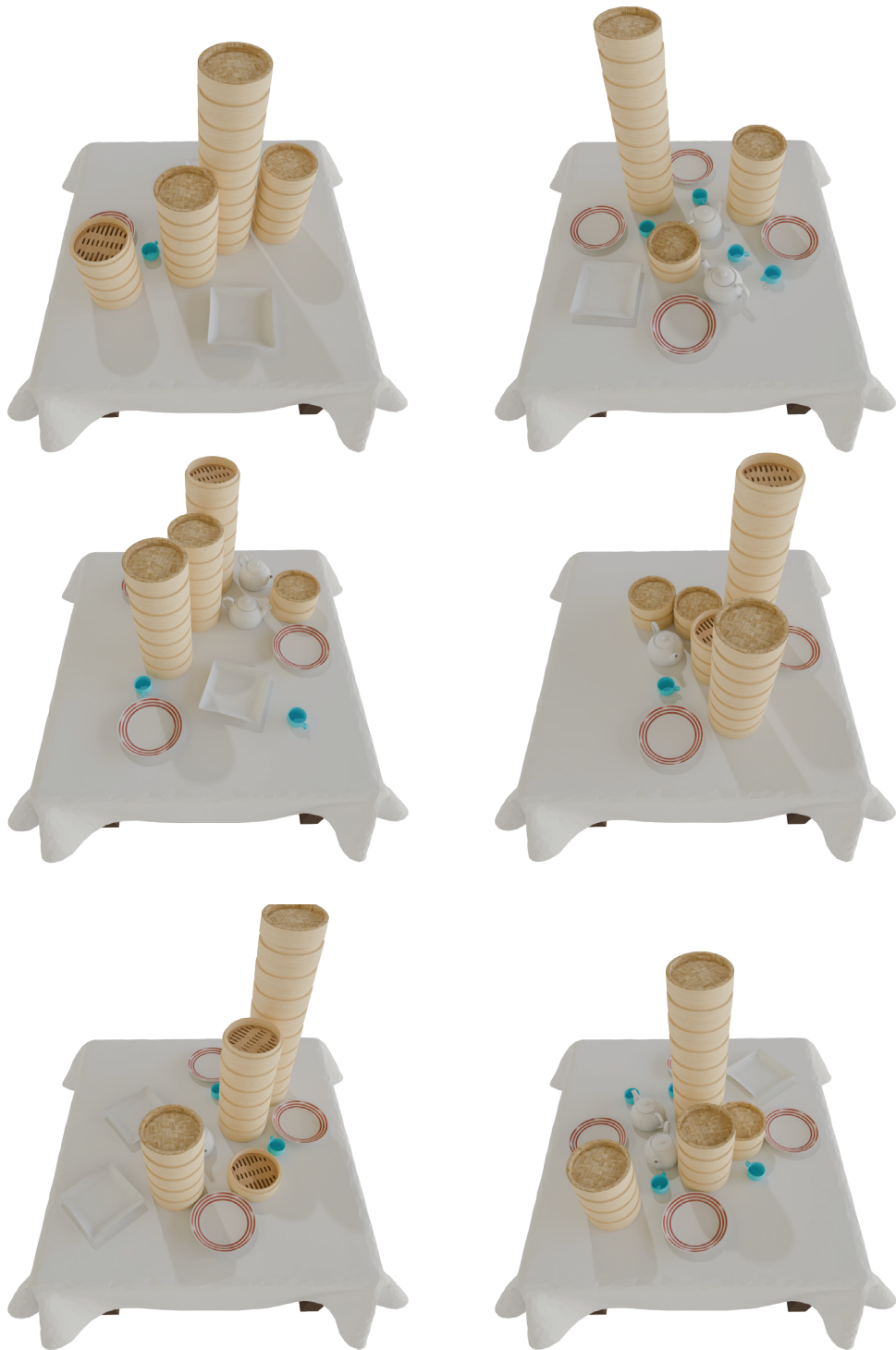


Figure 11: Unconditional generation results from a model trained on the Dimsum dataset after post training with RL using an object number reward. Notice that the model learns to make tall steamer tray stacks.

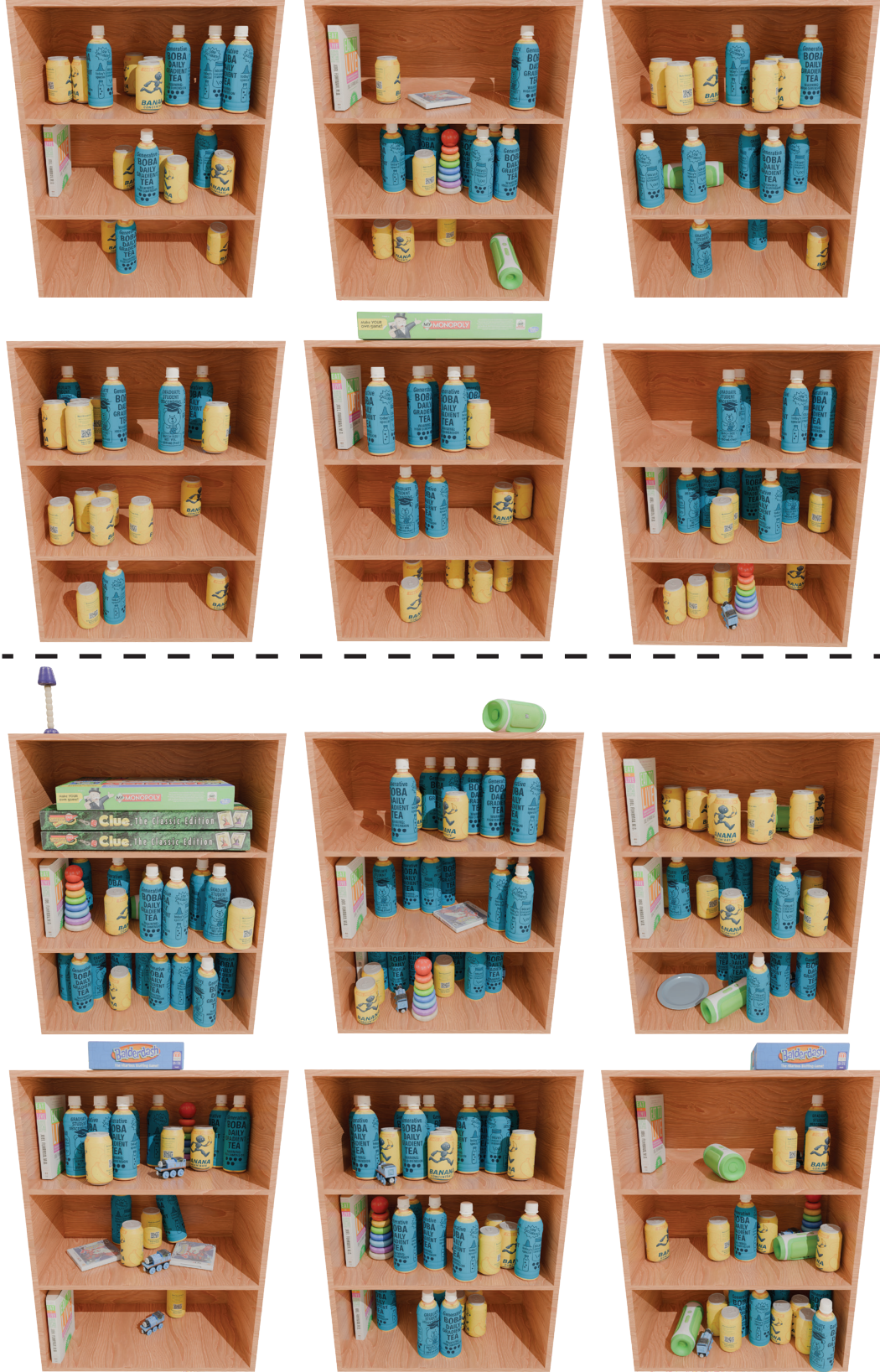


Figure 12: Unconditional generation results from a model trained on the Living Room Shelf dataset after post training with RL using an object number reward. Samples below the dotted line are from increasing the maximum object number by 20 before post training.

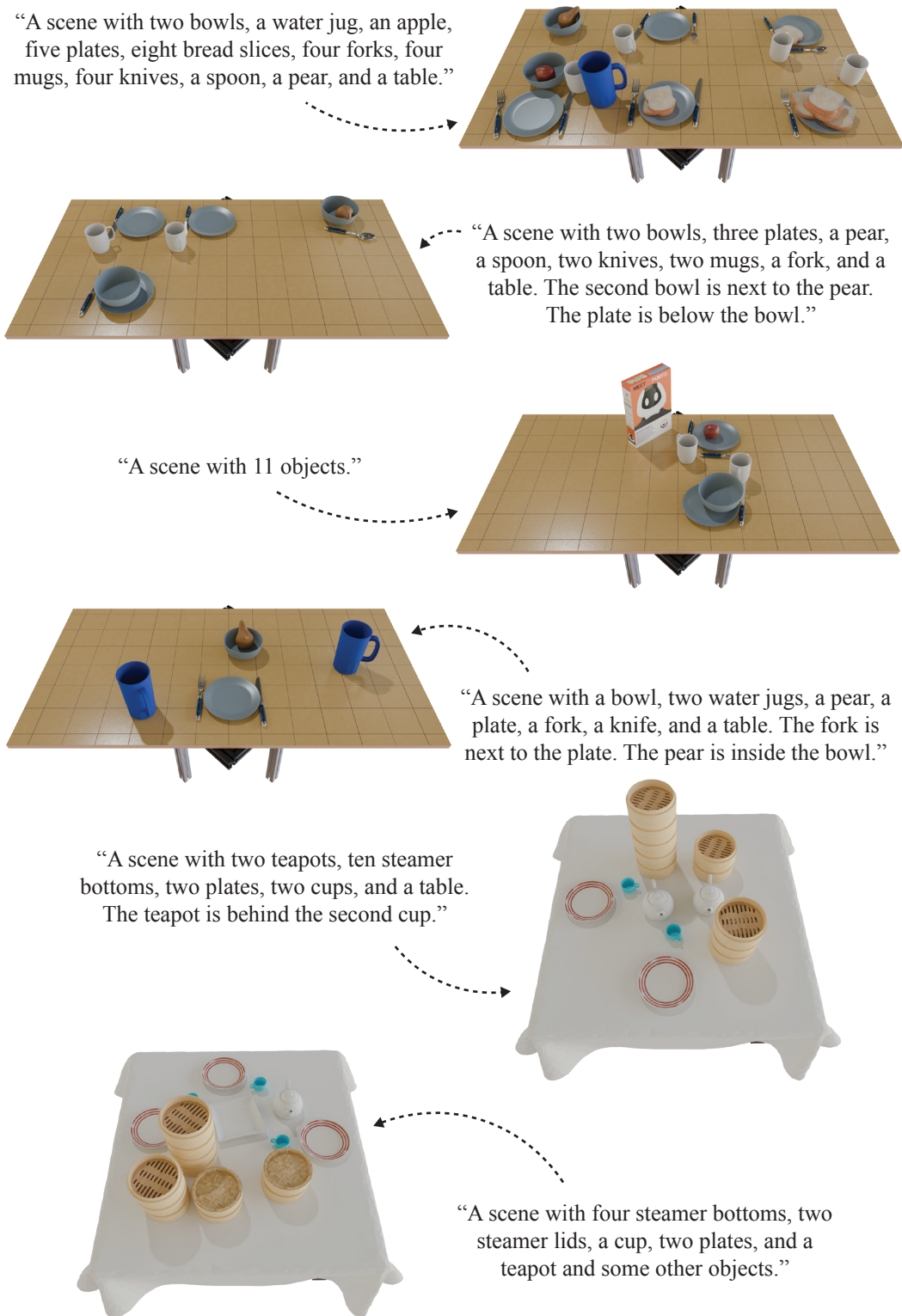


Figure 13: Text conditional generation results for models trained on the Breakfast Table (High-Clutter), Breakfast Table (Low-Clutter), and Dimsum Table datasets.



“A scene with a board game, a stacking ring, a tea bottle, a plate, a bread slice, and a shelf. The stacking ring is on the upper middle shelf. The bread slice is on the lower middle shelf. The board game is on the bottom shelf. The tea bottle is on the lower middle shelf.”

“A scene with five board games, a bowl, a toy train, a plate, and a shelf. The board game is on the lower middle shelf. The bowl is on the top shelf. The fourth board game is next to the second board game. The second board game is on the upper middle shelf. The toy train is on the bottom shelf.”



“A scene with 37 objects.”

“A scene with a shelf, three cans, three tea bottles, 15 apples, six pears, and seven avocados.”



Figure 14: Text conditional generation results for models trained on the Living Room Shelf and Pantry Shelf datasets.

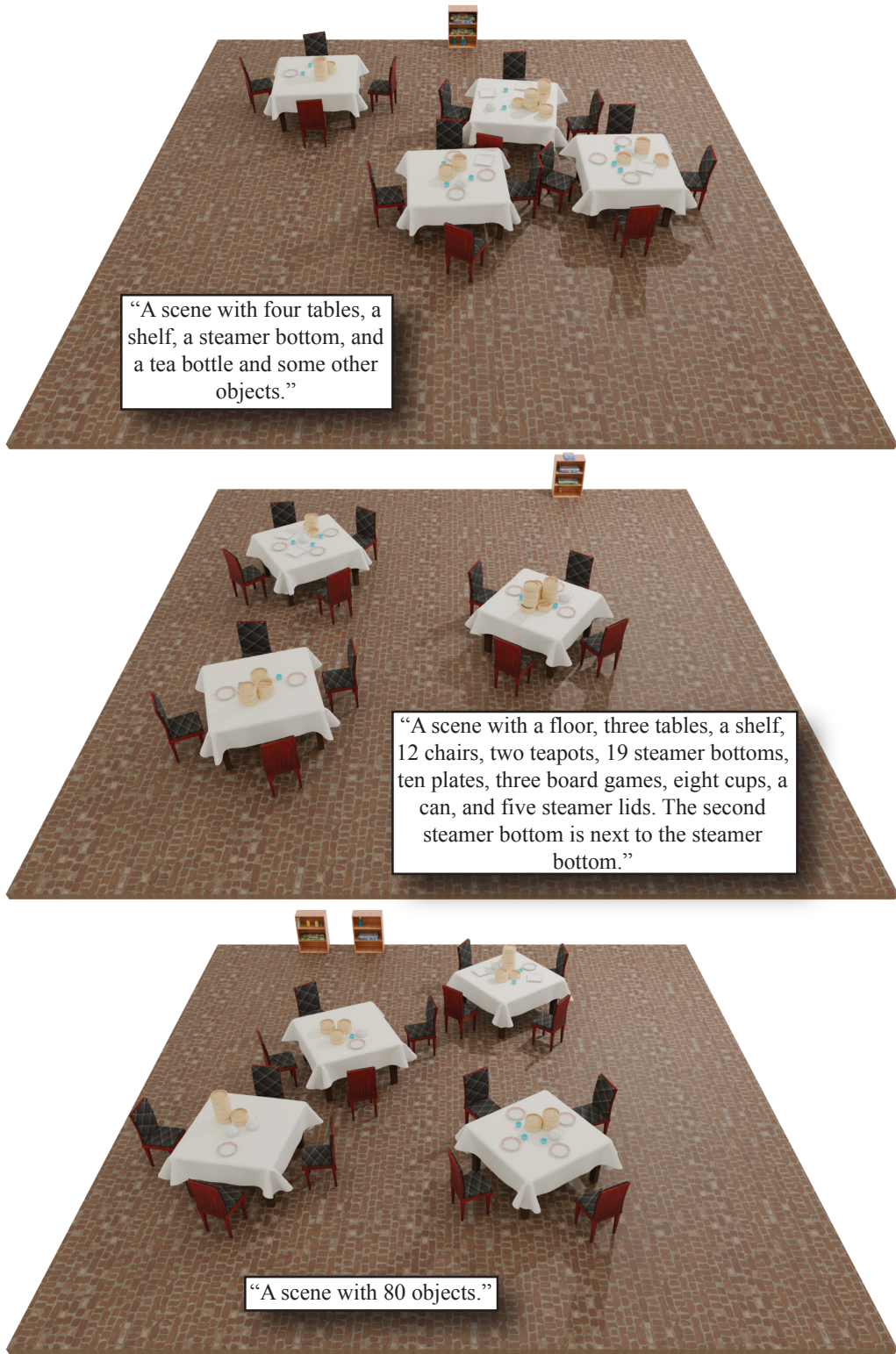


Figure 15: Text conditional generation results for a model trained on the Restaurant (High-Clutter) dataset.

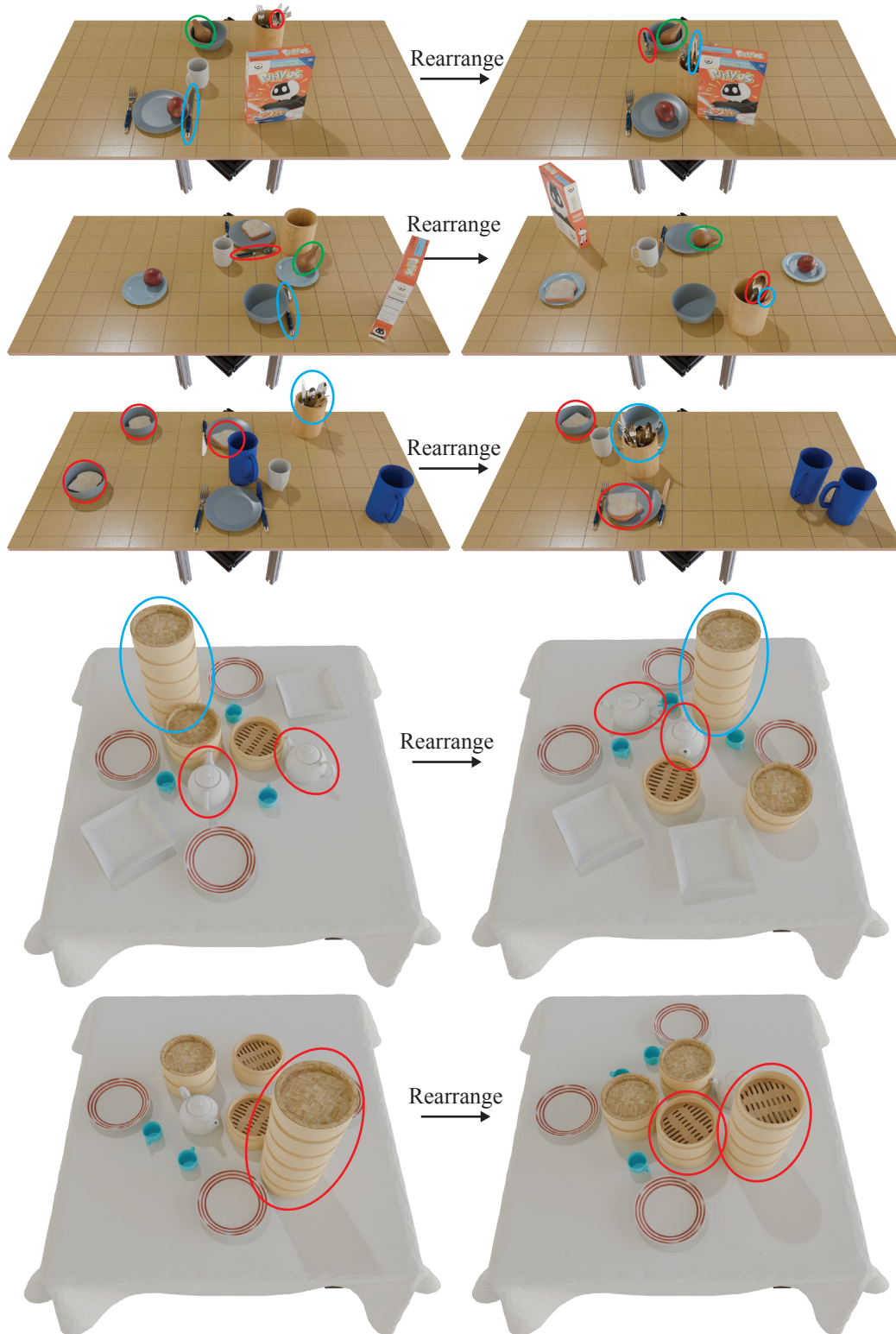


Figure 16: Rearrangement results for models trained on the Breakfast Table and Dimsum datasets. Some objects are highlighted in red, green, and blue ellipses to facilitate easier associations before and after rearrangement.

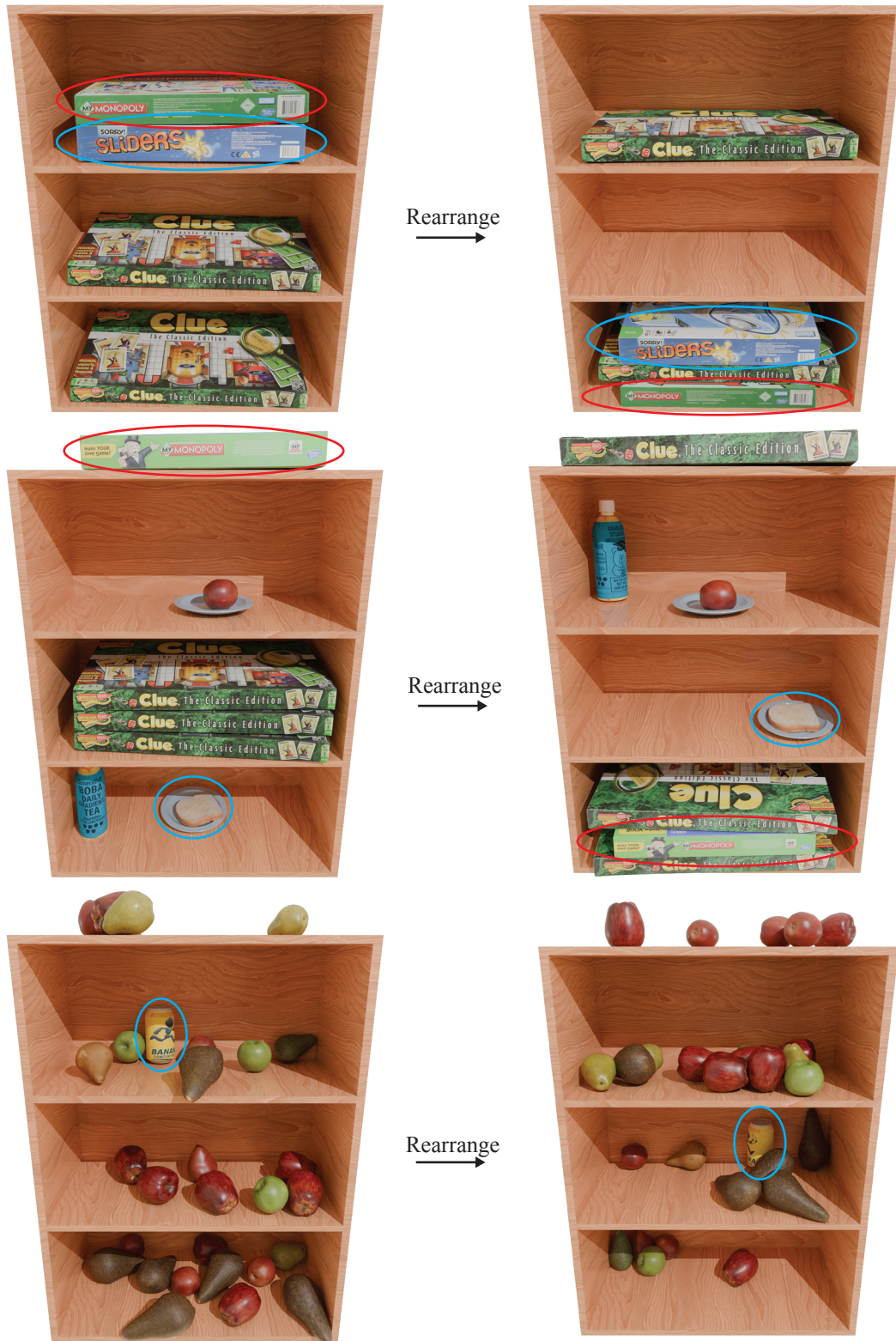


Figure 17: Rearrangement results for models trained on the Living Room Shelf and Pantry Shelf datasets. Some objects are highlighted in red and blue ellipses to facilitate easier associations before and after rearrangement.

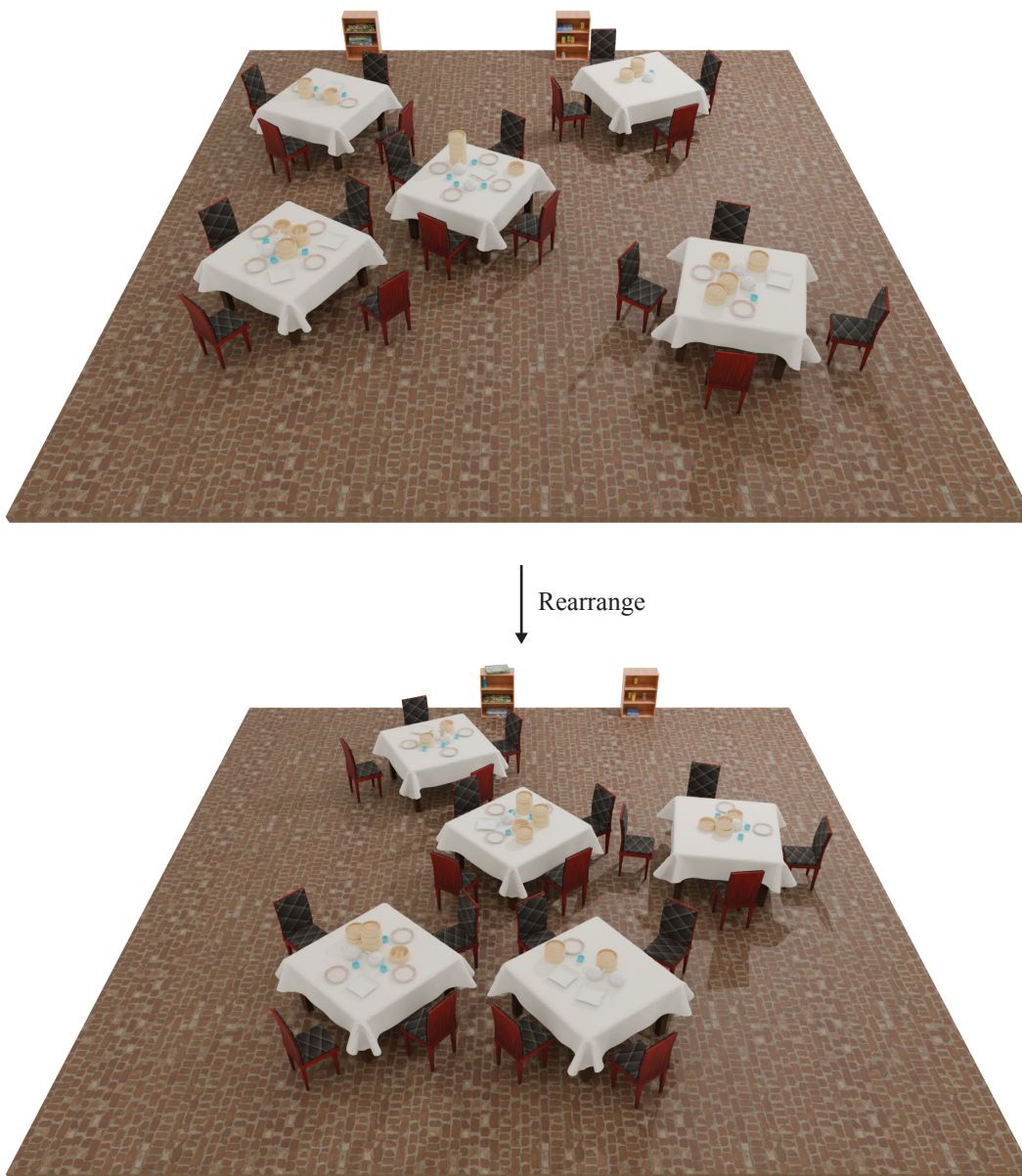


Figure 18: Rearrangement results for a model trained on the Restaurant (High-Clutter) dataset.

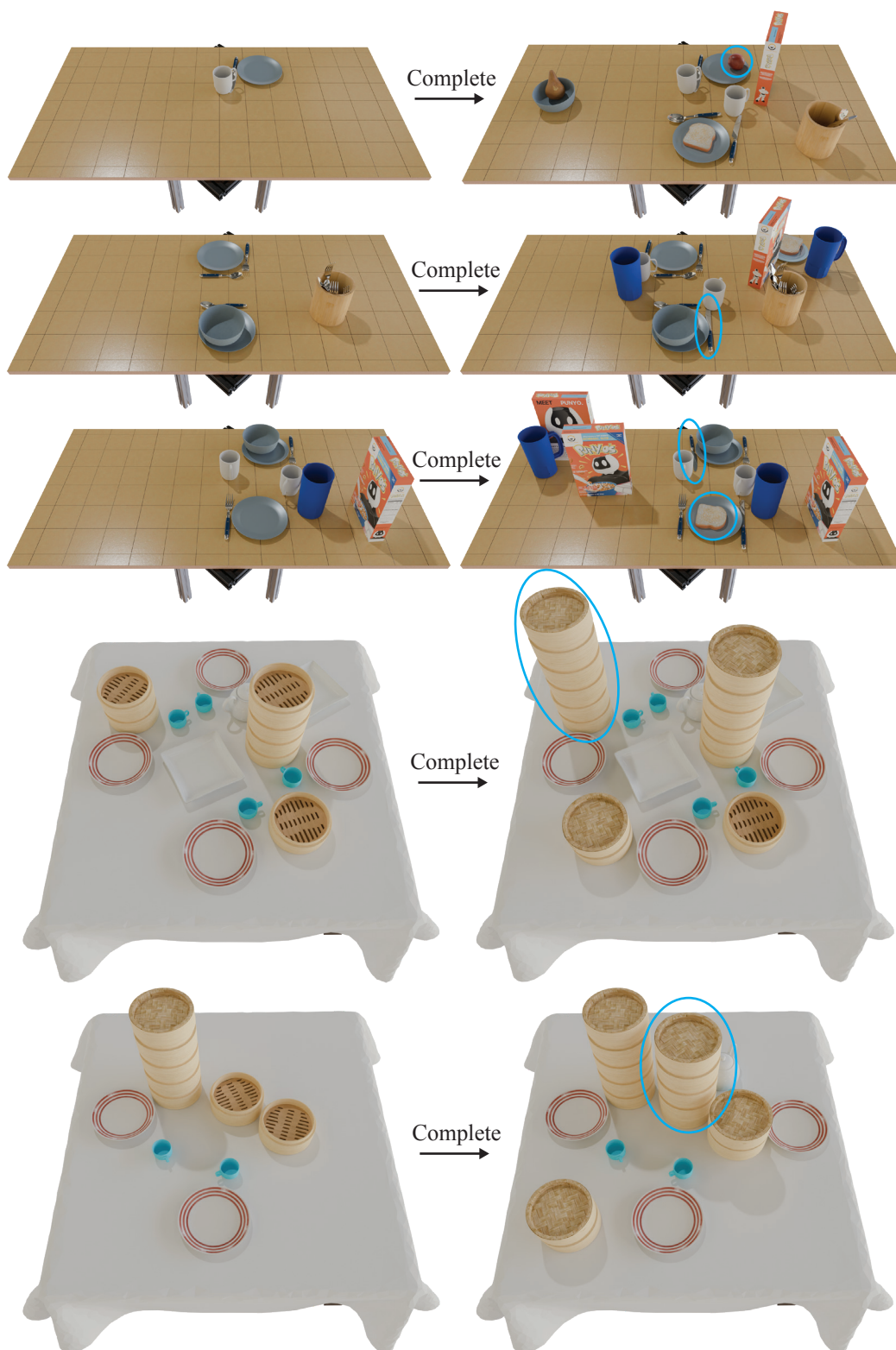


Figure 19: Completion results for models trained on the Breakfast Table and Dimsum datasets. Some of the added objects are highlighted with blue ellipses.

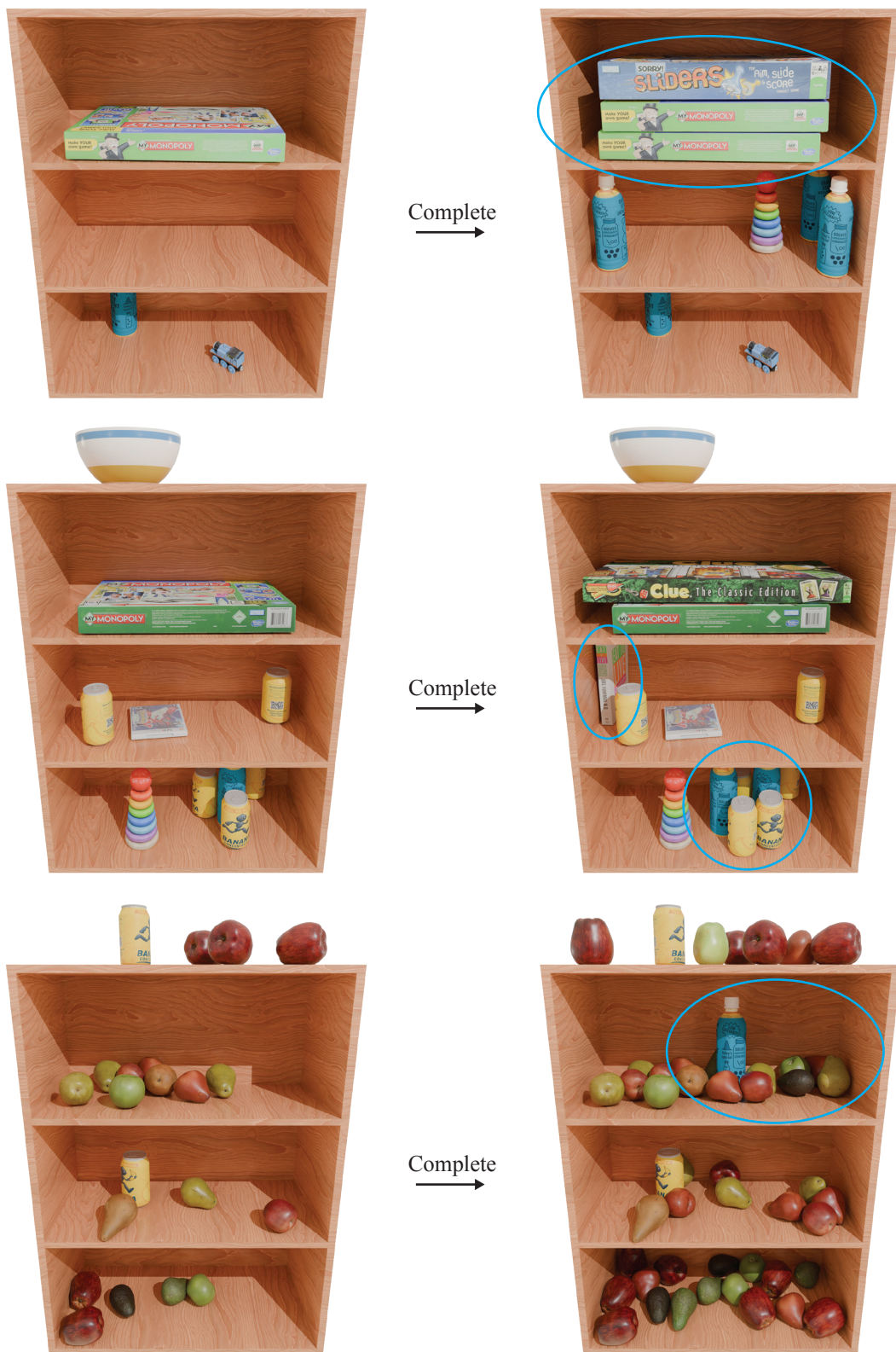


Figure 20: Completion results for models trained on the Living Room Shelf and Pantry Shelf datasets. Some of the added objects are highlighted with blue ellipses.

Table 4: Unconditional generation results on the Restaurant (Low-Clutter) dataset. * indicates that we adjusted the methods for compatibility with our scene representation.

Method	Restaurant (Low-Clutter Variant)			
	CA (50 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓
DiffuScene* [3]	78.41 \pm 8.14	1.60	1.52	7.22
MiDiffusion* [4]	75.34 \pm 10.48	1.01	1.45	3.50
Ours	66.11 \pm 6.93	1.18	1.35	2.39

Table 5: Conditional generation results on the Dimsum Table and Breakfast Table (Low-Clutter) datasets. * indicates that we adjusted the methods for compatibility with our scene representation.

Method	Dimsum Table				Breakfast Table (Low-Clutter Variant)			
	CA (100 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	APF ↑	CA (25 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	APF ↑
DiffuScene* [3]	87.64 \pm 6.34	0.22	0.94	0.90	69.13 \pm 8.71	3.12	2.62	0.83
MiDiffusion* [4]	79.26 \pm 10.48	0.08	0.92	0.84	67.91 \pm 8.29	1.78	2.53	0.77
Ours	60.85 \pm 6.52	0.18	0.89	0.90	58.46 \pm 5.35	1.34	2.42	0.87

We include qualitative examples for three conditional generation tasks. Prompted generation results are shown in Figures 13–15, illustrating scene synthesis from text prompts across all datasets. Rearrangement results are shown in Figures 16–18, where the model regenerates SE(3) object poses while keeping object asset IDs fixed. Completion results are shown in Figures 19–20, where the model fills in initially empty object slots, adding new objects while preserving the rest of the scene. Figure 21 demonstrates cross-dataset interpolation using a model jointly trained on Living Room Shelf and Pantry Shelf scenes, showing smooth transitions between distributions when guided by interpolated prompts.

D.4 Inference-Time Search

We include additional qualitative results for inference-time search with MCTS across several datasets. Figures 22–24 show initial and final scenes, illustrating how MCTS increases physical feasibility while preserving realistic structure.

Figure 22 presents examples from the Restaurant (Low-Clutter) and Dimsum datasets, where the search increases object count while maintaining coherent arrangements. Figure 23 shows results from the Living Room Shelf dataset, including an example where the object capacity was increased by 30 prior to search, demonstrating MCTS’s ability to extrapolate to higher-density scenes than were seen during training. Finally, Figure 24 shows a case from the Restaurant (High-Clutter) dataset where nearly the entire initial scene is replaced, except for a single retained chair, highlighting MCTS’s flexibility in reshaping scenes when necessary.

These results show that MCTS can adapt scene samples to better meet downstream objectives without additional training.

D.5 Can We Generate Novel Scenes?

To evaluate whether our model memorizes training scenes or learns to generalize, we compare unconditional samples to their nearest neighbors in the training set using the Sinkhorn-Knopp Optimal Transport distance [19]. Figures 25 and 26 visualize these comparisons across four datasets. For each generated scene (left), we show its closest training scene (right).

We adopt Optimal Transport (OT) because our scene representation is an unordered set of objects, making standard vector-space distances such as L2 ill-suited. OT enables comparison between such sets while respecting permutation invariance. We utilize the Sinkhorn approximation for computational efficiency, enabling scalable pairwise comparisons between high-dimensional object sets without requiring exact matching.

Table 6: Conditional generation results on the Living Room Shelf and Restaurant (High-Clutter) datasets. * indicates that we adjusted the methods for compatibility with our scene representation.

Method	Living Room Shelf				Restaurant (High-Clutter Variant)			
	CA (100 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	APF ↑	CA (50 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	APF ↑
DiffuScene* [3]	69.33 \pm 1.35	3.74	2.11	0.99	87.22 \pm 6.73	0.48	1.42	0.61
MiDiffusion* [4]	72.66 \pm 3.01	1.31	2.10	0.97	77.97 \pm 11.16	0.66	1.31	0.53
Ours	54.23 \pm 2.18	1.91	2.09	0.99	70.42 \pm 8.19	0.55	1.31	0.84

Table 7: Conditional generation results on the Restaurant (Low-Clutter) dataset. * indicates that we adjusted the methods for compatibility with our scene representation.

Method	Restaurant (Low-Clutter Variant)			
	CA (100 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	APF ↑
DiffuScene* [3]	79.58 \pm 9.00	1.02	1.50	0.74
MiDiffusion* [4]	77.96 \pm 7.24	1.31	1.45	0.59
Ours	67.39 \pm 7.56	1.31	1.36	0.86

Across all examples, we observe clear differences between the generated and nearest training scenes. While some similarities may exist, such as object types or broad spatial patterns, the generated scenes contain novel combinations of objects and distinct SE(3) configurations. This indicates that our model synthesizes new samples from the learned distribution rather than memorizing the training data.

These results support the conclusion that diffusion-based scene models can generate diverse, realistic, and physically plausible scenes without overfitting to the training set.

E Ablations

We ablate key components of our scene generation method to understand their contribution to the final model performance. Specifically, we study the effects of the training objective, rotation representation, training precision, and post-processing pipeline.

E.1 Training Objective

Table 8: Training objective ablation. Unconditional generation results on Breakfast Table (Low-Clutter) and Dimsum Table. “Mixed” refers to our standard model, and “DDPM” refers to our model trained with the continuous-only objective from [3].

Method	Breakfast Table (Low-Clutter Version)				Dimsum Table			
	CA (25 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓	CA (100 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓
DDPM	66.94 \pm 8.19	2.16	2.45	3.33	76.70 \pm 11.86	0.49	0.94	0.19
Mixed	57.85 \pm 4.98	1.67	2.39	3.50	57.66 \pm 4.70	0.63	0.89	0.20

We compare our mixed discrete-continuous training objective to a continuous-only DDPM objective that treats both object categories and poses as continuous variables, following Tang et al. [3]. As shown in Tables 8 and 9, the mixed objective consistently achieves lower classifier accuracy (CA) and FID across all datasets, indicating improved alignment with the data distribution and higher sample quality. While the DDPM-only variant slightly reduces mean total penetration (MTP) on some datasets, it performs worse on key alignment and realism metrics. These results highlight the benefit of jointly modeling discrete and continuous factors when training SE(3) scene generative models.

E.2 Rotation Representation

We evaluate three rotation parameterizations: axis-angle, quaternion, and the \mathbb{R}^9 representation projected onto SO(3) using SVD, following Geist et al. [20], which is used in our method. As

"A scene with an avocado, a stacking ring, five apples,
two cans, and some other objects."



"A scene with an avocado, a board game,
and some other objects."



Figure 21: Conditional generation results from a model jointly trained on the Living Room Shelf and Pantry Shelf datasets. The results show that the model can interpolate between the two data distributions when guided by text conditioning. All results use a classifier-free guidance weight of 5. Some of the objects that only appear in the Living Room Shelf dataset are highlighted in red, while some of the objects that only appear in the Pantry Shelf dataset are highlighted in blue.

shown in Table 10, the \mathbb{R}^9 +SVD representation achieves the best classifier accuracy (CA) and FID on both datasets, with competitive mean total penetration (MTP). These results suggest that learning in an unconstrained higher-dimensional space and projecting to $SO(3)$ at sample time yields more accurate and realistic generations.

E.3 Training Precision

We compare models trained with full precision (32-bit) and mixed precision (bfloat16), evaluating each using the same precision setting as used during training. As shown in Table 11, full precision consistently outperforms mixed precision across CA, FID, and MTP, with especially large gains on MTP, reflecting the metric’s sensitivity to small physical violations near the boundary between static equilibrium and penetration. These results suggest that while mixed precision may reduce training cost and is useful during development, it can degrade generation quality for physically grounded tasks that require fine-grained accuracy.

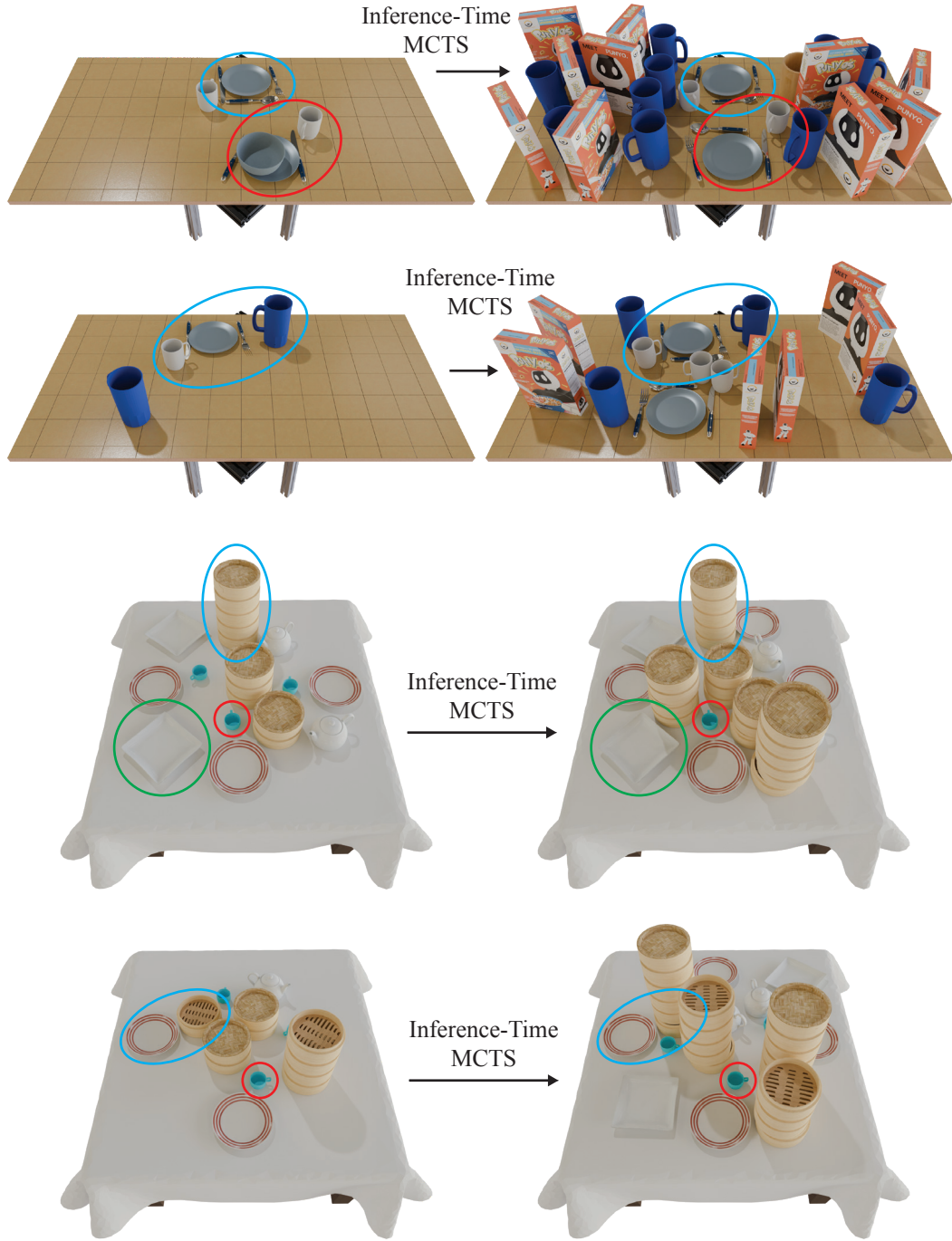


Figure 22: Inference-time MCTS results for models trained on the Restaurant (Low-Clutter) and Dimsum datasets. Initial samples are shown on the left, and final samples (after search) on the right. Objects that remain constant are highlighted with colored ellipses.

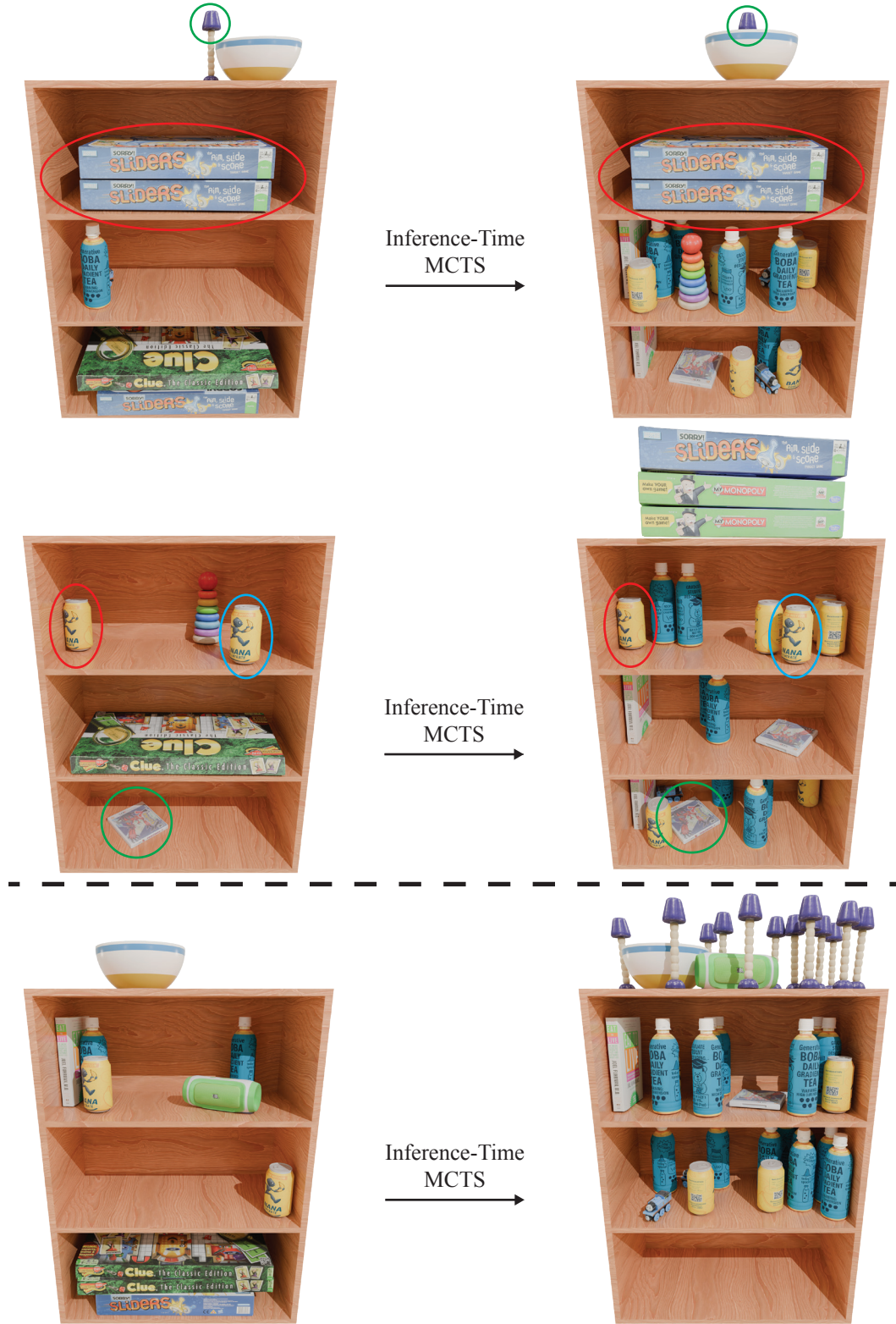


Figure 23: Inference-time MCTS results for models trained on the Living Room Shelf dataset. Initial samples are shown on the left, and final samples (after search) on the right. Objects that remain constant are highlighted with colored ellipses. The result below the dotted line is from increasing the maximum number of objects allowed by the scene representation by 30 before the search.

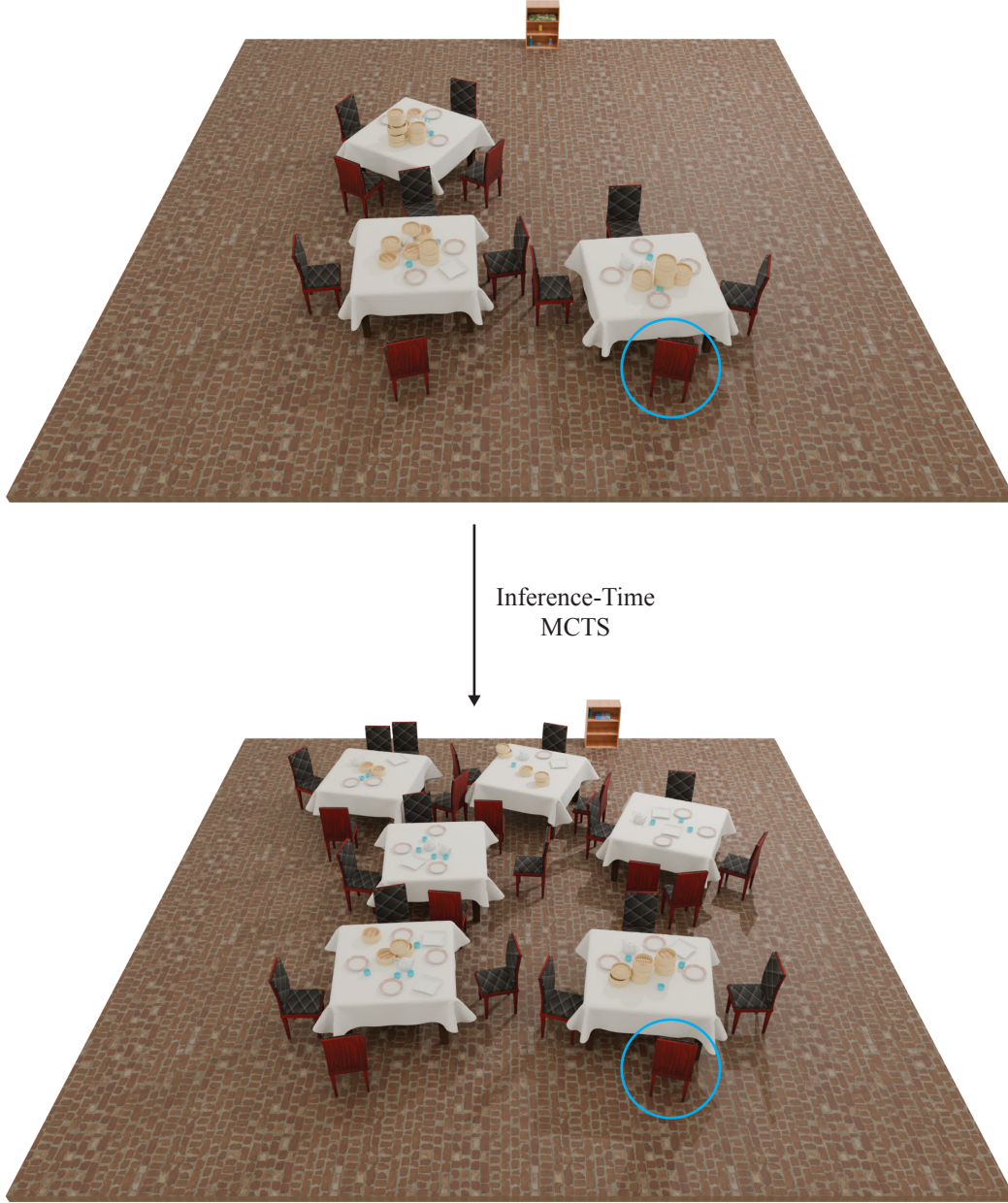


Figure 24: Inference-time MCTS results for a model trained on the Restaurant (High-Clutter) dataset. The initial sample is shown at the top, and the final sample (after search) is shown at the bottom. Only the highlighted chair from the initial scene remains present in the final one.

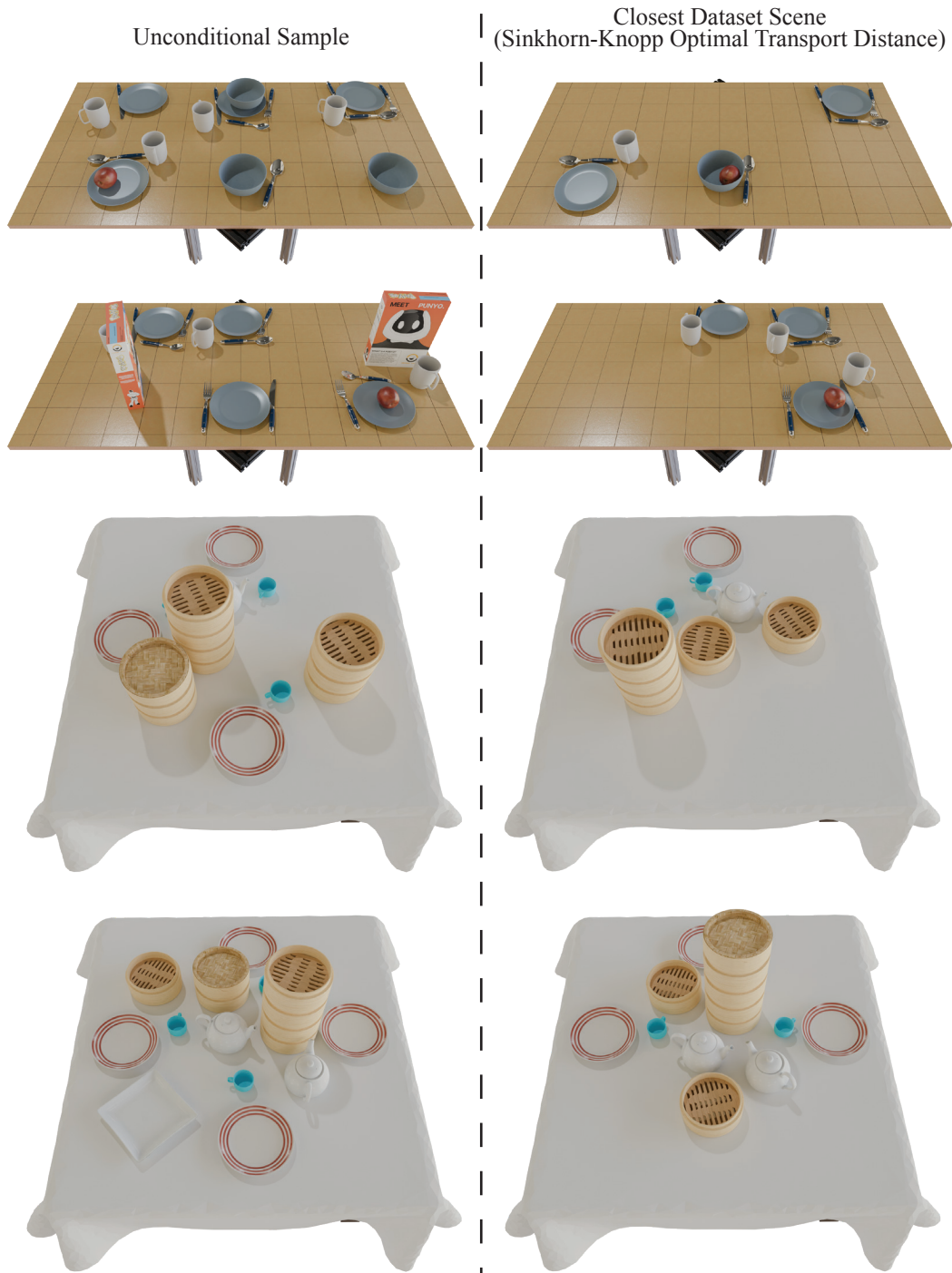


Figure 25: Unconditional samples and their closest dataset scenes for models trained on the Breakfast Table (High-Clutter) and Dimsum Table datasets. Notice that the training scenes are quite distinct from the generated scenes, indicating that our models learn the distribution rather than memorize the training data.

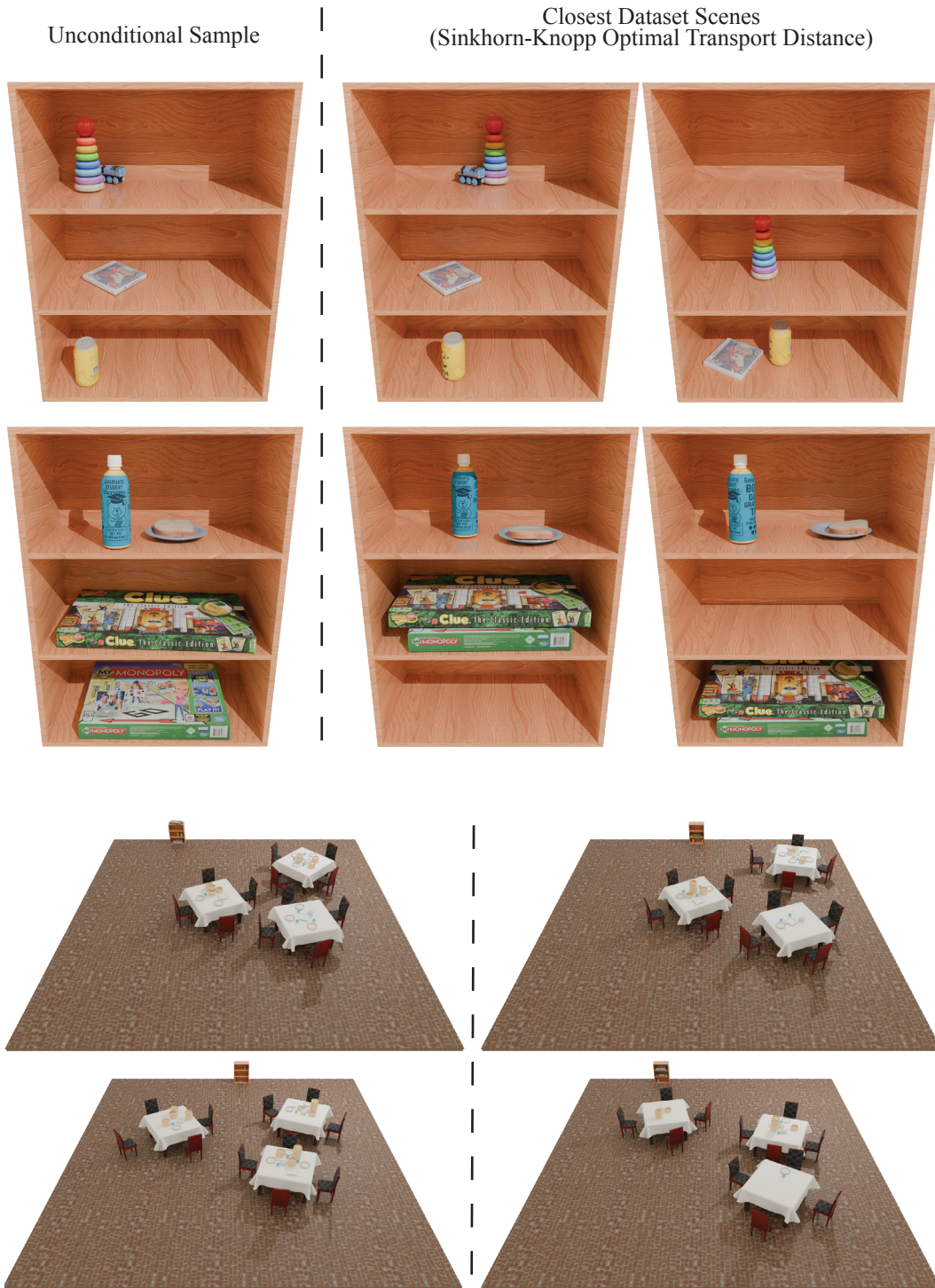


Figure 26: Unconditional samples and their closest dataset scenes for models trained on the Living Room Shelf and Restaurant (High-Clutter) datasets. Notice that the training scenes are quite distinct from the generated scenes, indicating that our models learn the distribution rather than memorize the training data.

Table 9: Training objective ablation. Unconditional generation results on Living Room Shelf and Restaurant (High-Clutter). “Mixed” refers to our standard model, and “DDPM” refers to our model trained with the continuous-only objective from [3].

Method	Living Room Shelf				Restaurant (High-Clutter Variant)			
	CA (25 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓	CA (50 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓
DDPM	68.48 \pm 1.65	3.96	2.13	0.03	80.75 \pm 5.54	0.88	1.42	13.59
Mixed	52.84 \pm 1.26	2.13	2.09	0.02	70.74 \pm 8.05	0.87	1.31	6.31

Table 10: Rotation representation ablation. Unconditional generation results on Breakfast Table (Low Clutter) and Living Room Shelf. \mathbb{R}^9 +SVD refers to the rotation representation from [20] that is used by our method.

Method	Breakfast Table (Low-Clutter Variant)				Living Room Shelf			
	CA (25 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓	CA (100 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓
Axis-Angle	66.57 \pm 6.97	1.25	2.51	3.25	69.91 \pm 3.90	2.61	2.20	0.03
Quaternion	61.38 \pm 7.64	1.00	2.46	3.31	61.58 \pm 4.61	2.53	2.10	0.03
\mathbb{R}^9 +SVD	57.85 \pm 4.98	1.67	2.39	3.50	52.84 \pm 1.26	2.13	2.09	0.02

E.4 Post Processing Pipeline

We qualitatively ablate the effects of our post-processing pipeline in Figure 27, comparing three variants: (1) no post-processing, (2) simulation only, and (3) our full pipeline with projection followed by simulation. The left column shows that unprocessed samples often contain physical violations, such as interpenetrations or unsupported objects. The middle column demonstrates that simulation alone can resolve some issues, but may also destabilize the scene. In contrast, our full pipeline (right column) eliminates interpenetrations while preserving the intended structure, resulting in physically stable and realistic scenes. These results highlight the importance of combining geometric projection and physics simulation to ensure feasibility. To make the comparison informative, we selected examples with prominent initial physical violations.

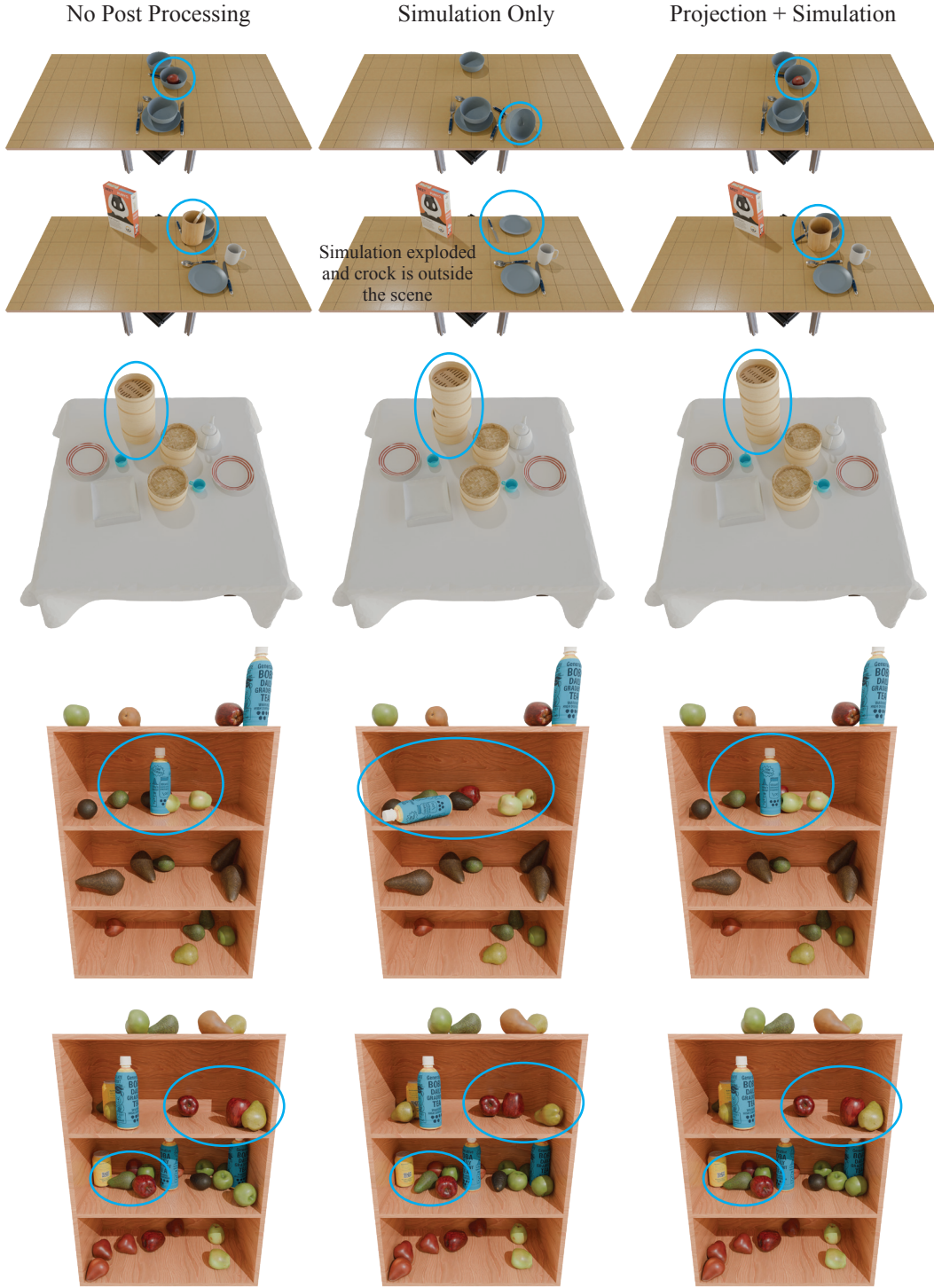


Figure 27: **Qualitative post processing ablation.** Interesting regions are highlighted with blue ellipses. The left column contains samples generated by the model without post processing. The middle column contains samples from the left after applying physics simulation only. The column on the right contains samples from the left after applying our complete post processing (non-penetration projection followed by physics simulation). Note that samples in the left column are not physically feasible. Samples in the other columns are physically feasible, but the samples in the right column are closer to the desired scenes from the left column.

Table 11: Training precision ablation. Unconditional generation results on Restaurant (High-Clutter) and Living Room Shelf. “ft32” uses full precision (32-bit) with `matmul_precision="high"`, while “bfloat” uses mixed precision (bf16) with `matmul_precision="medium"`. Evaluation is conducted in the training precision.

Precision	Restaurant (High-Clutter)				Living Room Shelf			
	CA (50 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓	CA (100 it, %) ↓	KL ($\times 10^4$) ↓	FID ↓	MTP (cm) ↓
bfloat	90.32 \pm 1.68	0.49	1.65	9.44	88.46 \pm 0.35	2.21	2.18	0.11
ft32	70.74 \pm 8.05	0.87	1.31	6.31	52.84 \pm 1.26	2.13	2.10	0.02

References

- [1] H. Fu, B. Cai, L. Gao, L.-X. Zhang, J. Wang, C. Li, Q. Zeng, C. Sun, R. Jia, B. 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.
- [2] D. Paschalidou, A. Kar, M. Shugrina, K. Kreis, A. Geiger, and S. Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [3] J. Tang, Y. Nie, L. Markhasin, A. Dai, J. Thies, and M. Nießner. Diffuscene: Denoising diffusion models for generative indoor scene synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.
- [4] S. Hu, D. M. Arroyo, S. Debats, F. Manhardt, L. Carlone, and F. Tombari. Mixed diffusion for 3d indoor scene synthesis. *arXiv preprint: 2405.21066*, 2024.
- [5] G. Izatt and R. Tedrake. *Capturing Distributions over Worlds for Robotics with Spatial Scene Grammars*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 2022. URL <https://dspace.mit.edu/handle/1721.1/144763>.
- [6] B. F. Labs. Flux. <https://github.com/black-forest-labs/flux>, 2024.
- [7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL <https://arxiv.org/abs/1810.04805>.
- [8] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results, 2018. URL <https://arxiv.org/abs/1703.01780>.
- [9] I. Loshchilov and F. Hutter. Decoupled weight decay regularization, 2019. URL <https://arxiv.org/abs/1711.05101>.
- [10] P. E. Gill, W. Murray, and M. A. Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002. doi: 10.1137/S1052623499350013. URL <https://doi.org/10.1137/S1052623499350013>.
- [11] K. Black, M. Janner, Y. Du, I. Kostrikov, and S. Levine. Training diffusion models with reinforcement learning, 2023.
- [12] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models, 2022. URL <https://arxiv.org/abs/2010.02502>.
- [13] Y. Zhang, E. Tzeng, Y. Du, and D. Kislyuk. Large-scale reinforcement learning for diffusion models, 2024. URL <https://arxiv.org/abs/2401.12244>.
- [14] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20*, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.
- [15] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. van den Berg. Structured denoising diffusion models in discrete state-spaces, 2023. URL <https://arxiv.org/abs/2107.03006>.
- [16] J. Ho and T. Salimans. Classifier-free diffusion guidance, 2022. URL <https://arxiv.org/abs/2207.12598>.
- [17] R. Tedrake and the Drake Development Team. Drake: Model-based design and verification for robotics, 2019.

- [18] X. Wang, C. Yeshwanth, and M. Nießner. Sceneformer: Indoor scene generation with transformers. In *2021 International Conference on 3D Vision (3DV)*, pages 106–115, 2021. doi: [10.1109/3DV53792.2021.00021](https://doi.org/10.1109/3DV53792.2021.00021).
- [19] M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf.
- [20] A. R. Geist, J. Frey, M. Zhobro, A. Levina, and G. Martius. Learning with 3D rotations, a hitchhiker’s guide to $SO(3)$. In R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 15331–15350. PMLR, 21–27 Jul 2024. URL <https://proceedings.mlr.press/v235/geist24a.html>.