

SceneFactor: Factored Latent 3D Diffusion for Controllable 3D Scene Generation

Supplementary Material

In this supplemental material, we provide details of data processing and caption generation in Section 6, show the additional qualitative and quantitative comparison to diffusion- and non-diffusion-based methods in Section 7, provide details of the evaluation metrics and the perceptual study in Section 8 and additional implementation details in Section 9.

6. Data Processing

Geometry. To make 3D-FRONT [22] data suitable for training and testing, we first combine 3D furniture and 3D scene meshes using 3D-FRONT annotation. 3D-FUTURE [23] models are preliminarily converted into high-quality watertight meshes using the Manifold [30] approach. This method can create meshes with double surfaces, so we remove all closed surfaces that lie within a mesh interior. To obtain the unsigned distance field of 3D-FRONT scenes with a resolution of 4.2 cm, we apply the virtual scanning tool mesh2sdf [69]. Preliminarily, we remove the ceiling from all 3D-FRONT scenes. In addition to the distance field, we regularly sample points with cor-

responding semantic labels belonging to scene layouts and furniture objects to form a semantic map of a scene with a resolution of 16.8 cm. The training chunks are obtained by randomly cropping from scene distance fields and semantic maps. We convert all test scenes into a test-suitable format by cutting the scenes into a regular grid of overlapping geometric and semantic chunks. All scene chunks are normalized to be centered at the origin and scaled to a unit cube.

Captions. To obtain captions for scene chunks, we use the 3D-FRONT object annotations to automatically generate seven types of captions. These caption types include descriptions with object counts or object lists without counts, subcategory information, and spatial relationships between objects. First, every scene annotation includes object instances of 8 categories depicted in Fig. 4. For every chunk, we add names of object categories into a caption if at least 35% of an object lies within a chunk. Here, we have two types of text captions: explicit lists of single objects as category names and aggregated lists where repeated objects are counted. Another caption type can be obtained from

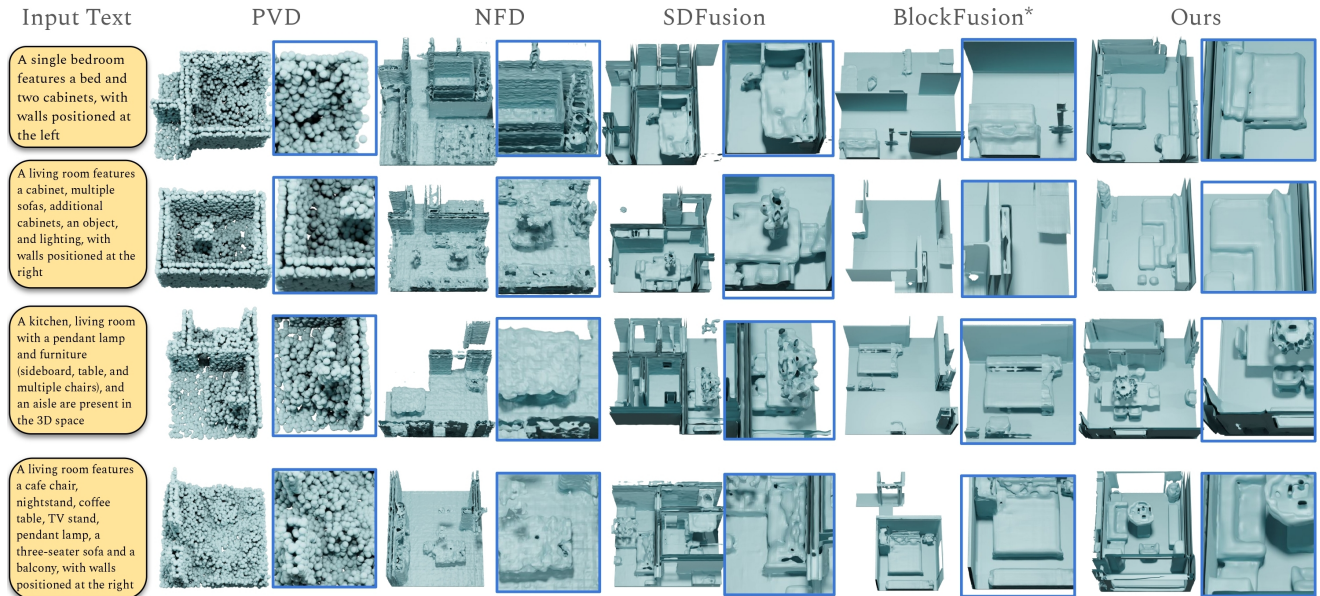


Figure 7. Qualitative comparison with state of the art on text-guided scene chunk generation using Qwen1.5 captions. In comparison with PVD [86], NFD [58], SDFusion [13], and BlockFusion [73] SceneFactor generates higher-fidelity, more coherent scene structures through our factored approach.

*Note that results for BlockFusion are generated unconditionally

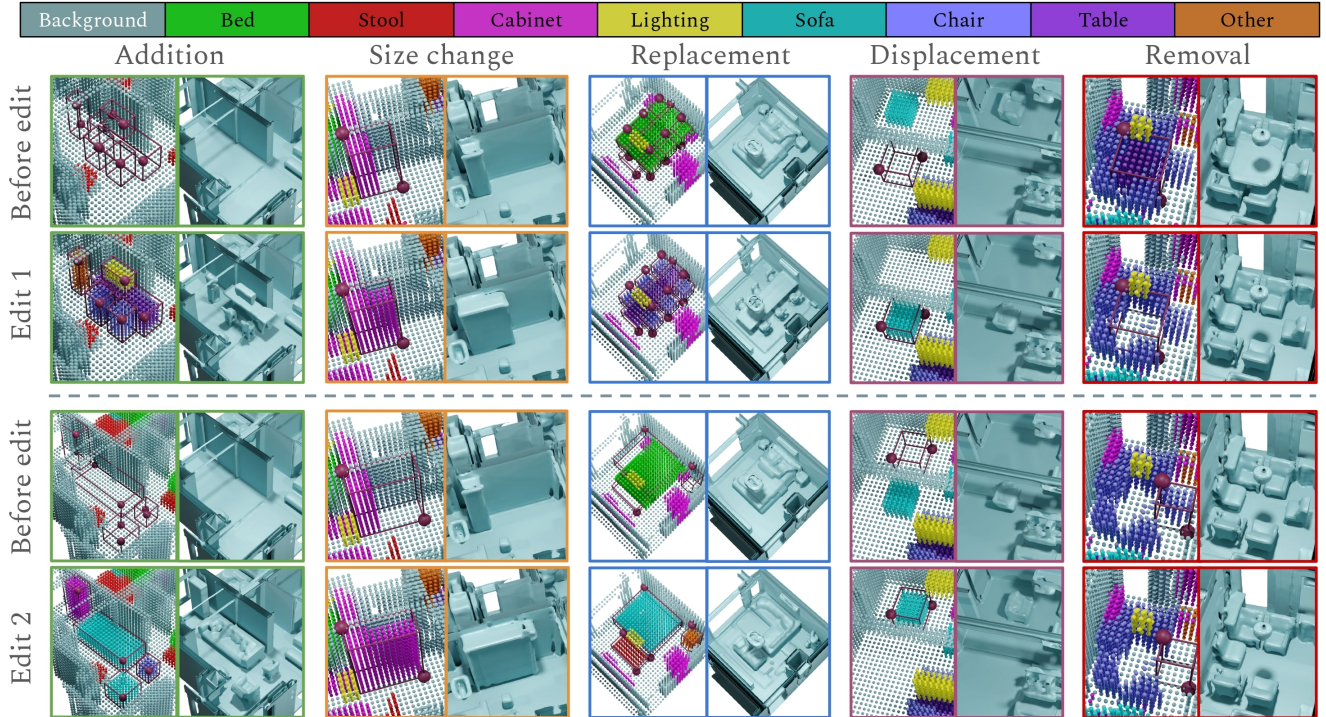


Figure 8. Additional qualitative scene editing results. Generated scenes and their corresponding semantic maps are shown in the top row, and two alternatives for each object synthesis-based edit are shown below.

the latter by adding spatial relationships between objects in a chunk. Second, using simple proximity checks based on Euclidean L_2 distance between object centers or object centers and wall points, we can identify if two or more objects form a group, stand across from each other, or stand next to a wall. For every caption, we also identify if there are walls along the borders of chunks. These three types of captions can be augmented using 33 subcategory names from 3D-FRONT annotation instead of category names. Finally, we have an extra room type caption, where for every chunk, we add room names from 3D-FRONT annotation to a caption if at least 25% of a room lies in a chunk.

LLM-Refined Captions. Finally, we train additional instances of SceneFactor, SDFusion [13], NFD [58], PVD [86] with the second set of captions – complex, natural text inputs. We utilize the large-language model Qwen1.5 [64] to refine our synthetic-looking captions using the following query: Reformulate the following synthetic description of a 3D scene into a human-readable but concise, extremely minimalistic, and non-list format in only one sentence: <caption>, where <caption> is the caption before LLM refinement.

Augmentations. During the training of the geometric and

semantic VQ-VAE autoencoders and diffusion models, random 90°-fold rotation and symmetric reflection across xz - or yz -plane augmentations are applied to all train scene chunks and input latent representations.

7. Additional Results

Additional Comparison to Diffusion-based Methods.

Fig. 7, 11, 12, 13, 14, 15 and 16 show additional qualitative comparisons with state-of-the-art baselines on scene chunk generation using synthetic and Qwen-refined captions. PVD [86] model uses explicit point cloud diffusion, which makes it significantly harder to generate clean and complete scenes. NFD [58] produces much cleaner scene layouts due to its signed distance field prediction. However, objects tend to lack details, with various low-level geometric artifacts due to the lack of structured latent space for generation. SDFusion [13] can generate more recognizable furniture. Nonetheless, due to direct text-to-geometry prediction and the absence of convolutional attention, SDFusion tends to generate more incoherent global structures (e.g., objects penetrating each other and inconsistent walls). Finally, BlockFusion [73] unconditional generations contain inconsistent wall structures, and triplane-based generation is unable to produce accurate furniture objects in arbi-

trary chunk locations. In Tab. 9, 10, we provide the quantitative evaluation of our method and baseline approaches for the geometric quality and text-guided generation using Qwen1.5 captions as input. We additionally showcase qualitative results of scene generation with SceneFactor in Fig. 17, 18, 19, 20 and 21.

Figs. 9 and 10 show additional qualitative comparisons for 3D scene generation with SDFusion [13] and BlockFusion [73]. SDFusion tends to produce more noticeable transitions between generated chunks, along with floating geometric artifacts and holes in furniture objects. Both SDFusion and BlockFusion generate significant artifacts, such as holes in the floor, due to the lack of conditioning on spatial information. BlockFusion struggles to outpaint objects from one chunk to the next chunk, which results in a significantly unnatural appearance of the generated room spaces.

Finally, we provide additional qualitative scene editing results for our method in Fig. 8. Our approach is able to produce diverse and consistent editing results for the same input scene.

Comparison to Non-diffusion-based Methods. In addition, we provide a comparison to 2D diffusion lifting-based approach Text2Room [29] and a retrieval-based method ATISS [48], for which we evaluate only independent chunks generation since these models are not applicable for large-scale scene generation.

Tab. 5 quantitatively evaluates the geometric quality of generated chunks against ATISS and Text2Room. Text2Room [29] takes significant time to generate one chunk (~ 3.5 hours); therefore, we limited the evaluation of this approach to 92 chunks. Our factored approach produces consistently improved geometry in comparison with these baselines. In Tab. 7, we also show that our approach significantly outperforms Text2Room by the CLIP score between rendered chunks and input text captions. We do not evaluate against the retrieval-based ATISS method because the CLIP score is biased towards non-generated but retrieved synthetic meshes placed on top of the floor. We found this comparison not meaningful. Instead, we evaluate our approach against ATISS using a pretrained neural listener model for the input text correspondence in Tab. 8. A neural evaluator is trained to distinguish the target chunk from a distracting chunk, given the text description. Given two chunks from different methods or one chunk from a method and another chunk from a GT set, the neural evaluator provides a confidence score for each of them based on the binary classification logits. If the absolute difference between two confidence scores ≤ 0.2 , we consider the comparison to be confused. ATISS is not able to handle a large diversity of text captions and is significantly inferior to our approach in terms of text coherence.

Additional Semantic Evaluation. We provide additional

analysis of our first-stage semantic map generation model, where the original latent diffusion model and the diffusion model explicitly trained with one-hot semantic maps are compared to each other. We first compute the average chunk semantic accuracy with respect to text input, where for every object class category mentioned in a caption, we check if the corresponding object has been predicted. For this metric, the latent-based model has accuracy of **91%** against 83% for the model without latent representation. In Tab. 4, we provide further evaluation, which is based on MMD/COV/1-NNA metrics.

8. Baseline Evaluation Setup

Metrics. Following the works for 3D shape generation, we use the following metrics on point clouds extracted from mesh surfaces:

$$\text{MMD}(S_g, S_r) = \frac{1}{|S_r|} \sum_{Y \in S_r} \min_{X \in S_g} D(X, Y),$$

$$\text{COV}(S_g, S_r) = \frac{|\{\text{argmin}_{Y \in S_r} D(X, Y) | X \in S_g\}|}{|S_r|},$$

$$1\text{-NNA}(S_g, S_r) = \frac{\sum_{X \in S_g} \mathbf{1}_X + \sum_{Y \in S_r} \mathbf{1}_Y}{|S_g| + |S_r|},$$

$$\mathbf{1}_X = \mathbf{1}[N_X \in S_g], \mathbf{1}_Y = \mathbf{1}[N_Y \in S_r],$$

where S_r and S_g are reference and generated sets of point clouds extracted from ground-truth and generated mesh surfaces, respectively, N_X is a point cloud that is closest to X in both generated and reference dataset, i.e., $N_X = \text{argmin}_{K \in S_r \cup S_g} D(X, K)$. We use Chamfer distance (CD) and Earth-mover distance (EMD) as $D(X, Y)$ to compute these metrics in 3D. To evaluate these metrics, we extract 4096 points from ground-truth and generated mesh surfaces or sample 4096 points from PVD point clouds.

We utilize the official implementations of NFD [58], PVD [86], SDFusion [13], BlockFusion [73], Text2Room [29], and ATISS [48]. For NFD and PVD, we do not implement the same or similar scene-aware generation mechanism, which inpaints missing chunks because PVD leverages the explicit point cloud representation in the diffusion model, and NFD demonstrates extremely poor results when using an inpainting mechanism resulting in empty chunks which degrade in quality along the generation sequence. Text2Room and ATISS approaches are also inapplicable for large-scale scene generation using the outpainting mechanism. We use the same context encoding for text captions as in SceneFactor and SDFusion for NFD

and PVD, while Text2Room, ATISS, and BlockFusion are designed to take text as input.

To evaluate geometric quality in Tab. 1, we normalize the ground-truth and predicted chunk meshes or point clouds into a unit cube and extract 4096 points from mesh surface or point cloud.

For the text-aware evaluation in Tab. 2, we train the neural listener model consisting of geometric encoder, text embedded, and language encoder. Geometric encoder consists of 5 ResNet blocks with GeLU activations and 2 linear layers with ReLU activations and takes uDF of geometric chunks as input. The input text is encoded using the same text encoder as in SceneFactor, but with an embedding dimension of 128. The text features are then processed using the LSTM [28] network. The resulting features are concatenated with geometric features and finally processed with a shallow MLP network with ReLU activations.

We also evaluate using CLIP [52] score, which reflects the consistency of generated geometry to text inputs in CLIP space. For the CLIP score evaluation in Tab. 6, we render 4 views of predicted meshes or point clouds and compute the cosine distance to text caption used for generation. We add a prefix 'a render of a 3D scene with ' to captions for CLIP score evaluation for ones not generated with Qwen1.5 [64] model.

Perceptual Study. To more effectively capture the perceptual quality of synthesized geometry, as well as adherence to text and editing inputs, we perform a perceptual study. We ask users to evaluate perceptual geometric quality as well as adherence to the text prompts, both as unary evaluation scores and binary comparisons between SceneFactor and each baseline. Perceptual geometric quality is assessed on a scale from 1 (Awful quality) to 5 (Great quality). Adherence to text input is assessed on a scale from 1 (Not matching) to 5 (Matching).

In particular, since we lack ground truth editing results as well as baselines that perform local spatial edits, we evaluate our editing performance through unary evaluation in the perceptual study. Editing results in the perceptual study are generated randomly across each possible editing operation. We ask users to assess (1) if the resulting edited scene is consistent with the given edit operation using a scale from 1 to 5; (2) the perceptual geometric quality of an edited scene using a scale from 1 to 5; and (3) if a scene remained unchanged outside of the editing region as either 1 (Yes) or 2 (No). In total, 21 participants took part in a perceptual study consisting of 53 questions per user. We provide the quantitative results of the conducted perceptual study in Fig. 6.

We developed a Django-based web application for the perceptual study. In total, we have 5 sections for our survey. For the first part, an unary study on perceptual geometric quality and text consistency for generated chunks and scenes, there are 25 questions and 5 randomly chosen

scenes and chunks for every approach. Here, the user is asked to provide a score from 1 to 5 based on the perceptual geometric quality of chunks and the consistency of generation to an input text caption. In addition to chunkwise comparison, for SDFusion, BlockFusion, and our approach, there is also a unary study on scenes, where users are asked to evaluate the geometric quality of the whole scene. For SDFusion and ours, users are asked to evaluate the consistency of one scene chunk to a text caption.

9. Implementation Details

Our method is implemented using PyTorch. Semantic and geometric VQ-VAE models are trained with an Adam [34] optimizer with learning rates $1e-4$ and $2e-4$ for the semantic and geometric VQ-VAEs. We use AdamW [41] with a learning rate of $1e-5$ for both semantic and geometric latent diffusion models. The semantic and geometric VQ-VAEs are trained on 2 NVIDIA A6000s each for 320k and 160k iterations (~ 50 hours) until convergence. The diffusion models are trained on 2 NVIDIA A100s each for 400k iterations (~ 100 and 150 hours, respectively).

VQ-VAE semantic and geometric models comprise 3 ResNet blocks in the encoder and 3 ResNet blocks in the decoder with bilinear upsampling layers and GeLU [25] nonlinearities. For the semantic VQ-VAE latent space, we encode semantic chunks into $(1, 4, 4, 4)$ latent grids, with only 1 feature channel using a dictionary size of 8192. Geometric chunks are encoded using the geometric VQ-VAE model into $(1, 16, 16, 16)$ latent grids, with 1 feature channel using a dictionary size of 32768.

The semantic diffusion model is trained using larger latent grids of size $(1, 8, 4, 8)$ that correspond to two twice bigger semantic chunks in both horizontal dimensions. We pad these grids with zeros to the shape of $(1, 8, 8, 8)$ to enable compression using 4 ResNet blocks in the encoder of the UNet model. The first 3 ResNet blocks combine convolutional operations with attention layers with 8 heads. To encode the context, we use the transformer-based model with BERT tokenizer and context dimension of 1280 and 77 maximum number of tokens.

Analogously, the geometric diffusion model is trained using larger latent grids of size $(1, 32, 16, 32)$ that correspond to two twice bigger geometric chunks in both horizontal dimensions. The UNet model encoder consists of 3 ResNet blocks with attention layers with 8 heads in each block. To encode the semantic context, we first encode the input semantic chunk of size $(1, 32, 16, 32)$ into one-hot representation with 10 class channels. This one-hot representation is encoded into a context feature grid of size $(128, 16, 8, 16)$ using the fully convolutional network with LeakyReLU activations [74].

A large, generated geometric latent grid can be decoded in batches or as a whole. Since the latent grid was generated

Method	Independent chunks					
	MMD ↓		COV ↑		1-NNA (0.5)	
	CD	EMD	CD	EMD	CD	EMD
w/o latent	0.263	0.473	0.335	0.344	0.784	0.784
Ours	0.222	0.458	0.495	0.491	0.598	0.631

Table 4. Semantic quality of synthesized 3D scene geometry as independent chunks.

with consistent outpainting, we do not generate noticeable seams when decoding the full grid as chunks in a minibatch. A minibatch of 32 latent chunks occupies <1 MB of memory; the peak memory allocation when processing this batch is ~11 GB. Alternatively, we can decode the full grid of the size 4×6 chunks, corresponding to 10.8 m×16.2 m, which occupies alone <1 MB of memory and the peak memory of ~75 GB when processed using an NVIDIA A100 with 80 GB of memory. Empirically, both strategies provide the same visual results.

Method	Independent chunks					
	MMD ↓		COV ↑		1-NNA (0.5)	
	CD	EMD	CD	EMD	CD	EMD
Text2Room	0.048	0.316	0.021	0.021	0.997	0.993
ATISS [48]	0.050	0.327	0.117	0.117	0.993	0.992
Ours	0.019	0.140	0.421	0.316	0.738	0.512

Table 5. Geometric quality of synthesized 3D scene geometry as independent chunks (left) and as chunks of outpainted 3D scenes (right).

Method	Independent chunks	Scene chunks
NFD [58]	26.59	26.59
PVD [86]	24.79	24.79
SDFusion [13]	28.01	27.70
Ours	29.81	29.40

Table 6. CLIP-Score evaluation of text-guided generation. Rendered views of chunks generated by our method better match text captions.

Method	Independent chunks	Scene chunks
Text2Room [29]	24.11	24.11
Ours	29.81	29.40

Table 7. CLIP-Score evaluation of text-guided generation. Rendered views of chunks generated by our method better match text captions.

Target (Tr)	Distractor (Dis)	P(Tr)	P(Dis)	P(conf.)	P(Dis=GT) - P(Tr) ↓
Ours	ATISS [48]	66%	34%	21%	-
ATISS [48]	GT	33%	67%	22%	34%
Ours	GT	42%	58%	33%	16%

Table 8. Quality of text-guided generation using a pretrained neural listener model. Our results are preferred over that of SDFusion [13], ATISS [48], and Text2Room [29], both in direct comparison as well as relative to ground truth.

Method	Independent chunks						Scene chunks					
	MMD ↓		COV ↑		1-NNA (0.5)		MMD ↓		COV ↑		1-NNA (0.5)	
	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD
NFD [58]	0.023	0.225	0.411	0.335	0.744	0.814	-	-	-	-	-	-
PVD [86]	0.021	0.221	0.396	0.285	0.729	0.876	-	-	-	-	-	-
SDFusion [13]	0.031	0.240	0.331	0.277	0.835	0.898	0.035	0.253	0.313	0.265	0.874	0.910
BlockFusion* [73]	0.048	0.305	0.177	0.110	0.953	0.986	0.054	0.330	0.186	0.091	0.961	0.993
Ours	0.021	0.165	0.399	0.300	0.772	0.769	0.026	0.249	0.365	0.270	0.839	0.910

Table 9. Geometric quality of synthesized 3D scene geometry as independent chunks (left) and as chunks of outpainted 3D scenes (right) generated with Qwen1.5 captions.

Method	Independent chunks	Scene chunks
NFD [58]	21.61	21.61
PVD [86]	20.32	20.32
SDFusion [13]	23.08	23.17
Ours	23.96	23.79

Table 10. CLIP-Score evaluation of text-guided generation using Qwen1.5 captions. Rendered views of chunks generated by our method better match text captions.

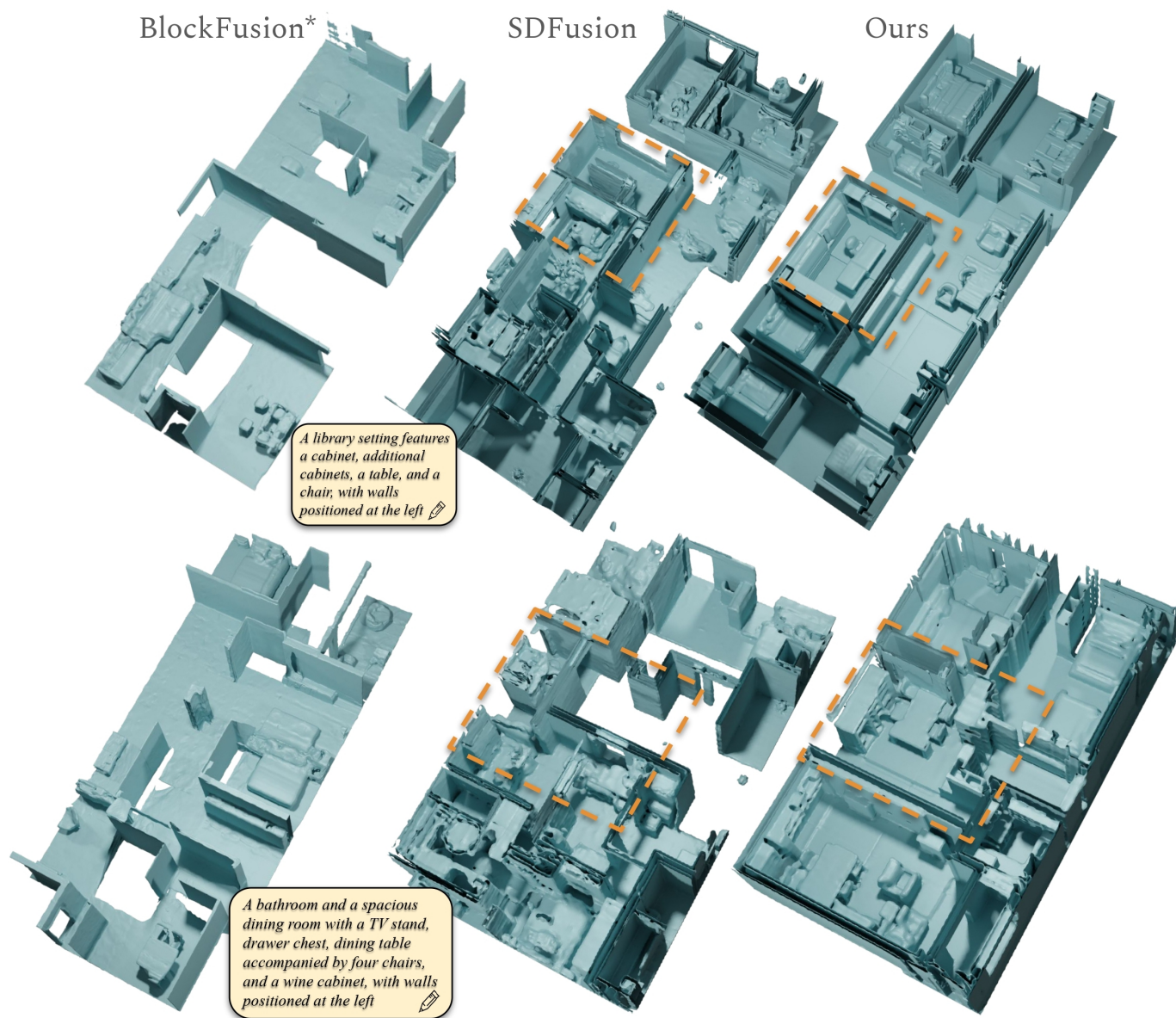


Figure 9. Additional qualitative comparisons for scene generation in comparison with SDFusion [13] and BlockFusion [73].

*Note that results for BlockFusion are generated unconditionally

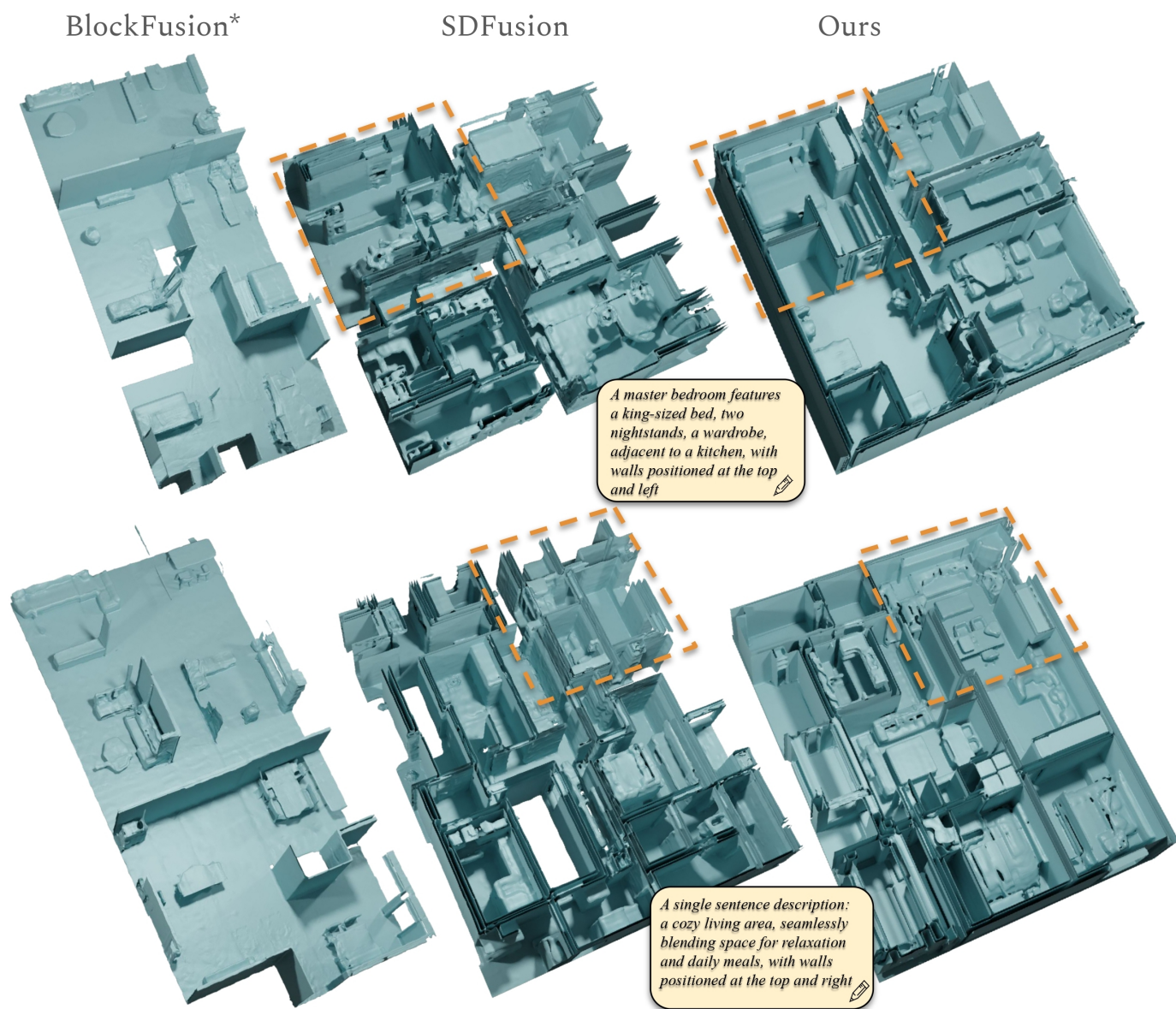


Figure 10. Additional qualitative comparisons for scene generation in comparison with SDFusion [13] and BlockFusion [73].

*Note that results for BlockFusion are generated unconditionally

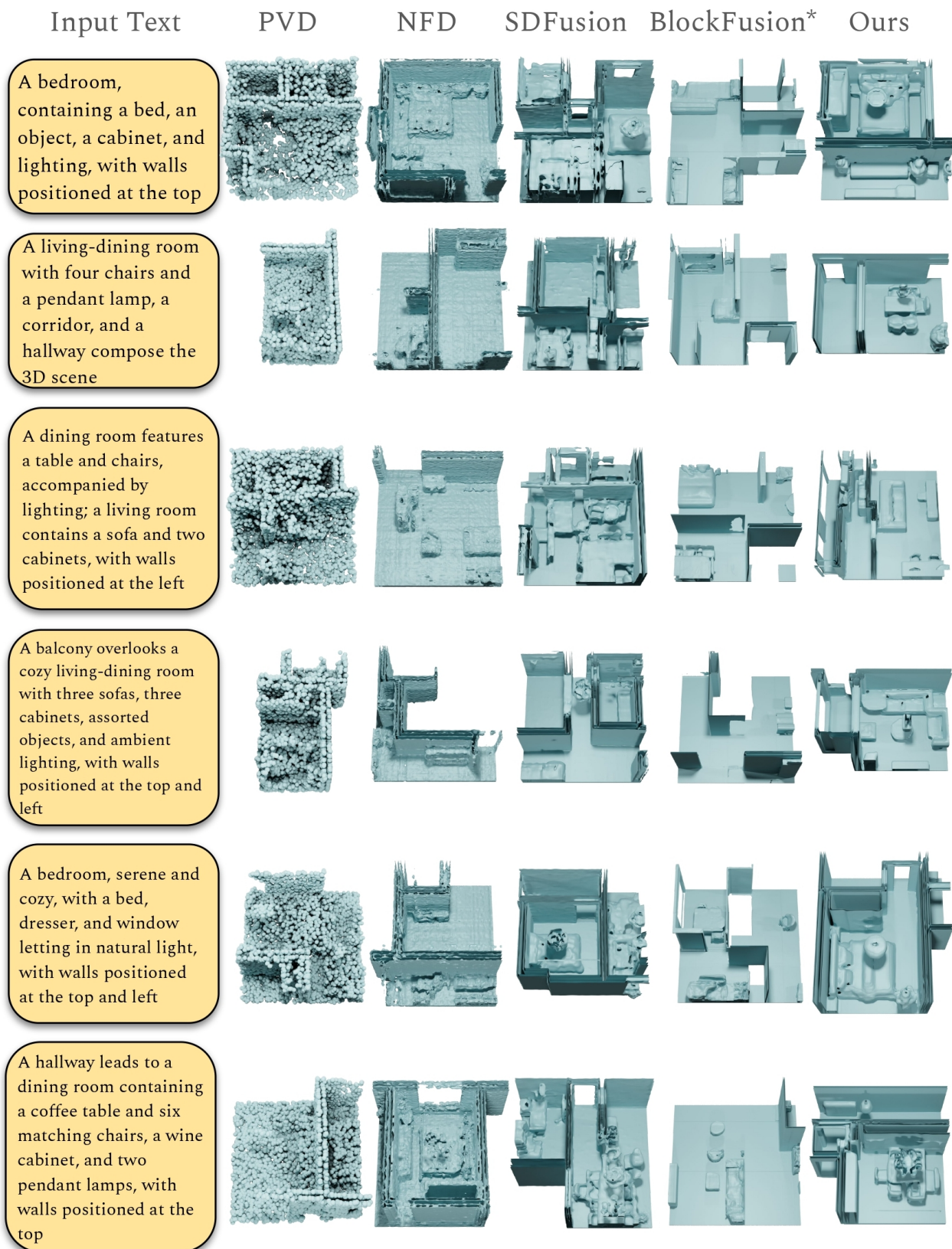


Figure 11. Additional qualitative comparisons to state-of-the-art diffusion-based 3D generative approaches PVD [86], NFD [58], SDFusion [13], and BlockFusion [73] using Qwen1.5 captions. Our approach produces sharper scene geometry and more coherent scene structure.

*Note that results for BlockFusion are generated unconditionally

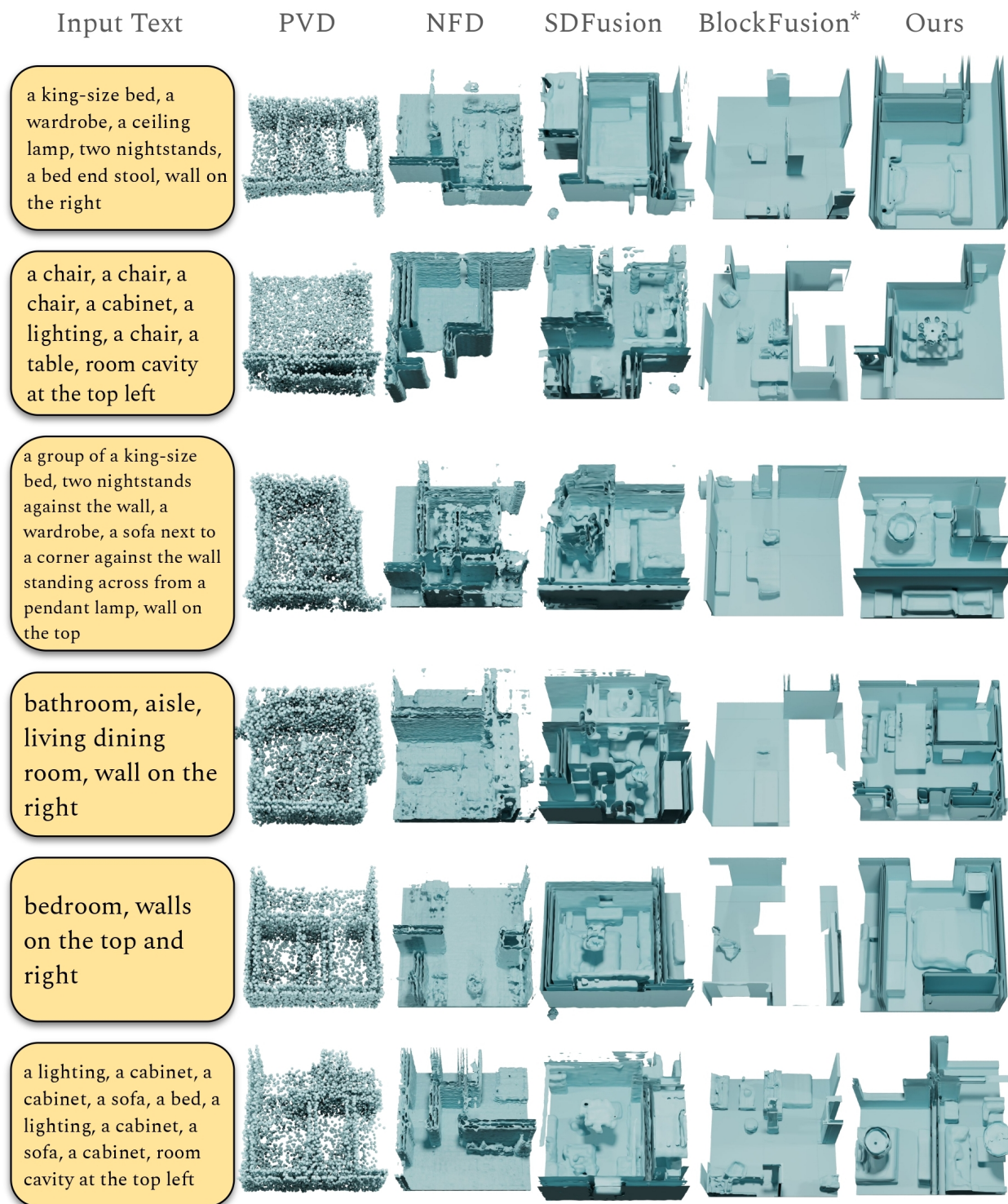


Figure 12. Additional qualitative comparisons to state-of-the-art diffusion-based 3D generative approaches PVD [86], NFD [58], SDFusion [13], and BlockFusion [73] using synthetic captions. Our approach produces sharper scene geometry and more coherent scene structure.

*Note that results for BlockFusion are generated unconditionally

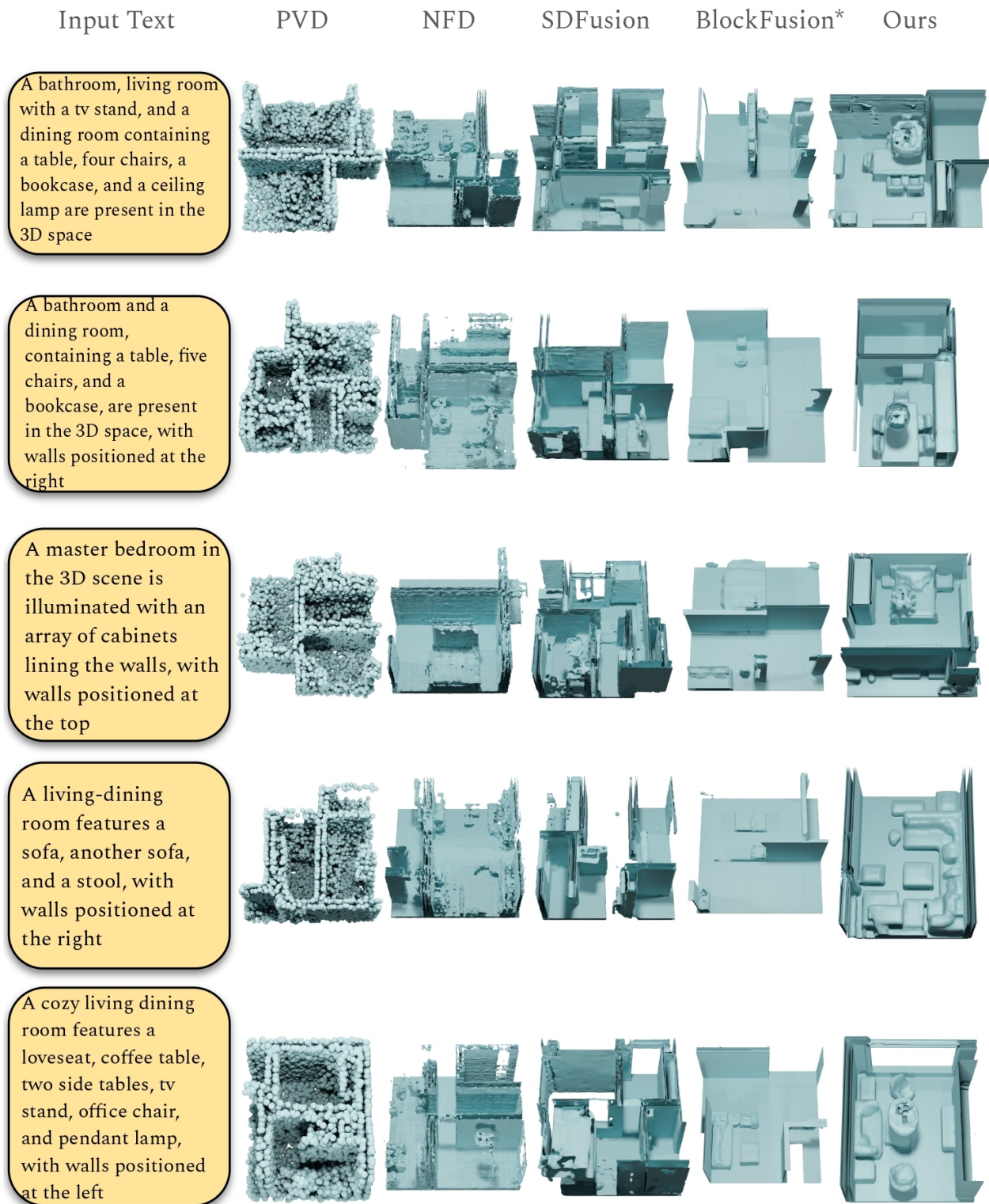


Figure 13. Additional qualitative comparisons to state-of-the-art diffusion-based 3D generative approaches PVD [86], NFD [58], SDFusion [13], and BlockFusion [73] using Qwen1.5 captions. Our approach produces sharper scene geometry and more coherent scene structure.

*Note that results for BlockFusion are generated unconditionally

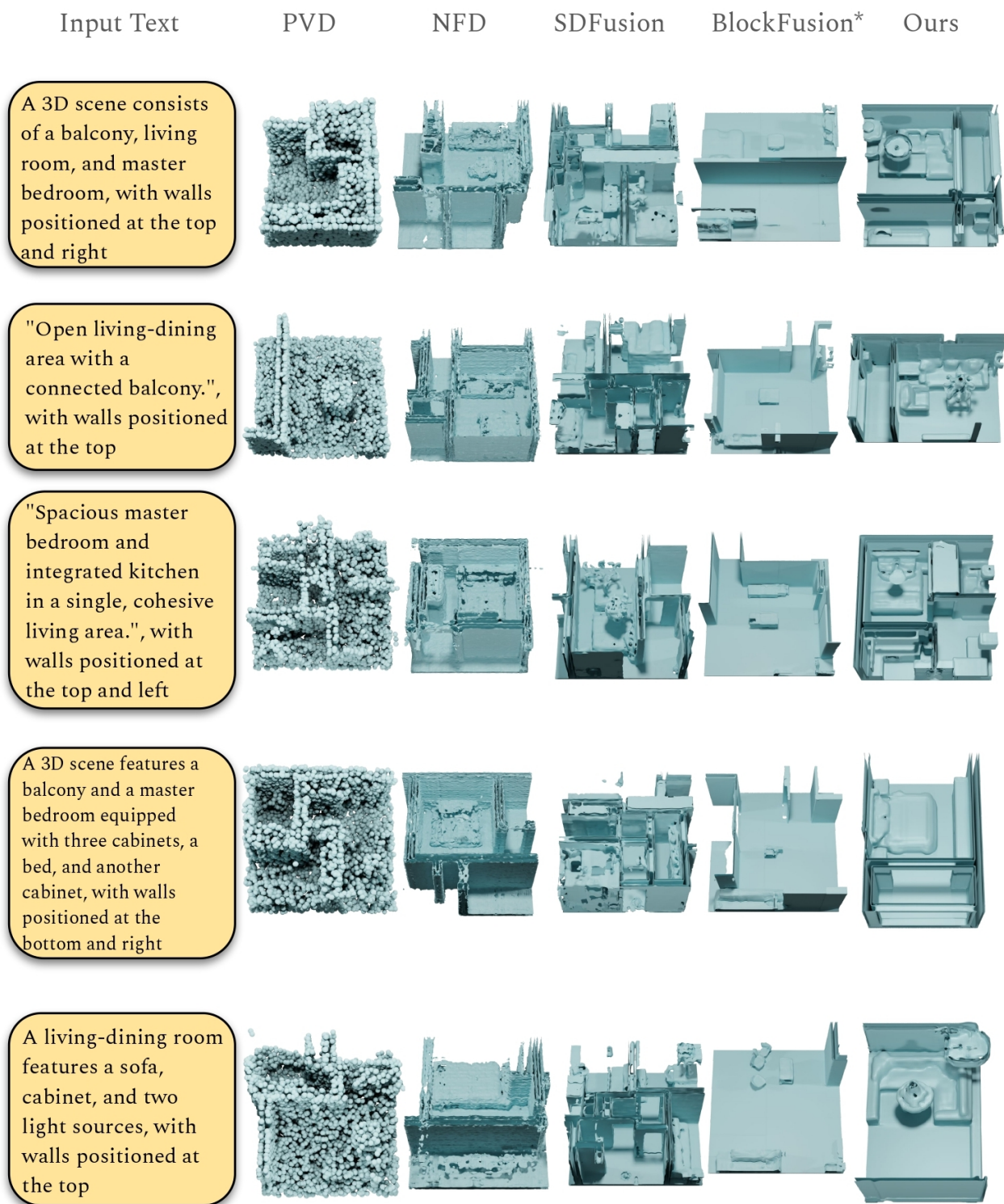


Figure 14. Additional qualitative comparisons to state-of-the-art diffusion-based 3D generative approaches PVD [86], NFD [58], SDFusion [13], and BlockFusion [73] using Qwen1.5 captions. Our approach produces sharper scene geometry and more coherent scene structure.

*Note that results for BlockFusion are generated unconditionally

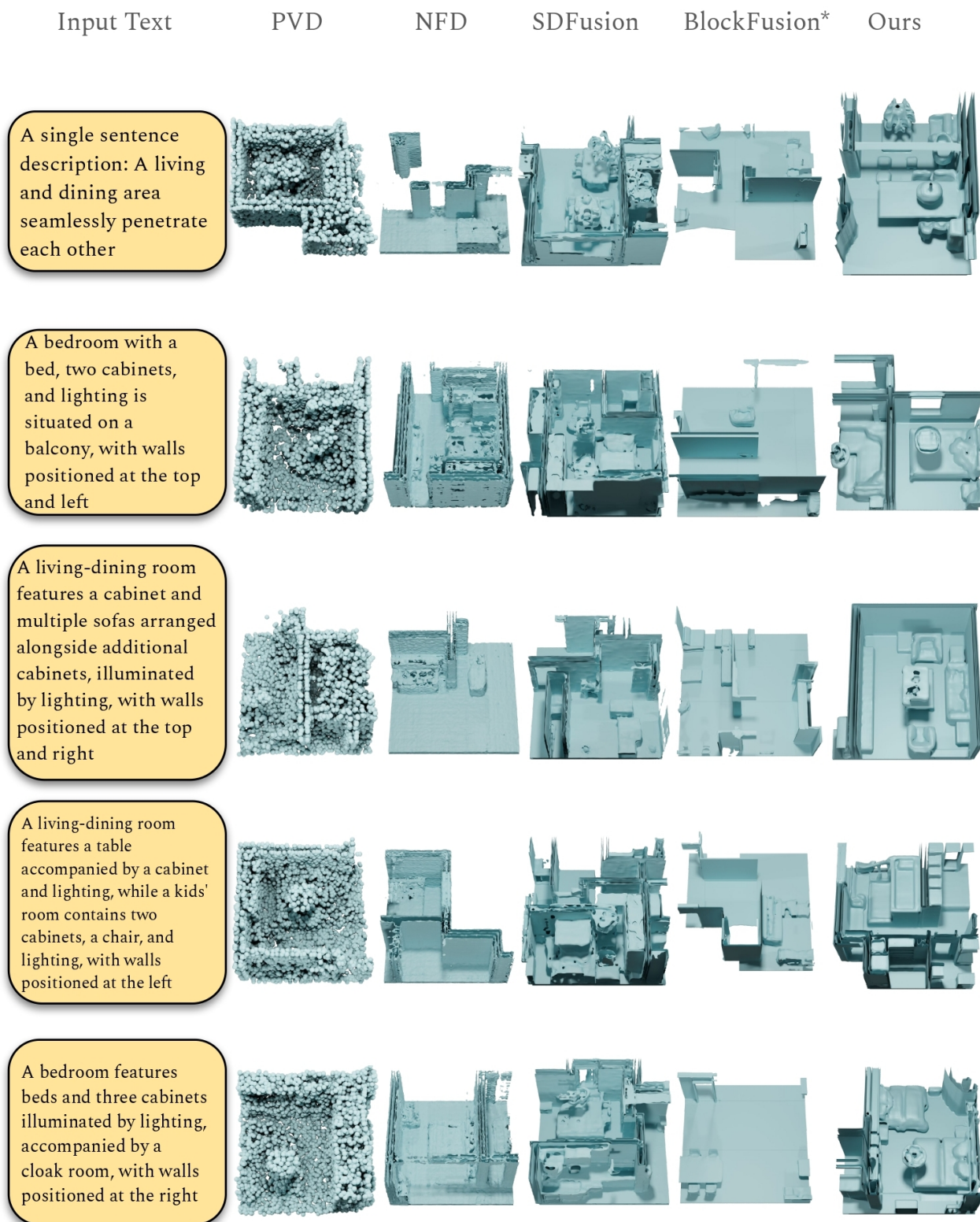


Figure 15. Additional qualitative comparisons to state-of-the-art diffusion-based 3D generative approaches PVD [86], NFD [58], SDFusion [13], and BlockFusion [73] using Qwen1.5 captions. Our approach produces sharper scene geometry and more coherent scene structure.

*Note that results for BlockFusion are generated unconditionally

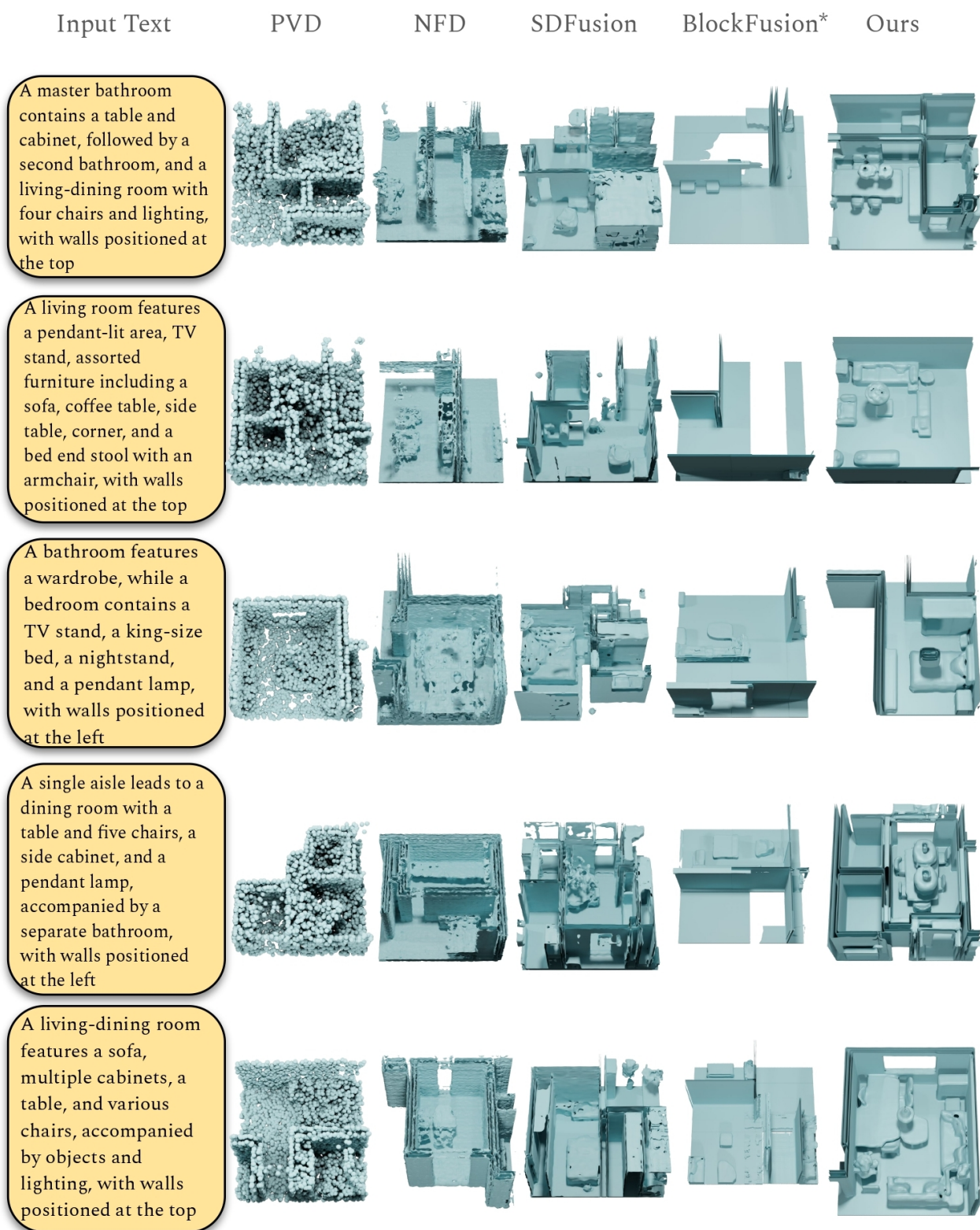


Figure 16. Additional qualitative comparisons to state-of-the-art diffusion-based 3D generative approaches PVD [86], NFD [58], SDFusion [13], and BlockFusion [73] using Qwen1.5 captions. Our approach produces sharper scene geometry and more coherent scene structure.

*Note that results for BlockFusion are generated unconditionally

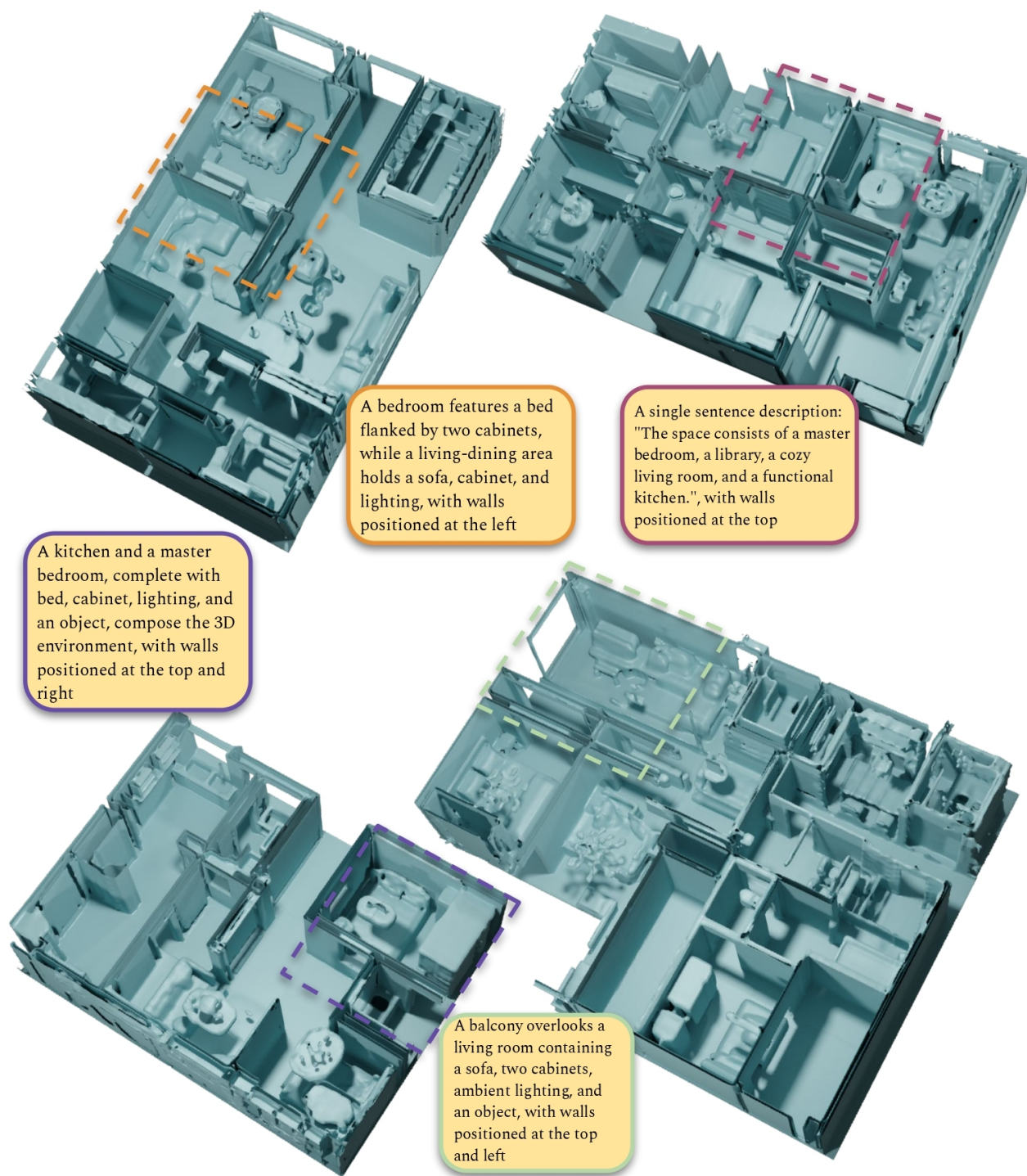


Figure 17. Additional qualitative results for scene generation with SceneFactor.

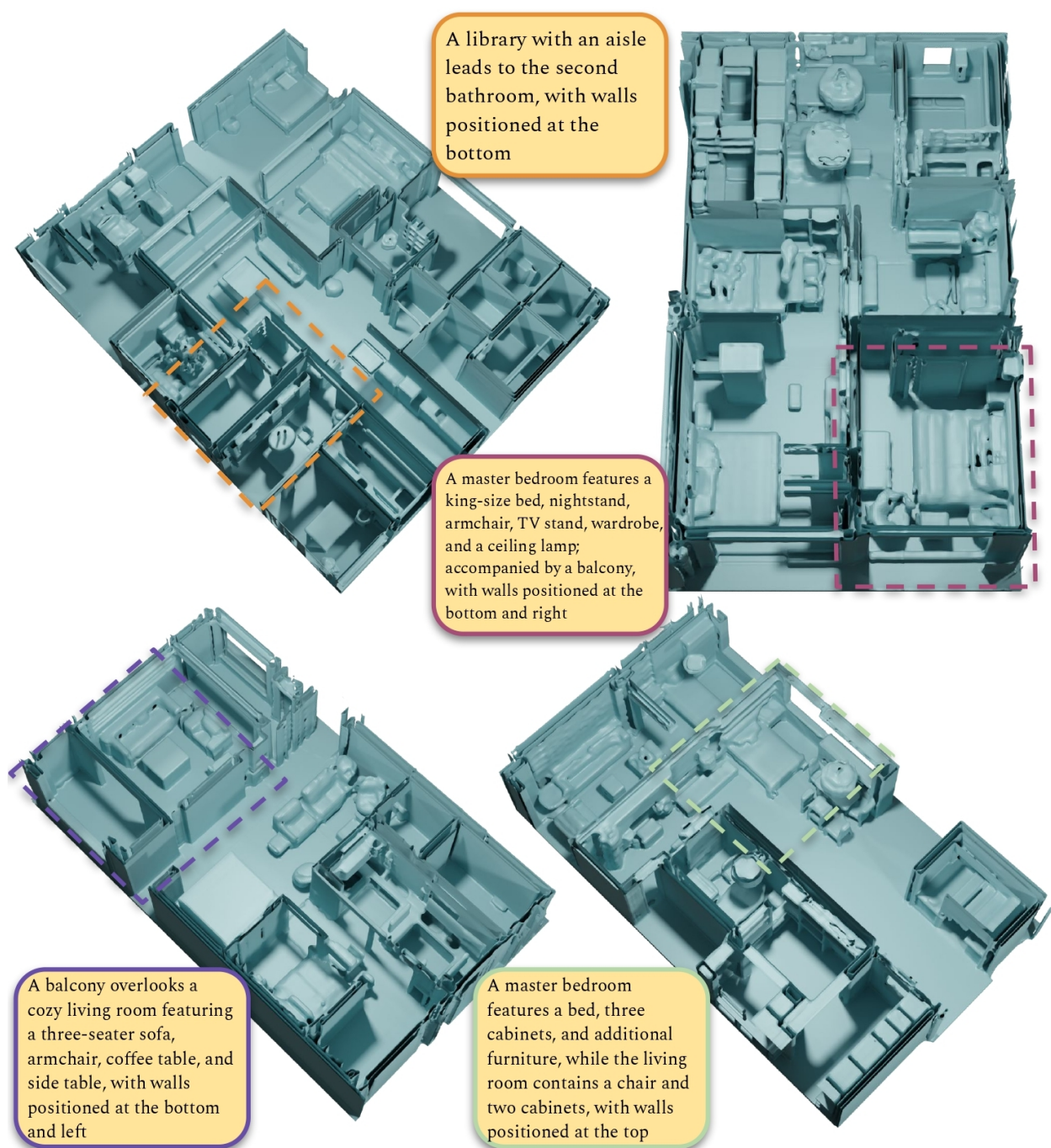


Figure 18. Additional qualitative results for scene generation with SceneFactor.

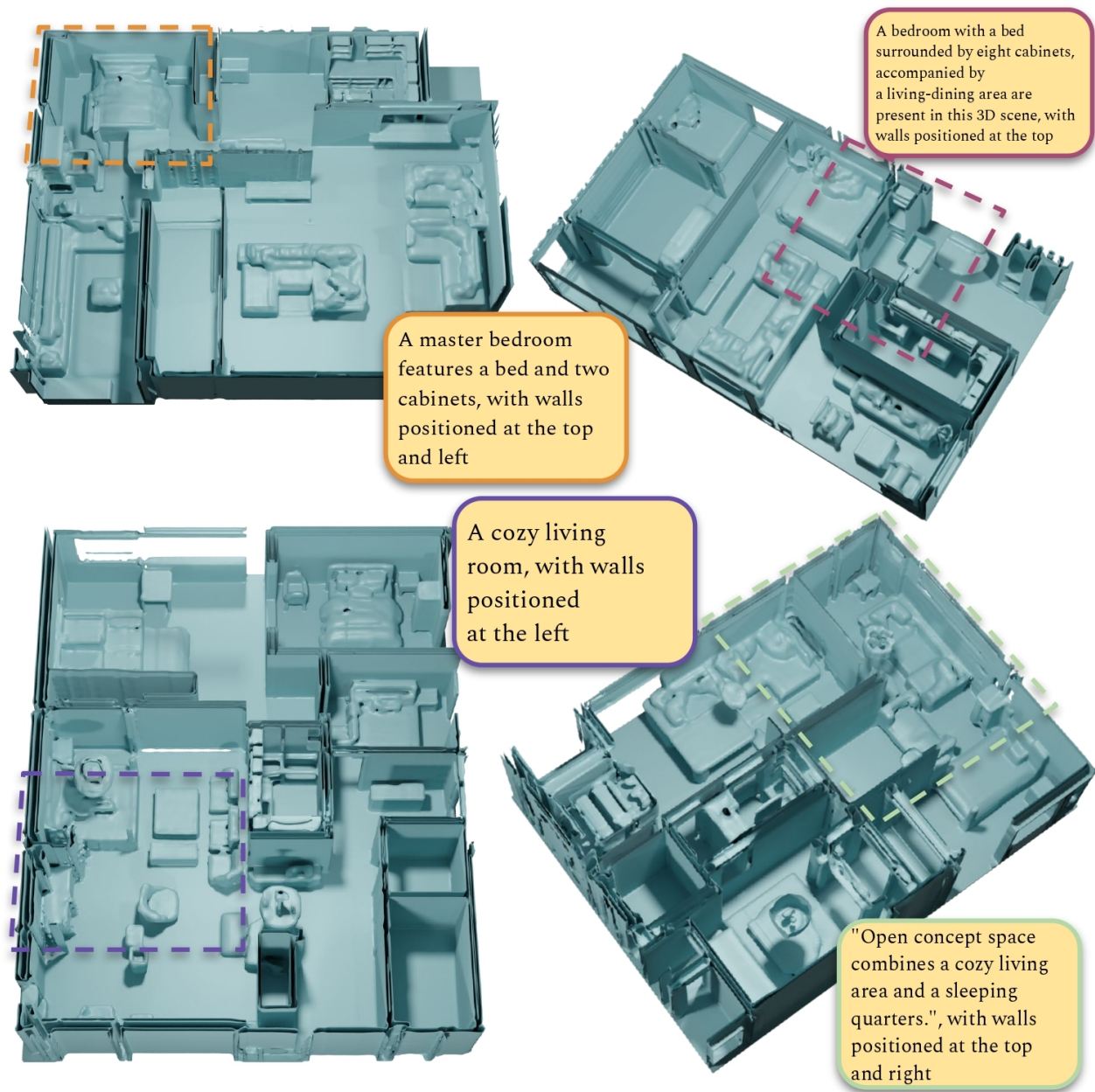


Figure 19. Additional qualitative results for scene generation with SceneFactor.

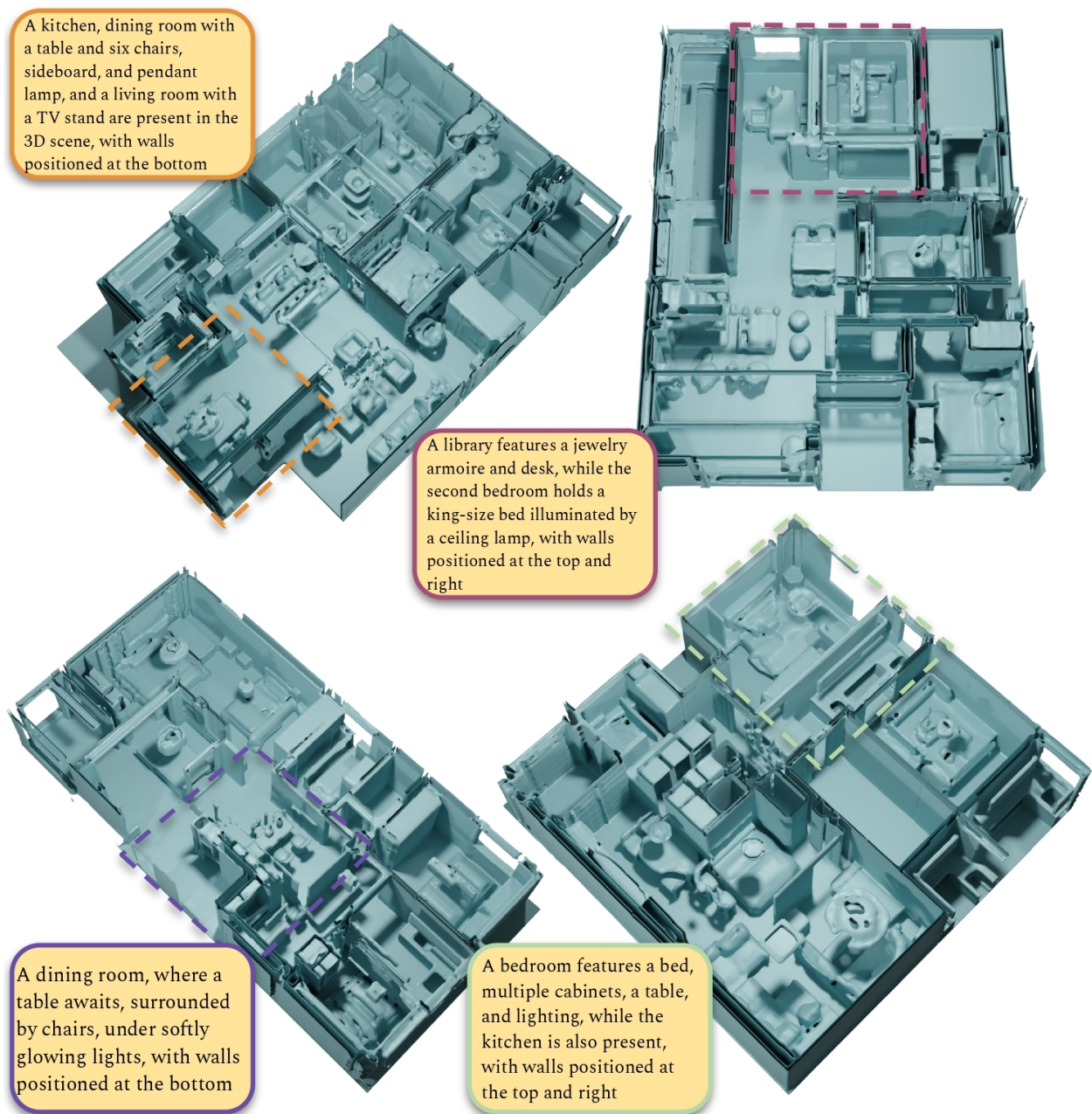


Figure 20. Additional qualitative results for scene generation with SceneFactor.

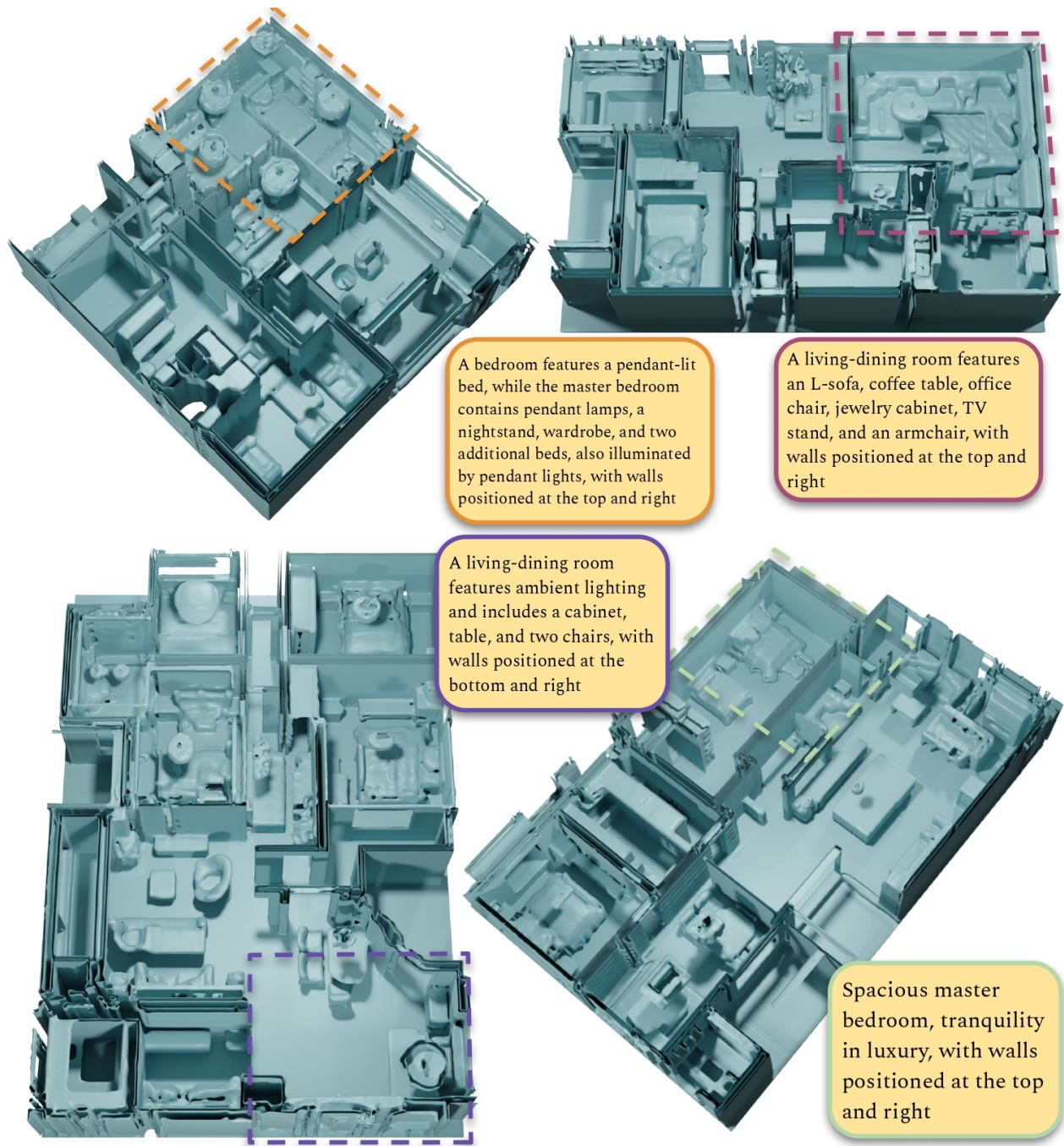


Figure 21. Additional qualitative results for scene generation with SceneFactor.