Taming Stable Diffusion for Text to 360° Panorama Image Generation

Supplementary Material

The supplementary material is organized as follows. In Sec. A, we provide more details about the network architecture. In Sec. B, we provide more details about the experiment setup and baseline methods. In Sec. E, we provide more qualitative comparisons. In Sec. F, we provide more details about the layout conditioned generation. In Sec. G, we provide more generalization results to out-domain prompts.

A. Network Architecture

In Sec. 3 of the main paper, we introduced our dual-branch architecture for panorama generation. Here we provide more details about the position of inserting EPPA into the UNet of SD and the feature dimension of each layer in Tab. A.1. We found that inserting EPPA earlier than the first DownSampler (Table A.1(8)) and later than the last UpSampler (Table A.1(47)) is memory-consuming due to large feature maps with no better performance. Therefore, we insert EPPA right after the DownSampler and before the UpSampler of each block.

B. Experiment Details

As mentioned in Sec. 4 of the main paper, here we provide more details about the experiment setup and baseline methods.

Dataset. Matterport3D dataset [3] is a large-scale scene understanding dataset with 10,800 panoramic images from 90 building-scale scenes. For text conditioned generation, we utilize BLIP-2 [18] to generate a short description of the full image with a prompt of "a 360 - degree view of". We use the same data split as [47], which contains 9,820 for training and 1,092 for evaluation. We note that the original Matterport3D dataset contains blurry regions near the upper and lower edges, as shown in Fig. B.1a. Therefore, our model is trained to generate images with similar blurry regions. For text and layout conditioned generation, we use the MatterportLayout [50, 67] dataset, which annotates room layout for 2,295 indoor panoramic images in the Matterport3D dataset, with 1,648 for training, 191 for validation, and 459 for testing.

Implementation Details. We implement our model in PyTorch based on the implementation of Stable Diffusion [31] from Diffusers [49]. When training our dual-branch model for text-conditioned generation, we jointly train the EPPA module and finetune the two branches with rank-4 LoRA to the new resolution. We randomly sample 20 views as the input of the perspective branch to encourage EPPA to understand the correspondence provided by SPE and attention mask instead of remembering the fixed camera poses. Following MVDiffusion [47], we train for 10 epochs with the AdamW optimizer, using a batch size of 4 and learning rate of 2e-4 for training, and a DDIM sampler [42] is used with a step size of 50 for inference. When training an additional ControlNet for text-layout conditioned generation, we extend training to 100 epochs due to less room layout annotations. Training is conducted on 4 NVIDIA A100 GPUs and takes about 8 hours for text conditioned generation and 15 hours for text-layout conditioned generation.

Perspective transformation details. In the main paper, we denote the transformation from equirectangular panorama $I^* \in \mathbb{R}^{C\times H\times W}$ to perspective image $I \in \mathbb{R}^{C\times h\times w}$ as $I = \mathrm{P}\left(I^*, R, \mathrm{FoV}, (h, w)\right)$, where the rotation matrix $R \in SO(3)$ describes the camera extrinsic matrix and FoV and image size (h, w) define the camera intrinsic matrix K. Specifically, given a pixel $p \in \mathbb{R}^2$ on the image plane of I, we shoot a ray $K^{-1}\left[p, 1\right]^T$ from the camera center, and then transform it to the 3D coordinate of panorama as $v = R^{-1}K^{-1}\left[p, 1\right]^T$. Subsequently, its corresponding pixel p^* on the image plane of I^* can be computed as:

$$p^* = \left[\frac{W(atan2(v_y, v_x)+\pi)}{2\pi}, \frac{H\left(atan2\left(v_z, \sqrt{v_x^2+v_y^2}\right)+\pi/2\right)}{\pi}\right],$$

which is used to bilinearly interpolate I from I^*. Note that we use different symbols here for easy explanation.

Evaluation Metrics. Previous works MVDiffusion [47] and Text2Light [4] both address the problem of text conditioned image generation, but in different domains. MVDiffusion generates 8 horizontal views with 90° FoV, thus limiting the evaluation to perspective images. Text2Light generates a full 180° vertical FoV, therefore focusing on evaluating the panorama quality. Ours is closer to the latter, but to showcase the effectiveness of our proposed method, we conduct a comparison in both. We also detail the implementation of layout consistency evaluation in the following.

- *In the panorama domain*, we value Fréchet Auto-Encoder Distance (FAED) [26] more, since it is customized for panorama and uses an auto-encoder trained on the target dataset as the feature extractor. Specifically, we train the auto-encoder similar to [26] but with RGB images instead of RGBD by removing the depth branch. The auto-encoder is trained on the training set of the Matterport3D dataset for 60 epochs with Adam optimizer and batch size of 4. An exponential learning rate scheduler is used with an initial learning rate of 1e-4 and decay rate of 0.99 for every epoch.
- *In the perspective domain*, the CS is measured between the perspective image and the text prompt captioned from GT view using BLIP-2 [18].

(a) Panorama image. (b) Layout distance map.

Figure B.1. An example of a panoramic image from the Matterport3D dataset [3] (a) and its room layout rendered as distance map (b). Regions near the upper and lower edges of the panoramic image are blurry in the original dataset.

• *When evaluating layout consistency*, to make the comparison fair for MVDiffusion, we mask out pixels outside its vertical FoV before feeding the generated panorama to HorizonNet for our method, so that we do not benefit from the larger FoV when estimating the layout. We finetune HorizonNet on the masked training set of MatterportLayout dataset for 100 epochs with Adam optimizer and batch size of 4. The initial learning rate is set to 1e-4 and halved if the validation loss does not decrease for 10 epochs.

Additional comparison with previous methods. We follow [47] to use the released weights of Text2Light [4] and MVDiffusion [47] as two of our baselines. For Text2Light, we use its first stage without super-resolution inverse tone mapping stage to get panoramic images at a resolution of 512×1024, which takes 80.6 seconds per image on a single NVIDIA A100 GPU. For MVDiffusion, we use its direct outputs for quantitative comparison in main paper Tab. 1. This favors MVDiffusion by avoiding inconsistency in stitching and interpolating in projection. Therefore, to make a comprehensive comparison, we additionally evaluate MVDiffusion in different settings in Tab. B.1. We detail these settings in the following.

• *MVDiffusion* is in its original setting that does not involve stitching and projection, and is used for comparison in main paper Tab. 1. It outputs 8 horizontal views with 90° FoV at a resolution of 512×512, which takes 102.2 seconds. One only difference from the original MVDiffusion paper is that we downsample the output images to 256×256 before evaluation to match the resolution of GT images.

• *MVDiffusion (projection)* uses the same weight as MVDiffusion, but we stitch its outputs into a panorama and then project the panorama back to perspective views for evaluation. This strictly follows our setting of panorama generation by considering the inconsistency between the perspective images. The performance drops significantly, which shows that inconsistency is a major issue for MVDiffusion.

• *MVDiffusion+LoRA* is MVDiffusion finetuned with LoRA on a lower resolution at 256×256. With lower resolution, the inference time is reduced to 27.4 seconds for a panorama with 90° vertical FoV at the resolution of 256×1024, while our method takes 15.1 seconds to generate a panorama with 180° vertical FoV at the resolution of 512×1024. This setting skips the stitching and projection thus does not reflect the actual panorama generation ability of MVDiffusion.

• *MVDiffusion+LoRA (projection)* follows the evaluation setting of MVDiffusion (projection) but uses the same weight as MVDiffusion+LoRA. The FID is better than MVDiffusion (projection), but still significantly worse than ours. This version is used for layout conditioned generation in Tab. 3 of the main paper, detailed in Sec. F.

While our method achieves better realism than baseline methods, it comes with a cost of higher computational complexity as discussed in Sec. 5 of the main paper. Specifically, the average inference time is 2.8 and 2.9 seconds per panorama for SD+LoRA and Pano Branch, respectively. However, we note that our model can be further optimized for higher speed as a significant amount of numpy operations are used for the EPPA module.

C. Loop Consistency Analysis

In Sec. 3.2, we describe two techniques to eliminate loop inconsistency, *i.e.*, latent rotation and circular padding. Qualitative results in Fig. C.1 show the stitched ends of generated panoramas with each column corresponding to one input text. We can see that latent rotation (b) can only mitigate loop inconsistency of SD+LoRA (a), while the results with circular padding combined (c) or alone (d) are more seamless.

D. Repetition analysis.

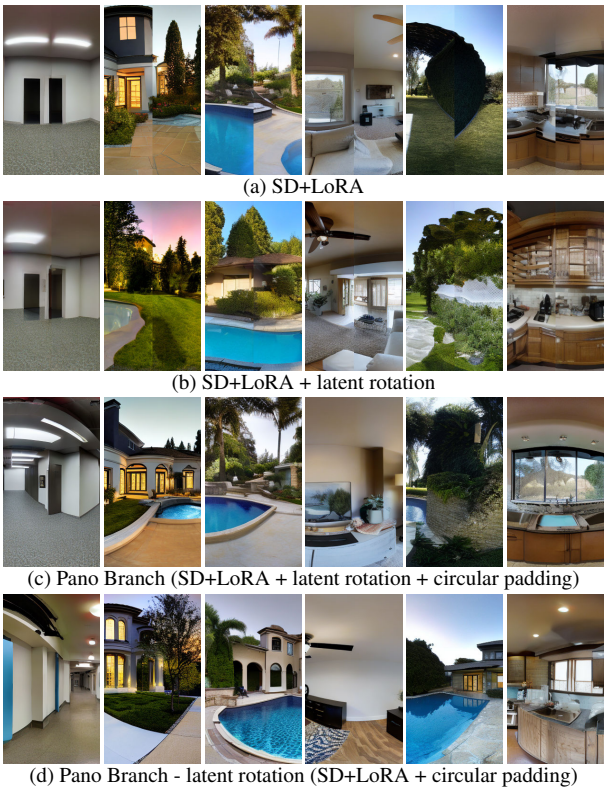
In Sec. 4.2, we qualitatively highlight the repetition issue of MVDiffusion. Here, we try to evaluate the repetition by projecting panorama to cubemap and computing a score $\mathrm{RS}\left(I_i, I_j\right) = max\left(100 * \cos\left(E_i, E_j\right), 0\right)$ between each

#	Layer	Output		Additional Inputs
		Pers Branch (20×)	Pano Branch	
(1)	Latent Map	4 × 32 × 32	4 × 64 × 128	
(2)	Conv.	320 × 32 × 32	320 × 64 × 128	
	CrossAttnDownBlock1			
(3)	ResBlock	320 × 32 × 32	320 × 64 × 128	Time emb.
(4)	AttnBlock	320 × 32 × 32	320 × 64 × 128	Prompt emb.
(5)	ResBlock	320 × 32 × 32	320 × 64 × 128	Time emb.
(6)	AttnBlock	320 × 32 × 32	320 × 64 × 128	Prompt emb.
(7)	DownSampler	320 × 16 × 16	320 × 32 × 64	
(8)	EPPA	320 × 16 × 16	320 × 32 × 64	
	CrossAttnDownBlock2			
(9)	ResBlock	640 × 16 × 16	640 × 32 × 64	Time emb.
(10)	AttnBlock	640 × 16 × 16	640 × 32 × 64	Prompt emb.
(11)	ResBlock	640 × 16 × 16	640 × 32 × 64	Time emb.
(12)	AttnBlock	640 × 16 × 16	640 × 32 × 64	Prompt emb.
(13)	DownSampler	640 × 8 × 8	640 × 16 × 32	
(14)	EPPA	640 × 8 × 8	640 × 16 × 32	
	CrossAttnDownBlock3			
(15)	ResBlock	1280 × 8 × 8	1280 × 16 × 32	Time emb.
(16)	AttnBlock	1280 × 8 × 8	1280 × 16 × 32	Prompt emb.
(17)	ResBlock	1280 × 8 × 8	1280 × 16 × 32	Time emb.
(18)	AttnBlock	1280 × 8 × 8	1280 × 16 × 32	Prompt emb.
(19)	DownSampler	1280 × 4 × 4	1280 × 8 × 16	
(20)	EPPA	1280 × 4 × 4	1280 × 8 × 16	
	DownBlock			
(21)	ResBlock	1280 × 4 × 4	1280 × 8 × 16	Time emb.
(22)	ResBlock	1280 × 4 × 4	1280 × 8 × 16	Time emb.
	MidBlock			
(23)	ResBlock	1280 × 4 × 4	1280 × 8 × 16	Time emb.
(24)	AttnBlock	1280 × 4 × 4	1280 × 8 × 16	Prompt emb.
(25)	ResBlock	1280 × 4 × 4	1280 × 8 × 16	Time emb.
(26)	EPPA	1280 × 4 × 4	1280 × 8 × 16	
	UpBlock			
(27)	ResBlock	1280 × 4 × 4	1280 × 8 × 16	(22), Time emb.
(28)	ResBlock	1280 × 4 × 4	1280 × 8 × 16	(21), Time emb.
(29)	ResBlock	1280 × 4 × 4	1280 × 8 × 16	(19), Time emb.
(30)	EPPA	1280 × 4 × 4	1280 × 8 × 16	
(31)	UpSampler	1280 × 8 × 8	1280 × 16 × 32	
	CrossAttnUpBlock1			
(32)	ResBlock	1280 × 8 × 8	1280 × 16 × 32	(18), Time emb.
(33)	AttnBlock	1280 × 8 × 8	1280 × 16 × 32	Prompt emb.
(34)	ResBlock	1280 × 8 × 8	1280 × 16 × 32	(16), Time emb.
(35)	AttnBlock	1280 × 8 × 8	1280 × 16 × 32	Prompt emb.
(36)	ResBlock	1280 × 8 × 8	1280 × 16 × 32	(13), Time emb.
(37)	AttnBlock	1280 × 8 × 8	1280 × 16 × 32	Prompt emb.
(38)	EPPA	1280 × 8 × 8	1280 × 16 × 32	
(39)	UpSampler	1280 × 16 × 16	1280 × 32 × 64	
	CrossAttnUpBlock2			
(40)	ResBlock	640 × 16 × 16	640 × 32 × 64	(12), Time emb.
(41)	AttnBlock	640 × 16 × 16	640 × 32 × 64	Prompt emb.
(42)	ResBlock	640 × 16 × 16	640 × 32 × 64	(10), Time emb.
(43)	AttnBlock	640 × 16 × 16	640 × 32 × 64	Prompt emb.
(44)	ResBlock	640 × 16 × 16	640 × 32 × 64	(7), Time emb.
(45)	AttnBlock	640 × 16 × 16	640 × 32 × 64	Prompt emb.
(46)	EPPA	640 × 16 × 16	640 × 32 × 64	
(47)	UpSampler	640 × 32 × 32	640 × 64 × 128	
	CrossAttnUpBlock3			
(48)	ResBlock	320 × 32 × 32	320 × 64 × 128	(6), Time emb.
(49)	AttnBlock	320 × 32 × 32	320 × 64 × 128	Prompt emb.
(50)	ResBlock	320 × 32 × 32	320 × 64 × 128	(4), Time emb.
(51)	AttnBlock	320 × 32 × 32	320 × 64 × 128	Prompt emb.
(52)	ResBlock	320 × 32 × 32	320 × 64 × 128	(2), Time emb.
(53)	AttnBlock	320 × 32 × 32	320 × 64 × 128	Prompt emb.
(54)	GroupNorm	320 × 32 × 32	320 × 64 × 128	
(55)	SiLU	320 × 32 × 32	320 × 64 × 128	
(56)	Conv.	4 × 32 × 32	4 × 64 × 128	

Table A.1. Detailed PanFusion pipeline. We highlight the inserted EPPA modules in orange.

Method	Horizontal 8 Views [47]		
	FID ↓	IS ↑	CS ↑
MVDiffusion [47]	25.27	6.90	26.34
MVDiffusion (projection)	32.56	6.40	25.70
MVDiffusion+LoRA	21.76	6.55	25.22
MVDiffusion+LoRA (projection)	30.04	5.69	24.90
PanFusion (Ours)	19.88	6.50	24.98

Table B.1. More quantitative comparison. We compare our method with MVDiffusion [47] in different settings. MVDiffusion with projection considers stitching and projection, which is closer to our setting. We also finetune MVDiffusion with LoRA [14] on low resolution to have a fair comparison for time efficiency and layout-conditioned generation.



(a) SD+LoRA

(b) SD+LoRA + latent rotation

(c) Pano Branch (SD+LoRA + latent rotation + circular padding)

(d) Pano Branch - latent rotation (SD+LoRA + circular padding)

Figure C.1. Loop consistency analysis. We stitch both ends of each generated panorama. Here, each column corresponding to one same input text. It is shown that latent rotation (b) can only mitigate loop inconsistency of SD+LoRA (a), while the results with circular padding combined (c) or alone (d) are more seamless.

pair of 4 horizontal views, where E_* is the CLIP embedding of image I_*. RS is averaged over all image pairs of 1,092 test samples, with higher values indicating more repetition. It is shown in Tab. D.1 that our method has the lowest RS while MVDiffusion has the most repetition.

	Text2Light	MVDiffusion	PanFusion (Ours)	GT image
RS ↓	88.81	90.79	88.13	86.49

Table D.1. Repetition analysis. Inspired by CLIP Score [29], we report the repetition score (RS) that measures the similarity between different parts of the generated panorama images. Lower RS indicates less repetition.

Method	Layout Consistency		Horizontal 8 Views [47]		
	3D IoU ↑	2D IoU ↑	FID ↓	IS ↑	CS ↑
SD+LoRA [14, 31]	68.02	71.41	**21.39**	5.03	25.84
PanFusion (Ours)	**68.46**	**71.82**	22.58	**5.10**	**26.04**

Table F.1. Layout-conditioned comparison with SD+LoRA. Our method achieves comparable or better results.

E. More Qualitative Comparisons

In Sec. 4.2 Fig. 4 of the main paper, we compared our method with previous methods qualitatively. Due to space limitations, we cropped the generated images to the vertical FoV of MVDiffusion for all methods. Here we provide more qualitative comparisons without cropping in Figs. E.1 to E.10, where Fig. E.1 has the same prompts as Fig. 4 in the main paper. Similarly, we evenly sample 4 horizontal views from the generated panorama for each panorama, in which the first view crosses the left and right borders to show how loop consistency is handled.

F. Layout Conditioned Generation Details

In Sec. 3.4 of the main paper, we showcased the benefits of our dual-branch method with the application of layout conditioned generation. Specifically, the room layout is rendered as a distance map, as shown in Fig. B.1b, and normalized to the range of $[-1, 1]$ as an additional spatial condition. To add layout condition to MVDiffusion, we follow [43] to project the layout condition to perspective views as a distance map instead of a depth map to ensure consistency among overlapped regions. However, when training the ControlNet for MVDiffusion at the original resolution of 512×512, it suffers from gradient explosion. Instead, we found finetuning MVDiffusion with LoRA on a lower resolution of 256×256 can make the training of the ControlNet converge, and also improve the realism of MVDiffusion. Therefore, we use MVDiffusion+LoRA as the base model for layout conditioned generation in main paper Tab. 3 to serve as a stronger baseline. In Figs. F.1 to F.2, we provide more quantitative comparison with MVDiffusion. We also compare with SD+LoRA in Tab. F.1 to show that our method can get comparable or better results.

G. Generalization to Out-domain Prompts

While our method is trained on the Matterport3D dataset, which contains mostly indoor scenes, we show that it can generalize to out-domain prompts and transfer its knowledge of layout understanding to outdoor scenes, as shown in Figs. G.1 to G.4.

H. Future Works

Future works might include introducing more controls over the style and content of the generated panorama images to support applications like virtual house tour, or extending the method to enable outpainting by exploiting the perspective branch to extract guidance from the input image. The dual-branch architecture can also potentially benefit texture generation for 3D models, where the global branch can operate on UV maps and the perspective branch can operate on rendered images.

Figure E.1. More qualitative comparisons.

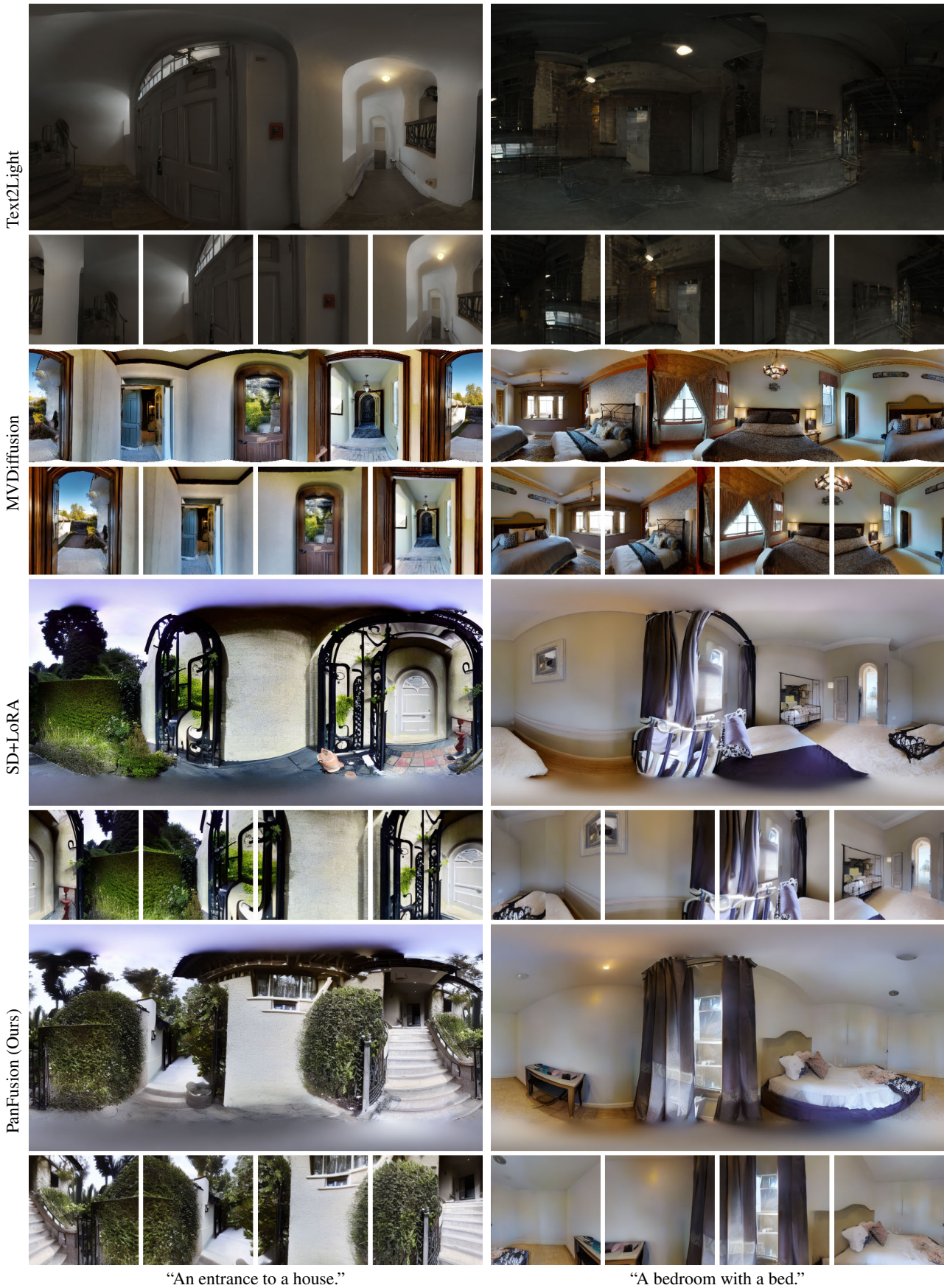


Figure E.2. More qualitative comparisons.

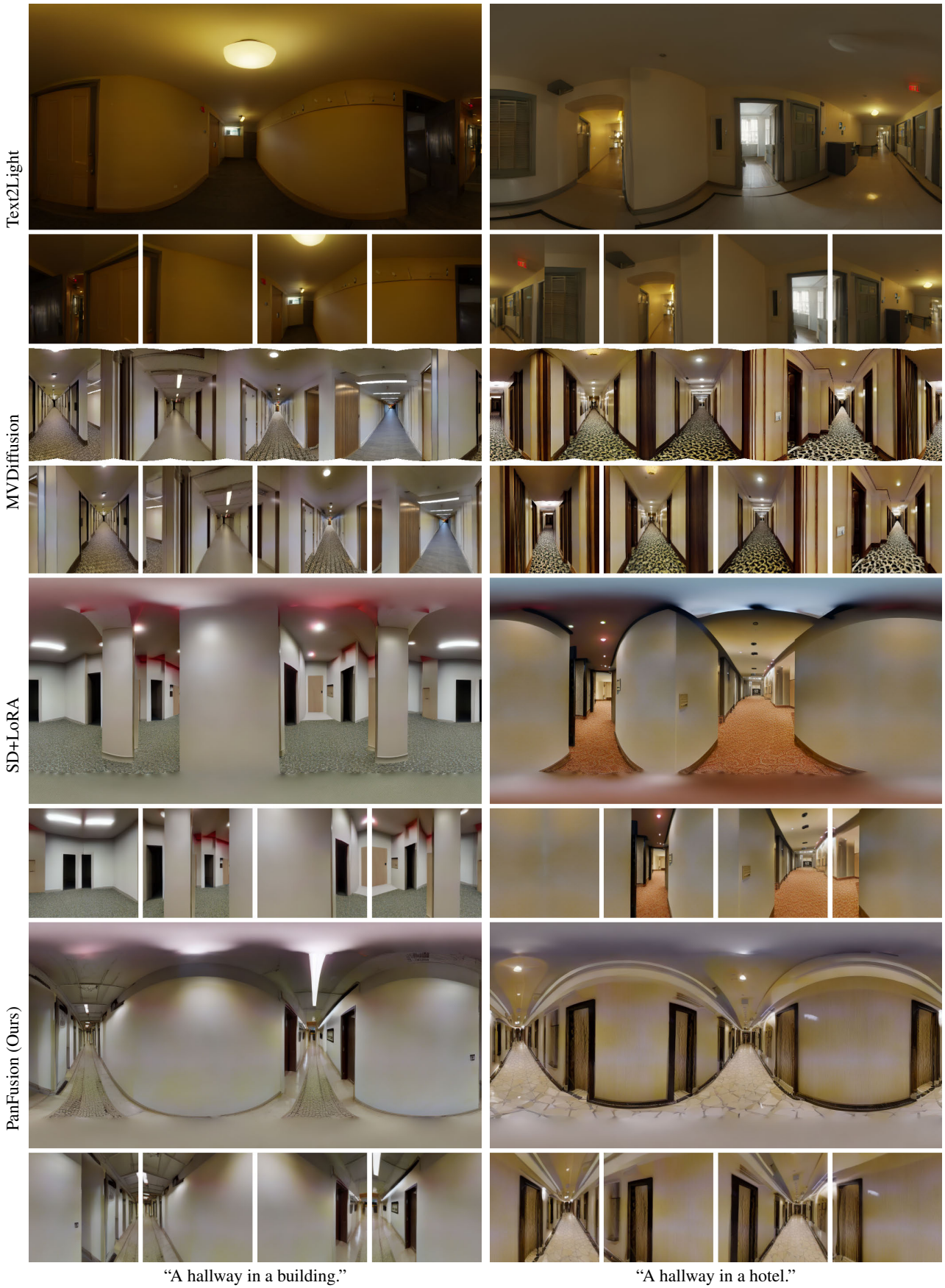


Figure E.3. More qualitative comparisons.

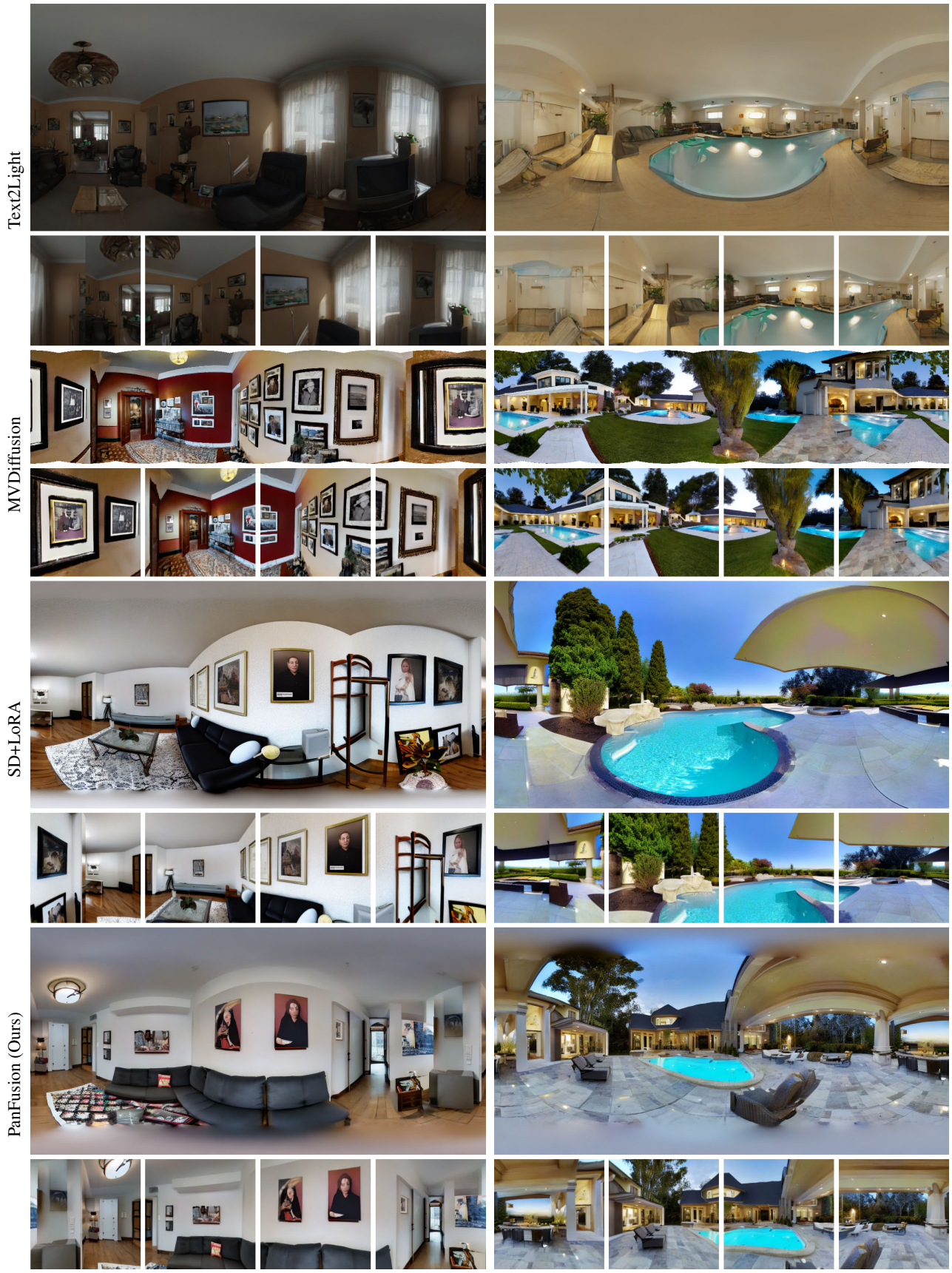


Figure E.4. More qualitative comparisons.

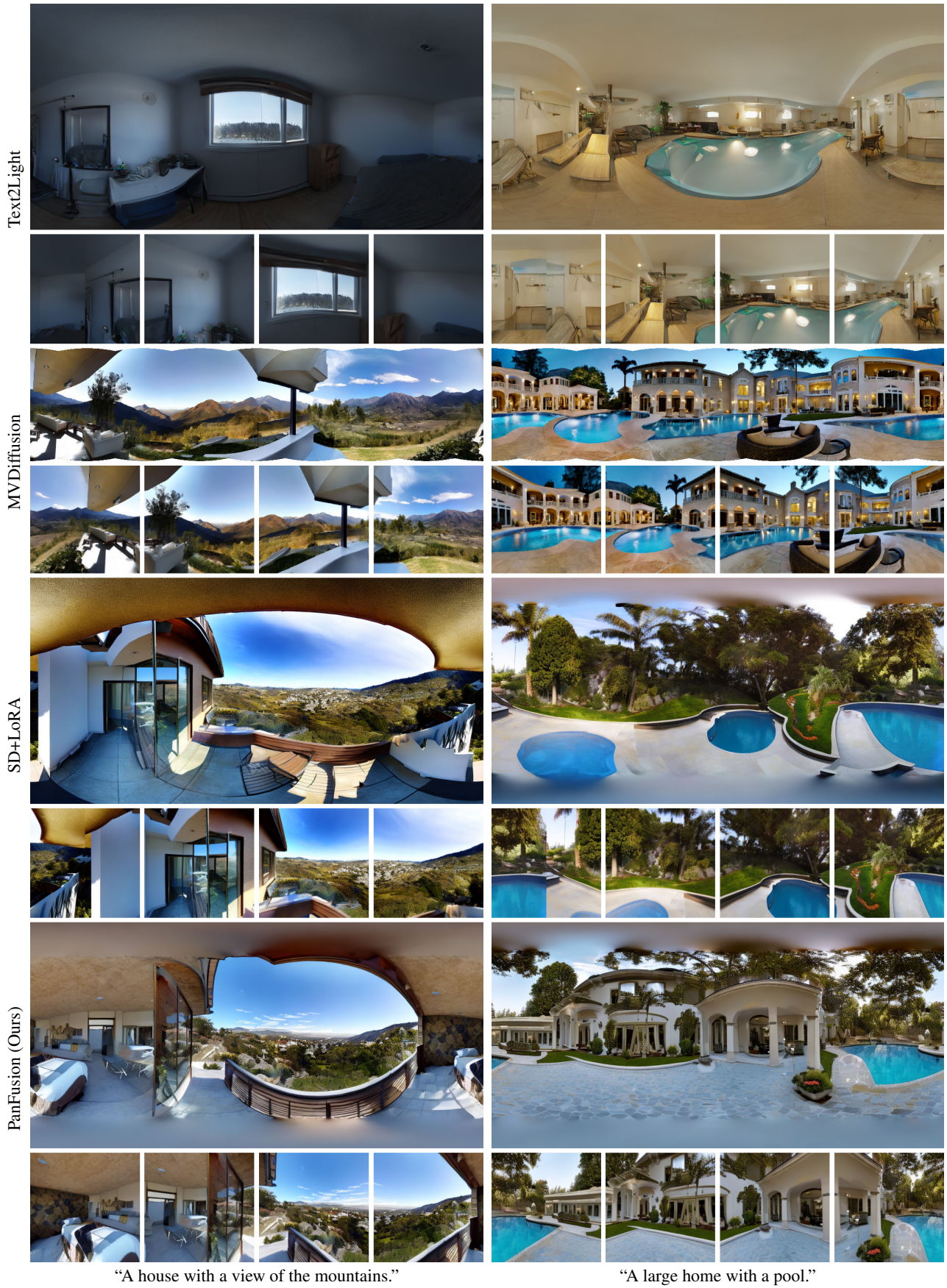


Figure E.5. More qualitative comparisons.

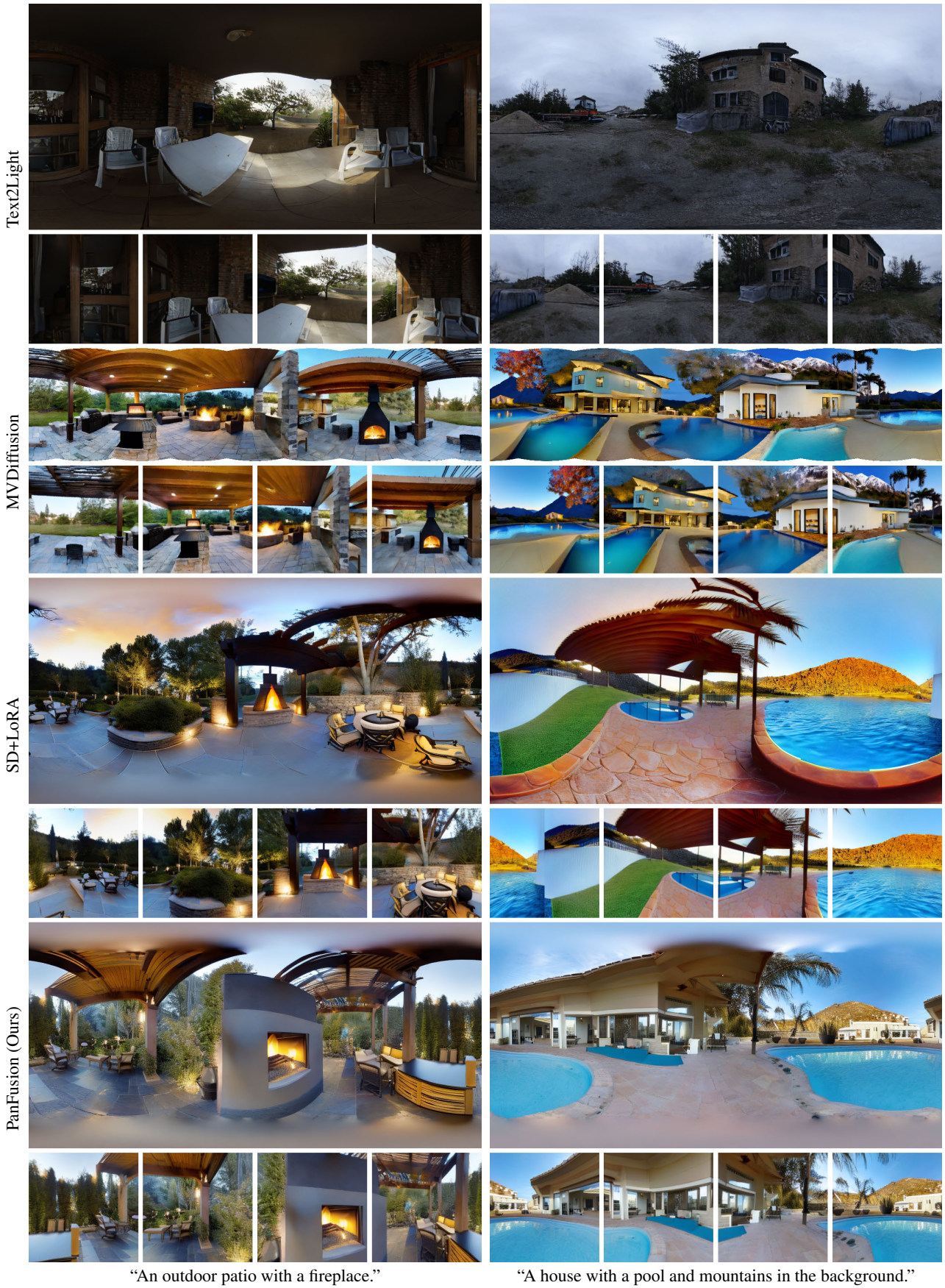


Figure E.6. More qualitative comparisons.

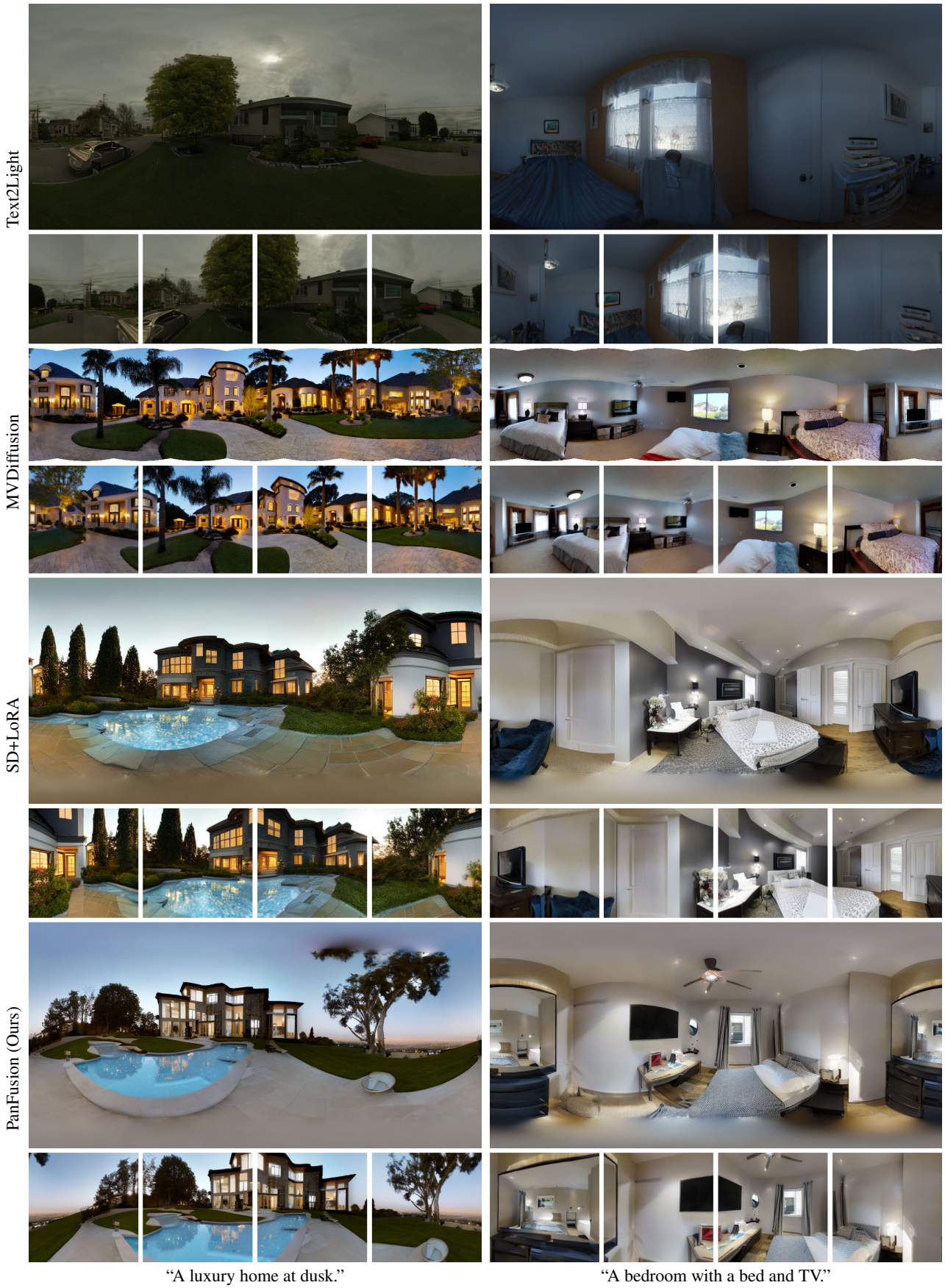


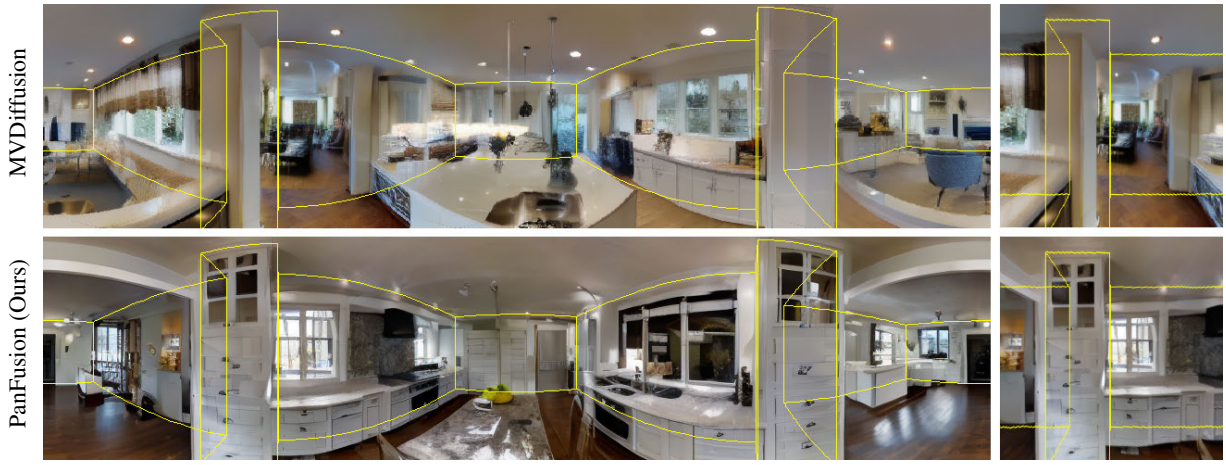
Figure E.7. More qualitative comparisons.

Figure E.8. More qualitative comparisons.

Figure E.9. More qualitative comparisons.

Figure E.10. More qualitative comparisons.

"A bathroom with a tub and sink."



"A kitchen and dining room."

"A hallway in an office."

Figure F.1. More layout-conditioned generation comparisons.

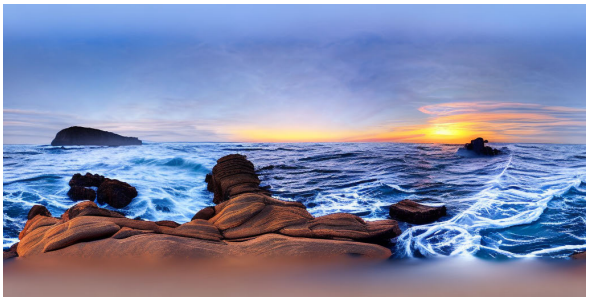
"An office with glass walls."

"A kitchen and dining room."

"A living room and dining room."

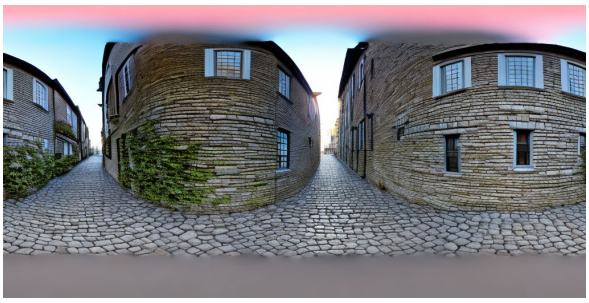
Figure F.2. More layout-conditioned generation comparisons.

“A futuristic kitchen.”



“Coastal cliff at sunset, waves crashing on rugged rocks.”

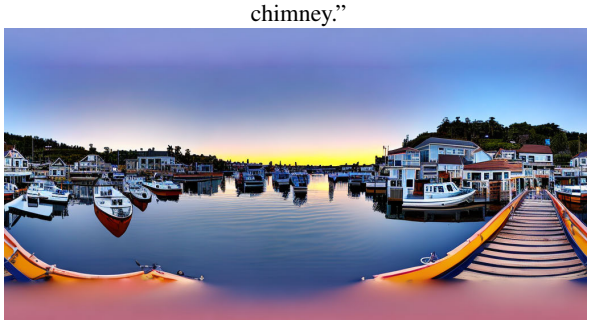
“Urban skyline at twilight, city lights twinkling in the distance.”



“Cobblestone alley, historic architecture bathed in soft morning light.”

“Snow-covered cottage, smoke rising from a charming stone chimney.”

“An underwater scene, where coral reefs teem with colorful fish beneath the clear blue ocean.”



“A peaceful coastal village at sunrise, with fishing boats docked along the quiet harbor.”

“The interior of a historic library, filled with rows of antique books, leather-bound and dust-covered.”

“A tranquil botanical garden, with exotic plants, blooming flowers, and meandering stone pathways.”

“The calm waters of a secluded lake, reflecting the colors of the surrounding autumn foliage.”

Figure G.1. Generalization to out-domain prompts.

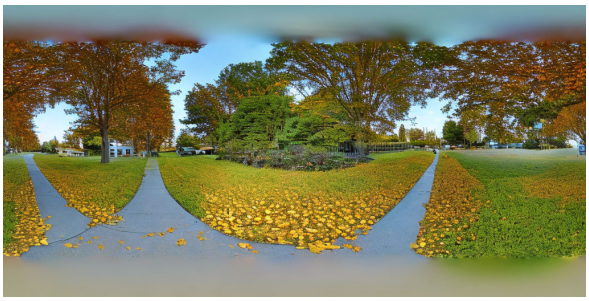
“Lighthouse in stormy seas.”

“Desert canyon, sculpted sandstone.”

“Balcony garden, blooming serenity.”

“Firelit cabin, crackling warmth amid snowy woods.”

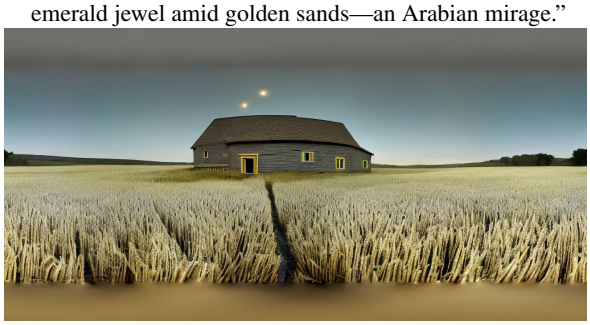
“Desert sunrise, silhouettes painted against the golden horizon.”



“Suburban street, autumn leaves carpeting the sidewalk in hues.”

“Desert oasis, palm trees surrounding a pristine pool, an emerald jewel amid golden sands—an Arabian mirage.”

“Alpine village, snow-covered rooftops, nestled between majestic peaks—a picture-perfect scene of winter tranquility.”



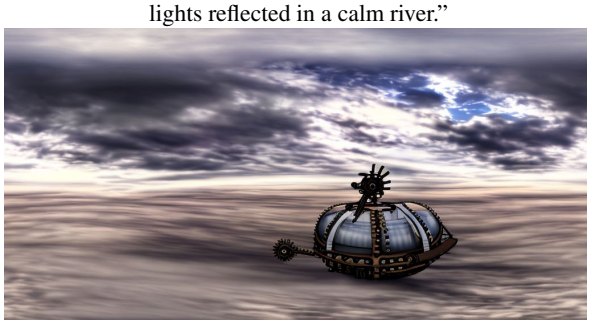
“Rustic farmhouse, weathered by time, surrounded by fields of golden wheat—a pastoral scene capturing the essence of simplicity.”

“Alpine meadow, wildflowers swaying in a mountain breeze, snow-capped peaks embracing a serene panorama—a high-altitude sanctuary.”

Figure G.2. Generalization to out-domain prompts.

“A futuristic cityscape with floating skyscrapers and neon lights reflected in a calm river.”

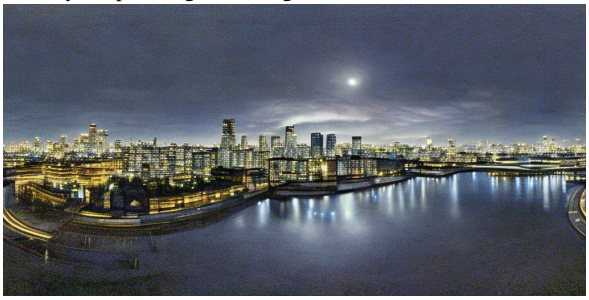
“In the heart of a bustling market, the aroma of exotic spices mingles with the vibrant colors of fresh produce.”



“Steampunk airship, navigating cloudy skies, gears turning, propellers whirring.”

“Urban rooftop garden, vibrant blooms against a backdrop of skyscrapers, a green refuge amid concrete and steel.”

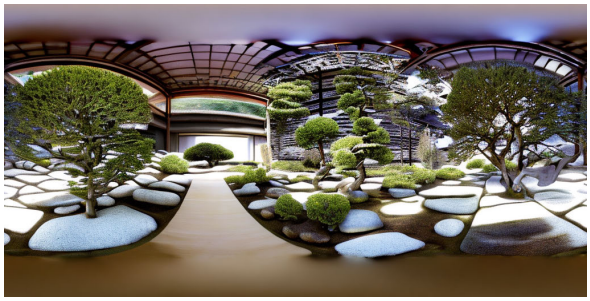
“Coastal cliffside, waves crashing on rugged rocks, seagulls soaring in the salty breeze—a dramatic meeting of land and sea.”



“Moonlit cityscape, reflections shimmering on rain-kissed streets, a quiet metropolis under the night sky—an urban nocturne.”

“Coastal lighthouse, guiding ships through the moonlit night.”

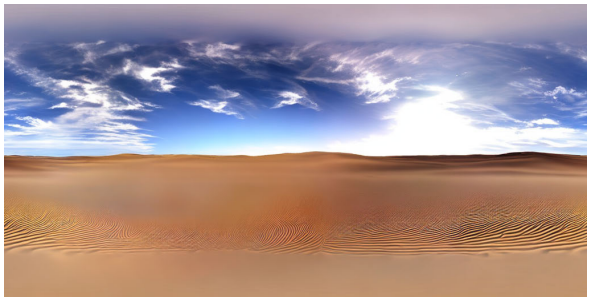
“Rooftop garden, city lights below, a quiet urban oasis.”



“Zen garden, raked pebbles, and bonsai trees—a serene oasis.”

“Tropical paradise, palm trees swaying, turquoise waters lapping sandy shores.”

Figure G.3. Generalization to out-domain prompts.

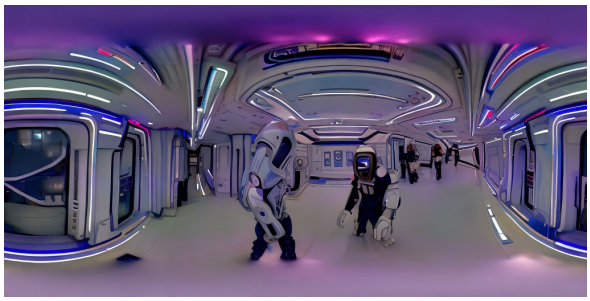


“Desert dunes, endless golden waves.”

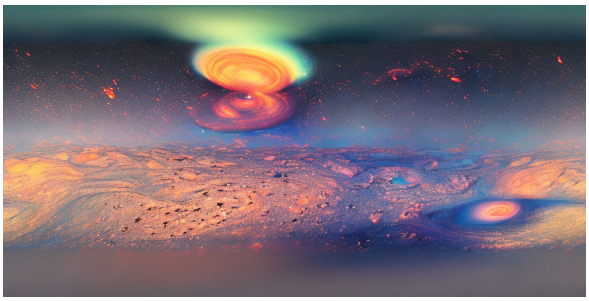
“Antique bookstore, leather-bound treasures.”

“Alpine cabin, snow-capped serenity.”

“Rain-soaked city streets, glistening reflections.”



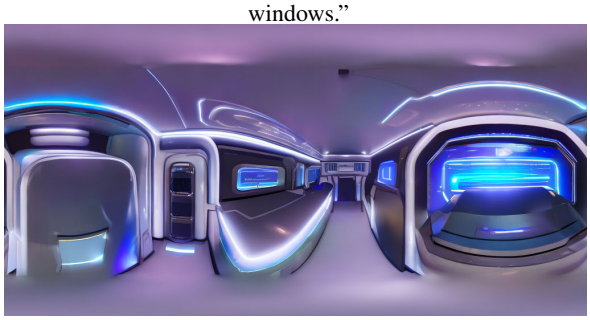
“Inside a bustling space station, people from different galaxies interact amid futuristic architecture and advanced robotics.”



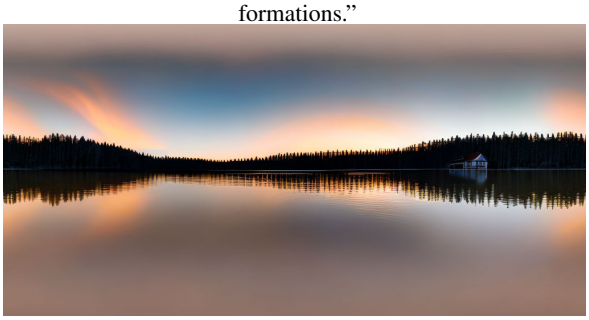
“On the surface of a distant planet, a landscape of alien rock formations and swirling, multicolored gases.”

“A cozy coffee shop on a rainy day, with the comforting scent of freshly brewed coffee and the sound of rain on the windows.”

“Standing on the edge of the Grand Canyon, marveling at the vastness of the canyon and the layers of colorful rock formations.”



“A spaceship interior adorned with holographic displays, sleek metallic surfaces, and advanced technology.”



“A serene lakeside cabin at dawn, with mist rising from the water and the first light of the day illuminating the landscape.”

Figure G.4. Generalization to out-domain prompts.