DiffDreamer: Towards Consistent Unsupervised Single-view Scene Extrapolation with Conditional Diffusion Models

Supplementary Material

A. Additional quantitative results

We supply quantitative ablation comparisons including DiffDreamer-diverse on LHQ [86] in Tab. 5, and additional quantitative results of DiffDreamer-diverse on ACID [42] in Tab. 6. We further report CLIP feature similarity between frames in Tab. 7, a metric used by several recent video generation methods, including [106, 98], with different intervals.

	20 steps			50 steps			100 steps			COLMAP
Method	FID ↓	KID ↓	IS ↑	FID ↓	KID ↓	IS ↑	FID ↓	KID ↓	IS ↑	
InfNat-0	39.45	0.12	2.80	36.53	0.11	2.79	26.24	**0.12**	2.72	612
Auto-regressive	70.53	0.53	1.99	77.81	0.63	1.91	90.69	0.81	2.14	2030
No anchored	38.41	0.17	2.70	46.40	0.24	2.63	58.67	0.40	2.79	1543
No lookahead	68.30	0.46	1.76	75.18	0.74	1.85	92.85	0.81	2.06	2457
DiffDreamer	**34.49**	**0.08**	2.82	38.86	0.12	2.90	51.0	0.28	2.99	**3124**
DiffDreamer-diverse	34.92	0.09	**3.19**	**30.78**	**0.10**	**3.27**	**24.04**	**0.12**	**3.26**	1403

Table 5: Quantitative ablation studies.

	20 steps			50 steps			100 steps			COLMAP
Method	FID ↓	KID ↓	IS ↑	FID ↓	KID ↓	IS ↑	FID ↓	KID ↓	IS ↑	
InfNat	59.93	0.22	2.36	57.47	0.26	2.28	48.27	0.27	2.28	1476
DiffDreamer	52.81	**0.12**	**2.69**	61.04	0.26	**2.86**	70.11	0.41	**2.82**	**3423**
DiffDreamer-diverse	**51.28**	0.15	2.37	**44.44**	**0.19**	2.40	**42.97**	**0.21**	2.43	1883

Table 6: Quantitative comparison of DiffDreamer-diverse's performance on ACID [42].

CLIP cos. sim. ↑	Pairwise Frame Interval				
Method	1	5	10	15	20
InfNat-0	0.98	0.95	0.92	0.90	0.88
DiffDreamer	**0.99**	**0.97**	**0.96**	**0.94**	**0.92**

Table 7: CLIP similarity between consecutive frames.

B. Additional qualitative results

Figures 12, 13, 14, and 15 show additional scene extrapolation results from our model with 50 steps of forward motion. The task of scene extrapolation has a multi-modal nature: given a single input image, there could be infinite ways of generation. Therefore, we show multiple rendering trajectories of over 50 steps for each input image and supporting videos with framerates upsampled using [64] (note that the videos are rendered at 128×128 and may appear blurry under higher resolution). To encourage diversity and prevent hitting mountains/the ground while generating longer sequences, we can additionally condition the diffusion model on randomly selected patterns from the input image while generating the pseudo future frames with a weight of 0.2. We select these patterns by simply performing free-form brush stroke masking, using the algorithm provided in [102, 103] and refer to this diversity-focused version as "DiffDreamer-diverse", encouraging diversity over long-range at a price of trading-off consistency. We supply results for this diversity-focused setting and show frames from a generated 500-step sequence in Fig. 8.

C. Flying-out

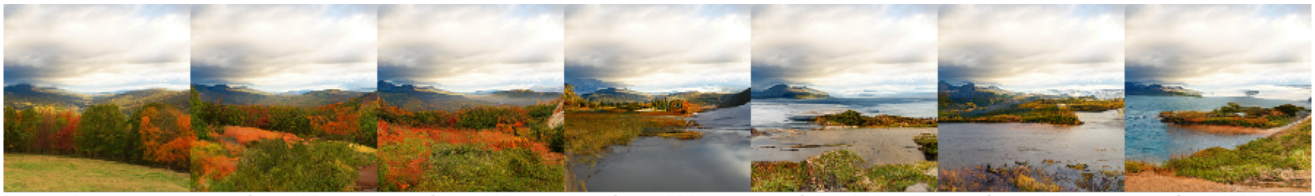
Even though we do not design our model specifically for flying-out setting, DiffDreamer has a significant advantage over naïve autoregression. Since we are working on outdoor scenes, dramatic depth discontinuities will appear [23]. This is especially obvious when the flying-out motion is not just a straight translation. Our bi-directional method is a good counter to this issue since future frame guidance and simultaneous refinement can alleviate the artifacts. We show example flying-out sequences in Fig. 11 and include accompanying videos with 100 steps.

D. Technical details

We use the U-Net backbone from [21] and train all models for 1M iterations with a mini-batch size of 128. We trained our model for roughly a week and 3 days respectively for LHQ [86] and ACID [42], on 2 NVIDIA RTX 8000 GPUs. We compare against the released pretrained InfNat and InfNat-zero models, which were trained for 8 days on 10 GPUs, and 6 days on 8 GPUs respectively. We build our model on top of Palette [71] and use the Adam optimizer with a learning rate of 1e-4 and a 10k linear learning rate warm-up schedule. We also employ 0.9999 EMA for our model. During both training and inference, we use a linear noise schedule of (1e-6, 0.01) with 2000 time steps. Following prior works [42, 40], we extract monocular-predicted disparity maps with MiDaS [62], and sky region masks using DeepLab [13]. We adopt the autocruise algorithm from [42] to sample the camera path for both training and inference. The autocruise algorithm uses the disparity map to estimate the skyline and horizon, then generate a camera trajectory that avoids hitting the ground or hills. We follow [40] during inference and use a camera speed of 0.1875. We train and evaluate our model on image resolution of 128×128 to be consistent with prior work [40].

E. Autocruise specifics

We use the autocruise algorithm from [42] to generate camera trajectories for both training and evaluation. As we



input frame 50 frame 100 frame 200 frame 300 frame 400 frame 500

Figure 8: **Perpetual view synthesis** of a sequence over 500 steps.

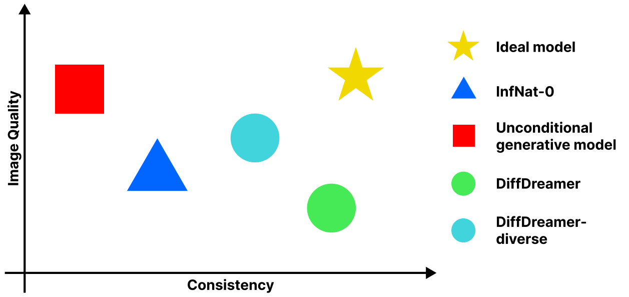


Figure 10: **Specturm** of the two dimensions of scene extrapolation tasks.

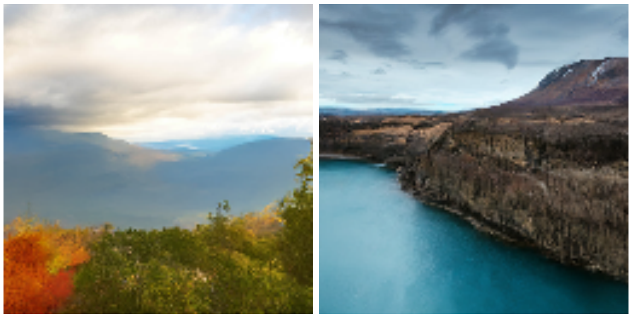


Figure 9: **Failure cases** when the model's output is not diverse enough to support future frames (left) or the autocruise algorithm gets too close to the mountains/ground (right).

only have raw images as training data, whose intrinsics are unknown and cannot be easily inferred, we follow [40] and randomly sample the field of view (FoV) between 45° and 70°, and fix to 55° during testing. Autocruise algorithm deploys a mechanism to predict the next camera pose by encouraging the next view to have a τ_{sky} fraction of sky regions (determined by thresholding disparity less than 0.08) and a fraction of τ_{near} fraction of nearby regions (determined by thresholding disparity larger than 0.4). We follow [40] to uniformly sample τ_{near} from [0.2, 0.4] and τ_{sky} from [0.25, 0.45] during training, and fix them to be 0.25 and 0.1 respectively during inference. In contrast to [42, 40], which only moves a small fraction $\tau_{\text{lerp}} = 0.05$ of the way to the target directions at each frame to ensure smooth camera pose changing, we only use $\tau_{\text{lerp}} = 0.05$ during inference of our next frame and increase $\tau_{\text{lerp}} = 0.3$ for generating the pseudo future frame. We uniformly sample τ_{lerp} from [0.0, 0.3] during training. We direct readers to [42, 40] for further specifics of the autocruise algorithm.

F. Mesh renderer specifics

We use a PyTorch implementation [80] of a 3D mesh renderer [24]. Following [42], each pixel is projected into the 3D space using its disparity and is then treated as a vertex connected with its neighbors to form a triangle mesh. To obtain the missing region masks, we follow [42] and threshold the gradient of the input disparity by 0.3 to make a mask, which refers to the regions with sharp disparity change. We project the mask to target the camera pose to get the final missing region mask.

G. Dataset pre-processing

Both of the LHQ [86] dataset and the ACID [42] dataset contains many samples unsuitable for training scene extrapolation models. This includes images focusing on the foreground and images of the ground, with camera poses pointed downward. Following [40], we filter out images whose minimum MiDaS [62] predicted disparity value is larger than 200.

H. Failure cases

There are two main causes for failures. First, we do not enforce diversity of outputs. During training, the model always sees real images. This means during our pseudo pairs generation, the corrupted version of the ground truth image will still be diverse, even if it is under a lower frequency due to warping artifacts. However, while we are going significantly beyond the input image's content, any future frame will solely rely on the model's outputs, which may not exhibit enough diverse content for moving forward. We show an example of this case in Fig. 9. We believe it is exciting to extend DiffDreamer to support vector conditioning, e.g., CLIP embedding conditioning, to enforce output diversity.

Second, as our model has significantly better geometry alignment than [42], the autocruise algorithm fails more often, causing the camera trajectory to hit mountains or the ground, despite our best efforts in tuning its parameters. We show an example of this failure case in Fig. 9.

Scene extrapolation performance can be evaluated along two axes: image quality and (geometric and appearance) consistency, as shown in Fig. 10. These dimensions are inherently at odds, forming a spectrum of trade-offs. An unconditional image synthesizer might prioritize image quality but would have no consistency between frames, while models emphasizing consistency may somewhat compromise raw image quality. Without enforcing consistency, models tend to converge towards random latent space exploration, as evidenced by the similarities in InfNat-0 videos. We see our work as a starting point for future research in navigating these trade-offs, ultimately striving towards consistent perpetual view generation.

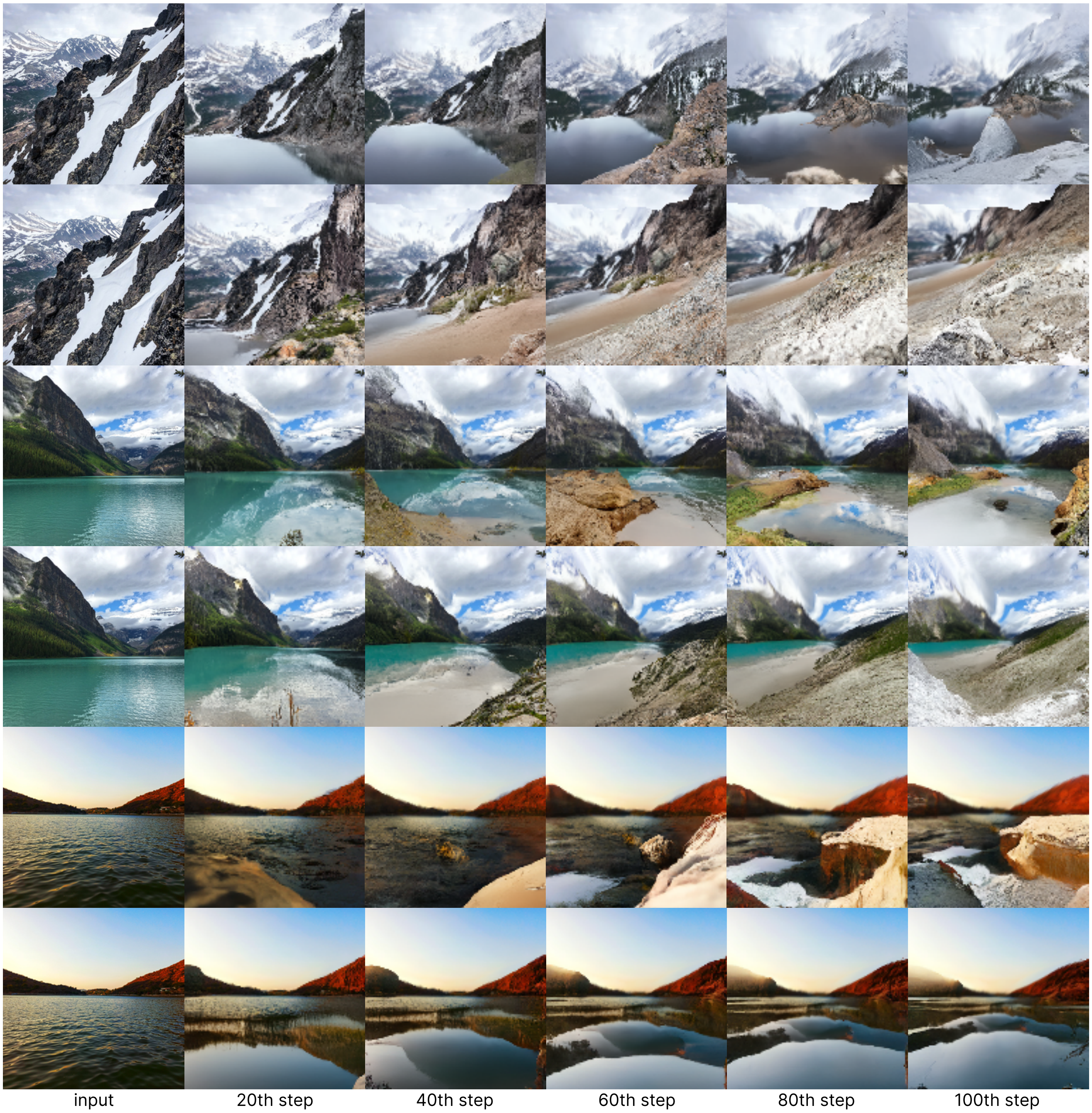


Figure 11: **Flying-out** 100 steps of the input images.

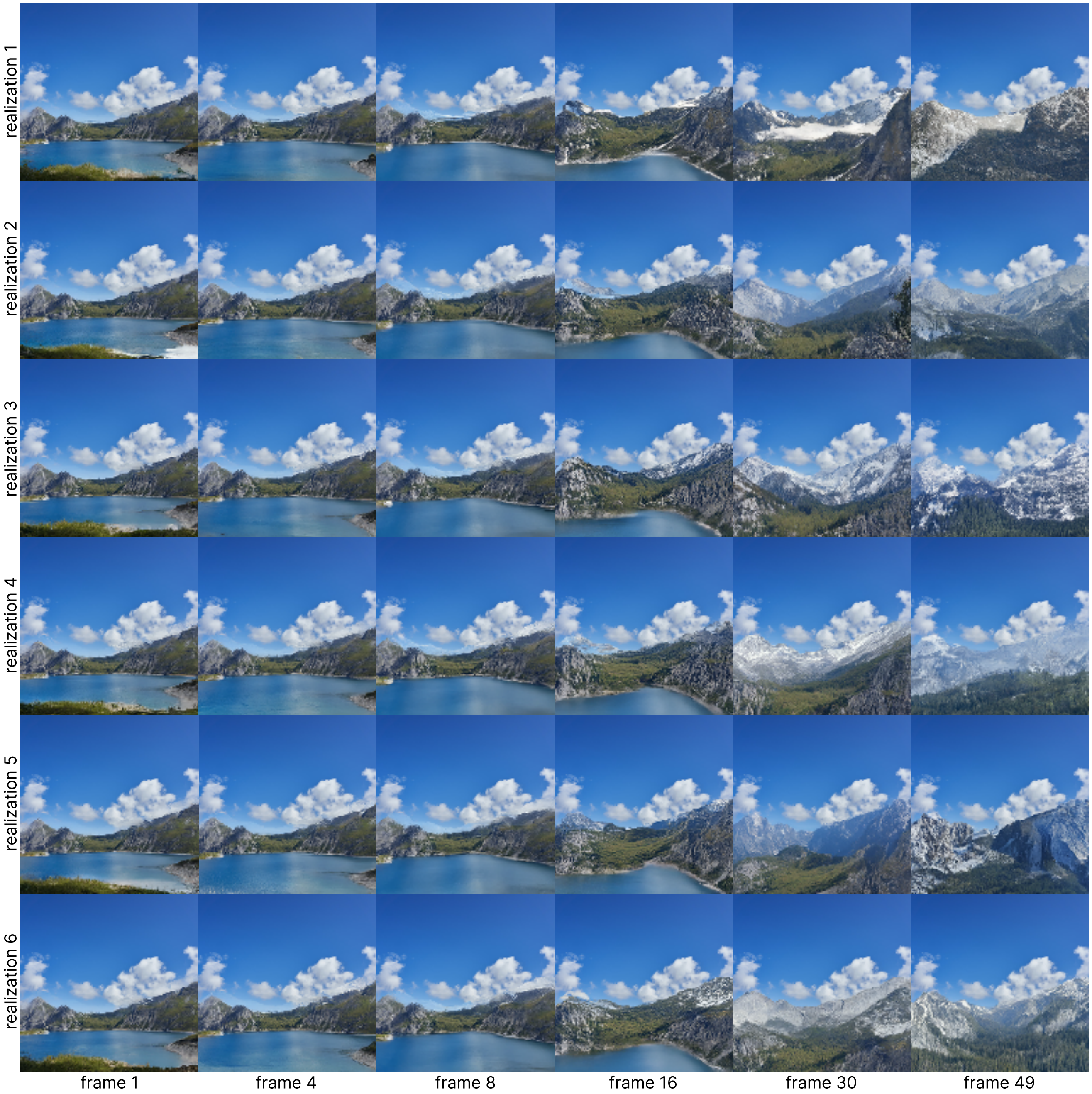


Figure 12: **Additional qualitative results:** Six distinct realizations, synthesized over 50 steps of forward motion.

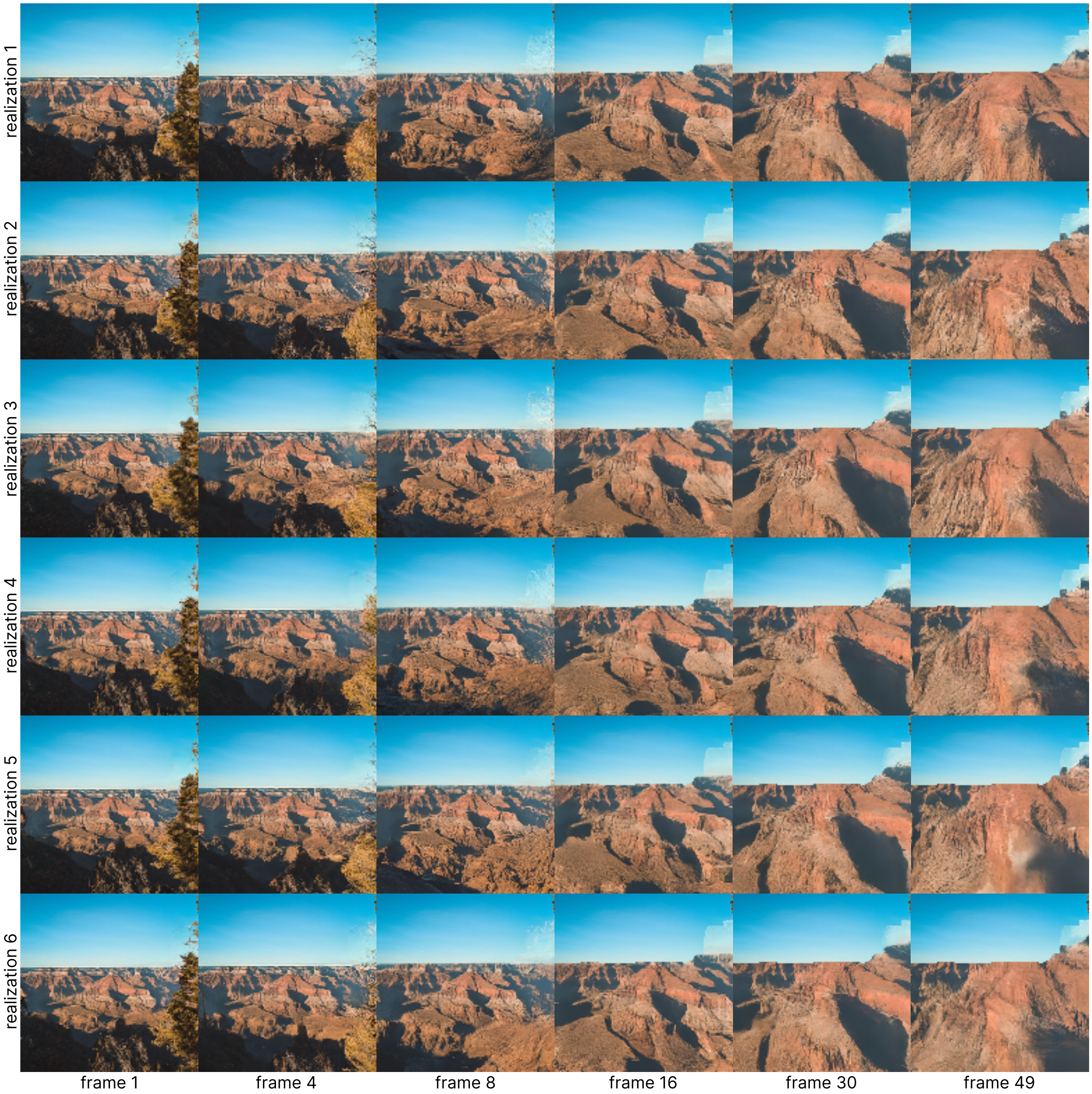


Figure 13: Additional qualitative results: Six distinct realizations, synthesized over 50 steps of forward motion.

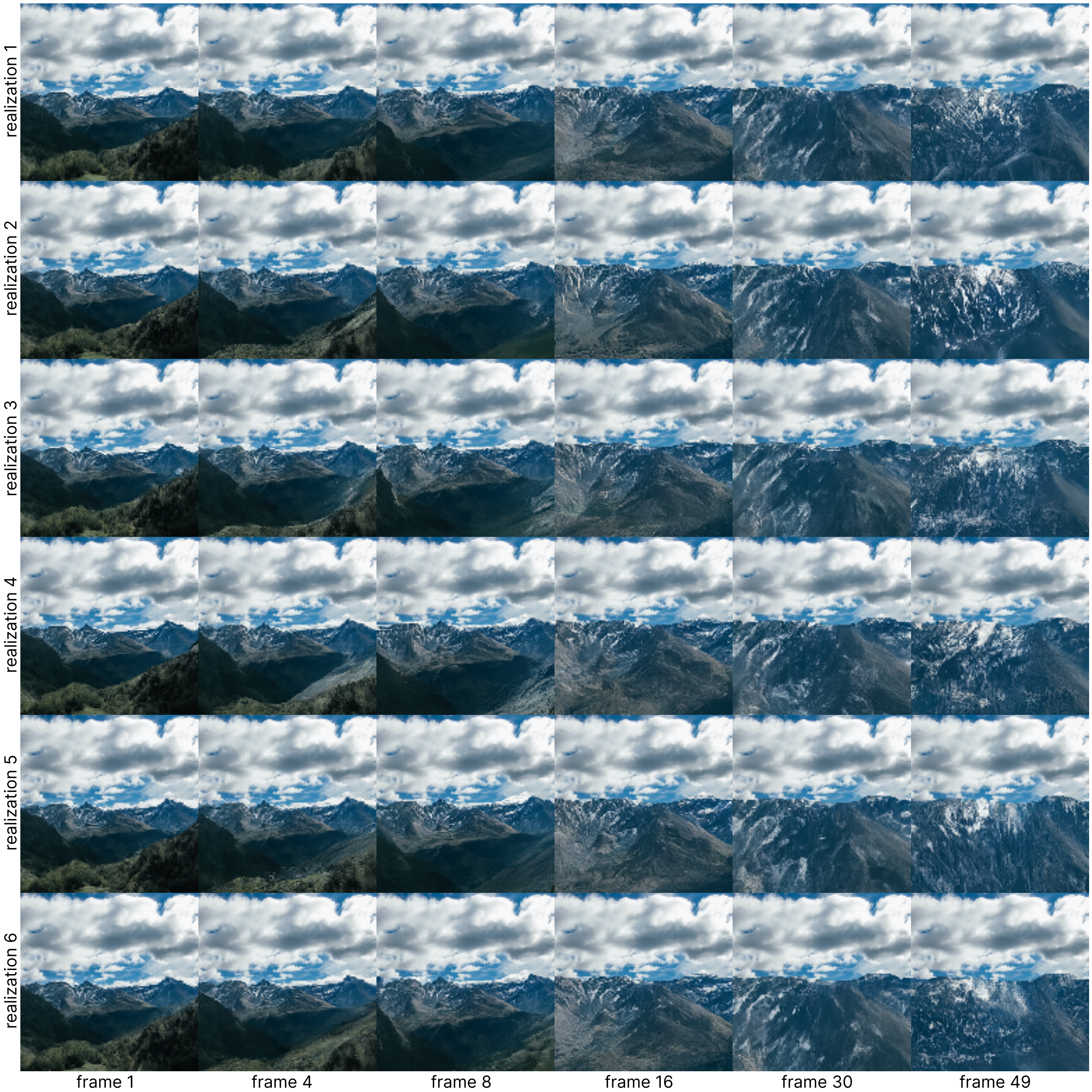


Figure 14: **Additional qualitative results:** Six distinct realizations, synthesized over 50 steps of forward motion. DiffDreamer is able to preserve consistency when there is no significant refinement needed.

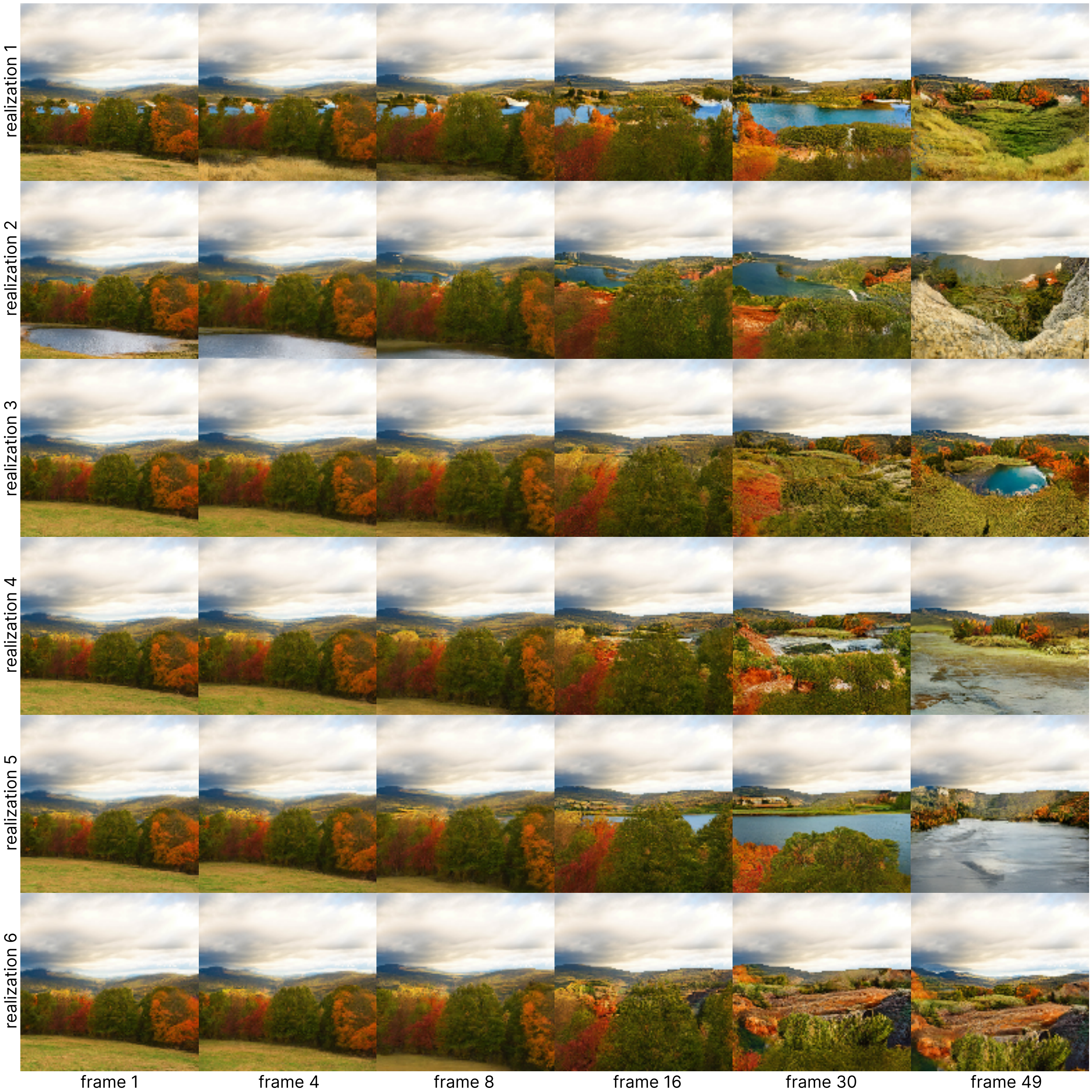


Figure 15: **Additional qualitative results:** Six distinct realizations, synthesized over 50 steps of forward motion, where we encourage output diversity by additionally conditioning on input patterns.