SwiftBrush: One-Step Text-to-Image Diffusion Model with Variational Score Distillation

Supplementary Material

6. Additional Details

Training Cost. When training our model on a single A100 40GB GPU, we can use a batch size of up to 16 and approximately $15,825$ training iterations per day. Specifically, we train the model with $65,000$ gradient updates. Therefore, distilling Stable Diffusion using SwiftBrush requires $65,000/15,825 \approx 4.11$ A100 GPU days. Additionally, even though the original BOOT requires $500,000$ iterations to be fully trained as reported in [9], our re-implementation only needs about $100,000$ to converge, which takes approximately $100,000/18,000 \approx 5.56$ A100 GPU day with a batch size of 16 per GPU.

Inference Speed. Next, we measure the inference time of our method using a batch size of 1 on an A100 40GB GPU. The inference time of one-step UNet to generate the latent from random noise is 25 ms, while those of text encoder and VAE are 80 ms and 5 ms, respectively. Consequently, the total inference time of SwiftBrush is $25 + 80 + 5 = 110$ ms. Furthermore, the speed of the text encoder and VAE in BOOT are the same as ours; however, there is a small overhead in the UNet (30ms) as additional layers are used. Therefore, the total inference time for BOOT is 115ms.

Memory Requirement. Furthermore, we evaluate SwiftBrush's memory consumption during both training (with a batch size of 16) and inference (with a batch size of 1) on an A100 GPU. Since our approach involves training only a student and an additional low-rank LoRA weight for our LoRA teacher, it maintains a minimal memory footprint.

Compare with Others. We also sum up inference speed, training time and memory requirements of existing methods and ours in Tab. 3. In terms of training costs, Guided Distillation and InstaFlow [21] require a considerable amount of training time, limiting their practical use. Conversely, LCM converges quickly but produces poor-quality images with one-step inference. Our method, however, offers both efficiency and high-quality output.

7. Additional Visual Results

Uncurated Samples. We display uncurated images conditioned on 25 random text prompts*. The images generated by various models are presented in Fig. 8, 9, 10 and 11 for a better visual comparison between our model and others.

*https://thuanz123.github.io/swiftbrush/assets/prompts.txt

Method	Training		Inference	
	Time (GPU days)	Memory (GB)	Time (ms)	Memory (GB)
Guided Distillation	108.8†	-	-	-
LCM	1.3†	33.6*	118*	11.3*
InstaFlow	199.2†	-	116*	12.3*
BOOT‡	5.6	30.2	115	8.3
Ours	4.1	26.4	110	8.2

Table 3. Comparison of our method against other works on inference and training time/memory. † means that we obtain the numbers from the corresponding papers. * means that we obtain the numbers using the official code of the corresponding papers. ‡ means that we re-implement the work and report the numbers. The units for inference speed, training time and memory usage are miliseconds, A100 GPU days and gigabytes, respectively.

Additional Visuals of Analysis. To further back up our claim in Sec. 4.3, we provide some uncurated samples using the text prompt *"An DSLR photo of a tiger in the city"*. The images generated by four configurations at every 2000 iterations are presented in Fig. 14 for a better visual comparison of each component's importance in our proposed method across different training stages.

Additional Visuals of Interpolation. Apart from investigating the role of each component in SwiftBrush, we also explore its properties when interpolating text prompt condition y (Fig. 12) or noise input z (Fig. 13). It is evident that our model generates a seamless transition in the output, showcasing excellent editability and controllability.

8. Additional Quantitative Results

We run SwiftBrush on CIFAR-10 32×32 and class-conditional ImageNet 64×64 benchmarks using EDM teacher models and compare it with Progressive Distillation (PD) and Consistency Distillation (CD), either using L2 or LPIPS loss, in Tab. 4. SwiftBrush is on par with CD-LPIPS and it significantly outperforms the others.

Method	Steps	CIFAR-10 32×32	ImageNet 64×64
PD	1	8.34	15.39
CD - L2	1	7.90	12.10
CD - LPIPS	1	**3.55**	<u>6.20</u>
Ours	1	<u>4.46</u>	**5.85**
EDM	35	1.97	-
EDM	79	-	2.44

Table 4. Comparison of our method against other works in FID on CIFAR-10 (32×32) and ImageNet (64×64) benchmarks.

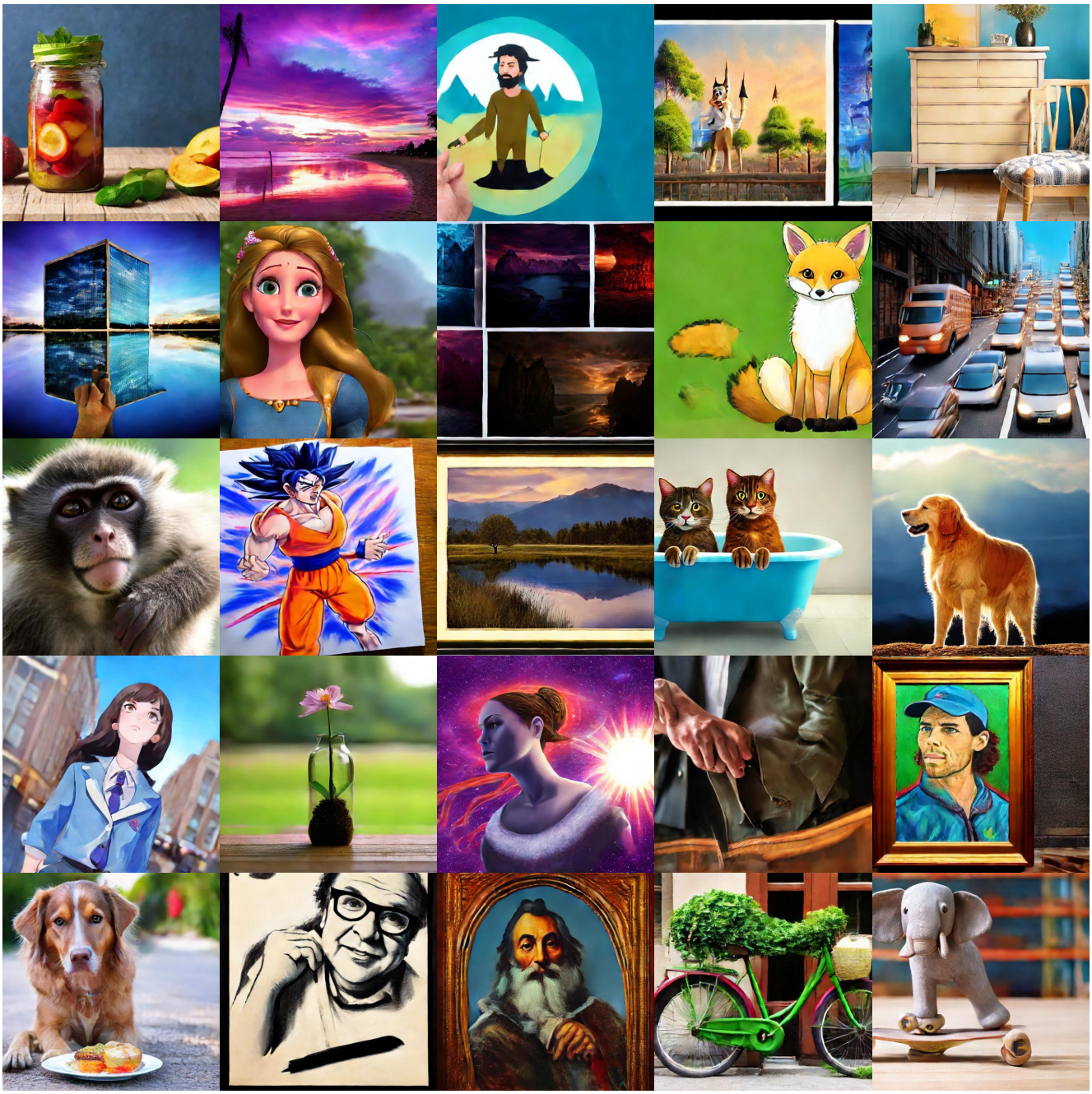

Figure 8. Uncurated samples from one-step **SwiftBrush**.

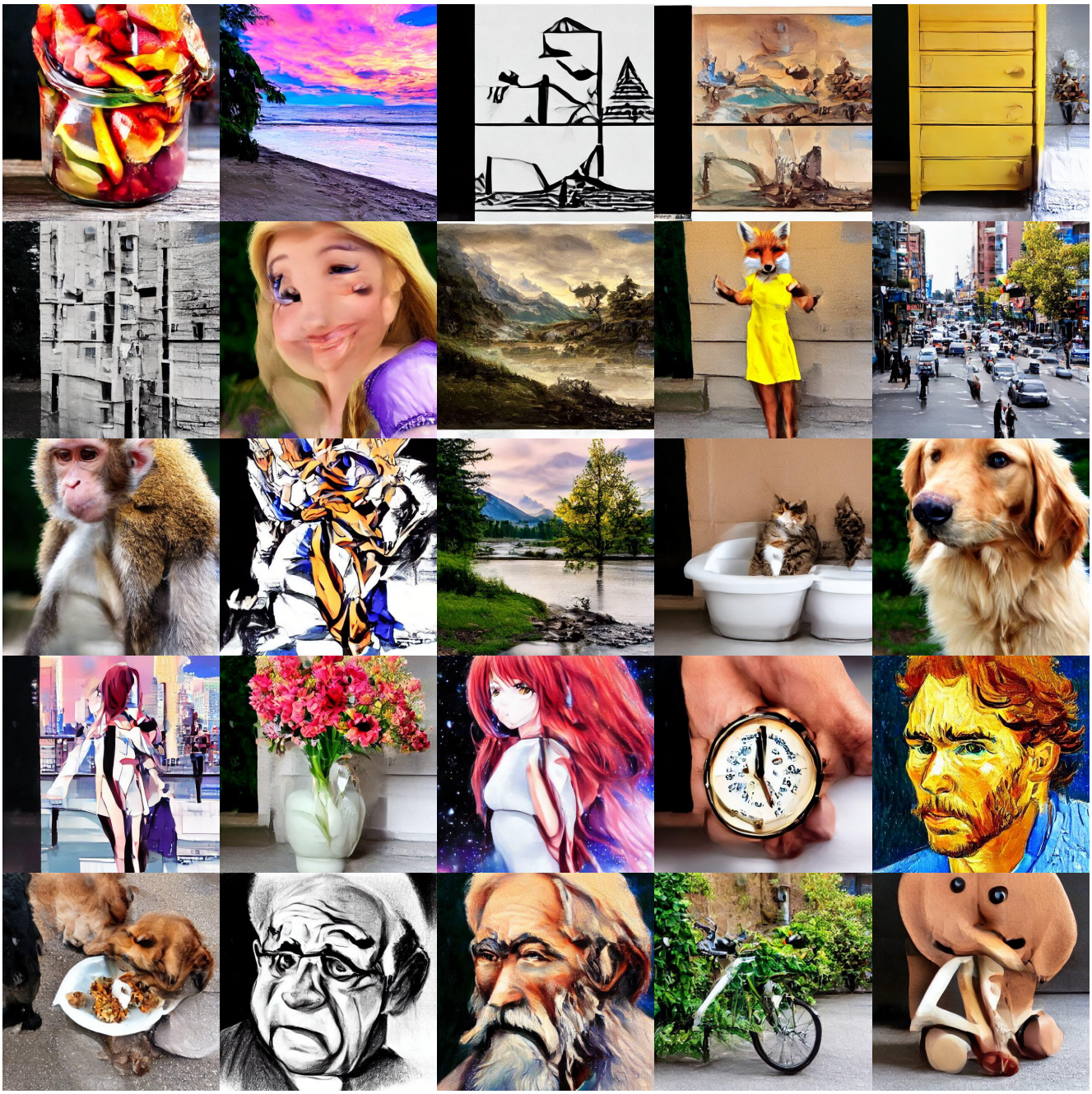

Figure 9. Uncurated samples from one-step **InstaFlow**. We use their provided pretrained model to generate.

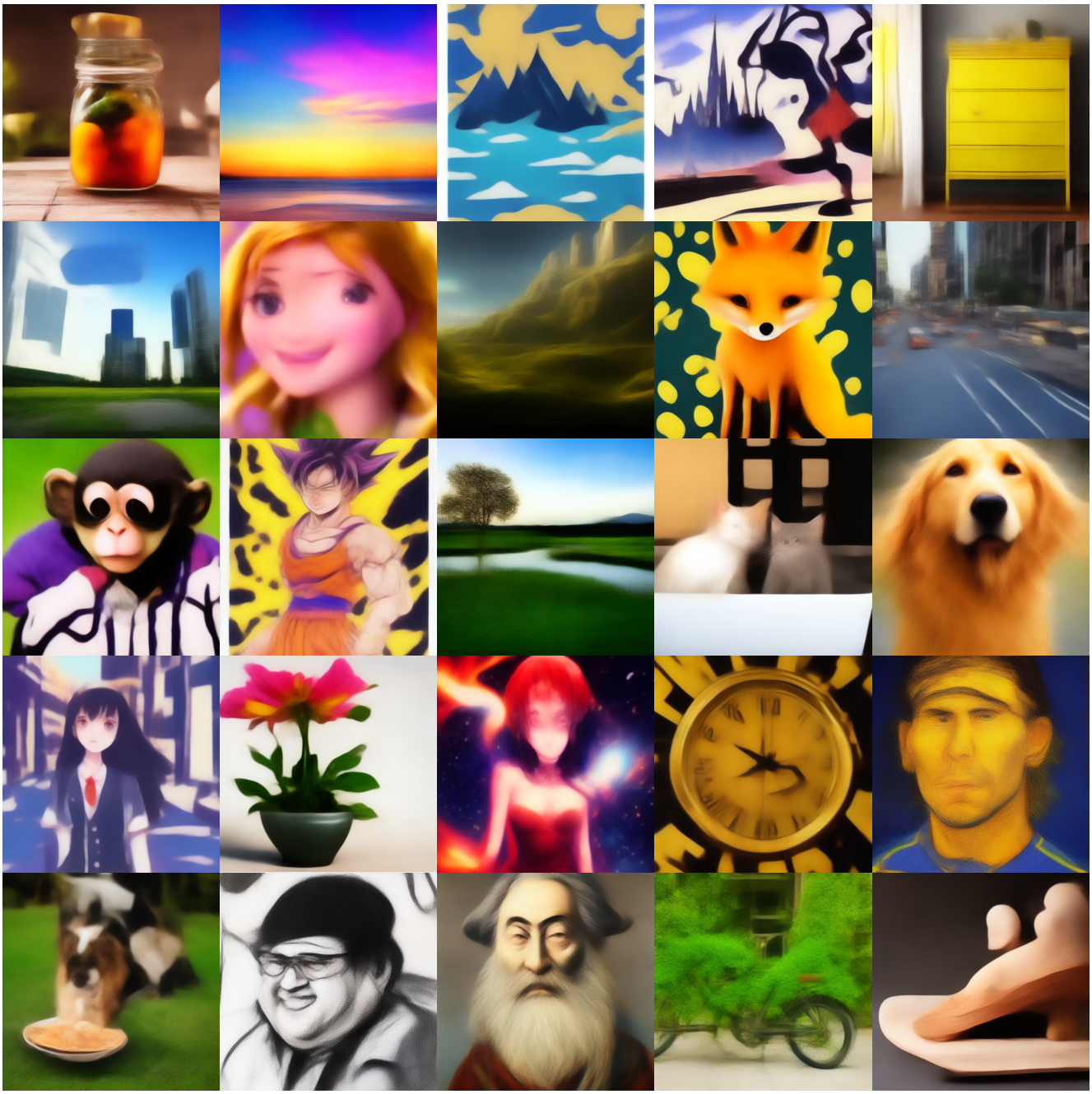

Figure 10. Uncurated samples from one-step **BOOT**. We re-implement and generate the images by ourselves.

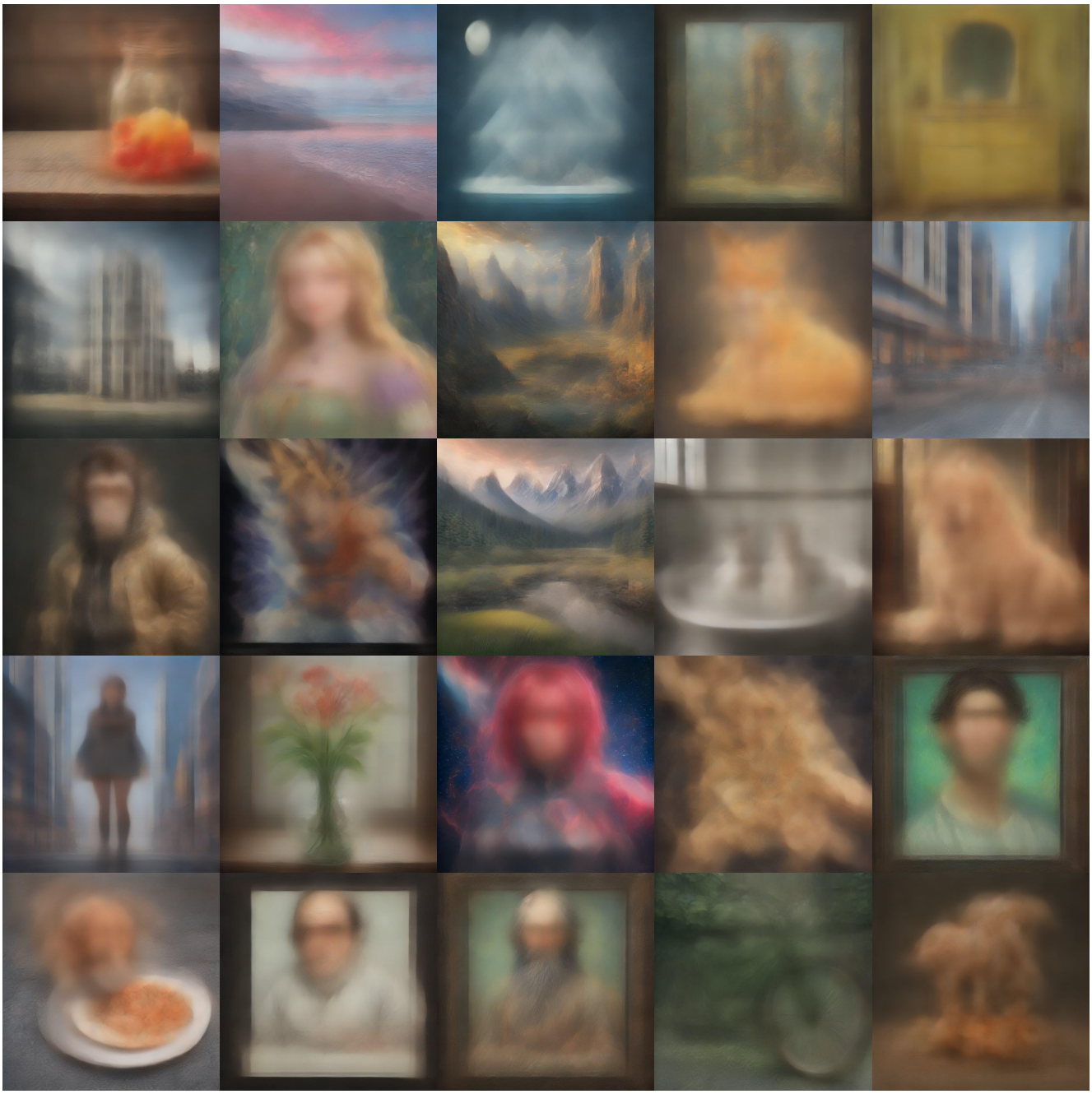

Figure 11. Uncurated samples from one-step **LCM**. We use their provided pretrained model to generate.

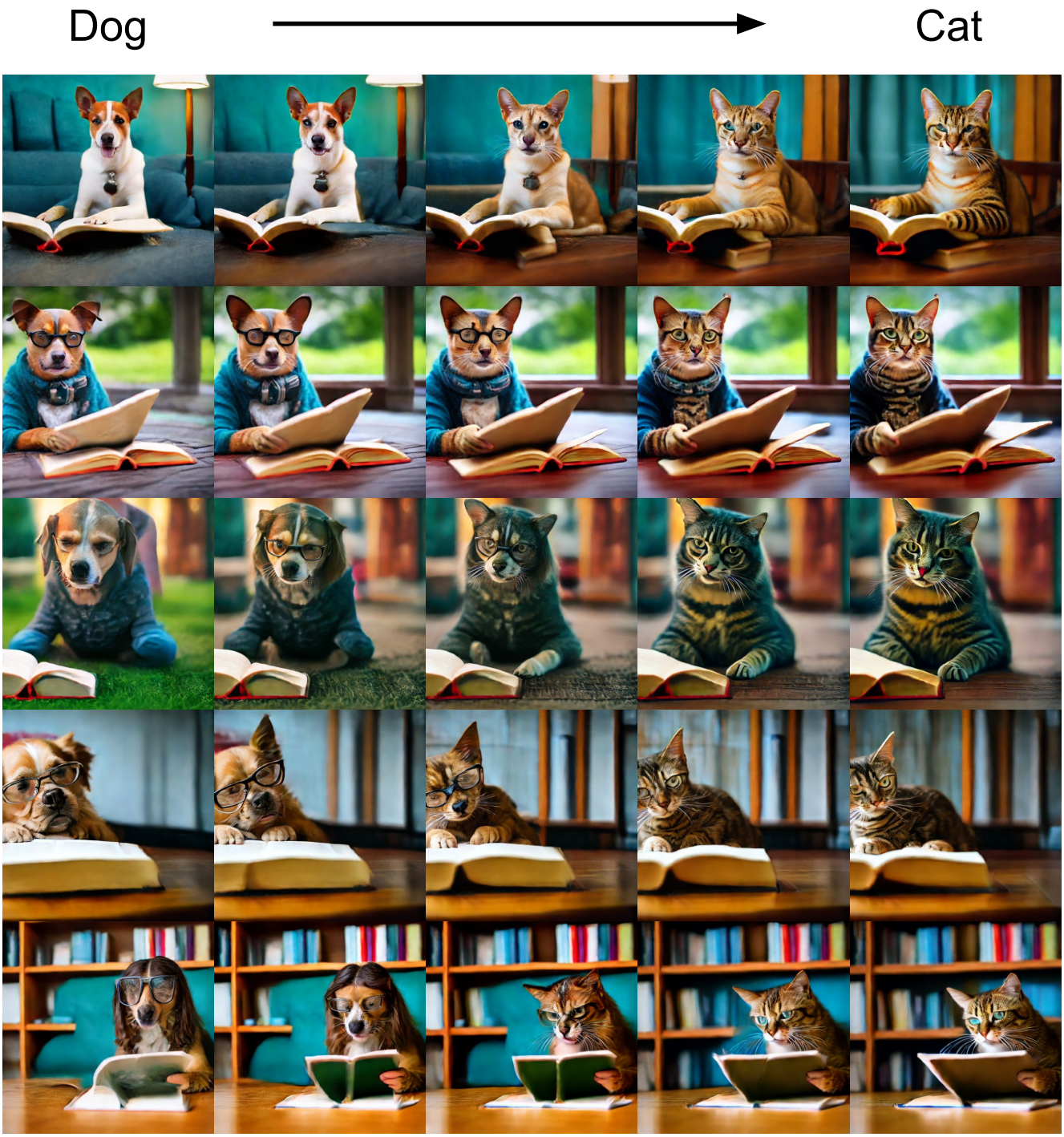

Figure 12. Results of interpolating the input prompt. The prompts used here are selected from a standard template: *"A DSLR photo of a {animal} reading a book"*. Here, 'animal' is dog or cat and we interpolate the text embedding using linear interpolation (Lerp). Same noise input z is used for images at each row.

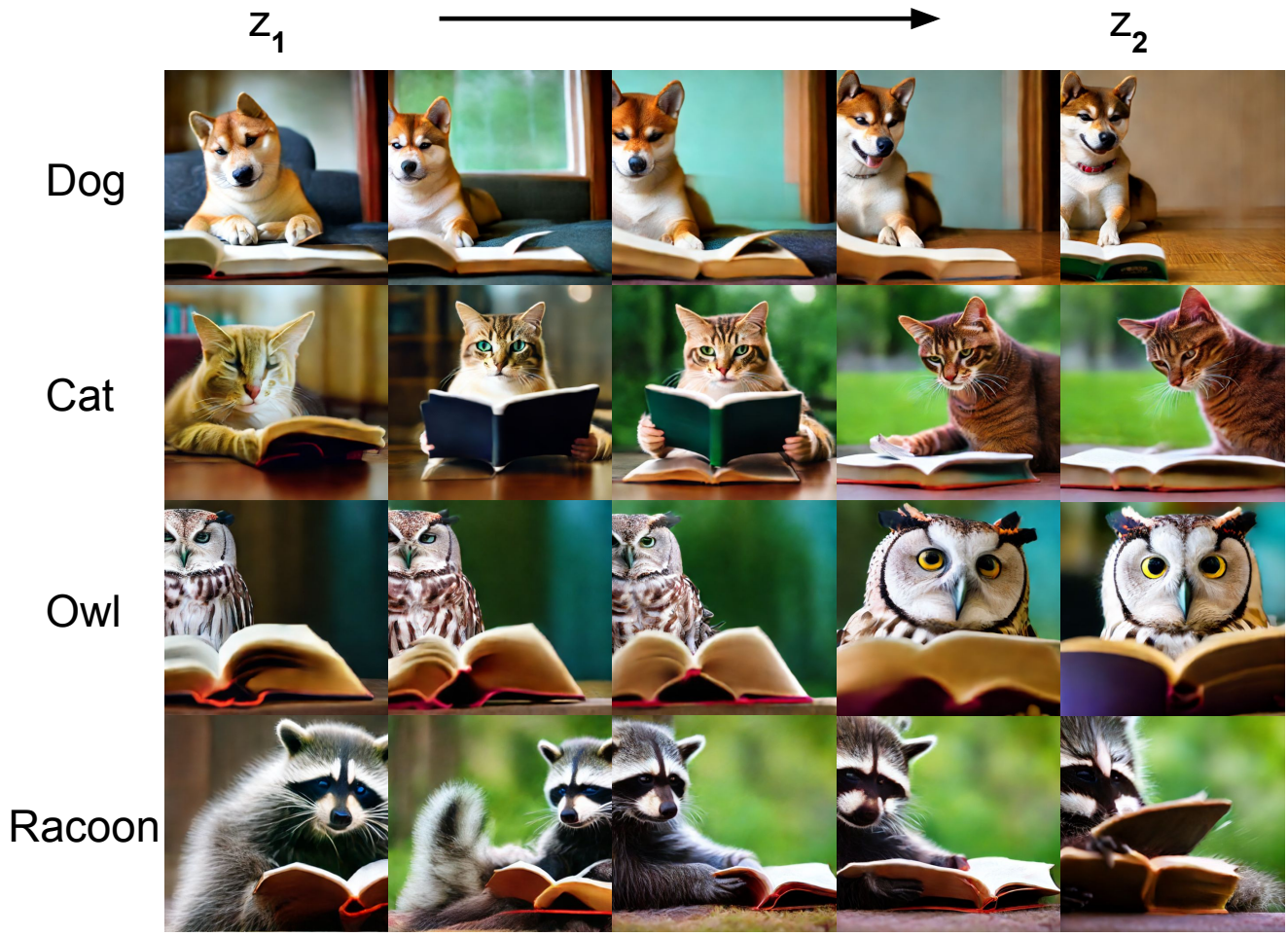

Figure 13. Results of interpolating the noise input. The prompts used here are selected from a standard template: *"A DSLR photo of a* {*animal*} *reading a book"*. Here, 'animal' is dog, cat, owl or racoon and we interpolate the noise input using spherical linear interpolation (Slerp). Same text input y is used for images at each row.

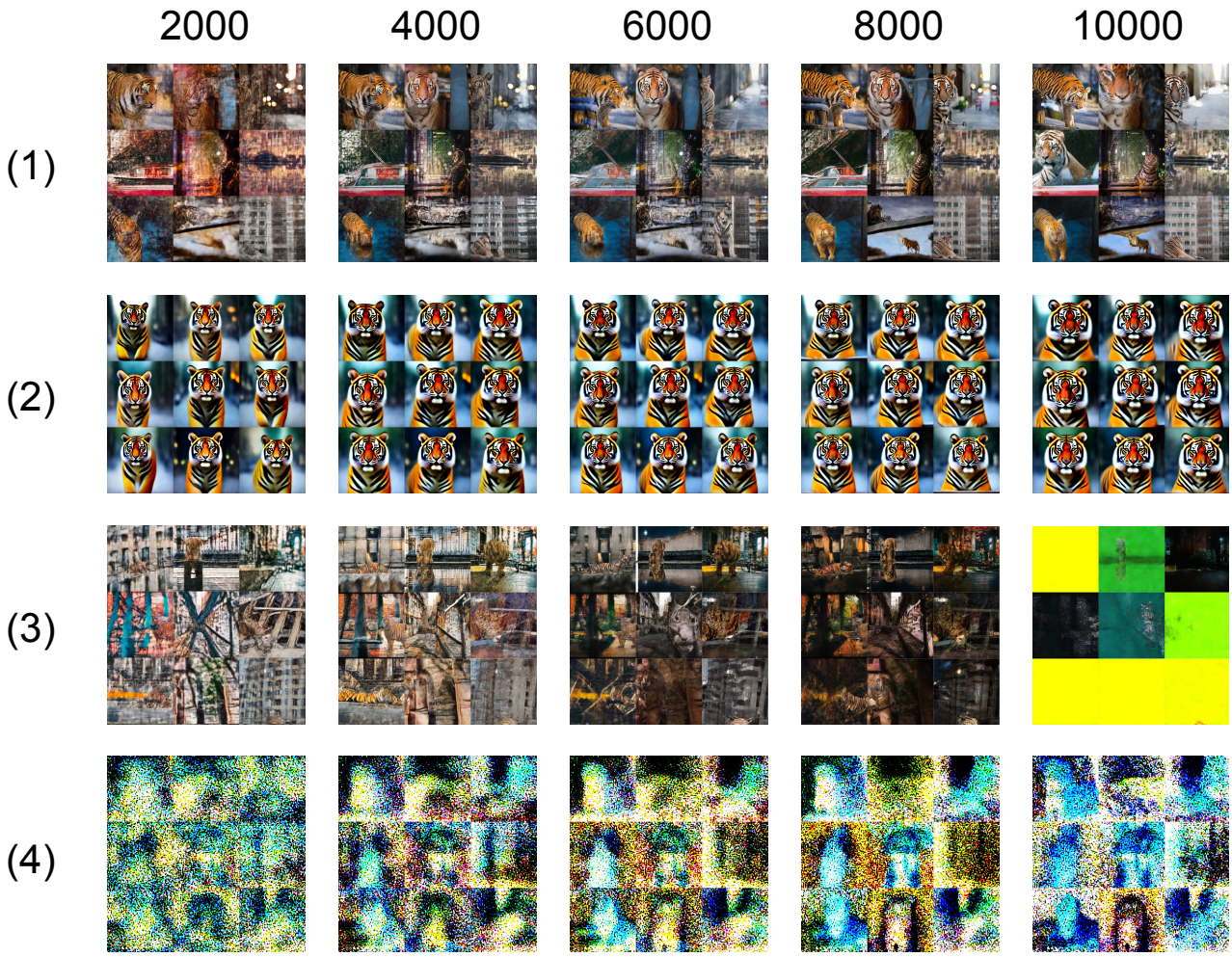

Figure 14. Visual results of the ablative study, where the all images are generated with the input prompt *"An DSLR photo of a tiger in the city"*. Images at same column are generated at the same iteration of training, whereas those at same row are generated using the same configuration. Here, (1) means "Full", (2) means "SDS", (3) means "Small-Rank LoRA" and (4) means "No Parameterization".