Diffusion-based Visual Anagram as Multi-task Learning
– Supplementary Material –

Zhiyuan Xu[*1,2], Yinhe Chen[*1,3], Huan-ang Gao[1], Weiyan Zhao[1,4], Guiyu Zhang[1], Hao Zhao[†1]
[1]Institute for AI Industry Research (AIR), Tsinghua University
[2]University of Chinese Academy of Sciences
[3]Central South University, [4]Beijing Institute of Technology

1. Computation Efficiency

In this section, we discuss the computational complexity of existing methods and our proposed approach for visual anagram generation.

Experimental Setup. We conduct the experiments on a server equipped with two AMD EPYC 7742 CPUs and 1TB RAM, running CUDA 12 and PyTorch 2.1.1, and each method only uses a single NVIDIA RTX 3090 GPU at a time. For Burgert *et al*. [1] and Tancik [5], number of iterations and number of inference steps per image are set to 10,000 and 500, respectively, which are the default values in their open-source code. For Geng *et al*. [2] and our proposed method, the number of inference steps is fixed at 30. These settings are consistent with the main text. Note that fine-tuning backbone diffusion models is not required for all methods, and we report the average computation time per image for each method in Tab. 1. The reported time exclude model and data loading time, and all methods are evaluated using the 2-view CIFAR10 dataset as in the main text.

Results. As shown in Tab. 1, among all tested methods, Burgert *et al*. [1] has the longest computation timedue to its use of Score Distillation Loss (SDL) [3], which involves a large number of iterative optimization steps. The other three methods run significantly faster, as they operate within the typical time frame of diffusion model inference. Our proposed method is slightly slower than our baseline method [2], as it incorporates additional modules to enhance visual anagram quality. Tancik [5] takes slightly more time than our method, primarily because it employs a latent diffusion model [4] where the latent code is not rotation-invariant, and therefore requires more inference steps to generate the final image.

Method	Computation Time (sec/img)
Tancik [5]	27.1
Burgert *et al*. [1]	3016.0
Visual Anagrams [2]	13.4
Ours	20.2

Table 1. **Computation Complexity.** We report the average computation time per image for each method.

2. Additional Qualitative Results

We provide additional qualitative results to compare our method with existing methods [1, 2, 5] in Fig. 1, including samples generated on prompts from the 2-view CIFAR10 and 3-view CIFAR10 datasets mentioned in the main text, together with some free-form examples.

References

[1] Ryan Burgert, Xiang Li, Abe Leite, Kanchana Ranasinghe, and Michael Ryoo. Diffusion Illusions: Hiding Images in Plain Sight. In *ACM SIGGRAPH 2024 Conference Papers*, SIGGRAPH ’24, pages 1–11, New York, NY, USA, July 2024. Association for Computing Machinery. 1, 2

[2] Daniel Geng, Inbum Park, and Andrew Owens. Visual anagrams: Generating multi-view optical illusions with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24154–24163, 2024. 1, 2

[3] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations*, Sept. 2022. 1, 2

[4] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. https://arxiv.org/abs/2112.10752v2, 2021. 1, 2

[5] Matthew Tancik. tancik/Illusion-Diffusion, July 2024. original-date: 2023-02-12T22:39:28Z. 1, 2

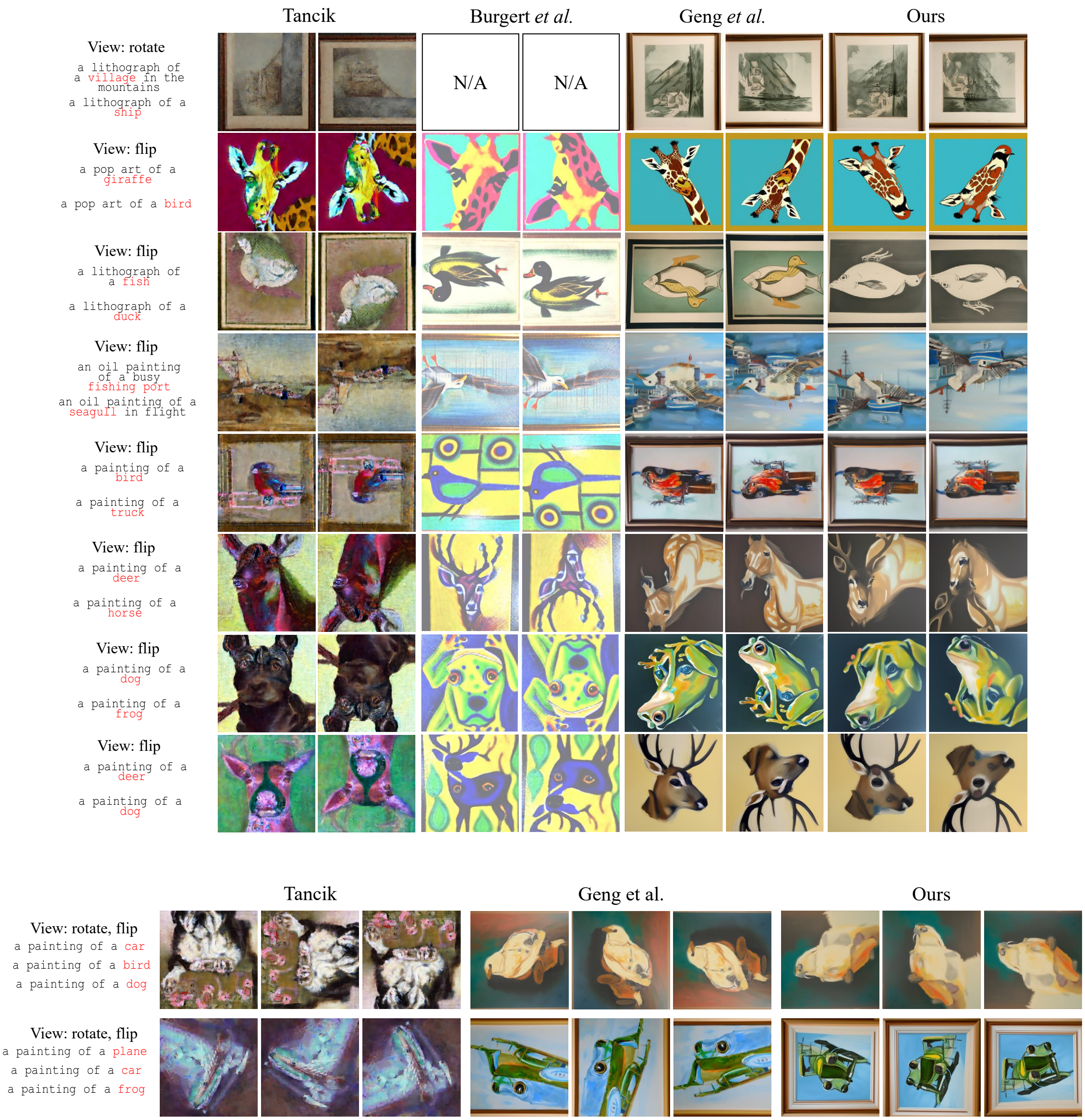

Figure 1. **Qualitative Results.** We provide additional qualitative results to compare our method with existing methods. Tancik [5] uses a latent diffusion model [4] but struggles with transformation inconsistencies in the latent code, as discussed in [2]. Burgert *et al.* [1] employs Score Distillation Loss (SDL) [3] which requires expensive iterative optimization and results in reduced image quality. Geng *et al.* [2] also encounters issues with concept segregation and concept domination. For Burgert *et al.* [1], we only present results for 2-view flippy visual anagram generation, as their released code does not support other configurations.