

Supplementary Materials: Auto DragGAN: Editing the Generative Image Manifold in an Autoregressive Manner

Anonymous Author(s)

Submission Id: 471

A DETAILS OF NETWORK ARCHITECTURE

The mapping network of StyleGAN2 [3] randomly samples a 512-dimensional latent code z from a normal distribution and maps it to a latent code w of dimension $l \times 512$, where l represents the number of layers in the generator network. These mapped latent codes serve as training samples for the Latent Regularizer. To obtain the outlier latent code, we introduce random noise to the randomly sampled latent code w . In the first stage of training, we initially add noise to the first six layers of w . Specifically, we perform a masking operation on the first six layers of w , randomly setting the vector values of these layers to zero with a 25% probability, followed by the addition of Gaussian noise. As illustrated in Figure 1, the Latent Regularizer structure adopts a standard transformer architecture. The noisy latent code w' is divided into two sets of vectors: the noisy vectors w'_1 from the first six layers and the remaining clean vectors w'_2 . The transformer encoder receives w'_1 . Meanwhile, the transformer decoder accepts two inputs: first, the output of the transformer encoder; and second, the element-wise sum of w'_2 mapped through an MLP and the position embedding. The output of the transformer decoder is concatenated with w'_2 to form the reconstructed latent code \hat{w} .

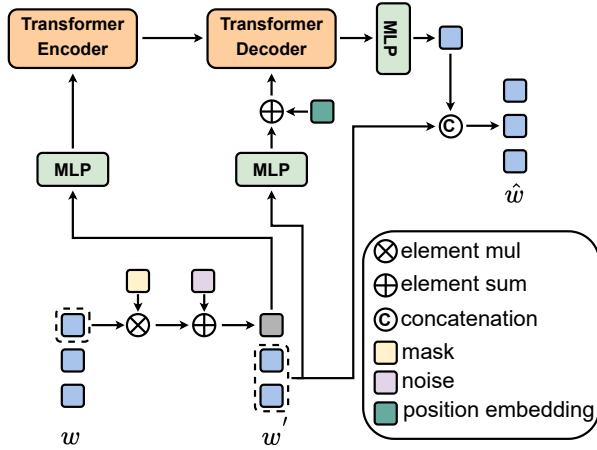


Figure 1: The network architecture of the Latent Regularizer.

For a randomly sampled w_0 , perform n times of slight perturbation to obtain the latent code motion sequence w_0, w_1, \dots, w_n as training samples. As illustrated in Figure 2, the Latent Predictor employs a straightforward teacher-forcing cross-attention Transformer Decoder [7] for motion sequence prediction. F_0 is the intermediate feature of w_0 . The feature vector sequence and patch vector sequence extracted from F_0 , after undergoing mapping and

concatenation, are used as the input for the transformer encoder. Meanwhile, the transformer decoder accepts two inputs: first, the output of the transformer encoder; and second, the element-wise sum of the latent code motion sequence w_0, w_1, \dots, w_{n-1} after projection and the position embedding. The output of the transformer decoder, after being mapped through an MLP, serves as the predicted latent code motion sequence $\hat{w}_0, \hat{w}_1, \dots, \hat{w}_n$.

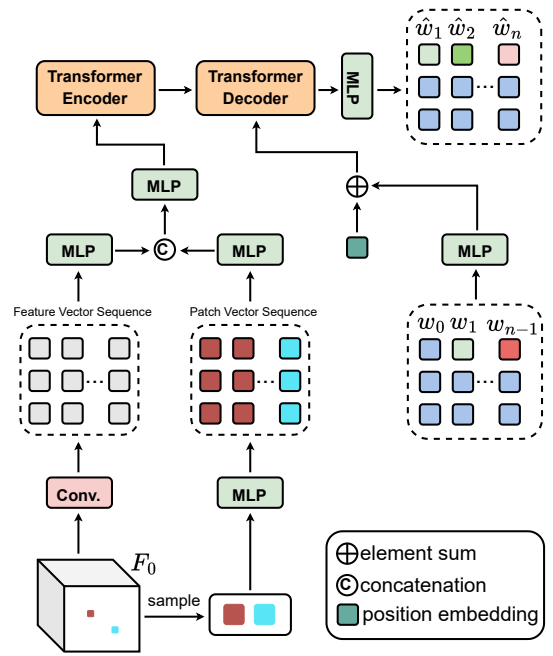


Figure 2: The network architecture of the Latent Predictor.

B MORE QUALITATIVE EVALUATIONS

We present more qualitative comparisons between our method and DragGAN [5] in terms of inference speed and image editing performance. Figure 3 presents the results on the AFHQCat [1] dataset. Figure 4 presents the results on the FFHQ [2] dataset. Figure 5 presents the results on the Landscapes HQ [6] dataset. Figure 6 presents the results on the Self-distilled StyleGAN Dog [4] dataset.

The results of these qualitative comparisons demonstrate that our method not only converges more rapidly than DragGAN [5], but also surpasses it in both inference speed and image editing performance.

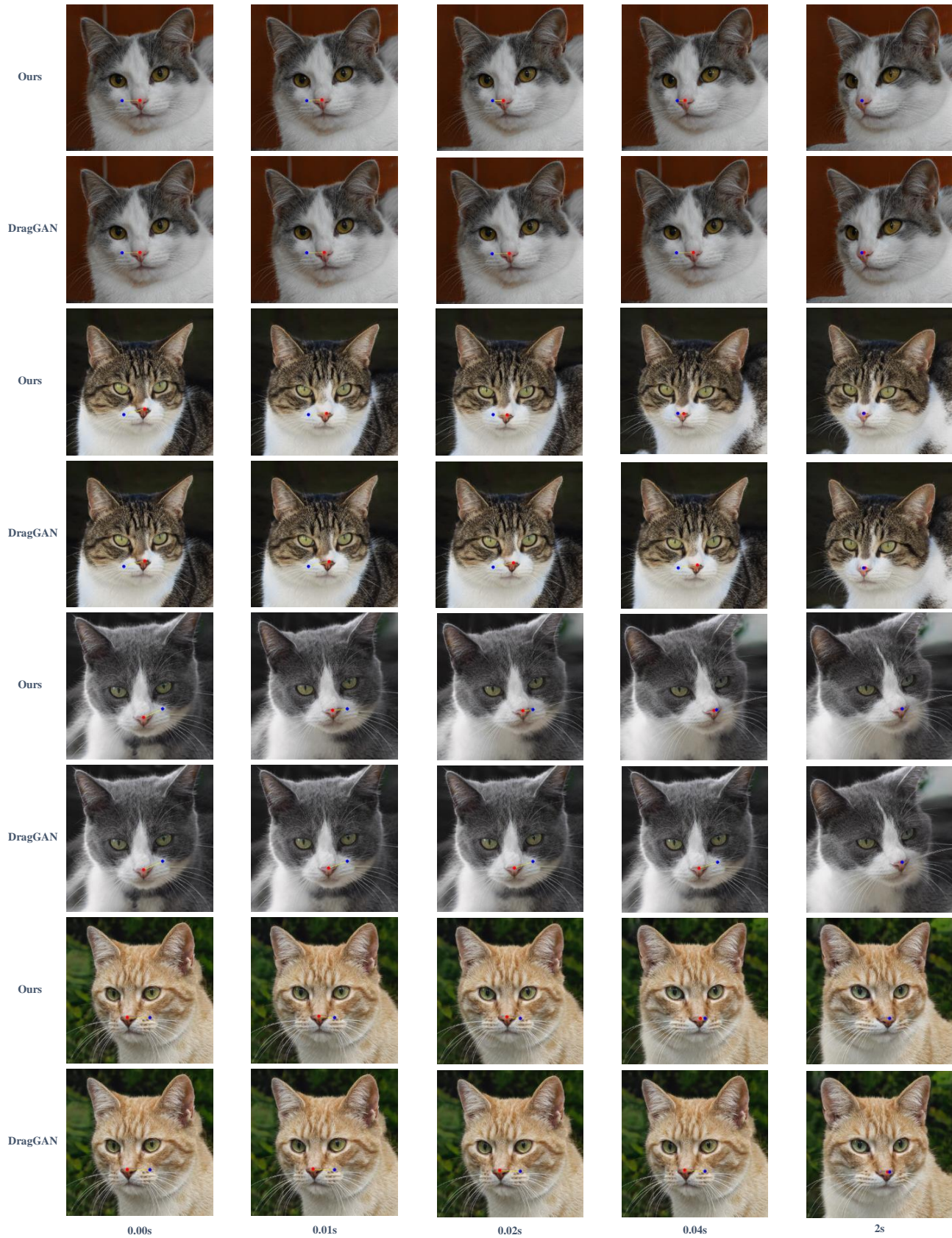


Figure 3: A qualitative comparison between our method and DragGAN [5] on the AFHQCat [1] dataset in terms of inference speed and image editing performance.

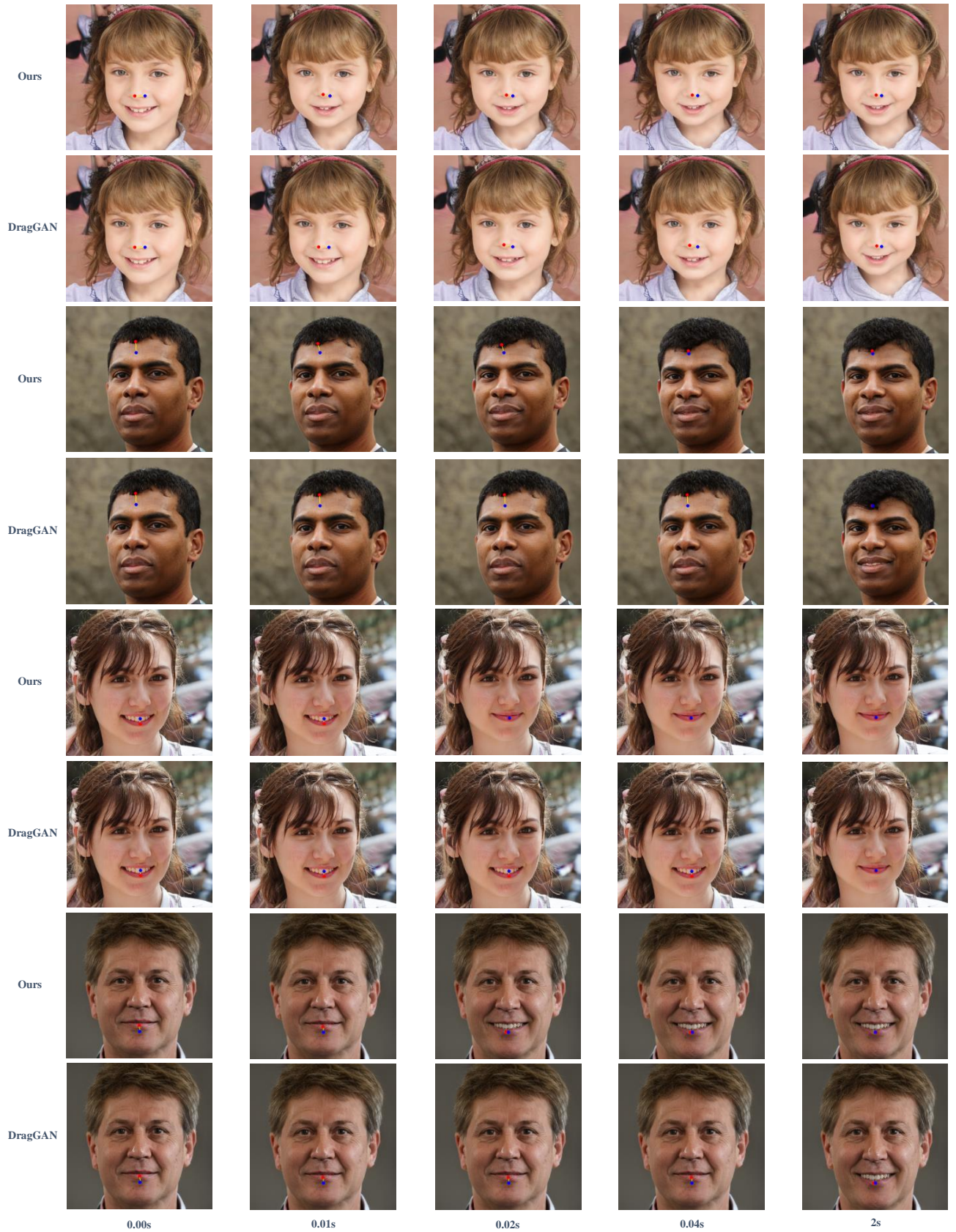


Figure 4: A qualitative comparison between our method and DragGAN [5] on the FFHQ [2] dataset in terms of inference speed and image editing performance.

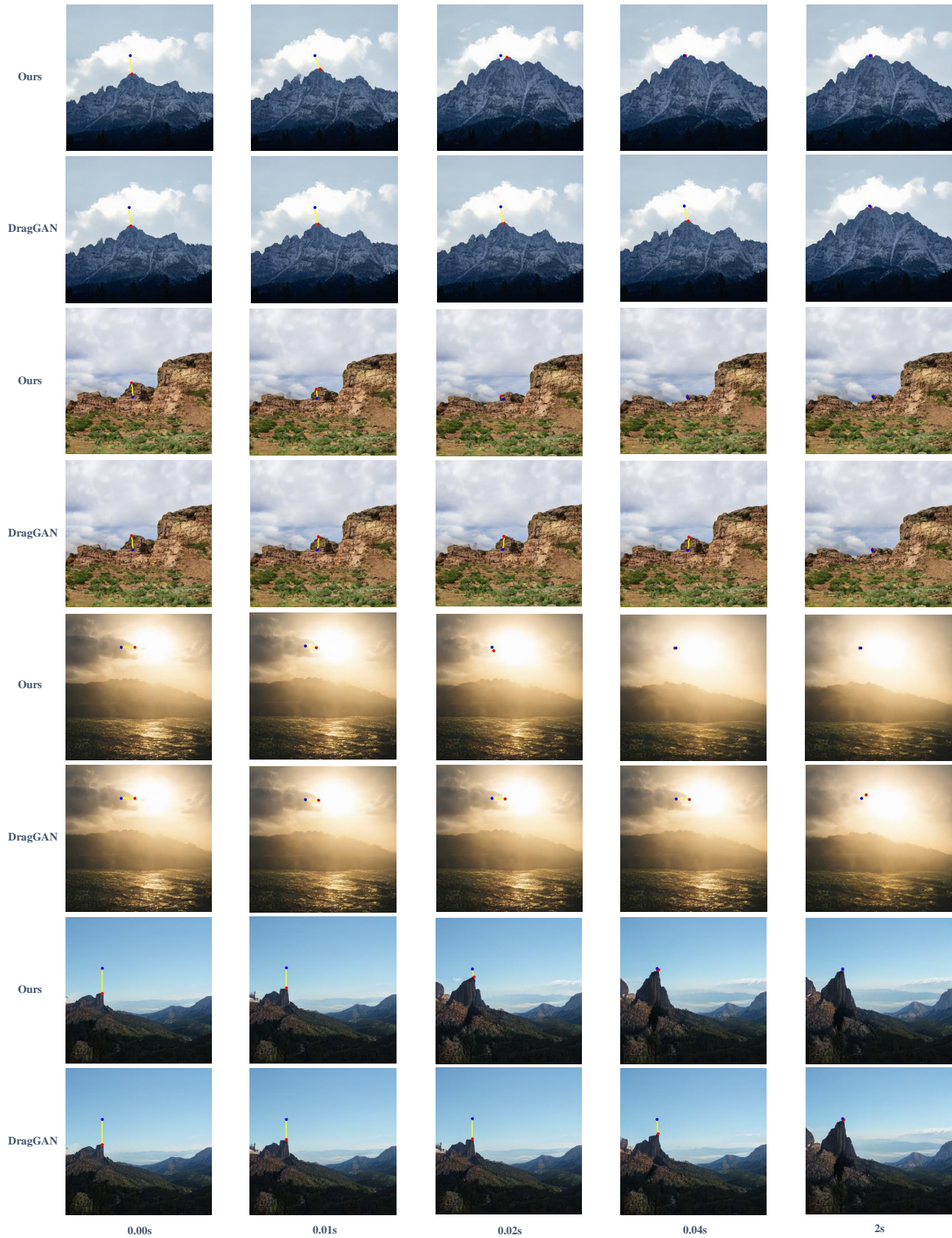


Figure 5: A qualitative comparison between our method and DragGAN [5] on the Landscapes HQ [6] dataset in terms of inference speed and image editing performance.

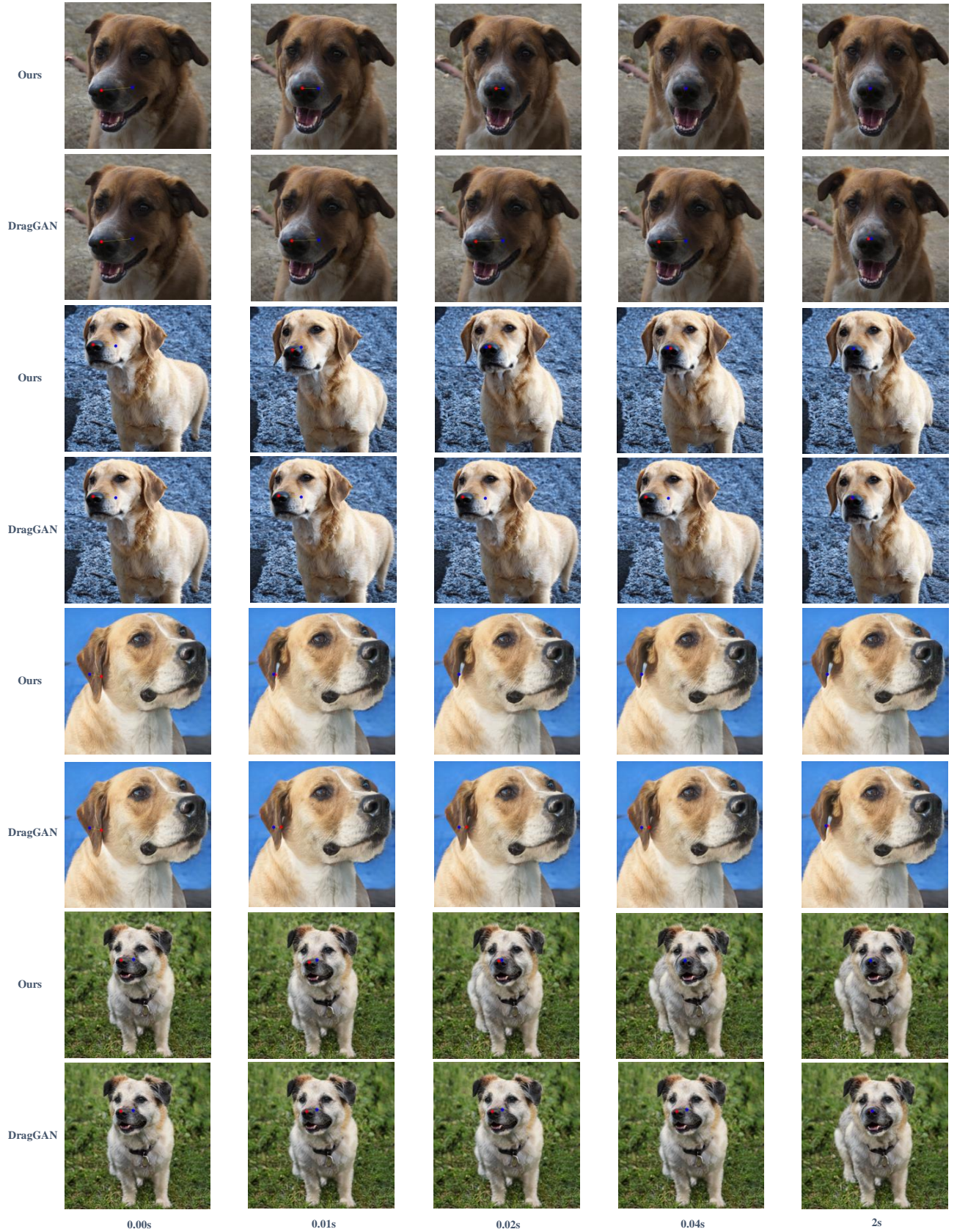


Figure 6: A qualitative comparison between our method and DragGAN [5] on the Dog [4] dataset in terms of inference speed and image editing performance.

REFERENCES

- [1] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. 2020. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8188–8197.
- [2] Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4401–4410.
- [3] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8110–8119.
- [4] Ron Mokady, Omer Tov, Michal Yarom, Oran Lang, Inbar Mosseri, Tali Dekel, Daniel Cohen-Or, and Michal Irani. 2022. Self-distilled stylegan: Towards generation from internet photos. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.
- [5] Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. 2023. Drag your gan: Interactive point-based manipulation on the generative image manifold. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.
- [6] Ivan Skorokhodov, Grigorii Sotnikov, and Mohamed Elhoseiny. 2021. Aligning latent and image spaces to connect the unconnectable. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14144–14153.
- [7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).