Lossy Image Compression with Foundation Diffusion Models – Supplementary Material –

Lucas Relic[1,2], Roberto Azevedo[2], Markus Gross[1,2], and Christopher Schroers[2]

[1] ETH Zürich, Switzerland
{lucas.relic, grossm}@inf.ethz.ch
[2] Disney Research | Studios, Zürich, Switzerland
{roberto.azevedo, christopher.schroers}@disneyresearch.com

A Additional Qualitative Results

Figs. 5 - 11 show additional qualitative comparisons between our method and the baselines from the main text.

We also show full image comparisons between the diffusion-based image compression baselines [6, 10] and strong autoencoder-based [5] and traditional [1] image codec baselines in Figs. 12 - 16.

B Additional Implementation Details

Foundation Model Backbone. We take Stable Diffusion v2.1 as our foundation model backbone, using the official code repository[3] and model checkpoint[4]. We use the default configuration, except for minor modifications detailed below.

We utilize the DDIM sampling formulation for the diffusion generative process. As mentioned in Sec. 4.4, it is prohibitively expensive to accumulate gradients over multiple DDIM steps during training of our method, therefore we slightly modify the sampling procedure as follows: Each DDIM step computes not only the previous partially denoised data $\mathbf{x}_{t-1}$ but also a prediction of the fully denoised data $\tilde{\mathbf{x}}_0$ as an intermediate variable (see Eq. (2)). Thus instead of running multiple DDIM steps to produce $\{\mathbf{x}_{t-1}, \mathbf{x}_{t-2}, ..., \mathbf{x}_0\}$, we perform one DDIM step and directly utilize $\tilde{\mathbf{x}}_0$ as the fully denoised data. Although $\tilde{\mathbf{x}}_0$ is slightly inconsistent with $\mathbf{x}_0$, for the small values of t where our method operates we observe that the difference is minor (see Fig. 1) and has negligible effect on optimization.

Stable Diffusion offers multiple prediction parameterizations, such ϵ or v prediction, which dictates the output of the diffusion model. We utilize ϵ prediction due to our modifications of the sampling algorithm during training. While the

[3] https://github.com/Stability-AI/stablediffusion
[4] https://huggingface.co/stabilityai/stable-diffusion-2-1/blob/main/v2-1_768-ema-pruned.ckpt

commonly used v prediction is more stable over the full generation process compared to ϵ prediction, in the low timestep range where our model operates v prediction reduces almost completely to ϵ prediction [9] and thus is effectively equivalent.

Train **Inference** **Difference**

Fig. 1: Comparison of the predicted original sample during training (single diffusion step), at inference time (multiple diffusion steps), and the difference between the two (shown from left to right, respectively). Best viewed digitally.

Entropy Model. Entroformer [8] is used as the entropy model for our method. We refer readers to the original paper and codebase[5] for specific information. We follow the architectural details in Appendix A.4 of the original paper, but replace the encoder and decoder with the Stable Diffusion VAE encoder and decoder. Correspondingly, we also change the output dimension of the entropy model to match the channel dimension of the Stable Diffusion latent space (*i.e.* configuration parameter `last_channels` $= 4$).

C Similarity of Quantization Error and Noise

In signal processing, quantization error has historically been modeled as uniform [4] noise. Ballé *et al.* [2] first introduced this to the field of neural compression and since then it has been widely adopted in other neural image compression methods [3, 5, 7, 8].

Intuitively, given a quantization bin with lower and upper bounds a and b, respectively, all values within this range are mapped to the bin center $c = \frac{(b-a)}{2}$ during quantization. Thus, assuming a smooth distribution of values before quantization (as is the case in the latent space of a variational autoencoder) and sufficiently narrow bin range, the error for values within one bin is approximately uniformly distributed across the bin width (*i.e.* $\mathcal{U}(a-c, b-c)$). Given a constant bin width, the error across all quantization bins is therefore also uniformly distributed.

[5] `https://github.com/damo-cv/entroformer`

D Details on User Study

Further Evaluation. In Sec. 5.1 of the main text we report median Elo across all participants and all images, and across all participants (*i.e.* we update Elo ratings of the methods after each game and after all comparisons of each participant, respectively). In Fig. 2 we additionally report Elo scores where all comparisons for each image are treated as a mini-tournament (*i.e.* we update Elo ratings of the methods after all comparisons of the same image). We report median Elo score over 10,000 Monte Carlo iterations as in the main text.

The overall ranking order of the methods does not change, although the variance of the Monte Carlo simulation is larger. This is because the Elo score update when a higher ranked method beats a lower ranked one is smaller, and as there are less frequent, aggregated score updates the stronger methods are not penalized due to their strength as much as when updating after every comparison.

Images Used. For all images used in our user study, we report the filenames and the bitrate (in bits per pixel) for each method in Tab. 1. We compress and decompress the entire image with each method and center crop the result to 512×512px to display to the user (see Sec. 5).

User Interface. Fig. 3 shows a screenshot of the user interface of our user study. The original image is shown in the center, while left and right show the reconstructed images. The order in which each pair comparison appears and the position of each method (left/right) are randomly selected for each rating. Our user interface supports synchronized zooming and panning, so the participant can examine smaller areas of each image if preferred. Zoom and pan levels are reset for each new comparison.

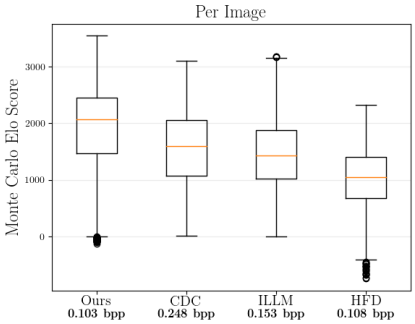

Fig. 2: Per-image Monte Carlo Elo ratings from the user study. Higher is better. The box extends to the first and third quartiles and the whiskers $1.5 \times IQR$ further.

	kodim01	kodim05	kodim07	kodim09	kodim08	kodim17	kodim19	kodim22	kodim23	kodim24
Ours	0.0852	0.154	0.101	0.0744	0.163	0.0927	0.0907	0.0984	0.0606	0.110
HFD	0.0940	0.161	0.111	0.0765	0.167	0.0966	0.0918	0.0992	0.0644	0.113
CDC	0.276	0.365	0.229	0.166	0.297	0.237	0.207	0.252	0.160	0.290
ILLM	0.0983	0.240	0.147	0.118	0.243	0.140	0.146	0.146	0.0644	0.187

Table 1: Filenames and corresponding bitrate of each method for all images used in the user study. Bitrate is expressed in bits per pixel.

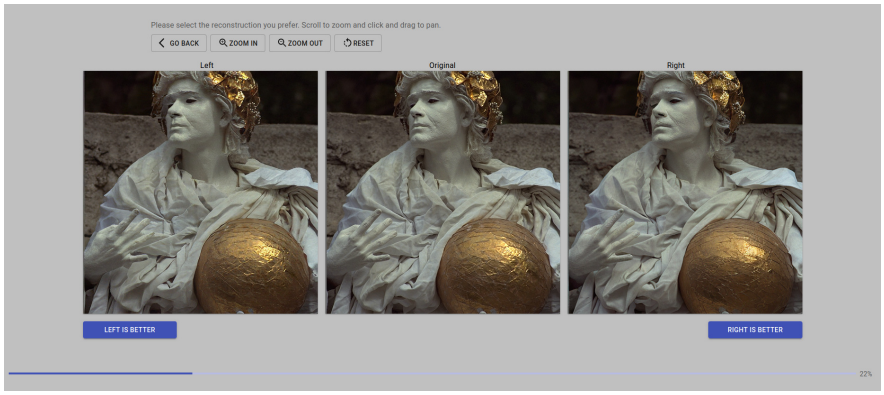

Fig. 3: Screenshot of the user interface of our subjective user study.

E Further Analysis on Timestep Prediction

Here we present additional evidence that our method is able to predict the ideal number of denoising timesteps over all bitrates (see Sec. 4).

We perform an experiment in which we use our method to compress an image at multiple bitrates and manually sweep over a range of denoising diffusion timesteps (similar to the naive latent diffusion implementation in Sec. 3 of the main text, but only over the timestep parameter). We record rate-distortion metrics for each quantization level and timestep pair. Fig. 4 shows the results of this experiment on images from the Kodak dataset, where for all bitrates our method predicts the number of denoising diffusion steps which results in the lowest distortion.

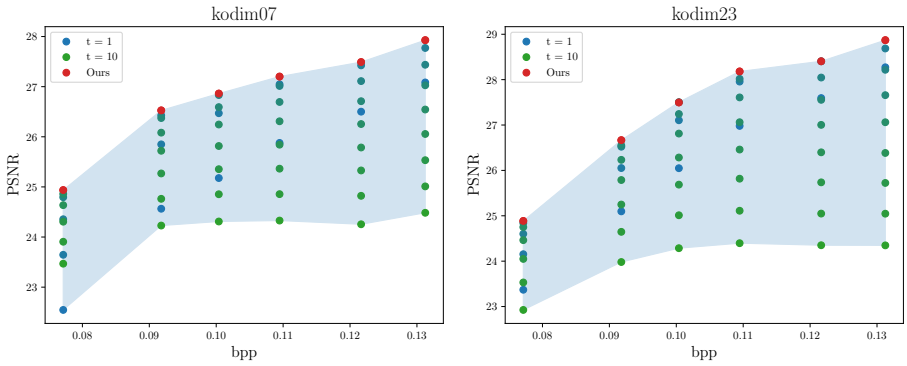

Fig. 4: Rate-distortion curve of our method with manually chosen denoising diffusion steps. The color gradient of the dots represents the number of denoising steps. Our predicted optimal number of steps is shown in red.

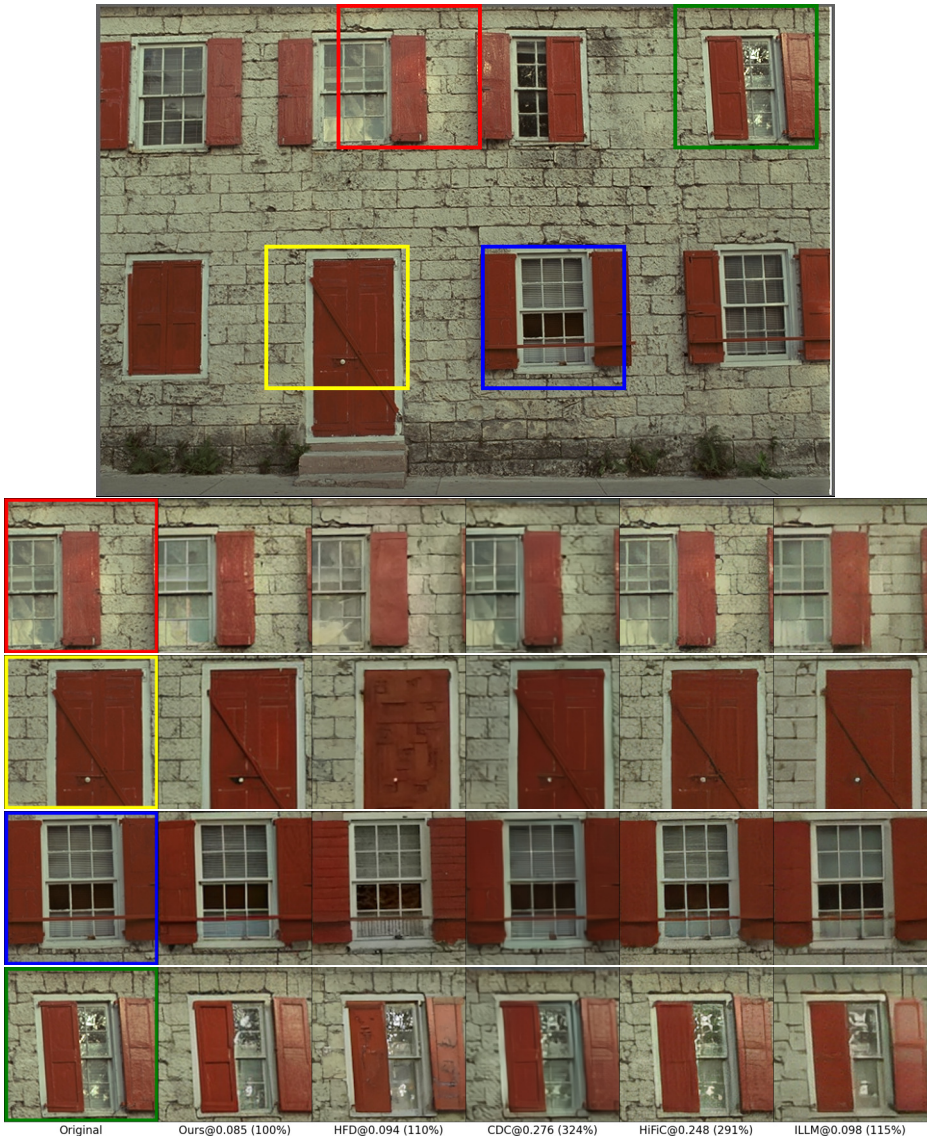

Fig. 5: Additional visual comparisons (kodim01). Images are labeled with Method@bpp (% bpp compared to Ours).

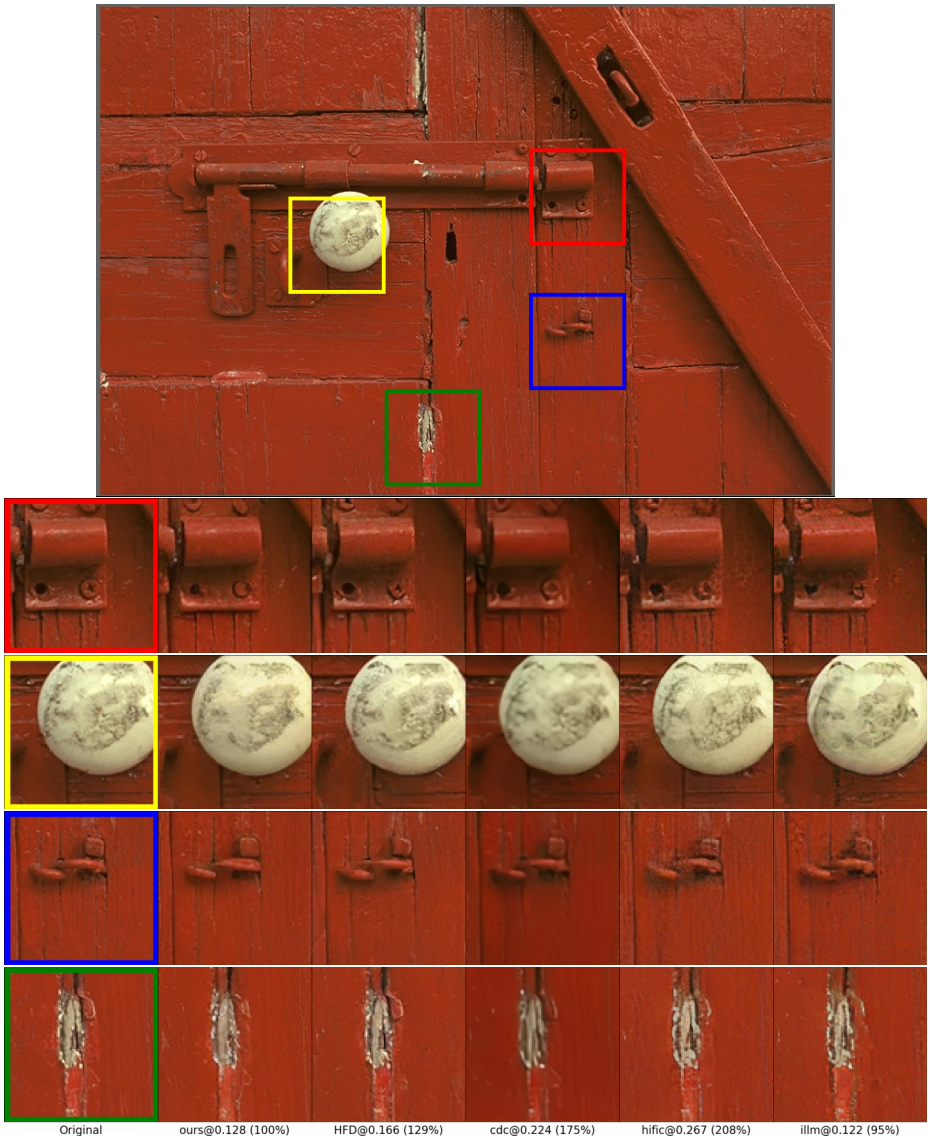

Fig. 6: Additional visual comparisons (kodim02). Images are labeled with Method@bpp (% bpp compared to Ours).

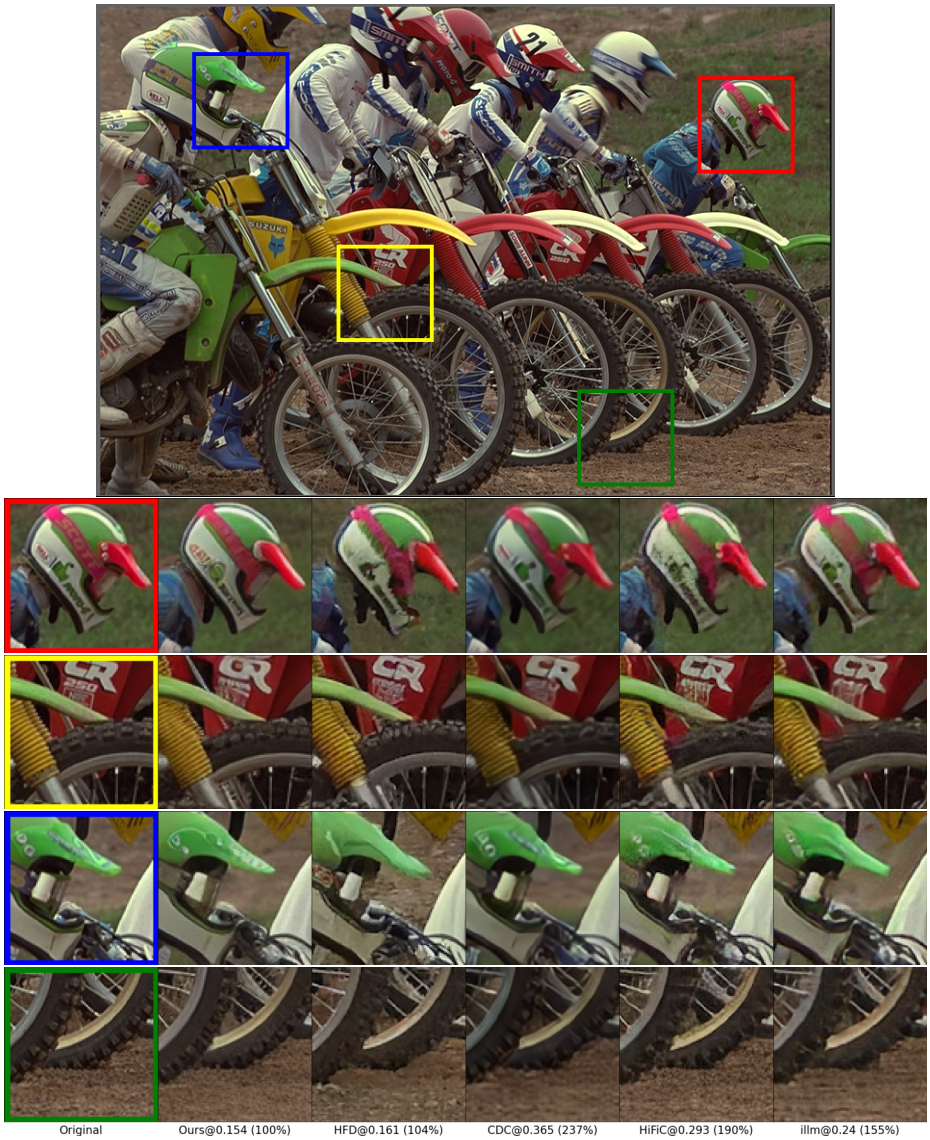

Fig. 7: Additional visual comparisons (kodim05). Images are labeled with Method@bpp (% bpp compared to Ours).

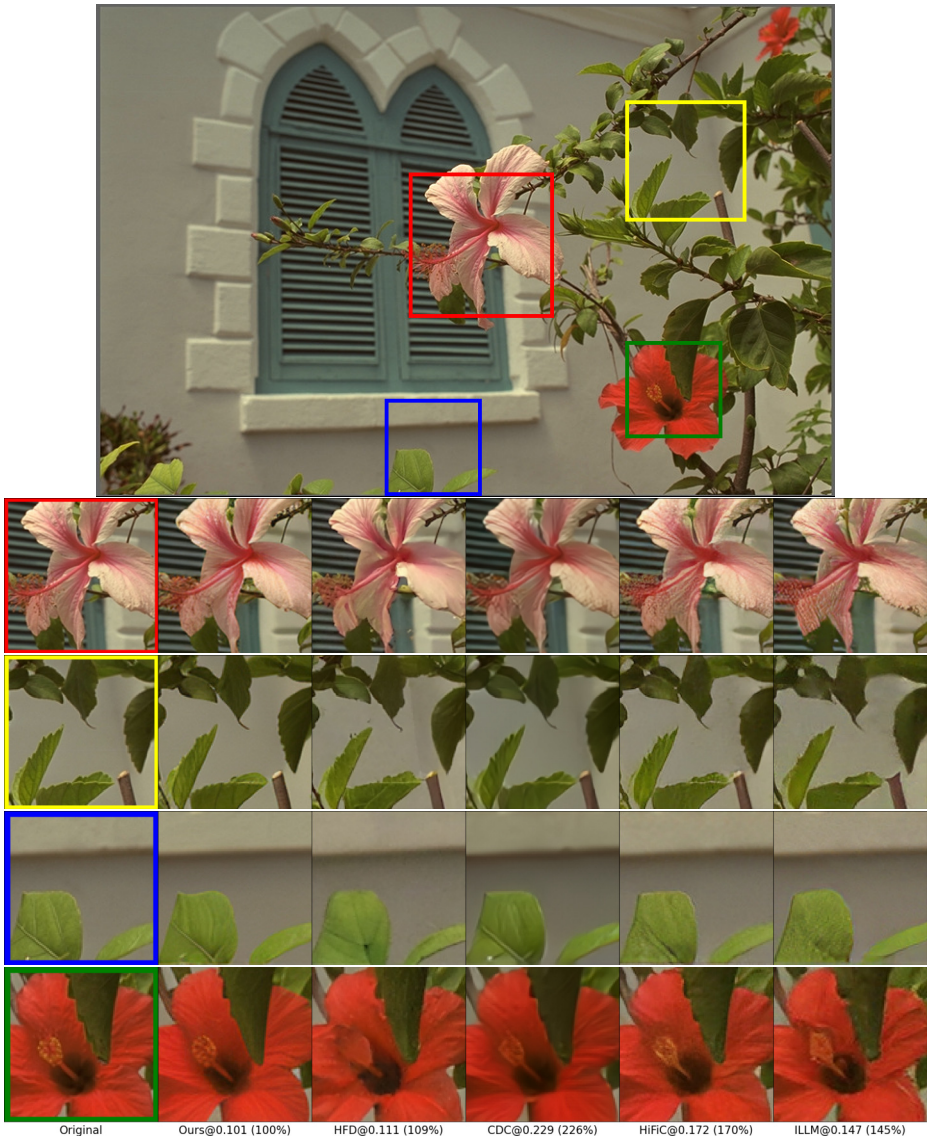

Fig. 8: Additional visual comparisons (kodim07). Images are labeled with Method@bpp (% bpp compared to Ours).

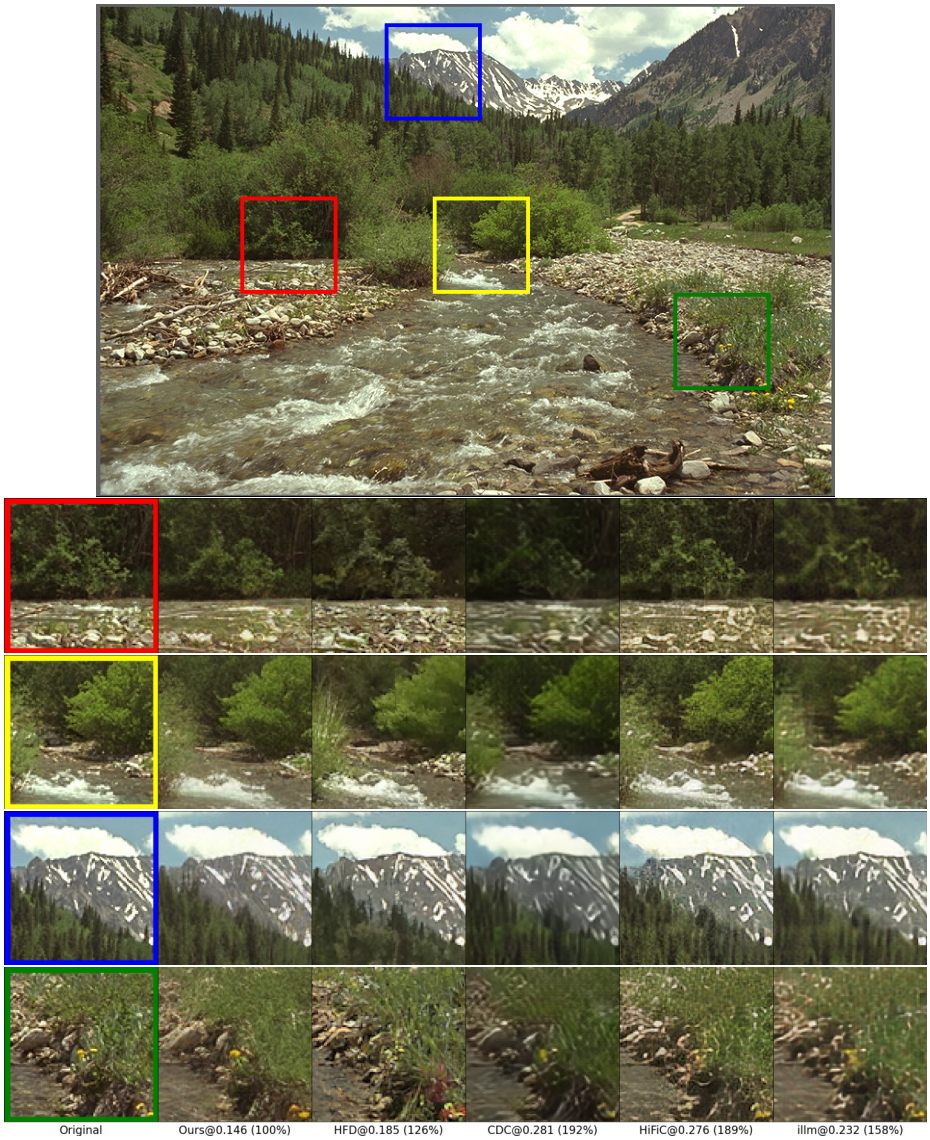

Fig. 9: Additional visual comparisons (kodim13). Images are labeled with Method@bpp (% bpp compared to Ours).

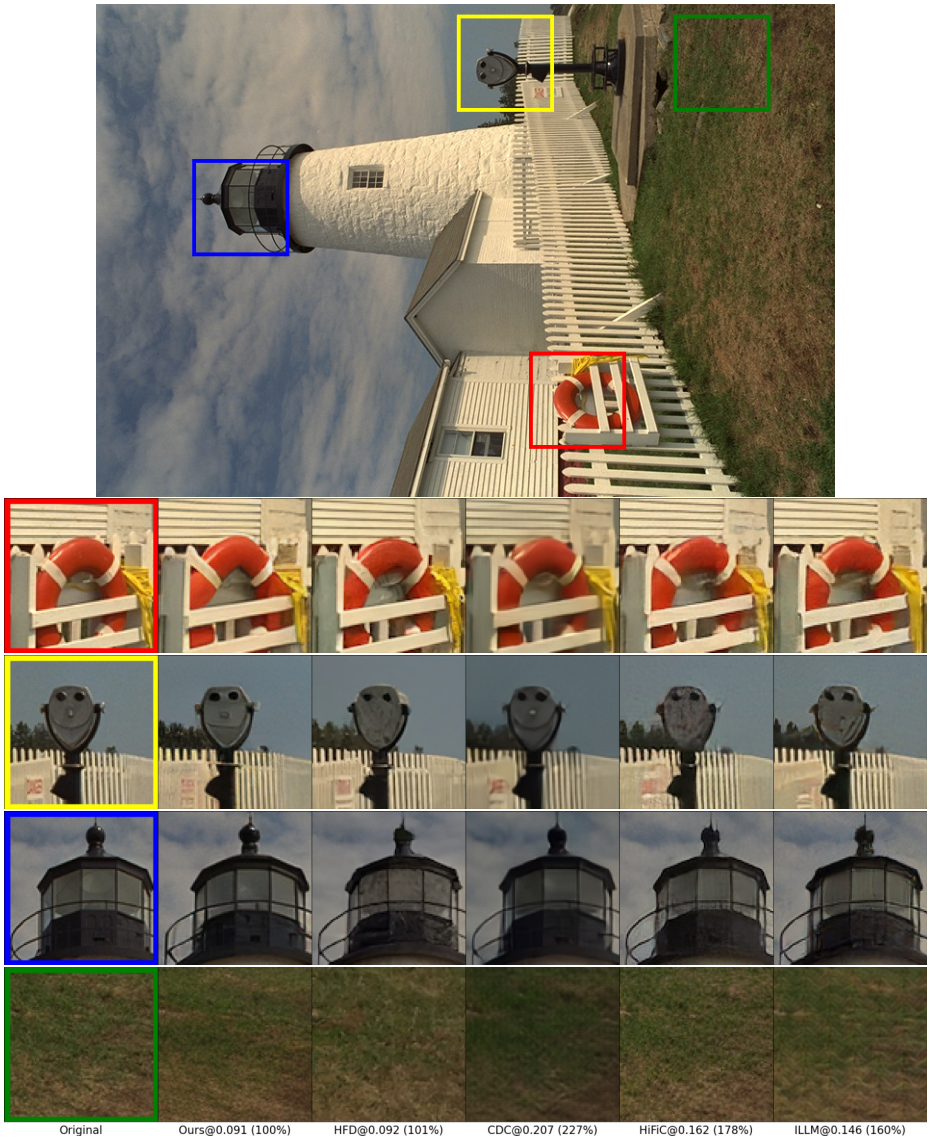

Fig. 10: Additional visual comparisons (kodim19). Images are labeled with Method@bpp (% bpp compared to Ours).

Fig. 11: Additional visual comparisons (kodim24). Images are labeled with Method@bpp (% bpp compared to Ours).

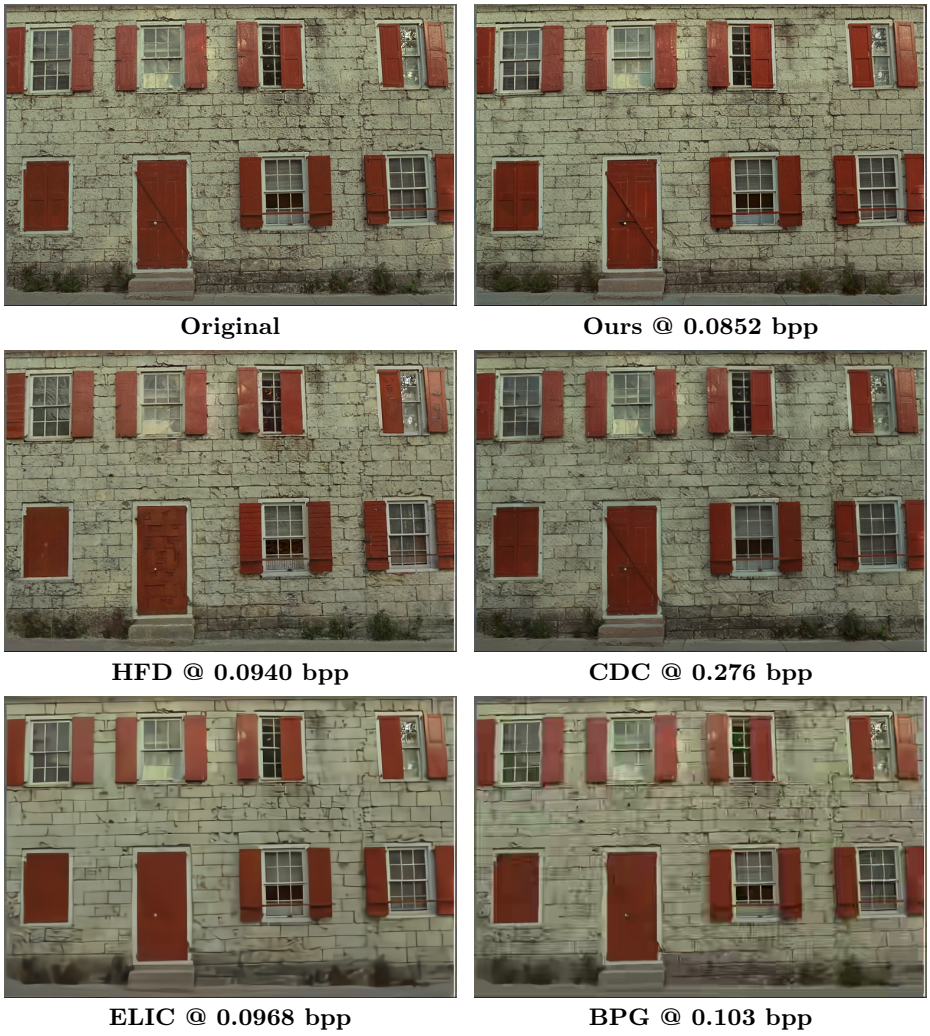

Fig. 12: Qualitative comparison of our method, HFD [6], CDC [10], ELIC [5], and BPG [1] on *kodim01*. Best viewed digitally.

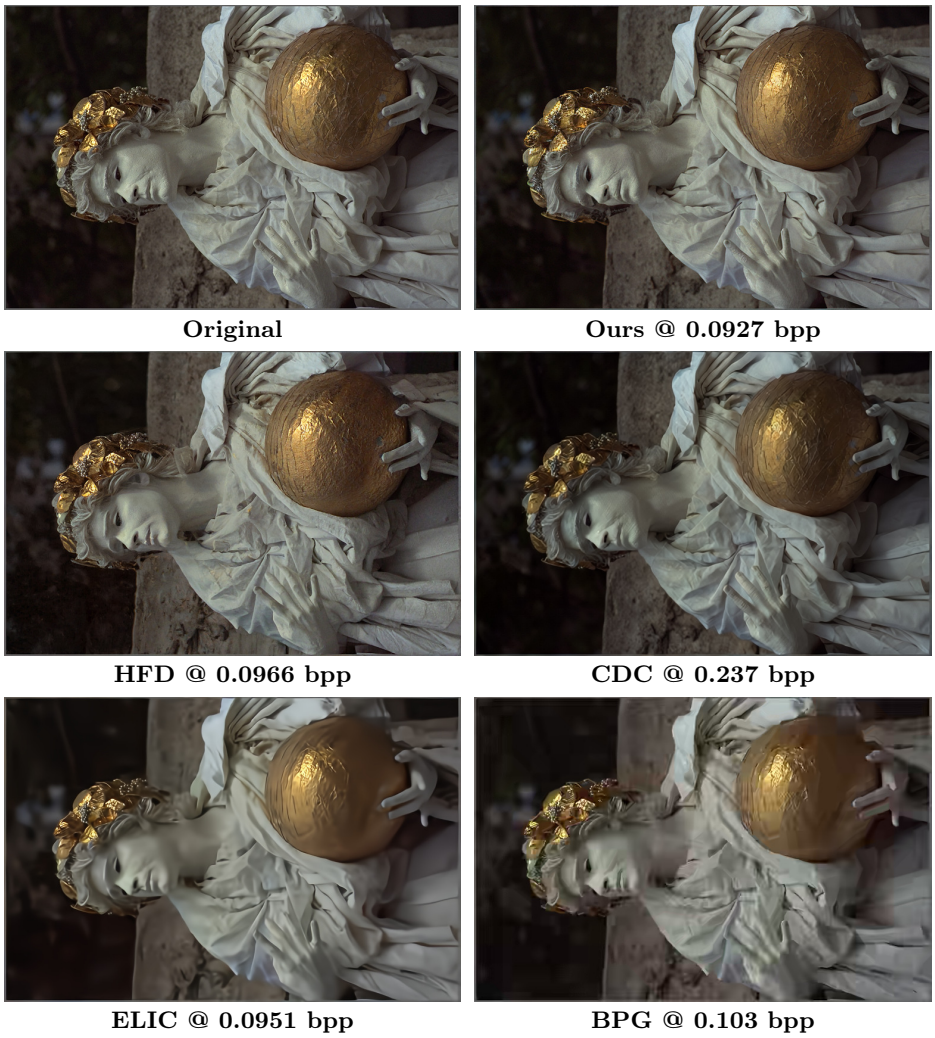

Fig. 13: Qualitative comparison of our method, HFD [6], CDC [10], ELIC [5], and BPG [1] on *kodim17*. Best viewed digitally.

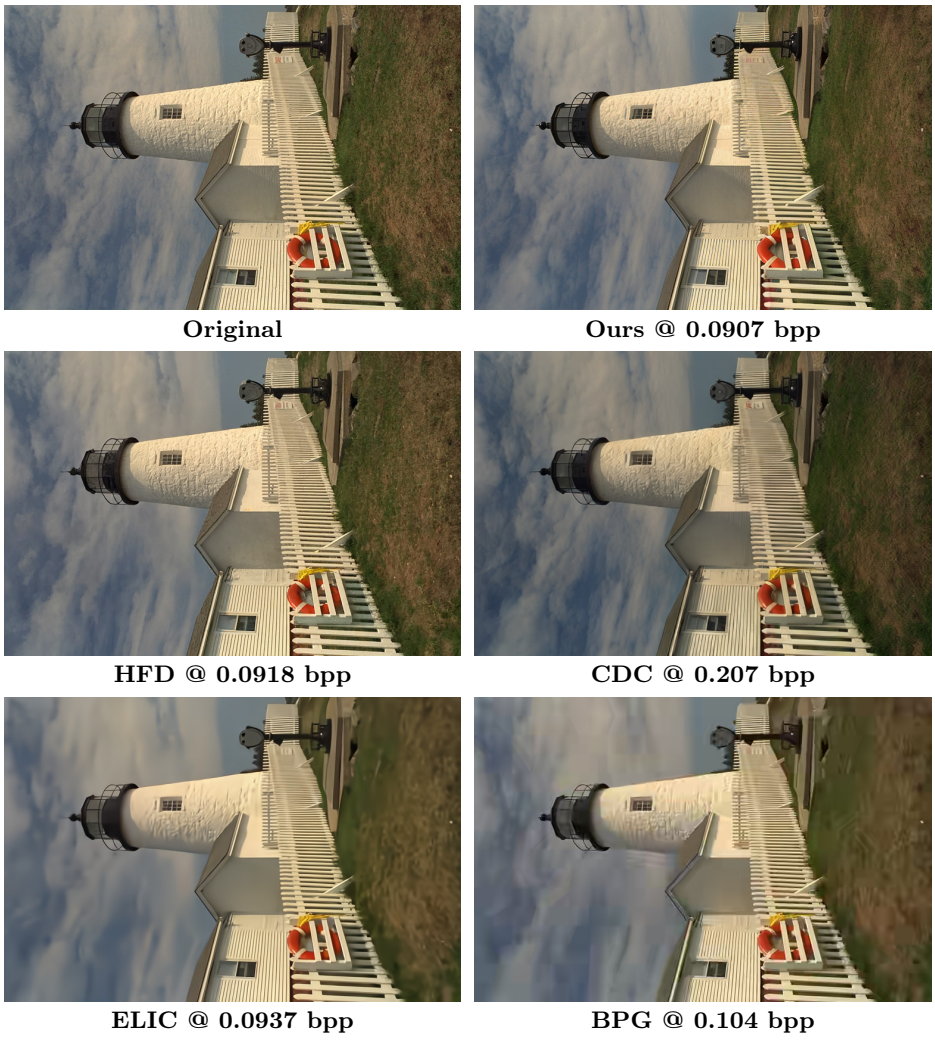

Fig. 14: Qualitative comparison of our method, HFD [6], CDC [10], ELIC [5], and BPG [1] on *kodim19*. Best viewed digitally.

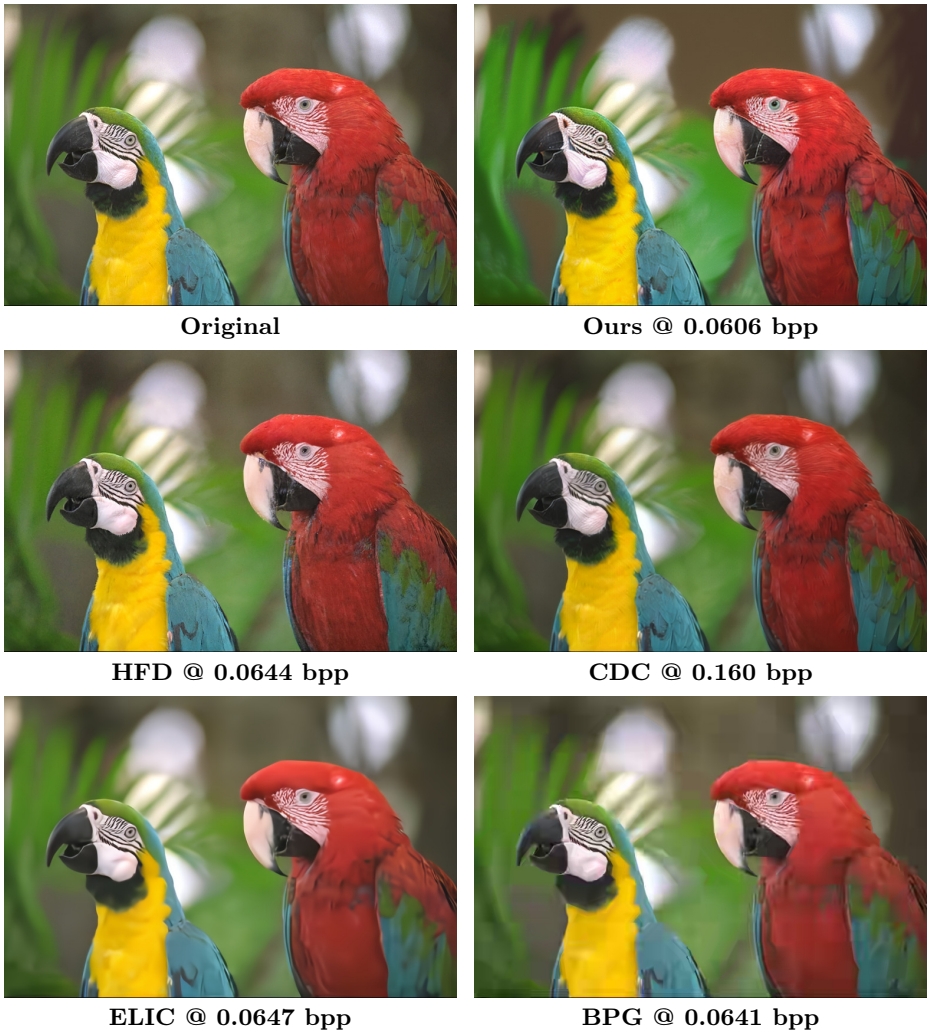

Fig. 15: Qualitative comparison of our method, HFD [6], CDC [10], ELIC [5], and BPG [1] on *kodim23*. Best viewed digitally.

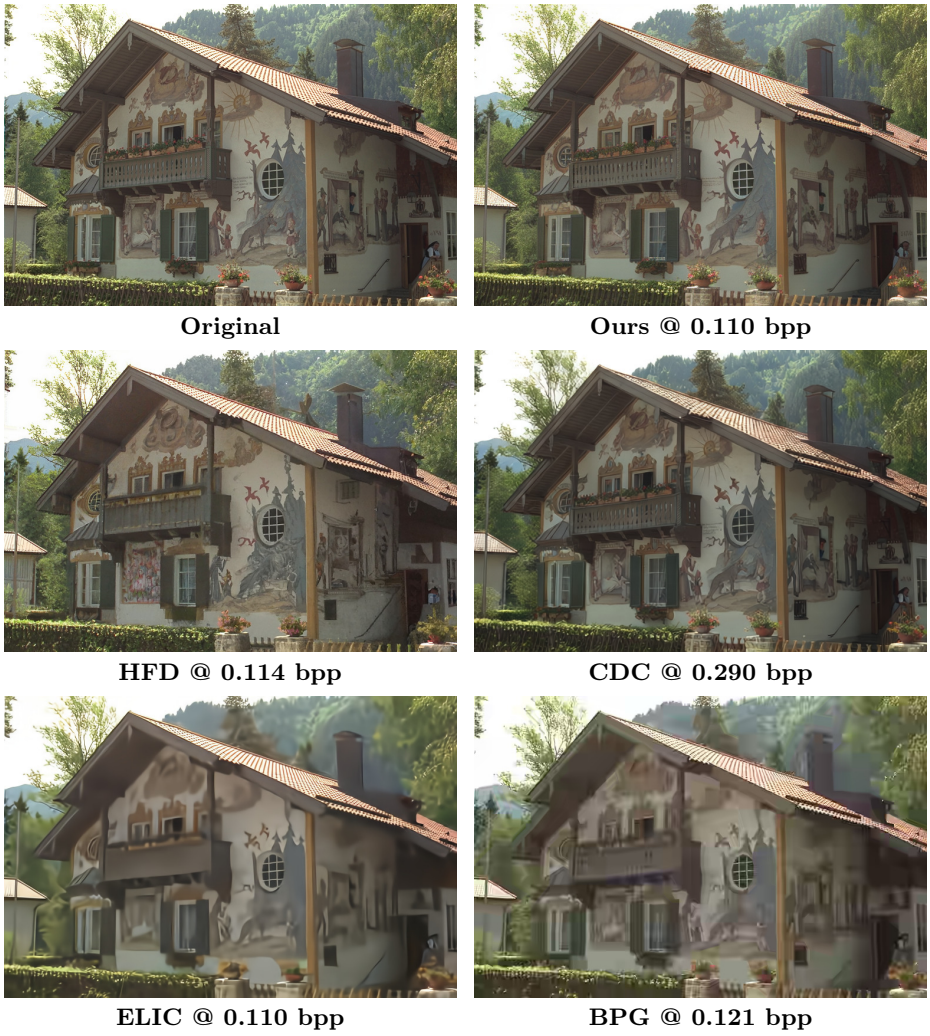

Fig. 16: Qualitative comparison of our method, HFD [6], CDC [10], ELIC [5], and BPG [1] on *kodim24*. Best viewed digitally.

References

1. BPG Image format. https://bellard.org/bpg/
2. Ballé, J., Laparra, V., Simoncelli, E.P.: End-to-end Optimized Image Compression (Mar 2017). https://doi.org/10.48550/arXiv.1611.01704
3. Ballé, J., Minnen, D., Singh, S., Hwang, S.J., Johnston, N.: Variational image compression with a scale hyperprior (May 2018). https://doi.org/10.48550/arXiv.1802.01436
4. Boncelet, C.: Chapter 7 - image noise models. In: Bovik, A. (ed.) The Essential Guide to Image Processing, pp. 143–167. Academic Press, Boston (2009). https://doi.org/https://doi.org/10.1016/B978-0-12-374457-9.00007-X, https://www.sciencedirect.com/science/article/pii/B978012374457900007X
5. He, D., Yang, Z., Peng, W., Ma, R., Qin, H., Wang, Y.: ELIC: Efficient Learned Image Compression With Unevenly Grouped Space-Channel Contextual Adaptive Coding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5718–5727 (2022)
6. Hoogeboom, E., Agustsson, E., Mentzer, F., Versari, L., Toderici, G., Theis, L.: High-Fidelity Image Compression with Score-based Generative Models (May 2023). https://doi.org/10.48550/arXiv.2305.18231
7. Minnen, D., Ballé, J., Toderici, G.: Joint Autoregressive and Hierarchical Priors for Learned Image Compression (Sep 2018). https://doi.org/10.48550/arXiv.1809.02736
8. Qian, Y., Lin, M., Sun, X., Tan, Z., Jin, R.: Entroformer: A Transformer-based Entropy Model for Learned Image Compression (Mar 2022). https://doi.org/10.48550/arXiv.2202.05492
9. Salimans, T., Ho, J.: Progressive Distillation for Fast Sampling of Diffusion Models. In: International Conference on Learning Representations (Jun 2022). https://doi.org/10.48550/arXiv.2202.00512
10. Yang, R., Mandt, S.: Lossy Image Compression with Conditional Diffusion Models. Advances in Neural Information Processing Systems **36**, 64971–64995 (Dec 2023)