Supplementary Material: Taming Latent Diffusion Model for Neural Radiance Field Inpainting

Chieh Hubert Lin[1,2], Changil Kim[1], Jia-Bin Huang[1,3], Qinbo Li[1], Chih-Yao Ma[1], Johannes Kopf[1], Ming-Hsuan Yang[2], and Hung-Yu Tseng[1]

[1]Meta [2]University of California, Merced [3]University of Maryland, College Park

1 More Visual Results

We organize all the video rendering results with an HTML interface [1]. Note that some videos have slightly different rendering trajectories since we directly obtain the videos from the authors for certain baselines. We have put our best effort into guessing and aligning the trajectories.

2 Object Insertion

A few prior works [8, 14] also demonstrates applications of inpainting objects in NeRF with text prompts. However, most of these results remain preliminary without complicated object-scene interaction. In such cases, synthetic 3D objects can be inserted with depth blend afterward and later baked into NeRF with optimization. Meanwhile, directly inpainting objects with the input masks is not effective as the masks strongly constrain the object shape. In Figure 1, similar to GaussianEditor [3], we show that our framework can achieve object inpainting by first inpainting the background then followed by inserting generated objects.

Fig. 1: Insert synthesized objects with depth blending.

3 Training Robustness

Despite we adopt adversarial loss in our optimization objectives, our algorithm is stable with R_1 regularizer [7] and gradient penalty [4]. We show the robustness by reporting the performance of three individual training trials of all scenes from SPIn-NeRF dataset. To save computation with the large amount of experiments, we use 1 GPU instead of 8 GPUs per experiment.

Trial	LPIPS	M-LPIPS	FID	KID
A	0.4147	0.3056	195.47	0.0466
B	0.4138	0.3055	194.59	0.0456
C	0.4142	0.3058	191.91	0.0486

[1] https://hubert0527.github.io/MALD-NeRF/

4 Further Customization

During the per-scene customization, due to repetitively using the same set of images as the finetuning data distribution, the model can sometimes memorize certain scene-dependent content. The behavior can lead to inpainting objects in the inpainting region, such as the baseball cap inpainted in Figure 5 of the main paper. Such a behavior can easily remedy by masking the unwanted objects from the scene-dependent customization. We show such a result on the right.

5 Evaluate Geometric Consistency with Optical Flow-based Metric

Following [13], we evaluate the cross-frame consistency with a flow warping score. We treat the image set of each scene as an image sequence, then calculate optical flow between an image pair I_t and I_{t+1} with RAFT [12], finally, obtain $\tilde{I}_t$ by warping I_{t+1} with the optical flow. We compute the consistency with $M_{t\to t+1} \cdot \|I_t - \tilde{I}_t\|_1$, where $M_{t\to t+1}$ is the disocclusion mask from time t to $t+1$. We found such a metric highly depends on the flow quality predicted by RAFT, and heavily favors blurry inpainting results. In practice, we found the blurry results from all baselines outperforms the consistency scores from the real images.

6 Implementation Details

NeRF. We use a self-implemented NeRF framework similar to ZipNeRF [2] that uses hash-based [9] positional encoding along with multiple MLPs to predict the final density and RGB quantities. A scene contraction is applied to the NeRF [1] as all the scenes we experimented on are unbounded scenes. We use two proposal networks to perform importance sampling, followed by the main network. The network designs are similar to the Nerfacto implemented in the Nerfstudio [11].

Hyperparameters. For all experiments, we train the networks with 8 V100 GPUs for 30,000 iterations at a ray batch size of 16,384 using distributed data-parallel. The choice of batch size is constrained by the amount of GPU VRAM after loading the NeRF and other deep image priors, such as the latent diffusion model network for generative inpainting and the ZoeDepth model (NK version) for the depth loss. All these deep image priors are inferenced without calculating gradients to reduce VRAM usage, and inference at a batch size of 1.

We use two separate optimizers for NeRF reconstruction and adversarial learning. The NeRF reconstruction uses an Adam optimizer with a learning rate decay from 0.01 to 0.0001, while adversarial learning uses an Adam optimizer with a learning rate 0.0001 throughout the training. Different from GANeRF [10],

we found using RMSProp makes the training unstable. For the adversarial learning, we use a discriminator architecture similar to StyleGAN2 [6]. We train the discriminator with 64×64 patches. We importance-sample 256×256 image patches based on the number of inpainting pixels within the patch, then slice the image patch into the discriminator training patch and train the discriminator at a batch size of 16. For the importance sampling strategy, we first exclude patches with insufficient inpainting pixels (we empirically set the threshold to 50%). Assume that each patch index i contains d_i inpainting pixels, we assign a probability $p_i = d_i / \sum_j d_j$ while sampling the patches for training.

As mentioned in Eq 4, Eq 5, and Eq 6 of the main paper, our networks are being trained with various loss terms. We balance the loss terms with $\lambda_{\text{pix}} = 1$, $\lambda_{\text{inter}} = 3$, $\lambda_{\text{distort}} = 0.002$, $\lambda_{\text{decay}} = 0.1$, $\lambda_{\text{adv}} = 1$, $\lambda_{\text{fm}} = 1$ and $\lambda_{\text{GP}} = 15$.

Iterative Dataset Update. We infer the latent diffusion model with a DDIM scheduler for 20 steps. During the iterative dataset update, we synchronize the random sampled image IDs across GPUs to ensure there is no overlap among GPUs, then update the 8 distinctly sampled images in the dataset with partial DDIM. For the partial DDIM, we first hard-blend the rendered pixels in the inpainting mask region with the real pixels outside the inpainting region into a 512×512 image, encode into the latent space with the auto-encoder of the latent diffusion model, then add the noise level at timestep t based on the current training progress and the HiFA scheduling. Therefore, as the training progresses, the final inpainted images will gradually converge to the current NeRF rendering results due to low noise levels. Since we update 8 images in each dataset update step, we set the frequency of iterative dataset update to one dataset update every 80 NeRF training steps, which is 8 times less frequent compared to InstructNeRF2NeRF [5]. The whole training approximately takes 16 hours on 8 V100 GPUs.

References

1. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: CVPR (2022) 2
2. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Zip-nerf: Anti-aliased grid-based neural radiance fields. In: ICCV (2023) 2
3. Fang, J., Wang, J., Zhang, X., Xie, L., Tian, Q.: Gaussianeditor: Editing 3d gaussians delicately with text instructions. In: CVPR (2024) 1
4. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: NeurIPS (2017) 1
5. Haque, A., Tancik, M., Efros, A.A., Holynski, A., Kanazawa, A.: Instruct-nerf2nerf: Editing 3d scenes with instructions. In: ICCV (2023) 3
6. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: CVPR (2020) 3
7. Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for gans do actually converge? In: ICML (2018) 1
8. Mirzaei, A., Aumentado-Armstrong, T., Brubaker, M.A., Kelly, J., Levinshtein, A., Derpanis, K.G., Gilitschenski, I.: Reference-guided controllable inpainting of neural radiance fields. In: ICCV (2023) 1

9. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM TOG (2022) 2
10. Roessle, B., Müller, N., Porzi, L., Bulò, S.R., Kontschieder, P., Nießner, M.: Ganerf: Leveraging discriminators to optimize neural radiance fields. ACM TOG (2023) 2
11. Tancik, M., Weber, E., Ng, E., Li, R., Yi, B., Kerr, J., Wang, T., Kristoffersen, A., Austin, J., Salahi, K., Ahuja, A., McAllister, D., Kanazawa, A.: Nerfstudio: A modular framework for neural radiance field development. In: ACM TOG (2023) 2
12. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: ECCV (2020) 2
13. Tseng, H.Y., Li, Q., Kim, C., Alsisan, S., Huang, J.B., Kopf, J.: Consistent view synthesis with pose-guided diffusion models. In: CVPR (2023) 2
14. Weber, E., Holynski, A., Jampani, V., Saxena, S., Snavely, N., Kar, A., Kanazawa, A.: Nerfiller: Completing scenes via generative 3d inpainting. In: CVPR (2024) 1