Appendix: Stochastic Structural SIMilarity and Its Unreasonable Effectiveness for Neural Field

Zeke Xie*, Xindi Yang*, Yujie Yang,
Qi Sun, Yixiang Jiang, Haoran Wang, Yunfeng Cai, and Mingming Sun

Baidu Research
*Equal Contributions: *{xiezeke,yangxindi}@baidu.com*

A. Experimental Settings and Details

In this section, we present the experimental settings and details for reproducing the results. The main principle of our experimental setting is to fairly compare *multiplex training* and standard training for NeRF and the variants. Our experimental settings follows original papers to produce the baselines, unless we specify otherwise.

S3IM Setting. We always choose the kernel size $K = 4$, the stride size $S = 4$, and the number of stochastic patches $M = 10$ without fine-tuning. We fine-tune the S3IM loss weight λ from $\{0.05, 0.1, 0.2, 0.5, 1, 2, 5\}$ for the NeRF family and $\{1, 2, 5, 10, 20, 50, 100\}$ for the NeuS family.

A.1. Models and Optimization

DVGO Setting We employ the sourcecode of DVGO (Version 2) in the original project [12] without modifying training hyperparameters. So we train DVGO via Adam [7] with the batch size $B = 8192$. The learning rate of density voxel grids and color/feature voxel grids is 0.1, and the learning rate of the RGB net (MLP) is 0.001. The total number of iterations is 5000. We multiply the learning rate by 0.1 per 1000 iterations.

TensoRF Setting We employ the sourcecode of the original project [1] without modifying training hyperparameters. The total number of iterations is 30000. The batch size is 4096. The initial number of voxels is 128^3, while the final number of voxels is 300^3. The upsamping iterations for voxels are 2000,3000,4000,5500 and 7000, respectively. Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.99$ is used.

NeRF Setting We employ a popular open-source implementation [16] of the original NeRF. Again, we follow its defaulted training setting. The learning rate is 0.0005, and the learning rate scheduler is $0.1^{iters/500000}$.

D-NeRF Setting We directly employ the sourcecode of D-NeRF in the original project [9]. We only slightly change the original batch size $B = 500$ to $B = 512$ for generating the squared stochastic patch. The learning rate is 0.0005. The total number of iterations is 800k. The learning rate decay follows the original paper.

NeuS Setting We employ the NeuS implementation of SDFStudio [17] and follow its default hyperparameters. The difference of the hyperparameters between SDFStudio and the original paper [13] is that SDFStudio trains 100k iterations, while the original paper trains 300k iterations.

A.2. Datasets

Replica Dataset Replica Dataset has no splitted training dataset and test dataset. In the experiments on Replica, if one image index is divisible by 10, we move the image to the test dataset; if not, we move the image to the training dataset. Thus, we have 90% images for training and 10% images for evaluation.

T&T Dataset Advanced T&T Dataset Advanced has no splitted training data and test data. We follow the original splitted way in the standard setting. In the experiments on Replica, if one image index is divisible by 10, we move the image to the test T&T Dataset Advanced; if not, we move the image to the training dataset. Similarly, we again have 90% images for training and 10% images for evaluation.

T&T Dataset Intermediate T&T Dataset has splitted training data and test data. We follow the original splitted way in the standard setting. In the experiments of sparse inputs, we randomly remove the training images. In the experiments of corrupted images, we inject Gaussian noise with the scale std into RGB values of the training images, and clip the corrupted RGB values into $[0, 1]$.

Dynamic Scenes: LEGO and Mutant The two dynamics scenes are used in the original D-NeRF paper. We use them in the same way without any modification.

B. Image Quality Metrics

As we mentioned above, PSNR and SSIM are two most popular image quality metrics. Beyond them, we have also seen other useful metrics. Multiscale SSIM [15] is a variant of SSIM, which incorporates image details at different resolutions. However, Multiscale SSIM still can only capture local structural information carried by nearby pixels.

Deep features learned by DNNs has unreasonable effectiveness as a perceptual metric for measuring the similarity between two sets of perceptual features [18]. Thus, the Learned Perceptual Image Patch Similarity (LPIPS) metric [18] serves as the third rendering quality metric in NeRF and related studies, because it agrees surprisingly well with humans. The Fréchet inception distance (FID) score [4] is another metric which measure the distance-based similarity of perceptive features, but it is more widely used for evaluating generative models, such as Generative Adversarial Networks [2, 3] (GAN) and Diffusion Models [10, 11, 5]. Both LPIPS and FID metrics inevitably have certain stochasticity, because deep features are learned from stochastic training of DNNs. However, the stochasticity does not affect the usefulness and popularity of LPIPS and FID as image quality metrics.

S3IM can also be considered as a image quality metric, while we only use S3IM as a differentiable training objective in this paper. More specifically, S3IM measures the structural similarity of two paired sets of pixels(/signals), which may or may not form images. By analyzing the stochastic patch consists of random pixels, S3IM can capture non-local structural information carried by nearby/distant pixels. S3IM also inevitably have certain stochasticity like LPIPS and FID, while S3IM does not depend on deep features given by DNNs.

SSIM is a well-known quality metric that can capture local structural similarity between images or patches. SSIM is considered to be correlated with the quality perception of the human visual system well and is widely used for evaluating NeRF [14, 6]. Suppose $\boldsymbol{a} = \{a_i | i = 1, 2, 3, \ldots, n\}$ and $\boldsymbol{b} = \{b_i | i = 1, 2, 3, \ldots, n\}$ to be two discrete non-negative signals paired with each other (e.g. two image patches extracted from the same spatial location from paired images). We denote the mean intensity of a signal as μ (e.g. $\mu_a = \frac{1}{n}\sum_{i=1}^{n} a_i$), the standard deviation of a signal as σ^2 (e.g. $\sigma_a^2 = \frac{1}{n-1}\sum_{i=1}^{n}(a_i - \mu_a)^2$), and the covariance between two signals as σ_{ab}^2 (e.g. $\sigma_{ab}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(a_i - \mu_a)(b_i - \mu_b)$).

SSIM is expressed by the combination of three terms which are the luminance, contrast, and structure comparison metrics:

$$\mathrm{SSIM}(a, b) = l(\boldsymbol{a}, \boldsymbol{b}) c(\boldsymbol{a}, \boldsymbol{b}) s(\boldsymbol{a}, \boldsymbol{b}). \tag{1}$$

The luminance $l(\boldsymbol{a}, \boldsymbol{b})$, contrast $c(\boldsymbol{a}, \boldsymbol{b})$, and structure comparison $s(\boldsymbol{a}, \boldsymbol{b})$ are, respectively, written as

$$l(\boldsymbol{a}, \boldsymbol{b}) = \frac{2\mu_a\mu_b + C_1}{\mu_a^2 + \mu_b^2 + C_1}, \tag{2}$$

$$c(\boldsymbol{a}, \boldsymbol{b}) = \frac{2\sigma_a\sigma_b + C_2}{\sigma_a^2 + \sigma_b^2 + C_2}, \tag{3}$$

$$s(\boldsymbol{a}, \boldsymbol{b}) = \frac{\sigma_{ab} + C_3}{\sigma_a\sigma_b + C_3}, \tag{4}$$

where C_1, C_2, and C_3 are small constants given by

$$C_1 = (K_1 L)^2, C_2 = (K_2 L)^2, \text{and } C_3 = C_2/2. \tag{5}$$

Following the common setting [14, 8], we set $K_1 = 0.01$ and $K_2 = 0.03$ in this paper. The data range L is 1 for pixel RGB values. The range of SSIM lies in $[-1, 1]$.

In practice of image quality assessment, people usually apply the SSIM index locally rather than globally. The local statistics μ_a, σ_a, and σ_{ab} are computed within a local $K \times K$ kernel window, which moves with a stride size s over the entire image. For example, for evaluating NeRF, people often use the kernel size 11×11, the stride size 1, and the circular

symmetric Gaussian weighting function $\boldsymbol{w} = \{w_i | i = 1, 2, \ldots, n\}$, with standard deviation of 1.5 samples, normalized to unit sum ($\sum_i w_i = 1$). The local statistics are then written as

$$\mu_a = \sum_{i=1}^{n} w_i a_i, \tag{6}$$

$$\sigma_a = \left(\sum_{i=1}^{n} w_i (a_i - \mu_a)^2 \right)^{\frac{1}{2}}, \tag{7}$$

$$\sigma_{ab} = \left(\sum_{i=1}^{n} w_i (a_i - \mu_a)(b_i - \mu_b) \right)^{\frac{1}{2}}. \tag{8}$$

At each step, the local statistics and SSIM index are calculated within the local window. The final SSIM metric for evaluating NeRF is actually the mean SSIM (MSSIM) which is computed by averaging the SSIM indexes over each step.

C. Supplementary Experimental Results

Table 1. Quantitative results of neural rendering of scenes in T&T Intermediate. Model: DVGO.

Scene	Training	PSNR(↑)	SSIM(↑)	LPIPS(↓)
M60	Standard	17.49	0.647	0.501
	Multiplex	**17.71**	**0.652**	**0.483**
Playground	Standard	22.74	0.669	0.467
	Multiplex	**22.75**	**0.681**	**0.444**
Train	Standard	17.19	0.566	0.533
	Multiplex	**18.24**	**0.581**	**0.491**
Truck	Standard	22.01	0.704	0.386
	Multiplex	**22.44**	**0.730**	**0.338**

In this section, we present supplementary experimental results.

We first present the quantitative results of DVGO on 4 scenes of T&T Intermediate in Table 1.

We present the quantitative results of DVGO, NeRF, and TensoRF on each T&T scenes in Tables 2, 3, and 4, respectively.

Table 2. Quantitative results of NeRF methods on T&T. Model: DVGO.

Scene	Training	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Scene 1	Standard	21.80	0.759	0.243
	Multiplex	**22.16**	**0.783**	**0.191**
Scene 2	Standard	23.96	0.847	0.250
	Multiplex	**25.37**	**0.872**	**0.171**
Scene 3	Standard	18.64	0.697	0.272
	Multiplex	**19.71**	**0.757**	**0.192**
Scene 4	Standard	25.27	0.800	0.179
	Multiplex	**25.55**	**0.823**	**0.151**
Mean	Standard	22.42	0.776	0.236
	Multiplex	**23.20**	**0.809**	**0.176**

Few-shot learning We evaluate *multiplex training* on a few-shot learning task, where we only keep eight training images. The few-shot learning quantitative results in Table 5 support the significant advantage of *multiplex training* via S3IM.

Sparse inputs We present the quantitative results of neural rendering from sparse training images in Table 6.

Robustness to corrupted images We present the quantitative results of neural rendering from Corrupted Truck in Table 7.

Table 3. Quantitative results of NeRF methods on T&T. Model: NeRF.

Scene	Training	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Scene 1	Standard	20.11	0.647	0.351
	Multiplex	**22.34**	**0.714**	**0.302**
Scene 2	Standard	23.09	0.774	0.349
	Multiplex	**25.20**	**0.848**	**0.247**
Scene 3	Standard	17.22	0.523	0.486
	Multiplex	**18.13**	**0.571**	**0.467**
Scene 4	Standard	23.65	0.692	0.269
	Multiplex	**24.88**	**0.767**	**0.198**
Mean	Standard	21.02	0.659	0.364
	Multiplex	**22.64**	**0.725**	**0.304**

Table 4. Quantitative results of NeRF methods on T&T. Model: TensoRF.

Scene	Training	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Scene 1	Standard	17.33	0.643	0.413
	Multiplex	**22.67**	**0.779**	**0.216**
Scene 2	Standard	24.44	0.847	0.240
	Multiplex	**25.14**	**0.859**	**0.214**
Scene 3	Standard	12.15	0.342	0.761
	Multiplex	**18.64**	**0.699**	**0.286**
Scene 4	Standard	24.84	0.766	0.207
	Multiplex	**24.93**	**0.770**	**0.205**
Mean	Standard	19.69	0.650	0.365
	Multiplex	**22.85**	**0.777**	**0.230**

Table 5. Quantitative results of few-shot learning. We only keep eight training images from the Truck Scene, T&T Intermediate. Model: DVGO.

Scene	Training	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Truck	Standard	11.37	0.343	0.704
	Multiplex	**13.43**	**0.372**	**0.610**

References

[1] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*, pages 333–350. Springer, 2022. 1

[2] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018. 2

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 2

[4] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 2

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 2

[6] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010. 2

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015*, 2015. 1

Table 6. Quantitative results of neural rendering from sparse training images. Size indicates the portion of training samples kept from the original training dataset.

Size	Training	PSNR(↑)	SSIM(↑)	LPIPS(↓)
20%	Standard	14.67	0.527	0.597
	Multiplex	**18.99**	**0.618**	**0.405**
40%	Standard	18.99	0.639	0.425
	Multiplex	**21.68**	**0.704**	**0.356**
60%	Standard	21.07	0.683	0.396
	Multiplex	**22.11**	**0.719**	**0.347**
80%	Standard	21.74	0.686	0.386
	Multiplex	**22.38**	**0.722**	**0.351**

Table 7. Quantitative results of neural rendering from corrupted training images.

Noise Scale	Training	PSNR(↑)	SSIM(↑)	LPIPS(↓)
0.2	Standard	21.36	0.663	0.453
	Multiplex	**21.94**	**0.686**	**0.420**
0.4	Standard	18.20	0.584	0.569
	Multiplex	**20.89**	**0.636**	**0.494**
0.6	Standard	16.16	0.542	0.684
	Multiplex	**18.06**	**0.571**	**0.599**

[8] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[9] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 1

[10] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. 2

[11] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019. 2

[12] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 1

[13] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *Advances in Neural Information Processing Systems*, 34:27171–27183, 2021. 1

[14] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 2

[15] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003. 2

[16] Lin Yen-Chen. Nerf-pytorch. https://github.com/yenchenlin/nerf-pytorch/, 2020. 1

[17] Zehao Yu, Anpei Chen, Bozidar Antic, Songyou Peng Peng, Apratim Bhattacharyya, Michael Niemeyer, Siyu Tang, Torsten Sattler, and Andreas Geiger. Sdfstudio: A unified framework for surface reconstruction, 2022. 1

[18] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 2