Improved Implicit Neural Representation with Fourier Reparameterized Training

Supplementary Material

In this file, we provide detailed proof of Theorem 2 in the main text, derivations of our initialization scheme and more experimental results on various tasks. We provide detailed proof of our Theorem 2 in section A and present the derivations of our initialization scheme in section B. A detailed ablation study on our hyper-parameters: frequency number and phase number, is presented in D. In section E, we provide 2D image approximation results of more activation functions and input adjustment techniques and show more visual examples and results under other metrics. In section F, we present more experimental results on the shape representation task. Lastly, in section G, we show more view synthesis results by our method and provide the full results of three NeRF frameworks [3, 4, 7].

A. Detailed proof of Theorem 2

Recall that we define $\mathbb{L}(k)$ as the loss function at frequency k:

$$\mathbb{L}(k) = |\mathcal{F}[f_{\mathbf{\Theta}}](k) - \mathcal{F}[g](k)|^2. \tag{1}$$

Then we have the following Theorem 2.

Theorem 2. Given a MLP with multiple hidden layers, reparameterize the weight matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$ of one hidden layer with a trainable coefficient matrix $\mathbf{\Lambda} \in \mathbb{R}^{d\times M}$ and the fixed basis matrix $\mathbf{B} \in \mathbb{R}^{M\times d}$. For any frequencies k_1 and k_2 such that $k_1 > k_2 > 0$, given any $\epsilon \geq 0$ and fixed i, for $j = 1, 2, \ldots, M$, there must exist a set of basis matrices such that

$$|\frac{\partial \mathbb{L}(k_1)}{\partial \lambda_{ij}} / \frac{\partial \mathbb{L}(k_2)}{\partial \lambda_{ij}}| \geq \max\{|\frac{\partial \mathbb{L}(k_1)}{\partial w_{i1}} / \frac{\partial \mathbb{L}(k_2)}{\partial w_{i1}}|, \ldots, |\frac{\partial \mathbb{L}(k_1)}{\partial w_{id}} / \frac{\partial \mathbb{L}(k_2)}{\partial w_{id}}|\} - \epsilon, \tag{2}$$

where $\mathbf{W}(i, j) = w_{ij}$ and $\mathbf{\Lambda}(i, j) = \lambda_{ij}$.

Proof. Before the detailed proof, simply denote: $\mathbb{L}_{\lambda_{ij}}(k_1) = \frac{\partial \mathbb{L}(k_1)}{\partial \lambda_{ij}}, \mathbb{L}_{w_{ij}}(k_1) = \frac{\partial \mathbb{L}(k_1)}{\partial w_{ij}}$.
First, the weight reparameterization for $\mathbf{W}$ is expressed as follows:

$$\mathbf{W} = \mathbf{\Lambda}\mathbf{B}, \tag{3}$$

by the matrix multiplication, for any $w_{ij} \in \mathbf{W}$, the follow equation holds true:

$$w_{ij} = \begin{bmatrix}\lambda_{i1}, \lambda_{i2}, \ldots, \lambda_{iM}\end{bmatrix} \begin{bmatrix} b_{1j} \\ b_{2j} \\ \vdots \\ b_{Mj} \end{bmatrix}, \tag{4}$$

where $B(i, j) = b_{ij}$. Regarding $w_{i1}, \ldots, w_{id}$ as the latent variables related with λ_{ij}, for all $\lambda_{ij} \in \mathbf{\Lambda}$, using the chain rule, we have the following relationships:

$$\mathbb{L}_{\lambda_{ij}}(k) = \sum_{t=1}^{d} b_{jt} \mathbb{L}_{w_{it}}(k). \tag{5}$$

Second, given two frequecies $k_1 > k_2 > 0$, for the i-th row of $\mathbf{\Lambda}$, we set that:

$$\tau = arg\max_{j}\{|\mathbb{L}_{w_{ij}}(k_1)/\mathbb{L}_{w_{ij}}(k_2)|\}. \tag{6}$$

Further, considering the elements of $\mathbf{B}$, for $j = 1, \ldots, M$, we make $|b_{jt}| < \alpha$ for $t \neq \tau$ and $b_{j\tau} = 1$. α is a positive upper bound. Then, according to equation 5, for the fixed i, for $j = 1, \ldots, M$, we have:

$$\begin{aligned}\mathbb{L}_{\lambda_{ij}}(k_1) &= \mathbb{L}_{w_{i\tau}}(k_1) + \sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_1)\\ \mathbb{L}_{\lambda_{ij}}(k_2) &= \mathbb{L}_{w_{i\tau}}(k_2) + \sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_2).\end{aligned} \tag{7}$$

We denote G_1 and G_2 as $\sum_{t\neq\tau}|\mathbb{L}_{w_{it}}(k_1)|$ and $\sum_{t\neq\tau}|\mathbb{L}_{w_{it}}(k_2)|$, respectively.

Without loss of generality, for any

$$0 \leq \epsilon \leq \frac{|\mathbb{L}_{w_{i\tau}}(k_1)|}{|\mathbb{L}_{w_{i\tau}}(k_2)|}, \tag{8}$$

set

$$\alpha \leq \min\{\frac{|\mathbb{L}_{w_{i\tau}}(k_2)|\epsilon}{G_1 + G_2|\mathbb{L}_{w_{i\tau}}(k_1)/\mathbb{L}_{w_{i\tau}}(k_2)| - G_2\epsilon}, |\frac{\mathbb{L}_{w_{i\tau}}(k_1)}{G_1}|\}, \tag{9}$$

then by inequalities involving absolute values, we have:

$$|\mathbb{L}_{\lambda_{ij}}(k_1)/\mathbb{L}_{\lambda_{ij}}(k_2)| = |\frac{\mathbb{L}_{w_{i\tau}}(k_1) + \sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_1)}{\mathbb{L}_{w_{i\tau}}(k_2) + \sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_2)}| \geq \frac{||\mathbb{L}_{w_{i\tau}}(k_1)| - |\sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_1)||}{|\mathbb{L}_{w_{i\tau}}(k_2)| + |\sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_2)|}. \tag{10}$$

From 9, we have that $\alpha \leq |\frac{\mathbb{L}_{w_{i\tau}}(k_1)}{G_1}|$. Then:

$$|\sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_1)| \leq \alpha\sum_{t\neq\tau}|\mathbb{L}_{w_{it}}(k_1)| = \alpha G_1 \leq |\mathbb{L}_{w_{i\tau}}(k_1)|, \tag{11}$$

which means that $|\mathbb{L}_{w_{i\tau}}(k_1)| - |\sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_1)| \geq 0$. Thus, the following inequalities holds true:

$$\frac{|\mathbb{L}_{w_{i\tau}}(k_1)| - |\sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_1)|}{|\mathbb{L}_{w_{i\tau}}(k_2)| + |\sum_{t\neq\tau} b_{jt}\mathbb{L}_{w_{it}}(k_2)|} \geq \frac{|\mathbb{L}_{w_{i\tau}}(k_1)| - \sum_{t\neq\tau}|b_{jt}||\mathbb{L}_{w_{it}}(k_1)|}{|\mathbb{L}_{w_{i\tau}}(k_2)| + \sum_{t\neq\tau}|b_{jt}||\mathbb{L}_{w_{it}}(k_2)|} \geq \frac{|\mathbb{L}_{w_{i\tau}}(k_1)| - \alpha G_1}{|\mathbb{L}_{w_{i\tau}}(k_2)| + \alpha G_2} \geq 0 \tag{12}$$

Substituting $\alpha \leq \frac{|\mathbb{L}_{w_{i\tau}}(k_2)|\epsilon}{G_1+G_2|\mathbb{L}_{w_{i\tau}}(k_1)/\mathbb{L}_{w_{i\tau}}(k_2)|-G_2\epsilon}$ into the above inequality, for the fixed i and for $j = 1, \ldots, M$, we have that:

$$\begin{aligned}|\mathbb{L}_{\lambda_{ij}}(k_1)/\mathbb{L}_{\lambda_{ij}}(k_2)| \geq& \frac{|\mathbb{L}_{w_{i\tau}}(k_1)| - \alpha G_1}{|\mathbb{L}_{w_{i\tau}}(k_2)| + \alpha G_2}\\ \geq& \frac{|\mathbb{L}_{w_{i\tau}}(k_1)|(G_1 + G_2|\mathbb{L}_{w_{i\tau}}(k_1)/\mathbb{L}_{w_{i\tau}}(k_2)| - G_2\epsilon) - G_1|\mathbb{L}_{w_{i\tau}}(k_2)|\epsilon}{|\mathbb{L}_{w_{i\tau}}(k_2)|(G_1 + G_2|\mathbb{L}_{w_{i\tau}}(k_1)/\mathbb{L}_{w_{i\tau}}(k_2)| - G_2\epsilon) + G_2|\mathbb{L}_{w_{i\tau}}(k_2)|\epsilon}\\ =& \frac{G_1|\mathbb{L}_{w_{i\tau}}(k_1)| + \frac{G_2|\mathbb{L}_{w_{i\tau}}(k_1)|^2}{|\mathbb{L}_{w_{i\tau}}(k_2)|}}{G_1|\mathbb{L}_{w_{i\tau}}(k_2)| + G_2|\mathbb{L}_{w_{i\tau}}(k_1)|} - \epsilon\\ =& |\frac{\mathbb{L}_{w_{i\tau}}(k_1)}{\mathbb{L}_{w_{i\tau}}(k_2)}||\frac{G_1 + \frac{G_2|\mathbb{L}_{w_{i\tau}}(k_1)|}{|\mathbb{L}_{w_{i\tau}}(k_2)|}}{G_1 + G_2\frac{|\mathbb{L}_{w_{i\tau}}(k_1)|}{|\mathbb{L}_{w_{i\tau}}(k_2)|}}| - \epsilon\\ =& |\frac{\mathbb{L}_{w_{i\tau}}(k_1)}{\mathbb{L}_{w_{i\tau}}(k_2)}| - \epsilon\\ =& \max\{\frac{|\mathbb{L}_{w_{i1}}(k_1)|}{|\mathbb{L}_{w_{i1}}(k_2)|}, \ldots, \frac{|\mathbb{L}_{w_{id}}(k_1)|}{|\mathbb{L}_{w_{id}}(k_2)|}\} - \epsilon.\end{aligned} \tag{13}$$

□

B. Initialization scheme

Recall that we consider the initialization of the coefficient matrix $\mathbf{\Lambda^{(n)}} \in \mathbb{R}^{d_n \times M}$. Inspired by Kaiming initialization [2], the initialization scheme of the i-th row in $\mathbf{\Lambda^{(n)}}$ should satisfy:

$$Var(\boldsymbol{w_i^{(n)}}\boldsymbol{x}) = Var(\boldsymbol{\lambda_i^{(n)}}\mathbf{B^{(n)}}\boldsymbol{x}), \tag{14}$$

where $\boldsymbol{w_i^{(n)}} \in \mathbb{R}^{1\times d_{n-1}}$ and $\boldsymbol{\lambda_i^{(n)}} \in \mathbb{R}^{1\times M}$ are the i-th row of the weights matrix $\mathbf{W^{(n)}} \in \mathbb{R}^{d_n \times d_{n-1}}$ and the coefficient matrix $\mathbf{\Lambda^{(n)}}$, respectively; $Var(\cdot)$ denotes the variance; $\boldsymbol{x} \in \mathbb{R}^{d_{n-1}\times 1}$ is the input of this layer; $\mathbf{B^{(n)}} \in \mathbb{R}^{M\times d_{n-1}}$ is the fixed basis matrix. We assume that the elements of $\mathbf{W^{(n)}}$, $\mathbf{\Lambda^{(n)}}$ and the bias vector $\boldsymbol{b^{(n)}}$ are statistically independent of each other. Therefore we can omit the bias vector $\boldsymbol{b^{(n)}}$ on the variance. We assume that the outputs of different neurons in each layer of the neural network are independent. Then the left-hand side of equation 14 expands as follows:

$$Var(\boldsymbol{w_i^{(n)}}\boldsymbol{x}) = Var(\sum_{j=1}^{d_{n-1}} w_{ij}^{(n)} x_j) = \sum_{j=1}^{d_{n-1}} Var(w_{ij}^{(n)} x_j), \tag{15}$$

where x_j is the j-th element of $\boldsymbol{x}$. We let x_j have the same distribution for $j = 1, \ldots, d_{n-1}$ accroding to Kaiming initialization [2]. Then, the equation 15 can be replaced by $\sum_{j=1}^{d_{n-1}} Var(w_{ij}^{(n)} x_1)$:

$$Var(\boldsymbol{w_i^{(n)}}\boldsymbol{x}) = Var(x_1 \sum_{j=1}^{d_{n-1}} w_{ij}^{(n)}). \tag{16}$$

Similarly, for the right-hand side, we also have that:

$$Var(\boldsymbol{\lambda_i^{(n)}}\mathbf{B^{(n)}}\boldsymbol{x}) = Var(x_1 \sum_{t=1}^{d_{n-1}}\sum_{j=1}^{M} \lambda_{ij}^{(n)} b_{jt}^{(n)}). \tag{17}$$

As the following equation holds true [1]:

$$Var(XY) = Var(X)Var(Y) + (E(X))^2 Var(Y) + (E(Y))^2 Var(x), \tag{18}$$

where X, Y are the independent random variables and $E(\cdot)$ denotes the mathematical expectation.
By this, we further expand equation 14 as follows:

$$\begin{aligned}
Var(\boldsymbol{w_i^{(n)}}\boldsymbol{x}) &= Var(x_1)Var(\sum_{j=1}^{d_{n-1}} w_{ij}^{(n)}) + (E(x_1))^2 Var(\sum_{j=1}^{d_{n-1}} w_{ij}^{(n)}) + (E(\sum_{j=1}^{d_{n-1}} w_{ij}^{(n)}))^2 Var(x_1), \\
Var(\boldsymbol{\lambda_i^{(n)}}\mathbf{B^{(n)}}\boldsymbol{x}) &= Var(x_1)Var(\sum_{t=1}^{d_{n-1}}\sum_{j=1}^{M} \lambda_{ij}^{(n)} b_{jt}^{(n)}) + (E(x_1))^2 Var(\sum_{t=1}^{d_{n-1}}\sum_{j=1}^{M} \lambda_{ij}^{(n)} b_{jt}^{(n)}) + (E(\sum_{t=1}^{d_{n-1}}\sum_{j=1}^{M} \lambda_{ij}^{(n)} b_{jt}^{(n)}))^2 Var(x_1).
\end{aligned} \tag{19}$$

Note that a sufficient condition to make equation 14 hold true is that $Var(\sum_{t=1}^{d_{n-1}}\sum_{j=1}^{M} \lambda_{ij} b_{jt}^{(n)}) = Var(\sum_{j=1}^{d_{n-1}} w_{ij}^{(n)})$ and $E(\lambda_{ij}^{(n)}) = 0$. Hence, we let $\lambda_{ij}^{(n)} \sim U(-a, a), a > 0$ to satisfy that $E(\lambda_{ij}^{(n)}) = 0$, which a is a parameter to be determined. And $Var(\lambda_{ij}^{(n)}) = \frac{a^2}{3}$. Further, from the above variance constraint, we have that:

$$Var(\sum_{t=1}^{d_{n-1}}\sum_{j=1}^{M} \lambda_{ij}^{(n)} b_{jt}^{(n)}) = \sum_{t=1}^{d_{n-1}}\sum_{j=1}^{M} {b_{jt}^{(n)}}^2 Var(\lambda_{ij}^{(n)}) = \sum_{j=1}^{M}\sum_{t=1}^{d_{n-1}} {b_{jt}^{(n)}}^2 Var(\lambda_{ij}^{(n)}) = \sum_{j=1}^{M}(\sum_{t=1}^{d_{n-1}} {b_{jt}^{(n)}}^2) Var(\lambda_{ij}^{(n)}). \tag{20}$$

We simply make $(\sum_{t=1}^{d_{n-1}} {b_{jt}^{(n)}}^2) Var(\lambda_{ij}^{(n)}) = \frac{Var(\sum_{j=1}^{d_{n-1}} w_{ij}^{(n)})}{M}$. We can have that :

$$Var(\lambda_{ij}^{(n)}) = \frac{Var(\sum_{j=1}^{d_{n-1}} w_{ij}^{(n)})}{M\sum_{t=1}^{d_{n-1}} {b_{jt}^{(n)}}^2} = \frac{a^2}{3} \tag{21}$$

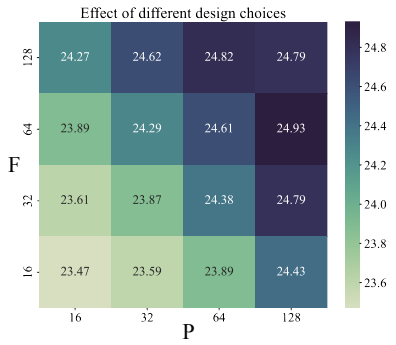


Figure 1. Visualization of the effect from differnet design choices (x-axis for phase number, y-axis for frequency number and colormap for PSNR value). More experimental details can be found in the Section D.

When $w_{ij}^{(n)} \sim U(-\sqrt{\frac{6}{d_{n-1}}}, \sqrt{\frac{6}{d_{n-1}}})$, we have that $a = \sqrt{\frac{6}{M\sum_{t=1}^{d_{n-1}} {b_{jt}^{(n)}}^2}}$. Thus, the following initialization scheme for ReLU activation function can be obtained as follows:

$$\lambda_{ij}^{(n)} \sim U(-\sqrt{\frac{6}{M\sum_{t=1}^{d_{n-1}} {b_{jt}^{(n)}}^2}}, \sqrt{\frac{6}{M\sum_{t=1}^{d_{n-1}} {b_{jt}^{(n)}}^2}}). \tag{22}$$

The initialization scheme for other activation function can be obtained by the similar deduction. As for the parameters without reparameterization, the initialization scheme is remained unchanged.

C. Results of image fitting on more metrics

In the main text, we have reported the results of our FR method on MLP+ReLU, MLP+ReLU+PE and MLP+Sin under the evaluation metric of PSNR. In this section, we adopt more popular evaluation metrics, *i.e.* SSIM and LPIPS. The evaluation is conducted directly on the representations from 2D color image approximation. From Table 1, there are average improvements of 4.5% and 22.7% in SSIM and LPIPS, respectively. These enhancements are notable in image reconstruction.

Table 1. More metrics results in image fitting.

metric	MLP+ReLU	MLP+ReLU+FR	MLP+ReLU+PE	MLP+ReLU+PE+FR	MLP+Sin	MLP+Sin+FR
SSIM↑	0.6985	0.7276	0.8587	0.9283	0.9747	0.9791
LPIPS↓	0.68	0.63	0.39	0.23	0.14	0.11

D. Design choices analysis

As discussed in the main text, our FR method has two hyper-parameters: frequency number F and phase number P. In this section, we provide experimental results to show the effects of different combinations of F and P. We vary the frequency number F and phase number P from 16 to 128, the approximation accuracy by different design choices can be found in Fig. 1. The approximation accuracy is the average PSNR on the first 3 images of Kodak 24 dataset. Generally, reparameterizing the weight matrix with more bases will lead to better approximation accuracy. Also, we need to have a balanced number of frequencies and phases to achieve good results.

E. 2D Color image approximation for more activation functions and input adjustment techniques

In the main text, we evaluated our FR method on MLP+ReLU, MLP+ReLU+PE and MLP+Sin. In this section, we apply our FR method to more activation functions, *i.e.* the Tanh, the Gauss [5] and the Garbor wavelet [6] activation functions. The advanced input adjustment technique, *i.e.* DINER [8], is also applied with our method. We follow the previous MLP structures and Fourier basis settings. The same training strategy is adopted for these experiments. For DINER coupled with Sin activation function, we early stops at 3000 iterations as its fast convergence. In Table 2, we report the PSNR achieved by

Table 2. Peak signal to noise ratio (PSNR) of 2D color image approximation results by different methods. MLP+Gauss denotes the MLP with Gauss activation function [5]. MLP+CGW denotes the MLP equipped with complex Gabor wavelet activation function [6]. MLP+ReLU+DINER denotes the MLP+ReLU coupled with the adjusted input features by a hash-table [8]. Detailed experiment settings can be found in Section 5.1 and Appendix E.

Method	Kodim 01	Kodim 02	Kodim 03	Kodim 04	Kodim 05	Kodim 06	Kodim 07	Kodim 08	Average
MLP + ReLU	19.37	26.12	25.11	24.57	17.31	21.69	20.79	15.68	21.33
MLP + ReLU + FR	20.34	26.58	27.21	25.72	18.33	22.25	22.47	16.64	22.44
MLP + ReLU + PE	24.47	31.41	31.53	30.16	22.87	26.54	29.33	21.14	27.18
MLP + ReLU + PE + FR	27.64	33.92	34.45	33.23	26.78	29.83	34.13	24.70	30.59
MLP + Sin	31.59	36.55	39.59	36.66	33.05	34.10	39.96	31.00	35.31
MLP + Sin + FR	33.45	38.68	39.58	37.96	34.64	34.45	39.76	32.16	36.34
MLP + Tanh	17.30	22.18	21.05	19.94	15.47	19.96	18.47	15.52	18.74
MLP + Tanh + FR	19.15	24.95	25.36	23.89	17.32	21.46	21.31	16.09	21.19
MLP + Gauss	24.83	30.29	31.32	30.40	24.79	25.78	29.40	22.42	27.40
MLP + Gauss + FR	24.86	30.19	31.40	31.05	24.98	25.53	27.75	24.98	27.59
MLP + CGW	26.53	32.18	32.60	31.97	25.96	28.29	32.19	23.70	29.18
MLP + CGW + FR	28.54	33.56	35.09	33.60	28.12	30.17	34.47	25.68	31.15
MLP + ReLU + DINER	45.65	50.28	37.57	44.01	39.69	42.54	44.50	41.15	43.17
MLP + ReLU + DINER + FR	45.81	50.53	38.06	43.79	40.42	43.13	44.45	41.35	43.44
MLP + Sin + DINER	45.00	50.36	37.84	41.83	40.76	44.50	44.47	41.62	43.30
MLP + Sin + DINER + FR	47.45	50.67	44.74	46.65	40.89	43.85	42.92	43.56	45.09

these three INRs, DINER and previous models for approximating the first 8 images in the Kodak 24 dataset. The same as our results in the main text, our FR consistently improves the approximation accuracy for all the evaluated activation functions and input adjustments techniques. Some visual examples of the learned approximations by different models can be found in the following Fig. 2, 4, 5.

F. Representing shapes for more activation functions and scenes

Following the previous experimental settings and model structures, we further evaluate our FR method with Tanh, Gauss [5] and complex Gabor wavelet [6] activation functions on the Thai statue and add the Dragon statue. Our FR method helps models to capture more accurate complex shapes of the statue. In Fig. 6, 7, we visualize the shape representation results by these methods.

G. Learning neural radiance fields for more scenes

In the task of learning neural radiance fields, We evaluate our FR method on the original NeRF [3] and two recent SOTAs with neural networks, *i.e.* the DVGO [4] and the InstantNGP [7]. For the original NeRF, we set $F = 128, P = 32$. As for the small two-hidden-layer MLP of the DVGO and the InstantNGP, where the width of the hidden layers is 128 and 64, we reparameterize the weight matrix between consecutive hidden layers and empirically set $F = 64, P = 64$ and $F = 32, P = 64$ to ensure over-complete bases. The same experimental settings and training strategies as the original works are adopted.

In Table 3, we list the full results of three frameworks on the Blender dataset [3]. Our FR method leads to more accurate view synthesis results. In Fig. 8, 9, 10, the detail of more reconstruction results is visualized.

Table 3. Peak signal to noise ratio (PSNR) of view synthesis results by different methods on the Blender dataset [3]. NeRF+FR, DVGO+FR and InstantNGP+FR denote the frameworks of the original NeRF [3], DVGO [4] and InstantNGP [7] trained with FR. NeRF is reproduced on the "NeRF-pytorch" codebase [9]. Detailed experiment settings can be found in Section 5.3 and Appendix G.

Framework	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship	Average
NeRF [3]	32.72	25.06	26.83	36.38	32.55	29.55	32.92	27.95	30.50
NeRF + FR (ours)	32.73	25.12	30.15	36.45	32.59	29.56	33.07	28.30	31.00
DVGO [7]	34.07	25.39	32.66	36.77	34.65	29.59	33.15	29.02	31.91
DVGO + FR (ours)	34.16	25.45	32.89	36.86	34.78	29.73	33.26	29.08	32.03
InstantNGP [4]	35.55	25.85	34.19	37.28	36.04	29.61	36.37	30.51	33.18
InstantNGP + FR (ours)	35.64	25.88	34.23	37.39	36.10	29.63	36.66	30.92	33.31

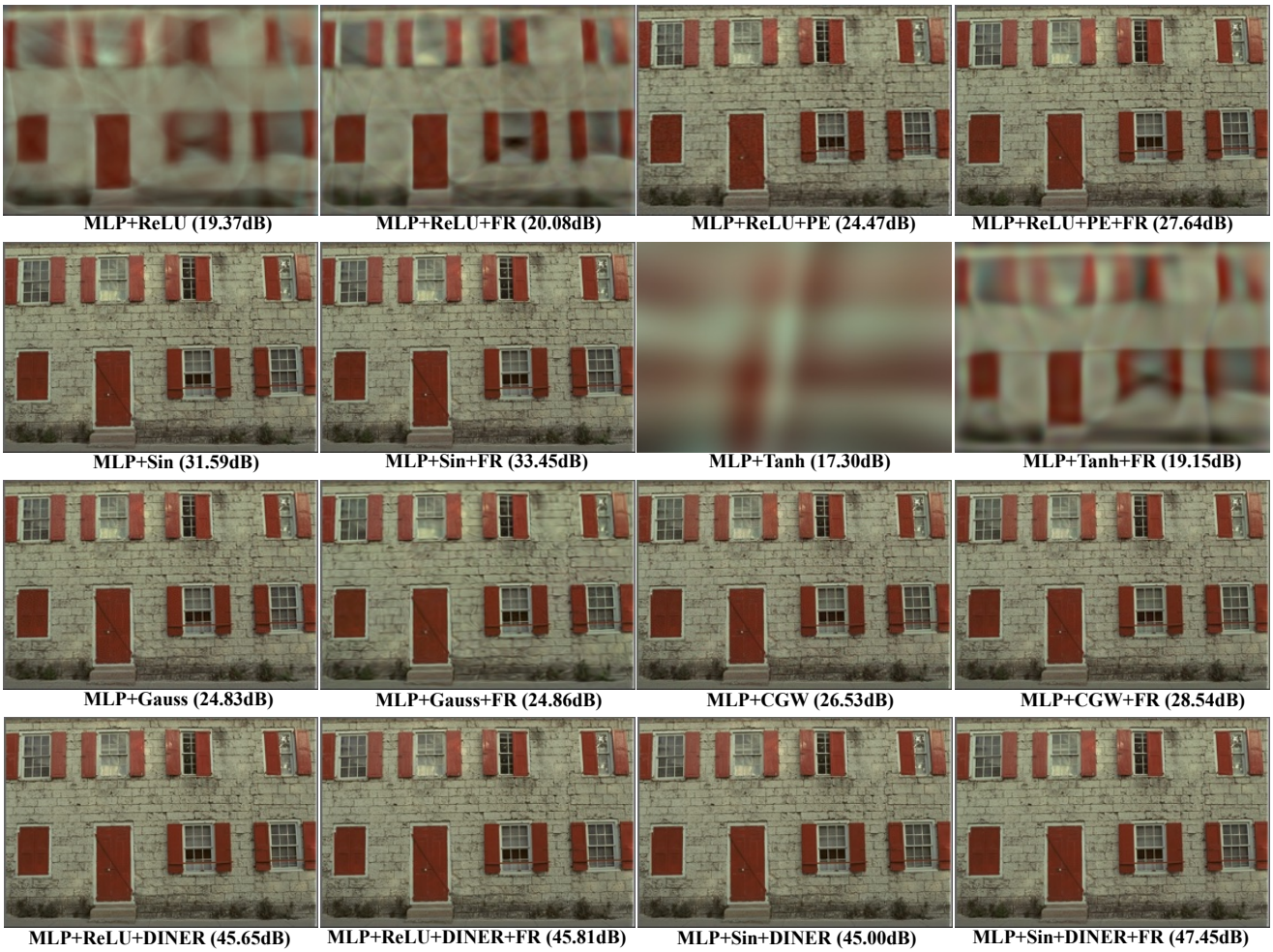


Figure 2. Peak signal to noise ratio (PSNR) of 2D color image approximation results on Kodim 01. MLP+Gauss denotes the MLP with Gauss activation function [5]. MLP+CGW denotes the MLP equipped with complex Gabor wavelet activation function [6]. MLP+ReLU+DINER denotes the MLP+ReLU coupled with the adjusted input features by a hash-table [8]. Detailed experiment settings can be found in Section 5.1 and Appendix E.

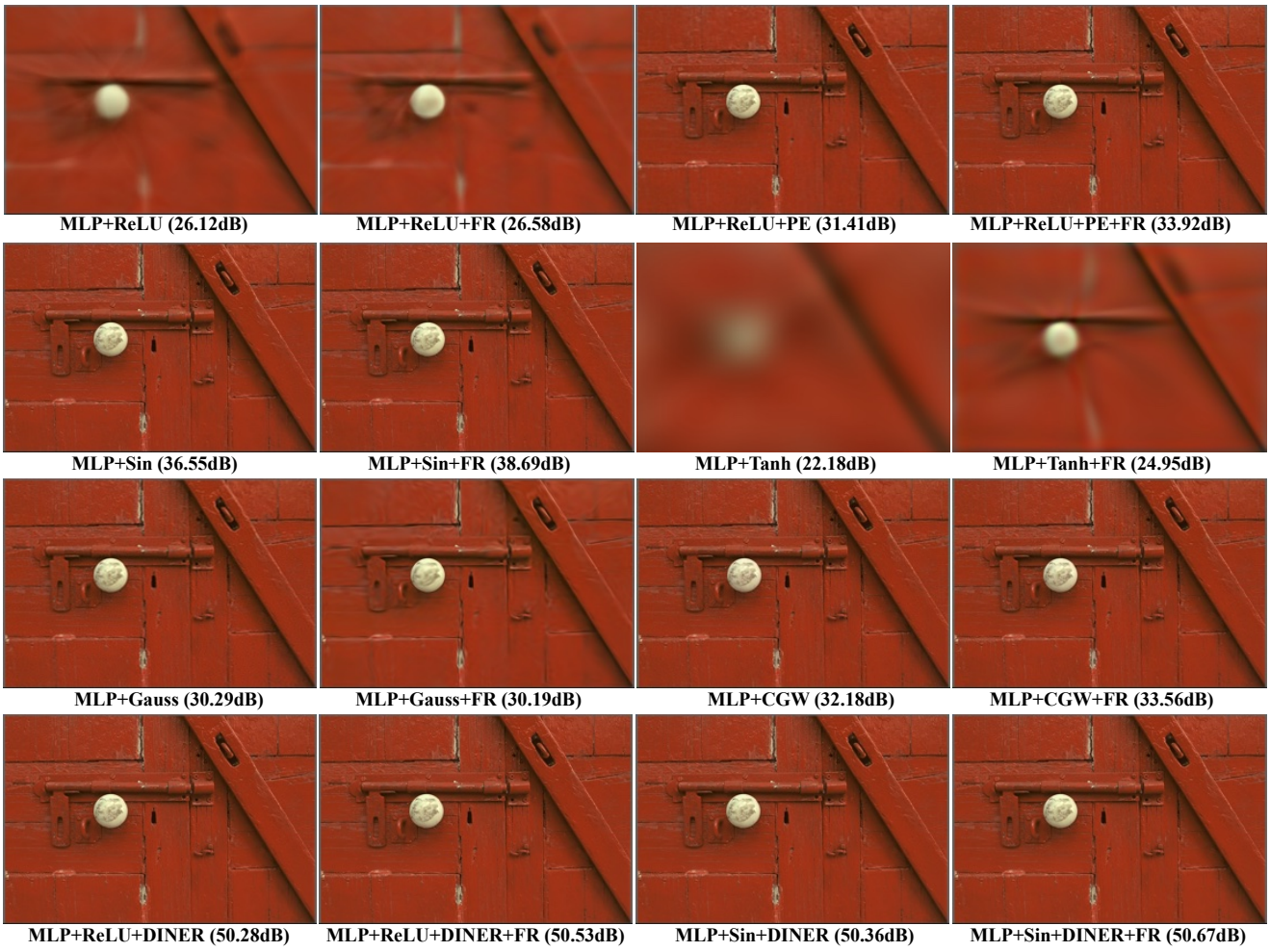


Figure 3. Peak signal to noise ratio (PSNR) of 2D color image approximation results on Kodim 02. MLP+Gauss denotes the MLP with Gauss activation function [5]. MLP+CGW denotes the MLP equipped with complex Gabor wavelet activation function [6]. MLP+ReLU+DINER denotes the MLP+ReLU coupled with the adjusted input features by a hash-table [8]. Detailed experiment settings can be found in Section 5.1 and Appendix E.

Figure 4. Peak signal to noise ratio (PSNR) of 2D color image approximation results on Kodim 04. MLP+Gauss denotes the MLP with Gauss activation function [5]. MLP+CGW denotes the MLP equipped with complex Gabor wavelet activation function [6]. MLP+ReLU+DINER denotes the MLP+ReLU coupled with the adjusted input features by a hash-table [8]. Detailed experiment settings can be found in Section 5.1 and Appendix E.

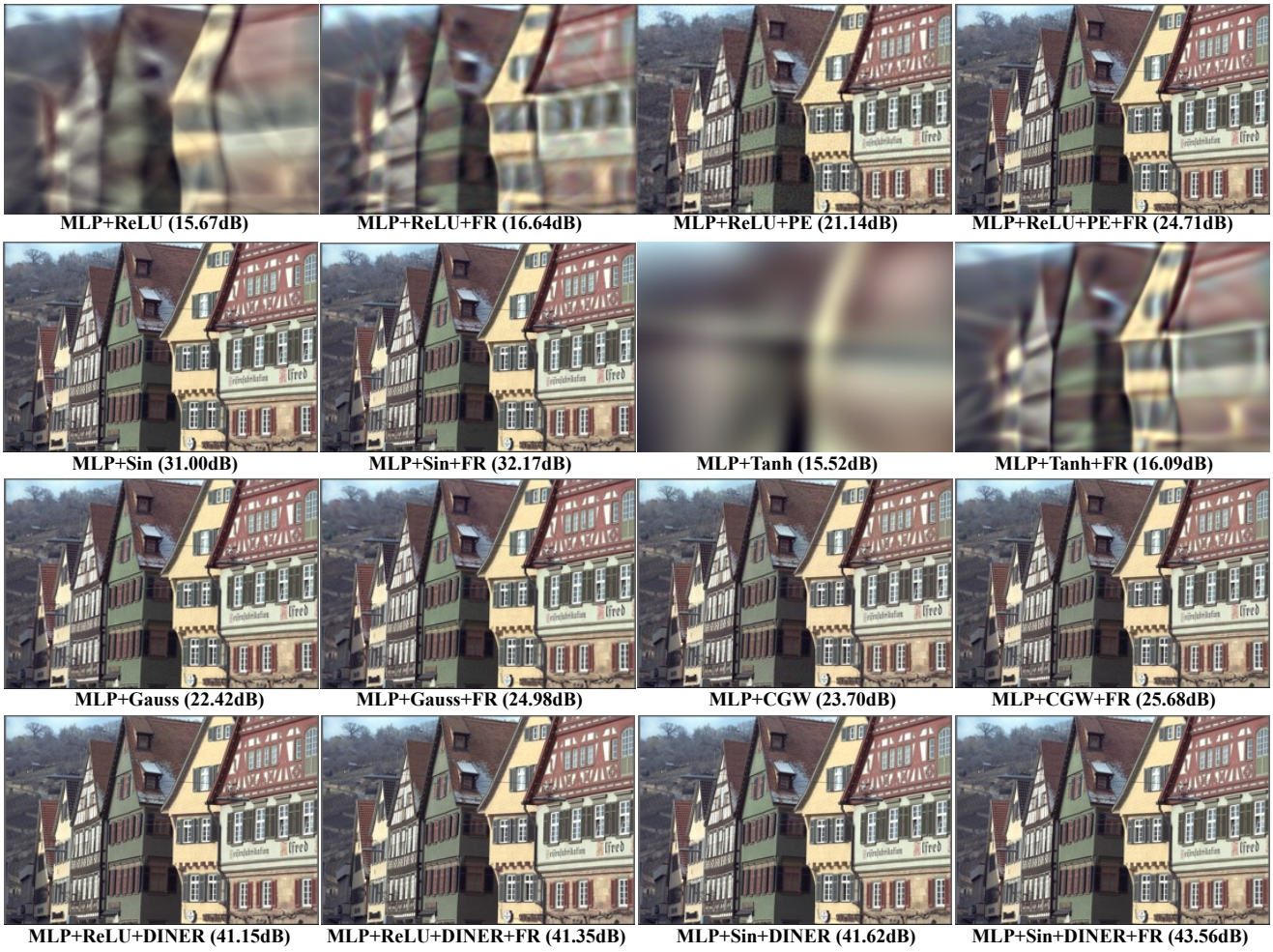


Figure 5. Peak signal to noise ratio (PSNR) of 2D color image approximation results on Kodim 08. MLP+Gauss denotes the MLP with Gauss activation function [5]. MLP+CGW denotes the MLP equipped with complex Gabor wavelet activation function [6]. MLP+ReLU+DINER denotes the MLP+ReLU coupled with the adjusted input features by a hash-table [8]. Detailed experiment settings can be found in Section 5.1 and Appendix E.

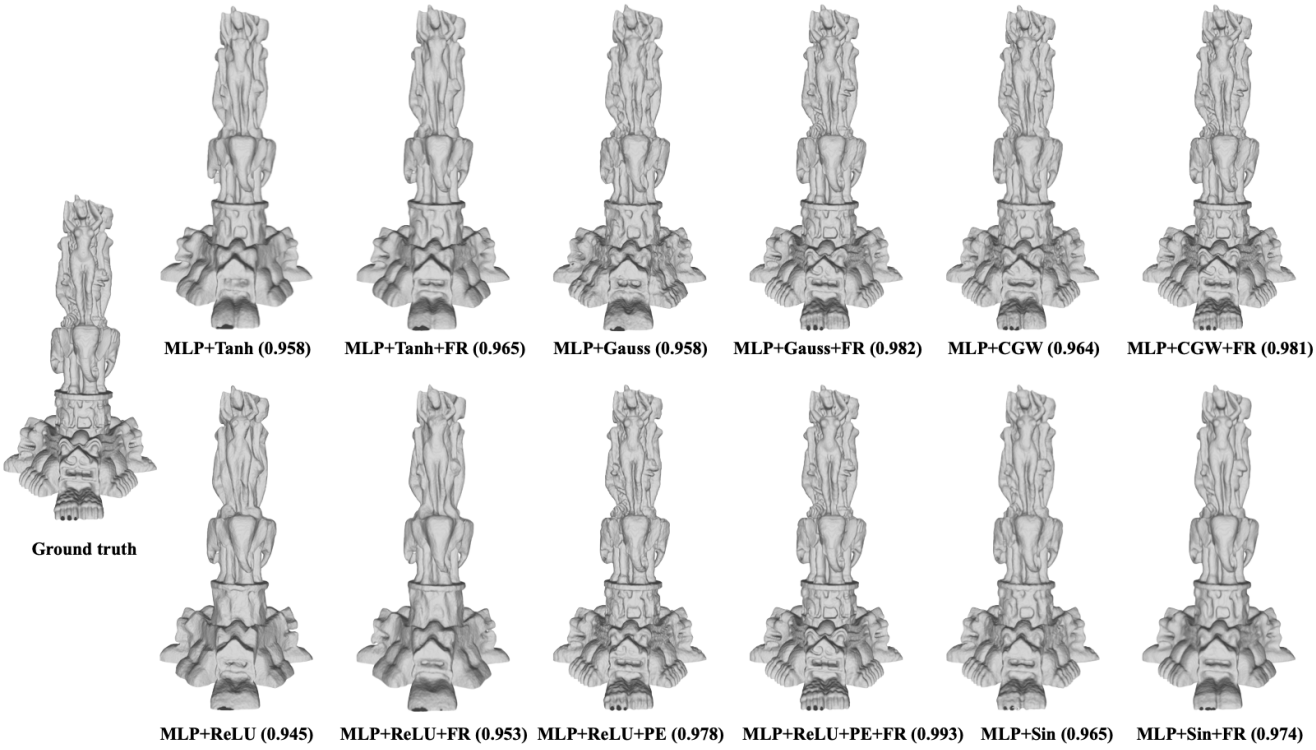


Figure 6. Visualization examples of the shape representation results (IOU) by different methods on Thai statue. Detailed experiment settings can be found in Section F.

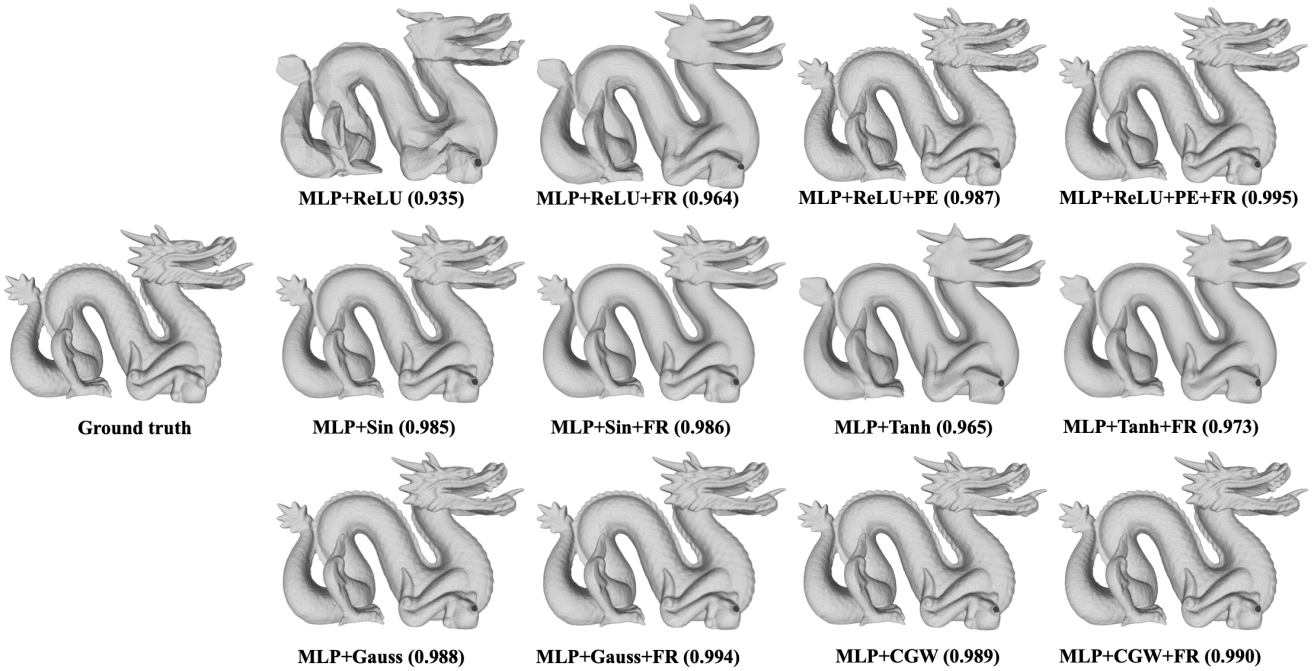
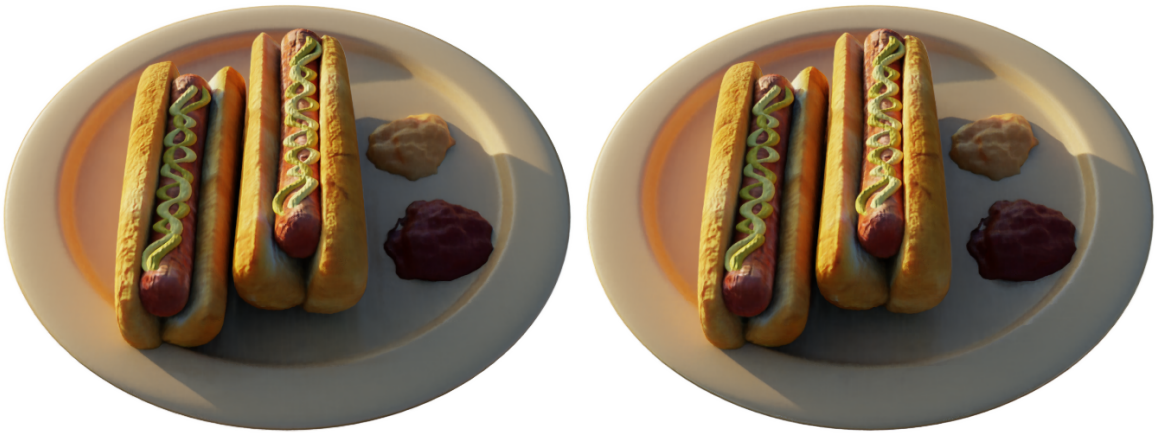
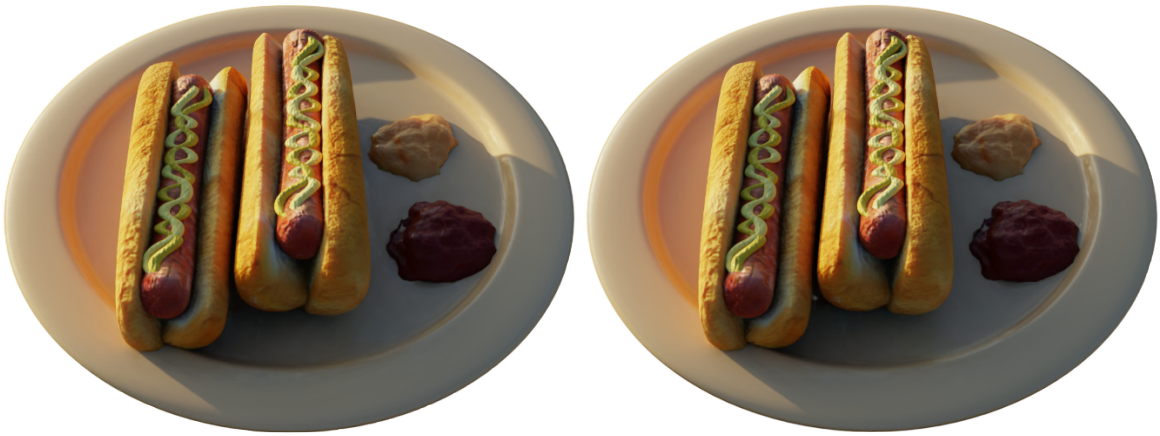


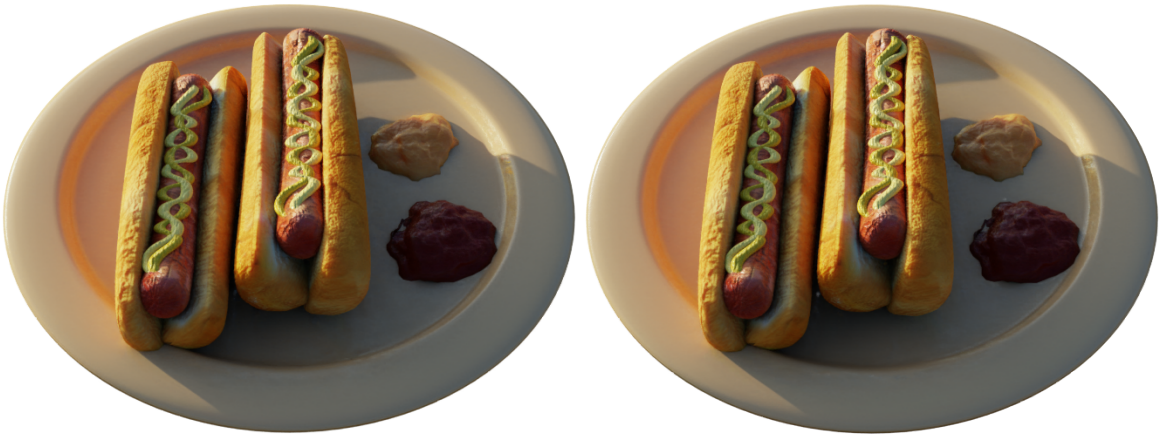
Figure 7. Visualization examples of the shape representation results (IOU) by different methods on Dragon Statue. Detailed experiment settings can be found in Section F.



NeRF (36.38dB) **NeRF+FR (36.45dB)**



DVGO (36.77dB) **DVGO+FR (36.86dB)**



InstantNGP (37.28dB) **InstantNGP+FR (37.39dB)**

Figure 8. Visualization examples of the view synthesis results (PSNR) of "Hotdog" by learning neural radiance fields. Detailed experiment settings can be found in Section G.

NeRF (32.92dB)

NeRF+FR (33.07dB)

DVGO (33.15dB)

DVGO+FR (33.26dB)

InstantNGP (36.37dB)

InstantNGP+FR (36.66dB)

Figure 9. Visualization examples of the view synthesis results (PSNR) of "Mic" by learning neural radiance fields. Detailed experiment settings can be found in Section G.

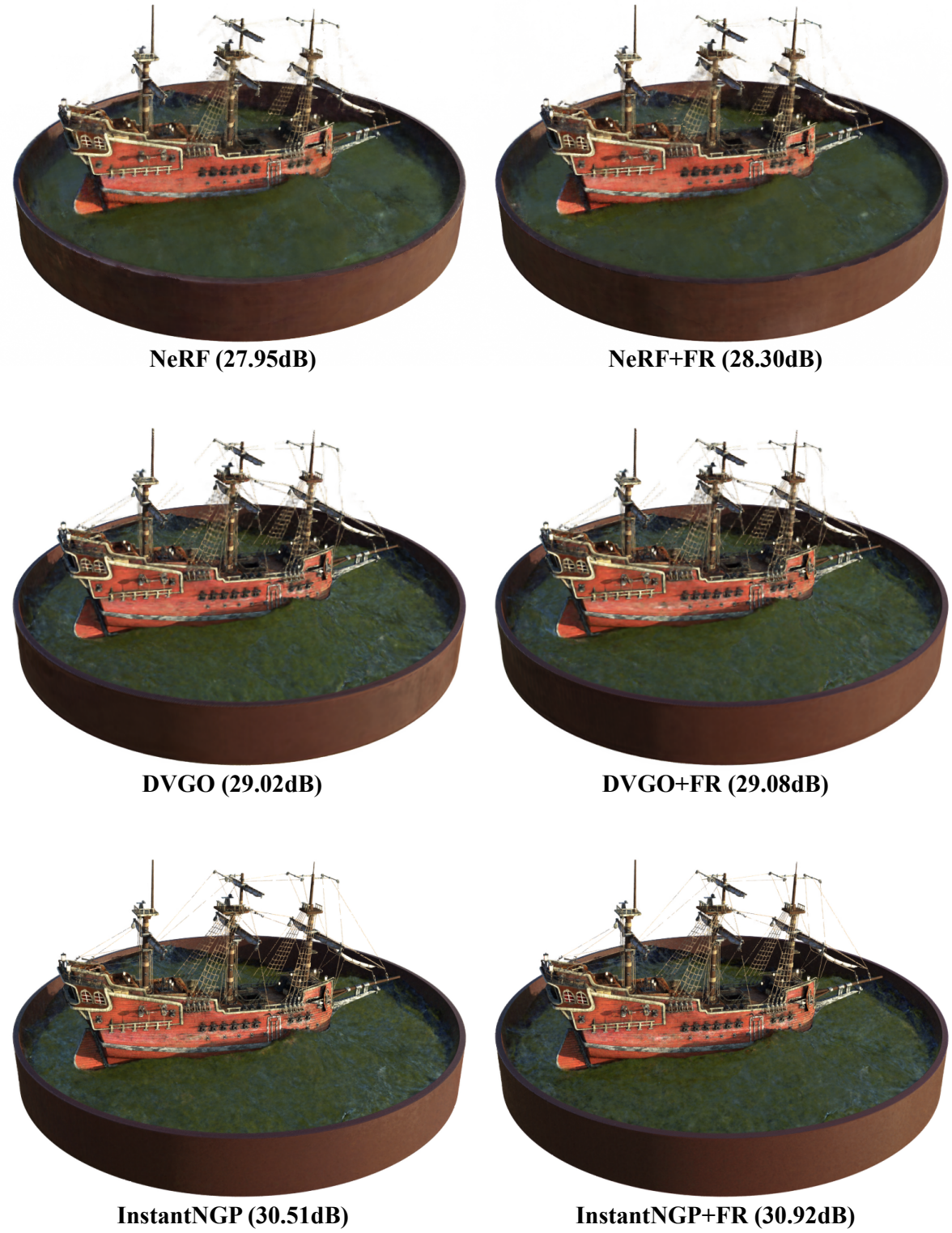


Figure 10. Visualization examples of the view synthesis results (PSNR) of "Ship" by learning neural radiance fields. Detailed experiment settings can be found in Section G.

References

[1] Leo A. Goodman. On the exact variance of products. *Journal of the American Statistical Association*, 55:708–713, 1960. 3

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[3] 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, 2020. 1, 5, 6

[4] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 1, 5, 6

[5] Sameera Ramasinghe and Simon Lucey. Beyond periodicity: Towards a unifying framework for activations in coordinate-mlps. In *European Conference on Computer Vision*, pages 142–158. Springer, 2022. 4, 5, 6, 7, 8, 9

[6] Vishwanath Saragadam, Daniel LeJeune, Jasper Tan, Guha Balakrishnan, Ashok Veeraraghavan, and Richard G. Baraniuk. Wire: Wavelet implicit neural representations, 2023. 4, 5, 6, 7, 8, 9

[7] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 5, 6

[8] Shaowen Xie, Hao Zhu, Zhen Liu, Qi Zhang, You Zhou, Xun Cao, and Zhan Ma. Diner: Disorder-invariant implicit neural representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6143–6152, 2023. 4, 5, 6, 7, 8, 9

[9] Lin Yen-Chen. Nerf-pytorch. https://github.com/yenchenlin/nerf-pytorch/, 2020. 6