DP-NeRF: Deblurred Neural Radiance Field with Physical Scene Priors Supplementary Material

Dogyoon Lee[1] Minhyeok Lee[1] Chajin Shin[1] Sangyoun Lee[1]
[1]Yonsei University
{nemotio, hydragon516, chajin, syleee}@yonsei.ac.kr

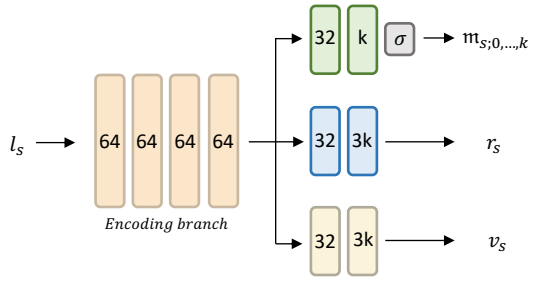


Figure 1. A detailed architecture of rigid blurring kernel (RBK).

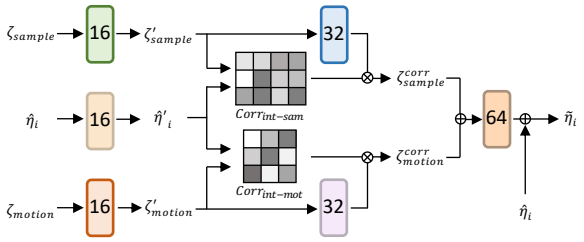


Figure 2. Detailed description about a correlation part of motion aggregation module (MAM). This picture includes process for computing correlation between ζ^p_{motion}, ζ^p_{sample}, and $\hat{\eta}^p_i$.

Notation	l_s	C_{d1}	C_{d2}	C_{d3}	C_{d4}	N_m	N_s	N_c	N_f
Value	64	128	64	32	16	$(1+k)$	(N_c+N_f)	64	64

Table 1. Notations for adaptive weight proposal (AWP) and corresponding values used in our experiment as default.

A. Additional Implementation Details

A.1. Training

DP-NeRF is implemented on two Nvidia RTX 3090 GPUs based on the published code and dataset for Deblur-NeRF [2] using PyTorch [5]. Training images are resized to 600×400 across the entire dataset to train the DP-NeRF. In addition, we start to optimize proposed components, rigid blurring kernel (RBK), adaptive weight proposal (AWP), and color composition (CC), after 1200 training iterations with a pure NeRF [3] to obtain coarse scene representation.

A.2. Architectural Detail

Figures 1, 2, 3, and 4 describe the DP-NeRF architecture to describe in detail. View information l_s for each image is embedded with 64 channels in a simple embedding layer.

Rigid Blurring Kernel (RBK). As shown in Figure 1, RBK consists of one shared encoding branch E and three decoding branches with simple MLPs ($\mathcal{W}$, $\mathcal{R}$, and $\mathfrak{L}$). Encoding branch E consists of an MLP with four fully-connected linear layers, with each layer having 64 dimensions and ReLU activation function. Decoding branches for $\mathfrak{m}_{s;0,\dots,k}$, r_s, and v_s also consist of an MLP with one linear layer with 32 dimensions and an output linear layer. The output channel dimensions for $\mathfrak{m}_{s;0,\dots,k}$, r_s, and v_s is k, $3k$, and $3k$, respectively, where k is a hyper-parameter that controls the number of rigid camera motions. Note that, $\mathfrak{m}_{s;0,\dots,k}$ is normalized along the motion axis k to ensure $\sum_{i=0}^{k} \mathfrak{m}_{s;i} = 1$.

Adaptive Weight Proposal (AWP). First of all, we additionally describe a motivation of the complex architecture design of AWP. Intuition of AWP architecture is to fully use the spatial occupancy information of samples on the rays. When we use the similar module based on inter-motion correlation with rendered depth values on each rays, we could not get the improved results. The reason was supposed to be insufficient occupancy information of one scalar rendered depth value per ray. Hence, we design AWP to fully reflect the information to use rich correlation between the rays. Here are description of AWP architecture with detailed notations. Please refer to Figure 2, Figure 4, and Table 1 for architecture design and corresponding notations. Note that, we omit the notation for the batch dimension for clarity. Before forwarding to motion aggregation module (MAM), extracted depth feature $\zeta^p_{i,j} \in \mathbb{R}^{N_m \times N_s \times C_{d1}}$ from the second-to-last layer of the NeRF is embedded in $\hat{\zeta}^p_{i,j} \in \mathbb{R}^{N_m \times N_s \times C_{d2}}$ via the simple four-layered MLP with ReLU activation, where N_m and N_s denote the dimensions of the motion axis and sample axis, respectively. N_m is the number of blurring rays, which is a summation of the number of motion (k) and original ray (1). N_s is the total number of samples, which is a summation of the number of

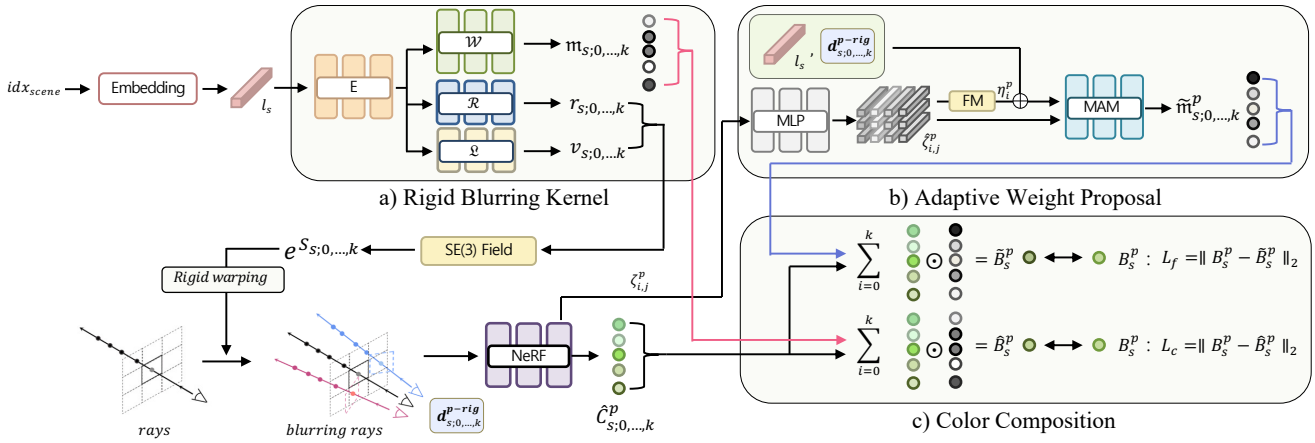


Figure 3. Overall pipeline of DP-NeRF, which is same as figure in the main paper.

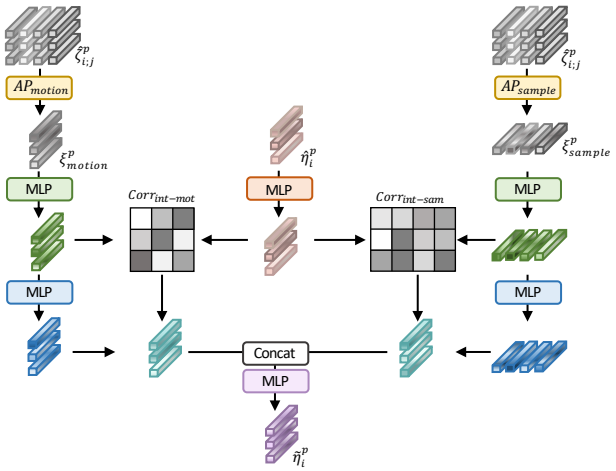


Figure 4. Architecture of motion aggregation module (MAM), which is same as figure in the main paper.

coarse samples (N_c) and fine samples (N_f). We then, apply feature modulation (FM) [8] to generate ray-wise representative features, $\eta_i^p \in \mathbb{R}^{N_m \times C_{d2}}$. To impose the view- and direction-information for the rays, l_s and positional embedded $\mathbf{d}_{s;0,\dots,k}^{p-rig}$ are concatenated and forwarded to the simple MLP to extract $\hat{\eta}_i^p \in \mathbb{R}^{N_m \times C_{d3}}$. To aggregate the implicit information between the modeled blurring rays, we forward $\hat{\zeta}_{i,j}^p$ and $\hat{\eta}_i^p$ to the MAM. The MAM then aggregates the extracted features from $\hat{\zeta}_{i,j}^p$ and $\hat{\eta}_i^p$ based on the computed correlation between the decomposed information along each motion and sample axis.

The MAM is formulated in the main paper as follows:

$$MAM(\hat{\zeta}_{i,j}^p, \hat{\eta}_i^p) = MLP\big(cat(\mathbf{corr}(\hat{\eta}_i^p, \xi_{motion}^p, \xi_{sample}^p))\big), \tag{1}$$

where $\zeta_{motion}^p \in \mathbb{R}^{N_s \times C_{d3}}$ and $\zeta_{sample}^p \in \mathbb{R}^{N_m \times C_{d3}}$ are embedded via a one-layer MLP and attentive pooling [1] along the motion and sample axis. Modulated feature η_i^p and attentive pooled features ζ_{motion}^p and ζ_{sample}^p are embedded again in $\hat{\eta}_i'^p$, $\zeta_{sample}'^p$ and $\zeta_{motion}'^p$ as the same channel dimension C_{d4} before computing the correlation maps. We then compute two correlations, $Corr_{int-mot}$ and $Corr_{int-sam}$, which represent the inter-motion and inter-sample correlations, respectively (Figure 2). Because $\hat{\zeta}_{i,j}^p$ consists of a sample dimension and a motion dimension, we apply inter-sample and inter-motion design to investigate the inter-sample and inter-motion correlations. The each type of correlation score maps enhances the correlation of each dimension and aggregate them with modulated features via matrix multiplication. The enhanced features, ζ_{sample}^{corr} and ζ_{motion}^{corr}, are then concatenated and forwarded to the 64-channel MLP, followed by the residual connection of $\hat{\eta}_i$ to generate our final motion-aggregated feature $\tilde{\eta}_i$. Note that, the dimensions of $\hat{\eta}_i$ and $\tilde{\eta}_i$ are the same.

Tone Mapping. Before color composition for each color from the blurring rays and the composition weights $\mathfrak{m}_{s;0,\dots,k}$ or $\tilde{\mathfrak{m}}_{s;0,\dots,k}^p$, we should note that each color $\hat{C}_{s;0,\dots,k}^p$ is tone-mapped from the predicted scene irradiance from a NeRF similar to [2]. Gamma function g for the tone mapping function is simply set as shown in Eq. 2 in a similar manner to [2] because there is no significant difference in performances when either a gamma function or learnable MLP is employed as the tone mapping function.

$$g(c') = c'^{\frac{1}{2.2}}, \tag{2}$$

where c' denotes the predicted radiance from the NeRF in DP-NeRF. The types of tone-mapping function does not significantly influence the predicted radiance due to the consistent exposure of the dataset. Hence, we can rewrite the full color composition of $\hat{B}_s^p$ and $\tilde{B}_s^p$ as shown in Eq. 3, with g omitted in the main paper due to page limitations.

$$\begin{aligned} \hat{B}_s^p &= \mathfrak{m}_{s;0}\, g(\hat{C}_{s;0}^p) + \sum_{i=1}^{k} \mathfrak{m}_{s;i}\, g(\hat{C}_{s;i}^{p-rig}) \\ \tilde{B}_s^p &= \tilde{\mathfrak{m}}_{s;0}^p\, g(\hat{C}_{s;0}^p) + \sum_{i=1}^{k} \tilde{\mathfrak{m}}_{s;i}^p\, g(\hat{C}_{s;i}^{p-rig}) \end{aligned} \tag{3}$$

B. Evaluation Results on Real Scene

Quantitative Results. We present quantitative evaluation results for camera motion and defocus blur in the real scene dataset in Tables 2 and 3. The results show that DP-NeRF improves the quantitative performance for all of the evaluation metrics, especially on LPIPS. In addition, it is clear that PSNR and SSIM cannot fully represent the realistic perceptual quality of rendered images as argued by Nerfies [4] in their paper. Hence, the perceptual quality should be evaluated by comparing LPIPS and the rendered image quality visually.

Qualitative Results. We present the qualitative results in Figures 5 and 6. The results show that DP-NeRF(RBK) and DP-NeRF(RBK+AWP) both outperform the NeRF and baseline in terms of perceptual quality for most of the scenes. It demonstrates that our model produces more accurate 3D reconstruction quality with a realistic, clean NeRF. In addition, DP-NeRF enhances the 3D geometric and appearance consistency in several scenes. For example, due to the complex depth and similar texture of the forest region, baselines [2,3] have difficulty in predicting the correct geometry in the **Heron** scene (4th-row in Figure 5). However, our model predicts the region more accurately, illustrating that DP-NeRF can model complex 3D space more accurately.

C. Additional Ablation Results

C.1. RBK Analysis

Modeling Analysis. We illustrate how our RBK models camera motion and defocus blur as ray rigid transformation by presenting a visualization of the modeled kernel and rendered images from each of the transformed camera views in Figures 7 and 8. Figures 7 (a) and 8 (a) shows the transformed ray origin and direction derived from the trained DP-NeRF with paired images. Note that, each colored transformed camera origin and ray direction pairs with a rendered image with the same colored box. Each image also has the same notation as presented in the main paper ($\hat{C}^p_{s;0}$ and $\hat{C}^p_{s;1,\dots,k}$) to assist in understanding. Figures 7 (b)-(f) and 8 (b)-(f) present rendered images from the transformed cameras, which are used to composite the blurred image $\tilde{B}^p_s$. Figure 7 (g) and 8 (g) present the composited blurred image from images (b)-(f) with composition weights as we mentioned in the main paper. We demonstrate that the composited blurred image $\tilde{B}^p_s$ is successfully generated, with a similar appearance to the reference image B^p_s (Figures 7 (h) and 8 (h)). Figures 7 and 8 show that DP-NeRF can model the blurred image $\tilde{B}^p_s$ so that it is similar to reference image B^p_s for both type of the blur.

The visualizations of given ray $\mathbf{r}^p_s (= \mathbf{r}^p_{s;0})$ and rigidly transformed (RT) rays $\mathbf{r}^{p-rig}_{s;1,\dots,k}$ in Figures 7 (a) and 8 (a) demonstrate that the camera motion and defocus blur can be successfully modeled with the RBK imitating camera movement or focus plain decision by rigidly warping the given scene camera with the $SE(3)$ field. It is obvious that camera motion blur can be successfully modeled using the RBK because it models the blurring process using rigid camera transformation. Figure 7 (a) presents the results of the camera shaking during image acquisition, which leads to camera motion blur.

Interestingly, for defocus blur, there is a plane where the RT rays intersect (Figure 8 (a)), and this is the predicted focus plane. Specifically, RBK imitates the defocus blurring by approximating the depth of field process locating the transformed rays on the virtual aperture. The predicted transforms of the rays naturally decide the focus plane as we described in Figure 8 (a). If the depth value of a given ray is not on focus plane, focus blur is naturally induced by the color composition of the given ray and the other RT rays, which penetrate the other surrounding parts of the scene.

In addition, Figures 7 (b)-(h) and 8 (b)-(h) show that we can render images related to the construction of the blurred image for a scene. This form of image decomposition can determine how the blurred image in a scene is generated during image acquisition with respect to change in the camera settings, such as camera motion or focus plain information. Furthermore, RBK modeling guarantees the geometric and appearance consistency of each rendered image.

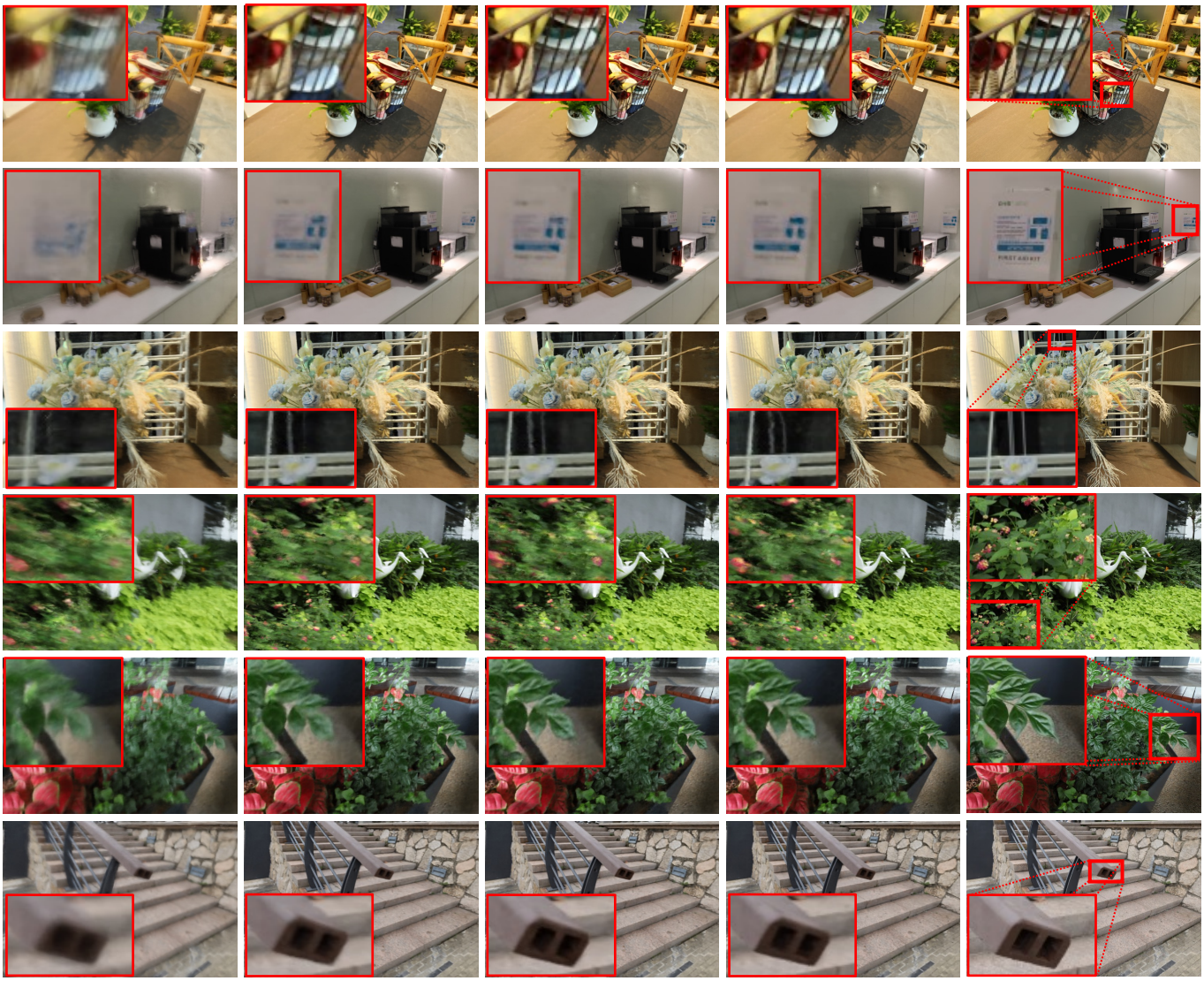
C.2. Effectiveness of the RBK and AWP

We present the effectiveness of the RBK and AWP with ablation analysis using the synthetic (Table 4) and real scene (Tables 5 and 6 for camera motion and defocus blur, respectively) datasets. Qualitative comparisons are also presented in Figures 5 and 6. For real scene dataset, it seems to be marginal improvement with AWP. Actually, blurring kernel is affected by the depth in case of the out-of-plane camera motion blur and general defocus blur as mentioned in [6,7]. However, for provided camera motion blur dataset, the blur type is close to in-plane camera motion blur, which leads to the marginal improvement. In addition, shape of the blurring kernel can change depending the depth in both blur types. Instead to directly model kernel shape change, we mitigate the issue through adaptive weights between transformed rays from AWP. Our model shows the more satisfying results with geometrically clean images with RBK. In addition, we can find that our model with AWP get more detailed clean results thanks to above adaptive weights approximation.

Camera Motion	Ball			Basket			Buick			Coffee			Decoration		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [3]	24.08	0.6237	.3992	23.72	.7086	.3223	21.59	.6325	.3502	26.48	.8064	.2896	22.39	.6609	.3633
Deblur-NeRF [2]	27.36	.7656	.2230	27.67	.8449	.1481	24.77	.7700	.1752	30.93	.8981	.1244	24.19	.7707	.1862
DP-NeRF	27.20	.7652	.2088	27.74	.8455	.1294	25.70	.7922	.1405	31.19	.9049	.1002	24.31	.7811	.1639

Camera motion	Girl			Heron			Parterre			Puppet			Stair			Average		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [3]	20.07	.7075	.3196	20.50	.5217	.4129	23.14	.6201	.4046	22.09	.6093	.3389	22.87	.4561	.4868	22.69	.6347	.3687
Deblur-NeRF [2]	22.27	.7976	.1687	22.63	.6874	.2099	25.82	.7597	.2161	25.24	.7510	.1577	25.39	.6296	.2102	25.63	.7675	.1820
DP-NeRF	23.33	.8139	.1498	22.88	.6930	.1914	25.86	.7665	.1900	25.25	.7536	.1505	25.59	.6349	.1772	25.91	.7751	.1602

Table 2. Quantitative results for the real scene camera motion blur. Each color shading indicates the best and second-best result, respectively.



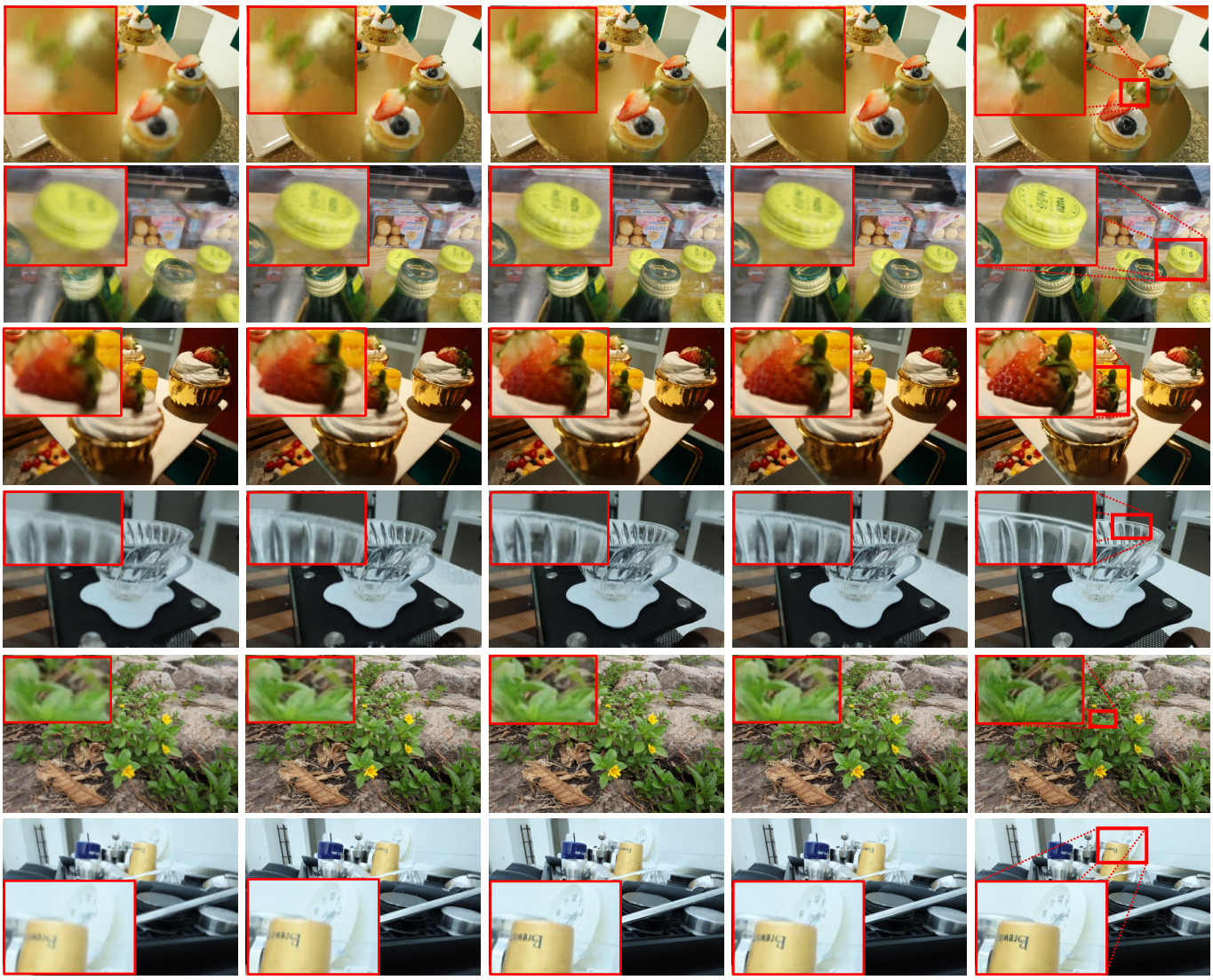
a) Naïve NeRF b) Deblur-NeRF c) DP-NeRF(RBK) d) DP-NeRF(RBK+AWP) e) Reference

Figure 5. Rendered novel view synthesis results of DP-NeRF for real scene camera motion blur. Figures (a)-(e) denote Naive NeRF, Deblur-NeRF, ours(RBK), ours(RBK+AWP), and ground truth images, respectively. Red colored box in corner of images are enlarged part of same colored box region in reference images.

Defocus	Cake			Caps			Cisco			Cupcake			Coral		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF	24.42	.7210	.2250	22.73	.6312	.2801	20.72	.7217	.1256	21.88	.6809	.2155	19.81	.5658	.2689
Deblur-NeRF [2]	26.27	.7800	.1282	23.87	.7128	.1612	20.83	.7270	.0868	22.26	.7219	.1160	19.85	.5999	.1214
DP-NeRF	26.16	.7781	.1267	23.95	.7122	.1430	20.73	.7260	.0840	22.80	.7409	.0960	20.11	.6107	.1178

Defocus	Cups			Daisy			Sausage			Seal			Tools			Average		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [3]	25.02	.7581	.2315	22.74	.6203	.2621	17.79	.4830	.2789	22.79	.6267	.2680	26.08	.8523	.1547	22.40	.6661	.2310
Deblur-NeRF [2]	26.21	.7987	.1271	23.52	.6870	.1208	18.01	.4998	.1796	26.04	.7773	.1048	27.81	.8949	.0610	23.46	.7199	.1207
DP-NeRF	26.75	.8136	.1035	23.79	.6971	.1075	18.35	.5443	.1473	25.95	.7779	.1026	28.07	.8980	.0539	23.67	.7299	.1082

Table 3. Quantitative results for the real scene defocus blur. Each color shading indicates the best and second-best result, respectively.



a) Naïve NeRF b) Deblur-NeRF c) DP-NeRF(RBK) d) DP-NeRF(RBK+AWP) e) Reference

Figure 6. Rendered novel view synthesis results of DP-NeRF for real scene defocus blur. Figures (a)-(e) denote Naive NeRF, Deblur-NeRF, ours(RBK), ours(RBK+AWP), and ground truth images, respectively. Red colored box in corner of images are enlarged part of same colored box region in reference images.

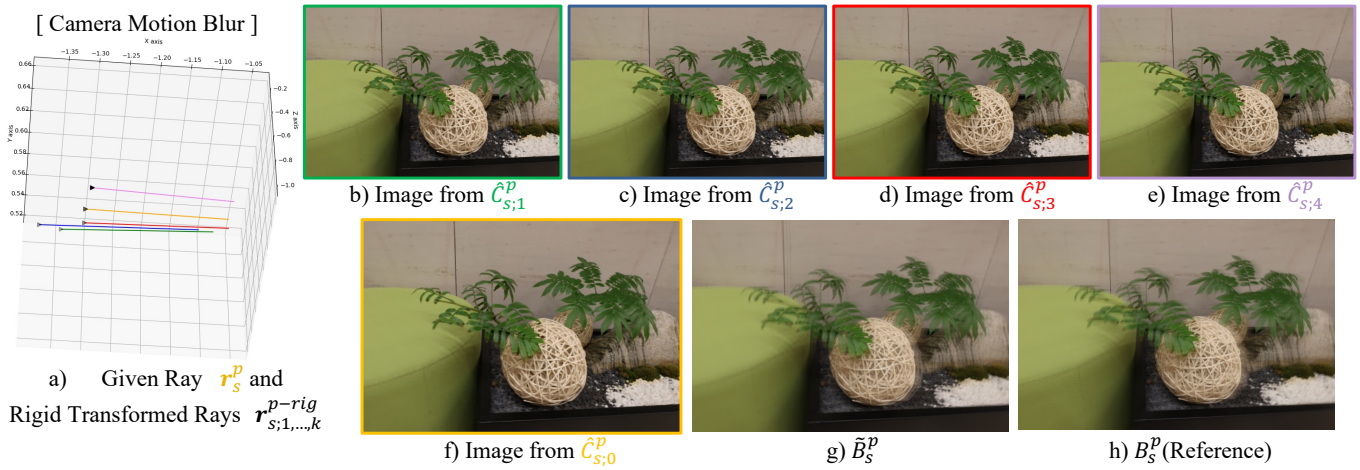


Figure 7. RBK analysis on camera motion blur for real **blurball** scene. Figure (a) denotes visualization result of original and rigid transformed camera origin and direction on given reference view, which is presented in Figure (h). Figures (b)-(f) denote rendered images from given and transformed cameras in Figure (a). Figure (g) denotes a composited blurred image from Figures (b)-to-(f) with composition weights to predict a reference image, Figure (h). Figures (g) and (h) are used in DP-NeRF as RGB reconstruction loss.

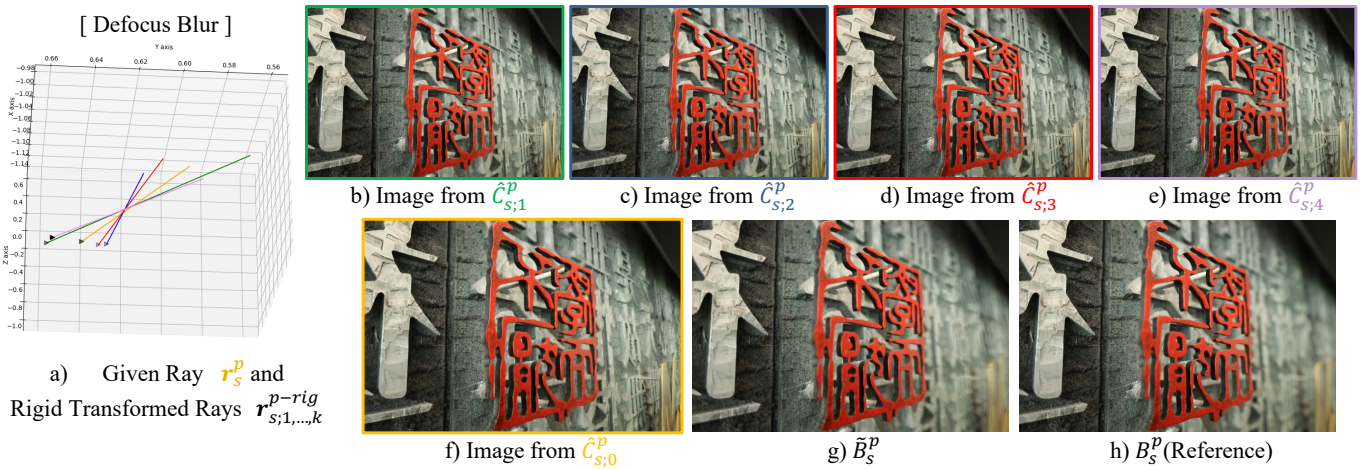


Figure 8. RBK analysis on defocus blur for real **defocusseal** scene. Figure (a) denotes visualization result of original and rigid transformed camera origin and direction on given reference view, which is presented in Figure (h). Figures (b)-(f) denote rendered images from camera and transformed cameras in Figure (a). Figure (g) denotes a composited blurred image from Figures (b)-(f) with composition weights to predict a reference image, Figure (h). Figure (g) and (h) are used in DP-NeRF as RGB reconstruction loss.

Camera Motion	Factory			Cozyroom			Pool			Tanabata			Trolley			Average		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [3]	19.32	.4563	.5304	25.66	.7941	.2288	30.45	.8354	.1932	22.22	.6807	.3653	21.25	.6370	.3633	23.78	.6807	.3362
DP-NeRF(RBK)	25.82	.7853	.2493	32.75	.9331	.0351	31.98	.8756	.0925	27.63	.8745	.1045	27.93	.8723	.1130	29.22	.8682	.1189
DP-NeRF(RBK+AWP)	25.91	.7787	.2494	32.65	.9317	.0355	31.96	.8768	.0908	27.61	.8748	.1033	28.03	.8752	.1129	29.23	.8674	.1184
Defocus	Factory			Cozyroom			Pool			Tanabata			Trolley			Average		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [3]	25.36	.7847	.2351	30.03	.8926	.0885	27.77	.7266	.3340	23.80	.7811	.2142	22.67	.7103	.2799	25.93	.7791	.2303
DP-NeRF(RBK)	28.56	.8672	.1052	32.00	.9207	.0410	31.18	.8482	.1577	26.51	.8586	.0802	26.00	.8277	.1200	28.85	.8645	.1008
DP-NeRF(RBK+AWP)	29.26	.8793	.1035	32.11	.9215	.0386	31.44	.8529	.1563	27.05	.8635	.0779	26.79	.8395	.1170	29.33	.8713	.0987

Table 4. Ablation study for the synthetic scene. Each color shading indicates the best and second-best result, respectively.

Camera Motion	Ball			Basket			Buick			Coffee			Decoration		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [3]	24.08	.6237	.3992	23.72	.7086	.3223	21.59	.6325	.3502	26.48	.8064	.2896	22.39	.6609	.3633
DP-NeRF(RBK)	27.15	.7641	.2112	27.35	.8367	.1347	24.93	.7791	.1545	30.72	.8949	.1070	24.15	.7730	.1700
DP-NeRF(RBK+AWP):	27.20	.7652	.2088	27.74	.8455	.1294	25.70	.7922	.1405	31.19	.9049	.1002	24.31	.7811	.1639
Camera motion	Girl			Heron			Parterre			Puppet			Stair			Average		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [3]	20.07	.7075	.3196	20.50	.5217	.4129	23.14	.6201	.4046	22.09	.6093	.3389	22.87	.4561	.4868	22.69	.6347	.3687
DP-NeRF(RBK)	22.19	.7934	.1575	22.55	.6831	.1970	25.81	.7635	.1931	25.19	.7497	.1493	25.68	.6446	.1799	25.57	.7682	.1654
DP-NeRF(RBK+AWP)	23.33	.8139	.1498	22.88	.6930	.1914	25.86	.7665	.1900	25.25	.7536	.1505	25.59	.6349	.1772	25.91	.7751	.1602

Table 5. Ablation study for the real scene camera motion blur. Each color shading indicates the best and second-best result, respectively.

Defocus	Cake			Caps			Cisco			Cupcake			Coral		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [3]	24.42	.7210	.2250	22.73	.6312	.2801	20.72	.7217	.1256	21.88	.6809	.2155	19.81	.5658	.2689
DP-NeRF(RBK)	25.80	.7704	.1251	23.72	.7003	.1486	20.68	.7232	.0889	22.51	.7331	.1003	20.02	.6052	.1183
DP-NeRF(RBK+AWP)	26.16	.7781	.1267	23.95	.7122	.1430	20.73	.7260	.0840	22.80	.7409	.0960	20.11	.6107	.1178
Defocus	Cups			Daisy			Sausage			Seal			Tools			Average		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
Naive NeRF [3]	25.02	.7581	.2315	22.74	.6203	.2621	17.79	.4830	.2789	22.79	.6267	.2680	26.08	.8523	.1547	22.40	.6661	.2310
DP-NeRF(RBK)	26.59	.8086	.1077	23.77	.6968	.1059	18.40	.5448	.1452	26.04	.7767	.0996	27.87	.8947	.0540	23.54	.7254	.1093
DP-NeRF(RBK+AWP)	26.75	.8136	.1035	23.79	.6971	.1075	18.35	.5443	.1473	25.95	.7779	.1026	28.07	.8980	.0539	23.67	.7299	.1082

Table 6. Ablation study for the real scene defocus blur. Each color shading indicates the best and second-best result, respectively.

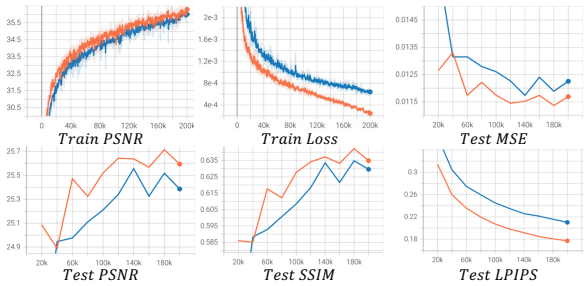


Figure 9. Training of DP-NeRF and Deblur-NeRF on *Stair* scene.

Model	PSNR(↑)	SSIM(↑)	LPIPS(↓)
DP-NeRF(RBK)	24.93	.7791	.1545
DP-NeRF(RBK+AWP)	25.47	.7883	.1466
DP-NeRF(RBK+AWP+Coarse-to-Fine)	25.70	.7922	.1405

Table 7. Ablation of coarse-to-fine optimization on *Buick* scene.

C.3. Ablation of Coarse to Fine Optimization

We present the optimization graph for Deblur-NeRF [2] and DP-NeRF in Figure 9. Though it seems to be harder to optimize the DP-NeRF due to higher degree of freedom, DP-NeRF converges more faster than Deblur-NeRF thanks to the shared rigid motions in a single view, which works as regularization to optimize the RBK. In addition, we attach ablation results of the coarse-to-fine optimization in Table 7. It reveals that the proposed optimization scheme helps the model to take the full advantage of AWP and improves the visual quality. The time required to train ***200000*** iterations are around ***9.5 hours*** for Deblur-NeRF and ***19 hours*** for DP-NeRF on *two NVIDIA RTX 3090 GPUs*. Note that, we present representative scenes for Figure 9 and Table 7 due to time limitation to experiment on all scenes.

C.4. Ablation of the Number of Rigid Motions

We present the quantitative and qualitative results for the ablation analysis of the number of rigid motions in Table 8 and Figures 10 and 11. As we mentioned, LPIPS represents perceptual quality better than PSNR and SSIM do. Based on this metric, the perceptual quality of DP-NERF gradually improves as the number of rigid motions increases. In addition, DP-NeRF exhibits higher perceptual quality than Deblur-NeRF [2] in terms of LPIPS for the same number of composition rays. Furthermore, RBK+AWP produces a better performance than does the RBK alone.

C.5. Visualization of Rigid Transformed Rays

We present additional visualization of rigid transformed rays predicted by the RBK for a specific view. Figure 12 and 13 show the transformed rays according to the change of the number of rigid motions k for camera motion and defocus blur, respectively. The rays colored as orange are original center rays in all visualization, which are forwarded to RBK to generate the other rigidly warped rays in both Figure 12 and 13. They demonstrate that RBK successfully model the camera shaking and virtual aperture for camera motion and defocus blur, respectively.

C.6. Additional Error Map Visualization

As mentioned in the main paper, we present a full error map visualization without the emphasis box in Figure 14. Note that, brighter colors indicate greater error. We control the brightness and contrast in all output identically, to reveal the error between the models more prominently. Our model produces less error than the baselines for the images regardless of the blur type.

D. Supplementary Video

We attach videos outlining novel view synthesis. These videos are generated with spiral-path camera poses, which are widely used for visual comparison in NeRF-based research. Please watch theses supplementary videos or visit out project page to observe the qualitative effectiveness of DP-NERF.

# of RM	Camera Motion - Pool									Defocus - Pool								
	DeblurNeRF			DP-NeRF(RBK)			DP-NeRF(RBK+AWP)			DeblurNeRF			DP-NeRF(RBK)			DP-NeRF(RBK+AWP)		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
2	31.48	.8658	.1348	31.97	.8741	.1102	32.30	.8822	.1031	29.89	.8087	.2280	30.24	.8246	.2023	30.52	.8301	.1971
3	31.52	.8670	.1289	31.61	.8682	.1032	31.80	.8709	.0966	30.07	.8117	.2094	29.22	.7923	.1891	29.39	.8006	.1871
4	31.49	.8672	.1257	31.98	.8756	.0925	31.96	.8768	.0908	30.43	.8230	.1931	31.18	.8482	.1577	31.44	.8529	.1563
5	31.59	.8710	.1169	31.40	.8671	.0912	31.88	.8763	.0849	30.52	.8244	.1832	30.99	.8421	.1485	31.07	.8473	.1484
6	31.59	.8688	.1149	31.43	.8662	.0868	31.74	.8733	.0807	30.70	.8310	.1712	31.42	.8549	.1368	31.73	.8621	.1362
7	31.42	.8628	.1121	31.65	.8699	.0851	31.67	.8714	.0806	30.73	.8317	.1728	31.15	.8484	.1328	31.62	.8589	.1302
8	31.56	.8685	.1097	31.49	.8683	.0840	31.49	.8668	.0796	30.92	.8352	.1625	31.37	.8532	.1267	31.77	.8609	.1234

Table 8. Quantitative results for ablation study of the number of rigid motions in the main paper, which is denoted as RM in the table. Each color shading indicates best and second-best result on each scene with each model, respectively. Note that, the number of kernel points in Deblur-NeRF is set to (the number of rigid motions + 1) for fair comparison, which means the same number of composition rays to create a blurred color for a pixel.

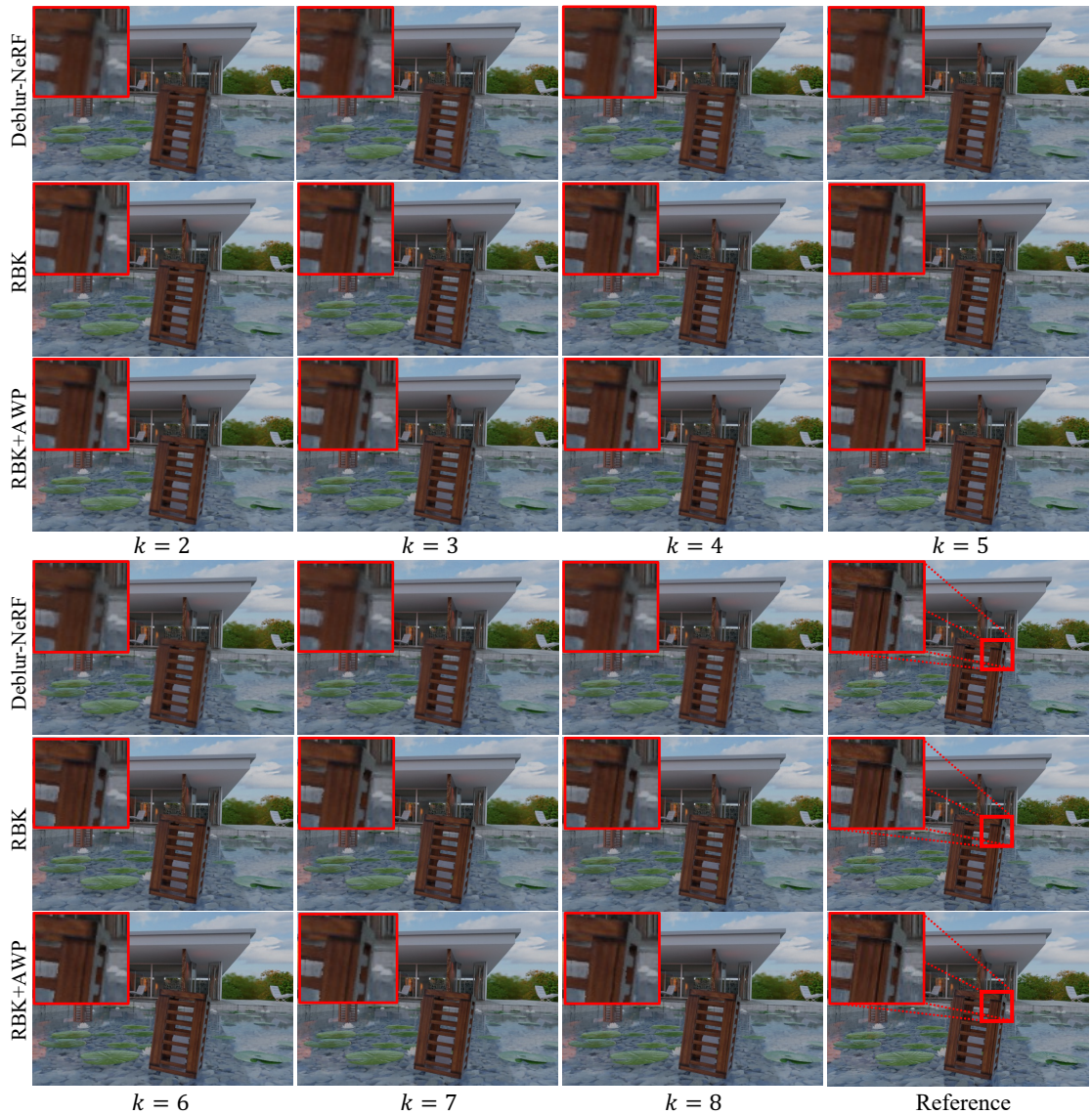
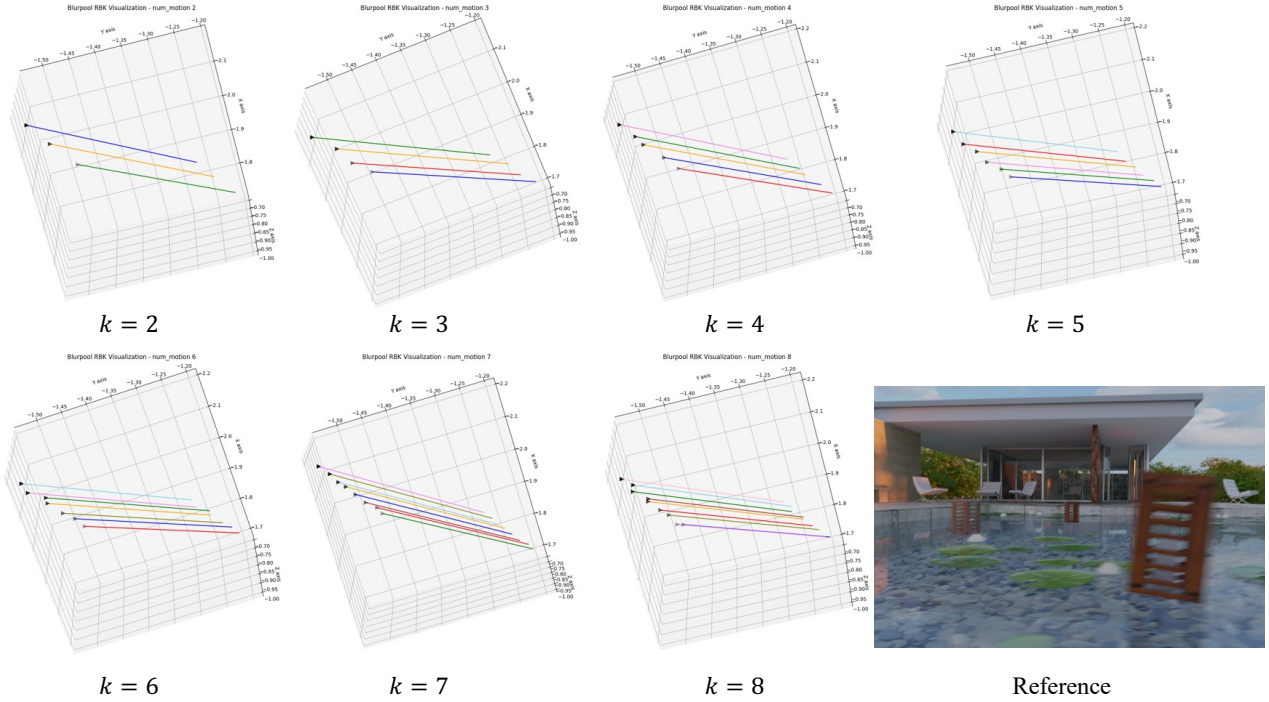


Figure 10. Qualitative results of ablation study depending on the number of rigid motion on camera motion blur scene. k denotes the number of rigid motion. Each row denotes Deblur-NeRF, DP-NeRF(RBK) and DP-NeRF(RBK+AWP), respectively. Red colored box in corner of images are enlarged part of same colored box region in reference images.

Figure 11. Qualitative results of ablation study depending on the number of rigid motion on defocus blur scene. k denotes the number of rigid motion. Each row denotes Deblur-NeRF, DP-NeRF(RBK) and DP-NeRF(RBK+AWP), respectively. Red colored box in corner of images are enlarged part of same colored box region in reference images.

[*Camera Motion Blur*]

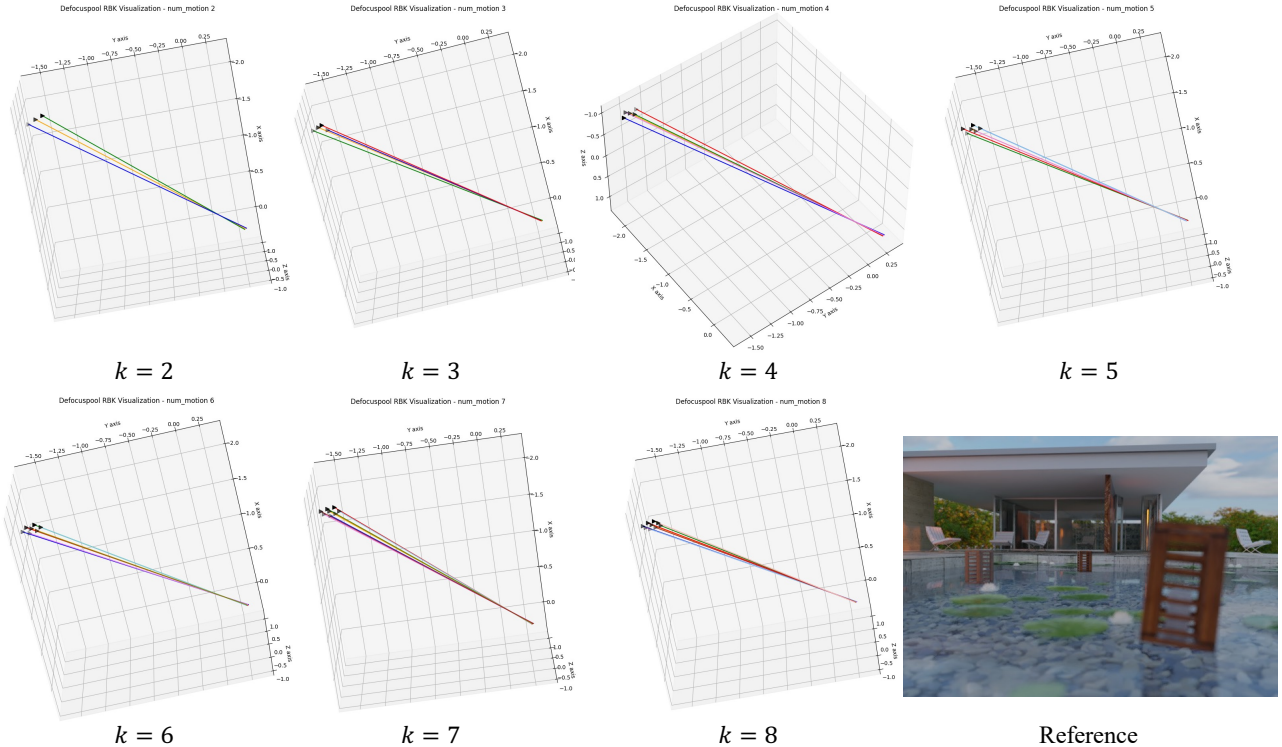


$k = 2$ $k = 3$ $k = 4$ $k = 5$

$k = 6$ $k = 7$ $k = 8$ Reference

Figure 12. Visualization of warped poses using rigid motions predicted by the RBK according to the change of the number of rigid motion k on camera motion blur.

[*Defocus Blur*]



$k = 2$ $k = 3$ $k = 4$ $k = 5$

$k = 6$ $k = 7$ $k = 8$ Reference

Figure 13. Visualization of warped poses using rigid motions predicted by the RBK according to the change of the number of rigid motion k on defocus blur.

Figure 14. Full error map visualization on defocus **Factory** and camera motion **Trolley** scene. In addition, we denote quantitative quality between reference and rendered images as average MSE(Mean Squared Error), denoted as green text on left bottom of each images.

References

[1] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020. 2

[2] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V Sander. Deblur-nerf: Neural radiance fields from blurry images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12861–12870, 2022. 1, 2, 3, 4, 5, 8

[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. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I*, pages 405–421, 2020. 1, 3, 4, 5, 7

[4] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 3

[5] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 1

[6] Pratul P Srinivasan, Rahul Garg, Neal Wadhwa, Ren Ng, and Jonathan T Barron. Aperture supervision for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6393–6401, 2018. 3

[7] Pratul P Srinivasan, Ren Ng, and Ravi Ramamoorthi. Light field blind motion deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3958–3966, 2017. 3

[8] Zelin Zhao and Jiaya Jia. End-to-end view synthesis via nerf attention. *arXiv preprint arXiv:2207.14741*, 2022. 2