

Supplementary Material– DOF-GS: Adjustable Depth-of-Field 3D Gaussian Splatting for Post-Capture Refocusing, Defocus Rendering and Blur Removal

Yujie Wang^{1,2,3}

Praneeth Chakravarthula^{3†}

Baoquan Chen^{1,2‡}

¹State Key Laboratory of General Artificial Intelligence, Peking University

²School of Intelligence Science and Technology, Peking University

³University of North Carolina at Chapel Hill

1. Derivatives of the differentiable DOF rendering

In this section, we elaborate on the derivation of key derivatives. Since the majority of the gradient calculations are already incorporated within the customized rasterization process [1], we focus solely on deriving the derivatives of the gradient pertaining to the affected or newly introduced variables in our differentiable depth-of-field (DOF) rendering process. For brevity, we omit the subscript index m for viewpoint.

Since the convolution of two Gaussians results in another Gaussian whose covariance matrix is the sum of the covariance matrices of the original Gaussians [9], the covariance of \mathcal{G}_k'' is derived as follows:

$$\Sigma_k'' = \Sigma_k' + \Sigma_k^{(\text{coc})} = \begin{bmatrix} \Sigma_k'(1, 1) + a & \Sigma_k'(1, 2) \\ \Sigma_k'(2, 1) & \Sigma_k'(2, 2) + a \end{bmatrix}. \quad (1)$$

By exploiting the chain rule, we can calculate the derivatives w.r.t. focal distance f , aperture parameter Q and depth value z_k through the following formulas:

$$\frac{d\Sigma_k''}{df} = \frac{d\Sigma_k''}{d\Sigma_k^{(\text{coc})}} \frac{d\Sigma_k^{(\text{coc})}}{df}, \quad (2)$$

$$\frac{d\Sigma_k''}{dQ} = \frac{d\Sigma_k''}{d\Sigma_k^{(\text{coc})}} \frac{d\Sigma_k^{(\text{coc})}}{dQ}, \quad (3)$$

and

$$\frac{d\Sigma_k''}{dz_k} = \frac{d\Sigma_k''}{d\Sigma_k'} \frac{d\Sigma_k'}{dz_k} + \frac{d\Sigma_k''}{d\Sigma_k^{(\text{coc})}} \frac{d\Sigma_k^{(\text{coc})}}{dz_k}. \quad (4)$$

From Equation (1), the derivative w.r.t. Σ_k' and $\Sigma_k^{(\text{coc})}$ can be obtained as: $\frac{d\Sigma_k''}{d\Sigma_k'} = \mathbf{I}$ and $\frac{d\Sigma_k''}{d\Sigma_k^{(\text{coc})}} = \mathbf{I}$. For calculating the derivative in Equation (2), we need to calculate the derivative w.r.t. f from a ($a = \frac{1}{2ln4} \left(R_k^{(\text{coc})}\right)^2$), which is given by

$$\frac{da}{df} = \frac{da}{dR_k^{(\text{coc})}} \frac{dR_k^{(\text{coc})}}{df} = \begin{cases} -\frac{1}{2ln4} \frac{R_k^{(\text{coc})} Q}{f^2}, & \text{if } z_k > f, \\ \frac{1}{2ln4} \frac{R_k^{(\text{coc})} Q}{f^2}, & \text{otherwise.} \end{cases} \quad (5)$$

And the derivative for Q is calculated as

$$\frac{dQ}{dz_k} = \frac{R_k^{(\text{coc})}}{2ln4} \left| \frac{1}{z_k} - \frac{1}{f} \right|. \quad (6)$$

Since $\frac{d\Sigma'_k}{dz_k}$ in Equation (4) has been provided in the rasterization-based rendering of 3DGS [1], we calculate gradient for z_k by additionally accumulating the derivative w.r.t. z_k relative to a :

$$\frac{da}{dz_k} = \begin{cases} -\frac{1}{2\ln 4} \frac{R^{(\text{coc})} Q}{(z_k)^2}, & \text{if } z_k < f, \\ \frac{1}{2\ln 4} \frac{R^{(\text{coc})} Q}{(z_k)^2}, & \text{otherwise.} \end{cases} \quad (7)$$

In our pipeline, which is designed to learn camera characteristics for the introduced depth-of-field rendering process and to recover scene details from uncalibrated images with moderate defocus blur, we heuristically initialize the focal distance f and aperture parameters Q for each input view and update them through optimization. The initialization strategy is discussed in Section 2.1. To prevent Gaussian points from being influenced by the focal distance parameters that are actively updated during the optimization process, we disable gradient flow from $\Sigma_k^{(\text{coc})}$ to z_k during the optimization.

2. Additional Implementation Details

In this section, we provide additional implementation details of our approach. Specifically, we outline the initialization of virtual cameras for training views in Section 2.1, describe the optimization settings in detail in Section 2.2, and present further information about parameter settings and the network structure of the In-Focus Localization Network (ILN) in Section 2.3.

2.1. Camera Parameter Initialization

As the labels for focal distances $\{f_m\}_{m=1}^M$ and aperture parameters $\{Q_m\}_{m=1}^M$ across different views are unavailable, an initialization process is required. We simply initialize the focal distances and aperture parameters to be the same value for different views, which avoids a too dedicated manual initialization. Specifically, to avoid introducing bias for focal distances before the optimization, we initialize f_m as the median diopter value among all Gaussian points:

$$\frac{1}{f_m} = \text{median} \left(\left\{ \frac{1}{z_k} \right\} \right), \quad m = 1, 2, \dots, M. \quad (8)$$

Initializing f_m to be the median diopter value also helps avoid heavily biased CoC radius values across different depth planes at early iterations. Meanwhile, we initialize the aperture parameter Q_m by ensuring that the CoC values for different points are not greater than a threshold τ . Specifically, the maximum CoC value across the scene occurs at the closest or most distant points when the focal plane is exactly at the opposite side. Thus Q_m is initialized by solving

$$Q_m \left| \frac{1}{\min \left\{ \frac{1}{z_k} \right\}} - \frac{1}{\max \left\{ \frac{1}{z_k} \right\}} \right| = \tau, \quad (9)$$

which ensures that the most CoC values across the entire scene are not greater than τ . In practice, we use 10-th and 90-th percentiles as minimum and maximum values ($\min \left\{ \frac{1}{z_k} \right\}$ and $\max \left\{ \frac{1}{z_k} \right\}$) respectively, to mitigate the influence of outliers within the initialized point cloud. The τ is empirically set to 15 for different scenes in our implementation, as the optimization process will actively update the camera parameters.

2.2. Optimization Process

We divide the entire training process into two stages: a warm-up stage and a refinement stage. In the initial 5,000 iterations, as the Gaussian points and camera parameters undergo significant changes, we employ only \mathcal{L}_{rec} as the training objective. This allows the Gaussian points and camera parameters to adjust to fit the training views using the DOF rendering process. After this stage, the Gaussian points and camera parameters become relatively stable, resulting in the rendered CoC map $\mathcal{M}^{(\text{coc})}$ becoming highly correlated with the pixel-wise blur extent within the training views. At this point, we begin leveraging CoC cues in conjunction with supervisions on the rendered All-in-Focus images to further refine and enhance scene details. Specifically, we introduce and use the In-Focus Localization Network (ILN) to identify in-focus regions within training views from the rendered CoC map and other information including rendered defocused image and depth map. Meanwhile, we deactivate \mathcal{L}_{rec} and activate the remaining three terms for subsequent iterations. Additionally, we observe that the initial point clouds, derived from blurry images that contain multi-view inconsistent blurry regions, are sparse. The sparse initialization requires a large amount of iterations for point cloud densification to recover the fine details. To enhance the optimization efficiency and detail recovery, we add 60,000 uniformly distributed points across the scene at the 2,000-th iteration, a point at which the scene S and camera parameters have been substantially optimized. The attributes of the newly added points are set according to its nearest neighbors. We briefly summarize the optimization process in Algorithm 1.

Algorithm 1: Optimization Process of DOF-GS

Input: Multi-view images $\{\mathbf{I}_m\}_{m=1}^M$ **Output:** 3D scene $\mathcal{S} = \{\mathcal{G}_k\}_{k=1}^K$, camera parameters $\{f_m, Q_m\}_{m=1}^M$

Estimate camera poses and a sparse initial point cloud from SfM;

Initialize camera parameters according to Section 2.1;

```
for  $iter < \text{number of iterations}$  do
    Randomly sample a viewpoint  $m$ ;
    Render a Defocused Image  $\tilde{\mathbf{I}}_m$ , CoC map  $\mathcal{M}_m^{(\text{coc})}$  and Depth map  $\tilde{\mathbf{D}}_m$ ;
    if  $iter < 5,000$  then
         $\mathcal{L} \leftarrow \mathcal{L}_{\text{rec}}$ ;
         $\mathcal{S}, f_m, Q_m \leftarrow \text{Adam}(\nabla \mathcal{L})$ ;
         $\triangleright$  Only activate  $\mathcal{L}_{\text{rec}}$ ;
         $\triangleright$  Backward and update 3D Gaussian points and related camera parameters;
    end
    else
        Render an All-in-Focus image  $\tilde{\mathbf{I}}_m^*$ ;
         $\mathcal{M}_m^* \leftarrow \text{ILN}(\tilde{\mathbf{I}}_m, \tilde{\mathbf{D}}_m, \mathcal{M}_m^{(\text{coc})}, \text{pe}(m), \text{pe}(\mathbf{x}))$ ;
         $\triangleright$  Use ILN to estimate an in-focus mask;
         $\mathcal{L} \leftarrow \mathcal{L}_{\text{detail}} + \lambda_{\text{mk}} \mathcal{L}_{\text{mk}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}$ ;
         $\triangleright$  Deactivate  $\mathcal{L}_{\text{rec}}$  and activate remaining loss terms;
         $\mathcal{S}, f_m, Q_m, \Omega \leftarrow \text{Adam}(\nabla \mathcal{L})$ ;
         $\triangleright$  Backward and update,  $\Omega$  denotes network parameters;
    end
end
```

2.3. Parameter Settings and Network Structure

Learning-Rate Settings. We set the initial learning rate for the center positions of 3D Gaussian points at 0.0005, which linearly decays to 0.000005 over 40,000 iterations. The learning rate for the scales of Gaussian points is set to 0.01. Moreover, the learning rate for focal distances is set at 0.05, and that for aperture parameters is set to 0.01. The In-Focus Localization Network is trained with a learning rate of 0.0005. The remaining parameters retain the settings given in the method[1].

Network Structure. In Table 1, we detail the structure of the In-Focus Localization Network (ILN). As indicated, ILN comprises four 2D convolutional layers. The first two layers are dedicated to extracting contextual information from inputs that combine the rendered CoC map $\mathcal{M}_m^{(\text{coc})}$, defocus image $\tilde{\mathbf{I}}_m$, and depth map $\tilde{\mathbf{D}}_m$. Consequently, the input to the first layer has 5 channels, while the second layer receives 48 channels, matching the output feature map dimension from the first layer. The third convolutional layer, which also incorporates injected positional encoding information with 48 channels, thus receives a total of 64 channels. The final layer receives the feature from the third layer and outputs the mask \mathcal{M}_m^* . To ensure that the output falls within the range [0,1], we apply a Sigmoid() activation after the last layer.

Table 1. Detailed structure of the In-Focus Localization Network. "Conv2d" denotes 2D convolutional layer.

Idx	Layer	Kernel	Stride	Padding	Input ch.	Output ch.
1	Conv2d + ReLU()	3×3	1	1	5	48
2	Conv2d	3×3	1	1	48	16
3	Conv2d	1×1	1	1	64	20
4	Conv2d + Sigmoid()	1×1	1	1	20	1

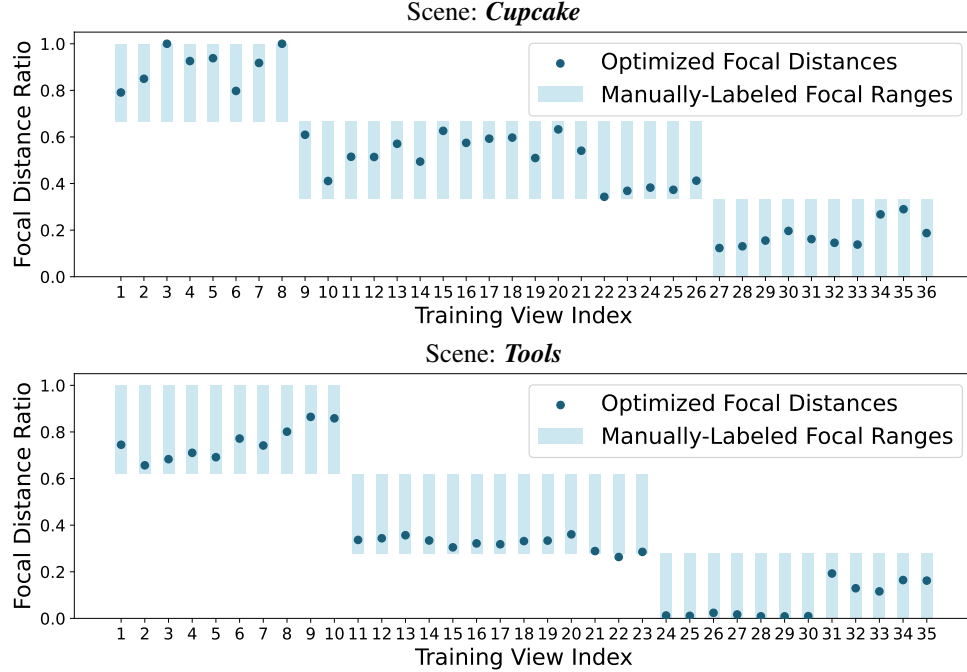


Figure 1. Visualization of focal distances post-optimization on two scenes from the real defocus dataset. The focal distances are transformed to diopters for visualization and normalized within the range $[0, 1]$ according to the diopter range of points within the point cloud. The focal distances optimized using our method for various views closely align with the ranges labeled manually.

3. Additional Experimental Results

In this section, we provide additional experimental results. Specifically, we provide more results for focal distance analysis in Section 3.1, showcase additional qualitative results for post-capture control of focal distance and aperture parameter in Section 3.2, investigate the impact of training with defocused images on rendered depth-of-field effects in Section 3.3, present quantitative results and more visual results for All-in-Focus novel view synthesis in Section 3.4, and include visual results from the variants explored in our ablation studies in Section 3.5.

3.1. Additional Results for Focal Distance Analysis

In addition to the focal distance analysis conducted on the *Cake* scene from the real defocus dataset, for which the results are provided in the main paper, we also examine the optimized focal distance parameters for training views of two additional scenes: *Cupcake* and *Tools*. The corresponding visualization results are presented in Figure 1. As shown in Figure 1, the focal distance parameters optimized on these two scenes closely match the manually annotated ranges of possible focal distances used during image capture. These results further demonstrate that the introduced finite-aperture camera model, combined with the proposed differentiable depth-of-field rendering process, is highly physically grounded and is able to effectively learn camera characteristics from defocused multi-view inputs.

3.2. Additional Results for Post-Capture Control

To demonstrate that our method enables depth-of-field rendering via post-capture control of aperture and focal distance parameters, we provide additional results on three scenes in Figure 2. As shown in Figure 2, the in-focus regions maintain sharp when the focal distance is set around their depth planes, while the out-of-focus regions demonstrate varying levels of blur with the change of aperture parameter. For instance, the fruit area in the first scene, the cookie pattern in the second scene, and the fine golden metallic structure in the third scene remain sharp at focal distances of 70, 109, and 48, respectively, even as the aperture parameter Q is adjusted from 101 to values exceeding 200 or even 300. Meanwhile, objects at other depth



Figure 2. Results for post-capture controllable depth-of-field rendering from our method. Adjustments to the focal distance primarily influence the positioning of in-focus and out-of-focus regions, with in-focus areas highlighted in pink. Meanwhile, increasing the aperture parameter increasingly enhances the blur effect in out-of-focus regions.

planes exhibit progressively pronounced blurriness, as seen in the enlarged out-of-focus regions presented in each column. Specifically, each column highlights the effects on a single region under varying aperture parameters, effectively illustrating the depth-of-field effect achieved by our approach. It can also be observed that the positions of the in-focus and out-of-focus regions shift as the focal distance is adjusted, which allows us to achieve refocusing effects. Furthermore, simultaneously altering the focal distance and aperture parameter produces compounded effects.

3.3. Influence of Training with Defocused Images

To validate the rationale behind training our method on multi-view defocused images for achieving post-capture control of depth-of-field rendering, we examine the rendered defocused images on scenes optimized using sharp All-in-Focus images. Specifically, the synthetic scenes from the dataset [4] provide ground-truth All-in-Focus images along with multi-view defocused images, enabling a comparative investigation. In our study, we train our approach separately on multi-view sharp images and on multi-view inconsistent defocused images. After optimization, we render defocused images from the scenes reconstructed under these two training settings. Exemplary results of the rendered defocused images from this investigation are presented in Figure 3.

As shown in Figure 3, depth-of-field (DOF) effects rendered from scenes reconstructed using sharp images during training exhibit undesired artifacts when camera parameters are adjusted in the DOF rendering process. Specifically, due to the absence of guidance from natural defocus blur in the training images, the 3D Gaussian points in the scene are primarily optimized to composite sharp All-in-Focus images. During the DOF rendering process, when these Gaussian points are convolved with depth-related blur kernels, they fail to generate natural and smooth defocus blur. Instead, the blurred Gaussian points cause objects in the out-of-focus regions to appear dilated and artificially "enhanced". For example, in the rear-focus rendered images, objects in closer regions, such as the stairs, the plant, and the 'Coca Cola' text, become increasingly dilated as the aperture parameter increases. Similarly, in the near-focus rendered images, the distant stone pillar does not exhibit any blur but instead becomes more dilated with increasing aperture, to the extent that it appears as a single, merged structure. These dilation effects result in a misleading "enhancement" of objects in the defocused regions, preventing the images from correctly conveying the intended depth and focus cues.

In contrast, the rear-focus and near-focus images rendered from scenes trained with defocused images exhibit natural and smooth defocus blur in closer and distant regions, respectively, effectively conveying correct focus cues. Additionally, the defocus blur adaptively changes as the aperture parameter is adjusted. This demonstrates the effectiveness of training with defocused images in achieving realistic depth-of-field rendering with our approach.

3.4. Additional Results for All-in-Focus Novel View Synthesis

In this section, we give comprehensive quantitative results for All-in-Focus novel view synthesis on both the real and synthetic datasets and present visual results.

Results on Real dataset. In Table 2, we present the numerical results of our approach on the task of All-in-Focus novel view synthesis, comparing them against vanilla NeRF [5], Mip-Splatting [8], and several state-of-the-art deblur-focused methods [2–4, 6, 7]. As shown in Table 2, the performance of Mip-Splatting [8] is slightly lower than that of vanilla NeRF [5]. This discrepancy arises because, although both methods produce reconstructed scenes with blurry regions caused by defocus blur, the multi-view inconsistencies introduced by defocus blur make the 3DGS-based Mip-Splatting method more prone to generating artifacts in novel views compared to the smoother MLP-based representations used in vanilla NeRF. This phenomenon is visually demonstrated in the first two rows of Figure 4.

NeRF-based deblur-focused methods, such as Deblur-NeRF [4], DP-NeRF [3], and PDRF [6], significantly improve performance compared to vanilla NeRF by differentially simulating defocus blur during the rendering process. Among these, the PDRF method proposed by Peng et al. [6] achieves the highest average PSNR score (24.31 dB) and SSIM value. Among the 3DGS-based methods, our approach achieves comparable performance to the state-of-the-arts BAGS and Deblur-GS. Specifically, BAGS achieves highest PSNR score, our method achieves highest SSIM score. The visual results in Figure 4 further demonstrate that our method is able to achieve sharp All-in-Focus renderings, suggesting that our approach effectively reconstruct the fine details from multi-view inputs with defocus blur.

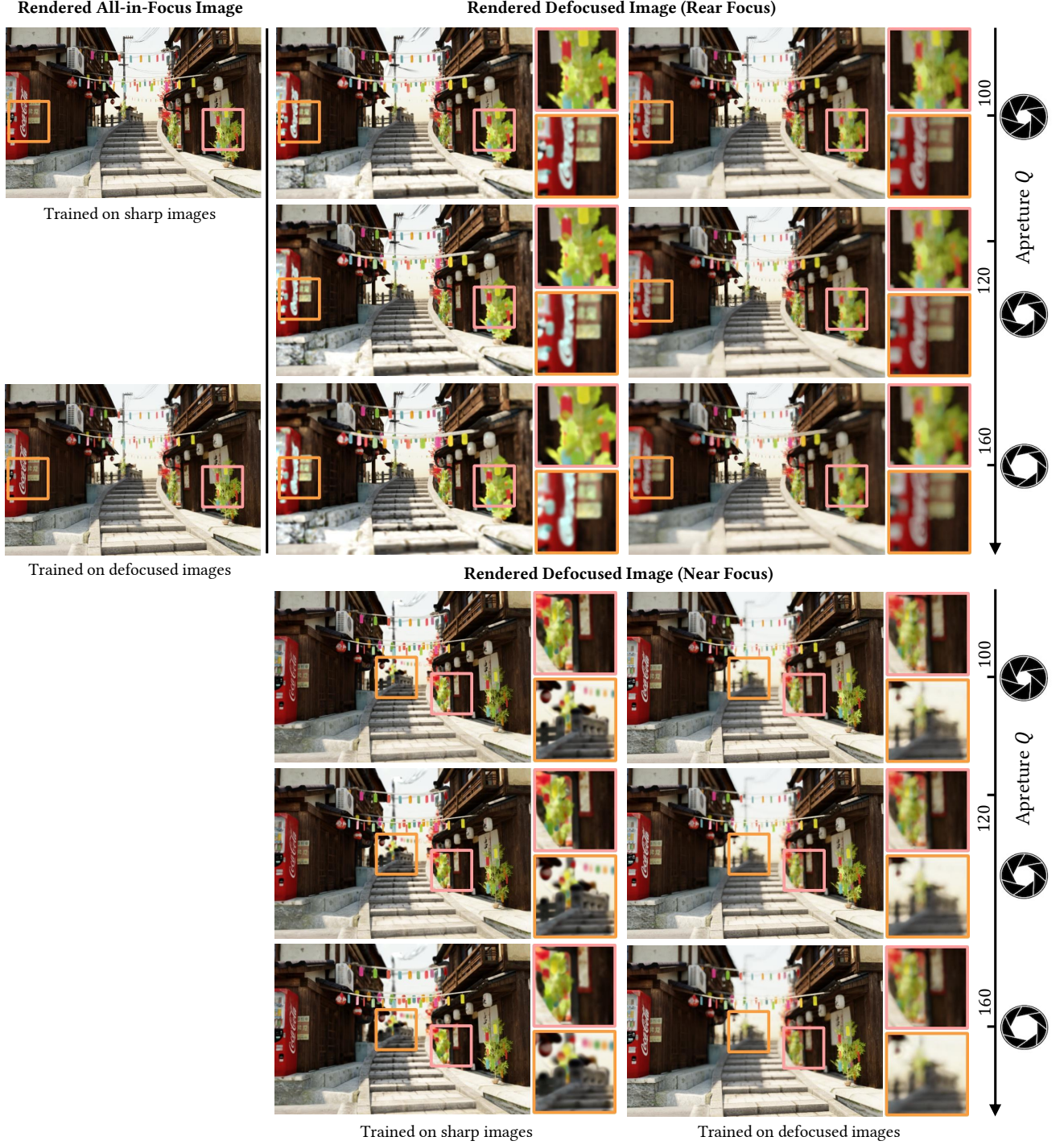


Figure 3. Rendered depth-of-field effects post-optimization from our approach under two training settings. The pink and orange blocks highlight the out-of-focus regions in the near-focus and rear-focus images. When adjusting camera parameters in our depth-of-field rendering module on scenes optimized on sharp All-in-Focus images, out-of-focus regions fail to exhibit natural smooth defocus blur. Instead, the Gaussian points after convolution operations cause objects in the out-of-focus regions to appear dilated and seemingly "enhanced," which undermines the ability of such defocus effects to effectively convey focus cues. In contrast, training on multi-view defocused images allows the optimized Gaussian points to form smooth, depth-related defocus blur via adjusting the camera parameters.

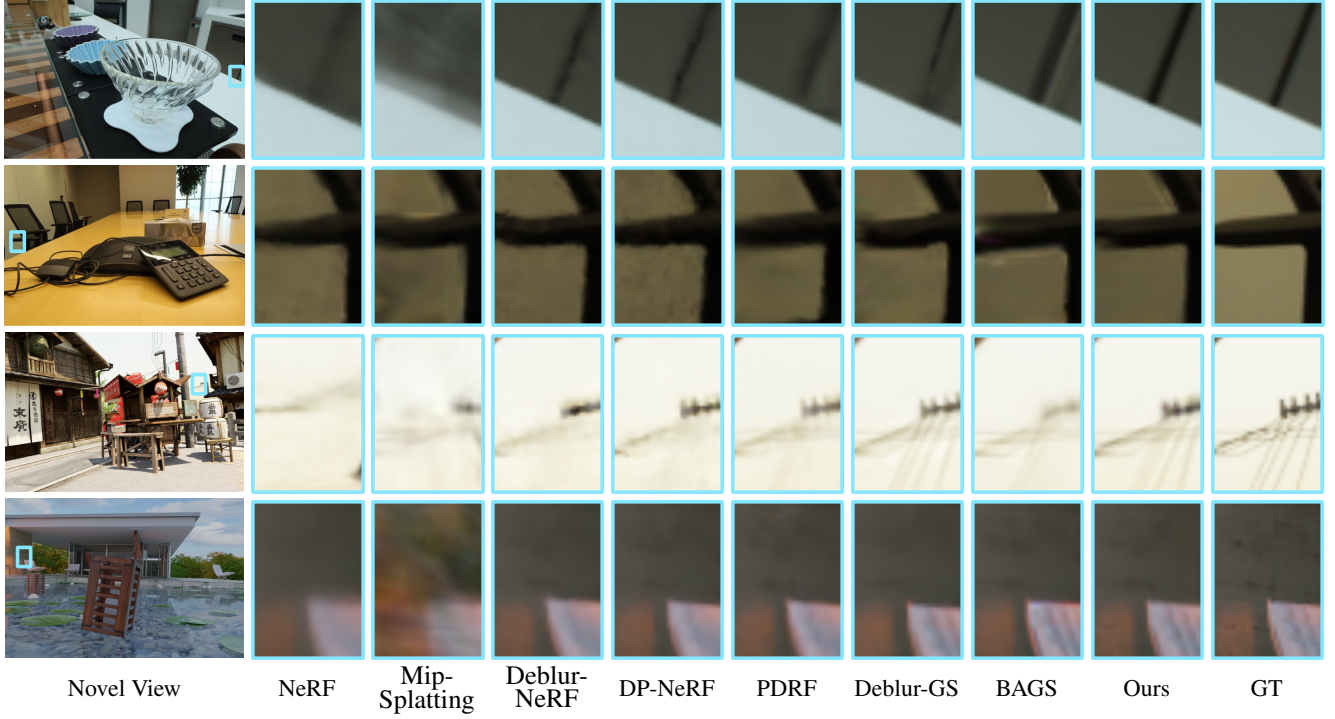


Figure 4. Visual results for All-in-Focus novel view synthesis on real and synthetic defocus dataset. Enlarged regions are provided for better visualization. The first and second rows show results on two scenes from the real defocus dataset, while the third and fourth rows show results from two scenes from the synthetic defocus dataset.

Results on Synthetic dataset. In Table 3, we present the numerical results for All-in-Focus novel view synthesis on the synthetic dataset. The performance trends are similar to those observed in the real dataset. Notably, Mip-Splatting produces a much lower average SSIM score than vanilla NeRF as it produces severe artifacts in rendered images for the *Pool* scene, which is demonstrated in Figure 4. Regarding the BAGS method, for which the authors provide dedicated settings for each scene within the real dataset but without providing configurations for the synthetic scenes, we apply it to the synthetic dataset using a uniform configuration. Following from their configurations on scenes from the real data, we set a relatively large number of iterations (45,000) with 9,000 dedicated to coarse-to-fine optimization.

The PSNR and SSIM score of our method surpass those of Deblur-NeRF and DP-NeRF and are on par with the other two 3DGS-based methods, BAGS and Deblur-GS. As shown in Table 3, the PSNR and SSIM scores for the 3DGS-based methods are slightly lower than those of the NeRF-based method PDRF [6]. This is likely caused by the differences in camera pose: the 3DGS-based methods, including our approach, BAGS, and Deblur-GS, rely on camera poses estimated using structure-from-motion (SfM) methods from multi-view inputs containing defocus blur. In contrast, NeRF-based methods, such as PDRF, utilize ground-truth camera poses provided by Blender. The visual results on synthetic scenes presented in Figure 4 exhibit similar trends to those observed in the real dataset scenes. Our method demonstrates enhanced detail recovery compared to the NeRF-based methods and achieves comparable detail restoration to BAGS and Deblur-GS.

Table 2. Numerical results on real defocus dataset.

	Cake		Caps		Cisco		Cupcake		Cups		Daisy		Sausage		Seal		Tools		Average	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
NeRF	24.60	.7296	22.68	.6316	20.97	.7267	21.85	.6776	24.96	.7556	22.85	.6237	17.79	0.4874	22.88	.6266	25.75	.8510	22.70	.6789
Mip-Splatting	21.75	.6268	21.62	0.5355	19.85	.6902	21.26	.6719	21.09	.6403	21.63	.6181	17.83	.4719	22.27	.6006	23.80	.8069	21.23	.6292
Deblur-NeRF	26.17	.7777	24.03	.7158	20.74	.7241	22.58	.7352	26.44	.8081	23.94	.7027	18.23	.5195	25.98	.7768	27.67	.8893	23.98	.7388
DP-NeRF	26.16	.7782	23.90	.7123	20.77	.7279	23.03	.7467	26.92	.8194	23.76	.6958	18.48	.5462	26.19	.7857	27.90	.8937	24.12	.7451
PDRF	27.00	.7961	24.15	.7157	20.72	.7279	22.98	.7459	26.22	.8071	24.37	.7419	18.91	.5662	26.44	.8025	28.01	.9001	24.31	.7559
BAGS	26.86	.8093	24.35	.7326	20.68	.7362	23.08	.7648	25.82	.8184	23.76	.7475	18.83	.5707	26.51	.8142	27.60	.9043	24.17	.7664
Deblur-GS	27.08	.8076	24.68	.7437	21.06	.7409	22.65	.7477	26.26	.8240	23.70	.7268	18.74	.5539	26.25	.8197	27.58	.9037	24.21	.7631
Ours	26.87	.8066	24.63	.7485	20.84	.7399	22.70	.7542	25.98	.8261	23.48	.7343	19.11	.5737	25.75	.8135	27.73	.9033	24.12	.7667

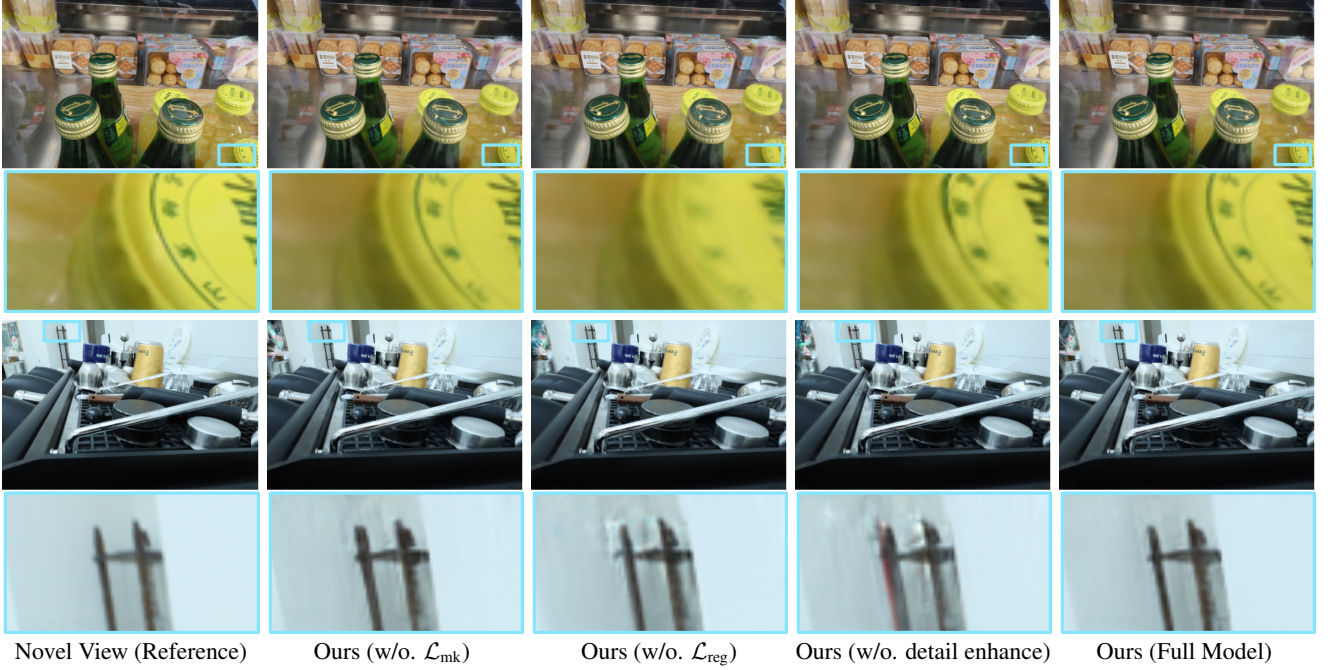


Figure 5. Visual results for ablation study. The variant ‘Ours (w/o. detail enhance)’ is prone to producing apparent artifacts at the closest or farthest regions, *i.e.*, regions that are easier to be absent from some input views. The phenomena stem from its reduced utilization of information from in-focus regions across different training views. All of the images are rendered under the All-in-Focus setting.

Table 3. Quantitative results on the synthetic defocus dataset.

	Cozyroom		Factory		Pool		Tanabata		Trolley		Average	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
NeRF	30.12	.8939	24.40	.7680	27.87	.7294	23.79	.7790	22.54	.7141	25.74	.7769
Mip-Splatting	29.83	.8990	23.04	.7866	19.78	.4212	23.64	.8109	22.37	.7393	23.73	.7314
Deblur-NeRF	31.89	.9172	26.36	.8455	30.07	.8159	26.19	.8515	25.57	.8151	28.02	.8491
DP-NeRF	32.23	.9248	27.14	.8608	31.48	.8532	27.05	.8668	26.67	.8377	28.91	.8687
PDRF	31.84	.9249	30.89	.9143	30.49	.8333	28.36	.9051	28.23	.8839	29.96	.8923
BAGS	32.07	.9346	30.81	.9291	28.72	.8232	29.46	.9328	23.38	.8096	28.89	.8859
Deblur-GS	32.03	.9269	29.89	.9096	30.50	.8344	27.56	.9071	27.18	.8755	29.43	.8907
Ours	32.25	.9345	29.56	.9057	30.60	.8348	27.46	.9080	27.11	.8741	29.39	.8914

3.5. Visual Results for Ablation Study

In Figure 5, we present the visual results from the variants examined during the ablation studies. As shown in Figure 5, the absence of the detail enhancement strategy leads to results that are less sharp (e.g., the circular line and text appear blurrier compared to the outputs of our full model) or introduce artifacts, such as those visible around the ladder region. These observations highlight the effectiveness of the detail enhancement strategy in our pipeline, which is achieved by supervising rendered All-in-Focus images using Circle-of-Confusion (CoC) cues. Additionally, removing the regularization term for the predicted in-focus mask leads to reconstructed scenes with more blurry regions and artifacts. Meanwhile, it can be observed that removing the regularizer for the predicted in-focus mask, which encourages the in-focus to be binary, makes the reconstructed scenes exhibit more blurry regions and artifacts. Figure 5 further demonstrates that, incorporating the loss term \mathcal{L}_{mk} that utilizes rendered CoC maps to supervise the predicted in-focus masks, also contributes to reducing artifacts, as shown in ladder region.

References

- [1] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (July 2023).

- [2] Byeonghyeon Lee, Howoong Lee, Xiangyu Sun, Usman Ali, and Eunbyung Park. 2024. Deblurring 3D Gaussian Splatting. *arXiv preprint arXiv:2401.00834* (2024).
- [3] Dogyoon Lee, Minhyeok Lee, Chajin Shin, and Sangyoun Lee. 2023. Dp-nerf: Deblurred neural radiance field with physical scene priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12386–12396.
- [4] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V Sander. 2022. Deblur-nerf: Neural radiance fields from blurry images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 12861–12870.
- [5] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *European Conference on Computer Vision*. 405–421.
- [6] Cheng Peng and Rama Chellappa. 2023. PDRF: Progressively Deblurring Radiance Field for Fast and Robust Scene Reconstruction from Blurry Images. In *The 37th AAAI Conference on Artificial Intelligence*.
- [7] Cheng Peng, Yutao Tang, Yifan Zhou, Nengyu Wang, Xijun Liu, Deming Li, and Rama Chellappa. 2024. BAGS: Blur Agnostic Gaussian Splatting through Multi-Scale Kernel Modeling. *arXiv preprint arXiv:2403.04926* (2024).
- [8] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. 2024. Mip-Splatting: Alias-free 3D Gaussian Splatting. *Conference on Computer Vision and Pattern Recognition (CVPR)* (2024).
- [9] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. 2001. EWA volume splatting. In *Proceedings Visualization, 2001. VIS'01*. IEEE, 29–538.