Local Implicit Ray Function for Generalizable Radiance Field Representation (Supplementary Material)

Xin Huang[1], Qi Zhang[2], Ying Feng[2], Xiaoyu Li[2], Xuan Wang[2], Qing Wang[1]

[1] School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China
[2] Tencent AI Lab

1. Additional Implementation Details

1.1. Network Details

Our feature extraction network (EDSR [4]) is based on the implementation from this URL (https://github.com/sanghyun-son/EDSR-PyTorch). The tail module of the EDSR network is removed, and two convolutional layers are added at the tail. The one outputs the feature maps for visibility weights, while the other one outputs the features maps for colors and densities. The implementation AE network is from GeoNeRF [3]. The ray function $\mathcal{R}$ is a three-layer MLP with 32 channels for each linear layer. Both networks $\mathcal{M}_w$ and $\mathcal{M}_\sigma$ are a two-layer MLP with 32 channels. The network $\mathcal{M}_c$ is a three-layer MLP with 32 channels. The transformer module $\mathcal{T}_1$ contains a single multi-head self-attention layer with the number of heads set to 4, while $\mathcal{T}_2$ contains four multi-head self-attention layers with the number of heads set to 4. The "MLP" used to reduce feature channels is a two-layer MLP and the number of channels is set to 32. For all MLP-based networks, ELU is used between each of two adjacent linear layers as the non-linear activation function. In our experiments, all networks are trained from scratch. Our code and model will be made available.

1.2. Dataset Details

We train our model on three real datasets: the real DTU multi-view dataset [2] and two real forward-facing datasets from LLFF [6] and IBRNet [8]. All 190 scenes (35 scenes from LLFF dataset, 67 scenes from IBRNet dataset and 88 scenes from DTU dataset) are used for training. We exclude the views with incorrect exposure from the DTU dataset as done in pixelNeRF [9]. Eight unseen scenes from LLFF dataset are used as our testing scenes. During the multi-scale training, the resolutions of all input views are consistent (252 × 189 for LLFF and IBRNet datasets, 200 × 150 for DTU dataset), while the resolution of each target view is randomly selected from 1 to 4 times the input resolution (from 252 × 189 to 1008 × 756 for LLFF and IBRNet datasets, from 200 × 150 to 800 × 600 for DTU dataset). When training our model on single-scale datasets, the image resolutions of input and target images are the same (504 × 378). During testing, the resolution of input views is 504 × 378. We evaluate our model on rendering novel views at multiple scales: ×0.5, ×1, ×2 and ×4 (×0.5 denotes 0.5 times the resolution of input views, and so on). During the dataset preprocessing, we use bicubic interpolation to downsample high resolution images.

Table S1. Fine-tuning results of our method and state-of-the-art methods. We fine-tune our pretrained model on each scene for 10k iterations with resolution of 1008 × 756. The resolution of testing views is also set to 1008 × 756.

	PSNR↑	SSIM↑	LPIPS↓
NeRF [7]	26.50	0.811	0.250
IBRNet [8]	26.73	0.851	0.175
NeuRay [5]	27.06	0.850	0.172
GeoNeRF [3]	26.58	0.856	0.162
Ours(10k)	26.85	0.865	0.159

2. Additional Experiments

2.1. Fine-tuning

Although our approach focuses on generalizations to unseen scenes, we also fine-turn our pre-trained model on each testing scene for comparison against previous methods. We follow the setting of IBRNet [8] and train our model on each of the eight testing scenes for 10k iterations. The resolution of images used for training and testing is set to 1008 × 756. Note that the multi-view images used for fine-tuning are single-scale. The results are reported in Tab. S1.

2.2. Comparisons with Two-stage Methods

To further evaluate our method on rendering novel views at high scales (×2 and ×4), we try to compare our method with two-stage methods. We first render novel views at ×1 scale via three baselines and then upsample the novel views via bicubic interpolation and a single-image super resolution method, LIIF [1]. The results are presented in Tab. S3.

Table S2. Quantitative comparisons of varying the number of source views on LLFF real forward-facing scenes.

	PSNR↑				SSIM↑				LPIPS↓				Avg.↓
	×0.5	×1	×2	×4	×0.5	×1	×2	×4	×0.5	×1	×2	×4	
4 views	24.80	24.03	22.93	22.31	0.864	0.825	0.761	0.711	0.134	0.168	0.269	0.403	0.080
6 views	26.11	25.51	24.24	23.50	0.893	0.866	0.804	0.750	0.106	0.131	0.233	0.377	0.066
8 views	26.75	25.93	24.58	23.79	0.905	0.877	0.816	0.760	0.100	0.124	0.227	0.373	0.063
10 views	26.46	25.91	24.56	23.78	0.900	0.877	0.815	0.760	0.101	0.126	0.231	0.376	0.064

Table S3. Quantitative comparisons of our LIRF against two-staget methods on rendering novel view at higher scales (×2 and ×4). We upsample low resolution novel views via bicubic interpolation (BI) or LIIF [1]. "M" denotes the number of vertices used to represent a conical frustum.

	PSNR↑		SSIM↑		LPIPS↓		Avg. ↓
	×2	×4	×2	×4	×2	×4	
IBRNet-BI	23.50	22.85	0.740	0.691	0.307	0.438	0.099
IBRNet-LIIF	23.80	23.11	0.760	0.712	0.278	0.421	0.093
NeuRay-BI	23.44	22.79	0.738	0.689	0.305	0.437	0.099
NeuRay-LIIF	23.70	23.02	0.757	0.709	0.276	0.419	0.094
GeoNeRF-BI	23.89	23.19	0.765	0.708	0.282	0.420	0.093
GeoNeRF-LIIF	24.26	23.53	0.788	0.733	0.251	0.400	0.087
Ours (M=4)	23.91	23.15	0.789	0.741	0.248	0.398	0.089
Ours (M=8)	24.58	23.79	0.816	0.760	0.227	0.373	0.081
Ours (M=10)	24.93	23.95	0.838	0.784	0.218	0.366	0.077

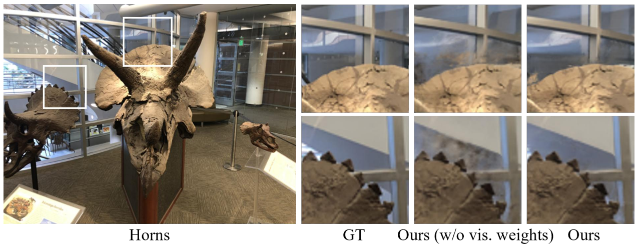

Figure S1. The qualitative results of our model without visibility weights. Ours denotes our full model.

It shows the superiority of our model on rendering novel views at high scales with respect to the two-stage methods, though they introduce external data priors.

2.3. Number of Source Views

To investigate the robustness of our model to the number of source views, our model is tested on unseen scenes with different numbers of source views (4, 6, 8, and 10). The quantitative results are shown in Tab. S2. The results show that our model produces competitive results when the number of source views is set to 6, 8, and 10. The model produces the best results when setting the number of source views to 8, since our model is trained with 8 source views. However, the performance of our method reduces a lot when the source views are sparse (4 views), since it is challenging to estimate visibility weights by matching sparse local image features.

2.4. Number of Vertices

We represent a conical frustum using several vertices. The results with different numbers of vertices are shown in Tab. S3. Using more vertices, our performance improves, but the rendering time increases too. Considering computing burdens and inspired by the voxel-based volume rendering, we use M = 8 vertices to approximate a conical frustum. The samples within the conical frustum can be calculated by our implicit ray function.

2.5. Comparisons of Rendering Time

LIRF (45s for rendering an image with ×1 scale) is about three times slower than IBRnet (15s for rendering an image with ×1 scale). However, once the conical frustums are constructed, we directly infer rays from the conical frustums to render multi-scale views. Compared with baselines on rendering multi-scale views, we save the time of querying features from feature maps, especially on rendering high resolution views.

3. Additional Results

3.1. Qualitative Results for Ablation Studies

As shown in Tab. S4, three ablations (Ours(single ray), Ours w/o vis. weights and Ours(U-Net feat.)) mainly affect the performance of our LIRF. To further investigate their contributions to our model, the qualitative results are shown in Figs. S1, S3 and S4.

Ours w/o vis. weights. We remove the visibility weights estimation module to evaluate the impact of the visibility weights. Figure S1 shows the performance of our model without visibility weights. Our method produces renderings with ghosting artifacts on the boundary of objects due to occlusions.

Ours (single ray). To investigate the contribution of our local implicit ray function, we render a pixel from a single ray instead of conical frustums. The results are presented in Fig. S3. One can see that our model (single ray) produces renderings that are excessively aliased when rendering novel views at ×0.5 scale. Besides, our model (single ray) produces renderings containing artifacts at thin structures when rendering novel views at ×2 scale.

Ours (U-Net feat.). Moreover, the feature extraction network is also important to our method, especially on rendering novel views at high scales. We therefore extract 2D image features via the U-Net in IBRNet [8]. Our model with the U-Net is trained from scratch on our multi-scale dataset. The rendered testing views are presented in Fig. S4. Our model produces renderings with more blurred artifacts when the image features are extracted by the U-Net.

3.2. A Failure Case

As discussed in the limitations, though the visibility weights can mitigate the artifacts caused by occlusions, they fail in some challenging scenes such as the *orchids* scene. Figure S2 shows a failure example on the *orchids* scene. The multi-view images of this scene are captured sparsely, which is challenging for our model to estimate the accurate visibility weights. The baselines also struggle with this challenging scene, such as the renderings by IBRNet [8] with blurred artifacts. After fine-tuning on this scene for 10k iterations, our model produces results with fewer artifacts on the boundary of objects.

3.3. Per-Scene Results

To evaluate our approach compared to previous methods on each individual scene, per-scene results on the eight testing scenes are presented in Tab. S4. We report the arithmetic mean of each metric averaged over the four testing scales used for testing. Our method yields a significant improvement in three error metrics across most scenes.

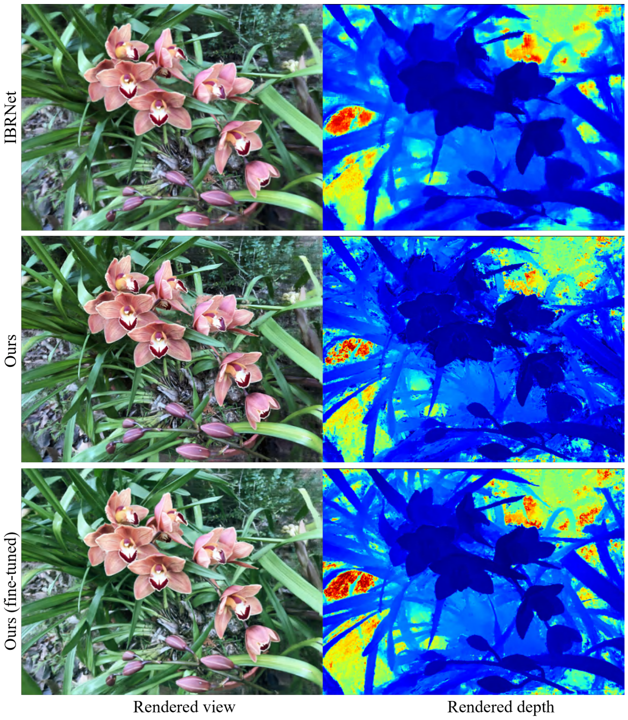

Figure S2. A failure example on *orchids* scene. Our model fails to predict the geometry on the boundaries of flowers due to occlusions. The renderings by IBRNet [8] also contain blurred artifacts. After fine-tuning on this scene for 10k iterations, our model produces results with fewer artifacts

References

[1] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *CVPR*, pages 8628–8638, 2021. 1, 2

[2] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *CVPR*, pages 406–413, 2014. 1

[3] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. GeoNeRF: Generalizing nerf with geometry priors. In *CVPR*, pages 18365–18375, 2022. 1, 5

[4] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPRW*, pages 136–144, 2017. 1

[5] Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Christian Theobalt, Xiaowei Zhou, and Wenping Wang. Neural rays for occlusion-aware image-based rendering. In *CVPR*, pages 7824–7833, 2022. 1, 5

[6] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM TOG*, 38(4):1–14, 2019. 1, 5

[7] 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 *ECCV*, pages 405–421, 2020. 1

[8] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. IBRNet: Learning multi-view image-based rendering. In *CVPR*, pages 4690–4699, 2021. 1, 3, 4, 5

[9] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, pages 4578–4587, 2021. 1

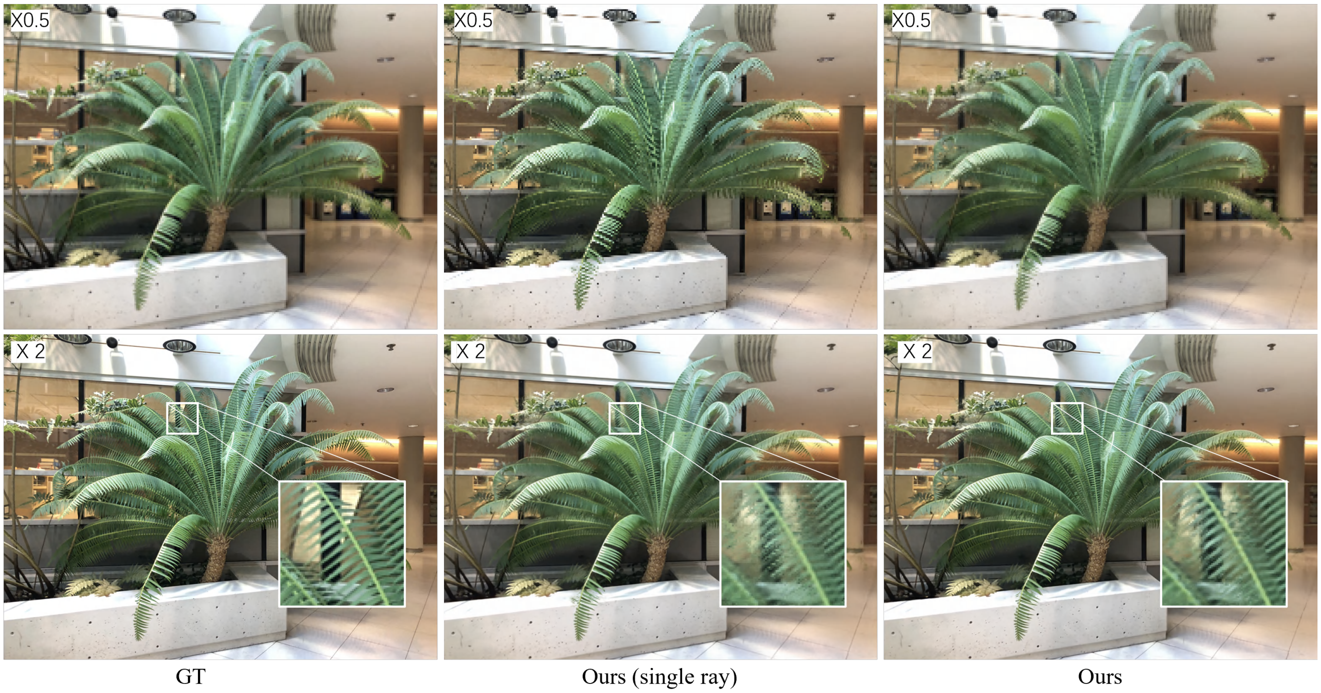

Figure S3. The qualitative results of our model that renders a pixel from a single ray. The top row shows the novel views rendered at ×0.5 scale. Our model (single ray) produces aliased novel view. The bottom row shows the novel views rendered at ×2 scale. Our model (single ray) produces novel view with artifacts at thin structures.

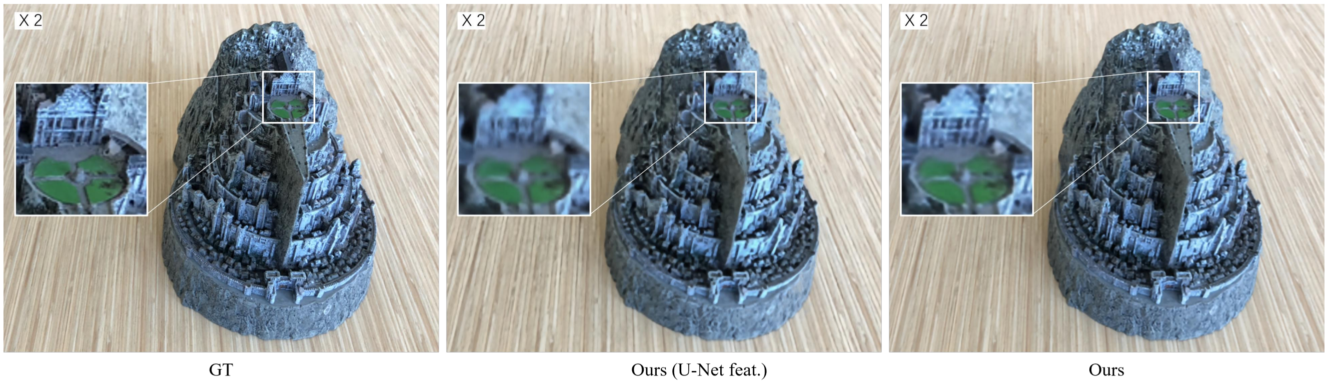

Figure S4. The qualitative results of our model that extracts image features via the U-Net in IBRNet [8]. Our model (U-Net feat.) produces novel views with more blurred artifacts when the image features are extracted by the U-Net.

Table S4. Per scene quantitative comparisons of our LIRF and its ablations against IBRNet [8], NeuRay [5] and GeoNeRF [3] on LLFF [6] multi-scale testing dataset. Metrics are averaged over four testing scales ($\times 0.5$, $\times 1$, $\times 2$ and $\times 4$). $*$ denotes training on the same multi-scale training set as our method.

	Average PSNR↑							
	fern	*flower*	*fortress*	*horns*	*leaves*	*orchids*	*room*	*trex*
IBRNet	23.40	25.80	28.12	24.78	19.69	19.20	27.49	22.89
NeuRay	23.21	25.89	28.18	24.78	19.48	18.87	27.09	22.81
GeoNeRF	23.73	26.35	28.88	25.19	19.75	19.81	27.02	21.91
IBRNet*	22.38	24.61	26.32	23.68	18.48	18.07	25.49	21.80
NeuRay*	21.26	23.56	25.43	22.45	17.89	17.50	25.12	20.84
GeoNeRF*	23.55	26.21	28.17	24.92	19.89	19.50	26.24	21.47
Ours	25.21	26.77	29.07	26.59	21.59	19.39	28.59	24.91
Ours w/o scale	25.05	26.63	29.25	26.56	21.33	19.28	28.75	25.06
Ours w/o patch	25.18	26.64	29.17	26.47	21.47	19.44	28.72	25.16
Ours w/o position	24.78	26.69	28.18	26.16	20.97	19.33	28.42	24.64
Ours w/o direction	24.60	26.31	28.34	25.65	20.86	19.22	28.66	24.73
Ours w/o vis. weights	24.72	25.95	27.92	25.50	20.66	18.98	27.75	24.75
Ours (U-Net feat.)	24.21	26.03	28.67	25.33	20.44	19.14	27.12	23.67
Ours (single ray)	24.49	26.60	28.11	25.78	20.83	19.28	28.07	24.41

	Average SSIM↑							
	fern	*flower*	*fortress*	*horns*	*leaves*	*orchids*	*room*	*trex*
IBRNet	0.741	0.836	0.832	0.805	0.678	0.629	0.899	0.794
NeuRay	0.739	0.836	0.833	0.808	0.668	0.617	0.896	0.790
GeoNeRF	0.768	0.847	0.844	0.825	0.683	0.659	0.897	0.795
IBRNet*	0.717	0.820	0.801	0.790	0.650	0.593	0.877	0.786
NeuRay*	0.675	0.761	0.723	0.733	0.568	0.531	0.862	0.735
GeoNeRF*	0.774	0.852	0.833	0.829	0.704	0.655	0.893	0.802
Ours	0.825	0.870	0.897	0.876	0.787	0.666	0.924	0.872
Ours w/o scale	0.817	0.865	0.896	0.870	0.776	0.656	0.921	0.866
Ours w/o patch	0.821	0.867	0.898	0.875	0.782	0.668	0.923	0.870
Ours w/o position	0.806	0.858	0.882	0.863	0.761	0.652	0.913	0.856
Ours w/o direction	0.806	0.861	0.891	0.864	0.756	0.651	0.920	0.864
Ours w/o vis. weights	0.807	0.849	0.886	0.853	0.748	0.635	0.910	0.860
Ours (U-Net feat.)	0.777	0.849	0.858	0.831	0.728	0.642	0.898	0.829
Ours (single ray)	0.799	0.856	0.873	0.849	0.757	0.651	0.904	0.851

	Average LPIPS↓							
	fern	*flower*	*fortress*	*horns*	*leaves*	*orchids*	*room*	*trex*
IBRNet	0.282	0.201	0.195	0.252	0.285	0.316	0.214	0.272
NeuRay	0.282	0.191	0.189	0.246	0.293	0.311	0.206	0.265
GeoNeRF	0.251	0.187	0.170	0.226	0.283	0.287	0.207	0.267
IBRNet*	0.297	0.208	0.221	0.263	0.297	0.339	0.235	0.279
NeuRay*	0.359	0.269	0.296	0.336	0.369	0.395	0.262	0.331
GeoNeRF*	0.245	0.176	0.181	0.224	0.264	0.288	0.212	0.265
Ours	0.217	0.174	0.152	0.191	0.219	0.288	0.190	0.219
Ours w/o scale	0.223	0.177	0.149	0.193	0.226	0.296	0.188	0.220
Ours w/o patch	0.221	0.175	0.149	0.187	0.222	0.289	0.185	0.216
Ours w/o position	0.234	0.181	0.165	0.199	0.254	0.311	0.188	0.223
Ours w/o direction	0.231	0.182	0.153	0.196	0.233	0.303	0.187	0.223
Ours w/o vis. weights	0.233	0.193	0.162	0.209	0.247	0.320	0.199	0.225
Ours (U-Net feat.)	0.266	0.199	0.200	0.246	0.269	0.313	0.227	0.263
Ours (single ray)	0.240	0.188	0.181	0.214	0.257	0.318	0.200	0.229