Table 1: Additional ablation studies on Matterport3D

Dataset	Matterport3D (1.0m)			
method	PSNR↑	WS-PSNR↑	SSIM↑	LPIPS↓
w/o appearance	5.44	5.24	0.001	0.707
w/o geometry	26.25	25.25	0.839	0.263
full	**28.10**	**27.10**	**0.876**	**0.195**

A More Results of Qualitative Comparisons

Qualitative comparisons with baseline methods on Replica and Matterport3D can be seen in Fig. 1 and Fig. 2, respectively.

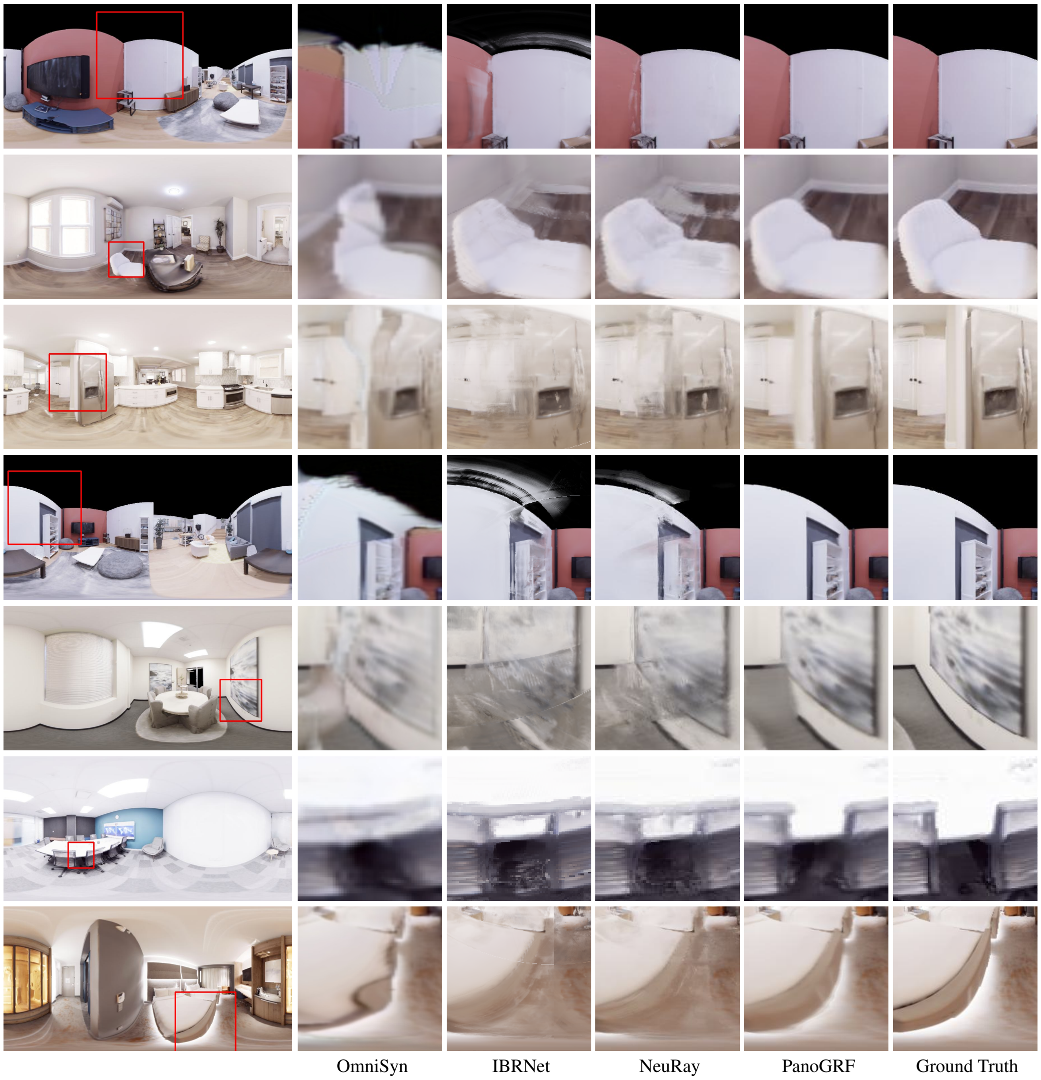


Figure 1: Qualitative comparisons with baseline methods on Replica.

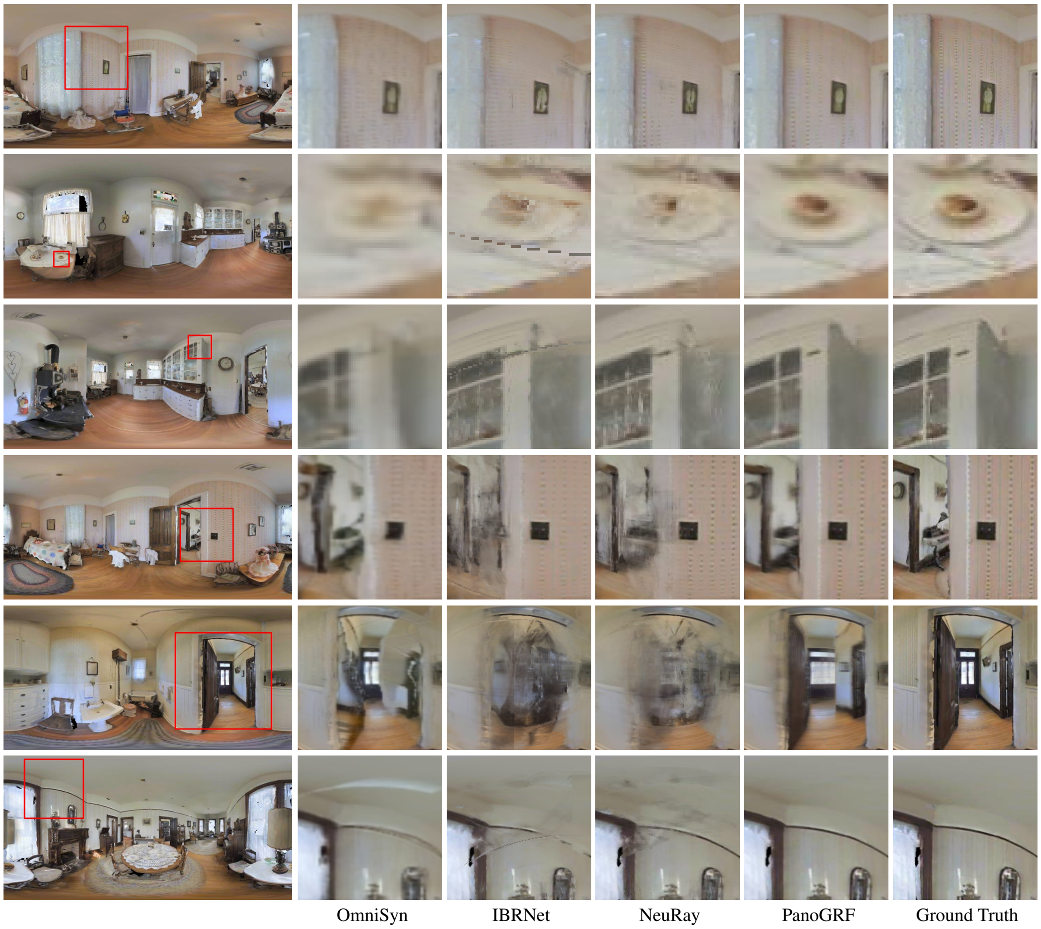


Figure 2: Qualitative comparisons with baseline methods on Matterport3D.

Table 2: Ablation studies for depth estimation on Matterport3D

setting	$L_1 \downarrow$	$L_2 \downarrow$	RMSE↓	WS-L_1↓	WS-L_2↓	WS-RMSE↓
MVS only	0.1731	0.5048	0.5831	0.1984	0.2806	0.4731
Mono only	0.2452	0.3175	0.4731	0.2445	0.2729	0.4522
full	**0.1441**	**0.2047**	**0.3877**	**0.1502**	**0.1624**	**0.3546**

Table 3: The impact of different backbones for depth estimation on Matterport3D

setting(backbone)	$L_1 \downarrow$	$L_2 \downarrow$	RMSE↓	WS-L_1↓	WS-L_2↓	WS-RMSE↓	parameters(M)
MVS(resnet-18)	0.1731	0.5048	0.5831	0.1984	0.2806	0.4731	29.75
MVS(resnet-34)	0.1654	0.4820	0.5676	0.1844	0.2577	0.4493	39.39
MVS(resnet-50)	0.1598	0.4725	0.5630	0.1822	0.2446	0.4442	47.10
MVS(resnet-101)	0.1642	0.4994	0.5717	0.1835	0.2519	0.4441	65.22
MVS(resnet-18)+Mono	**0.1441**	**0.2047**	**0.3877**	**0.1502**	**0.1624**	**0.3546**	58.95
MVS(resnet-34)+Mono	0.1549	0.2231	0.4120	0.1684	0.1878	0.3844	68.59
MVS(resnet-50)+Mono	0.1519	0.2379	0.4188	0.1675	0.1955	0.3887	76.30
MVS(resnet-101)+Mono	0.1735	0.2673	0.4452	0.1831	0.2129	0.4023	94.42

Table 4: The impact of different numbers of depth candidates N_{mono} for depth estimation on Matterport3D

N_{mono}	$L_1 \downarrow$	$L_2 \downarrow$	RMSE↓	WS-L_1↓	WS-L_2↓	WS-RMSE↓
1	0.1586	0.2498	0.4317	0.1745	0.1971	0.3937
3	**0.1432**	**0.1993**	**0.3865**	0.1529	0.1649	0.3580
5	0.1441	0.2047	0.3877	**0.1502**	**0.1624**	**0.3546**
7	0.1496	0.2252	0.4104	0.1640	0.1896	0.3832
9	0.1645	0.2912	0.4511	0.1752	0.2215	0.4059
16	0.1596	0.2379	0.4188	0.1675	0.1955	0.3887
32	0.1735	0.2673	0.4452	0.1831	0.2129	0.4023
48	0.1689	0.2618	0.4333	0.1776	0.2047	0.3941
64	0.1604	0.2432	0.4162	0.1669	0.2004	0.3853

Table 5: The impact of different values of σ for depth estimation on Matterport3D

σ	$L_1 \downarrow$	$L_2 \downarrow$	RMSE↓	WS-L_1↓	WS-L_2↓	WS-RMSE↓
0.1	0.1544	0.2515	0.4261	0.1672	0.1915	0.3830
0.5	0.1441	**0.2047**	**0.3877**	**0.1502**	**0.1624**	**0.3546**
1.0	0.1689	0.2686	0.4424	0.1803	0.2124	0.4029
1.5	**0.1426**	0.2457	0.4254	0.1563	0.1759	0.3723

B Spherical Projection

Equirectangular-to-spherical The transformation from the equirectangular image coordinate system to the polar coordinate system is defined as:

$$\begin{aligned} \phi &= v/H * \pi \\ \theta &= u/W * 2\pi - 0.5\pi, \end{aligned} \tag{1}$$

where ϕ, θ represent the latitude and longitude of the sphere, u, v represent the rows and columns of the panorama, H and W represent the height and width of the panorama respectively.

Spherical-to-cartesian The transformation from the polar coordinate system to the 3D Cartesian coordinate system is:

$$\begin{aligned} x &= sin(\phi) * cos(\theta) \\ y &= cos(\phi) \\ z &= sin(\phi) * sin(\theta). \end{aligned} \tag{2}$$

Cartesian-to-spherical The camera cartesian coordinate (x, y, z) is transformed into the polar coordinate (θ, ϕ, t) ($t \in \mathbb{R}^+$ denotes its spherical depth in view I_j) by:

$$\begin{aligned} t &= \sqrt{x^2 + y^2 + z^2} \\ \theta &= arctan(\frac{z}{x}) \\ \phi &= arccos(\frac{y}{t}). \end{aligned} \tag{3}$$

Table 6: Quantitative comparison with Cross Attention Renderer [1] on Matterport3D and Replica

Dataset	Matterport3D (1.0m)			Replica(1.0m)		
method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
CAR [1]	22.87	0.7679	0.3108	23.60	0.8594	0.2515
PanoGRF	**27.78**	**0.8158**	**0.2444**	**29.87**	**0.9046**	**0.1604**

Table 7: Comparisons with NeuRay [7] given multi-view inputs on Matterport3D

Dataset	Matterport3D (1.0m)			
method	PSNR↑	WS-PSNR↑	SSIM↑	LPIPS↓
NeuRay [7]	27.82	26.74	0.8614	0.2312
PanoGRF	**28.99**	**27.91**	**0.8762**	**0.2071**

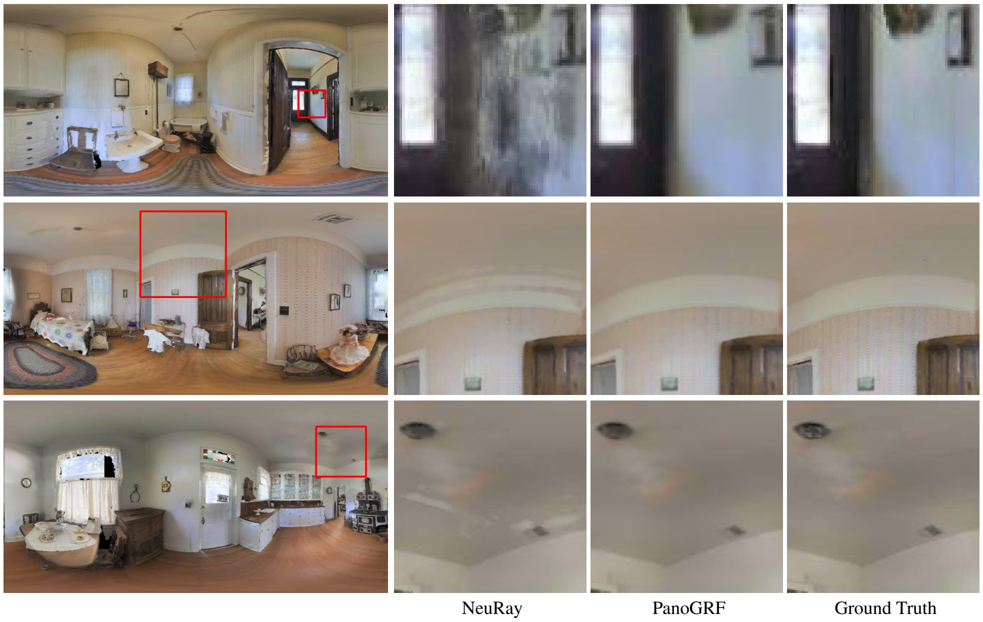


Figure 3: Qualitative comparisons between PanoGRF and NeuRay on Matterport3D with multi-view panoramic inputs.

Spherical-to-equirectangular The spherical polar coordinate (θ, ϕ, t) is turned into the equirectangular image coordinate (u, v) by the inverse process of Eq. 1.

C More Ablation Studies for $360°$ View Synthesis

We conducted ablation studies on Matterport3D, and the results are shown in Table. 1. In the "w/o appearance feature" ablation study, we replaced the appearance feature vector with a zero vector to disable the appearance feature while keeping other modules unchanged. We found that the model without appearance features loses its ability to infer the color of novel views entirely, as the generalizable renderer heavily relies on appearance cues from input views. In the "w/o geometry feature" ablation study, we replaced the geometry feature vector with a zero vector to disable the geometry feature while keeping other modules unchanged. We observed that although the model can still infer normal results, its performance is significantly worse than the original (full) model.

D Comparisons with NeuRay [7] Given Multi-view Inputs

Our method is not limited to two panoramas. For instance, when rendering a test view in the renderer module, we use N input panoramas as reference images, and the renderer does not need to be modified. In the 360° spherical depth estimator module, for each reference image, we use the other $N-1$ input panoramas as source images. We average the multiple cost-volumes obtained during 360° multi-view matching process between the reference image and each source image. The rest is unchanged. In this way, our method can be applied to the multi-view panoramic inputs.

To further verify the effectiveness of our method, we conducted comparative experiments with NeuRay using multiple panoramic image as inputs. We placed the four input viewpoints at the

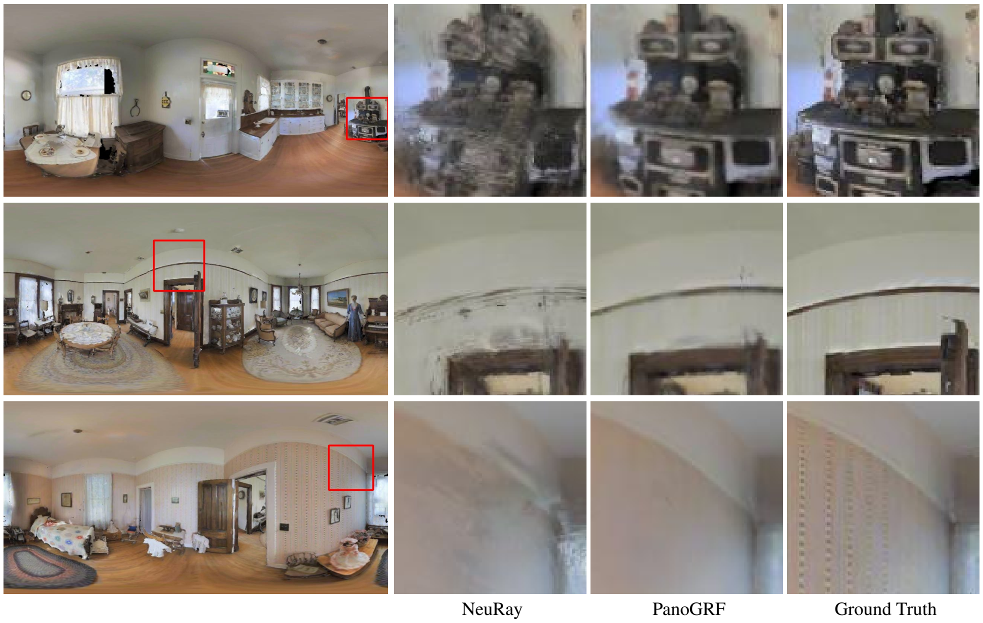


Figure 4: Qualitative comparisons with NeuRay beyond the camera baseline on Matterport3D. We synthesized novel views at positions 0.25 meters above the middle point between two input viewpoints. The input viewpoints are 1.0 meters apart. Our method can achieve better results than NeuRay beyond the camera baseline.

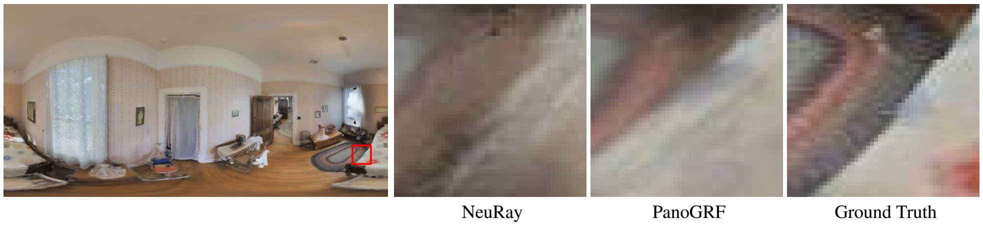


Figure 5: Failure case. The carpet area behind the bed was not visible in the input viewpoints, but it becomes visible in higher viewpoints. NeuRay and PanoGRF tend to produce different and blurry results compared to the ground truth because they lack generative power. Combining existing diffusion generative models could potentially address this issue. We leave this as future work.

corners of a horizontal square and tested the rendering performance at the center viewpoint and other viewpoints located at -0.4, -0.2, -0.1, -0.05, 0.05, 0.1, 0.2, and 0.4 meters in the vertical direction from the center viewpoint. The diagonal length of the square is 1.0 meters. Table. 7 and Fig. 3 present the quantitative and qualitative comparison results between PanoGRF and NeuRay. As shown, PanoGRF still largely outperforms NeuRay with multiple panoramic inputs.

This experiment is added during the rebuttal period. Due to the limited time, we trained PanoGRF only for 20k iterations and NeuRay for 80k iterations. The learning decay strategies are similar to the setting of two views.

E Comparisons with NeuRay [7] beyond Camera Baseline and Failure Case

We conducted an additional experiment where novel views were generated at positions 0.25 and 0.5 meters above the middle point between two input viewpoints. The input viewpoints are 2.0 meters apart. We compared the quantitative and qualitative results of our method with NeuRay's as shown in Table. 8 and Fig. 4. PanoGRF consistently surpassed NeuRay's performance, indicating its capacity to yield superior results beyond the camera baseline.

We also present the failure case of PanoGRF under such condition in Fig. 5. In some viewpoints, a previously occluded area may be visible and since this area has not been seen in either of the two

Table 8: Comparisons with NeuRay [7] beyond Camera Baseline on Matterport3D

Dataset		Matterport3D (1.0m)			
distance	method	PSNR↑	WS-PSNR↑	SSIM↑	LPIPS↓
0.25m	NeuRay	20.66	20.05	0.714	0.409
0.25m	PanoGRF	**21.98**	**21.30**	**0.763**	**0.348**
0.5m	NeuRay	20.39	19.94	**0.876**	**0.195**
0.5m	PanoGRF	**21.99**	**21.42**	**0.769**	**0.349**

existing viewpoints, the synthesis of this area is not effective. The area, which has not been seen in either of the existing viewpoints may be able to be filled in by combining with the existing diffusion generative approach. This is the next direction we plan to investigate.

F Experiments for Mono-guided Spherical Depth Estimator

F.1 Ablation Studies for Mono-guided Spherical Depth Estimator

To further validate the effectiveness of the key components, namely 360° multi-view stereos and 360° monocular depth, we conducted ablation studies specifically focused on spherical depth estimation. For evaluation purposes, we selected three commonly used metrics: L_1, L_2, and RMSE. Additionally, we also used WS-L_1, WS-L_2, and WS-RMSE as metrics, which incorporate weighted latitudes of equirectangular images to simulate WS-PSNR [11]. This approach aims to mitigate the impact of equirectangular projection distortion. We selected the first 1000 panorama pairs from Matterport3D as our test data, with a camera baseline of 1 meter. For the evaluation, we considered depth values within the range of $[0.1, 10]$ as valid.

The quantitative results of our ablation studies are presented in Table 2, while the qualitative results can be observed in Fig. 6. The experiments clearly demonstrate the importance of each module in achieving accurate depth estimation. Removing either the 360° multi-view stereo or the 360° monocular depth significantly reduces the depth accuracy. Using only 360° monocular depth does not guarantee multi-view consistency, resulting in potential discrepancies between the predicted scale and the ground truth in certain regions. The occlusion problem poses a challenge for using only 360° MVS, particularly at the boundaries of objects. Consequently, the depth predictions in these regions are inaccurate and lack detail.

F.2 Different Backbones for 360° Multi-view Stereo

Introducing 360° monocular depth to 360° multi-view stereo does result in an increase in the number of model parameters. However, it is important to note that the improvement in depth accuracy is not attributed to the increase in parameters. In our experiments, we increased the model size of 360° MVSNet by using larger backbones, specifically ResNet [3].

From Table. 3, we found that L_2 and RMSE of 360° MVSNet(ResNet-101) are still far inferior to those of 360° MVSNet(ResNet-18) together with the 360° monocular depth network [4]. This observation suggests that simply increasing the model size does not effectively address the view inconsistency problem in the wide-baseline setting. On the other hand, the introduction of 360° monocular depth provides a qualitative improvement to the accuracy of 360° multi-view stereo. By incorporating monocular depth information, the model gains additional cues that help mitigate the view inconsistency issue and improve depth estimation performance. Furthermore, when larger backbones were used as replacements, it was found that the performances of 360° MVS+Mono deteriorated. This could be attributed to the excessive model parameters leading to overfitting.

F.3 Different Hyperparameters for Mono-guided Spherical Depth Estimator

We conducted evaluations to assess the impact of different values for σ (standard deviation) and N_{mono} (number of monocular depth candidates) on mono-guided spherical depth sampling. The results are presented in Table 4 and Table 5.

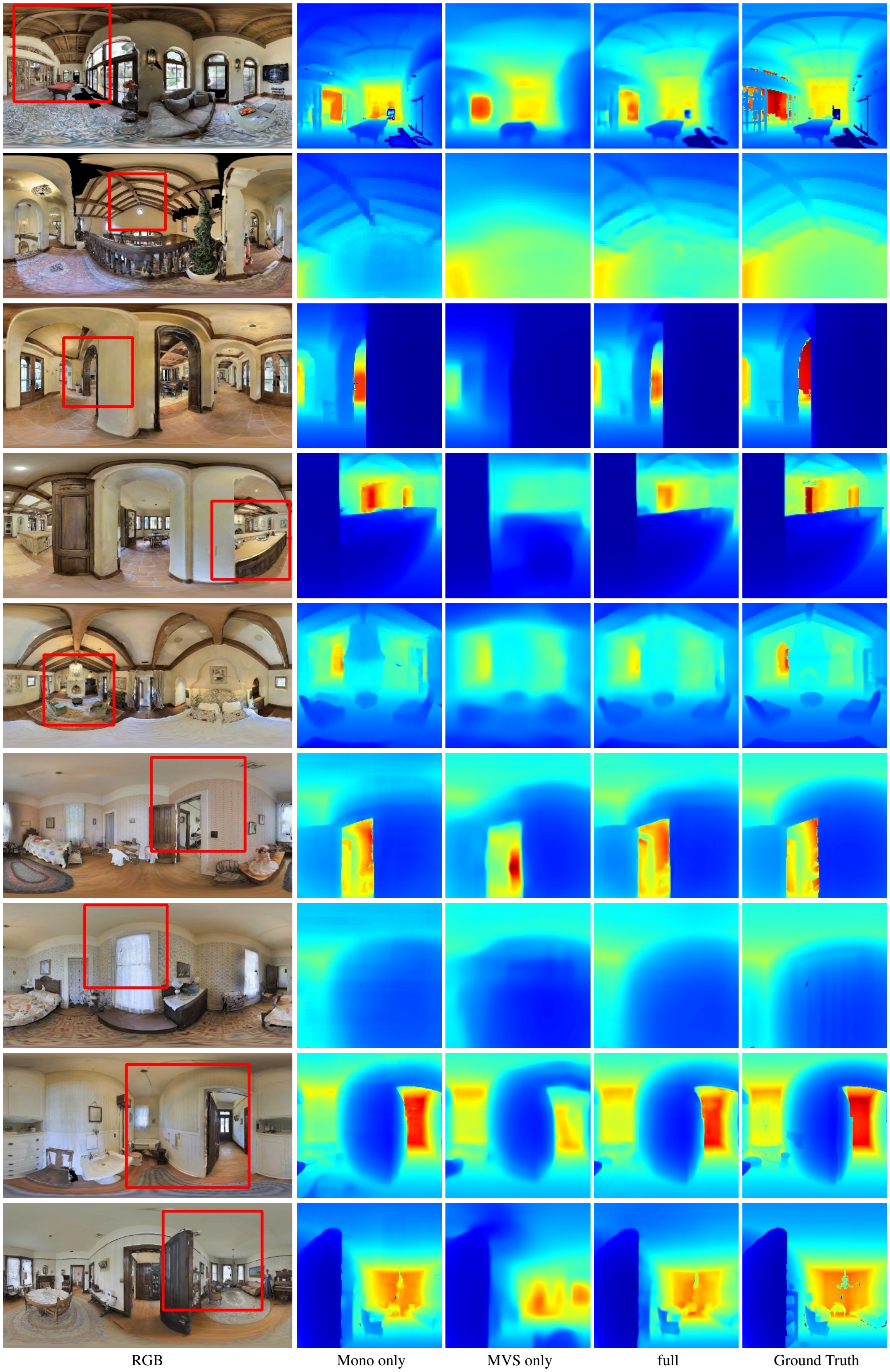
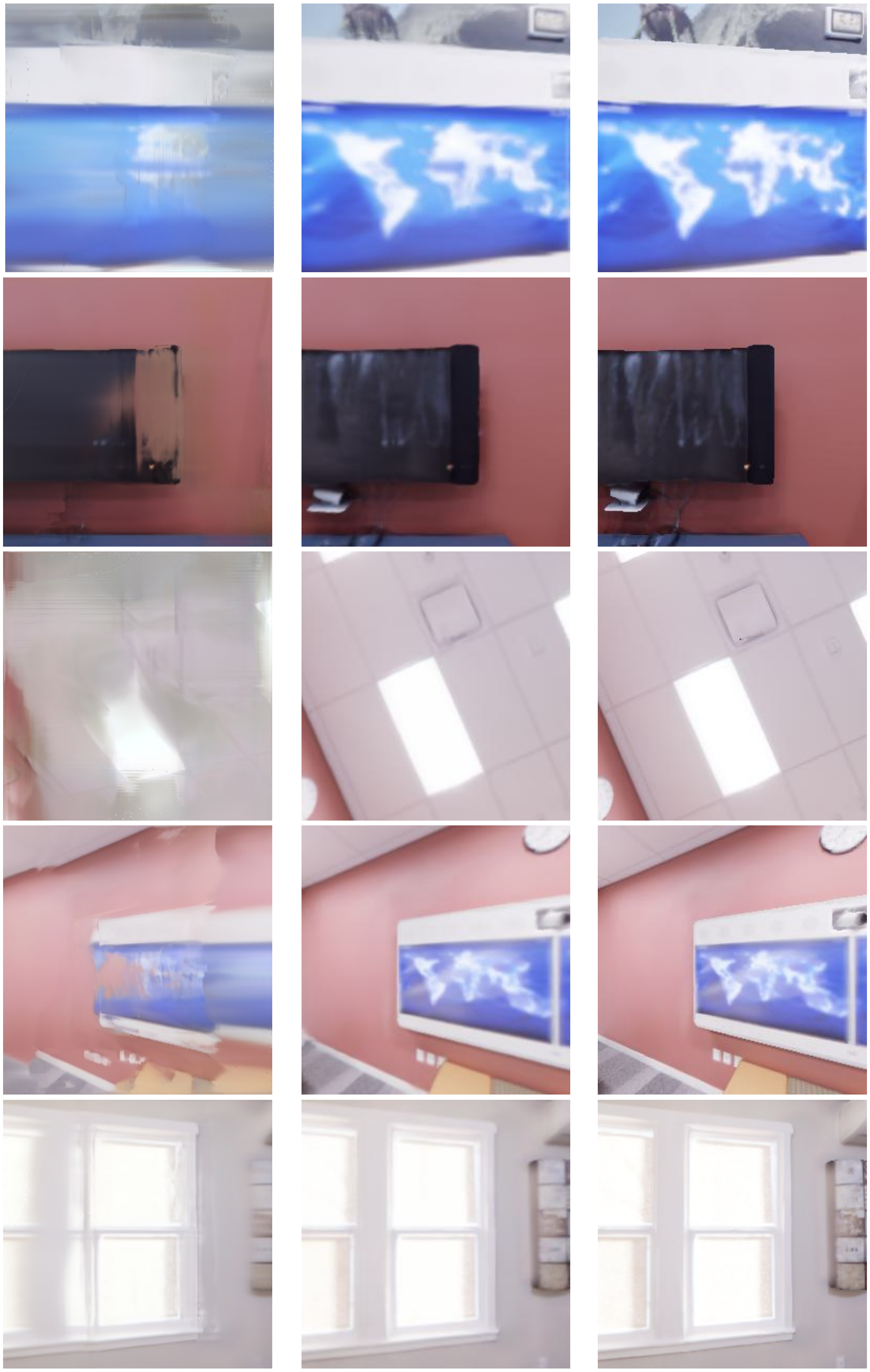


Figure 6: Qualitative results of ablation studies for depth estimation on Matterport3D. *Mono only* and *MVS only* respectively refer to the results obtained when using only 360° monocular depth and 360° multi-view stereo.



Cross Attention Renderer PanoGRF Ground Truth

Figure 7: Qualitative comparisons with Cross Attention Renderer on Replica

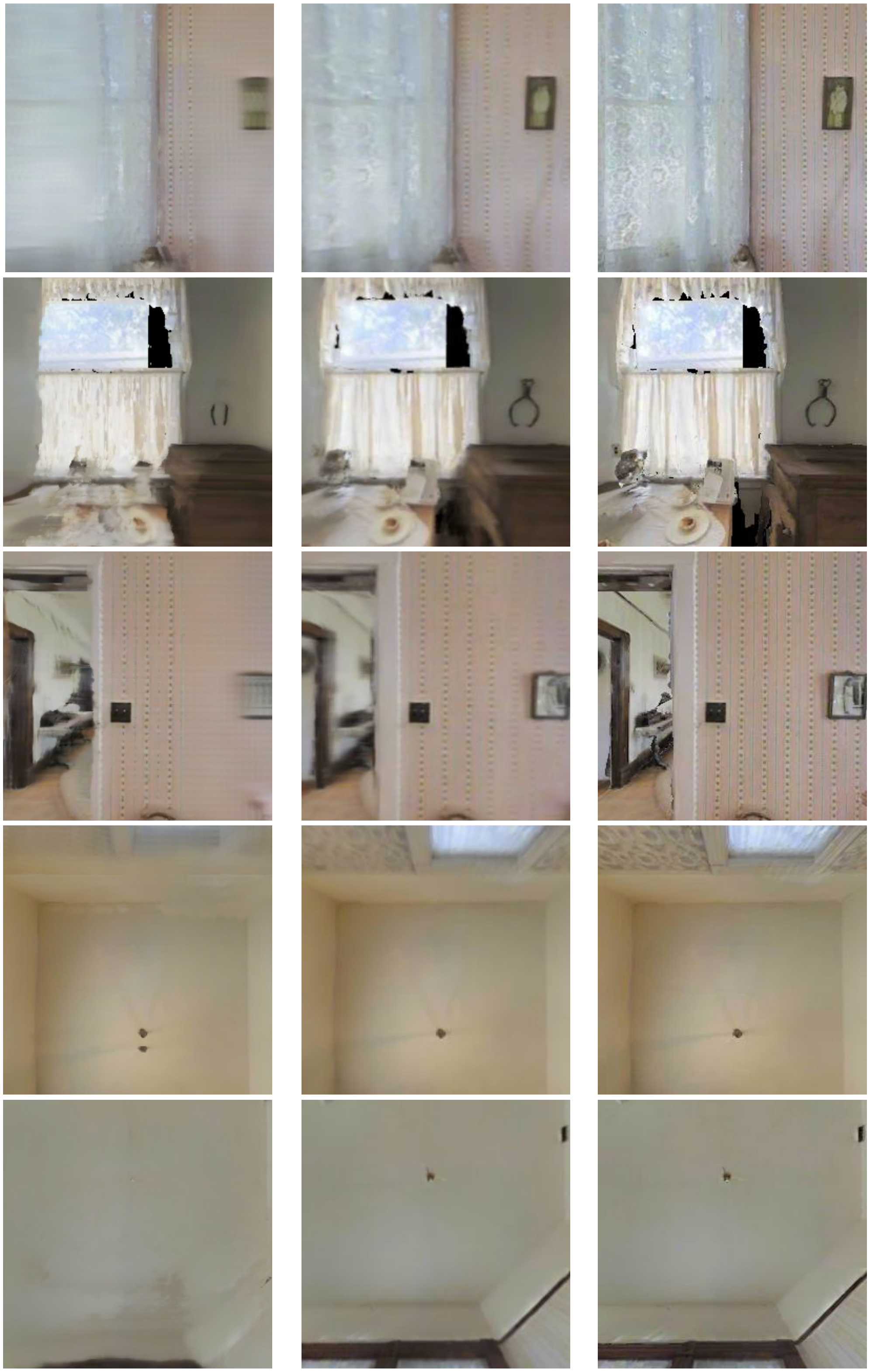


Figure 8: Qualitative comparisons with Cross Attention Renderer on Matterport3D

Table 9: The quantitative results of fine-tuning of the general renderers. The best results are in bold.

baseline		1.0m				0.2m			
method	setting	PSNR↑	WS-PSNR↑	SSIM↑	LPIPS↓	PSNR↑	WS-PSNR↑	SSIM↑	LPIPS↓
NeuRay	gen	27.751	27.253	0.8470	0.2565	34.318	33.376	0.9409	0.1158
PanoGRF	gen	**28.818**	**28.487**	**0.8778**	**0.1996**	35.817	35.056	0.9554	0.0959
NeuRay(+Consist)	ft	26.755	26.250	0.8181	0.2820	33.354	32.359	0.9240	0.1304
PanoGRF	ft	24.341	23.941	0.8175	0.2834	**35.828**	**35.138**	**0.9578**	**0.0912**
PanoGRF (+Depth)	ft	27.399	27.202	0.8556	0.2446	33.691	33.066	0.9333	0.1342

The reliability of 360° monocular depth estimation is not perfect. Therefore, our paper employs uniform sampling to compensate for the remaining depth candidates. In the ablation experiments, we consistently maintain $N_{mono} + N_{uni} = 64$, where N_{uni} represents the sample number of uniform distribution. We discovered that the configuration yields the best results with regard to the metrics of WS-L1, WS-L2, and WS-RMSE.

G Comparisons with Cross Attention Renderer [1]

The Cross Attention Renderer (CAR) [1] is a method that operates on a wide-baseline perspective pair. We divided the two panoramas into cube maps and utilized the corresponding sides of the cube maps as inputs for CAR. Specifically, we rendered the corresponding side of the cube maps at the middle viewpoint. For example, we input the left side of the cube maps and generate the left side of the cube maps at the intermediate viewpoint as the output. We repeated this process for all six corresponding pairs of cube maps. In contrast, our method directly takes two panoramas as input and performs ray casting based on perspective projection. We then render the results for each side of the cube maps at the intermediate viewpoint, preserving the panoramic nature of the input.

We conducted quantitative comparative experiments on Matterport3D and Replica. The qualitative comparisons with CAR on Replica and Matterport3D can be seen in Fig. 7 and Fig. 8. The results clearly demonstrate that PanoGRF significantly outperforms CAR in terms of rendering quality. CAR suffers from limitations associated with its input field-of-view (FoV), particularly in edge regions that are only visible from a single perspective view. CAR relies on a pure stereo-matching method for geometric estimation, which leads to suboptimal rendering performance, as evidenced by the results presented in Table. 6. In contrast, PanoGRF is specifically designed to handle full FoV inputs, and it mitigates the issue of view inconsistency by incorporating the 360° monocular depth network.

H Fine-tuning of PanoGRF

We conducted per-scene fine-tuning for NeuRay [7] and PanoGRF on the first test scene of Matterport3D with baselines of 1.0 and 0.2 meters, respectively. The general renderers were fine-tuned for 10k iterations, and the quantitative results are presented in Table.9. Under the baseline of 1.0 meters, the general renderer of PanoGRF, denoted as PanoGRF-gen, achieved the best performance. NeuRay-ft, the fine-tuned renderer of NeuRay, underwent fine-tuning using RGB loss and depth consistency loss, following the methodology described in their original paper [7]. PanoGRF-ft was fine-tuned with only RGB loss. However, the results of NeuRay-ft and PanoGRF-ft were inferior to NeuRay-gen and PanoGRF-gen, respectively. This suggests that fine-tuning under a wide-baseline setting does not improve the performance of general renderers. It is likely that fine-tuning with a wide baseline leads to overfitting. Even with the introduction of a depth uncertainty loss [9] by supervising the renderer depth of PanoGRF with predicted spherical depths during the fine-tuning process, the results of PanoGRF-ft remained inferior to those of the general renderer of PanoGRF. On the other hand, the performance of PanoGRF-ft under the baseline of 0.2 meters was slightly better than that of PanoGRF-gen. However, adding the depth loss [9] was still unable to improve PanoGRF-gen when fine-tuning. Inaccurate predicted depths in certain regions may be the cause, misleading the estimation of NeRF's geometry and thereby reducing the rendering performance.

---

[1] Cross Attention Renderer: https://github.com/yilundu/cross_attention_renderer

Additionally, NeuRay-ft was still inferior to NeuRay-gen under the narrow baseline. This may be attributed to the limited field-of-view, which can result in the aggregation of incorrect features. When projecting 3D sample points onto other source perspective views (cube-maps), these points may fall outside the image borders of the source perspective view or be located behind the source perspective camera.

I Quantitative Comparisons with Baseline Methods for Narrow-baseline Panoramas

Table 10: Quantitative comparisons with baseline methods on Matterport3D under the baseline of 0.2 and 0.5 meters. The best results are in bold.

baseline	0.2m				0.5m			
method	PSNR↑	WS-PSNR↑	SSIM↑	LPIPS↓	PSNR↑	WS-PSNR↑	SSIM↑	LPIPS↓
S-NeRF	20.79	19.52	0.6967	0.3756	17.95	16.81	0.6278	0.4856
OmniSyn	28.95	28.26	0.9132	0.1804	26.59	26.07	0.8897	0.2005
IBRNet	30.53	29.63	0.9271	0.1363	28.22	27.26	0.8844	0.1987
NeuRay	33.54	32.33	0.9485	0.1074	30.88	29.81	0.9196	0.1536
PanoGRF	**34.29**	**33.27**	**0.9515**	**0.0977**	**31.41**	**30.46**	**0.9238**	**0.1318**

We compared PanoGRF with baseline methods under the baseline of 0.2 and 0.5 meters on Matterport3D. As shown in Table. 10, the quantitative results demonstrate that PanoGRF consistently outperforms all the baseline methods under the baseline of 0.2 and 0.5 meters. These findings indicate that our method is applicable to both wide-baseline and narrow-baseline panoramas. In comparison to generalizable methods designed for perspective views, our method is particularly well-suited for synthesizing panoramic views by leveraging the aggregated features based on spherical projection.

J More Details of PanoGRF

J.1 Renderer

J.1.1 Training

PanoGRF employs the Adam optimizer [5] with an initial learning rate of 4.0e-4. The pre-training process of PanoGRF was conducted on an A100 GPU for 100k iterations, which required approximately two days. The learning rate is halved every 20k iterations, and a batch size of 512 was used during training.

J.1.2 Architecture

We adopted a coarse-to-fine sampling approach, similar to NeRF [8], and sampled 64 points in both phases. We followed a similar architecture as NeuRay [7] to build our renderers. The coarse and fine renderers share the same image encoder, geometry feature extractor, and visibility encoder. But they have different distribution decoders and aggregation networks $\mathcal{F}$, similar to NeuRay. During training, the weights of the 360° spherical depth estimator are fixed due to GPU memory limitations. Our image encoder, visibility encoder, distribution decoder, and aggregation network are implemented similarly to NeuRay, except for the padding mode. In the convolution layers of the image encoder and visibility encoder, we employ circular padding horizontally and zero padding vertically instead of direct zero padding. This is done to adapt to equirectangular image inputs and simulate Circular CNNs [10]. The circular padding helps account for the continuity of pixels at the leftmost and rightmost edges of equirectangular images, which are neighbors, while the top and bottom edges are not. The distribution network consists of 5 sub-networks, each with 3 fully connected layers. The appearance feature extractor is a ResNet [3] which contains 13 residual blocks and outputs the appearance feature map with 32 channels. The architecture details of the geometry feature extractor are shown in Fig. 9. The image encoder in the geometry feature extractor contains 14 residual blocks. The spherical depth input for the geometry feature extractor is first downsampled to 1/4 of its original resolution and then fed into the extractor to reduce GPU occupancy.

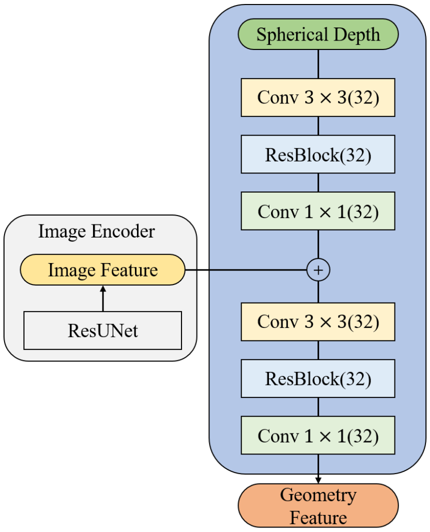


Figure 9: Architecture of geometry feature extractor.

J.2 Spherical Depth Estimator

J.2.1 Training

We initially train the 360° monocular depth network and then freeze its weights when training the remaining components of 360° MVSNet. The Adam optimizer is employed with a fixed learning rate of 0.0001. Both the 360° monocular depth network and 360° MVSNet are trained for 100k iterations, which required approximately one day on a single V100 GPU. The batch size is set to 2.

J.2.2 Architecture

The 360° monocular depth network [4] and 360° MVSNet utilize ResNet-18 as the feature extractor. The 3D CNN regularization network is composed of 3 downsampling and 3 upsampling blocks, similar to [6]. The depth decoder consists of 2 convolution blocks. The feature map obtained from the middle layer in the 360° monocular depth network is concatenated with the regularized spherical cost volume and then decoded into 360° depth by the depth decoder. In the multi-view matching process, we compute the similarity by subtracting the feature vectors.

K Datasets

For Matterport3D, we split the training and testing set following SynSin [13]. The first 10 scenes of the test set are used for evaluation. In the case of the Replica dataset, we render a total of 18 scenes for evaluation. Additionally, we utilize the Residential dataset provided in [2], which comprises three scenes. For this dataset, we select the first and last panorama as the input views.

L Training Details of Baseline Methods

We trained NeuRay [7] and IBRNet [12] for 400k and 250k iterations, respectively. Spherical NeRF (S-NeRF) [8] underwent training for 2000 epochs, equivalent to approximately 256k iterations with the batch size of 4096. For OmniSyn [6], the in-painting network was trained for 50k iterations, which took approximately 3 days on a TitanRTX GPU. The depth estimator of OmniSyn was trained for 100k iterations. Due to limitations in GPU memory, OmniSyn was trained at a resolution of 256 × 256, and its output was resized to 512 × 1024 for evaluation purposes. CAR [1] was trained for 300k iterations.

References

[1] Yilun Du, Cameron Smith, Ayush Tewari, and Vincent Sitzmann. Learning to render novel views from wide-baseline stereo pairs. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[2] Tewodros Habtegebrial, Christiano Gava, Marcel Rogge, Didier Stricker, and Varun Jampani. Somsi: Spherical novel view synthesis with soft occlusion multi-sphere images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15725–15734, 2022.

[3] 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.

[4] Hualie Jiang, Zhe Sheng, Siyu Zhu, Zilong Dong, and Rui Huang. Unifuse: Unidirectional fusion for 360 panorama depth estimation. *IEEE Robotics and Automation Letters*, 6(2):1519–1526, 2021.

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[6] David Li, Yinda Zhang, Christian Häne, Danhang Tang, Amitabh Varshney, and Ruofei Du. Omnisyn: Synthesizing 360 videos with wide-baseline panoramas. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pages 670–671. IEEE, 2022.

[7] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7824–7833, 2022.

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

[9] Barbara Roessle, Jonathan T Barron, Ben Mildenhall, Pratul P Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12892–12901, 2022.

[10] Stefan Schubert, Peer Neubert, Johannes Pöschmann, and Peter Protzel. Circular convolutional neural networks for panoramic images and laser data. In *2019 IEEE intelligent vehicles symposium (IV)*, pages 653–660. IEEE, 2019.

[11] Yule Sun, Ang Lu, and Lu Yu. Weighted-to-spherically-uniform quality evaluation for omnidirectional video. *IEEE signal processing letters*, 24(9):1408–1412, 2017.

[12] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021.

[13] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. SynSin: End-to-end view synthesis from a single image. In *CVPR*, 2020.