Supplementary Material for: "Hybrid Neural Rendering for Large-Scale Scenes with Motion Blur"

1. Overview

This document begins by presenting additional implementation details on network structures, the process of generating synthetic data, and content-aware blur detection in Section 2. We then investigate model capacity in Section 3 before conducting additional experiments and presenting more results on challenging scenarios and diverse datasets in Section 4.

2. More Implementation Details

Network Architectures Our network consists of three major components. *1) The image feature extractor (G).* It extracts multi-scale image features using 3×3 kernels. We double the intermediate channels when down-sampling with a stride of 2. Totally, it has six layers with intermediate channels of {6, 6, 12, 12, 24, 24}. *2) MLPs in the feature aggregation.* In Figure 3 of the main paper, the blue MLP consist of three layers with intermediate channels of {128, 128, 128}, the green MLP has four layers with intermediate channels of {64, 64, 64, 1}, and the yellow MLP has three layers with intermediate channels of {45, 45, 45}. *3) MLPs for color and volume density prediction.* In Figure 2 of the main paper, the purple MLP used to predict volume density has five layers with intermediate channels of {256, 256, 256, 256, 1}. Additionally, the black MLP that generates color from the hybrid feature has one layer with three output channels.

Synthetic Data The 480×640 RGB-D image sequences synthesized from Habitat-sim [5, 8, 9] are sharp. Following the train and test splits on ScanNet, we divide the entire dataset into training sets (every 5th image) and testing sets. We then simulate motion blurs [3] on the training sets (RGB images) to obtain blurry training images. Specifically, each image has a probability of 0.75 of being blurred using a motion blur simulator [3]. The simulator has two hyperparameters: size and intensity, which control the properties of the motion blur. The size parameter k controls the moving distance, and is uniformly sampled from the range $k \in (3, 16)$. The intensity parameter ϕ determines the level of shake in the moving direction. As we assume that the camera moves in one direction at a given moment while capturing high frame rate videos, we randomly choose a small intensity value of $\phi \in (0, 0.1)$. Note that our model is trained using randomly sampled small patches, and therefore, this blur simulation process is equivalent to operating on small patches.

Content-Aware Blur Detection To calculate the blurriness score, we use a method called "variation of the Laplacian" [7]. First, we apply a blur kernel (*i.e.*, mean filter in our paper) to the original RGB image I to reduce the influence of noise. Then, we use the Laplacian operator to extract the high-frequency components $\hat{H}$ from the denoised image $\hat{I}$. Finally, we obtain the blurriness score by computing the variation of all high-frequency components $S = var(\hat{H})$.

To prevent the inaccurate blurriness score of one frame from affecting the subsequent frames, we implement our content-aware detection (Sec. 3.2 in the main paper) using a sliding window scheme in practice. Specifically, the sliding window incorporates 10 frames (*i.e.*, $N = 10$) to calculate quality-aware weights ω_i^b, and moves in steps of 5 on the time axis. If a frame is covered multiple times by the sliding window, the final quality-aware weight is the average of all weights belonging to that frame. In this paper, we use consecutive frames since videos are provided. Similarly, neighboring frames can also be used to compute overlapping regions in place of consecutive frames.

3. Model Capacity

To further confirm that the improvements of our method, which builds upon Point-NeRF, are not dependent on larger model capacities, we conducted two experiments on Point-NeRF: *1) Gradually increasing the number of points from 4.2 million to 7.4 million, and 2) Increasing the channels of each point descriptor from 32 to 63.* Results in Table 1 indicate that naively increasing the capacity of Point-Nerf does not improve performance. Moreover, the total number of parameters optimized in Point-NeRF with 7.4 million points (model size: 1.2 GB) already exceeds that of our hybrid model with 4.2 million points (model size: 682 MB, PSNR: 31.25).

Number of points	4.2M	5.3M	6.3M	7.4M	4.2M
Number of channels	32	32	32	32	63
Point-NeRF (PSNR)	30.54	30.56	30.55	30.50	30.58

Table 1. Results on "Scene241_01" while increasing the number of point descriptors and channels of each descriptor. Naively increasing the capacity of Point-NeRF cannot boost performance.

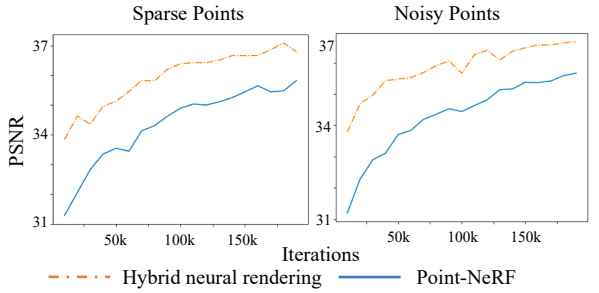

Figure 1. The trend of PSNR across different training iterations is shown for the case of sparse points and noisy points. When using image features (*i.e.*, hybrid neural rendering), the performance is better than the neural-3D-feature-only design (*i.e.*, Point-NeRF).

4. More Experiments

Sparse and Noisy Points We conduct experiments on challenging situations using sparse (0.6M points) and noisy (add gaussian noise $\mathcal{N}(0, 0.05)$) points on 'VangoRoom' (to validate the effectiveness of image features, all blur-related designs are disabled, and images used for training and testing are blur-free). As shown in Fig. 1, using image features converges faster and achieves better performance than Point-NeRF, which only utilizes neural 3D features.

ARKITScenes Dataset We present results on the ARKITScenes dataset ("Scene_40776204") [2]. Fig. 2 demonstrates the effectiveness of our hybrid rendering and blur-handling designs, where all components contributing to the sharpness of rendered images (note the bird). Quantitative results in Table 2 indicate that our method outperforms Point-NeRF on all three metrics.

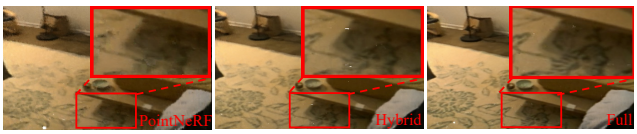

Figure 2. Qualitative results on the ARKITScenes dataset. Each component of our proposed method contributes to the sharpness of rendered images.

NeRF Synthetic Dataset We provide results on the NeRF synthetic dataset ("Chair") [6], which only provides posed RGB images without depth information. When trained for 66 minutes, our method leveraging image-based features produces results with more high-frequency details, as displayed in Fig. 3. This observation is also corroborated by quantitative results presented in Table 3.

"Scene_40776204"	PSNR↑	SSIM↑	LPIPS↓
Point-NeRF	31.02	0.947	0.212
Ours(H)	**32.55**	**0.961**	**0.127**
Ours(Full)	31.97	0.957	0.135

Table 2. Quantitative results on the ARKITScenes dataset.

ScanNet Dataset In Fig. 4, we show more qualitative results and compare them with Point-NeRF to demonstrate the superiority of our approach. We observed that the outputs of Point-NeRF suffer from blurry outputs and noisy edges, while "Ours (H)" contains more details and smoother appearance, evident in the toy plane in the third row and characters on the posters in the last two rows. Moreover, the image sharpness is further improved while applying our designs to handle blur artifacts.

Video Results In the video results, we compare our approach with other baselines, including NeRF [6], IBRNet [10], NPBG [1], Point-NeRF [11] and DeblurNeRF [4], to demonstrate the superiority of our approach in terms of quality and consistency. Moreover, we also validate the effectiveness of our designs, including hybrid rendering, handling blurriness, random drop, and neural 3D features.

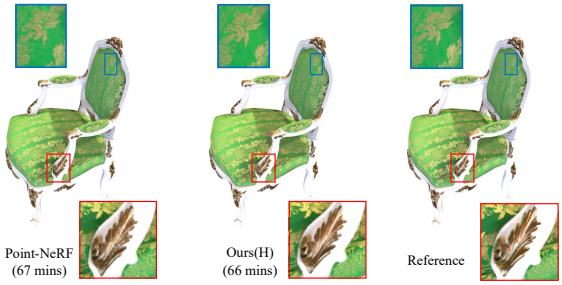

Figure 3. Qualitative results on the NeRF synthetic data. Leveraging image features assists in high-frequency details generation.

"Chair"	PSNR↑	SSIM↑	LPIPS↓
Point-NeRF$_{40k}$ (67 mins)	33.80	0.986	0.017
Ours (H)$_{20k}$ (66 mins)	34.40	0.988	0.016
Point-NeRF$_{200k}$	35.70	0.992	0.010
Ours (H)$_{200k}$	**36.23**	**0.993**	**0.009**

Table 3. Quantitative results on the NeRF synthetic dataset. Our method outperforms Point-NeRF in both scenarios where training is performed for the same amount of time and the same number of iterations.

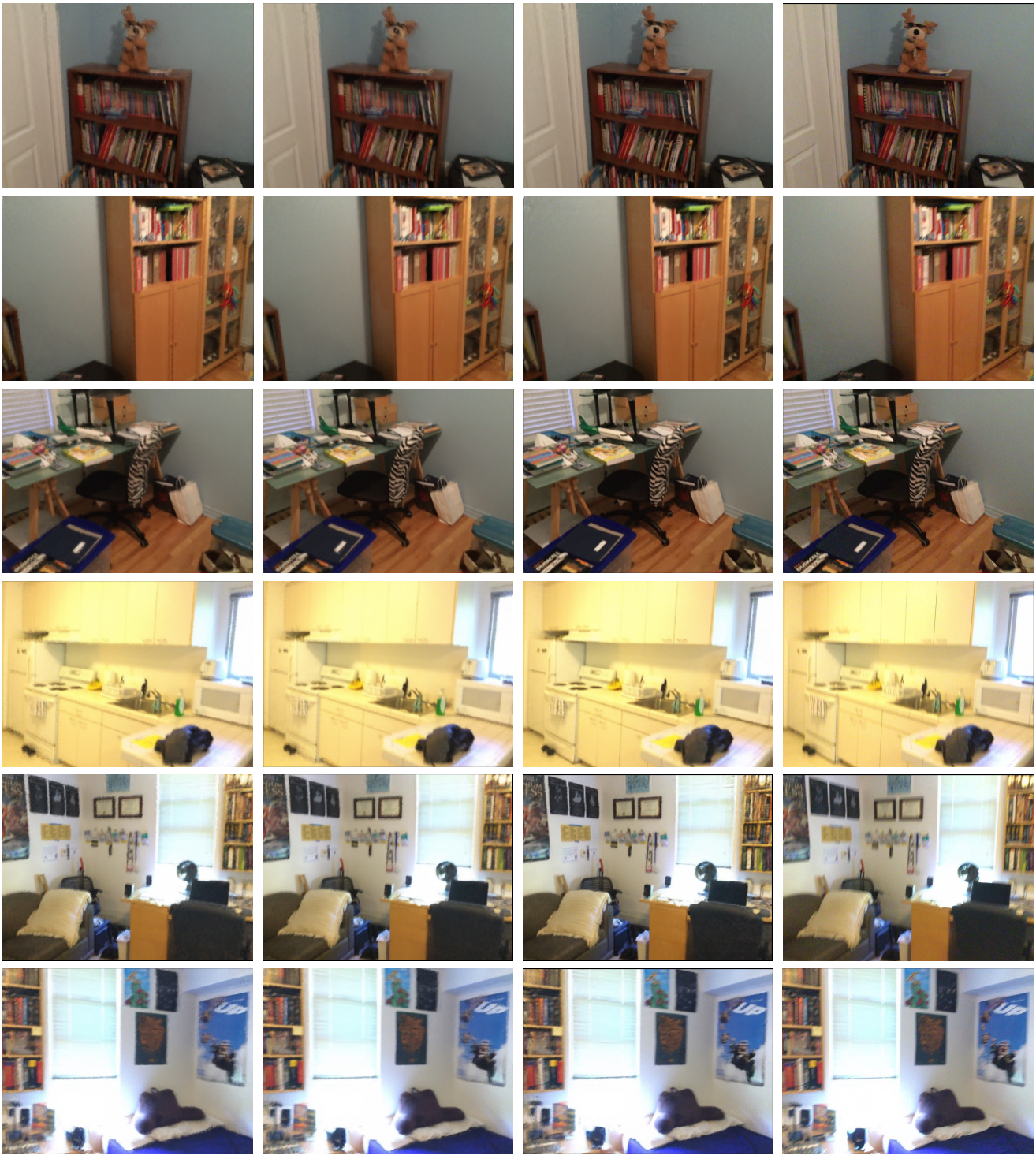

Figure 4. More results compared to Point-NeRF.

References

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *European Conference on Computer Vision*, pages 696–712. Springer, 2020.

[2] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. ARKitscenes - a diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[3] LeviBorodenko. Motionblur, 2020. https://github.com/LeviBorodenko/motionblur.

[4] 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.

[5] Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[6] 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.

[7] José Luis Pech-Pacheco, Gabriel Cristóbal, Jesús Chamorro-Martinez, and Joaquín Fernández-Valdivia. Diatom autofocusing in brightfield microscopy: a comparative study. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 3, pages 314–317. IEEE, 2000.

[8] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[9] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[10] 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.

[11] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022.