Supplementary Material for "HDR-Plenoxels: Self-Calibrating High Dynamic Range Radiance Fields"

Kim Jun-Seong[1*] Kim Yu-Ji[2*] Moon Ye-Bin[1] Tae-Hyun Oh[1,2†]

[1]Dept. of Elect. Eng. [2]Grad. School of AI
Pohang University of Science and Technology (POSTECH)
{junseong.kim, ugkim, ybmoon, taehyun}@postech.ac.kr
https://github.com/postech-ami/HDR-Plenoxels

Contents

Supplementary Material

This supplementary material presents technical details, results, and experiments not included in the main paper due to the space limit.

A Technical Details

In this section, we explain the details of experimental settings (in Sec. A.1) and method (in Sec. A.2) of our high dynamic range radiance Plenoxels (HDR-

*Authors contributed equally to this work.
†Joint affiliated with Yonsei University, Korea.

Plenoxels) and other baselines. We build our code based on PyTorch open library [8] and use one NVIDIA GeForce RTX 3090 GPU or A100 for training and rendering novel view synthesis.

A.1 Experimental Settings

Synthetic Dataset. To generate the synthetic dataset, we use Blender [2] to modify the various camera. Each image is created in OpenEXR format and conducted post-processing like changing exposure and white balance at high dynamic range (HDR) colorspace for comparing physical camera tone mapping. Synthetic scenes have five different scenes, *i.e.*, book [3], classroom (CC0) [14], kitchen (CC-BY) [4], palace (CC-BY) [1], and room [3]. To demonstrate the ability of novel view synthesis to our HDR-Plenoxels, we generate synthetic data with complex geometry and various radiometric conditions. In the test stage, we split the test image into left- and right- half. We train the left-half of a test image and test with the right-half because our method needs trained parameters of the tone mapping module for rendering. The left-half of the test image is trained, and the right-half is used at test time. Exposure was set to $\pm$3EV at the basis image, white balance was applied by multiplying each color channel with 1.25 separately, and camera response function (CRF) was set to gamma correction with $\gamma = 3$. We show experimental results on five synthetic datasets. Each scene is created at an 800×800 pixels resolution, with views sampled from the roughly forward-facing camera. We use 43 views of each scene as training input and 7 for testing.

Real Dataset. We take all real scenes with exposure bracketing setting, and changing white balance measured in Kelvin. Various camera shooting conditions, *i.e.*, exposure value with three intervals and white balance with 3000K, 3500K, and 4000K, are applied sparsely to whole datasets. Our real datasets are taken with Canon EOS 5D Mark *IV*, captured 30 to 50 views, and taken 1/5 as a test set. Most images are taken in strong sunlight or darkroom, so each pixel is easily saturated and needs an HDR colorspace to represent the accurate color of a pixel. The experimental results on four real datasets are shown in Sec. B.

Hyperparameters. We follow the learning processes suggested by Plenoxels [15] with some modifications. We train our HDR-Plenoxels with 10 epochs, a total of 128,000 iterations each, with 5,000 rays per batch. We set a learning rate of the spherical harmonics (SH) and tone mapping parts to pure exponential decay. The learning rate of SH and tone mapping starts from 0.01, and both decays to $5 \cdot 10^{-5}$ at step 250,000 to match each other's learning speed. Voxel opacity σ is updated using a delayed exponential function with decaying up to 0.05 during 250,000 iterations.

For learning stability of opacity σ and SH, total variance (TV) loss is applied only for the 3 epochs until the first upsampling process. We apply the weight of TV loss for opacity σ and SH as $\lambda_{\mathrm{TV},\sigma} = 5 \cdot 10^{-4}$, $\lambda_{\mathrm{TV,SH}} = 1 \cdot 10^{-2}$. Each loss is updated with RMSProp optimizer.

Our HDR-Plenoxels are a voxel grid-based method, so it is important to find the proper range of the initial grid for retaining the expressible volume. Several

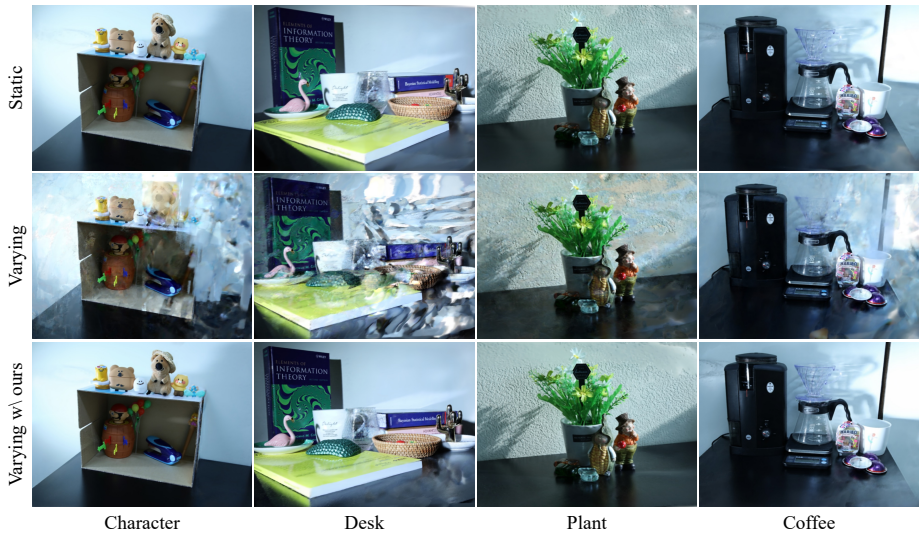

Fig. S1. Results of static and varying camera settings in real scenes. The static and varying represent different camera conditions (*i.e.*, exposure, white balance, and camera response function (CRF)). In static camera conditions, all views of the scene have the same exposure, white balance, and CRF, and the varying one is vice versa. The first two rows are results from original Plenoxels, and last row is from HDR-Plenoxels. Each column represents a different real scene.

scenes contain large depth in synthetic Blender data, which is hard to express with a default concentric sphere grid. To properly determine the grid range, we first compute the rough 3D geometry of the scene, and we then obtain the camera poses with the 3D boundary of the scene through the COLMAP [12,13] sparse reconstruction. We start grid voxel resolution in (128, 128, 64) and then upsample the resolution in the order of (256, 256, 128), (512, 512, 256), and (800, 800, 512) after each 25,600 iterations.

We similarly obtain the unknown camera pose through COLMAP and initialize the grid in the real data. In particular, we conduct undistortion of the entire image through calibration before the training learning due to the lens distortion frequently appearing in the image. The resolution of the real image is updated in the order of (128, 128, 128), (256, 256, 256), and (750, 600, 300).

A.2 Baseline Method Details

NeRF in the Wild (NeRF-W) [6]. We conduct the comparison of NeRF-W based on the PyTorch version of NeRF-W implementation [9] with the same width and depth of the original model [6]. We train the model with the left half side of the images in the training set and test the novel view synthesis on the right half side of the images. We use NeRF-A (appearance) without transient embeddings for a fair comparison because our dataset has no transient parts.

Approximate Differentiable One-Pixel Point Rendering (ADOP) [11]. We use the official code of [10]. Given the COLMAP dense reconstruction results,

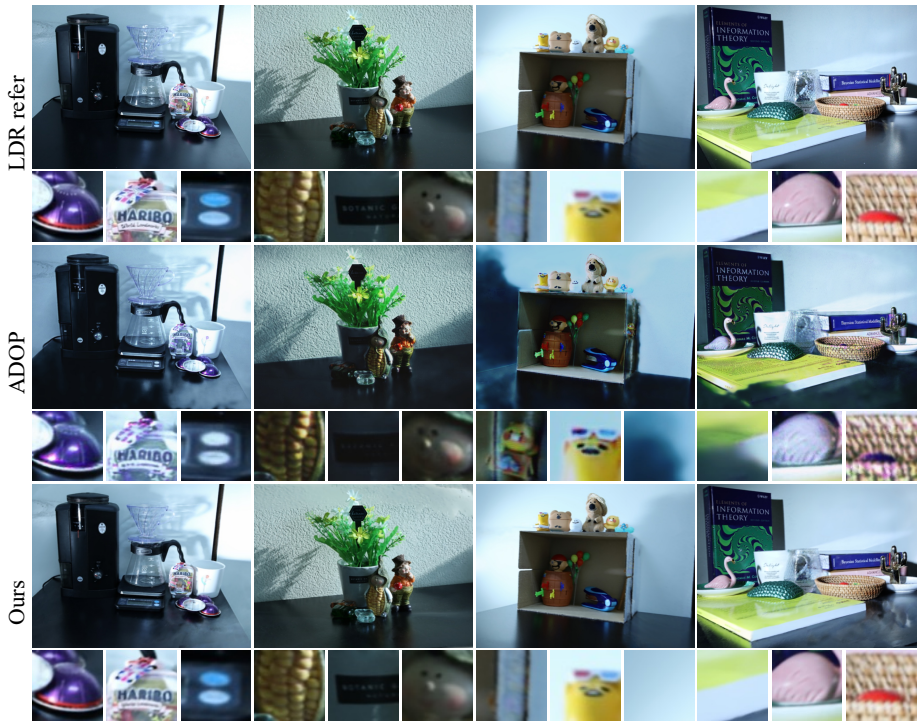

Fig. S2. Comparisons of qualitative results to baseline. The first row represents the LDR reference image, which is used at training. All experiments are trained with varying conditioned data and rendered with the tone mapping stage. Our results represent fine-grained rendering results compared to the ADOP baseline. Especially, ours shows satisfying color representations compared to ADOP.

we train the ADOP model for 100 epochs, which are enough to show the model performance on the novel view synthesis. We use the mask covering the right half side of the images for dense reconstruction and model optimization, following NeRF-W [6].

B Novel View Synthesis of Real Scenes

To evaluate our method, we compare against novel view synthesis baseline, which handles the varying appearance of images. We present several experimental results to verify the effectiveness of our method in qualitative and quantitative views of real scenes.

B.1 Qualitative Results

We compare our HDR-Plenoxels with original Plenoxels in different camera settings, and its qualitative results are in Fig. S1. The results of original Plenoxels with static camera condition located in the first row, represent our upper bound performance of novel view synthesis. As described in (Sec. A.2), the right half of the test image is unseen data and meaning novel view synthesis.

Table S1. Quantitative results of novel view synthesis on real scenes. $\mathcal{S}$ denote the static and $\mathcal{V}$ is the varying datasets. The blue and red color stand for the **best** and the **second best**, respectively. We report the averaged results of all the views in each test data. Our method shows the highest or the second-best performance compared to other models.

Type	Method	Character			Desk			Plant			Coffee		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
$\mathcal{S}$	Baseline	32.40	0.955	0.278	25.53	0.895	0.303	24.58	0.833	0.324	25.87	0.922	0.301
$\mathcal{V}$	Baseline	**19.13**	0.762	0.526	**13.75**	**0.553**	**0.518**	**21.29**	**0.623**	0.511	17.49	0.751	0.476
	ADOP	17.56	**0.801**	**0.114**	10.298	0.390	**0.518**	18.26	0.529	**0.192**	**18.44**	**0.822**	**0.085**
	Ours	**33.14**	**0.960**	**0.343**	**28.32**	**0.907**	**0.312**	**24.27**	**0.790**	**0.369**	**27.40**	**0.928**	**0.269**

In the second row, original Plenoxels with varying camera settings show poor renderings results, especially in right half, where we split as test image.

We also compare the qualitative results with baseline models, ADOP, and our HDR-Plenoxels, as shown in Fig. S2. Both ADOP and our results represent comparable novel view synthesis with predicting fine-detailed 3D geometry. However, ADOP shows biased results in estimating satisfying color and shadows compared to ours. If a hole occurs during the point-cloud generation, the reconstruction result also shows vacancy in the rendered result because ADOP is a point-cloud-based rendering model. The training stage of ADOP is unstable if they are in local optima, resulting in the imperfect color of novel view synthesis. In contrast, our HDR-Plenoxels successfully reconstruct 3D real scenes with achieving a highly favorable tone-mapping stage. Due to the properties of SH, which regularize complex color information on a few bases, we can optimize the model fast and stable.

B.2 Quantitative Results

We compare our HDR-Plenoxesl with other methods, consisting of the original Plenoxels (denoted as Baseline) in both static and varying conditioned data and ADOP in the varying data, and its quantitative results are in Table S1.

ADOP performs the gamma correction ($\gamma = 1/2.2$) as default by design, not the learned CRF function; thus, we first linearized the image by applying inverse gamma correction and then learned CRF. ADOP is based on a dense reconstructed point-cloud, which can recover structurally detailed scenes. However, ADOP shows low performance in inferring the overall white balance and changing colors, including shading, which leads to low scores on SSIM and PSNR. In contrast, our HDR-Plenoxels show overall high performance in all three metrics demonstrating that ours can understand physically appropriate tone-mapping part and 3D structure as well.

C Controllable Rendering at Novel View Synthesis

This section represents controllable rendering results with arbitrary exposure, white balance, and CRF settings. Our tone mapping module consists of two stages, *i.e.*, white balance and CRF. Each stage is designed following the physical properties and represented by an explicit function to change the values of each

Fig. S3. Novel view synthesis results with varying exposure rendering. The exposure condition changes from dark to bright from left to right. The novel view rendering result of the middle column has a basis exposure value.

Fig. S4. Novel view synthesis results with varying white balance rendering. The white balance changes *w.r.t.* red, green, and blue, in the results of the second, third, and last columns, respectively. The novel view rendering result of the first column has a basis white balance value.

stage. To eliminate the ambiguity between exposure and white balance, we apply a white balance module with a scale suggested by Kim *et al.* [5]. In our white balance parameters, the exposure value is represented by a scale of white balance, which enables us to control exposure value as well.

C.1 Exposure and White Balance

We show controllable rendering results of arbitrary exposure in Fig. S3. To change the exposure value, we set the basis exposure value by globally averaging the white balance values of full view. With scaling basis exposure value, we can control the exposure of novel view rendering.

To control the white balance, we set the basis white balance by channel-wise averaging the white balance values of full view. By changing respective red, green, and blue components on the basis of the white balance, we can control the white balance of novel view rendering, as shown in Fig. S4. This controllable rendering allows us to get the most advantages of synthesizing novel views with HDR, enabling editing HDR images and video in novel views through freely controllable rendering.

C.2 Camera Response Function

To verify the ability of our CRF module, we train our HDR-Plenoxels with images of two different CRF rendering the same scene and transfer each CRF. After transferring each CRF, the novel view rendering results show distinct rendering style, as shown in Fig. S5. The deliberate comparisons of CRF show the shape difference between filmic and standard CRF according to each RGB color channel. The results of transferred CRF rendering imply that HDR-Prenoxels can learn robustly even in various CRFs. Also we can apply diverse CRF to en-

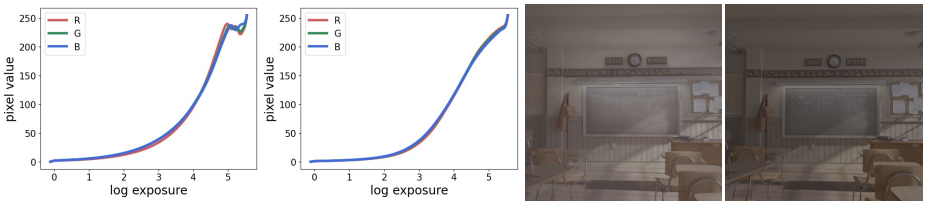

Fig. S5. Novel view synthesis changing with CRF. The first two plots show CRF learned from images modified with two different view transforms implemented in Blender (Standard and Filmic). Just by exchanging learned CRFs, we can render the same synthetic scene in different styles.

able various rendering styles and more free HDR image and video editing.extreme case

C.3 Comparison between HDR-Plenoxels and NeRF in the Wild

In our experiments, we use NeRF-A (appearance), which means without transient part. For controllable rendering, NeRF-A interpolates their appearance between each view, which has ambiguity in rendering results and cannot control explicitly. In contrast, HDR-Plenoxels uses a tone-mapping module based on explicit functions and can control LDR rendering with quantified value input. Our synthetic dataset contains three different exposures, and we conduct controllable rendering, which reconstructs median exposure given brighter and darker values or embeddings. In our HDR-Plenoxels, we get minimum and maximum value of exposure after training and get median value for rendering median exposure. In NeRF-A, we interpolate between appearances embeddings which are assigned to brighter and darker images, respectively. We measure quantitative quality between them with median exposure ground truth images. As shown in Table S2, ours can control radiometric calibration more accurately and also get precise geometry.

Table S2. Controllable rendering comparison at a classroom image.

Ours			NeRF-A		
PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
33.08	0.951	0.154	23.52	0.907	0.247

D Additional Experiments

D.1 Denoising Effects

We build our synthetic datasets using the Blender [2], which can render with or without the shot noise. To verify the robustness of our model under the such noise, we compare the PSNR performance according to the presence of the shot noise. Our model marks 29.53 and 31.58 in PSNR, for kitchen data with and without shot noise,

Fig. S6. Denoising effect of HDR-Plenoxels.

respectively. Although shot noise degrades the numerical performance slightly (middle), it represents similar qualitative results to the model trained on denoised images (right), as shown in Fig. S6. The left one is the shot noise input used to train the middle result. As our model aggregates multi-view information, it somewhat shows a denoising effect.

D.2 Extreme Camera Conditions

Our original exposure setting has three levels in the ±3EV range. However, empirically, our model can robustly learn even under harsher exposure conditions. For more extreme cases, *e.g.*, very dark or bright, we train and evaluate on our kitchen data with five exposure levels in respective ±4EV, ±5EV, and ±6EV ranges. For ±3EV, ±4EV, ±5EV, ±6EV cases, ours obtains 31.58, 30.30, 29.10, 28.48 in PSNR, respectively. Although PSNR steadily decreases as the exposure gap becomes wider, ours at the most extreme setting obtains higher PSNR than ADOP in the original setting (20.13). Even in extreme conditions, our model shows high-quality HDR novel view synthesis (left) result with +6EV input LDR image (right) as shown in Fig. S6. Our method is robust in various exposure settings, even in extreme cases.

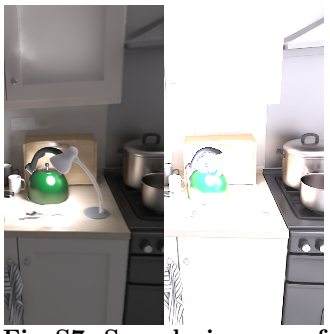

Fig. S7. Sample images of extreme exposure.

D.3 Generality of the Tone Mapping Module

To verify the generality of the tone mapping module, we apply our tone mapping module to vanilla NeRF [7]. We trained vanilla NeRF on our kitchen dataset. Vanilla NeRF trained on images from varying cameras results in blurry and foggy images (middle). NeRF with our tone-mapping module (right) shows clear novel view rendering result similar to a model trained static camera setting (left). Our tone mapping function enabled vanilla NeRF to learn radiance fields from varying cameras robustly. As tone-mapping module is computationally light and easily attachable after the ray-marching; it can generally be employed in the various volume rendering models.

Fig. S8. NeRF with our tone mapping module.

References

1. Bergonzini, C.: Lone monk. https://www.blender.org/download/demo-files/ 2
2. Community, B.O.: Blender - a 3d modelling and rendering package. http://www.blender.org (2018) 2, 7
3. Gyzen, K.: Cozy room. https://www.turbosquid.com/ko/3d-models/cozy-room-3d-model-1641507 2
4. Jay-Artist: Country-kitchen cycles. https://www.blendswap.com/blend/5156 2
5. Kim, S.J., Lin, H.T., Lu, Z., Süsstrunk, S., Lin, S., Brown, M.S.: A New In-Camera Imaging Model for Color Computer Vision and Its Application. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **34**(12) (2012) 6
6. Martin-Brualla, R., Radwan, N., Sajjadi, M.S.M., Barron, J.T., Dosovitskiy, A., Duckworth, D.: NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2021) 3, 4
7. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European Conference on Computer Vision (ECCV) (2020) 8
8. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems (NeurIPS) (2019) 2
9. Quei-An, C.: Nerf_pl: a pytorch-lightning implementation of nerf. https://github.com/kwea123/nerf_pl/ (2020) 3
10. Rückert, D., Franke, L., Stamminger, M.: Adop: Approximate differentiable one-pixel point rendering. https://github.com/darglein/ADOP (2021) 3
11. Rückert, D., Franke, L., Stamminger, M.: ADOP: Approximate Differentiable One-Pixel Point Rendering. arXiv:2110.06635 (2021) 3
12. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016) 3
13. Schönberger, J.L., Zheng, E., Pollefeys, M., Frahm, J.M.: Pixelwise view selection for unstructured multi-view stereo. In: European Conference on Computer Vision (ECCV) (2016) 3
14. Seux, C.: Classroom. https://www.blender.org/download/demo-files/ 2
15. Yu, A., Fridovich-Keil, S., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance Fields without Neural Networks. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2022) 2