Supplementary: Light Field Neural Rendering

Mohammed Suhail[1,2*] Carlos Esteves[4] Leonid Sigal[1,2,3] Ameesh Makadia[4]
suhail33@cs.ubc.ca machc@google.com lsigal@cs.ubc.ca makadia@google.com
[1]University of British Columbia [2]Vector Institute for AI [3]Canada CIFAR AI Chair [4]Google

A. Additional implementation details

A.1. MLP architecture in ablation

We detail the architecture of the '1-MLP' model introduced in Section 4.3. We present the detailed architecture in Tab. A.1. We use a series of `DenseGeneral(DG)` layers and a final `Dense` available in Flax [3]. The architecture was determined by running a sweep over various depths. We found that further increase in model capacity leads to poor generalization.

Layer	Input Dimension	Output Dimension
DG(F, 256)	B x N x P x F	B x N x P x 256
DG_{12}(256, 256)	B x N x P x 256	B x N x P x 256
DG(P, 1)	B x N x P x 256	B x N x 256
DG(N, 1)	B x N x 256	B x 256
Dense(256, 1)	B x 256	B x 3

Table A.1. **1-MLP Architecture.** We use notations B for batch size, N for number of reference views, P for number of epipolar projection and F for feature dimension. DG_{12} represents 12 layers of `DenseGeneral`. We also add skip connections at every fourth layer in DG_{12}. The final output corresponds to the predicted color.

A.2. Spherical light field encoding

For 360° scenes, we use the two-sphere light field parametrization [2]. Each ray is represented by two points on the sphere, by the 4D tuple $(\theta_1, \phi_1, \theta_2, \phi_2)$. To encode this representation we found advantageous to use the spherical harmonics basis instead of the sinusoidals. For a given ray, we evaluate a number of spherical harmonics at each intersection and concatenate them to obtain its encoding,

$$\tilde{Y}^{\ell}_{m}(\theta_1, \phi_1, \theta_2, \phi_2) = \left[Y^{\ell}_{m}(\theta_1, \phi_1) \parallel Y^{\ell}_{m}(\theta_2, \phi_2)\right], \quad (1)$$

where $Y^{\ell}_{m}(\theta, \phi)$ denotes the spherical harmonics of degree ℓ and order m evaluated at (θ, ϕ). In our experiments, we concatenate all the zonal and sectoral harmonics ($m = 0$ and $m = \ell$) upto a maximum degree of 4.

Model	PSNR ↑	SSIM ↑	LPIPS ↓	Avg. ↓
$\text{Ours}_{\text{P.E.}}$	33.18	0.979	0.027	0.0123
Ours	33.85	0.981	0.024	0.0110

Table A.2. Light field encoding ablation on Blender dataset.

To demonstrate the efficacy of the spherical harmonics encoding we conduct an ablation on the blender dataset where we replace the spherical encoding with the regular positional encoding in NeRF [5]. We refer to this model as $\text{Ours}_{\text{P.E.}}$. We report the average metric on the blender dataset for this ablation in Tab. A.2.

A.3. Metric computation

To compute the SSIM metric we use the function available in scikit-image package. To compute the LPIPS on forward facing scene (RFF and Shiny), similar to NeX, we use the VGG model[†] from [7]. On the blender scenes, similar to Mip-NeRF, we use the VGG model available in tensorflow hub to compute LPIPS. We use two different implementation on LPIPS to ensure fairness of comparison.

B. Additional results

B.1. Real-forward-facing dataset (RFF)

The RFF dataset introduced by Mildenhall et al. [4] consists of 8 forward facing captures with each scene consisting of around 20 to 62 images. We present the scene-wise breakdown of the results in Table B.3. The metrics for NeRF and NeX are the ones reported in NeX [6].

B.2. Shiny dataset

The Shiny dataset introduced in NeX [6] presents 8 scenes with challenging view dependent effects, captured by forward-facing cameras. We present the scene-wise breakdown of the results in Tab. B.4. The metrics for NeRF and NeX are the ones reported in NeX [6].

[†]We use the library provided by https://github.com/richzhang/PerceptualSimilarity.

	PSNR			SSIM			LPIPS		
Model	NeRF	NeX	Ours	NeRF	NeX	Ours	NeRF	NeX	Ours
Fern	25.49	25.63	24.86	0.866	0.887	0.886	0.278	0.205	0.135
Flower	27.54	28.90	29.82	0.906	0.933	0.939	0.212	0.150	0.107
Fortress	31.34	31.67	33.22	0.941	0.952	0.964	0.166	0.131	0.119
Horns	28.02	28.46	29.78	0.915	0.934	0.957	0.258	0.173	0.121
Leaves	21.34	21.96	22.47	0.782	0.832	0.856	0.308	0.173	0.110
Orchids	20.67	20.42	21.05	0.755	0.765	0.807	0.312	0.242	0.173
Room	32.25	32.32	34.54	0.972	0.975	0.987	0.196	0.161	0.104
Trex	27.36	28.73	30.34	0.929	0.953	0.968	0.234	0.192	0.143

Table B.3. Scene-wise breakdown of quantitative results on the Real Forward-Facing dataset.

	PSNR			SSIM			LPIPS		
Model	NeRF	NeX	Ours	NeRF	NeX	Ours	NeRF	NeX	Ours
CD	30.14	31.43	35.25	0.093	0.958	0.989	0.206	0.129	0.041
Tools	27.45	28.16	26.55	0.938	0.953	0.945	0.204	0.151	0.130
Crest	20.30	21.23	21.73	0.670	0.757	0.797	0.315	0.162	0.079
Seasoning	27.79	28.60	28.34	0.898	0.928	0.936	0.276	0.168	0.102
Food	23.32	23.68	22.88	0.796	0.832	0.821	0.308	0.203	0.151
Giants	24.86	26.00	27.06	0.844	0.898	0.928	0.270	0.147	0.065
Lab	29.60	30.43	35.28	0.936	0.949	0.989	0.182	0.146	0.066
Pasta	21.23	22.07	21.63	0.789	0.844	0.855	0.311	0.211	0.096

Table B.4. Scene-wise breakdown of quantitative results on the Shiny dataset.

B.3. Blender dataset

The Blender dataset introduced by Mildenhall et al. [5] consists of 8 scenes each containing 800×800 resolution images rendered from viewpoints randomly sampled on a hemisphere around the object. We present the scene-wise breakdown of the results in Tab. B.5. The metrics for NeRF and Mip-NeRF were obtained from Mip-NeRF [1].

C. Additional experiments and visualizations

C.1. Plücker coordinates

Our experiments use ray parametrizations specific to the camera configuration of each type of scene. For forward facing scenes, we employ the light slab light field representation, while for 360° scenes we use the two-sphere. In this section, we explore the alternative of using Plücker coordinates, which are generic and can represent any kind of camera configuration. Since our architecture is agnostic to the light field parametrization, we simply replace the input ray representation with the 6D Plücker coordinates to perform this experiment. When using Plücker coordinates, we observe a drop of 0.18 dB PSNR on the RFF dataset as compared to the light slab representation. Similarly, we observe a drop of 0.25 dB PSNR on the Blender dataset when replacing the two-sphere parametrization with Plücker coordinates. This suggests that for particular configurations, the specific (and lower dimensional) ray parametrizations have a slight advantage over a generic parametrization such as Plücker coordinates.

C.2. Handling view-dependent effects

While our model is built around geometric constraints (such as epipolar geometry), the attention-based modeling provides the capability to downweigh such constrains when not useful, in order to more directly associate a color to ray coordinates (as in the Vanilla-NLF model described in Section 4.3). We speculate that this, together with the convolutional features (which bring some context) explains our superior performance on view-dependent effects.

We run a mini-ablation to investigate this hypothesis. Figure C.1 shows, for the Lab scene from the Shiny dataset, one crop with transparency/refraction and another that is diffuse and contains sharp details. Our model works well on both regions which indicates its flexibility, in contrast

Model	PSNR			SSIM			LPIPS		
	NeRF	Mip-NeRF	Ours	NeRF	Mip-NeRF	Ours	NeRF	Mip-NeRF	Ours
Chair	34.08	35.14	35.30	0.975	0.981	0.989	0.026	0.021	0.012
Drums	25.03	25.48	25.83	0.925	0.932	0.955	0.071	0.065	0.045
Ficus	30.43	33.29	33.38	0.967	0.980	0.987	0.032	0.020	0.010
Hotdog	36.92	37.48	38.66	0.979	0.982	0.993	0.030	0.027	0.009
Lego	33.28	35.7	35.76	0.968	0.978	0.989	0.031	0.021	0.010
Materials	29.91	30.71	35.10	0.953	0.959	0.990	0.047	0.040	0.011
Mic	34.53	36.51	35.32	0.987	0.991	0.992	0.012	0.009	0.008
Ship	29.36	30.41	30.94	0.869	0.882	0.952	0.150	0.138	0.084

Table B.5. Scene-wise breakdown of quantitative results on the Blender dataset.

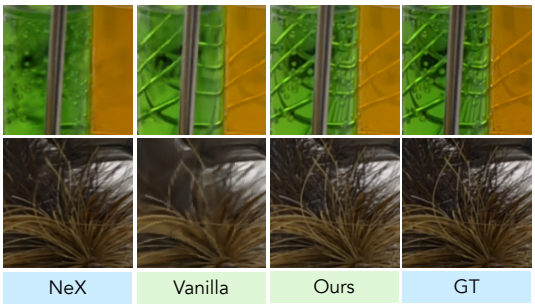

Figure C.1. Crops from two different regions of the *Lab* scene in the Shiny dataset. The Vanilla-NLF model (described in Section 4.3) is able to retrieve a majority of the refraction details but fails to reproduce high frequencies. NeX reproduces sharp details but not the refractions. Our model does well in both regions.

with the baselines.

C.3. Visualizing view attention

The attention weights β^j (in Eq. 3) correspond to "importance" of each reference view when rendering a target pixel. We visualize these attention weight for a test image in the chair scene from Blender dataset in Fig. C.2. We explain the visualization process in the figure caption.

References

[1] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 2

[2] E. Camahort, A. Lerios, and D. Fussell. Uniformly sampled light fields. In *Eurographics Workshop on Rendering Techniques*, pages 117–130, 1998. 1

[3] J. Heek, A. Levskaya, A. Oliver, M. Ritter, B. Rondepierre, A. Steiner, and M. van Zee. Flax: A neural network library and ecosystem for JAX, 2020. URL http://github.com/google/flax. 1

[4] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 1

[5] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, pages 405–421, 2020. 1, 2

[6] S. Wizadwongsa, P. Phongthawee, J. Yenphraphai, and S. Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8534–8543, 2021. 1

[7] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

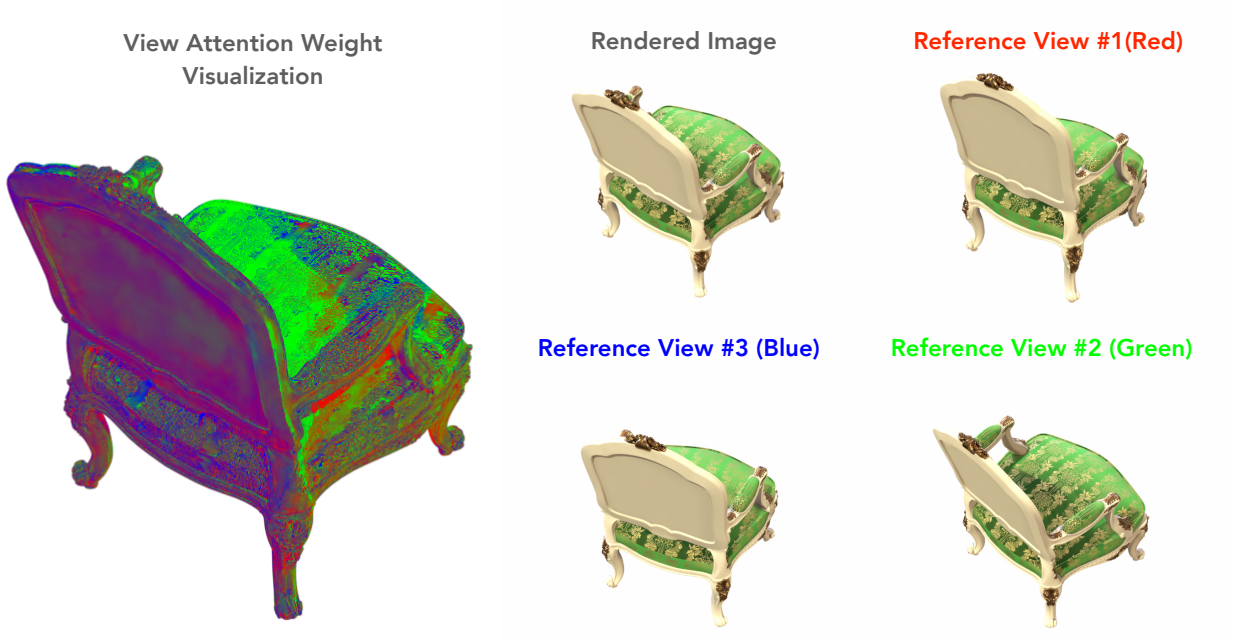

Figure C.2. **View attention weight visualization.** We visualize the attention weights β^j for each rendered pixel for a test image from the chair scene. For each target pixel, we consider three reference views. Thus we have three attention weights β^1, β^2 and β^3 corresponding to reference views 1, 2 and 3 respectively. We treat these attention weights as RGB value and visualize them as an image as shown above. Intuitively, this image shows the contribution of each reference view when rendering a pixel. For example, the cushion is predominantly green as it is most visible in second reference view. Similarly the back of the chair contains almost equal mix of red and blue as it is equally visible in reference views 1 and 3. We do not show the attention weights for the background pixels for clarity of visualization.