NeX: Real-time View Synthesis with Neural Basis Expansion Supplementary Material

A. Additional Implementation Settings

A.1. Image Preparation Details

We calibrate a set of input images using an a Structure-from-Motion (SfM) algorithm in an open-source software package COLMAP [4]. For COLMAP, we use a "simple radial" camera model with a single radial distortion coefficient and a shared intrinsic for all images. We use "sift feature guided matching" option in the exhaustive matcher step of SfM and also refine principle points of the intrinsic during the bundle adjustment. Note that accurate camera poses and intrinsic parameters are crucial for our method, and errors in these parameters can lead to poor results.

A.2. Ray Sampling for Training

During training, generating a reasonable sized output image via the rendering equation for all pixels at once is not feasible due to the memory limit on our GPU. To solve this, we only sample a subset of pixels from the entire image in each iteration of the optimization. And to facilitate the computation of image gradient needed in our loss function, if a pixel (x, y) is sampled in the process, we also sample $(x+1, y)$ and $(x, y+1)$ so that the image gradients in both x and y directions can be computed through finite difference. In our implementation, we sample 2667 sets of these triplet pixels, resulting in 8001 samples.

For evaluation, we use 3 metrics: PSNR, SSIM, and LPIPS. Functions for computing PSNR and SSIM come from scikit-image software package, and for LPIPS, we use a VGG variant from [5] [1].

B. Additional Experimental Details

B.1. Comparison on Real Forward-Facing Dataset

Real Forward-Facing dataset is provided by NeRF [3] and contains 8 scenes. We show a per-scene breakdown of the results from Table 1 in the main paper in Table B.1. These scores from NeRF are computed from undistorted versions of their results using our estimated radial distortion parameter. We provided their original reported scores for reference in Table B.2. A qualitative comparison can be seen in Figure 3 in the main paper as well as in our supplementary video, which shows that our method achieves sharper fine detail.

We measured the training time on a single NVIDIA V100 with a 20-core Intel Xeon Gold 6248. For scene Fern with 17 input photos, the training took around 18 hours. For scene Flower with 30 input photos, the training took around 27 hours.

B.2. Comparison on Shiny Dataset

Our own dataset, Shiny, consists of 8 scenes with more challenging view-dependent effects compared to Real Forward-Facing dataset. Table B.3 shows the image resolution and number of images for each scene. To generate results for NeRF, we use the code implemented by the authors[2] using TensorFlow and train on each scene with their default setting for 200k iterations.

We show a per-scene breakdown of the results from Table 2 in the main paper in Table B.4. Our approach achieves better performance than NeRF on all metrics in all scenes. A full visual comparison is provided in our supplementary webpage.

B.3. Comparison on Spaces Dataset

The authors of DeepView have not made their code publicly available, but they have released their output results. So, we run our algorithm on their Spaces dataset and compare our results to theirs. Table B.5 shows a per-scene breakdown of the results from Table 3 in the main paper. A full visual comparison is provided in our supplementary webpage.

B.4. Details for Types of Basis Ablation Study

In Section 4.3.2, we evaluate our algorithm using different sets of basis functions. The experiment is done by changing the neural basis $\vec{H}_\phi$ in Algorithm 1 to other kinds of basis functions such as $\vec{H}_{FS}$, $\vec{H}_{TS}$ and $\vec{H}_{SH}$.

Our Fourier's basis is similar to the positional encoding

[1] https://github.com/richzhang/PerceptualSimilarity

[2] https://github.com/bmild/nerf

Table B.1: Per-scene breakdown results from NeRF's Real Forward-Facing dataset.

	PSNR↑				SSIM↑				LPIPS↓			
	SRN	LLFF	NeRF	Our	SRN	LLFF	NeRF	Our	SRN	LLFF	NeRF	Our
Fern	20.29	23.09	25.49	**25.63**	0.700	0.828	0.866	**0.887**	0.529	0.243	0.278	**0.205**
Flower	23.94	25.81	27.64	**28.90**	0.819	0.907	0.906	**0.933**	0.390	0.168	0.212	**0.150**
Fortress	25.70	29.56	31.34	**31.67**	0.816	0.934	0.941	**0.952**	0.494	0.171	0.166	**0.131**
Horns	23.15	25.13	28.02	**28.46**	0.801	0.905	0.915	**0.934**	0.479	0.197	0.258	**0.173**
Leaves	17.21	19.85	21.34	**21.96**	0.556	0.769	0.782	**0.832**	0.526	0.226	0.308	**0.173**
Orchids	16.97	18.73	**20.67**	20.42	0.575	0.703	0.755	**0.765**	0.528	0.308	0.312	**0.242**
Room	25.63	28.45	32.25	**32.32**	0.908	0.957	0.972	**0.975**	0.351	0.175	0.196	**0.161**
T-rex	21.71	24.67	27.36	**28.73**	0.784	0.903	0.929	**0.953**	0.412	0.204	0.234	**0.192**

Table B.2: (For reference only) Original reported scores from NeRF [3] where test images are not undistorted.

	PSNR↑			SSIM↑			LPIPS↓		
	SRN	LLFF	NeRF	SRN	LLFF	NeRF	SRN	LLFF	NeRF
Fern	21.37	21.37	25.17	0.822	0.887	0.932	0.459	0.247	0.280
Flower	24.63	25.46	27.40	0.916	0.935	0.941	0.288	0.174	0.219
Fortress	26.63	29.40	31.16	0.838	0.957	0.962	0.453	0.173	0.171
Horns	24.33	24.70	27.45	0.921	0.941	0.951	0.376	0.193	0.268
Leaves	18.24	19.52	20.92	0.822	0.877	0.904	0.440	0.216	0.316
Orchids	17.37	18.52	20.36	0.746	0.775	0.852	0.467	0.313	0.321
Room	28.42	28.42	32.70	0.950	0.978	0.978	0.240	0.155	0.178
T-rex	22.87	24.15	26.80	0.916	0.935	0.960	0.298	0.222	0.249

used in NeRF [3] and can be computed by:

$$\vec{H}_{FS}(v) = \left[\cos(2^{-1}\pi v_x),\sin(2^{-1}\pi v_x),\ldots,\cos(2^{N}\pi v_y),\sin(2^{N}\pi v_y)\right]. \tag{1}$$

For forward-facing scenarios, the viewing angle v only covers a hemi-sphere. So, v_z can be fully determined from v_x and v_y through $v_z = \sqrt{1 - v_x^2 - v_y^2}$, and we can parameterize the viewing angle with just v_x and v_y and define the FS basis only on these two parameters.

To calculate other basis functions used in Section 4.3.1, let the following complex-valued functions $K_{a,b}^{(m)}$ and $P_{a,b}^{(m)}$ be defined as:

$$K_{a,b}^{(m)}(v) = \left(\left(\frac{v_x}{1-a}\sqrt{\frac{v_z-a}{v_z+1}}\right)+\left(\frac{v_y}{1-a}\sqrt{\frac{v_z-a}{v_z+1}}\right)i\right)^m \tag{2}$$

$$P_{a,b}^{(m)}(v) = \frac{v_z-1}{1-a}+\frac{m+1}{b+2m+2} \tag{3}$$

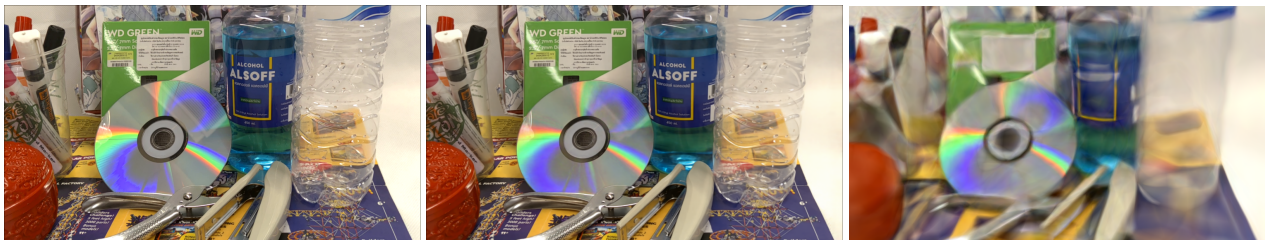
The general form of the set of basis functions is:

$$\begin{aligned}\vec{H}(v) = \Big[&\mathrm{Re}(K_{a,b}^{(2^0)}(v)),\\ &\mathrm{Im}(K_{a,b}^{(2^0)}(v)),\\ &\mathrm{Re}(P_{a,b}^{(2^0)}(v)\cdot K_{a,b}^{(2^0)}(v)),\\ &\mathrm{Im}(P_{a,b}^{(2^0)}(v)\cdot K_{a,b}^{(2^0)}(v)),\\ &\ldots,\\ &\mathrm{Re}(K_{a,b}^{(2^N)}(v)),\\ &\mathrm{Im}(K_{a,b}^{(2^N)}(v)),\\ &\mathrm{Re}(P_{a,b}^{(2^N)}(v)\cdot K_{a,b}^{(2^N)}(v)),\\ &\mathrm{Im}(P_{a,b}^{(2^N)}(v)\cdot K_{a,b}^{(2^N)}(v))\Big]\end{aligned} \tag{4}$$

where if $a = -1, b = 0$, then it reduces to the spherical harmonics basis (SH). If $a = 0, b = 0$, then it reduces to the hemispherical harmonics basis (HSH) [1]. For Jacobi basis

Ground truth Nex (Ours) NeRF

Figure B.1: A qualitative comparison on Shiny dataset between ground truth(left), NeX (center), and NeRF[3] (right). A full comparison on all scenes can be found in our supplementary webpage



Ground truth Ours NSVF

Figure B.2: A qualitative comparison on scene CD between ground truth (left), NeX (center), and NSVF [2] (right). We use NSVF code open-sourced by the authors[3]. NSVF does not perform well for this problem setup because it focuses on object captures where a bounding volume can be tightly defined.

(JH), we set $a = \cos(45°) = 1/\sqrt{2}$ and $b = 2$.

Here are examples of the first five terms for each basis that we use in Section 4.3.1:

$$\begin{aligned}
\vec{H}_{SH}(v) =& [v_x/2, v_y/2, v_z v_x/2, v_z v_y/2, (v_x^2 - v_y^2)/4, \ldots] \\
\vec{H}_{HSH}(v) =& [v_x \sqrt{\tfrac{v_z}{v_z+1}}, v_y \sqrt{\tfrac{v_z}{v_z+1}}, v_x \tfrac{2v_z-1}{2} \sqrt{\tfrac{v_z}{v_z+1}}, \\
& v_y \tfrac{2v_z-1}{2} \sqrt{\tfrac{v_z}{v_z+1}}, \tfrac{v_x^2 - v_y^2}{(1-a)^2} \tfrac{v_z}{v_z-1}, \ldots] \\
\vec{H}_{JH}(v) =& [v_x \sqrt{\tfrac{v_z-a}{v_z+1}}, v_y \sqrt{\tfrac{v_z-a}{v_z+1}}, v_x \left(\tfrac{v_z-1}{z-a} + \tfrac{2}{b+4}\right) \sqrt{\tfrac{v_z}{v_z+1}}, \\
& v_y \left(\tfrac{v_z-1}{z-a} + \tfrac{2}{b+4}\right) \sqrt{\tfrac{v_z}{v_z+1}}, \tfrac{v_x^2 - v_y^2}{(1-a)^2} \tfrac{v_z}{v_z-1}, \ldots]
\end{aligned} \tag{5}$$

Figure 5 in the main paper already shows PSNR scores of these basis functions. SSIM and LPIPS scores from the same experiment are shown in figure B.3 and B.4 respectively.

References

[1] Pascal Gautron, Jaroslav Krivanek, Sumanta Pattanaik, and Kadi Bouatouch. A Novel Hemispherical Basis for Accurate and Efficient Rendering. 2004. 2

[2] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *arXiv preprint arXiv:2007.11571*, 2020. 3

[3] 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. *arXiv preprint arXiv:2003.08934*, 2020. 1, 2, 3

[4] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[5] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman,

and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 1

Table B.3: Image resolution and the number of images for each scene in our Shiny dataset. For most scenes, only 20-50 images are enough to produce good results. However, scenes with complex view-dependent effects like CD require more images.

	image resolution	number of images
CD	1008×567	307
Tools	1008×756	58
Crest	1008×756	50
Seasoning	1008×756	45
Food	1008×756	49
Giants	1008×756	32
Lab	1008×567	303
Pasta	1008×756	35

Table B.4: Per-scene breakdown results on our Shiny dataset.

	PSNR↑		SSIM↑		LPIPS↓	
	NeRF	Ours	NeRF	Ours	NeRF	Ours
CD	30.14	**31.43**	0.937	**0.958**	0.206	**0.129**
Tools	27.54	**28.16**	0.938	**0.953**	0.204	**0.151**
Crest	20.30	**21.23**	0.670	**0.757**	0.315	**0.162**
Seasoning	27.79	**28.60**	0.898	**0.928**	0.276	**0.168**
Food	23.32	**23.68**	0.796	**0.832**	0.308	**0.203**
Giants	24.86	**26.00**	0.844	**0.898**	0.270	**0.147**
Lab	29.60	**30.43**	0.936	**0.949**	0.182	**0.146**
Pasta	21.23	**22.07**	0.789	**0.844**	0.311	**0.211**

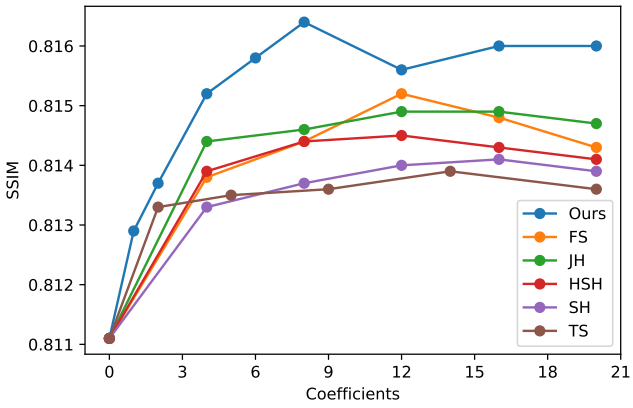


Figure B.3: Number of coefficients versus SSIM score (higher is better)

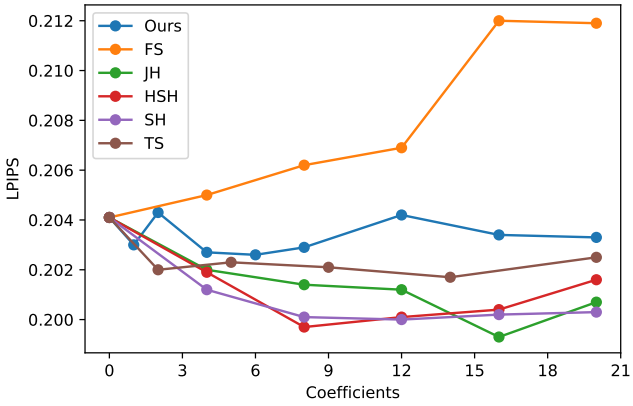


Figure B.4: Number of coefficients versus LPIPS score (lower is better)

Table B.5: Per-scene breakdown results on Spaces dataset.

	PSNR↑			SSIM↑			LPIPS↓		
	Soft3D	DeepView	Ours	Soft3D	DeepView	Ours	Soft3D	DeepView	Ours
scene000	32.66	32.54	**37.61**	0.971	0.983	**0.989**	0.093	0.059	**0.049**
scene009	31.46	31.07	**35.40**	0.962	0.972	**0.981**	0.123	0.091	**0.080**
scene010	32.94	31.22	**37.61**	0.973	0.979	**0.989**	0.137	0.095	**0.095**
scene023	31.52	31.14	**35.69**	0.969	0.978	**0.986**	0.142	0.102	**0.098**
scene024	33.88	33.15	**37.77**	0.978	0.983	**0.989**	0.119	**0.081**	0.090
scene052	30.08	30.22	**34.02**	0.947	0.971	**0.979**	0.119	0.081	**0.076**
scene056	30.64	31.04	**34.77**	0.956	0.975	**0.981**	0.141	0.087	**0.087**
scene062	32.56	32.07	**35.34**	0.969	0.980	**0.984**	0.151	**0.098**	0.121
scene063	29.72	32.72	**35.44**	0.952	0.979	**0.987**	0.122	0.078	**0.073**
scene073	30.28	30.85	**34.81**	0.960	0.977	**0.986**	0.111	0.073	**0.065**