Putting NeRF on a Diet: Semantically Consistent Few-Shot View Synthesis Supplementary Materials

Ajay Jain
UC Berkeley
ajayj@berkeley.edu

Matthew Tancik
UC Berkeley
tancik@berkeley.edu

Pieter Abbeel
UC Berkeley
pabbeel@cs.berkeley.edu

A. Experimental details

View selection For most few-view Realistic Synthetic experiments, we randomly subsample 8 of the available 100 training renders. Views are not manually selected. However, to compare the ability of NeRF and DietNeRF to extrapolate to unseen regions, we manually selected 14 of the 100 views mostly showing the right side of the Lego scene. For DTU experiments where we fine-tune pixelNeRF [12], we use the same source view as [12]. This viewpoint was manually selected and is shared across all 15 scenes.

Simplified NeRF baseline The published version of NeRF [6] can be unstable to train with 8 views, often converging to a degenerate solution. We found that NeRF is sensitive to MLP parameter initialization, as well as hyperparameters that control the complexity of the learned scene representation. For a fair comparison, we tuned the Simplified NeRF baseline on each Realistic Synthetic scene by modifying hyperparameters until object geometry converged. Table 1 shows the resulting hyperparameter settings for initial learning rate prior to decay, whether the MLP f_θ is viewpoint dependent, number of samples per ray queried from the fine and coarse networks, and the maximum frequency sinusoidal encoding of spatial position (x, y, z). The fine and coarse networks are used in [6] for hierarchical sampling. ✗ denotes that we do not use the fine network.

Implementation Our implementation is based on a PyTorch port [11] of NeRF's original Tensorflow code. We re-train and evaluate NeRF using this code. For memory efficiency, we use 400×400 images of the scenes as in [11] rather than full-resolution 800×800 images. NV is trained with full-resolution 800×800 views. NV renderings are downsampled with a 2x2 box filter to 400×400 to compute metrics. We train all NeRF, Simplified NeRF and DietNeRF models with the Adam optimizer [4] for 200k iterations.

Metrics Our PSNR, SSIM, and LPIPS metrics use the same implementation as [12] based on the scikit-image Python package [10]. For the DTU dataset, [12] excluded some poses from the validation set as ground truth photographs had excessive shadows due to the physical capture setup. We use the same subset of validation views.

Table 1. **Simplified NeRF training details** by scene in the Realistic Synthetic dataset. We tune the initial learning rate, view dependence, number of samples from fine and coarse networks for hierarchical sampling, and the maximum frequency of the (x, y, z) spatial positional encoding.

Scene	LR	View dep.	Fine	Coarse	Max freq.
Full NeRF	5×10^{-4}	✓	128	64	2^9
Lego	5×10^{-5}	✓	✗	128	2^5
Chair	5×10^{-5}	✗	✗	128	2^5
Drums	5×10^{-5}	✗	✗	128	2^5
Ficus	5×10^{-5}	✗	✗	128	2^5
Mic	5×10^{-5}	✗	✗	128	2^5
Ship	5×10^{-5}	✗	✗	128	2^5
Materials	1×10^{-5}	✗	✗	128	2^5
Hotdog	1×10^{-5}	✗	✗	128	2^3

For both Realistic Synthetic and DTU scenes, we also included FID and KID perceptual image quality metrics. While PSNR, SSIM and LPIPS are measured between pairs of pixel-aligned images, FID and KID are measured between two sets of image samples. These metrics compare the *distribution* of image features computed on one set of images to those computed on another set. As distributions are compared rather than individual images, a sufficiently large sample size is needed. For the Realistic Synthetic dataset, we compute the FID and KID between all 3200 ground-truth images (across train, validation and testing splits and across scenes), and 200 rendered test images at the same resolution (25 test views per scene). Aggregating across scenes allows us to have a larger sample size. Due to the setup of the Neural Volumes code, we use additional samples for rendered images for that baseline. For the DTU dataset, we compute FID and KID between 720 rendered images (48 per scene across 15 validation scenes, excluding the view-

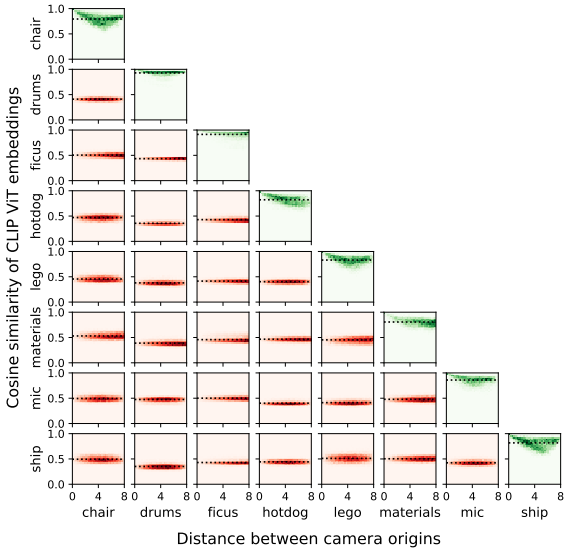

Figure 1. **CLIP ViT embeddings are more similar between views of the same scene than across different scenes.** We show a 2D histogram for each pair of Realistic Synthetic scenes comparing ViT embedding similarity and the distance between views. The dashed line shows mean cosine similarity, and green histograms have mean similarity is greater than 0.6. On the diagonal, two views from the upper hemisphere of the same scene are sampled. Embeddings of different views of the same scene are generally highly similar. Nearby (distance 0) and diagonally opposing (distance 8) views are most similar. In comparison, when sampling views from different scenes (lower triangle), embeddings are dissimilar.

point of the source image provided to pixelNeRF) and 6076 ground-truth images (49 images including the source viewpoint across 124 training and validation scenes). FID and KID metrics are computed using the `torch-fidelity` Python package [7].

B. Per-scene metrics

Embedding similarity In Figure 1, we compare the cosine similarity of two views with the distance between their camera origins for each pair of scenes in the Realistic Synthetic dataset. When sampling both views from the same scene, views have high cosine similarity (diagonal). For 6 of the 8 scenes, there is some dependence on the relative poses of the camera views, though similarity is high across all camera distances. For views sampled from different scenes, similarity is low (cosine similarity around 0.5).

Quality metrics Table 2 shows PSNR, SSIM and LPIPS metrics on a per-scene basis for the Realistic Synthetic dataset. FID and KID metrics are excluded as they need a larger sample size. We bold the best method on each scene, and underline the second-best method. Across all scenes in the few-shot setting, DietNeRF or DietNeRF fine-tuned for 50k iterations with $\mathcal{L}_{\text{MSE}}$ performs best or second-best.

C. Qualitative results and ground-truth

In this section, we provide additional qualitative results. Figure 2 shows the ground-truth training views used for 8-shot Realistic Synthetic experiments. These views are sampled at random from the training set of [6]. Random sampling models challenges with real-world data capture such as uneven view sampling. It may be possible to improve results if views are carefully selected.

In Figure 3, we provide additional renderings of Realistic Synthetic scenes from testing poses for baseline methods and DietNeRF. Neural Volumes generally converges to recover coarse object geometry, but has wispy artifacts and distortions. On the Ship scene, Neural Volumes only recovers very low-frequency detail. Simplified NeRF suffers from occluders that are not visible from the 8 training poses. DietNeRF has the highest quality reconstructions without these distortions or occluders, but does miss some high-frequency detail. An interesting artifact is the leakage of green coloration to the back of the chair.

Finally, in Figure 4, we show renderings from pixelNeRF and DietPixelNeRF on all DTU dataset validation scenes not included in the main paper. Starting from the same checkpoint, pixelNeRF is fine-tuned using $\mathcal{L}_{\text{MSE}}$ for 20k iterations, whereas DietPixelNeRF is fine-tuned using $\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{SC}}$ for 20k iterations. DietPixelNeRF has sharper renderings. On scenes with rectangular objects like bricks and boxes, DietPixelNeRF performs especially well. However, the method struggles to preserve accurate geometry in some cases. Note that the problem is under-determined as only a single view is observed per scene.

D. Adversarial approaches

While NeRF is only supervised from observed poses, conceptually, a GAN [2] uses a discriminator to compute a realism loss between real and generated images that need not align pixel-wise. Patch GAN discriminators were introduced for image translation problems [3, 13] and can be useful for high-resolution image generation [1]. SinGAN [8] trains multiscale patch discriminators on a single image, comparable to our single-scene few-view setting. In early experiments, we trained patch-wise discriminators per-scene to supervise f_θ from novel poses in addition to $\mathcal{L}_{\text{SC}}$. However, an auxiliary adversarial loss led to artifacts on Realistic Synthetic scenes, both in isolation and in combination with our semantic consistency loss. SinGAN often pasted the same image multiple times across views, with realistic textures but implausible geometry. On the Lego scene, adding a SinGAN-style loss to NeRF led to 17.90 PSNR, while

Table 2. **Quality metrics for each scene in the Realistic Synthetic dataset with 8 observed views.**

PSNR ↑	**Lego**	**Chair**	**Drums**	**Ficus**	**Mic**	**Ship**	**Materials**	**Hotdog**
NeRF	9.726	21.049	17.472	13.728	26.287	12.929	7.837	10.446
NV [5]	17.652	20.515	16.271	19.448	18.323	14.457	16.846	19.361
Simplified NeRF	16.735	21.870	15.021	**21.091**	24.206	17.092	20.659	24.060
DietNeRF (ours)	23.897	24.633	**20.034**	20.744	26.321	**23.043**	21.254	25.250
DietNeRF, $\mathcal{L}_{\text{MSE}}$ ft (ours)	**24.311**	**25.595**	20.029	20.940	**26.794**	22.536	**21.621**	**26.626**
NeRF, 100 views	**31.618**	**34.073**	**25.530**	**29.163**	**33.197**	**29.407**	**29.340**	**36.899**

SSIM ↑	**Lego**	**Chair**	**Drums**	**Ficus**	**Mic**	**Ship**	**Materials**	**Hotdog**
NeRF	0.526	0.861	0.770	0.661	0.944	0.605	0.484	0.644
NV [5]	0.707	0.795	0.675	0.815	0.816	0.602	0.721	0.796
Simplified NeRF	0.775	0.859	0.727	0.872	0.930	0.694	0.823	0.894
DietNeRF (ours)	0.863	0.898	0.843	0.872	0.944	**0.758**	0.843	0.904
DietNeRF, $\mathcal{L}_{\text{MSE}}$ ft (ours)	**0.875**	**0.912**	**0.845**	**0.874**	**0.950**	0.757	**0.851**	**0.924**
NeRF, 100 views	**0.965**	**0.978**	**0.929**	**0.966**	**0.979**	**0.875**	**0.958**	**0.981**

LPIPS ↓	**Lego**	**Chair**	**Drums**	**Ficus**	**Mic**	**Ship**	**Materials**	**Hotdog**
NeRF	0.467	0.163	0.231	0.354	0.067	0.375	0.467	0.422
NV [5]	0.253	0.175	0.299	0.156	0.193	0.456	0.223	0.203
Simplified NeRF	0.218	0.152	0.280	0.132	0.080	0.283	0.151	0.139
DietNeRF (ours)	0.110	0.092	**0.117**	0.097	0.053	0.204	0.102	0.097
DietNeRF, $\mathcal{L}_{\text{MSE}}$ ft (ours)	**0.096**	**0.077**	**0.117**	**0.094**	**0.043**	**0.193**	**0.095**	**0.067**
NeRF, 100 views	**0.033**	**0.025**	**0.064**	**0.035**	**0.023**	**0.125**	**0.037**	**0.025**

DietNeRF had 24.31 PSNR (Table 2).

GRAF [9] also uses an adversarial loss, but has a different setup than DietNeRF. GRAF generates new object models, while DietNeRF synthesizes novel views of an observed object. Unlike SinGAN, GRAF uses many unposed, single-category images while DietNeRF trains on a few posed images of one scene (Realistic Synthetic datset) or multiple posed images of different categories (DTU). Despite these differences, adding our semantic consistency loss to generative models can be helpful. Since GAN generated objects do not correspond to individual images in the CUB dataset, we modified $\mathcal{L}_{\text{SC}}$ to compare representations of two renderings, rather than the representation of one rendering and one ground truth image. We performed an experiment on the CUB birds dataset used by [9] and find that a variant of our semantic consistency loss improves GRAF's validation FID from 36.32 to 33.44.

E. Probabilistic interpretation

Our loss regularizes a conditional generative model in unobserved regions. Let $f_\theta(\mathbf{p})$ be NeRF's rendered image with parameters θ at pose $\mathbf{p}$, and $p_D(x|\mathbf{p})$ be the unknown distribution over the true image x at pose $\mathbf{p}$. NeRF can define a generative model: a diagonal Gaussian centered at the rendered image $p_\theta(x|\mathbf{p}) = \mathcal{N}(x; f_\theta(\mathbf{p}), \mathbf{I})$. Then, NeRF maximizes conditional likelihood of observed images:

$$
\begin{aligned}
&\arg_\theta \min \mathbb{E}_{\mathbf{p}} \, \mathrm{KL}(p_D(x|\mathbf{p}) \parallel p_\theta(x|\mathbf{p})) \\
&\quad = \arg_\theta \min \mathbb{E}_{\mathbf{p}} \mathbb{E}_{p_D(x|\mathbf{p})}[-\log p_\theta(x|\mathbf{p})] \\
&\quad = \arg_\theta \min \mathbb{E}_{\mathbf{p}} \mathbb{E}_{p_D(x|\mathbf{p})}[\|f_\theta(\mathbf{p}) - x\|^2] \qquad (1) \\
&\quad \approx \arg_\theta \min \frac{1}{|D|} \sum_{(x,\mathbf{p}) \in D} \underbrace{\|f_\theta(\mathbf{p}) - x\|^2}_{\mathcal{L}_{\text{MSE}}, \text{ NeRF loss}} \qquad (2)
\end{aligned}
$$

since $p_D(x|\mathbf{p})$ doesn't depend on θ and log likelihood is MSE for Gaussians. NeRF estimates the expectation (1) with only a few $(x, \mathbf{p})$ pairs: those available in the dataset D. While the sample mean (2) with a finite number of poses $\mathbf{p}$ is unbiased, in practice we usually only have a few observed pairs per scene. To complete unseen regions better, we add a regularizer that retains the expectation over $\mathbf{p}$. If $q(z|x) = \mathcal{N}(z; \phi(x), \mathbf{I})$ for semantic net ϕ, DietNeRF's

Figure 2. **Training views used for Realistic Synthetic scenes.** These views are randomly sampled from the available 100 views. This is a challenging setting for view synthesis and 3D reconstruction applications as objects are not uniformly observed. Some views are mostly redundant, like the top two Lego views. Other regions are sparsely observed, such as a single side view of Hotdog.

minimizes a combined loss:

$$\begin{aligned}
&\arg_\theta \min \mathbb{E}_{\mathbf{p},x}[\mathrm{KL}(p_D \| p_\theta) + \lambda \mathrm{KL}(q(z|x) \| q(z|f_\theta(\mathbf{p})))] \\
&= \arg_\theta \min \frac{1}{|D|} \textstyle\sum_D \mathcal{L}_{\text{MSE}} + \lambda \mathbb{E}_{\mathbf{p},q(z|x)}[\|\phi(f_\theta(\mathbf{p})) - z\|^2] \\
&= \arg_\theta \min \frac{1}{|D|} \textstyle\sum_D \mathcal{L}_{\text{MSE}} + \mathbb{E}_{\mathbf{p}} \underbrace{\lambda \phi(f_\theta(\mathbf{p}))^T \phi(x)}_{\mathcal{L}_{\text{SC}},\ \text{DietNeRF regularizer}}
\end{aligned}$$

The regularizer could be estimated with many samples of $\mathbf{p}$.

References

[1] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12873–12883, June 2021. 2

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. 2

[3] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[4] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 1

[5] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, July 2019. 3

[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. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2

[7] Anton Obukhov, Maximilian Seitzer, Po-Wei Wu, Semen Zhydenko, Jonathan Kyl, and Elvis Yu-Jing Lin. High-fidelity performance metrics for generative models in PyTorch, 2020. Version: 0.2.0, DOI: 10.5281/zenodo.3786540. 2

[8] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Computer Vision (ICCV), IEEE International Conference on*, 2019. 2

[9] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[10] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014. 1

[11] Lin Yen-Chen. PyTorchNeRF: a PyTorch implementation of NeRF, 2020. 1

[12] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1

[13] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017. 2

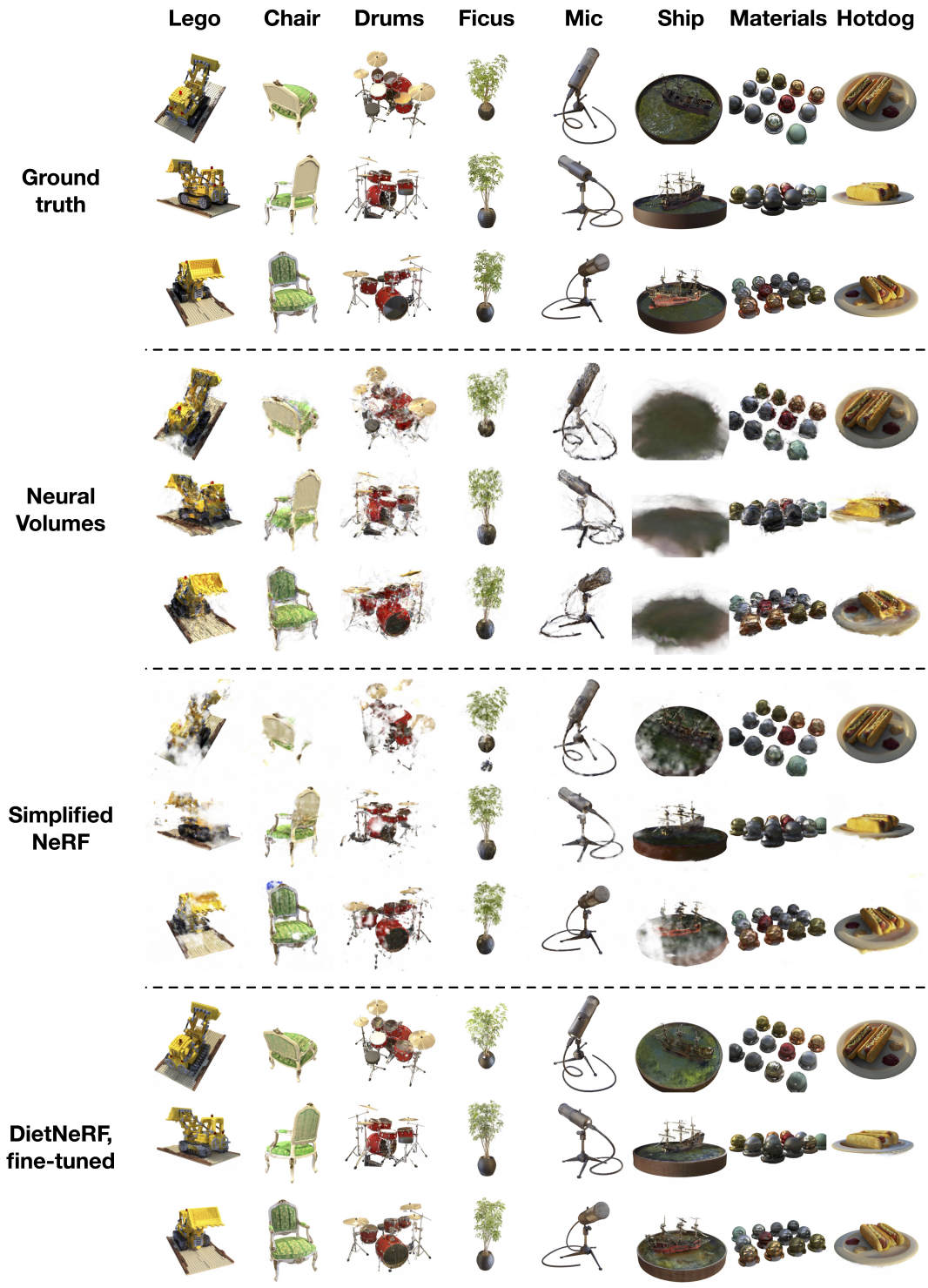

Figure 3. **Additional renderings of Realistic Synthetic scenes.**

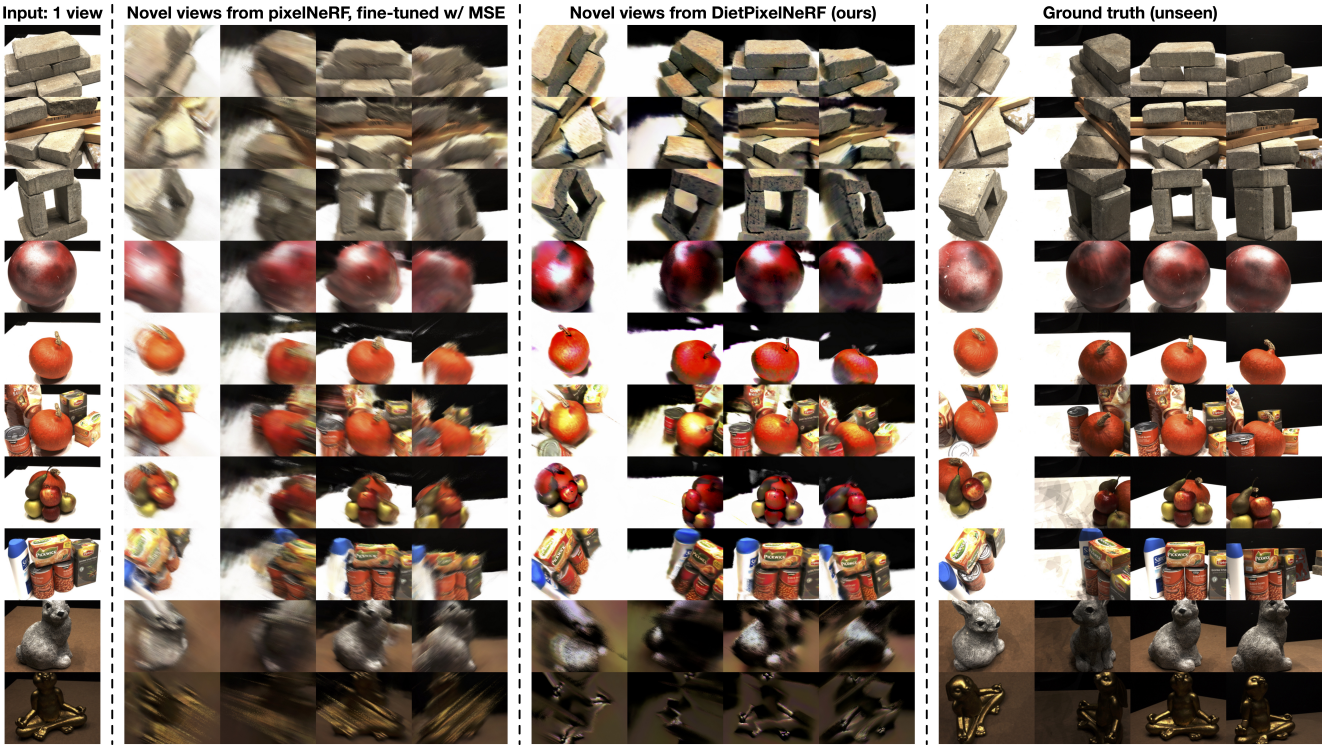

Figure 4. **One-shot novel view synthesis:** Additional renderings of DTU scenes generated from a single observed view (left). Ground truth views are shows for reference, but are not provided to the model. pixelNeRF and DietPixelNeRF are pre-trained on the same dataset of other scenes, then fine-tuned on the single input view for 20k iterations with $\mathcal{L}_{\text{MSE}}$ alone (pixelNeRF) or $\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{SC}}$ (DietPixelNeRF).