DE-TensoRF: Data Efficient and Fast NeRF

Anushree Bannadabhavi
University of British Columbia
anushree.bannadabhavi@gmail.com

Aditya Chinchure
University of British Columbia
aditya10@cs.ubc.ca

Abstract

Neural radiance fields (NeRFs) are models that can synthesize novel views of complex 3D scenes from a set of 2D images associated with their camera positions. However, NeRFs suffer from two drawbacks: they require a large amount of training data in the form of images from multiple camera locations, and they take a long time to train. To overcome the second challenge, TensoRF uses tensor decomposition and factorization to speed up rendering. We propose to extend TensorRF to make it data efficient (DE-TensoRF). We explore two techniques to improve TensoRF's robustness to data scarcity and occlusion. First, we use priors about the geometric structure and symmetry of the scene to obtain unseen views with new camera positions using existing seen views. Second, we use a pre-trained CLIP ViT model to ensure semantic similarity between all views as a conditioning method, as well as an additional loss term. We conduct experiments on objects from the Synthetic-NeRF dataset, which contain images of objects with complex geometry. Unlike TensoRF, DE-TensoRF is able to produce higher quality 3D reconstructions with a limited set of seen views. Code: github.com/aditya10/DE-TensoRF.

1. Author Contribution

We shared an equal workload on this project. Anushree worked on implementing the symmetry pseudo-labels, while Adi worked on semantic consistency based techniques. In the interest of space, we share detailed author contributions in Appendix A.

2. Introduction

Building compact 3D scene representations is a fundamental problem in computer vision, and has applications in AR/VR, graphics, and engineering. In recent years, Neural Radiance Fields (NeRFs) [10] and subsequent work have shown impressive performance on learning 3D scene representations from images and camera poses, and enabling the sampling of novel views.

While NeRFs produce high quality scene representations using a dense set of views, they struggle to perform well with limited data or insufficient training time, leading to artifacts such as blurriness and incompleteness in reconstructions of the scene. For example, training a single NeRF model for a scene can require upwards of 100 images, and over 24 hours on multiple GPUs [10]. Methods such as DietNeRF [6], PixelNeRF [17] and SinNeRF [16] propose methods that can reduce the number of views required, whereas InstantNGP [11] and TensoRF [2] make GPU-level optimizations and factorize 3D space respectively to reduce training time.

Our goal is to make NeRFs that are both fast and data efficient. We propose DE-TensoRF, a data-efficient implementation of the TensoRF [2] model that can perform high-quality 3D reconstruction with as few as 3 images in under 30 minutes of training time. NeRFs struggle with 3D reconstruction when a portion of the scene is occluded. However, if the object under consideration is symmetric, this can easily be resolved by using the available images to obtain symmetrically flipped unseen views. Leveraging this physical property, we propose a general symmetric transformation technique to obtain new images and camera poses of the occluded side of the scene using existing known views.

Another direction we explore to achieve data-efficiency is ensuring semantic consistency across views. We evaluate two key ideas, (i) semantic conditioning and (ii) semantic consistency loss. The semantic latent representation of any object should remain the same even when the object is viewed from different angles, i.e., a *chair* is a *chair* from any camera pose. Therefore, conditioning the model on this latent representation provides the model with additional semantic information about the scene, thus helping with novel view synthesis. Additionally, inspired by DietNeRF [6], we also explore the semantic consistency loss that guides the model by ensuring that the high-level image features of a generated novel view and the image features of a randomly sampled ground-truth train views are similar.

We propose DE-TensoRF, a modification of TensoRF [2], to incorporate these techniques. Our contributions are,

1. a method to obtain alternate symmetric views of the

scene that can be used as additional training data for DE-TensoRF,

2. a method to condition the TensoRF model on image features of known views to enforce semantic similarity across all the generated novel views,

3. an additional semantic consistency loss, that ensures that different reconstructed views are semantically consistent, using a CLIP ViT model, and

4. a thorough evaluation to show the benefits and limitations of each proposed technique.

3. Related Work

3.1. Novel View Synthesis and Neural Radiance Fields

Synthesizing photo-realistic novel views is an extensively researched topic in Computer Vision. Light field sample interpolation techniques [9] [3] have been used previously to reconstruct photorealistic novel views. With the advent of deep learning, recent approaches like DeepVoxels++[5] proposed an encoder-decoder architecture to learn latent 3D voxel embeddings that encode both shape and appearance learnt from multi-view images of an object. DeepView[4] approaches view synthesis using multiplane images estimated by CNNs. Although these methods produce impressive results, they are limited either by storage costs of their discrete voxel grids or by their inability to render high resolution images. Recently, NeRFs[10] and their follow-up works have shown great success in view synthesis by optimizing neural radiance fields to render photorealistic novel views of scenes with complex geometry and appearance. In contrast to previous techniques, NeRFs enable finer sampling of 3D space by encoding a continuous volume within the parameters of the MLP, producing high quality renderings. NeRF models are purely multilayer perception (MLP) based and thus also have minimal memory requirement.

3.2. Fast NeRF models

Despite their low memory requirements, NeRF models require significant training time, ranging from several hours to days. To address this issue, TensoRF [2] models the radiance field of a scene as a 4D tensor and introduces novel vector-matrix decomposition to factorize tensors into compact vector and matrix factors enabling fast reconstruction ($<$ 30min) with a lower memory footprint. InstantNGP [11] introduces efficient multiresolution hash encoding that allows for the use of a smaller neural network, thus reducing the number of floating point and memory access operations. DVGO [15] speeds NeRF up by replacing the MLP with a voxel grid and KiloNeRF [13] addresses this issue by utilizing thousands of tiny MLPs instead of a single large MLP. Although these methods achieve significant speed-up without compromising the quality, they still require dense views to reconstruct 3D scenes accurately.

3.3. Data efficient NeRF

Data-efficient NeRF models have recently gained attention for their ability to generate novel views of complex scenes from just a few input images. PixelNeRF [17] is a fully convolutional framework that enables a scene prior across multiple scenes, so that new scenes can be reconstructed from a sparse set of views. DietNeRF [6] improves few-shot quality by introducing an auxiliary semantic consistency loss across different reconstructed views. SinNeRF [16] considers the more ambitious task of training a NeRF using only a single view, using semantic and geometry regularizations. These methods have all shown promising results, but are not as quick to train as TensoRF. We propose DE-TensoRF, a data-efficient NeRF model that leverages assumptions about the physical properties of objects, such as symmetry, to obtain additional unseen views. We combine this approach with a CLIP-based [12] semantic consistency loss, similar to DietNeRF [6] to achieve higher quality 3D reconstruction on a limited set of images.

4. Method

DE-TensoRF incorporates two methods to improve data efficiency in the existing TensoRF model. First, we provide a short summary of TensoRF in Section 4.1. Next, we define a method to obtain alternate unseen views of a scene using symmetry, in Section 4.2. Finally, we explain how we use CLIP-ViT for semantic consistency across ground truth and novel views, in Section 4.3.

4.1. Background: TensoRF

NeRFs synthesize images by sampling 5D coordinates, location (x, y, z) and viewing direction (θ, ϕ), along camera rays to obtain color (r, g, b) and volume density σ. We maintain the architecture design of TensoRF [2], where the 3D voxel grid is factorised into three components, where each component consists of a 1D vector and a 2D tensor, following a novel Vector-Matrix (VM) decomposition method. We succinctly summarize the TensoRF model first.

We feed a camera pose C_i as an input to the model. Each C_i for the dataset of length N is first processed to obtain a list of rays $[r_0, ..., r_{H \times W}]$ of $H \times W$ length, where H and W is the height and width of the view. Next, for each point on the ray, we obtain coordinate features P representing (x, y, z, θ, ϕ) using the decomposed 3D space. These features are then fed into an MLP, which produces features E that represent the (r, g, b, σ) for the given point. The model is trained using MSE loss, where the RGB values obtained are compared against the ground truth images, pixel-wise, for each view.

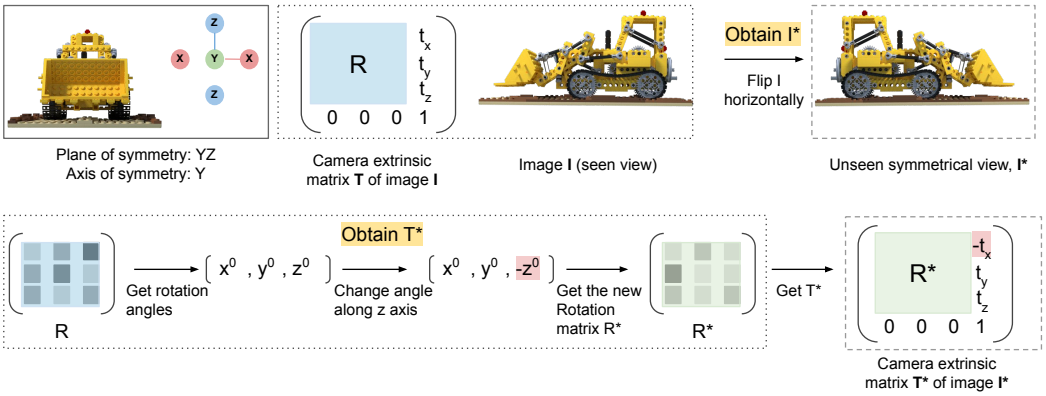

Figure 1. Our proposed technique to obtain symmetric unseen views. Image I from the train data is flipped to obtain pseudo-label I^* and the camera is rotated along z-axis and translation is done along x-axis to obtain the corresponding camera extrinsic matrix T^*.

We present DE-TensoRF, a data-efficient version of TensoRF, that can better represent 3D objects with $N << N'$ images, where $N' = 100$ is the full set of images for a scene in the synthetic NeRF dataset. We experiment with $N = 2, 3, 5, 10$ in our experiments.

4.2. Obtaining unseen views using symmetry

NeRFs when trained on a sparse set of input views overfit on the training data and produce inaccurate geometries since they have no prior knowledge about natural objects and symmetries. For instance, TensoRF when trained on 3 views of the left half of Lego is unable to render textures on the right half of the Lego (Fig. 3). This can easily be resolved using the textures of the known views since the image under consideration is symmetric. Therefore, we exploit the property of symmetry to generate pseudo-ground truth labels using known views.

For a given image I with associated camera extrinsic matrix T, we first determine an axis of symmetry, and then apply a horizontal or vertical flip accordingly, to obtain I^*. The camera extrinsic matrix is given by $T = [R\ t]$, where R is the 3×3 rotation matrix and t is a 3×1 matrix that represents translation along the x, y and z axis.

We follow the below steps to obtain the symmetric camera extrinsic matrix T^* of the unseen view,

1. we first determine the plane of symmetry and the axis of symmetry.

2. To obtain the camera location of the unseen symmetric view, we translate along the axis perpendicular to the plane of symmetry. Specifically, t_a is changed to $-t_a$ where a is either x, y or z for YZ, XZ or XY planes of symmetry respectively. We now have the new translation matrix t^*.

3. To obtain the camera rotation matrix R^* such that the camera points at the unknown symmetric part of the scene, we first obtain the x, y and z rotation angles from R. We then flip the sign of the angle along the axis in the plane of symmetry that is perpendicular to the axis of symmetry. For instance, for an object that is symmetric along the YZ plane and y axis, we change the angle along the z axis. Using the new angles, we obtain the new rotation matrix R^*.

4. Finally we obtain the new camera extrinsic matrix, $T^* = [R^*\ t^*]$

With I^* and T^* we now have pseudo labels for the unknown portion of the scene which can be used to train the NeRF model and thus improve the rendering quality of the underlying scene.

4.3. Semantic consistency between views

The theory behind semantic consistency is that the semantic representation of an object is similar from all views, i.e. a *Lego truck* is a *Lego truck* as viewed from any direction. We leverage this semantic consistency in two ways. First, we attempt to condition the MLP in TensoRF with image features from CLIP-ViT [12], which can provide valuable information to the model when a novel camera pose is reconstructed. Second, we use this consistency property in a loss function. These methods are described in detail below.

4.3.1 Semantic Conditioning (SC)

To condition our MLP with semantic features of the object, we first obtain image features $F = [F_1, ..., F_N]$ using the CLIP-ViT model for images $[I_1, ..., I_N]$ in the ground truth image set. During training, we sample an image feature F_i for every batch of rays that are fed into the MLP. Then for every coordinate feature p_k for each camera pose in the training set,

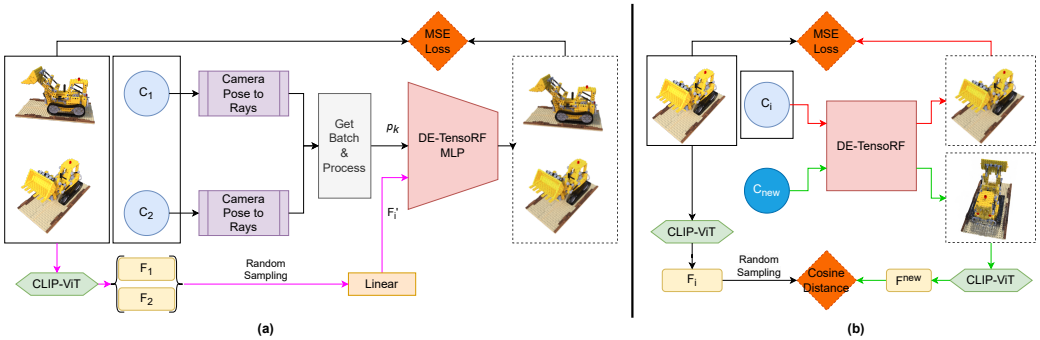

Figure 2. (a) Our implementation of Semantic Conditioning. We use CLIP-VIT [12] to obtain semantic features of the provided images, and train the TensoRF MLP to use these features. (b) Semantic Consistency Loss is implemented as a separate forward pass (green arrows) to the MSE loss (red arrows), where a novel view of size 128×128 is synthesized, and CLIP features are compared using cosine distance.

$$F_i' = W_c F_i + b_c \tag{1}$$

$$E = MLP_{im}(p_k \bullet F_i') \tag{2}$$

where W_c and b_c represent a linear layer that compresses the 512-dim feature tensor F_i from CLIP-ViT to a 32-dim tensor F_i', the $\bullet$ is the concatenation operation, and the MLP_{im} is a modification to the MLP in TensoRF where the input dimension is increased from $dim(p_k)$ to $dim(p_k) + 32$.

During testing, for a given novel camera pose C_{new}, we find the closest camera pose in the training set, and use the respective image feature from F to condition MLP_{im}.

4.3.2 Semantic Consistency Loss (SL)

The TensoRF model is supervised pixel-wise using MSE loss over the generated and ground truth views. In addition to this method of supervision, we use a semantic consistency loss (SL) applied to generated views from randomly selected camera positions, in a separate forward pass, as seen in Fig. 2(b).

We start by encoding the dataset images using the CLIP-ViT model, to obtain image features $F = [F_1, ..., F_N]$. In each training iteration, the model is first trained with MSE loss based on the ground truth camera positions and their respective images. Next, we randomly generate a new camera position C_{new}, feed it into the model to obtain a render of this novel view, I^{new}. We obtain a semantic feature vector F^{new} by encoding I^{new} using CLIP-ViT. Finally, we calculate the cosine distance (which is equivalent to negative of cosine similarity) between F^{new} and a sampled F_i from F.

Note that applying SL at every iteration is unreasonable because (1) the loss would only behave well when I^{new} is of a reasonable quality, and (2) backpropogation through the CLIP-ViT model is time consuming and memory intensive.

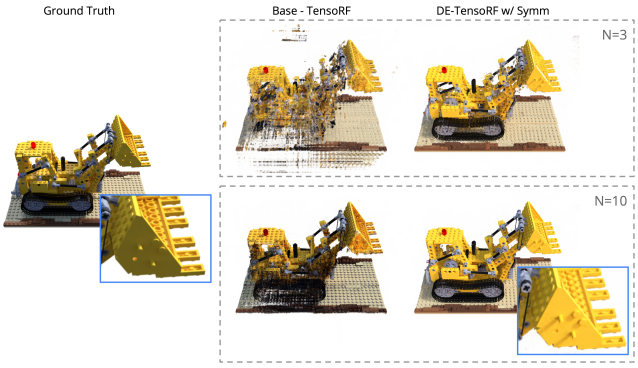

Figure 3. **Results with Symmetry**. Using symmetry produces significantly higher quality renders for the occluded side compared to the baseline model. However, we observe that luminance values between the GT and symmetry images are different, as seen in the blue boxes.

Therefore, we only use a 128×128 image size for I^{new}, and introduce three hyperparameters to control the frequency of applying SL loss, `sl-freq`, the start iteration after which the loss can be applied, `sl-start`, and a weight multiplier to the loss value, `sl-weight`.

5. Evaluation

We based our code on TensoRF and train our model following their setup. Implementation details are in C. We use TensoRF as the baseline for comparison and experiment with a set of N training images. We experiment with $N = 2, 3, 5, 10$ (see Appendix B) to evaluate the data efficiency improvements we present in our model, and the impact of the number of available views. Specifically, (1) we evaluate the effect pseudo ground-truth labels generated using symmetry in reconstructing unseen views and also present limitations of this method, (2) analyze the results of conditioning the NeRF on image features and (3) evaluate semantic loss and its effect on train time. We present

Experiment	N	PSNR ↑	LPIPS [A] ↓	LPIPS [V] ↓	SSIM ↑	FID ↓	Time (s)
Base TensoRF	2	14.126	0.372	0.349	0.657	232.56	720.76
w/ Symmetry	2	14.612	0.318	0.316	0.680	213.21	688.83
w/ SC	2	13.782	0.386	0.360	0.642	251.48	738.74
w/ Symm + SC	2	14.511	**0.315**	**0.314**	0.679	223.97	680.89
w/ Symm + SC + SL	2	**14.734**	0.322	0.324	**0.695**	**193.76**	1794.90
Base TensoRF	3	16.263	0.261	0.258	0.727	187.35	756.01
w/ Symmetry	3	16.851	**0.221**	**0.237**	0.748	**159.61**	712.70
w/ SC	3	16.161	0.268	0.264	0.721	194.15	723.78
w/ Symm + SC	3	**16.910**	0.222	0.239	**0.749**	162.83	732.12
w/ Symm + SC + SL	3	16.639	0.223	0.256	0.747	164.43	1816.42
Base TensoRF	5	**19.620**	0.151	0.171	**0.814**	128.45	751.95
w/ Symmetry	5	18.901	0.166	0.187	0.792	107.79	775.08
w/ SC	5	19.393	**0.144**	**0.168**	**0.814**	131.09	746.37
w/ Symm + SC	5	18.678	0.171	0.193	0.790	**107.25**	769.22
w/ Symm + SC + SL	5	17.885	0.184	0.219	0.777	133.93	1813.56
Base TensoRF	10	25.768	0.074	0.102	0.888	65.03	660.99
w/ Symmetry	10	22.034	0.096	0.118	0.854	66.55	728.26
w/ SC	10	**26.118**	**0.071**	**0.101**	**0.890**	**61.69**	656.20
w/ Symm + SC	10	22.620	0.091	0.114	0.858	68.08	734.36
w/ Symm + SC + SL	10	21.429	0.117	0.160	0.836	82.50	1741.77

Table 1. **Results on the lego dataset.** We see that using symmetry based pseudo images improved scores when $N = 2, 3$. SC results in a modest improvement in scores across all N, whereas SL only improves the SSIM and FID scores in the most data scarce $N = 2$ model. [A] is AlexNet, [V] is VGG.

both quantitative and qualitative evaluations.

5.1. Datasets

We focus our experimentation on the 'lego' and 'chair' objects from the Synthetic NeRF dataset [10], as they have meaningful physical properties that we can use as priors. The objects also have complex geometry and realistic non-Lambertian materials. We build $N = 2, 3, 5, 10$ datasets such that all the train images only contain either the left or the right side of the object and the other side of the object is completely occluded. We use the original test set of Synthetic NeRF with 200 novel poses for evaluation.

5.2. Metrics

We use four different metrics for quantitative evaluation - Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Learned Perceptual Image Patch Similarity (LPIPS) and Frechet Inception Distance (FID). Detailed information on these metrics is provided in Appendix D.

6. Results

Through quantitative and qualitative results, we show that our proposed techniques outperform the baseline TensoRF model in terms of quality of reconstruction, and on metrics such as FID. We show results on 'lego' here, and on 'chair' in Appendix F.

Reconstruction of unseen views using symmetry. For each of the aforementioned values of N, we use symmetry to obtain pseudo labels. The model severely lacks train data for $N = 2, 3$ and therefore, pseudo ground-truth labels help significantly, evident from the improved scores across all metrics (Table 1). Qualitatively, baseline TensoRF fails to reconstruct the unseen side of Lego, while DE-TensoRF trained with pseudo-labels is able to fill in the missing regions (Fig.3). However, for $N = 5, 10$, baseline TensoRF has better scores, most likely due to illumination effects in the scene. Our method for generating pseudo-labels does not take into account the location of the light source and shadow-effects. Although DE-TensoRF produces a better rendering quality by generating a more accurate reconstruction in terms of texture and colours (Fig.3), it is unable to capture the illumination setup of the underlying scene which results in decreased scores (Table 1).

Experiments with Semantic Conditioning. We evaluate two configurations: (i) DE-TensoRF w/ SC and (ii) DE-TensoRF w/ Symmetry + SC. From Table 1, improved scores can be seen for cases where $N = 5, 10$ with SC having the best scores for $N = 10$ case compared to all other techniques. This is due to the availability of a larger set of image features to sample from, thus increasing robustness. Qualitatively, SC helps in removing artifacts and incorrect colour effects as shown in Fig.4. However, we also

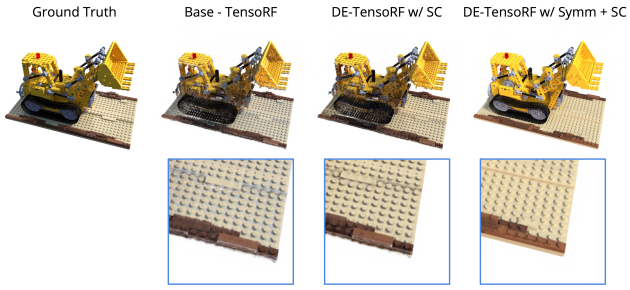

Figure 4. **Results with SC** for $N = 10$. We observe higher texture quality compared to baseline on the occluded side.

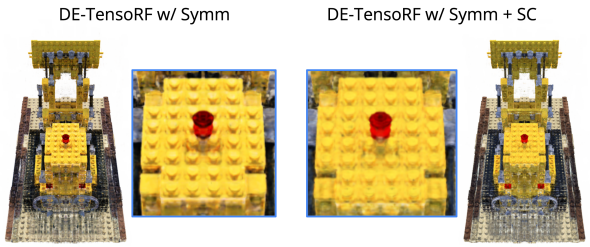

Figure 5. **Smoothing effect of SC** is visible in certain cases, which may not be beneficial.

observe a smoothing effect in high texture regions of the image (Fig.5).

To summarize, when $N < 5$, since the model severely lacks train data, addition of pseudo-labels using symmetry result in a considerable performance improvement. However, as N increases, the model benefits more from semantic conditioning to fill in the missing regions of the scene for 3D rendering.

Experiments with Semantic Loss. We evaluate the effect of introducing semantic loss to DE-TensoRF conditioned with image features and trained with pseudo-labels. We observe that the addition of semantic loss does not improve the performance except for improvement in FID score for $N = 2$ case (Table 1). For $N = 2$, the model simply does not have enough train views to produce a good reconstruction and therefore seems to benefit from the introduction of SL. However, when the number of train views increase, semantic loss leads to blurrier outputs because of the averaging effect the loss has over all views resulting in no quantitative or qualitative improvement (Fig. 6). Additionally, introduction of semantic loss doubles the training time. An important point to note here is that several factors like size of the rendered image, `sl-freq`, `sl-start` and `sl-weight` influence the effectiveness of semantic consistency loss, and while some hyperparamter tuning was done (Appendix E), we did not have sufficient time to experiment with this extensively.

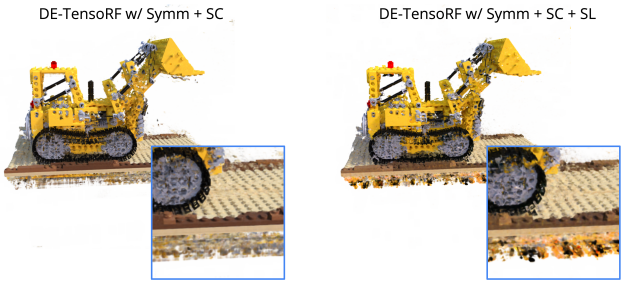

Figure 6. **Results with SL.** We observe addition blurriness and discolored textures with SL.

Summary of the experiments. DE-TensoRF outperforms the baseline TensoRF and achieves data-efficiency while further reducing the time required for training with a limited set of inputs. From the several qualitative and quantitative comparisons, we observed that symmetric pseudo-labels and SC results in improvements in scores across all N. As N increases ($N >= 5$), SC is the most beneficial technique. For $N < 5$, if the plane of symmetry is known, pseudo-label generation proves most promising for achieving better perceptual rendering quality, texture and colour reconstruction of the occluded side.

7. Limitations and Future work

Our proposed techniques help DE-TensoRF achieve data-efficiency. However, the proposed symmetric transformation does not apply to all scenes, as the plane of symmetry must be known, and the model does not handle illumination effects, making it a challenging approach in real life. Although SC results in modest improvements, it does not improve the rendered image quality significantly. This is likely due to CLIP embeddings only storing high-level information about the scene. Other alternatives to CLIP, like DINO-VIT [1], are worth exploring to improve SC and SL. Additionally, is it important to evaluate our techniques on real-world images and datasets.

8. Conclusion

We propose DE-TensoRF, a modified TensoRF model that uses priors based on symmetry and semantic consistency to allow good quality 3D reconstruction using as few as 3 images. While our work has several strong assumptions and limitations, we hope that it inspires future work in making NeRF models fast, robust and efficient.

References

[1] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerg-

ing properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[2] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022.

[3] M. Cohen, S.J. Gortler, R. Szeliski, and Grzeszczuk. The lumigraph. *SIGGRAPH*, 1996.

[4] John Flynn, Michael Broxton, Paul E. Debevec, Matthew DuVall, Graham Fyffe, Ryan S. Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. *CoRR*, 2019.

[5] Tong He, John Collomosse, Hailin Jin, and Stefano Soatto. Deepvoxels++: Enhancing the fidelity of novel view synthesis from 3d voxel embeddings. In Hiroshi Ishikawa, Cheng-Lin Liu, Tomas Pajdla, and Jianbo Shi, editors, *Computer Vision – ACCV 2020*, 2021.

[6] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5885–5894, October 2021.

[7] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[9] Marc Levoy and Pat Hanrahan. Light field rendering. *SIGGRAPH*, 1996.

[10] 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 *ECCV*, 2020.

[11] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 2022.

[12] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[13] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14315–14325, 2021.

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[15] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022.

[16] Dejia Xu, Yifan Jiang, Peihao Wang, Zhiwen Fan, Humphrey Shi, and Zhangyang Wang. Sinnerf: Training neural radiance fields on complex scenes from a single image. 2022.

[17] 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)*, pages 4578–4587, June 2021.

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

A. Author Contributions

In terms of ideation and development, Anushree worked on the idea of symmetry and implemented code to use this physical property to increase our dataset size, within the dataloader. Moreover, she spent two weeks learning Blender and related tools such as BlenderNeRF to validate our method for using symmetry. Adi worked on the semantic consistency part of the project. He set up the CLIP-ViT model and incorporated it both as a means of conditioning the output, and as a loss in the training step of the TensoRF model.

In terms of writing, both authors contributed equally. The abstract, introduction, related work, evaluation and results sections were written collaboratively at the same time, over Zoom. For the method section, Adi wrote Sec 4.1 and 4.3 and Anushree wrote Sec 4.2.

We use Bing AI to (1) research related works, using prompts such as "What are some related works to DietNeRF?" and (2) rephrase and make concise the related works section, using prompts like "Can you make this paragraph shorter? <a related works paragraph>". Note that we still fact-check the results that Bing AI gives us, as well as write how the related works connect to our project.

B. Datasets

In Fig 7 we show the different datasets we create for $N = 2, 3, 5, 10$. Note that in all datasets, the model is only able to see the left side of the lego truck and the right side remains occluded.

C. Evaluation Details

We use a batch size of 2048 rays, Adam optimizer[7] with initial learning rate 0.001. During the optimization process, we apply learning rate decay at every training step until it is reduced by a factor of 0.1 at the end. All our models are trained on a single Nvidia RTX 2080 Ti with 12 GB of VRAM, and takes under 30 min to train the model for 30000 iterations. The hyperparameters are available in our codebase config files for all our main experiments.

D. Metrics

PSNR measures the pixel-wise distance between the ground truth and rendered images, whereas SSIM considers overall luminance, contrast and structural similarity across regions of the image, which is far more representative of how humans would perceive the rendered image. Additionally we also used LPIPS [18] that measures the perceptual similarity between images using a neural network, AlexNet [8] and VGG encoders [14] in our case. Unlike PSNR and SSIM that look at pixel intensities, the FID score compares the distribution of generated novel views with the distribution of ground truth test views. We also report training time for all models.

E. Hyperparamter Search

We attempted a hyperparameter search for `sem-freq` and `sem-start`, when `sem-weight` was set to 1. We show training PSNR in Fig 8.

In the figure, `tensorf_lego_VM_10_20` denotes, for example, `sem-start` is set to iteration 10 and `sem-freq` is set to every 20 iterations.

Ultimately, we ran additional experiments and chose `sem-freq` $= 1$, `sem-start` $= 1000$, and `sem-weight` $= 0.01$. The reason for choosing such a low weight is that the MSE values at each iteration where of the order 10^{-2}, and using 0.01 would not overwhelm the MSE loss, while also still operating in every iteration.

F. Quantitative and Qualitative Results on 'chair'

We evaluate our proposed techniques on a second object - chair from the NeRF synthetic dataset. Although it is a simple object, it has a shiny surface, complex patterns and shadows. Quantitative results are presented in Table 2 and qualitative results are shown in Figure. 9. Except for improvement in FID scores for $N = 2, 3$, we do not see any other score improvements. Qualitatively, adding pseudo-labels fails at producing good reconstruction most likely due to the shiny surface, illumination effects and complex patterns.

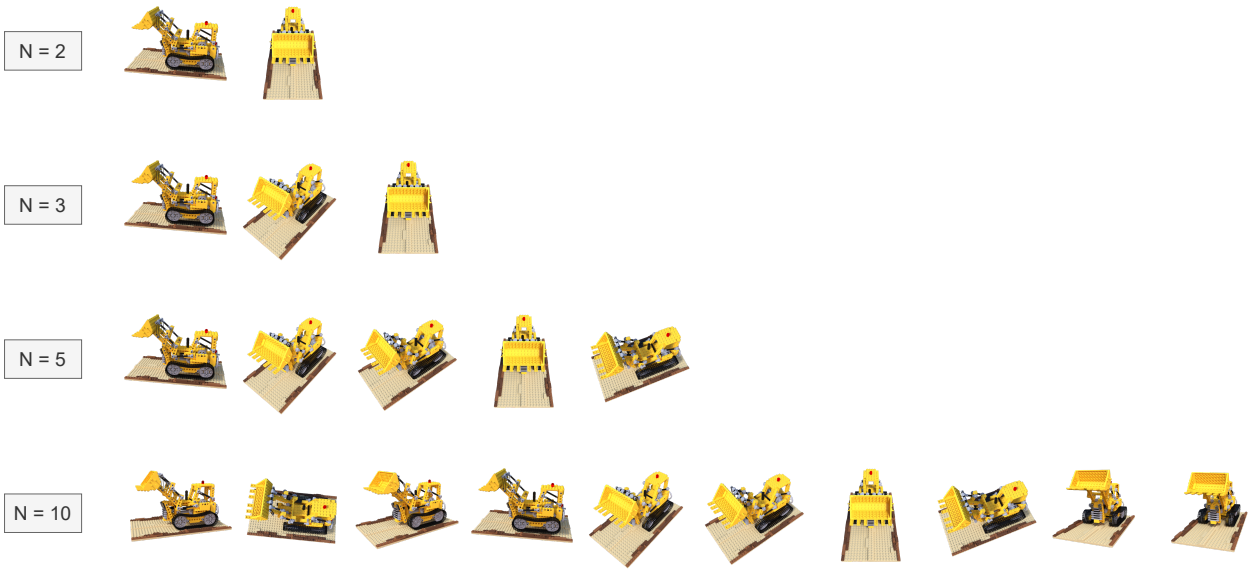

Figure 7. **Our dataset images for different N**.

Experiment	N	PSNR ↑	LPIPS [A] ↓	LPIPS [V] ↓	SSIM ↑	FID ↓	Time (s)
Base	2	16.110	**0.334**	**0.283**	**0.725**	325.39	809.25
Symm	2	**16.337**	0.378	0.313	0.712	340.23	779.27
Cond	2	16.092	0.341	0.288	0.723	**323.72**	826.57
Symm + Cond	2	16.153	0.390	0.323	0.702	326.71	806.08
Sem + Symm + Cond	2	15.286	0.340	0.301	0.721	325.15	2931.52
Base	3	**19.559**	**0.209**	**0.185**	**0.816**	181.31	697.26
Symm	3	18.567	0.253	0.227	0.787	201.10	714.66
Cond	3	19.531	0.210	0.188	0.814	**178.66**	715.14
Symm + Cond	3	18.237	0.270	0.238	0.777	203.25	730.95
Sem + Symm + Cond	3	18.786	0.236	0.242	0.799	201.03	2878.31
Base	5	**25.046**	**0.084**	**0.097**	**0.902**	**59.56**	756.05
Symm	5	19.429	0.230	0.218	0.798	151.73	729.23
Cond	5	24.942	0.089	0.101	0.899	62.47	731.67
Symm + Cond	5	19.305	0.241	0.227	0.791	163.55	762.41
Sem + Symm + Cond	5	18.974	0.232	0.239	0.800	159.96	2947.05
Base	10	**26.875**	**0.072**	**0.082**	0.917	**50.62**	715.49
Symm	10	21.601	0.147	0.151	0.847	84.90	741.14
Cond	10	26.758	0.073	0.085	0.917	53.43	731.43
Symm + Cond	10	21.639	0.153	0.155	0.848	90.50	744.91
Sem + Symm + Cond	10	21.124	0.167	0.189	**0.838**	107.65	2865.34

Table 2. **Results on the chair object.** We see improvements in FID scores in $N = 2, 3$, but otherwise we do not observe improvements. This may indicate that that our model does not handle the complex patterns or luminance in this dataset well, or we made a mistake with selecting the plane of symmetry.

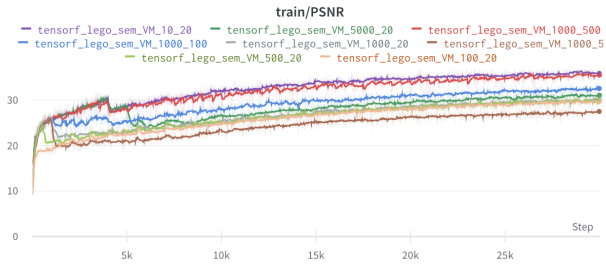

Figure 8. **Hyperparameter tuning** for `sem-freq` and `sem-start`. We see that with a higher `sem-freq`, the PSNR drop is significant, while decreasing `sem-start` also hurts training.

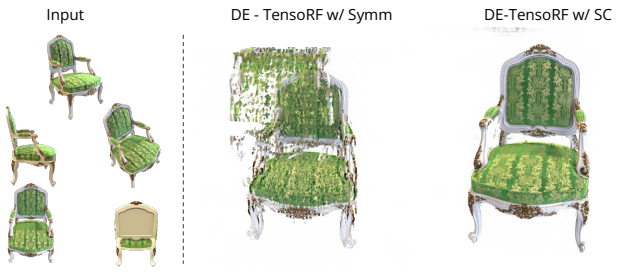

Figure 9. **Qualitative results for N=5 on the chair object.** This shows a case where our proposed symmetric transformation fails possibly due to complex patterns on the object and luminance where as SC helps produce good quality rendering of a novel view