SUPPLEMENTARY MATERIALS

Anonymous authors
Paper under double-blind review

In this document, we provide implementation details(Section 1) and additional qualitative results of GeCo-NeRF on Blender and DTU dataset(Section **??**). Next, we show ablation study of backbone architecture, by plugging our method to NeRF (3) to validate that our method is effective in synthesizing realistic novel view by enforcing geometric consistency regardless its backbone architecture.(Section **??**). We also give experimental quantitative analysis of our model regarding the number of input images given (Section **??**) to the model, in comparison with our baseline model. Finally, we give our analysis on our model's performance and the limitations of our method.

1 IMPLEMENTATION DETAILS

Network architecture. We use mip-NeRF (3) as network backbone and our neural radiance field is paramterized as fully connected ReLU network with hidden dimension of 256 and depth of 8 layers. We use 128 samples along the ray for both coarse and fine sample levels. Our model is implemented using the PyTorch framework (5) on top of the pytorch-NeRF code base (5).

Training details. We train our neural networks using ADAM optimizer (2). The learning rate is first linearly warmed up from 0 to 5×10^{-4} for the first 5k iterations, and then controlled by the cosine decay schedule to the minimum learning rate of 5×10^{-6}. We clip gradients by value at 0.1 and then by norm at 0.1. We train each model for 70k iterations for 6 hours in total on a two Nvidia 3090Ti GPU. We write the optimization algorithm for GeCo-NeRF's unobserved viewpoint consistency modeling as follows in Algorithm 1.

Unseen view patch generation. The pose for novel viewpoints are generated by adding a small divergence angle to the pose of a randomly selected given input pose. For each iteration, divergence angle is uniformly sampled within a range value that linearly increases as a function of iteration steps. We convert the ground truth 4-by-4 pose matrix into Euler angles, and add the sampled divergent angle values to x- and y-axis Euler angle vector components. The z-axis component of Euler angle vector remains zero to prevent roll rotation from occurring. Then we convert the divergent angle back into pose matrix and generate corresponding rays with the same module used to generate ground truth viewpoint rays. We generate 120×120 sized patchwise rays by bilinearly upsampling 60×60 rays in a strided grid.

Novel pose sampling scheme. Our training details regarding pose sampling are as follows: we generate unobserved viewpoints by sampling camera location and poses on a half-sphere, with camera direction always directed towards its center, just like ground truth camera orientations given in our datasets. For progressive camera pose sampling, we sample noise value uniformly within interval of [-d, +d] and add it to the original Euler rotation angles of reference poses. Sampling range parameter d grows linearly from 3 to 9 degrees throughout the course of optimization. This has an effect of generating viewpoints progressively further away from original reference viewpoints as training steps increase.

Consistency loss weight decay. Due to divergent behaviours that are shown by NeRF as it undergoes few-shot optimization, and to match different rates of decrease between vanilla NeRF loss and consistency modeling loss, we add a weighting hyperparameter to our regularization loss. We find that this scheme is effective in preventing the regularization loss from growing out of proportion and degrading the reconstruction quality. We exponentially decay the loss weight with decay parameter as 20000, which indicates that decay weight result 0.367 when the number of iteration steps reach the said number.

2 ALGORITHM

We present the algorithm of consistency modeling in GeCoNeRF as follows in Algorithm 1.

Algorithm 1 Consistency Modeling Algorithm.

initialization;
while $i < N$ **do**
  for $j \leftarrow 1$ **to** $m-1$ **do**
    $\alpha \leftarrow \exp(-\sigma_\theta(g_i(\boldsymbol{p}(\hat{z}_j))) \cdot \Delta_j)$
    $\boldsymbol{c} \leftarrow \boldsymbol{c}_\theta(g_i(\boldsymbol{p}(\hat{z}_j))\boldsymbol{v})$
    $\boldsymbol{C}_u \leftarrow \boldsymbol{C}_u + A \cdot (1-\alpha) \cdot \boldsymbol{c}$
    $D_u \leftarrow D_u + A \cdot (1-\alpha) \cdot \hat{d}_j$
    $A \leftarrow A \cdot \alpha$
  end
  if *regularization step* **then**
    Backprojection: $x \leftarrow D_u(p_u)K^{-1}p_u$
    Reprojection: $p_{u\to k} \leftarrow KR_{u\to k}x$
    Inverse warping:
    $I_{k\to u}(p_u) \leftarrow \text{sampler}(I_k; p_{u\to k})$
    Mask generation:
    $M(p_u) \leftarrow \left[\|D_u(p_u) - D_k(p_{u\to k})\| < \tau\right]$
    Regularization loss calculation:
    $\mathcal{L}_{cons} \leftarrow \sum_{l=1}^{L} \frac{1}{C_l m_l} \|M_l \odot (\phi_l(I_{k\to u}) - \phi_l(I_u))\|$
  end
  Minimize $\mathcal{L}_{cons}$
end

3 ROBUSTNESS TO NUMBER OF INPUT VIEWS

To see the varying effects of our regularization loss regarding the number of input viewpoints, experimented on scenarios with 3, 6, 15 reference views on Blender dataset, and observed PSNR differences between our model and vanilla model in each setting. We selected 3, 6, 15 reference views, since RegNeRF and InfoNeRF both show saturating improvement against their baseline at 15 reference views. The reason for our usage of 15 viewpoints, and not standard viewpoints (usually over 100), for abundant view setting is due to observations given in previous few-shot NeRF papers RegNeRF (4) and InfoNeRF (1): they observe that their vanilla baselines start to achieve better performance once enough input views are given, generally around 11-15, varying by the dataset.

Table 1: **Analysis with varying input view numbers.**

Methods	3 views	6 views	15 views
mip-NeRF (3)	17.94	22.55	**26.13**
GeCo-NeRF (ours)	**19.23**	**22.89**	25.83

As shown in table 4, while our model displays better performance over our baseline model at few-shot settings, the difference between the two models becomes smaller as more input views are given. Finally, 15-view setting, our model is overtaken at by the baseline. This is consistent with results given in previous few-shot NeRF papers (1; 4) that commit to similar experiment. We speculate that NeRF reaches a point regarding inputs viewpoints where the regularization no longer benefits its optimization but rather constrains it from achieving higher performances.

4 LIMITATIONS

Our major limitations are as follows:

(i) Our method uses thresholding technique for occlusion handling and mask generation, and this makes our method sensitive to differences depending on different scenes and datasets. Even though we use singular threshold throughout different scenes, there are cases where small protrusions in the scene (such as lego blocks in Blender Lego scene) are masked out due to viewpoint-point based depth difference values that go just over the threshold values, and subjected less to novel viewpoint regularization.

(ii) Our observed view consistency modeling depends on the assumption that there are large areas of surfaces that are covered by multiple viewpoints and thus able to be subjected to warping-based regularization. However, this validity of this assumption depends heavily on the nature of the scene and viewpoint selected, as there can be scenarios where a certain ground truth view shares no viewpoint with other ground truth views due to self-occlusion and large angular discrepancy between viewpoints. In such cases, all patches that are warped to the viewpoint would be masked out completely, resulting unnecessary computational cost for one third of the consistency modeling regularization steps.

5 BROADER IMPACT

Our work enables few-shot optimization and rendering of NeRF, which allows neural radiance fields to be optimized with sparse images, which makes it more readily applicable to real-life applications, such 3D reconstruction and novel view synthesis. This in turn opens up new possibilities in multiple different ways, such as augmented reality, 3D scanning and easily manipulable visual effects.

REFERENCES

[1] Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[2] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. iclr. 2015. *arXiv preprint arXiv:1412.6980*, 9, 2015.

[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. In *ECCV*, 2020.

[4] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[5] Lin Yen-Chen. Nerf-pytorch. `https://github.com/yenchenlin/nerf-pytorch/`, 2020.