[Supplementary Material] Robustifying The Multiscale Representation for Neural Radiance Fields

Nishant Jain[1]
njain@cs.iitr.ac.in
Suryansh Kumar[2†]
sukumar@vision.ee.ethz.ch
Luc Van Gool[2,3]
vangool@vision.ee.ethz.ch

[1] Indian Institute of Technology Roorkee, India
[2] ETH Zürich Switzerland
[3] KU Leuven Belgium

Abstract

This draft contains more experimental analysis and details relevant to the work presented in the main paper. Firstly, we demonstrate our approach's applicability to the challenging realistic case. To this end, we show our method's performance on the popular Tank and Temples dataset [9], as well as on our sequence taken from a regular freely moving camera. Next, we discuss the inner working of multiple rotation averaging via the modern graph neural network framework and extensively evaluate our graph neural network on pose-graphs by considering different scenarios of inducing noise. Later, additional details and results on our joint optimization, along with the analysis of error in predicting poses, are presented to better understand our take on the problem. Finally, we provide all the implementational details related to network architecture, hyperparameters to reproduce our results and some discussion on baselines used to prove the effectiveness of our method.

A Results on Challenging Real-World Sequence

A.1 Tank and Temples

Tanks and Temples is a well-known challenging dataset containing real-world scenes [9]. It consists of images showing large scale scenes to simulating the realistic conditions and largely used for evaluating 3D reconstruction methods. Further, this dataset can be very useful for testing unconstrained view-synthesis methods. Accordingly, we used couple of sequence to test our method's performance.

Specifically, we used "truck" and "tank" sequence, which consists of image set containing a 360° view of the subject captured freely at a varying distance from the object. Since there are no ground-truth poses provided by the dataset, we used COLMAP[14] to estimate the initial poses and feed them to our network (see Fig.1). Table (1) shows the comparison

© 2022. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.
†Corresponding Author.

	MipNeRF			Ours		
	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑
Truck	23.1	0.308	0.812	**24.7**	**0.296**	**0.828**
Tank	24.8	0.313	0.823	**26.6**	**0.302**	**0.851**

Table 1: Performance comparison of our approach and Mip-NeRF[2] on the Truck and Tank sequence [9]. The result shows that our method performs better as compared to Mip-NeRF on both the freely moving sequences demonstrating our method's advantage. It can be inferred from the above statistics that just relying on COLMAP poses for solving image based rendering on unconstrained sequence can demonstrably give inferior results. On the other hand, our joint optimization can handle outlier camera pose and provide favorable novel view rendering at the same time.

between our method and Mip-NeRF[2] for these two sequence. Clearly, our method efficiently optimizes over the poses esimated by the COLMAP and is able to generate better image renderings when compared Mip-NeRF+COLMAP setting. For qualitative analysis, we also provide visual comparison of the synthesized novel views on the truck sequence in Fig.6. **Observation:** *(a) The experimental result shows that our method can further enhance and broaden the scope of multi-scale neural radiance field modeling so that its theory can be successfully applied in an unconstrained environment without using geometric 3D scaffold as an input prior for the image rendering [13]. (b) Relying on only COLMAP poses [14] may provide good results on synthetic scene or a well controlled sequence. But for a general real-world video sequence, it may provide erroneous camera poses thereby leading to inferior results. Hence, our joint formulation provide robustness to pose at the same time gives better multi-scale image rendering.*

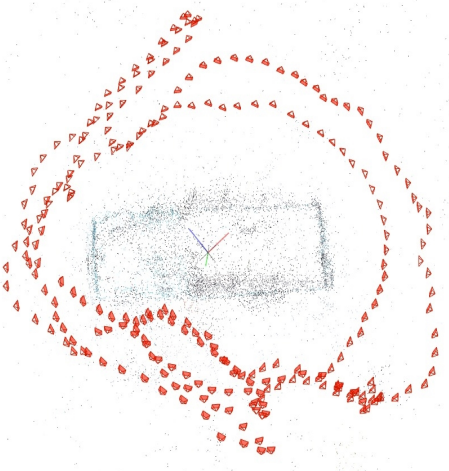


Figure 1: The camera pose trajectory recovered using COLMAP [14] on the truck sequence [9]. The video sequence consists of 251 poses taken freely with overall view of the truck.

A.2 Our dataset taken from a freely moving camera

To further investigate the real world use case of our approach, we test our proposed algorithm on a video taken from a freely moving camera. It contains the images of a black box taken by randomly moving camera around it. This image set mimics both pose and scale issues similar to any randomly collected set of day-to-day multi-view images. Figure 3 shows a

(e) Mip-NeRF [2] (f) Ours

Figure 2: Visual result comparison our method (right) with Mip-NeRF (left) on the Truck sequence [2]. The initial camera poses are generated using COLMAP [14]. As evident from these results, our method performs better than Mip-NeRF [2] on this challenging real-world sequence.

subset of the image set and Figure 4 (left part) shows approximate camera pose recovered using COLMAP[14].

Generating initial poses. We use COLMAP[14] for the initial keypoint detection and feature matching which is then used for predicting initial relative camera pose. It is well known that matching keypoints between any image pair results in multiple proposals for relative orientation between them, out of which some can be incorrect whereas other can be accurate. We used this pose prior to initialize our pose-graph network. Contrary to Mip-NeRF [2], our

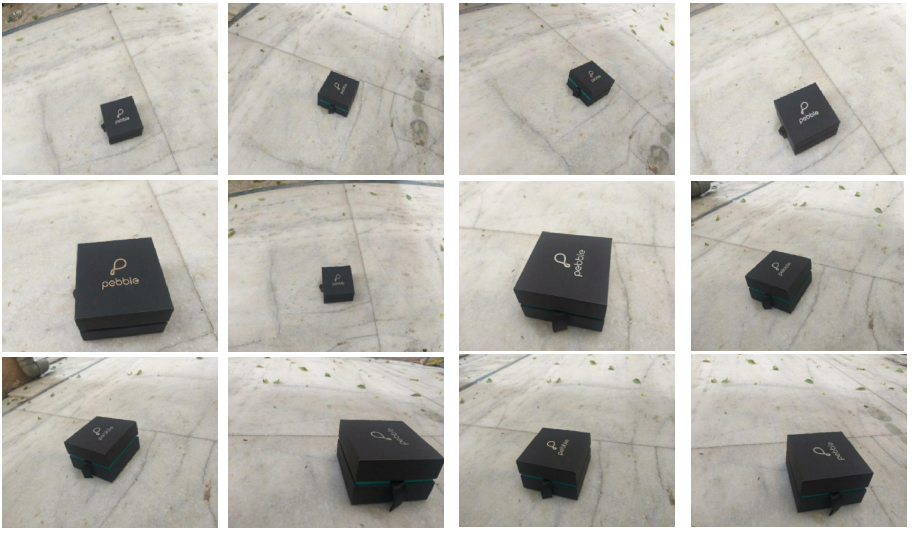
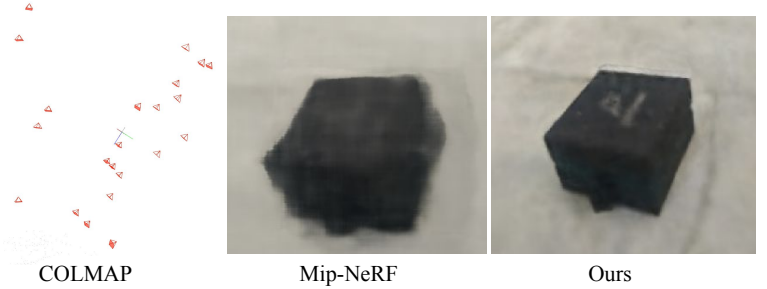


Figure 3: Images corresponding to our black box dataset taken using freely moving camera in an unconstrained manner (discussed in Sec. §A.2). We imitated a very general case of day-to-day captured multi-view images taken at different distance from the object with camera jittering effects as well.

method can take care of bad camera pose priors via rotation averaging and simultaneously optimize for multi-scale rendering.

Experimental Analysis. Figure 4 shows the reconstructed box obtained by using our proposed algorithm. It is apparent from this figure that our algorithm can infer accurate image-based rendering of the scene (novel synthesized view) despite having uncertain poses and images taken freely at arbitrary scales. Again the experimental results endorses the previously outlined observation , *our method can be successfully provide better results for image based rendering as compared to Mip-NeRF [2] in an unconstrained environment without using geometric 3D scaffold as an input prior for the scene [13].*



COLMAP Mip-NeRF Ours

Figure 4: **Left:**: Approximate camera pose recovered using COLMAP[14] (described in Sec. §A.2). The camera poses are plotted to show the randomness in the camera motion estimated using COLMAP. Hence, using it blindly make the framework prone to high pose errors. **Mid** : Rendered images using Mip-NeRF. **Right** : Rendered images using our algorithm on this dataset showing the applicability of our method on day-to-day video sequence.

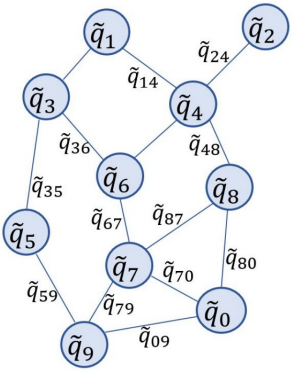


Figure 5: An example view graph extracted form the input poses. The vertex set $\mathcal{V}_j$ of this viewgraph correspond to initial absolute orientation of each of the images and the edge set $\mathcal{E}_{ij}$ correspond to the relative orientations between the image pairs.

B Graph Neural Networks for MRA

Here, we discuss in details the working of our pose network performing multiple rotation averaging (MRA) based on message passing graph neural networks (GNNs). We first discuss the working of Message Passing Networks(MPNN) involving communication between a graph node and its neighbours.

***(i)* Message Passing Scheme.** Given a directed view-graph $\mathcal{G}$ (see fig. 5) with N cameras and M pairwise relative orientation, we used the message passing neural network approach to operate on it. Let $m_j^{(t)}$ be the message functions that correspond to the message from the neighboring nodes $u \in \mathcal{N}_j$, $\psi^{(t)}$ as the update functions (T layers), and $h_j^{(t-1)}$ the state of node j at time step $(t-1)$. The feature node state $h_j^{(t)}$ at time t in the graph is updated as:

$$h_j^{(t)} = \psi^{(t)}\left(h_j^{(t-1)}, m_j^{(t)}\right) \tag{1}$$

$\psi^{(t)}$ corresponds to concatenation operation followed by a 1D convolution and ReLUs. The message function condensed message $m_j^{(t)}$ at node j due to all neighbor $\mathcal{N}_j$ is expressed as

$$m_j^{(t)} = \Omega_{\mathcal{V}_i \in \mathcal{N}_j} h_{i \rightarrow j}^{(t)} \tag{2}$$

Here, $\Omega(.)$ denotes a differentiable function like the softmax activation function, . , $h_{i \rightarrow j}^{(t)} := \Psi^{(t)}(h_j^{(t-1)}, h_i^{(t-1)}, e_{ij})$ is the accumulated message for the edge $\mathcal{E}_{ij}$ at t. $\Psi^{(t)}$ is concatenation operations followed by 1D convolution and ReLU activation. For our problem, $\mathcal{N}_j$ is the set of all neighboring cameras connected to $\mathcal{V}_j$ and e_{ij} is the edge feature of the edge $\mathcal{E}_{ij}$. For more details refer [4, 12]

***(ii)* Robustifying Poses using GNN.** The GNNs pipeline for estimating robust pose consists of three major steps: *(a) Cleaning the view-graph.* We first estimate the relative rotations $\tilde{R}_{ij}$ from the noisy rotations R_i due to input data. Next, we apply cycle consistency check to remove the outliers [1, 6]. Local cyclic graph structure of the view-graph along a cycle from an orientation close to identity and indicates the presence of outlier in the cycle. *(b)*

Generating a noisy initial solution using the extracted relative rotations. For this, we build a minimum spanning tree (MST) using all the nodes in the viewgraph by fixing the root to be the node with maximum neighbours (greatest fan-out). Then, we generate an initial absolute rotation $\hat{R}_i$ for each node by propagating from the root to that node along the tree. *(c) Refining the initial solution using Graph Neural Networks*. For applying GNNs, we require features corresponding to each node in a view graph. We use the rotation matrix in the initial solution corresponding to every node in the graph as its input feature. Furthermore, we also pass the observed relative rotations $\tilde{R}_{ij}$ as edge features to the GNN following the formulation described in the previous paragraph. Also, instead of directly passing these relative rotations as edge features, we instead pass the discrepancy between these observed relative rotations and the initial solution resulting in the the edge feature $e_{uv} = \hat{R}_v^{-1}\tilde{R}_{uv}\hat{R}_u$, to the GNN.

The resultant input view graph then becomes $\mathcal{G} = \{\hat{R}_i, e_{ij}\}$ and the supervised problem becomes $R_j^f := f(\{\hat{R}_i, e_{ij}\};\Theta)$ trained using the rotation averaging loss function. Moreover, we know that relative rotation between any 2 nodes in the viewgraph is invariant to any constant angular deviation in form of rotation matrix R to both the nodes and thus, both the solution sets $\{R_iR, R_jR\}$ and $\{R_i, R_j\}$ result in the same discrepancy when using the rotation averaging loss function. To handle this issue involving an unknown global ambiguity in the rotations, we also include a regularizer as a discrepancy between the absolute rotations in the objective. This results in the aggregate loss function described in Sec. §3.2.1:

$$\mathcal{L} = \sum_{\mathcal{G}\in\mathcal{D}} \sum_{\mathcal{E}_{ij}\in\mathcal{E}} d_Q(q_{ij}^f, q_{ij}) + \beta \sum_{\mathcal{V}_j\in\mathcal{V}} d_Q(q_j^f, q_j) \tag{3}$$

Given the formulation and pipeline of our pose-refining method, we now analyse its performance separately from the complete reconstruction pipeline, to further demonstrate its effectiveness in resolving errors in the input rotations.

B.1 Experiments and Analysis

B.1.1 Synthetic View Graphs

As discussed in the main paper, our pose network is pre-trained by using supervision from 1200 synthetic viewgraphs proposed by Purkait *et al.* [12]. This dataset consists of both outliers (fraction varying between 0 and 0.3) and noisy rotations where noise is sampled from a normal distribution with std. uniformly varying between 5°and 30°. We train our model on 80% of the viewgraphs and remaining are used for testing. The mean and median values resulting from evaluating our trained pose robustifying network on this dataset are as follows:

number of examples	mean error	median error
120	2.09°	1.1°

Such statistics shows that our approach is effective in reducing the overall camera motion error, which otherwise could directly affect the structure estimation leading to inferior 3D reconstruction result. Next, we study the performance of our method on the pose estimation when tested on Blender Synthetic Dataset.

Scene	Noisy Rotation (°)		Improved Rotation (°)	
	mean	rms	mean	rms
Lego	1.78	4.24	**0.031**	**0.041**
Ship	2.12	4.02	**0.052**	**0.063**
Drums	1.98	3.65	**0.038**	**0.052**
Mic	2.36	5.12	**0.073**	**0.091**
Chair	1.76	3.45	**0.056**	**0.071**
Ficus	2.46	5.34	**0.065**	**0.096**
Materials	1.58	3.55	**0.046**	**0.074**
Hotdog	2.23	4.78	**0.052**	**0.071**

Table 2: The performance of our MRA pose-network pipeline on the view-graphs generated from the poses available in the Blender Synthetic Dataset[11]. For this experiment, we perturb the poses using the method described in the paper. The results show the performance for both the cases , before and after applying our MRA pose network.

Scene	MipNeRF[0]			Ours		
	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑
Chair	28.7	0.054	0.95	**33.7**	**0.035**	**0.97**
Ficus	23.91	0.059	0.91	**28.6**	**0.041**	**0.93**
Mat.	14.2	0.103	0.78	**25.1**	**0.067**	**0.89**
Hotdog	23.81	0.063	0.90	**33.1**	**0.031**	**0.95**

Table 3: Experiments on the remaining scenes of the Blender dataset. Synthetic errors, as described in the paper, are introduced in the poses for this experiment. These results further verify the improved performance resulting from evaluating our algorithm against the baseline Mip-NeRF[0], in presence of synthetically introduced pose errors and scale issues in the dataset.

B.1.2 Blender Pose Graphs

We tested our trained pose network using the Blender synthetic dataset to generate the pose-graph. For this experiment, we introduced noise to 20% of the camera poses obtained from Blender synthetic dataset. We used these noisy poses to construct the input view-graph for our trained pose network. Table 2 shows the results before and after applying our pose method. The results demonstrate the robustness of our approach in pose estimation. Our method can handle noise in the view-graph and reduce the mean error favorably, thus minimizing the impact of noise in the poses.

B.1.3 Pose comparison with COLMAP [14] on Noisy Correspondence

Of course, we can have good image correspondences for a high-quality rendered image sequence; however, real world images can be noisy affecting the correspondence match leading to a bad pose estimate using frameworks like COLMAP [5, 7]. Hence, we conducted an experiment to show the robustness of our approach. Here, we used the Lego dataset to generate viewgraph by adding noise to the matched correspondences. Image correspondences were used to estimate pairwise relative rotations. Using it, we estimate the absolute rotations from these noisy relative rotations using (a) COLMAP and (b) our pose network. The pose difference for both of these cases is shown in Fig.(6). The difference in the recovered pose shows (using line) the effectiveness of our approach in handling pose errors. Our approach gives better results compared to COLMAP and the recovered pose is consistently better across images.

C Additional Results

Here, we discuss the result comparison with Mip-NeRF [2] for remaining scenes from the Blender Dataset (not provided in the main paper). Table 3 contains results for the remaining

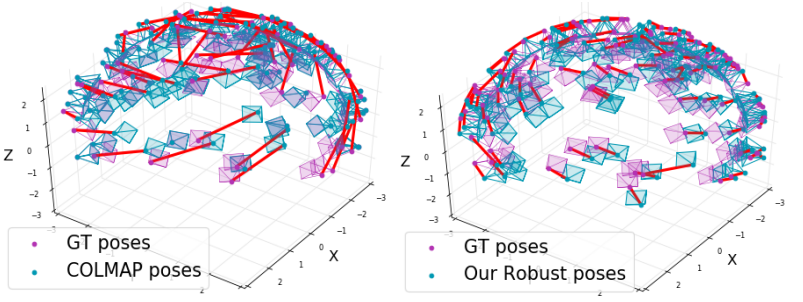


Figure 6: Comparison of camera pose predicted by COLMAP [14] and our methods on Lego dataset, when noise is added to the pairwise matched correspondences. **Left:** Estimated camera pose from COLMAP[14] and the ground truth pose trajectory with the red line representing the pose difference. **Right:** Our recovered camera pose plotted against the ground-truth poses. It is easy to infer that our approach has better pose estimate. (Red line shows pose error).

cases under single scale settings. Similar to the main paper, ground truth poses are perturbed to account for noise. The statistics indicate that our method provides better results than the Mip-NeRF and showing a substantial improvement on the remaining 4 scenes.

After evaluating both Mip-NeRF[2] and our method on 16 different scenes (8 multi-scale and 8 single scale) corresponding to the Blender Synthetic dataset, it seems evident that correct poses plays a significant role when integrating multi-scale modeling with NeRF [11]. Thus, it becomes necessary, if not compulsory, for multi-scale neural radiance field modeling to inherit robust pose optimization formulation for better scene representation. To further support our case, we provided a supplementary video for convincing comparison. Next, we provide additional details on the joint optimization of our propose loss function.

C.1 Joint optimization of our loss function with $\lambda = 0.5$

To understand the relation between rendering cost function and pose cost function optimization, we study the our overall loss function, when equal weights are assigned to both the terms of Eq.(10) in the main paper , $\lambda = 0.5$. We recorded its performance for both single and multi-scale image case as described in the main paper. Additionally, we noted our method's pose error performance in the predicted rotation matrices.

Table 4 provide the results of this study using pose error metric and image rendering metric (PSNR, LPIPS). As discussed in the paper, these results demonstrate the difficulty in jointly optimizing the pose network and rendering network parameters. This is due to the highly non-linear nature of the joint optimization problem. Hence, our approach of biasing the λ over iteration gives a better way to optimize the proposed loss function than hard coding it.

C.2 Multi Scale Images with Ground-Truth Pose

We further analyze the multi-scale case using the multi-scale Blender dataset, but this time without perturbing the ground truth poses. Table (5) compares the PSNR, LPIPS and SSIM values for this case. Once again, our method performance is similar to the current baselines, and the difference to the best method [2] is minor, thereby showing the effectiveness of our

Scene	Mean Error (°)			Single-Scale Dataset		Multi-scale Dataset	
	Single	Multi	Ours	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓
Lego	0.73	0.86	**0.03**	24.02(**27.10**)	0.08(**0.05**)	22.20(**27.01**)	0.06(**0.04**)
Ship	0.58	0.74	**0.05**	23.78(**25.45**)	0.10(**0.07**)	23.34(**26.59**)	0.07(**0.07**)
Drums	0.76	0.83	**0.03**	20.2(**24.98**)	0.08(**0.07**)	15.62(**26.07**)	0.07(**0.04**)
Mic	0.38	0.42	**0.07**	24.2(**30.03**)	0.06(**0.03**)	22.60(**30.08**)	0.04(**0.01**)
Chair	0.41	0.47	**0.06**	27.40(**33.70**)	0.03(**0.05**)	25.70(**35.23**)	0.07(**0.03**)
Ficus	0.67	0.78	**0.07**	24.22(**28.60**)	0.07(**0.04**)	22.70(**29.28**)	0.07(**0.03**)
Materials	0.85	1.12	**0.05**	20.26(**25.10**)	0.07(**0.07**)	18.83(**24.80**)	0.09(**0.06**)
Hotdog	0.51	0.62	**0.05**	26.56(**33.10**)	0.06(**0.03**)	23.31(**32.50**)	0.08(**0.03**)

Table 4: Performance of the joint optimization with $\lambda = 0.5$ in Eq.(10) of the main paper. We recorded the pose estimation error and image rendering accuracy on both single scale and multi-scale Blender dataset. For pose-error mean angular error (in degrees) across all the rotations, are tabulated. As a reference to show the degradation of the performance, we have added the mean angular error in rotations using our approach for joint optimization, for the single scale case. The table also provides PSNR and LPIPS image rendering accuracy metric results for better understanding. **The bracket values for the PSNR and LPIPS denote our method's result as proposed in the main paper.**

approach. The point to note is, using our method, we don't have to rely on a separate module for estimating accurate pose and it is recovered jointly with the object's neural representation.

	Lego			Ship			Drums			Mic		
	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑
Mip-NeRF	29.34	0.045	0.938	28.64	0.0651	0.778	26.9	0.0452	0.92	34.7	0.0088	0.978
BARF	11.18	0.52	0.70	9.23	0.76	0.48	11.2	0.68	0.66	12.18	0.53	0.74
RM-NeRF(ours)	29.30	0.045	0.929	28.57	0.0653	0.778	26.7	0.0455	0.93	34.3	0.0082	0.969

Table 5: Comparison of the 3 metrics: PSNR, LPIPS and SSIM, resulting from the evaluation of our proposed RM-NeRF, BARF[10] and Mip-NeRF[2] on 4 scenes of the multi-scale blender dataset with ground truth poses.

D Implementational Details

Our approach requires optimization of two networks *(i)* Graph Neural Network (GNN) for robust rotation averaging optimization based on message passing strategy [9, 12], *(ii)* Multi-layer Perceptron (MLP) network optimization for neural multi-scale scene representation. Our GNN architecture for pose optimization is inspired from Purkait *et al.* [12] FineNet, whereas the MLP based rendering network is similar to Mip-NeRF [2]. We train the GNN pose-network in a supervised setting using 1200 synthetic view-graphs proposed by Purkait *et al.* [12] as described in Sec.(B.1). Later, we use the pre-trained pose network for our overall optimization. We implemented our approach using JAX [3] and simulated our code on a 32 GB Nvidia V100 GPU computing machine. First, we train the pose network for 250 epochs with the learning rate 5×10^{-5} and a weight decay of 10^{-4}. To overcome the overfitting problem, we drop a fraction of edges (0.25) in the view-graph input to the pose network. The β parameter in Eq. (7) loss function, in the main paper, is set to 0.1 at train time. Also, the value of parameter b used in Eq. (9) is kept fixed at 10. The MLP model for learning the multi-scale scene representation is trained for 100k iterations for every scene using the Adam optimizer [8] with a batch size of 4096 rays and a learning rate varying logarithmically from 5×10^{-4} to 5×10^{-5}. Similar to [2], we use "coarse" and "fine" sampling schemes with 128 samples for rendering.

Baselines. As discussed in the main paper, we have defined three new baselines to test against our method. These baselines are based on combining the methods that are capable to solve pose errors and multi-scale issues separately. The first baseline (Base A) involves directly combining the Mip-NeRF [2] multi-scale modelling with the encoding scheme presented in BARF [10] along with updating the input poses by making them trainable. This

is based on the simple intuition that as both are working individually for different use-cases (pose error and multi-scale issues), combining them should also work. Given this fails to work, can be observed from the results in Table (1) of the paper, we also use another baseline where we first run BARF [10] on the multi-scale setting with varying resolution and then run Mip-NeRF [2] on the poses recovered from it (Base B). The failure of this baseline (refer Table (3) in the paper) along with previous baseline (Base A), points to the fact that BARF [10] is not able to estimate the correct poses in the multi-scale setting. Even if any kind of annealing is used to combine both, then also the resultant baseline is expected to fail as the object would be incorrectly localized by BARF and therefore, even multi-scale modeling would be of no use. Next, to further deal with this changing focal length problem, we further define new baseline (Base C), where we combine Mip-NeRF [2] with NeRF– [15] by just updating the NeRF– [15] with the Mip-NeRF [2] positional Integrated Positional Encoding(IPE) scheme. It also fails in localizing the object correctly, again can be observed from Table (1) in the paper. The failure of all these three baselines to deal with both pose and multi-scale issues simultaneously proves the importance of our approach *i.e.* even though independently both the issues (multi-scale problems and pose errors) can be dealt efficiently with existing algorithms, it is not straightforward to solve them simultaneously.

References

[1] Khurrum Aftab, Richard Hartley, and Jochen Trumpf. Generalized weiszfeld algorithms for lq optimization. *IEEE transactions on pattern analysis and machine intelligence*, 37(4):728–745, 2014.

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul 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.

[3] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[4] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[5] Venu Madhav Govindu. Robustness in motion averaging. In *Asian Conference on Computer Vision*, pages 457–466. Springer, 2006.

[6] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *International journal of computer vision*, 103(3):267–305, 2013.

[7] Rajbir Kataria, Joseph DeGol, and Derek Hoiem. Improving structure from motion with reliable resectioning. In *2020 International Conference on 3D Vision (3DV)*, pages 41–50. IEEE, 2020.

[8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[9] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36 (4), 2017.

[10] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5741–5751, 2021.

[11] 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. *Communications of the ACM*, 65(1):99–106, 2021.

[12] Pulak Purkait, Tat-Jun Chin, and Ian Reid. Neurora: Neural robust rotation averaging. In *European Conference on Computer Vision*, pages 137–154. Springer, 2020.

[13] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *European Conference on Computer Vision*, pages 623–640. Springer, 2020.

[14] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[15] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf–: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021.