Enhanced Stable View Synthesis Supplementary Material

Nishant Jain*
Indian Institute of Technology
Roorkee, India
njain@cs.iitr.ac.in

Suryansh Kumar*†
ETH Zürich
Switzerland
sukumar@ethz.ch

Luc Van Gool
ETH Zürich
Switzerland
vangool@ethz.ch

Abstract

This draft accompanies the main paper. It provides more experimental results showing the suitability of our proposed approach. Furthermore, it discusses the graph neural network-based multiple rotation averaging and our software implementation details.

1. Synthetic Objects

We further evaluate our proposed method on a synthetic object-centric dataset to investigate its advantages in cases where all points lie within a threshold distance from the camera. The aim is to examine whether integrating a monocular depth with the estimated stereo structure is helpful for these cases. We use the same setup as SVS [11] for evaluation, separating ten images as target novel images and use the remaining 39 as the source images for evaluating interpolation and extrapolation. Table 1 compares all the baselines and our method on this dataset. It consists of results for both view interpolation (left value) and extrapolation setups (right value) for this dataset as done in the SVS paper [11]. It can be observed that even in this case, which doesn't involve *far-away* points, our method is either similar in performance or performs better, especially in terms of PSNR values. Also, performance gap on the extrapolation task is marginally higher than the interpolation counterpart, for the PSNR values. This further highlights the importance of our method for the nearby region where we try to maximize the consistency between RGB-D features and stero-estimated projection of image features, for places where the monocular network is highly confident.

2. Scene-Agnostic Model

We analyze the scene-agnostic version of our approach and SVS, also comparing with the FVS [10] method. The model is trained on scenes corresponding to training data and then is directly evaluated on a disjoint set of test scenes without any tuning. Table 2 shows the results for this version on the four scenes of the Tanks and Temples data set used in the paper, namely *truck*, *M60*, *playground* and *train*, where the model is trained using the 15 other scenes from this dataset. It can be observed that our approach can offer significantly better results, even in the scene-agnostic setup, when compared with SVS and FVS. Also, for both our method and SVS, the results are improved from scene-specific finetuning compared to the scene-agnostic setup, which can be observed by comparing the statistics presented in Table 2 here and Table 1 in the main paper.

3. Graph Neural Networks for MRA

We now discuss our pose refining scheme inspired from NeuRoRA [8]. It uses a Message Passing Neural Network (MPNN) to predict robust poses given a completely initialized view graph. Given the estimated relative rotations using an SFM algorithm, they are used to initialize absolute rotations by fixing a source vertex as the frame of reference and then calculating absolute rotation of each vertex w.r.t. this frame by traversing along the minimum spanning tree. This is followed by a cyclic consistency check to remove outliers. Finally, we have the initialized observed relative rotations and initialized absolute rotations. These comprise a completely initialized view-graph.

Now, for each node k in this graph, with neighbouring set denoted by $\mathcal{Q}_k$, the state of this node at step t, denoted by h_k^t, is generated by processing the aggregated signal feature s_j^t it receives from all the nodes $v \in \mathcal{Q}_k$ and its state at step $t-1$:

$$h_k^t = \rho(h_k^{t-1}, s_k^t) \tag{1}$$

where ρ is some function to process these features jointly. The aggregated signal s_k^t is just a processed combination of updated states of edges corresponding to this node k:

$$s_k^t = \psi_{i \in \mathcal{Q}_k}^a \psi^b(h_k^{t-1}, h_i^{t-1}, r_{ij}) \tag{2}$$

*Equal Contribution
†Corresponding Author (k.sur46@gmail.com)

	65			106			118		
	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑
FVS [10]	30.1/25.2	0.03/0.07	0.97/0.95	32.3/27.1	0.03/0.08	0.95/0.93	34.9/28.7	0.02/0.07	0.97/0.94
SVS [11]	31.9/26.1	0.02/0.06	0.97/0.95	33.8/29.7	0.02/0.04	0.98/0.95	36.7/30.8	0.02/0.05	0.97/0.96
Ours	32.4/26.9	0.03/0.07	0.98/0.96	34.3/30.6	0.02/0.03	0.98/0.96	37.1/31.3	0.02/0.05	0.97/0.96

Table 1. Performance comparison with on DTU dataset [5]. We use the popular metrics i.e., PSNR, LPIPS and SSIM for the comparison.

	Truck			M60			Playground			Train		
	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑
FVS [10]	21.9	0.14	0.84	15.8	0.32	0.77	21.7	0.21	0.83	17.3	0.28	0.75
SVS [11]	22.2	0.16	0.85	18.4	0.24	0.79	22.4	0.20	0.83	17.3	0.21	0.79
Ours	**23.4**	**0.14**	**0.88**	**19.6**	**0.23**	**0.85**	**22.6**	**0.18**	**0.89**	**19.2**	**0.16**	**0.84**

Table 2. Performance comparison with state-of-the-art methods on Tanks and Temples dataset [7] in a scene-agnostic setup. We use the popular metrics i.e., PSNR, LPIPS and SSIM for the comparison.

where, ψ^b is a processing function responsible for updating the signal accumulated from the edge between nodes i and k, r_{ij} is a feature representation for this edge. This is followed by a differentiable operation ψ^a, which can be interpreted as some activation function. For our case, both and ψ_b are concatenation with 1D convolutions and a ReLU activation. Please refer [4, 8] for further details.

Pose-refining GNN. Given the completely initialized view graph, we denote the rotations corresponding to its vertices as $\tilde{R}_i$. Also, the edges of these view graph comprise the relative rotations between the 2 nodes. The edge feature r_{ij} described above is the calculated using these observed relative rotation between the two nodes denoted as $\tilde{R}_{ij}$. The input to the GNN is the set of rotations $\tilde{R}_i$ and the edge feature r_{ij}. This edge feature is calculated as the discrepancy between the initialized absolute rotations $\tilde{R}_i$ and observed relative rotations $\tilde{R}_{ij}$ as follows:

$$r_{ij} = \tilde{R}_j^{-1} \tilde{R}_{ij} \tilde{R}_i \tag{3}$$

This leads to a supervised learning problem for the GNN, where, using the input graph denoted as $\{\tilde{R}_i, r_{ij}\}$, the aim is to estimate the absolute rotations $\hat{R}_i$ as close as possible to the correct rotations $\{R_i\}$ in the source node frame:

$$\{\hat{R}_i\} = \mathcal{G}(\{\tilde{R}_i\}; \Theta) \tag{4}$$

where Θ denote the pose-GNN parameters. The network is trained for this setup using the rotation averaging loss described in the paper. Specifically, the goal is to minimize the discrepancy between observed relative rotations $\tilde{R}_{ij}$ and estimated relative rotations $\hat{R}_j \hat{R}_i^{-1}$. Given only relative rotations are used in this loss function, this it might be same even if any constant angular deviation to the predicted rotations. Thus following NeuRoRA [8], we also add a weighted regularizer term to learn a one-to-one mapping between inputs and outputs which minimizes the discrepancy between initialized absolute rotations and predicted absolute rotations. This leads to the following aggregated cost function $\mathcal{L}_{mra}$ for a given graph with $\mathcal{E}$ denoting its edge set and $\mathcal{V}$ denoting the set of nodes:

$$\mathcal{L}_{mra} = \sum_{\mathcal{E}_{ij} \in \mathcal{E}} d_Q(\hat{R}_j \hat{R}_i^{-1}, \tilde{R}_{ij}) + \beta \sum_{\mathcal{V}_i \in \mathcal{V}} d_Q(\hat{R}_i, \tilde{R}_i) \tag{5}$$

where d_Q is some distance metric between two rotations. We also follow quaternion representation for these rotations similar to NeuRoRA [8].

4. Adapting to other methods

To further show the effectiveness of our joint feature estimation and pose updating proposition, we experimented with other popular frameworks, namely IbrNet [3] and S-IbrNet [2], on two scenes of the Tanks and Temples dataset namely Truck and Playground (P.G.), in our setup. This is done by updating their feature generation modules and integrating our pose-refining module for joint optimization. Table 3 shows the PSNR values for this experiment showing numbers for these methods before and after (E-IbrNet, E-SIbrNet) the integration and also compares these results with the method proposed in the paper. It can be observed that both the methods (IbrNet, S-IbrNet) have their PSNR values improved significantly (around 1.7, 0.9 on average across the two scenes) and finally, our proposed method in the paper performs the best on both the scenes.

	IbrNet [3]	S-IbrNet [2]	E-IbrNet	E-SIbrNet	Ours
Truck	19.7	22.5	21.9	23.6	**24.1**
P.G.	22.2	23.1	23.3	23.7	**23.9**

Table 3. PSNR comparison. E-IbrBet and E-SIbrNet show the results when our approach is put to [3] and [2] network design.

5. Ablation on depth threshold (σ)

For all the experiments, we have set the depth threshold value based on our empirical observations. We extensively

analyzed this quantity on the Tanks and Temples dataset and have set it as 0.66 (relative to median depth) for all the datasets used in the paper. Table 4 shows the analysis of PSNR values on two scenes of Tanks and Temples dataset (Truck, M60) for various values of this quantity (relative to the median depth). It can be observed that the best results are at the value of 0.66, thereby providing justification for the selected value.

σ	0.25	0.50	0.66	1.0	1.25	1.5	2.0
Truck	23.4	23.8	**24.1**	23.6	23.6	23.5	23.5
M60	20.4	20.3	**20.8**	20.4	20.5	20.2	20.3

Table 4. PSNR values for ablation on the depth threshold value.

6. Training and Evaluation details

We now discuss the implementation details involving the training and evaluation setup along with values for parameters involved in our approach. The training is performed on the tanks and temples dataset for our method, SVS [11] and FVS [10]. Then, for the results on all the scenes corresponding to various datasets discussed in the paper, we tune our method and SVS for scene-specific Network fine-tuning as discussed in SVS [11]. For the baselines including NeRF++ [14], SC-NeRF [6] and Point-NeRF [13] also require per-scene fitting. This scene-specific training/tuning involves using a source image set of that scene for learning and a disjoint test set of images corresponding to the same scene for evaluation. For the scene agnostic scenario, trained models on tanks and temples are directly evaluated on the test set of the scene. Also, the $\mathcal{L}_{rgb}$ loss term, used in the Eq.(13) in the main paper, corresponds to the perceptual loss described in Eq.(6) in the SVS paper [11].

Architecture details. The monocular depth prediction is a DPT [9] architecture trained on Omnidata [1]. The network for predicting confidence score for each pixel is just-another head starting from fifth last layer of the depth prediction network with the same architecture as that of the depth prediction head. This is trained while keeping the depth prediction network as frozen. Note, the complete architecture is just used for obtaining depth and confidence per pixel and then is linked with rest of the pipeline. Also, the confidence weighted loss involves l_2 regularization of the predicted weights to avoid the solutions, where each/most of the weights are assigned to a single/some pixels. The network $\mathcal{F}_\theta$ for projecting monocular features consists of a ResNet-50 architecture equipped with the Channel Exchanging Layers [12]. The functions ϕ_α, ϕ_β and ϕ_μ are each 3 layered CNN networks followed by a BatchNorm layer and a ReLU activation. For the SVS features, the architecture used in the SVS paper is followed involving a U-Net for encoding images. As discussed in the paper, the rendering network is also a U-Net architecture same as in SVS paper [11].

Hyperparameter details. The training is performed using an Adam optimizer setting learning rate to 10^{-3}, β_1 to 0.9, β_2 to 0.999 and ϵ to 10^{-8}. The model is trained for 600,000 iterations on the training set with batch size of 1 and 3 source images sampled per iteration, following the SVS setup [11]. The tuning on the testing scene is carried out for $100,00$ iterations. The confidence threshold parameter τ is set to 0.05. The predefined depth threshold (σ) for applying Eq.(8) in the main paper is two-thirds the median depth of the scene.

References

[1] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10786–10796, 2021. 3

[2] Sun et al. Learning robust image-based rendering on sparse scene geometry via depth completion. In *CVPR*, pages 7813–7823, 2022. 2

[3] Wang et al. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, pages 4690–4699, 2021. 2

[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. 2

[5] Peter Hedman, Tobias Ritschel, George Drettakis, and Gabriel Brostow. Scalable inside-out image-based rendering. *ACM Transactions on Graphics (TOG)*, 35(6):1–11, 2016. 2

[6] Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-calibrating neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5846–5854, 2021. 3

[7] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. 2

[8] Pulak Purkait, Tat-Jun Chin, and Ian Reid. Neurora: Neural robust rotation averaging. In *European Conference on Computer Vision*, pages 137–154. Springer, 2020. 1, 2

[9] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 3

[10] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *European Conference on Computer Vision*, pages 623–640. Springer, 2020. 1, 2, 3

[11] Gernot Riegler and Vladlen Koltun. Stable view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer*

Vision and Pattern Recognition, pages 12216–12225, 2021. 1, 2, 3

[12] Yikai Wang, Wenbing Huang, Fuchun Sun, Tingyang Xu, Yu Rong, and Junzhou Huang. Deep multimodal fusion by channel exchanging. *Advances in Neural Information Processing Systems*, 33:4835–4845, 2020. 3

[13] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022. 3

[14] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 3