RANRAC: Robust Neural Scene Representations via Random Ray Consensus Appendix

Benno Buschmann[1,2], Andreea Dogaru[2], Elmar Eisemann[1], Michael Weinmann[1], and Bernhard Egger[2]

[1] Delft University of Technology, Netherlands
[2] Friedrich-Alexander-University Erlangen-Nürnberg, Germany

1 Overview

In the scope of this supplemental, we provide more details regarding our implementation of RANRAC for LFNs [11] and NeRFs [6], as well as the implementation of RobustNeRF [9] used for benchmarking. Furthermore, we present more information on the RANRAC hyperparameters for the application to LFNs and NeRFs. Finally, we provide a more detailed qualitative evaluation that demonstrates the potential of our approach with respect to different types of inconsistencies for different scenes.
Project Page: https://bennobuschmann.com/ranrac/

2 RANRAC for LFNs

2.1 Implementation Details

For the inference with LFNs we used 500 iterations with a learning rate of 0.001 and an exponential schedule. Regarding all other aspects, we follow the implementation published by the authors of LFNs [11]. The integration of the parallel inference of the different sample sets allows for a faster inference of the randomly sampled hypotheses, as the individual hypotheses, their inference, and their evaluation do not depend on each other. On an A100 GPU with 40 GB of RAM, all 2048 hypotheses can be inferred in parallel resulting in a total runtime of about a minute. We also performed tests on an Nvidia RTX 3060 GPU with 12 GB, where the inference is split into batches of 512 hypotheses.

2.2 Experimental Hyperparameters

We provide experimental results regarding the effect of different hyperparameter choices of RANRAC by varying one parameter and keeping all others fixed to the designated optimum to observe the isolated influence. If not specified otherwise, the reported results are based on 50 randomly selected images per class. We used reasonable amounts of occlusion between 20% and 30%.

Effect of the number of initial samples – As discussed, the amount of initial samples taken is a trade-off between the probability of finding enough clean(er) sample sets, and the reconstruction quality from those samples. In Figure 1, we provide insights on the relationship between the reconstruction quality and the number of clean samples. As expected, quality increases with more clean samples. Furthermore, the number of samples required for meaningful reconstructions varies between datasets, which is explained by the average amount of high-frequency details of a certain class, as well as its general complexity, and the strength of LFNs on a specific class. Furthermore, we observe that the quality greatly depends on where the samples land, as illustrated by the 99th percentile. This supports our finding that not only a sufficiently clean sample set but also one leading to meaningful reconstructions has to be found.

Fig. 1 shows the reconstruction performance of RANRAC for different classes and numbers of samples while keeping the other parameters (i. e. margin, iterations, amount of inconsistency) fixed. It is well visible that there is not a single optimum across classes. For the plane class, a small sample size is enough. The quality is decent for ∼40-120 samples, it drops quite drastically above that limit. In contrast, the chair class needs more samples, i.e. ∼60-280 samples, and the quality even keeps slightly improving. The car class lies in between, i.e., at ∼60 samples a decent performance is reached and the performance still slightly, but not steadily, improves until ∼160 samples, and then slowly decreases again. The difference is explained by attributes of the classes. The chair class contains a lot of zoomed-in observations, closer to the object, opposed to the plane and the car class. This results in more high-frequency details which need to be captured. LFNs have difficulties reconstructing the chair class, even in clean environments. Furthermore, the object covers a larger part of the image compared to the other classes. Not all occlusion attributes can be fixed at once. Although, object occlusion and image occlusion are controlled, the occlusion-to-object ratio is much lower on average for the chair class (28%) compared to car (35%) and plane (62%). Thereby, an additional sample is more likely to be clean and beneficial for the former two classes.

As the reconstruction quality is already degenerate for plane, and stagnating for car, we did not perform tests with more than 280 samples. When tuning per class is not feasible, we propose choosing 90 samples. We used this for the evaluation of all classes in the main paper. The choice is within the identified optimal range of the plane and car class, and the performance penalty for the chair class is not too significant (less than 1dB). A choice on the lower end of feasible options for the examined amount of occlusion has additional benefits: Taking fewer samples reduces run-time and is desirable for environments with a higher amount of occlusions. We, however, do not claim that this choice is optimal for all classes and environments.

Effect of the margin – The selection of a reasonable choice for the inlier margin is bound by two factors. For tight margins, some clean samples are wrongfully excluded due to small high-frequency variations that are not represented by the initial down-sampled reconstruction. If it is too loose, some occluded sam-

ples are no longer separated. In Fig. 3, we show the consensus sets generated by RANRAC for different margins. Note that it is expected to have some occluded samples being explained by the model.

In Fig. 2, we compare the reconstruction quality of RANRAC for different margins. A maximum reconstruction performance is observed at a margin of 0.25 in terms of Euclidean distance in a color space normalized to $(-1, 1)$. It is further observed that the plane class is more resilient to a lower margin than the others and the SSIM even starts reducing at a margin of 0.15, most likely due to the lower amount of high-frequency variations within this class. As the looser margin is not required to include any of those details, the inclusion of slight amounts of occlusion caused by it is visible earlier.

Effect of the number of iterations – As expected, the quality improvement via additional random hypotheses (iterations) behaves approximately logarithmic (Figure 2). Powers of two are generally desirable for the parallel inference. We settled on 2048 iterations, as there is a significant improvement compared to 1024 iterations, and no improvement with 4096 iterations. Any measured decrease with more iterations can only be caused by noise. For completeness, we added another test row with even fewer iterations (Table 1), though, the performance implications are negligible, 512/2048 iterations can be inferred in parallel with 12/40GB of VRAM. As expected the downward trend continues, but slowly enough for fewer iterations to potentially be useful for certain applications.

Iterations	512	256	128	64	32	16	8	4	1
Plane	24.93	24.79	24.35	24.08	23.87	23.29	23.25	22.62	21.45
Car	24.05	24.14	23.85	23.85	23.45	23.28	22.66	22.53	21.92
Chair	18.97	18.78	18.84	18.76	18.66	18.46	18.28	18.12	17.69

Table 1: Additional results on lower iteration counts (↑ PSNR in dB, other hyperparameters fixed as reported)

2.3 Generation of Occlusions

For the ShapeNet data [1], the occlusions need to be simulated in a randomized but controllable way, and they have to represent the attributes of real-world occlusions. Variation in position and color is required, while the shape is not important, as our algorithm randomly draws the samples according to a uniform distribution, and does not rely on semantics. However, it is the most challenging environment to have a single connected patch of occlusion, as the information loss about the occluded object part is less curable. As this is also the most common type of occlusion under real-world conditions, we particularly focus on demonstrating the robustness of our approach to this scenario. Furthermore, using a monotone color as occlusion would oversimplify the problem and allow for unrealistically high choices for the margin. The reason is that a single monotone color is fairly easy to separate from the object, as long as it is not the same color

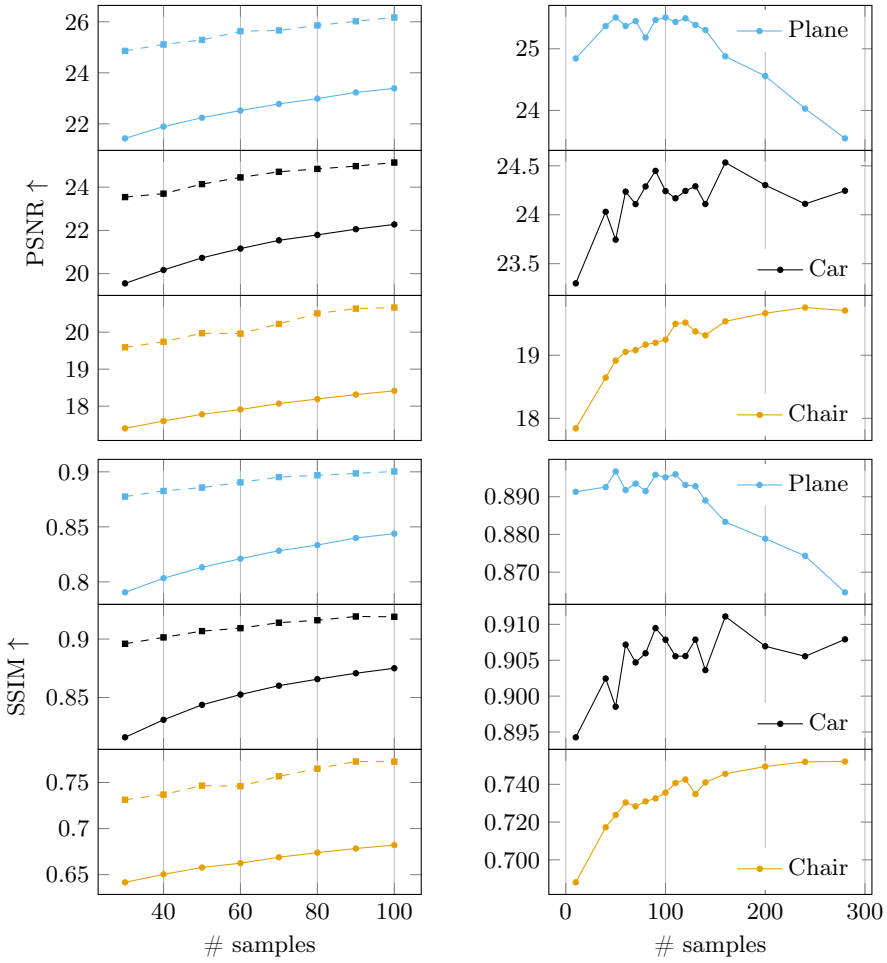

Fig. 1: On the left, we show the reconstruction performance of LFNs on uniform-randomly down-sampled clean images. For every class and amount of samples, randomly selected instances have been inferred with 512 different draws of the samples each. Solid lines represent the average reconstruction performance, while dashed lines represent the average per-instance 99th percentile of the reconstruction performance, to emphasize the influence of the position of the samples. On the right, we show the reconstruction quality of RANRAC in the context of LFNs with varying amounts of samples.

as the object. Therefore, we occlude with noise patches, where each color channel is sampled separately from a normal distribution with decently large standard deviation. This results in randomly varying color across the patch and presents a bigger challenge, as it introduces distractions more similar to the object and more likely to be embedded in a local optimum of the latent space. The final

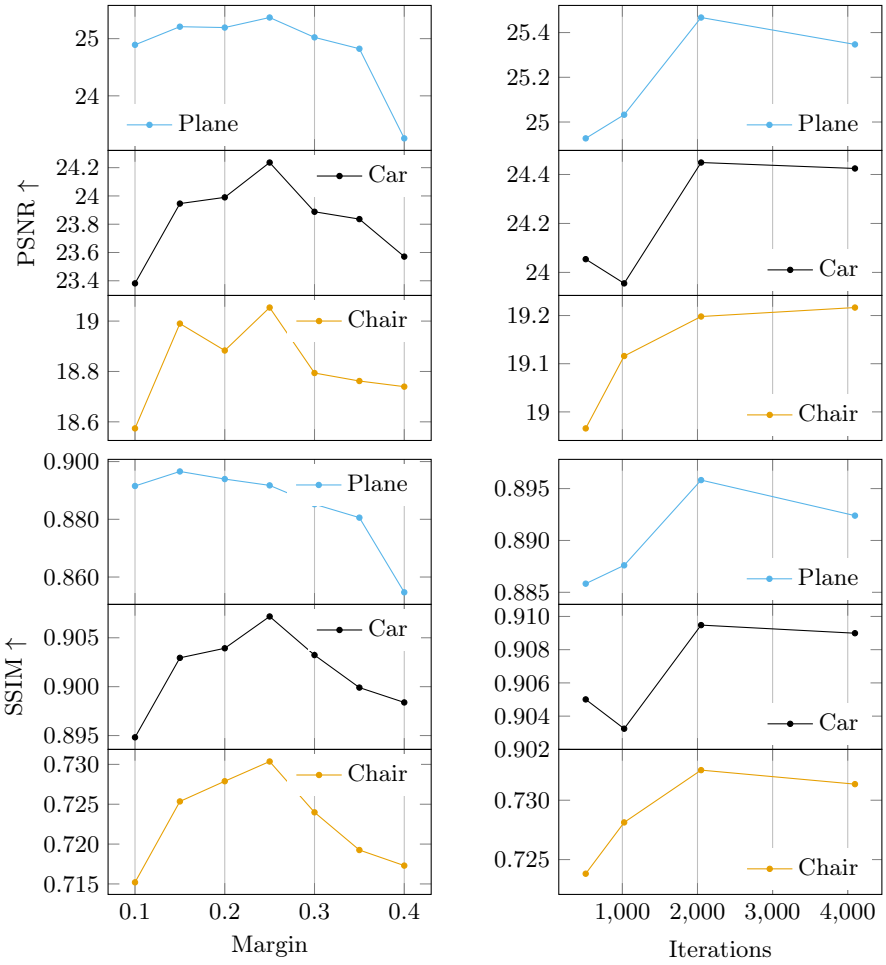

Fig. 2: Reconstruction performance of RANRAC with different numbers of iterations (right) and different inlier margins (left).

challenge is the placement of the occlusions. In addition to being randomized, the placement should model the common type of partial occlusion. This reduces noise in the results due to completely occluded objects which are just unrecoverable. Furthermore, simulating the occlusion by smaller patches which just cover the object itself would oversimplify the problem, as the shape would still be well reconstructable. The correct way of simulating the occlusion is therefore to place an occlusion patch at about the edge of the object, which has the right size to cover the desired relative part of the object. This still has to be done in a randomized way. A good practical realisation is drawing the coordinates of the center point of the occlusion patch independently from a distribution that

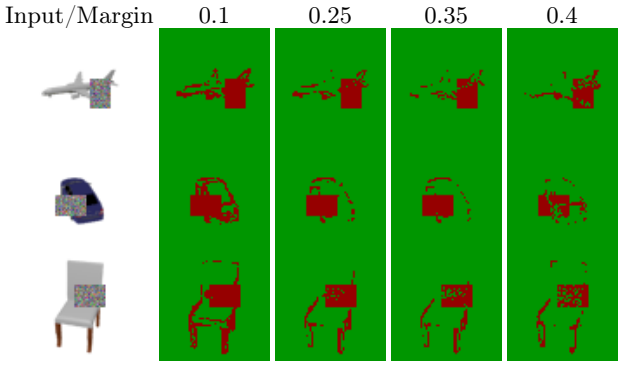

Fig. 3: Examples of the final consensus set produced by RANRAC for different margins (outliers are highlighted in red, inliers in green). Both effects of a larger margin are well visible: The increasing amount of occlusion that is not separated and the increasing amount of high-frequency details that are preserved.

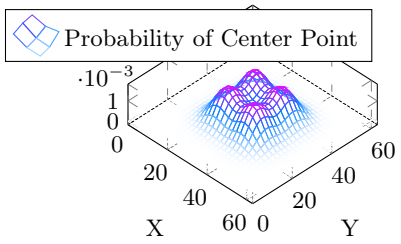

Fig. 4: Illustration of the probability of certain center points of the occlusion patch when drawn as described to achieve the desired partial occlusion of the objects.

consists of two symmetric Gaussian spikes at about the expected object borders (Figure 4). Their mean is used to control the desired amount of object occlusion.

The size of the patch is drawn from independent normal distributions for the size in both dimensions, the mean is used to control the desired amount of image occlusion. Finally, the accomplished relative occlusion of the object is measured, and if it does not fall into the specified bandwidth of object occlusion, e.g. 20%-30%, a new patch is generated. This could be considered as rejection sampling of the desired distribution.

3 RANRAC for NeRFs

3.1 Implementation Details

When applying RANRAC in the context of NeRF-based reconstruction, we use the Instant-NSR implementation [3] with Adam optimizer, a learning rate of 0.01, a distortion loss lambda of 0.001, and a multi-step learning rate schedule with gamma 0.33. The parameters vary for the RANRAC hypothesis evaluation,

where a shallow fit is adequate, and the final inference. Within the hypothesis evaluation, 4000 training iterations are sufficient; for the final inference, 20,000 training iterations are used. The hash grid is reduced to a maximum size of 2^{18}, to ensure a decent performance with a wider range of GPUs (due to cache alignment [7]). We use mixed precision training. Regarding the remaining parameters of the geometry and texture representation, we used the default parameters used by the authors of Instant-NSR. On acceptance, we will publish all configuration files together with the source code to provide all of the details and facilitate reproducing our results. The run-time is about a minute per iteration and less than five minutes for the final inference on an Nvidia RTX 4090 GPU.

Regarding the RANRAC hyperparameters, a pixel margin of $\epsilon_{pix} = 0.15$ in terms of Euclidean distance in a color space normalized to $(0, 1)$ worked well for the determination of actual artifacts. We consider an entire observation as an inlier based on a margin of 90%-98% of its pixels being inliers, which proved to be a good choice to separate minor artifacts (due to the sparse sampling) from artifacts caused by actual inconsistencies. For the dataset with less heavy occlusion, 50 iterations were sufficient. For the dataset with heavier occlusion, we reported results after 500 iterations in the main paper. But even after less than 100 iterations, a significantly improved quality can already be noted. For other inconsistencies we evaluated with 100 iterations.

3.2 Further Implementation Details regarding Baseline (RobustNeRF) for Benchmarking

To allow a fair, isolated comparison of the robustness method, we implemented the losses introduced by RobustNeRF [9] on top of the Instant-NSR method [3] used for our implementation. Even though the authors of RobustNeRF claim the generality of their method, some practical challenges and incompatibilities hamper the transfer. We resolved any appearing conflict as natural as possible and actively refined their method where it was necessary to achieve a decent performance in the provided scenario and allow a decent integration into fast NeRF variants. The details are presented in the following paragraphs.

Patch-based Sampling – A key component of the robust losses is the patch-based sampling to evaluate the inliers in a neighbourhood. This leads to a conflict with the occupancy grid used by Instant-NSR [3] or iNGP [7]. When only updating a small amount of patches, compared to thousands of randomly distributed locations, the occupancy estimate naturally becomes more biased and less accurate. Depending on the sampled patches, it can happen that none of them are considered as occupied according to the grid, and, therefore, no gradient is available and no update takes place. This can lead to divergence without recovery. We selected only converging runs of RobustNeRF when benchmarking.

Furthermore, the amount of patches used as reported by the robustNeRF authors is not feasible for tiny-cuda-nn, we had to reduce it from 64 to 16 to allow the model to fit into the 24GB VRAM of an Nvidia RTX 4090 GPU, implying a slightly more stochastic gradient descent.

Remark on Mask Computation – The authors mention in the paper that the patch mask is computed based on the smoothed neighbourhood inlier mask of the second step [9]. However, in the source code they published [8], the patch mask is computed based on the per-pixel inlier mask of the first step. The difference is naturally rather negligible. We still considered it worth noting, and report that we applied it as provided by their implementation.

Single object reconstruction – When directly applying the method as described in the paper to the single-object reconstruction scenario, a reconstruction of the white background is obtained, and the object is wrongfully considered as the distraction and removed. For that reason, we had to use a higher margin of 0.8 instead of median, to allow their method to work at all for this task.

Finally, we used mixed precision for ours and full precision for theirs, as with their method, sometimes no sample is considered as being occupied and an inlier, and therefore no gradient update is possible. This does not play well with mixed precision in pytorch lightning. If at all, this difference benefits their method.

3.3 Details regarding the Occluded Watering Pot Dataset

The watering pot dataset was captured to evaluate the photo-realistic 360-degree reconstruction of a single occluded object from lazily captured real-world images (resembling a typical cultural heritage application). We capture clean and occluded images (Fig. 5) from the same range of perspectives using a smartphone camera. The clean images are randomly split into train and test images. The occluded images are partially added to the training images to achieve the desired amount of distraction. Foreground masks containing object and occlusion are automatically obtained using SegmentAnything [4] with a simple box query. The camera parameters are obtained using COLMAP [10]. Erroneous estimates of camera parameters or masks are conveniently dealt with by our method, making the entire object reconstruction pipeline robust. The 4000×2252 images are down-scaled by a factor of five before training.

In Figure 6, we provide a comparison of the reconstruction quality achieved based on different methods. The artifacts of occluded perspectives are well visible and removed by the robust methods. RobustNeRF's rigorous exclusion of non-photo-consistent samples leads to artifacts for view-dependent details while our RANRAC approach allows a seamless reconstruction even for such details of view-dependent appearance.

3.4 Inconsistent Camera Parameters

Structure-from-motion pipelines are a powerful tool for pose estimation. It is, however, common that the registration fails for some images. These images will then induce artifacts into the neural field reconstruction. With RANRAC we can simply separate all such perspectives from the training data without removing other valuable information like view-dependent appearance and fit the model to a clean set of observations. Complementing the evaluation in the main paper, we present additional qualitative results (Fig. 7) as well as difference images (Fig.

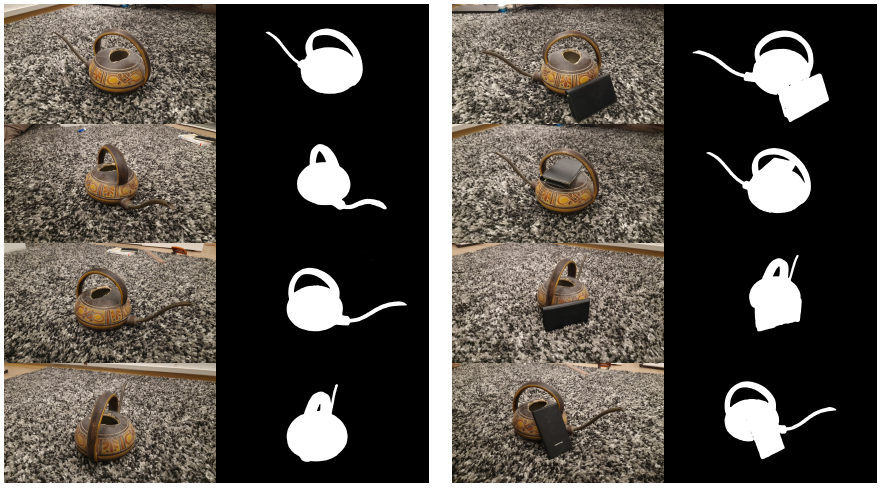

Fig. 5: Examples of the clean (left) and occluded (right) images with foreground masks from our Occluded Watering Pot dataset

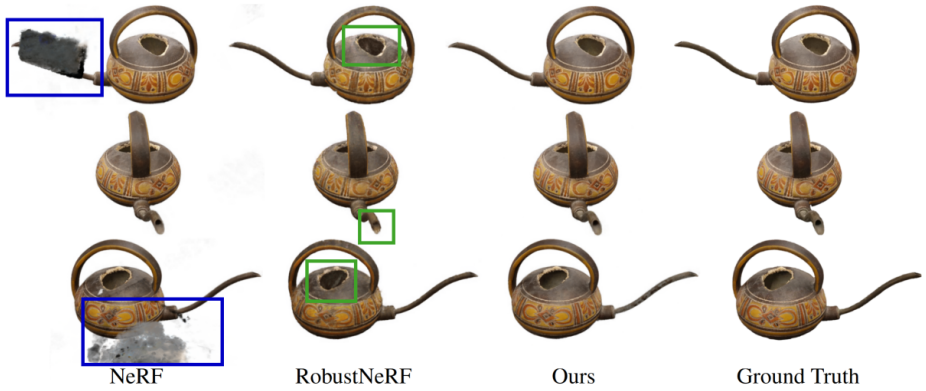

Fig. 6: Comparison of reconstruction quality achieved by different methods: Whereas the occlusions lead to well-visible artifacts (blue) in the reconstructions based on NeRF, these artifacts are completely removed by our RANRAC approach. Additionally, the slight inaccuracies of robustNeRF due to the rigorous exclusion of valid samples can be noted (green). They are best visible for view-dependent effects at concavities. Meanwhile, RANRAC preserves these details.

8) to provide an intuitive assessment regarding how the errors behave across the scene.

3.5 Unfocused Perspectives

Within a lazy capture of an object, typically, some of the frames will be out of focus. We also investigate RANRAC's capabilities to deal with such blurred

Fig. 7: Qualitative results for scenes with noisy camera parameters: The floating artifacts in the NeRF reconstructions are well visible and not present in the reconstructions of RANRAC. Additionally, RobustNeRF rigorously enforces photo-metric consistency, leading to the exclusion of view-dependent specular details (Hi-Hats of drums, specular reflections on the water/microphone/steel balls/flower pot, view-dependent appearance of leaves). RANRAC, on the other hand, preserves these details.

Fig. 8: Error visualization for scenes with noisy camera parameters: We observe the global artifacts in the NeRF reconstruction and the local artifacts for scenes with strong view-dependent details in the robustNeRF reconstructions indicated by a reddish highlighting (i.e., the larger the error the better visible it is in the difference image). In contrast, RANRAC does not produce global artifacts and has much lower prediction error for view-dependent details, solely caused by the scene representation itself.

frames in the set of input images. In Figure 9, we complement the results of the main paper with further qualitative results. Additionally, we present difference images in Figure 10 to better illustrate where errors are larger or smaller respectively. For sparse and irregular viewing angles both robust methods struggle separating them from blur resulting in RANRAC's only failure case (Fig. 11). This could be addressed with additional priors.

Fig. 9: Qualitative results for scenes with unfocused images: RANRAC successfully separates the blurred observations from clean ones while RobustNeRF again struggles with view-dependent appearance.

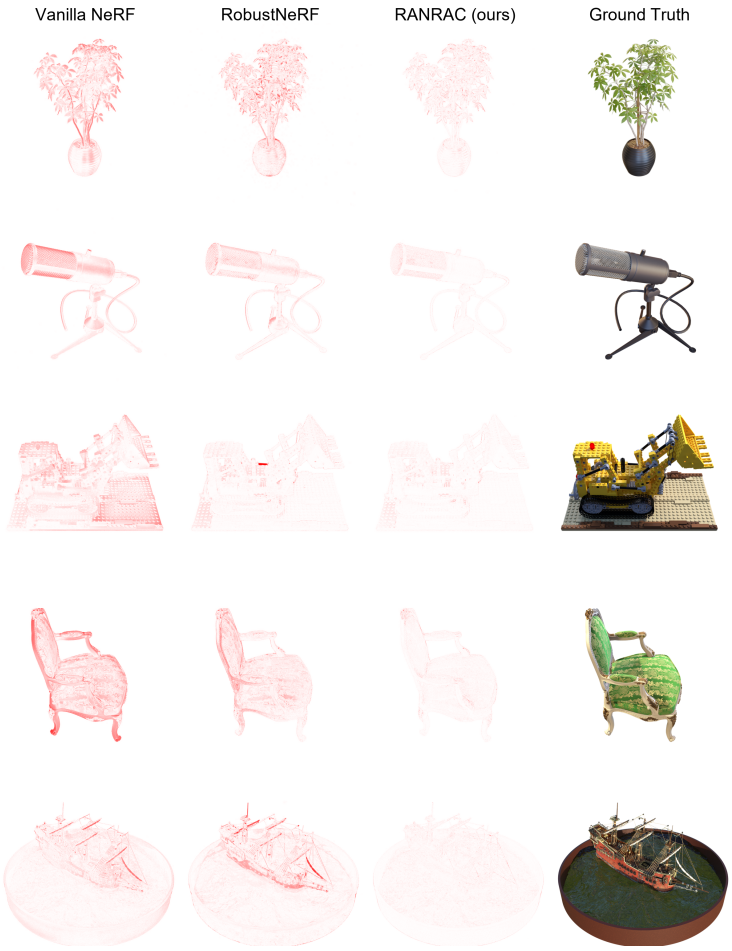

Fig. 10: Error visualization for scenes with blurred perspectives: We observe the global artifacts in the NeRF reconstruction and the local artifacts for view-dependent details in the RobustNeRF reconstructions indicated by a reddish highlighting (i.e., the larger the error the better visible it is in the difference image). In contrast, RANRAC does not produce global artifacts and has much lower prediction error for view-dependent details, solely caused by the scene representation itself.

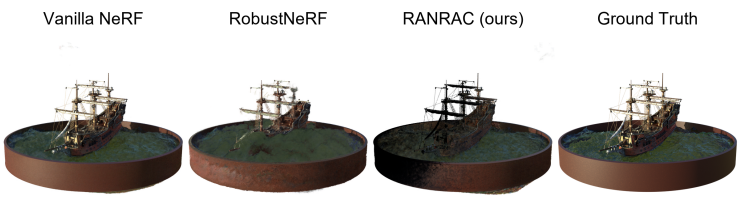

Fig. 11: Both robust methods struggle separating the few perspectives of the ship with low viewing angle from the blurred images. While RANRAC still removes blurred perspectives and achieves a strongly improved reconstruction (Figs. 9, 10), for those few perspectives the appearance is not correctly reconstructed.

3.6 Further Baselines

Complementing our comparison with the state-of-the-art method RobustNeRF [9], we provide a sampled comparison with other methods for further context (Table 2). RANRAC consistently obtains stronger object reconstructions for all analyzed sources of inconsistency. We generated results with Nerfbusters [12], and add the reported results of NeRF-W [5] and Ha-NeRF [2]. For NeRF-W [5, Supp. Table 3] and Ha-NeRF [2, Supp. Table 2], their reported results on *clean* data are worse than RANRAC's results on *inconsistent* data.

PSNR ↑	Wat. Pot Occl.	Lego Blur	Ficus Err. Poses
RANRAC (*ours*)	**27.11**	**34.79**	**31.41**
RobustNeRF [9]	26.83	29.14	23.17
Nerfbusters [12]	23.75	28.24	24.28
Nerf-W [5] (clean)	-	32.89	-
Ha-NeRF [2] (clean)	-	32.23	-

Table 2: Comparison of RANRAC applied to NeRF with additional baselines.

References

1. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: ShapeNet: An Information-Rich 3D Model Repository. Tech. Rep. arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago (2015) 3
2. Chen, X., Zhang, Q., Li, X., Chen, Y., Feng, Y., Wang, X., Wang, J.: Hallucinated neural radiance fields in the wild. In: CVPR (2022) 14
3. Guo, Y.C.: Instant neural surface reconstruction (2022), https://github.com/bennyguo/instant-nsr-pl 6, 7
4. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment anything. arXiv:2304.02643 (2023) 8
5. Martin-Brualla, R., Radwan, N., Sajjadi, M.S.M., Barron, J.T., Dosovitskiy, A., Duckworth, D.: NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In: CVPR (2021) 14
6. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020) 1
7. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. In: SIGGRAPH (2022) 7
8. Sabour, S., Vora, S., Duckworth, D., Krasin, I., Fleet, D.J., Tagliasacchi, A.: Robustnerf github, https://robustnerf.github.io, 2023-11-23 8
9. Sabour, S., Vora, S., Duckworth, D., Krasin, I., Fleet, D.J., Tagliasacchi, A.: Robustnerf: Ignoring distractors with robust losses. In: CVPR (2023) 1, 7, 8, 14
10. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR (2016) 8
11. Sitzmann, V., Rezchikov, S., Freeman, W.T., Tenenbaum, J.B., Durand, F.: Light field networks: Neural scene representations with single-evaluation rendering. In: NeurIPS (2021) 1
12. Warburg*, F., Weber*, E., Tancik, M., Hołyński, A., Kanazawa, A.: Nerfbusters: Removing ghostly artifacts from casually captured nerfs. In: ICCV (2023) 14