NeRF *On-the-go*: Exploiting Uncertainty for Distractor-free NeRFs in the Wild

Supplementary Material

In this **supplementary document**, we first provide additional details in Sec. A, then, we further provide additional experiment results, more thorough ablation studies and performance analysis in Sec. B. We also provide a **supplementary video** where we show additional visual comparisons.

A. Details

A.1. Dataset Details

Synthetic Dataset. We evaluate on the synthetic dataset [13] provided in D^{2}NeRF [52]. This dataset includes five sequences with floating objects in the room generated by Kubric [12]. Upon careful examination, we notice that the training and test images within the *Chair* scene are misaligned in terms of their coordinate systems, therefore we decide to temporarily exclude this particular scene.

RobustNeRF Dataset. As illustrated in the original RobustNeRF, there are unintentional changes throughout the capturing process (both the training and test set) for the dataset, including the tablecloth movement in the *A*ndroid scene and the curtain in the *S*tatue scene, which may adversely affect the performance of SAM-based methods. In contrast, both RobustNeRF [39] and our method can naturally accommodate these unintentional changes.

***On-the-go* Dataset.** *On-the-go* dataset is acquired with an assortment of devices, including an iPhone 12, a Samsung Galaxy S22 and a DJI Mini 3 Pro drone. During the capture of each sequence, the exposure, white balance, and ISO are fixed. This dataset features a wide range of dynamic objects including pedestrians, cyclists, strollers, toys, cars, robots, and trams), along with diverse occlusion ratios ranging from 5% to 30%. This diversity ensures a rich and challenging environment for our assessments. The resolution of images captured by the iPhone 12 and DJI drone(*Drone* sequence) is 4032×3024, whereas the resolution of sequences captured by the Samsung Galaxy S22(*Arc de Triomphe* and *Patio* sequence) is 1920×1080.

A.2. Implementation Details of NeRF On-the-go

Our work is built upon the Mip-NeRF 360 [1] codebase [¶]. In addition to our proposed loss, we keep the original distortion loss and interval loss in Mip-NeRF 360 [1]. We run our method on a server with an AMD EPYC 9554 64-core processor and 4 NVIDIA RTX 4090 GPUs. For each scene, we run 250000 iterations with a batch size of 16384, which typically takes 12 hours to finish. Through our assessment, we observed that our model, after only one hour of training, already demonstrated superior quality compared to RobustNeRF, even after it underwent 12 hours of training. We downsample images by 8x to keep it the same as RobustNeRF (except *Arc de Triomphe* and *Patio* is downsampled by 4x to make it roughly the same as RobustNeRF). We select the dilated sample patches with a size of 32×32 and a dilation rate of 4. The SSIM window size is 5×5. For hyperparameters in loss terms, we set $\lambda_1 = 100, \lambda_2 = 0.5, \lambda_3 = 0.5, \lambda_4 = 0.1$ for all datasets.

A.3. Baseline Details

RobustNeRF [39]. For our own run of RobustNeRF [39], we enable the appearance embedding (GLO) since it delivers consistently better results as illustrated in RobustNeRF [39] as shown in Table 2.

Mip-NeRF 360 + SAM. This is a baseline that we introduce for evaluation. For RobustNeRF [39] dataset, we use an interactive tool[‖] to click each distractor in every image. For *On-the-go* dataset, we pre-identify the dynamic objects' categories and consider this as an oracle for this method. To detect the dynamic object's bounding box, we employed YOLOv8[**] to generate the bounding box for the distractors. Following this, Segment Anything Model (SAM) [22] is applied with the detected bounding box to get the corresponding segmentation. In the absence of a 'robot' class in YOLOv8, we identify the robot in the *S*pot scene by selecting the bounding box encompassing the largest area of yellow. Some imperfect masking results are shown in Fig. A, primarily attributable to factors such as partial observation, reflections of distractors, and ambiguous classifications, like the categorization of a statue as a human.

B. Additional Experiments

B.1. Evaluation

Kubric Dataset [13]. We evaluate on Kubric synthetic dataset provided in D^{2}NeRF [52], with qualitative results shown in Table A. Our performance aligns with RobustNeRF, this is due to saturation on this simple dataset. We include the result of this dataset solely for the sake of a comprehensive evaluation.

Comparison on RobustNeRF Dataset [39]. In this section, we present the results obtained from the *B*abyYoda scene, as summarized in Table B. Our methodology yields improved outcomes compared to the open-source implementation of RobustNeRF. However, these results do not

[¶] https://github.com/google-research/multinerf

[‖] https://github.com/open-mmlab/playground

[**] https://github.com/ultralytics/ultralytics.git

	Car			Cars			Bag			Pillow		
	LPIPS↓	MS-SSIM↑	PSNR↑	LPIPS↓	MS-SSIM↑	PSNR↑	LPIPS↓	MS-SSIM↑	PSNR↑	LPIPS↓	MS-SSIM↑	PSNR↑
NeRF-W [27]	0.218	0.814	24.23	0.243	0.873	24.51	0.139	0.791	20.65	0.088	0.935	28.24
NSFF [24]	0.200	0.806	24.90	0.620	0.376	10.29	0.108	0.892	25.62	0.782	0.343	4.55
NeuralDiff [48]	0.065	0.952	31.89	0.098	0.921	25.93	0.117	0.910	29.02	0.565	0.652	20.09
D^2NeRF [52]	0.062	0.975	34.27	0.090	0.953	26.27	0.076	0.979	34.14	0.076	0.979	36.58
RobustNeRF [39]	**0.013**	0.988	37.73	0.063	0.957	26.31	**0.006**	**0.995**	**41.82**	**0.018**	**0.990**	**38.95**
Ours	0.023	**0.989**	**39.83**	**0.035**	**0.982**	**27.00**	0.016	0.993	39.50	0.039	0.986	38.41

Table A. **Novel view synthesis results on the Kubric Dataset.** The numbers for baseline methods are taken from [39].

Figure A. **Sample Masking Results of Mip-NeRF 360 [1] + SAM.** The predicted dynamic segments are highlighted in blue. Although state-of-the-art methods for object detection and instant segmentation are used with known dynamic object categories, they still have incorrect predictions, overlooked objects, or incomplete segmentation of objects.

	BabyYoda		
	LPIPS↓	SSIM↑	PSNR↑
RobustNeRF [39]	0.20	0.83	30.87
RobustNeRF* [39]	0.31	0.81	29.19
Ours	0.24	0.83	29.96

Table B. **Novel View Synthesis Results on the BabyYoda Scene of RobustNeRF dataset.** RobustNeRF* [39] denotes our own run using the official code release. Our method is superior compared with RobustNeRF* [39], although it does not quite achieve the results reported in the RobustNeRF paper.

quite reach the performance levels reported in the original RobustNeRF paper. We didn't put this result in the main paper because the distractors in this dataset varies across all images, which doesn't fit our setting.

***On-the-go* Dataset.** Additional qualitative results of *On-the-go* dataset are shown in Fig. B. Our method consistently outperforms all baseline methods in various environments. The performance of different baseline methods closely aligns with the sequences depicted in the Table 6. While NeRF-W [27] is capable of removing distractors, it does so at the expense of detail loss. RobustNeRF [39], due to its threshold-based nature, occasionally fails to preserve thin structures. Furthermore, Mip-NeRF 360 + SAM struggles due to the imperfect segmentation, as illustrated in Fig. A.

B.2. Ablation Study

Loss Ablation. To evaluate the effectiveness of our loss functions, we conduct a supplementary loss ablation on a low occlusion scene (*Tree*) as presented in Table C. While

	LPIPS↓	SSIM↑	PSNR↑
(a) w/o $\mathcal{L}_{\text{reg}}$	0.244	0.703	20.19
(b) ℓ_2 in $\mathcal{L}_{\text{uncer}}$	0.240	0.709	20.53
(c) $\mathcal{L}_{\text{uncer}}$ for NeRF	0.354	0.601	18.84
Ours	**0.226**	**0.718**	**20.68**

Table C. **Ablations on Loss Functions.** We compare different loss choices for training on the *T*ree sequence.

Table 4 in the main paper is evaluated on a high occlusion sequence, Table C is evaluated on a low occlusion sequence. We find that for both occlusion scenarios, each component of our method contributes to the overall performance enhancement. Although in scenarios with relatively low occlusion, the design choice (b) still can achieve satisfactory quality except for certain views, the performance drop is more pronounced in high occlusion scenarios. Furthermore, in both occlusion scenarios, we observe that (c) $\mathcal{L}_{\text{uncer}}$ for NeRF exhibits a significant performance decline. This decline can primarily be attributed to our SSIM formulation, which is tailored more toward optimizing uncertainty rather than scene representation.

Dilated Patch Ablation. We continue to test various dilation rates on a low occlusion scene *T*ree in Table F with patch size fixed to be 32×32. We observe that the performance closely resembles that of high occlusion scenes as depicted in Table 3. Notably, unlike in high occlusion situations, a dilation rate of 8 is able to sustain performance without collapsing. Nevertheless, to maintain consistency in hyperparameter settings across all occlusion scenarios, we set the dilation rate at 4.

Due to the space constraints in the main paper, the qual-

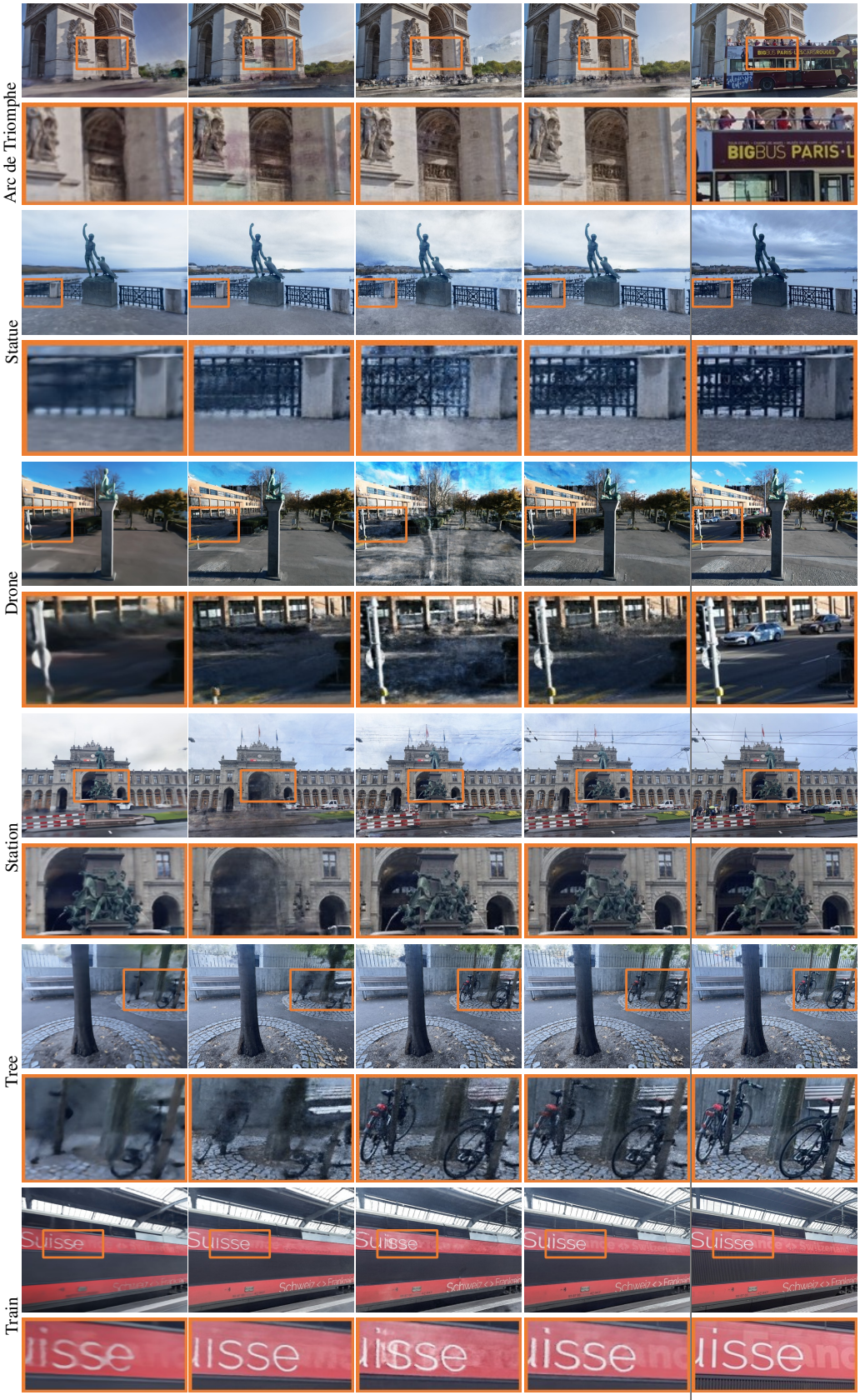

Figure B. **Additional Novel View Synthesis Results on Our *On-the-go* Dataset.** For GT, we show captured test views that might contain some dynamic objects due to restrictions of the capture environment.

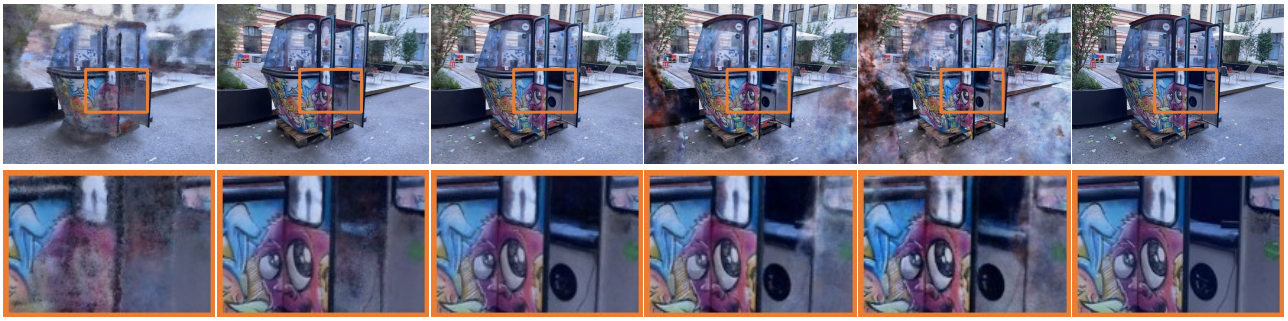

Figure C. **Ablations on Dilation Rate with a Patch Size at** 32×32**.** A dilation rate of 4 results in superior rendering quality.

	LPIPS↓	SSIM↑	PSNR↑
1	0.363	0.592	18.51
2	0.257	0.694	20.07
4 (Ours)	**0.226**	**0.718**	20.68
8	0.235	0.714	**20.69**
16	0.248	0.702	20.37

Table D. **Ablations on Patch Dilation Rates on the Tree Scene.** Comparisons of various dilated rates for the dilation sampling, with a patch size of 32×32.

	LPIPS↓	SSIM↑	PSNR↑
ResNet-50	0.480	0.444	16.16
DINOv1	0.237	**0.720**	21.36
DINOv2 (Ours)	**0.235**	0.718	**21.41**

Table E. **Novel View Synthesis Results with Different Feature Extraction Module.**

itative results of Table 3 are shown in Fig. C. These qualitative results align with the trends observed in Table 3, indicating that a lower uncertainty ratio (< 4) effectively removes distractors but reduces the reconstruction quality, whereas a higher dilation ratio (> 4) tends to reintroduce the distractors due to the loss of local information.

Feature Extraction Module. In this paragraph, we change the feature extractor module $\mathcal{E}$ to Resnet-50 [14] and DINOv1 [4] as detailed in Table E. We find that there are negligible differences between DINOv1 and DINOv2. However, we observe that the Resnet-50 features are less effective in removing distractors. We attribute this difference to the Resnet features' emphasis on color information, in contrast to the DINO features that prioritize instance information, essential for efficient distractor removal.

B.3. Analysis

Our SSIM Formulation. In this section, we will show the mathematical proof that our method can impose a larger uncertainty difference between distractors and static backgrounds. To simplify notation, we denote the $L(P, \hat{P}), C(P, \hat{P}), S(P, \hat{P})$ in Eq. (7) as l, c, s.

Proof. Let l_1, c_1, s_1 represent the luminance, contrast, and structure similarity between the distractor patch and the ground-truth patch. Similarly, l_2, c_2, s_2 represent these similarities for the distractor-free patch and ground truth patch. Therefore, we have the following conditions:

$$\begin{aligned} 0 < l_1 < l_2 < 1, \\ 0 < c_1 < c_2 < 1, \\ 0 < s_1 < s_2 < 1. \end{aligned} \tag{12}$$

Our assumptions in Eq. (12) are directly grounded in the properties proved in the original SSIM paper (Section III.B). In such cases, the similarity between rendered patches and ground truth would naturally decrease. Our empirical results also support this validity: our modified SSIM loss consistently outperforms the original one in various datasets.

To prove that our reformulation in Eq. (8) places greater emphasis on the differences between dynamic and static elements compared to Eq. (7), we need to demonstrate the following inequality:

$$\frac{(1-l_1)(1-c_1)(1-s_1)}{(1-l_2)(1-c_2)(1-s_2)} > \frac{1-l_1 \cdot c_1 \cdot s_1}{1-l_2 \cdot c_2 \cdot s_2}. \tag{13}$$

The left-hand side of this equation of the ratio of our SSIM formulation between distractors and static backgrounds, and the right-hand side is the ratio of conventional SSIM Loss. This can be equivalently expressed as:

$$\frac{(1-l_1)(1-c_1)(1-s_1)}{1-l_1 \cdot c_1 \cdot s_1} > \frac{(1-l_2)(1-c_2)(1-s_2)}{1-l_2 \cdot c_2 \cdot s_2}. \tag{14}$$

Taking the natural logarithm of both sides, we get:

	LPIPS↓	SSIM↑	PSNR↑
Conventional SSIM	0.455	0.459	16.33
Ours	**0.235**	**0.718**	**21.41**

Table F. **Novel View Synthesis Results on the Patio-High Scene of *On-the-go* dataset.**

$$\ln\left(\frac{(1-l_1)(1-c_1)(1-s_1)}{1-l_1\cdot c_1\cdot s_1}\right) > \ln\left(\frac{(1-l_2)(1-c_2)(1-s_2)}{1-l_2\cdot c_2\cdot s_2}\right). \tag{15}$$

We aim to prove that the function $f(x,y,z) = \ln\left(\frac{(1-x)(1-y)(1-z)}{1-xyz}\right)$ is monotonically decreasing for $0 < x,y,z < 1$. Given the function's symmetry across variables, it is sufficient to take the partial derivative with respect to one variable, say x, and show that it is negative. The partial derivative of $f(x,y,z)$ with respect to x is given by:

$$\begin{aligned}\frac{\partial f(x,y,z)}{\partial x} &= -\frac{1}{1-x} + \frac{yz}{1-xyz} \\ &= \frac{yz-1}{(1-x)(1-xyz)}.\end{aligned} \tag{16}$$

Given $0 < x,y,z < 1$, both terms $1-x$ and $1-xyz$ are positive. Since $yz < 1$ (as both y and z are less than 1), the numerator $yz-1$ is negative. Therefore, the entire expression for $\frac{\partial f(x,y,z)}{\partial x}$ is less than zero:

$$\frac{\partial f(x,y,z)}{\partial x} < 0. \tag{17}$$

This implies that $f(x,y,z)$ is monotonically decreasing with respect to x in the given domain. By the symmetry of f, the same holds for y and z, completing the proof. □

We compare the effectiveness of the conventional SSIM formulation and our modified SSIM approach in the *Patio-High* scene as shown in Table F. Our SSIM formulation can successfully remove distractors while conventional SSIM fails to do so.

Parameter-tuning Free. Here we show our method's superiority against RobustNeRF [39] that no explicit outlier ratio assignment is required for training on scene *P*atio-High. As shown in Fig. D, multiple experiments with different ratios need to be run for RobustNeRF [39] to gain its best performance. However, our method does not need any hyperparameter tuning and still archives much better results than RobustNeRF [39].

Fast Convergence. In Fig. E, we show the convergence curve comparison between RobustNeRF [39] and our method under different occlusion conditions(*T*ree and *P*atio-High), using SSIM metrics as the basis for comparison. Our method demonstrates significantly faster convergence — nearly one magnitude faster — and exhibits markedly better performance after reaching convergence.

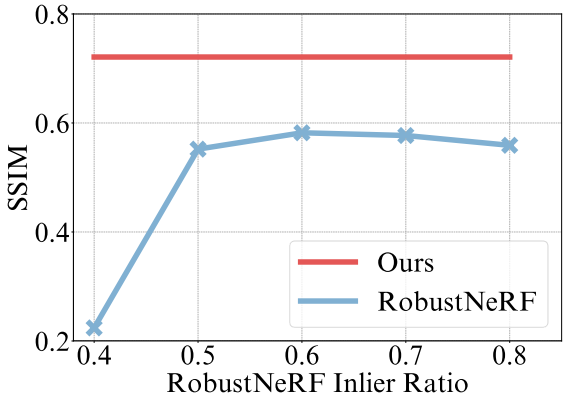

Figure D. **The Performance of RobustNeRF [39] under Different Inlier Ratios Compared to Our Method.**

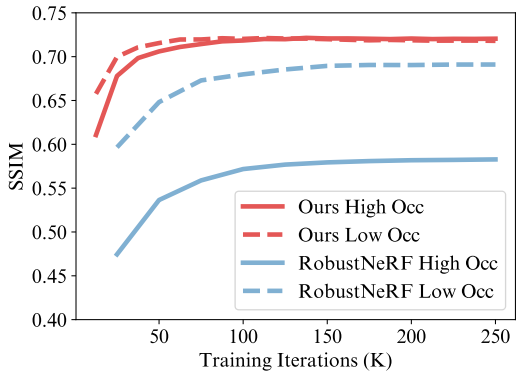

Figure E. **SSIM Evaluation Metrics across Training Iterations under Different Occlusion Conditions.**

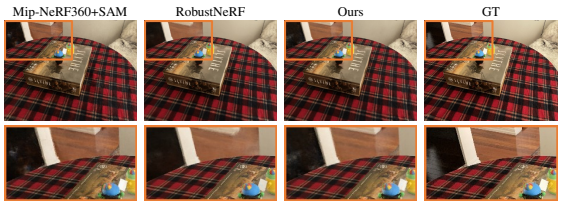

Figure F. **Failure cases.**

Failure Case. Similar to baseline methods, we also struggle in regions with strong view-dependent effects, see Fig. F. Moreover, inherited from the limitation of our base model Mip-NeRF360, we also require sufficient training views. Our performance will degrade significantly when the training views become sparse.