AccidentalGS: 3D Gaussian Splatting from Accidental Camera Motion -Supplementary Material-

Mao Mao, Xujie Shen, Guyuan Chen, Boming Zhao, Jiarui Hu, Hujun Bao, Zhaopeng Cui*
State Key Lab of CAD&CG, Zhejiang University

In this supplementary material, we first provide the derivation of the rotation matrix approximation and the experiment of comparison with the w/o simplified rotation matrix in Section A. Next, the generation of datasets is described in Section B. In Section C, Section D, and Section E, we respectively demonstrate the effectiveness of our method in filtering abnormal depth, its performance on telescope-like images, and the optimization results of camera intrinsic parameters on the Replica-Accidental dataset. Additionally, in Section F, we offer further details on implementation including experimental settings, test-time optimization, and the evaluation of camera trajectories, etc. Lastly, in Section G, we provide extra qualitative results, including more camera trajectory and novel view synthesis results on the Replica-Accidental dataset, as well as more novel view synthesis results on the real-world dataset DfUSMC and Orbbec-Accidental. Also, we provide our comparison results with COGS [4] and SPARF [10].

A. Rotation Matrix Approximation for Accidental Motion

For a rotation vector $\boldsymbol{\mu} = [\mu_x, \mu_y, \mu_z]^{\mathrm{T}}$, the corresponding rotation matrix $\mathbf{R}$ can be expressed as:

$$\mathbf{R} = \exp([\boldsymbol{\mu}]_\times), \tag{1}$$

where $[\boldsymbol{\mu}]_\times$ is the skew-symmetric matrix of the rotation vector $\boldsymbol{\mu}$, which can be calculated as:

$$[\boldsymbol{\mu}]_\times = \begin{bmatrix} 0 & -\mu_z & \mu_y \\ \mu_z & 0 & -\mu_x \\ -\mu_y & \mu_x & 0 \end{bmatrix}. \tag{2}$$

Assuming the rotation vector $\boldsymbol{\mu}$ has a small magnitude, we can approximate the exponential of the skew-symmetric matrix using the first-order Taylor expansion as:

$$\exp([\boldsymbol{\mu}]_\times) \approx \mathbf{I} + [\boldsymbol{\mu}]_\times. \tag{3}$$

Here, $\mathbf{I}$ is the 3×3 identity matrix. Thus, the rotation matrix $\mathbf{R}$ for a small rotation can be approximated as:

$$\mathbf{R} \approx \mathbf{I} + [\boldsymbol{\mu}]_\times \tag{4}$$

$$= \begin{bmatrix} 1 & -\mu_z & \mu_y \\ \mu_z & 1 & -\mu_x \\ -\mu_y & \mu_x & 1 \end{bmatrix}, \tag{5}$$

where the approximation holds when the $\boldsymbol{\mu}$ is small, meaning the rotation angles around each axis are small [7]. **Validation on advantages of simplified camera model and BA.** We compared ours with the non-simplified version. The results in the table 1 show that the simplified camera model and BA provide more accurate camera focal length and pose (ATE: 2.08 $\rightarrow$ 0.28). And ours optimizes faster (BA time on *Rm-0*: 85 $\rightarrow$ 30 min). These results show that the non-simplified camera model and BA struggle to handle accidental motions due to the high complexity of the optimization function, frequently resulting in poor local optima, and our simplifications enhance both accuracy and efficiency.

		Rm-0	Rm-1	Rm-2	Of-0	Of-1	Of-2	Of-3	Of-4	**Avg.**
Ours w/o Simplified.	**f** (GT:600)	1141	1180	1101	1107	1058	1134	1114	1102	1117
	ATE ↓	1.79	1.89	1.72	2.56	1.79	1.94	1.78	3.19	2.08
Ours	**f** (GT:600)	**586**	**630**	**625**	**610**	**578**	**617**	**589**	**714**	**623**
	ATE ↓	**0.10**	**0.16**	**0.13**	**0.34**	**0.39**	**0.17**	**0.10**	**0.82**	**0.28**

Table 1. **Relevant Experiments of Simplified Camera Model and BA**. focal length (**f**) is initialized as 1200 (pixels).

B. Dataset Details

B.1. Synthetic Dataset

We have generated a synthetic dataset named **Replica-Accidental** following the description of accidental camera motion in this study [13]. We utilize the realistic Replica-Dataset [9] renderer SDK to generate images across 8 indoor scenes (Room-0 to Room-2 and Office-0 to Office-4). The following are the license terms for the Replica-Dataset and its SDK:

*Corresponding author.

- **Provider**: Facebook Technologies, LLC
- **License and Terms of Use**: Non-commercial, research, or educational purposes as per the agreement. The dataset is provided "as is" without any warranties, and disputes are governed by California law.

Details about them including viewport variation can be found in Tab. 2. The dataset includes two components: ground truth camera poses and corresponding RGB images. For each scene, the data is divided into three parts: training part (48 frames), extrapolation part (30 frames), and test part (6 frames). The test part is further divided into an easy part (3 frames) and a hard part (3 frames). The input to the Replica renderer includes the camera poses and the material properties of the scene. The output is the corresponding rendered RGB images. The material properties of the scene are already provided; therefore, we need to obtain the camera poses for rendering.

B.1.1. Camera Pose Perturbation

Let $\delta_t = \begin{bmatrix}\delta_{t_x} & \delta_{t_y} & \delta_{t_z}\end{bmatrix}^\top$ represent the small perturbation in translation. Each component of δ_t is sampled from a uniform distribution within the range $[-\Delta T_{\text{range}}, \Delta T_{\text{range}}]$.

Let $\delta_{\text{angle}} = \begin{bmatrix}\delta_{\theta_x} & \delta_{\theta_y} & \delta_{\theta_z}\end{bmatrix}^\top$ represent the small perturbation in rotation, which is a three-dimensional vector. Each component of δ_{angle} is sampled from a uniform distribution within the range $[-\Delta\theta_{\text{range}}, \Delta\theta_{\text{range}}]$. When generating the training set, ΔT_{range} is set to 0.003 meters and $\Delta\theta_{\text{range}}$ is set to 0.03030303 degrees. For the *easy* part of the test set, ΔT_{range} is set to 0.01 meters. For the *hard* part of the test set, ΔT_{range} is set to 0.1 meters. For both parts of the test set, $\Delta\theta_{\text{range}}$ is set to 0.2 degrees. Then the rotation matrices for small perturbations around the x-, y-, and z-axes are given by:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\delta_{\theta_x}) & -\sin(\delta_{\theta_x}) \\ 0 & \sin(\delta_{\theta_x}) & \cos(\delta_{\theta_x}) \end{bmatrix}, \tag{6}$$

$$R_y = \begin{bmatrix} \cos(\delta_{\theta_y}) & 0 & \sin(\delta_{\theta_y}) \\ 0 & 1 & 0 \\ -\sin(\delta_{\theta_y}) & 0 & \cos(\delta_{\theta_y}) \end{bmatrix}, \tag{7}$$

$$R_z = \begin{bmatrix} \cos(\delta_{\theta_z}) & -\sin(\delta_{\theta_z}) & 0 \\ \sin(\delta_{\theta_z}) & \cos(\delta_{\theta_z}) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{8}$$

$$R_{\delta_{\text{angle}}} = R_z \cdot R_y \cdot R_x, \tag{9}$$

where the 4x4 matrix $T_\delta \in SE(3)$ represents the camera pose perturbation, which combines the translation perturbation vector δ_t and the rotation perturbation matrix $R_{\delta_{\text{angle}}}$, is given by:

$$T_\delta = \begin{bmatrix} R_{\delta_{\text{angle}}} & \delta_t \\ \mathbf{0} & 1 \end{bmatrix}, \tag{10}$$

Training Part. Assume we need to obtain the camera poses for M training images, denoted as $\{T_i^{\text{train}}\}$. Given the pose of the first frame T_1^{train}, the pose of the $(i+1)$-th frame can be computed as:

$$T_{i+1}^{\text{train}} = T_\delta^{i+1} \cdot T_i^{\text{train}}, \quad \text{for } i = 1 \text{ to } M-1, \tag{11}$$

where T_δ^{i+1} represents the relative transformation from the $(i+1)$-th frame to the i-th frame.

Extrapolated Part Based on the Training Part. To demonstrate the effect of Joint Optimization in the main paper, we generate extrapolated views based on the training part of the Replica-Accidental dataset. Each scene is extrapolated 30 frames. Based on the training set, we linearly extrapolate the poses. For each scene, we linearly interpolate the camera poses using the starting and ending poses of the training set. The new pose is given by:

$$\text{pose}_{\text{new}} = (1-\alpha) \cdot \text{pose}_{\text{start}} + \alpha \cdot \text{pose}_{\text{end}}, \tag{12}$$

$$\alpha \in \{1.5, 2.0, 2.5, \ldots, 8.0\} \cup \{-0.5, -1.0, -1.5, \ldots, -8.0\}. \tag{13}$$

Firstly, we convert the rotation part of the camera poses to Euler angles and extract translation vectors. We perform linear extrapolation on these Euler angles and translation vectors correspondingly using the formula above. Finally, we convert the interpolated Euler angles back to rotation matrices to obtain the new poses.

Testing Part. The camera poses for N testing images are denoted as $\{T_i^{\text{test}}\}$. Given the pose of the first frame T_1^{test}, the pose of the $(i+1)$-th frame can be computed as:

$$T_{i+1}^{\text{test}} = T_\delta^{i+1} \cdot T_1^{\text{test}}, \quad \text{for } i = 1 \text{ to } N-1. \tag{14}$$

B.2. Real-World Dataset

To more comprehensively evaluate the method's effectiveness, We propose the Orbbec-Accidental dataset. We captured video clips of seven real-world scenes using the Orbbec Femto Bolt camera. The acquisition process adhered to the definition of accidental motion. The Orbbec Femto Bolt camera does not perform zooming during capture and provides ground-truth camera focal lengths. The dataset consists of four indoor scenes (*Toy, Office, Machine, Kitchen*) and three outdoor scenes (*Terrace, Sculpture, Bike*). For each scene, 48 frames were used as input, and 6 frames were reserved for novel view synthesis testing.

DfUSMC [3] follows accidental motion [13], and provides video clips with small motion, but lacks accurate camera poses and focal lengths. There are 10 scenes: *Bikes, Dinos, Flowers, Frogs, Louvre, Notre-Dame, Plants, Stairs, Stones, Trees*. Each scene has a 30-frame video. We selected nine videos (excluding *Trees* because it is a highly

Scenes	Translation			Rotation		
	train	test(easy)	test(hard)	train	test(easy)	test(hard)
Rm-0	0.019	0.014	0.253	0.296	0.254	0.197
Rm-1	0.023	0.012	0.107	0.237	0.296	0.259
Rm-2	0.022	0.015	0.116	0.273	0.255	0.248
Off-0	0.041	0.013	0.112	0.258	0.270	0.230
Off-1	0.023	0.013	0.100	0.254	0.215	0.225
Off-2	0.021	0.010	0.108	0.316	0.226	0.213
Off-3	0.020	0.013	0.095	0.259	0.231	0.245
Off-4	0.029	0.013	0.126	0.209	0.245	0.267
Mean	0.025	0.012	0.126	0.263	0.249	0.235

Table 2. **Explanation of Camera Poses in Self-generated Synthetic Dataset Replica-Accidental.** We have generated a synthetic dataset named **Replica-Accidental** following the description of accidental camera motion in this study[13]. Translation and Rotation in *train* part denote the largest relative translation and rotation across all frames. Translation and Rotation in *test* part denote the largest relative translation and rotation to the reference view. Translation and rotation are measured in **meters** and **degrees**. For each scene, the training part comprises 48 images, while the testing part comprises 6 images. The number of testing part for the easy part and the hard part are both half of the total. The easy part has small viewport variation related to the reference view, while the hard part has large.

dynamic scene). For each video, the first 20 frames were used as the training set. 3 frames were selected as the test set (frames 24, 27, and 30).

B.3. Data in the Wild

The lunar dataset was obtained from the NASA website. We collected thirty consecutive images over time, with each image taken one hour apart. The mountain dataset was collected from YouTube, with each scene containing 60 images from consecutive video frames.

The following are the license terms for the moon and mountain images:

The Moon Images:

- **License and Terms of Use**: Used under NASA's non-commercial use policy. NASA content is generally not subject to copyright in the United States and can be used for educational or informational purposes.

The Mountains Images:

- **Source**: From YouTube.
- **License and Terms of Use**: Fair use on YouTube.

C. Robustness to Abnormal Depth.

Specifically, depth values exceeding $Q_3 + 1.5\text{IQR}$ or falling below $Q_1 - 1.5\text{IQR}$ are removed as outliers. Please see the supplementary material for more details.

$$Q_3 = P_{75}(\{D_k\}), Q_1 = P_{25}(\{D_k\}), \text{IQR} = Q_3 - Q_1. \tag{15}$$

As the Sculpture scene of the Orbbec-Accidental Dataset illustrated in Fig. 1, By eliminating anomalous depth values, the densification of Gaussian points can be optimized, thereby facilitating the generation of more photorealistic rendering images.

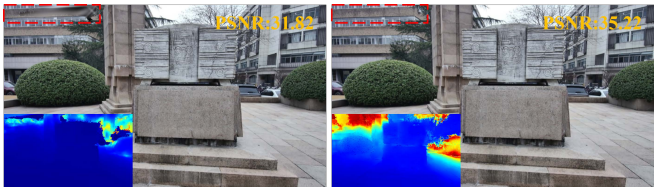

Figure 1. Novel view rendering for the Sculpture scene of the Orbbec-Accidental. Left: without depth outlier filtering; Right: with depth outlier filtering.

D. Evaluation with Telescope-like Images.

Telescope-like images are typically captured for distant objects (e.g., the Moon) located at extreme distances. When either the camera itself moves or remains stationary while the distant object is in motion, the resulting images appear as though they were captured under accidental camera movement. We also evaluate our AccidentalGS in such challenging scenarios by collecting lunar images from the NASA website*.Taking 30 lunar images as input, we apply AccidentalGS to estimate the camera parameters and construct a 3D Gaussian Splatting model. As shown in Fig. 2, we can see that AccidentalGS can still achieve promising results for both the reconstruction and novel view synthesis.

E. Estimation of Camera Intrinsics

Tab. 3 presents the optimized camera focal length and pose results of AccidentalGS without given GT intrinsics and extrinsics. The initial focal length is set to 1200 pixels (GT is

*https://svs.gsfc.nasa.gov/gallery/moonphase

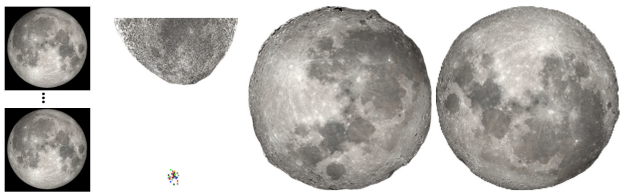

Figure 2. **Evaluation with Telescope-like Images.** The input comprises 30 lunar images captured hourly.

	Rm-0	Rm-1	Rm-2	Off-0	Off-1	Off-2	Off-3	Off-4	Avg.
f	585.66	630.37	625.35	610.02	578.18	617.03	588.57	713.82	623.20
RPE_t↓	0.089	0.116	0.104	0.338	0.331	0.144	0.057	0.442	0.203
RPE_r↓	0.255	0.265	0.62	0.781	1.025	0.339	0.265	0.74	0.536
ATE↓	0.097	0.163	0.13	0.344	0.388	0.17	0.096	0.819	0.276

Table 3. **AccidentalGS's Results of Camera Focal Length and Extrinsics Estimation on the Replica-Accidental without Providing GT Focal Length.** The GT Focal Length is 600 (pixels).

600 pixels). Both estimated errors are minimal, showing the accuracy of Ours.

F. Implementation Details

F.1. Computing Resources

The experiments were conducted using the following computational resources:

- **GPU:** Single NVIDIA GeForce RTX 3090, 24576 MiB. The actual GPU memory usage is from 8000 MiB to 24500 MiB, depending on the specific scene.
- **CPU:** Intel(R) Xeon(R) Gold 6139M CPU @ 2.30GHz, x86_64.
- **CUDA:** Version 11.6.
- **Python:** Version 3.9.

The approximate execution times for different optimization stages are as follows:

- **Initialization stage**
  - Synthetic datasets: about 20 minutes per scene.
  - Real-world datasets: about 40 minutes per scene.
- **Joint Optimization Stage**
  - Synthetic datasets: about 30 minutes per scene.
  - Real-world datasets: about 50 minutes per scene.

F.2. Experimental Setting

For the synthetic dataset, all camera poses are initialized to the identity matrix, and the inverse depth values are uniformly sampled within the range [0.01, 1]. Ground truth focal length is used for synthetic images. In the initialization stage, for the real-world dataset DfUSMC, all camera poses are initialized to the identity matrix, and the inverse depth values in all scenes except *Dinos* are uniformly sampled within the range [1, 5]. The inverse depth values are all initialized to 0.1 of *Dinos* Scene.

We utilize the Adam optimizer. In the initialization stage, initial learning rates for poses, inverse depth, and focal length are 1e-3, 5e-4, and 1e-3, halved every 2000 iterations out of a total of 8000. During the joint optimization process, the learning rate for the poses and inverse depth are set to 1e-5 and 5e-5.

In joint optimization, $\mathcal{L}_{\text{re}}$ and $\mathcal{L}_{\text{d}}$ are set to 0.05 and 1. The training iteration is 14 K. For the synthetic, $\mathcal{L}_{\text{re}}$ and $\mathcal{L}_{\text{d}}$ are set to 0.05 and 0.0005. The training iteration is 8 K. We use GMFlow [12] as an off-the-shelf optical flow estimation method. For both synthetic and real-world datasets, we use its *GMFlow-scale2-regrefine6-sintelft* pretrained model.

During the initialization phase on real-world datasets, the total number of iterations is set to 10K. The initial learning rates for all variables are the same as those used in the synthetic dataset settings. For most scenes, the learning rate is halved every 4K iterations, while for the Bikes scene, it is halved every 9K iterations.

F.3. COLMAP

The default parameters cause COLMAP [8] to fail in reconstructing scenes on Replica-Small-Motion and DfUSMC. A significantly lower minimum triangulation angle than the default value in COLMAP is required to achieve any reconstruction results; however, most of the 3D reconstructions contain substantial noise.

F.4. Strategies of 3D Gaussian

In the joint optimization process, whether for synthetic or real-world scenes, the densification and pruning of the 3D Gaussian points start at the 500th iteration and are performed every 100 iterations. The densification gradient threshold is set to 0.0002. Points with viewspace gradients greater than the densification gradient threshold undergo densification.

For synthetic scenes, the total number of iterations is 8K. The densification and pruning of the point cloud stop at the 5Kth iteration. The camera extent parameter is set to 0.01. Points with viewspace gradients greater than this value are split, while those with smaller gradients are cloned. The threshold for point cloud pruning is set with $\text{opacity}_{\min} = 0.001$. Points with opacity below this threshold are removed.

For real-world scenes, the total number of iterations is 14K. The densification and pruning of the point cloud stop at the 7Kth iteration. The camera extent parameter is set to 0.004, except for the dinos scene where it is set to 1. Points with viewspace gradients greater than this value are split, while those with smaller gradients are cloned. The threshold for point cloud pruning is set with $\text{opacity}_{\min} = 0.005$. Points with opacity below this threshold are removed.

		Rm-0	Rm-1	Rm-2	Of-0	Of-1	Of-2	Of-3	Of-4	Avg.	Avg.(Ours)
SPARF [10] Truong et al.	**PSNR** ↑	20.64	21.71	22.79	25.02	24.72	20.56	26.60	26.80	23.60	**36.38**
	SSIM↑	0.72	0.65	0.84	0.81	0.85	0.75	0.83	0.84	0.79	**0.96**
	RPE$_r$↓	5.61	7.95	6.43	5.45	8.08	5.43	5.36	6.88	6.40	**0.49**
	ATE ↓	0.40	0.38	0.42	0.59	0.66	1.19	0.27	0.43	0.54	**0.18**
COGS [4] Jiang et al.	**PSNR** ↑	25.39	23.62	23.42	28.92	27.70	25.09	22.98	28.53	25.71	
	SSIM↑	0.76	0.69	0.86	0.86	0.90	0.80	0.80	0.86	0.82	
	RPE$_r$↓	3.84	49.47	67.07	3.79	57.07	53.49	48.30	50.30	41.67	
	ATE ↓	1.18	1.65	1.29	1.98	3.80	3.07	1.26	4.29	2.31	

		Toy	Office	Terr.	Sculp.	Mach.	Bike	Kit.	Avg.	Avg.(Ours)
SPARF [10] Truong et al.	**PSNR** ↑	19.59	17.54	19.66	17.21	16.92	16.86	21.37	18.45	**32.65**
	SSIM↑	0.73	0.67	0.61	0.44	0.63	0.48	0.80	0.62	**0.95**
COGS [4] Jiang et al.	**PSNR** ↑	24.97	24.38	27.01	24.73	23.31	18.66	29.47	24.65	
	SSIM↑	0.85	0.82	0.83	0.67	0.80	0.56	0.91	0.78	

Table 4. **Replica-Accidental and Orbbec-Accidental Dataset.**

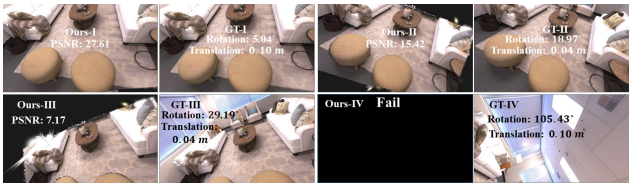

Figure 3. **Evaluation of Ours on Rm-0-S of Replica-Small-Motion for Novel View Synthesis under Significant Viewpoint Changes.** The camera rotations for images I, II, III, and IV are 5.04°, 18.97°, 29.19°and 105.43°, respectively. AccidentalGS performs well in the I and II but underperforms in the latter two.

F.5. Large Camera Views Trial

Our proposed SmallGS focuses on scenarios with small or even accidental camera motion [13], particularly when standard methods don't work. To better evaluate the capability of our method, we have generated more challenging data and conducted further testing on the Rm-0-S of the Replica-Small-Motion dataset. In the learning stage, we find that our method still performs well (with an average of PSNR 31.45, SSIM 0.91, and LPIPS 0.11, which slightly decreases) when the input images have the camera translation and camera rotation up to 0.03 m and 8.53° respectively compared to the reference/central view. This rotation is relatively large for small motion scenarios. When the max translation and rotation further increase (e.g., 0.05 m and 16.23°), the performance of our method fails. The failure stems from two factors: first, the decreased accuracy of optical flow estimation, and second, exceeding the range within which the camera rotation matrix can be simplified. However, in this case, the standard pose estimation can naturally be utilized. Regarding the novel view synthesis stage, as shown in Fig. 3, our method can still provide high-quality novel view rendering with a camera rotation of 19°. Of course, when the novel view has large differences compared to the input, ours will decrease (e.g., the camera rotation reaches 30°).

F.6. Camera Focal Length

For all comparative experiments on the synthetic dataset Replica-Small-Motion, including our method and all baselines mentioned in the main text, we use the ground truth values for the camera's focal length. Real-world dataset (Sec. B.2) lacks corresponding accurate camera focal lengths. Our method and NeRFmm estimate the focal length independently. Other baselines lack the capability to estimate the camera's focal length. To ensure a comprehensive comparison, we use the image width (in pixels) as the focal length for these baselines.

F.7. Test-time Optimization

After the Initialization and Joint Optimization, our method fixes the 3D Gaussian Field and inverse depth. Using the images from the test set as input, we optimize the camera poses using only the Reprojection and Photometric loss. Ours and all the baselines methods in the main paper initialize the camera poses for the test set with the identity matrix. Additionally, we retain the best result for all baselines.

F.8. Evaluation with Extrapolation

To demonstrate the effect of Joint Optimization in the main paper, we first estimate the camera poses on the training set. Similar to Section B, these estimated poses are then extrapolated to obtain new camera poses. Utilizing these new camera poses, we perform novel view synthesis on the trained 3D Gaussian Field. Finally, we compare the rendered images corresponding to those GT images with the ground truth camera poses for evaluation.

F.9. Trajectory Evaluation

Since the estimated camera trajectories and the GT camera trajectories are not in the same coordinate system, we first align the estimated camera trajectories to the GT camera trajectories. Following BARF [6] and NoPe-NeRF [1], we align the two sets of pose trajectories (optimized and GT)

globally with a Sim(3) transformation using the Umeyama algorithm [11] in the *evo* [2] toolbox. We then use *evo* (the version is 1.25.2) to calculate the rotational and translational Relative Pose Errors (RPE) and the Absolute Trajectory Error (ATE).

F.10. CUDA Differentiable Rasterization Code

3D Gaussian Splatting [5] provides their differentiable CUDA rasterization engine. The following are the license terms for the 3D Gaussian Splatting rasterization code: **License**: Gaussian-Splatting License. **Providers**: Inria and the Max Planck Institute for Informatik (MPII). **Terms**: The software can be used for research purposes by both academic and industrial users, free of charge. It cannot be used for commercial purposes without prior consent. The software is provided "as is" without any warranties. Based on this code, we incorporated gradients with respect to camera poses and α-blending depth map rendering.

G. More Experiment Results

G.1. Comparison with COGS and SPARF

COGS [4] and SPARF [10] are designed for extremely sparse views with wide baselines, unlike our focus on tiny-baseline sequences. In COGS [4], the mono-depth for unprojection is supervised by the gradient-detached rendered depth for multi-view consistency, while ours incorporates a simplified camera model and **online adaptive depth consistency** loss (the pseudo-GT depth is also optimized throughout our joint optimization), better suited for accidental motion scenarios. SPARF [10] uses the rendered depth from training views to supervise unseen viewpoints. Our method (see Tab.1, 5 in the main paper) achieves better performance. VRAM on RTX 3090 prevented SPARF testing, but we test it on A100. As shown in Tab. 4, on our datasets, COGS [4] yields inferior rendering quality due to its noisy depth and pose estimates. SPARF [10] struggles with small baseline inputs due to pose estimation difficulties and overfitting risks.

G.2. Camera Trajectory

Fig. 4 presents additional camera trajectory comparison results on the Replica-Accidental dataset. It can be observed that our camera trajectories exhibit a closer match to the ground truth.

G.3. Novel Views on the Replica-Accidental Dataset

Figs. 5 to 7 show additional novel view rendering results on the Replica-Accidental dataset. Our approach successfully recovers fine details in both color and geometry.

G.4. Novel Views on the Real-World Dataset

Figs. 8 to 11 display additional novel view rendering results on the real-world dataset. Our technique accurately retrieves intricate details in both RGB and depth maps. Figs. 12 to 16 display additional novel view rendering results on the real-world dataset Orbbec-Accidental. Our technique can get sharper renderings.

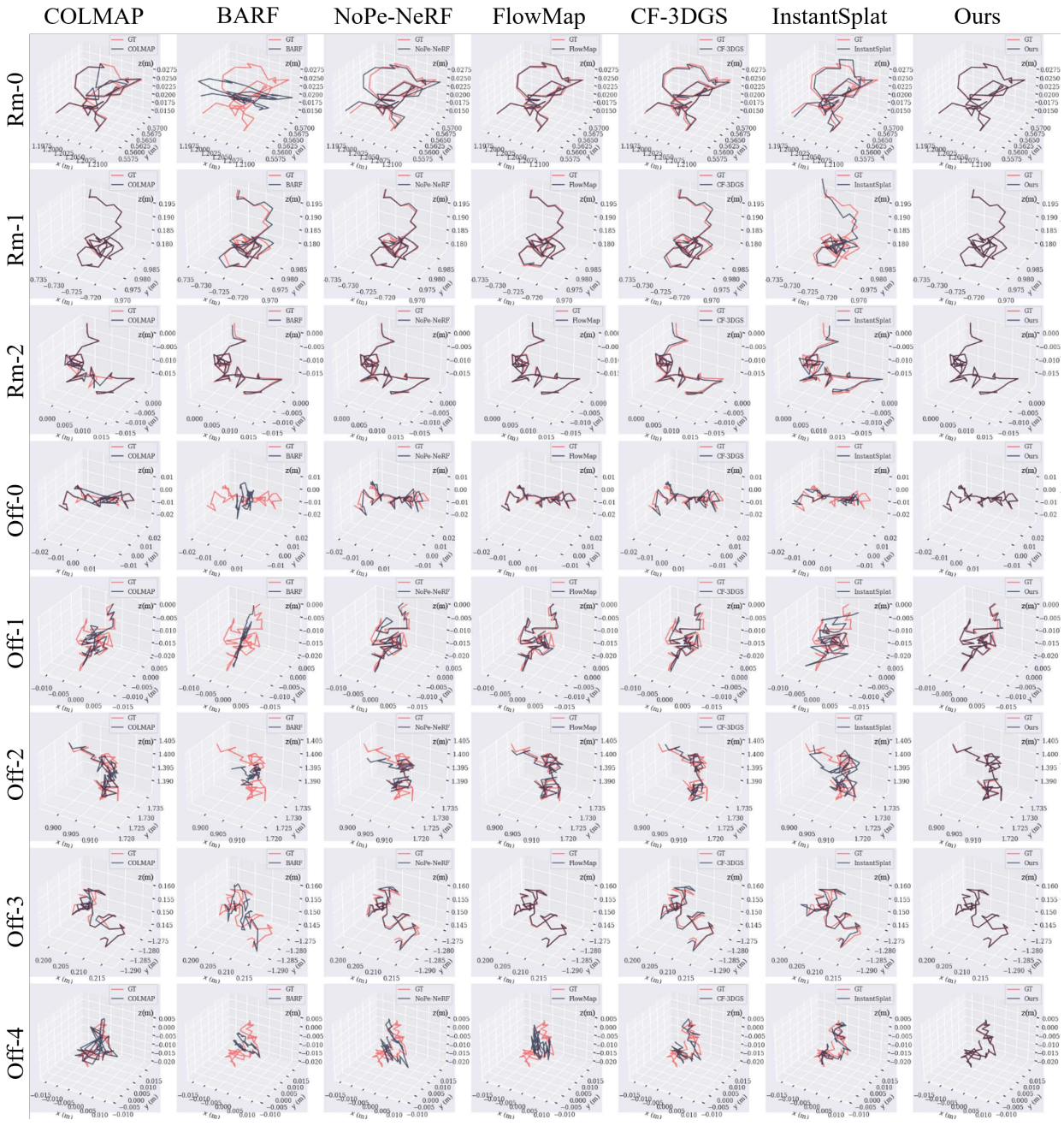

Figure 4. **Camera Trajectories on the Replica-Accidental Dataset.** Our camera trajectories exhibit a closer match to the ground truth. All methods are conducted with the GT camera intrinsic.

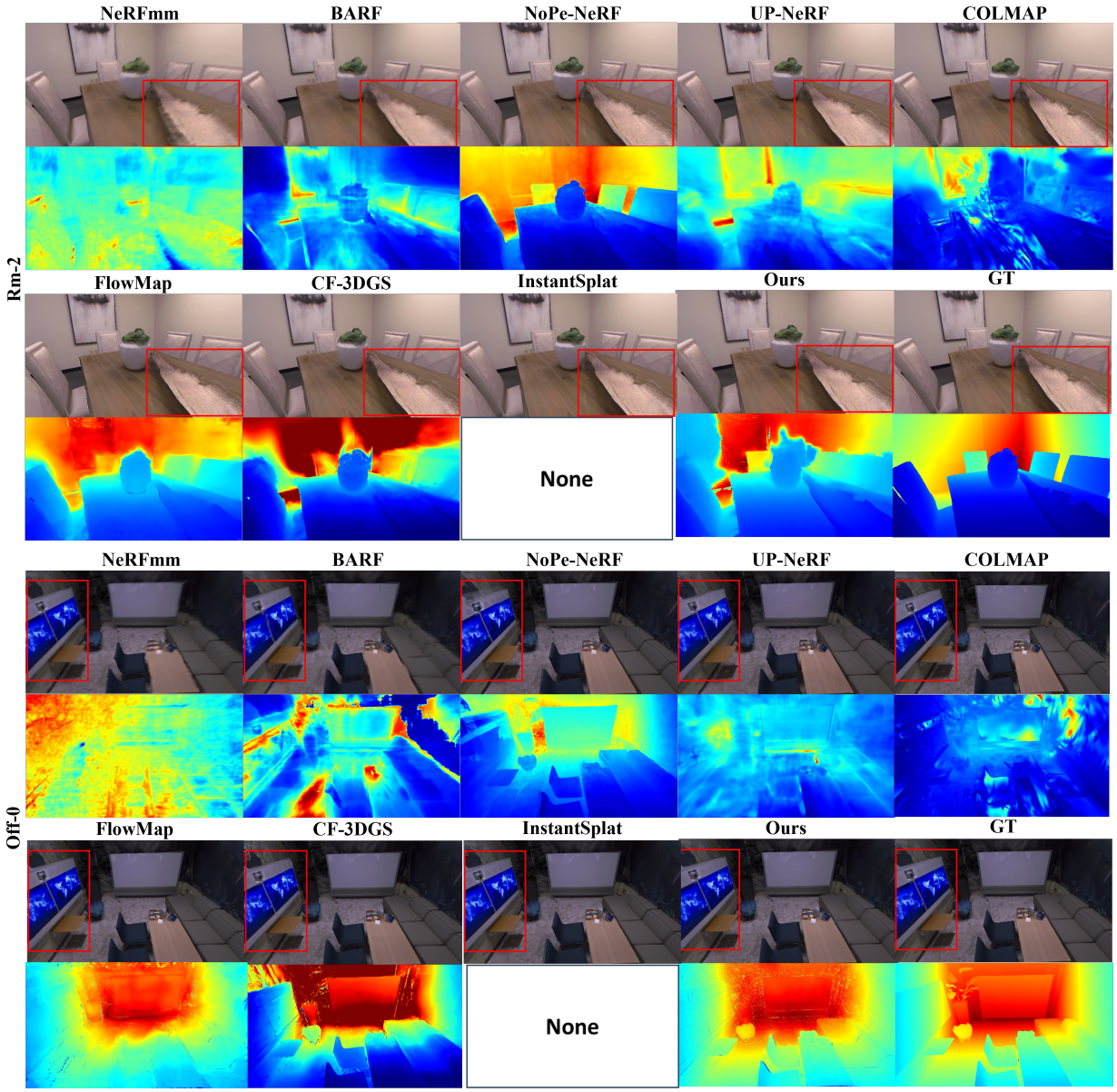

Figure 5. **Qualitative results of novel view synthesis and depth prediction on Room-2 and Office-0.** For each scene, we show the color (first row) and depth (second row) rendered image.

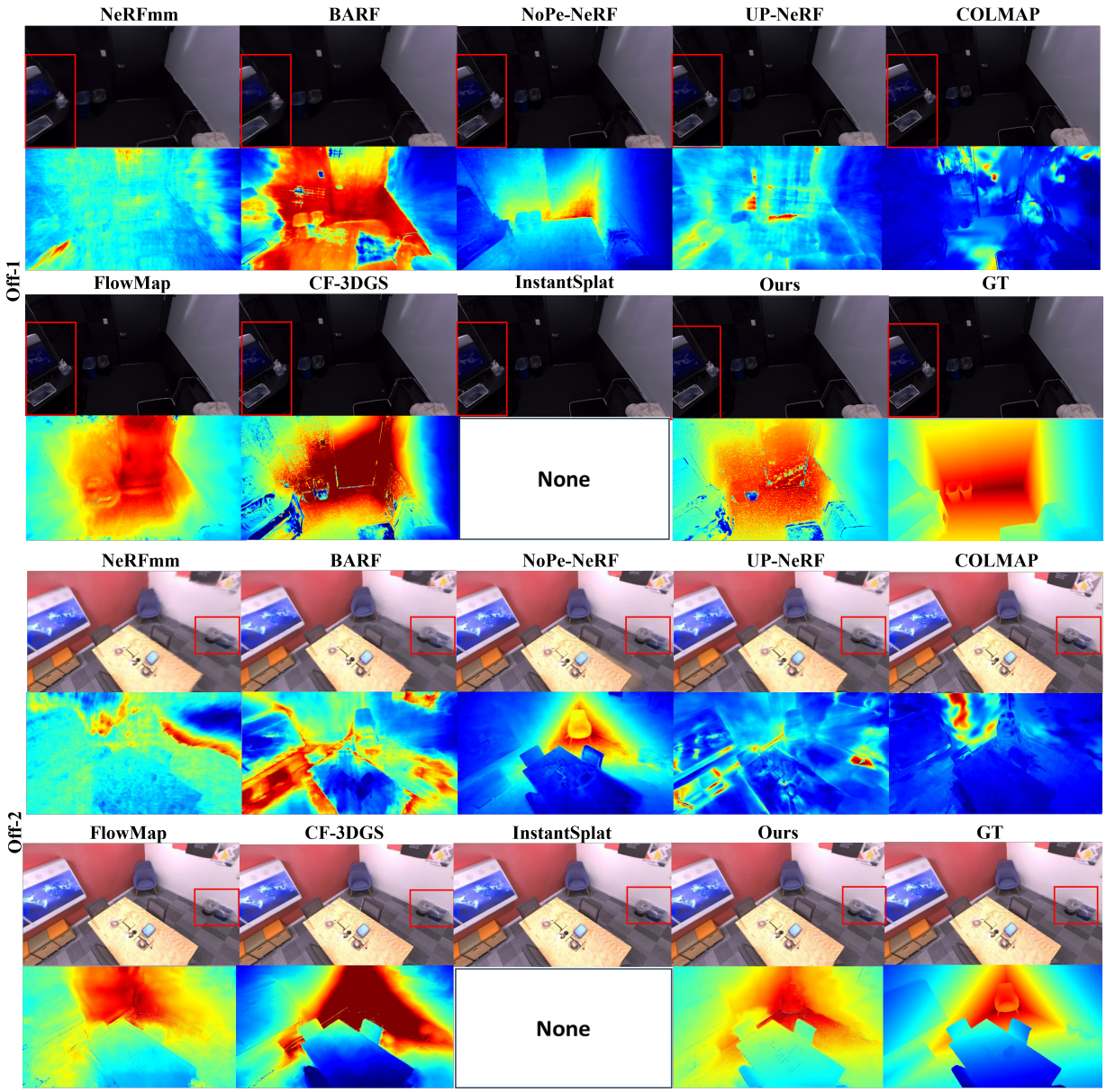

Figure 6. **Qualitative results of novel view synthesis and depth prediction on Office-1 and Office-2.** For each scene, we show the color (first row) and depth (second row) rendered image.

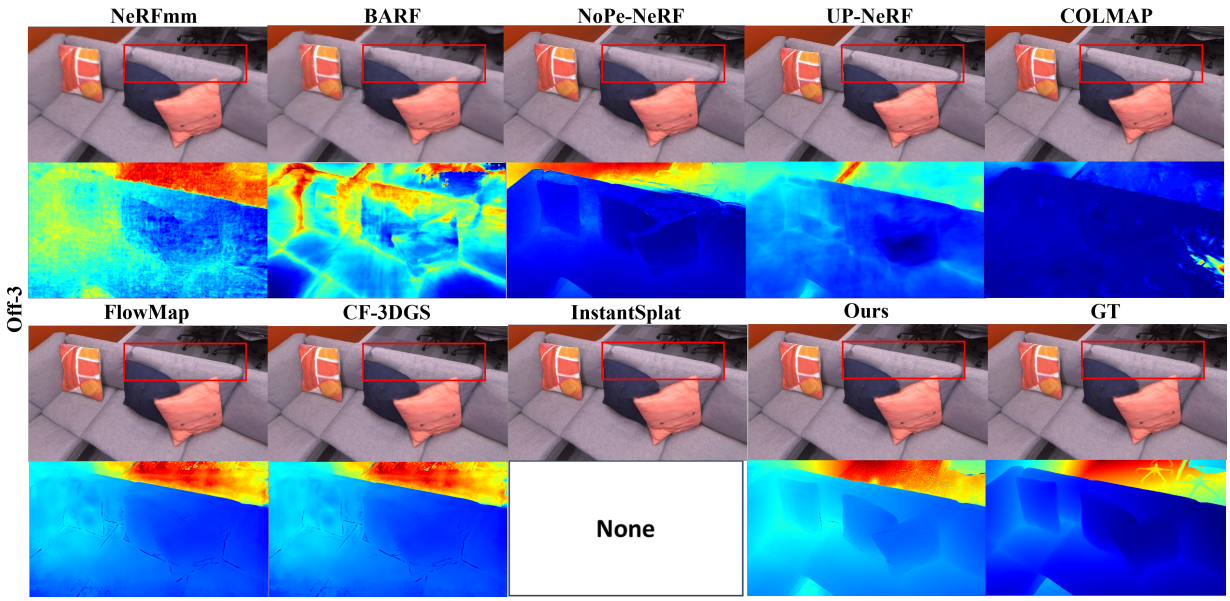

Figure 7. **Qualitative results of novel view synthesis and depth prediction on Office-3 and Office-4.** For each scene, we show the color (first row) and depth (second row) rendered image.

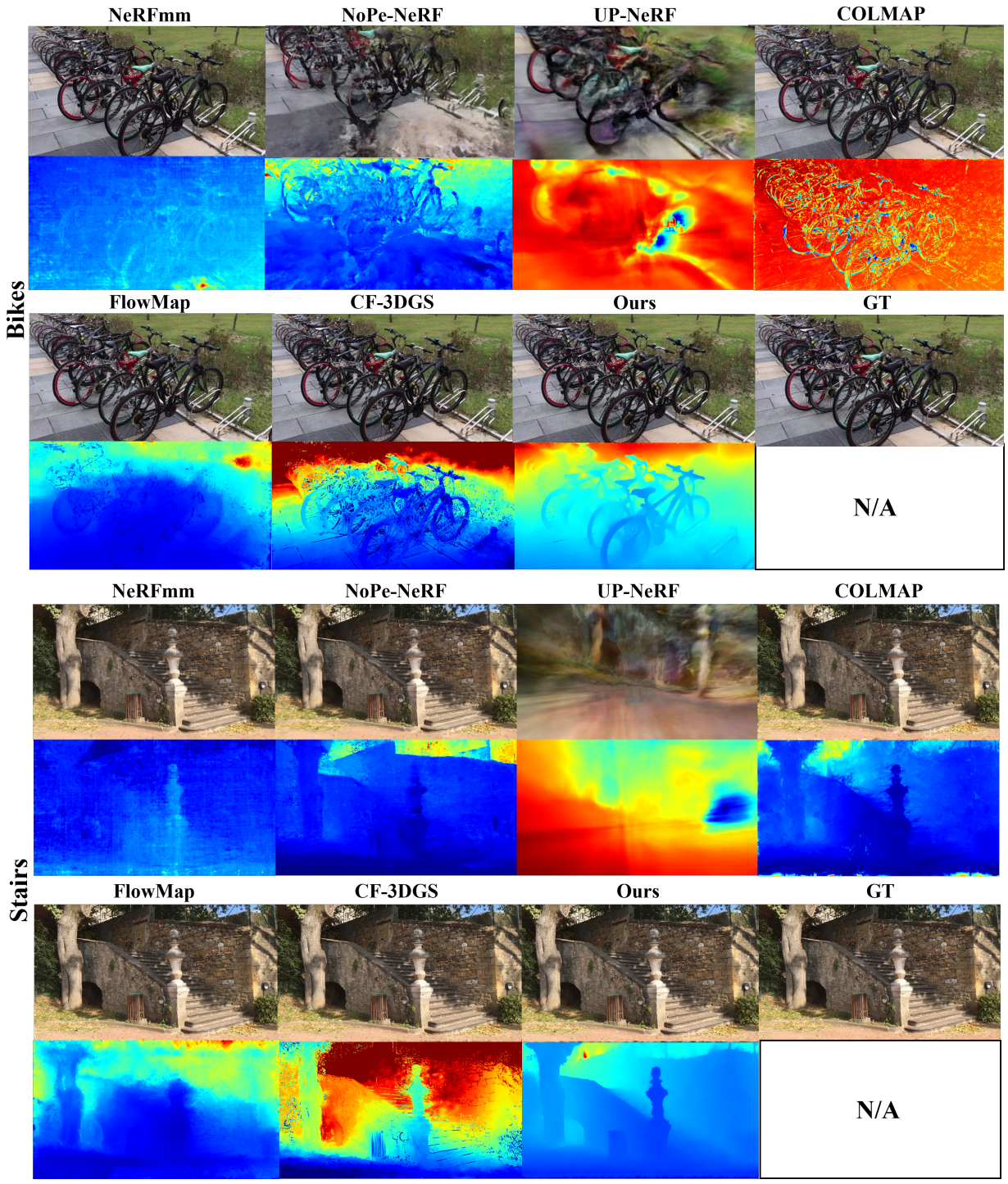

Figure 8. **Qualitative results of novel view synthesis and depth prediction on Bikes and Stairs. For each scene, we show the color (first row) and depth (second row) rendered image.** Our technique accurately retrieves intricate details in both color and geometry.

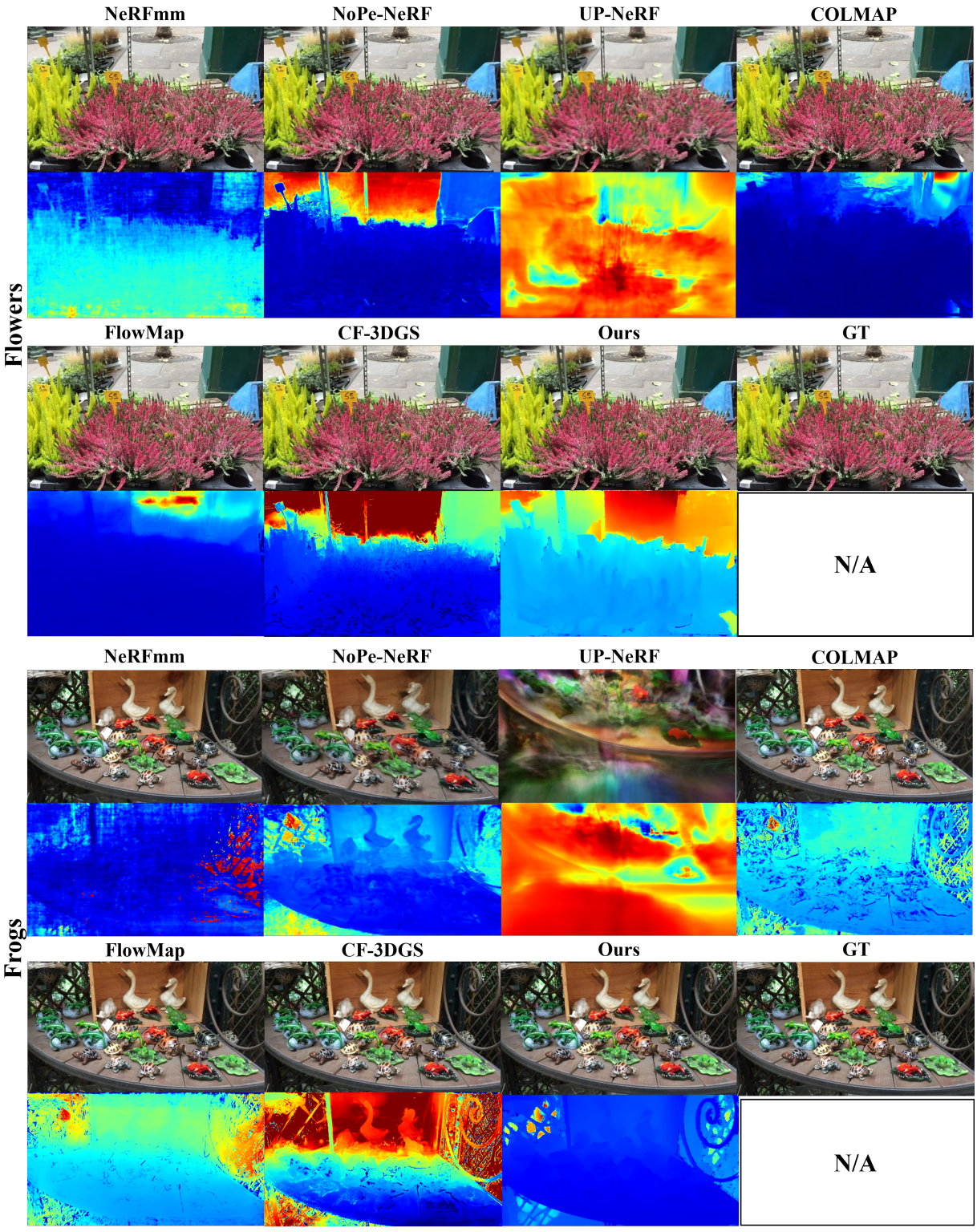

Figure 9. **Qualitative results of novel view synthesis and depth prediction on Flowers and Frogs.** For each scene, we show the color (first row) and depth (second row) rendered image.

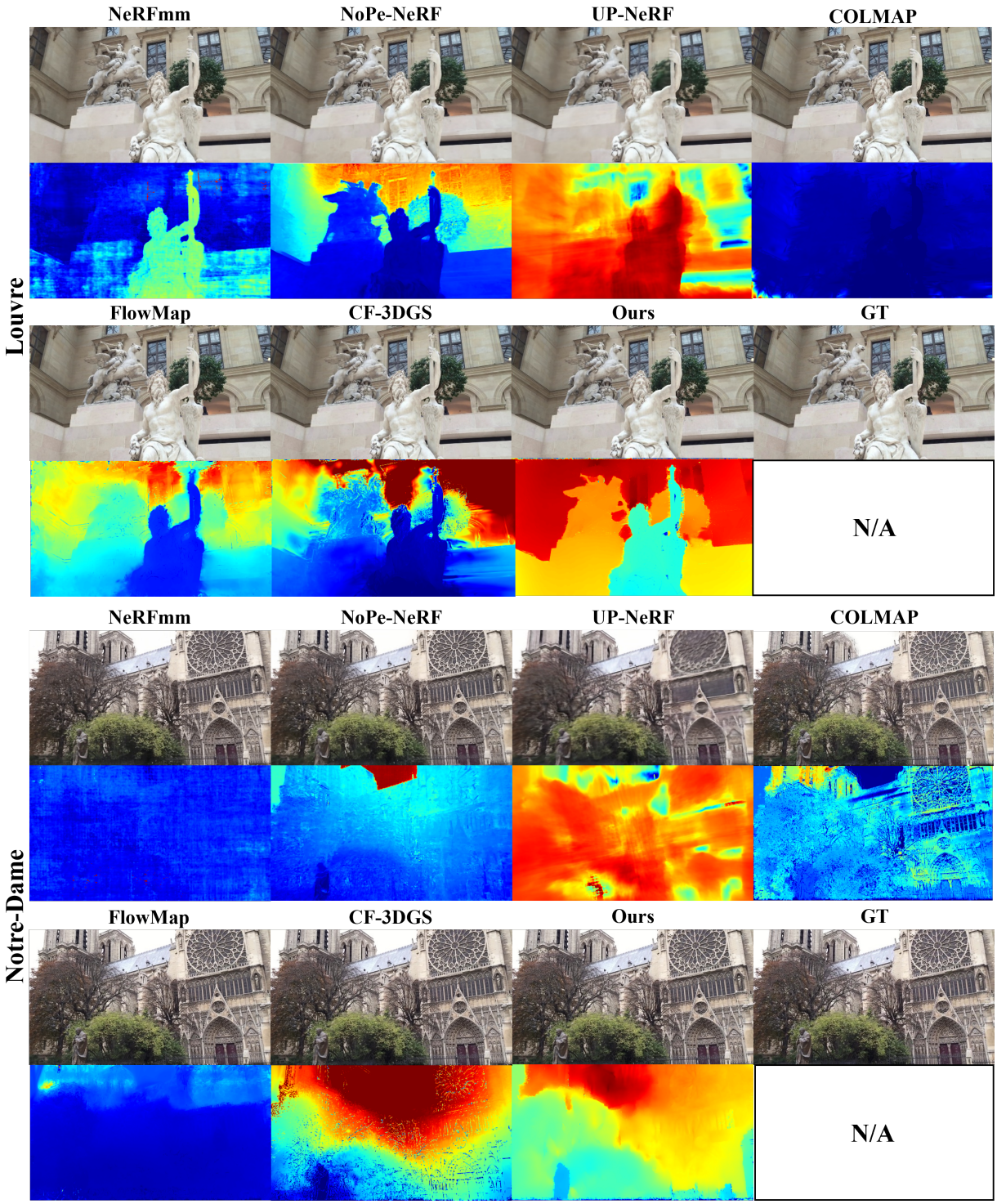

Figure 10. **Qualitative results of novel view synthesis and depth prediction on Louvre and Notre-Dame.** For each scene, we show the color (first row) and depth (second row) rendered image.

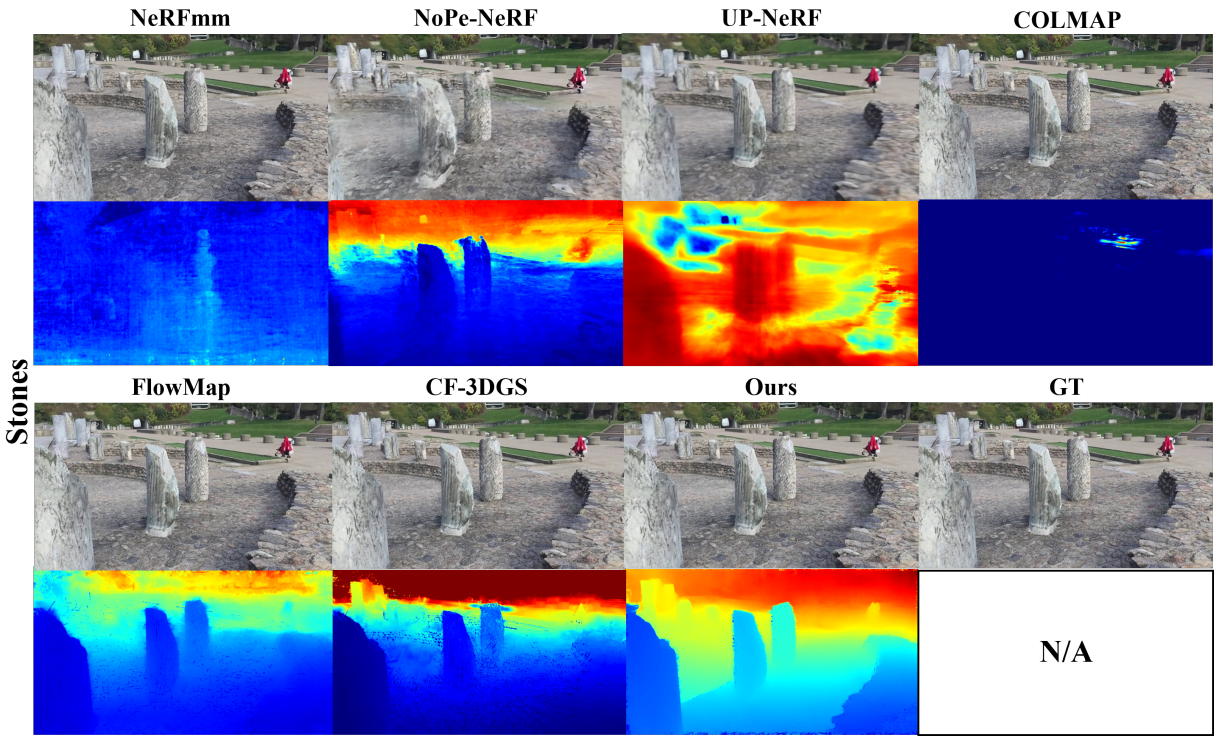

Figure 11. **Qualitative results of novel view synthesis and depth prediction on Stones.** For each scene, we show the color (first row) and depth (second row) rendered image.

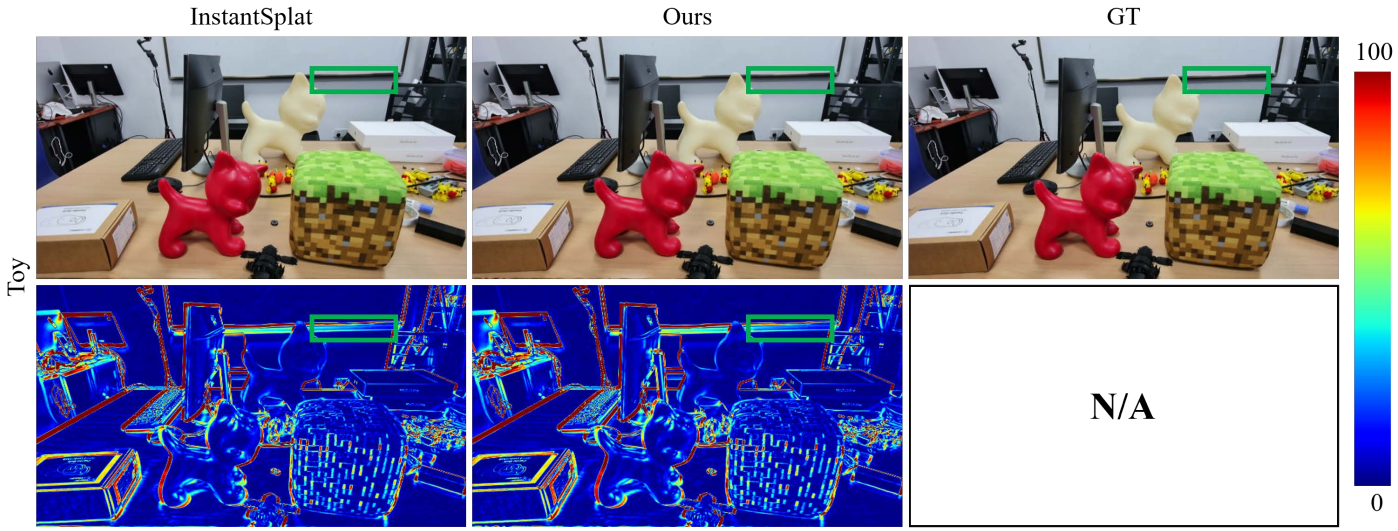

Figure 12. **Qualitative results of novel view synthesis on Toy.**The second and fourth rows respectively display the error maps of the rendered images. Please zoom in for a further comparison.

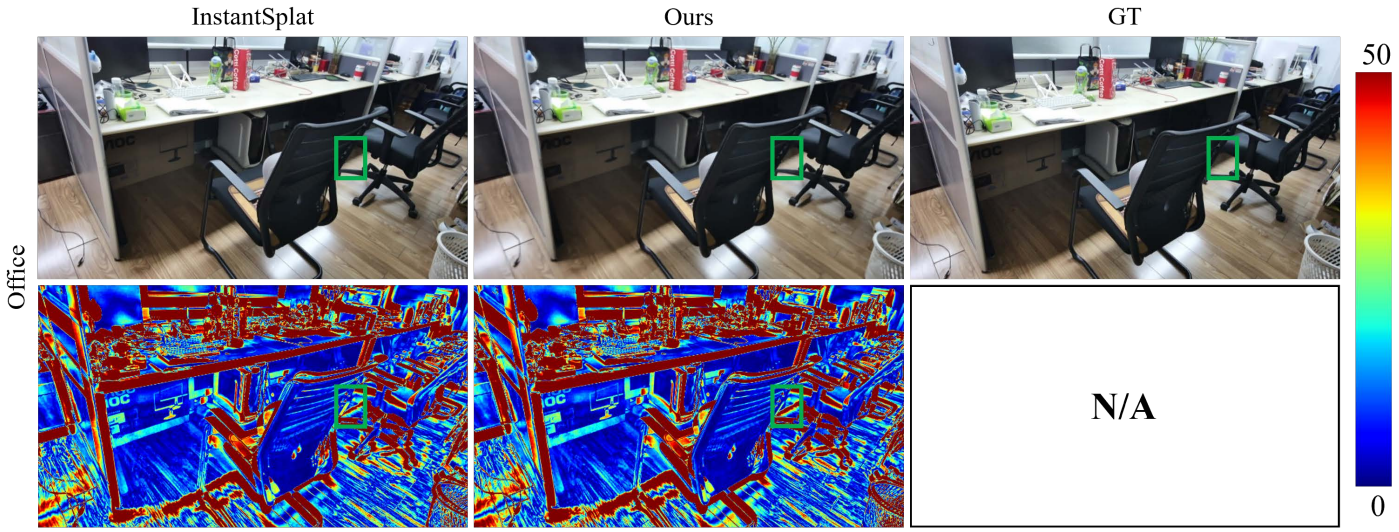

Figure 13. **Qualitative results of novel view synthesis on Office.**The second and fourth rows respectively display the error maps of the rendered images. Please zoom in for a further comparison.

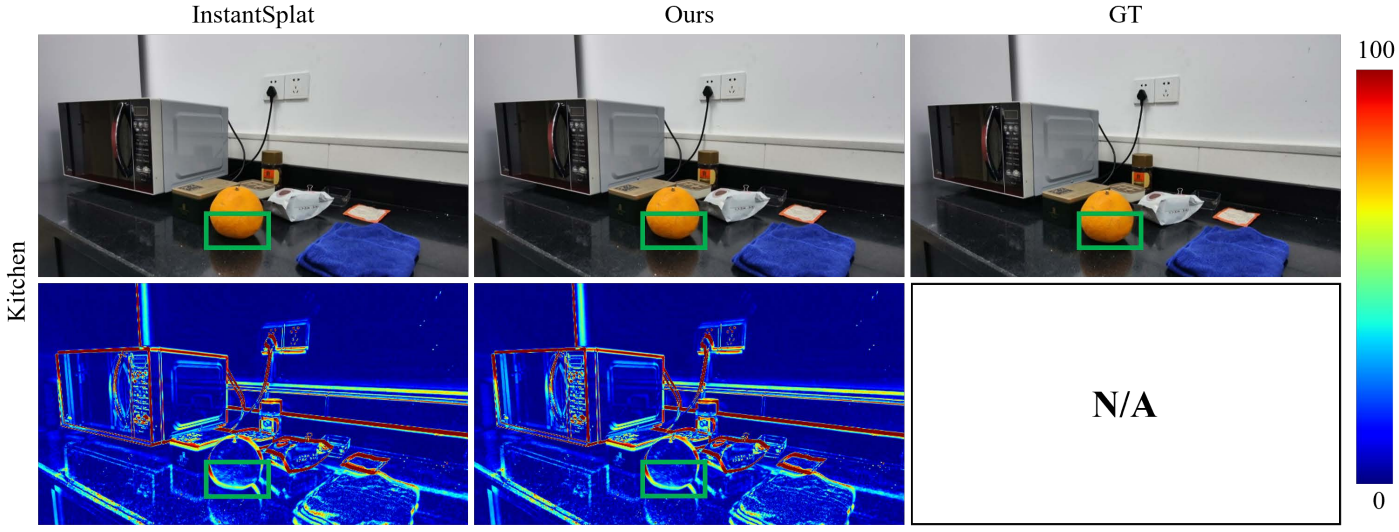

Figure 14. **Qualitative results of novel view synthesis on Kitchen.**The second and fourth rows respectively display the error maps of the rendered images. Please zoom in for a further comparison.

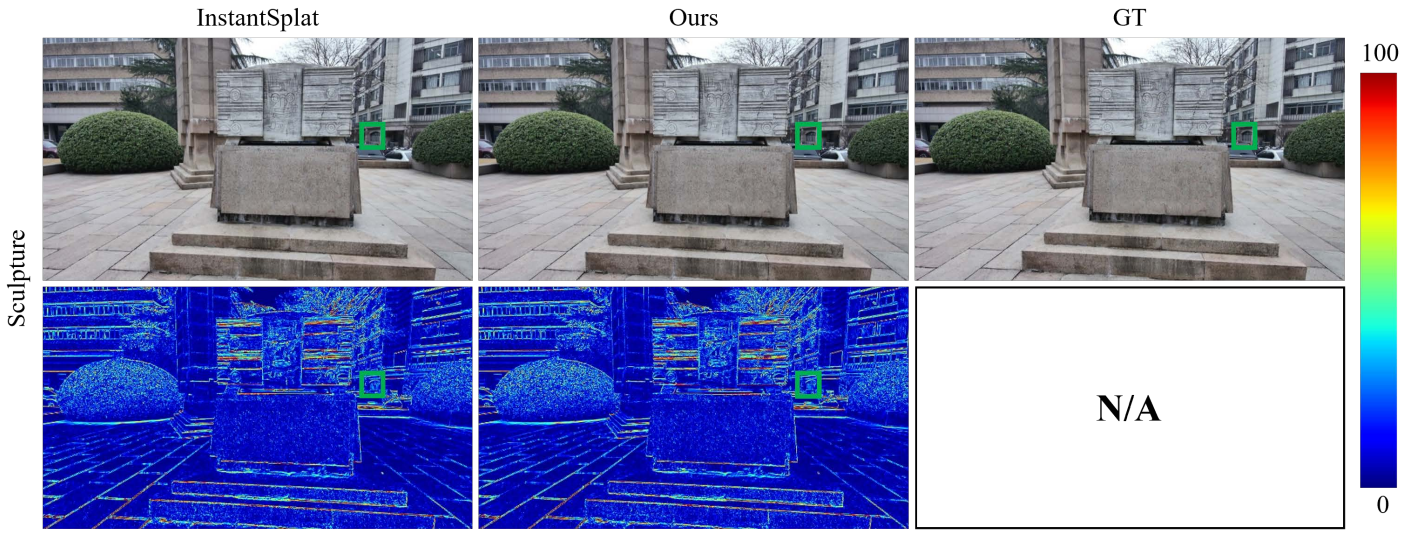

Figure 15. **Qualitative results of novel view synthesis on Sculpture.**The second and fourth rows respectively display the error maps of the rendered images. Please zoom in for a further comparison.

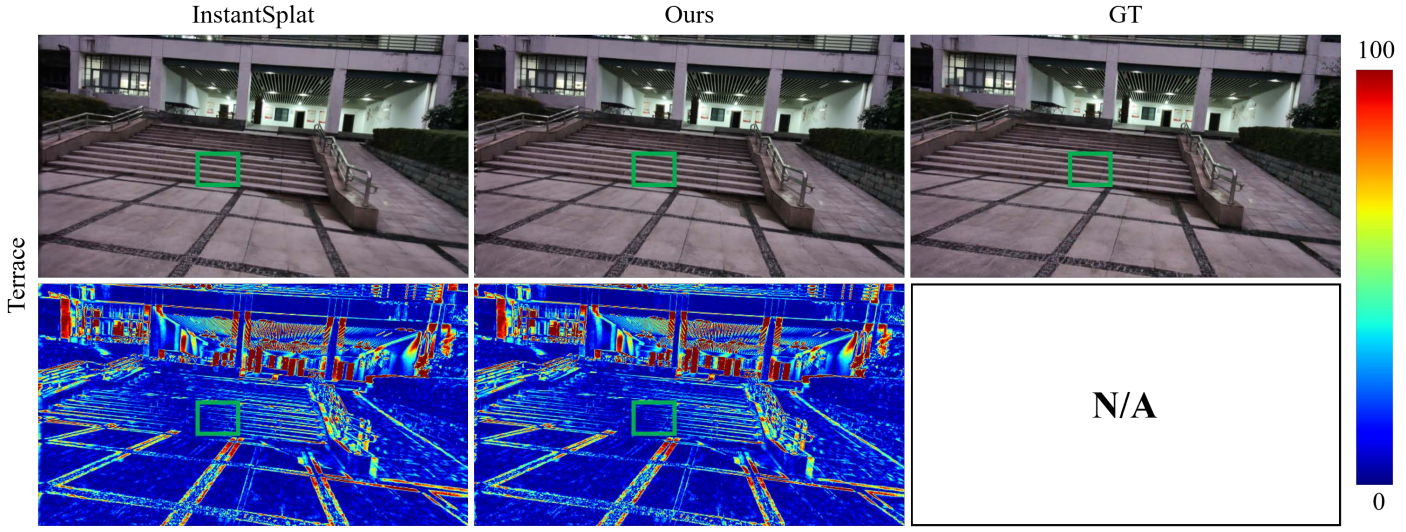

Figure 16. **Qualitative results of novel view synthesis on Terrace.** The second and fourth rows respectively display the error maps of the rendered images. Please zoom in for a further comparison.

References

[1] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4160–4169, 2023. 5

[2] Michael Grupp. evo: Python package for the evaluation of odometry and slam. 2017. 6

[3] Hyowon Ha, Sunghoon Im, Jaesik Park, Hae-Gon Jeon, and In So Kweon. High-quality depth from uncalibrated small motion clip. In *Proceedings of the IEEE conference on computer vision and pattern Recognition*, pages 5413–5421, 2016. 2

[4] Kaiwen Jiang, Yang Fu, Mukund Varma T, Yash Belhe, Xiaolong Wang, Hao Su, and Ravi Ramamoorthi. A construct-optimize approach to sparse view synthesis without camera pose. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 1, 5, 6

[5] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 6

[6] 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. 5

[7] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 2017. 1

[8] 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. 4

[9] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 1

[10] Prune Truong, Marie-Julie Rakotosaona, Fabian Manhardt, and Federico Tombari. Sparf: Neural radiance fields from sparse and noisy poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4190–4200, 2023. 1, 5, 6

[11] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04): 376–380, 1991. 6

[12] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8121–8130, 2022. 4

[13] Fisher Yu and David Gallup. 3d reconstruction from accidental motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3986–3993, 2014. 1, 2, 3, 5