COLMAP-Free 3D Gaussian Splatting

Yang Fu[1*] Sifei Liu[2] Amey Kulkarni[2] Jan Kautz[2]
Alexei A. Efros[3] Xiaolong Wang[1]
[1]UC San Deigo [2]NVIDIA [3]UC Berkeley

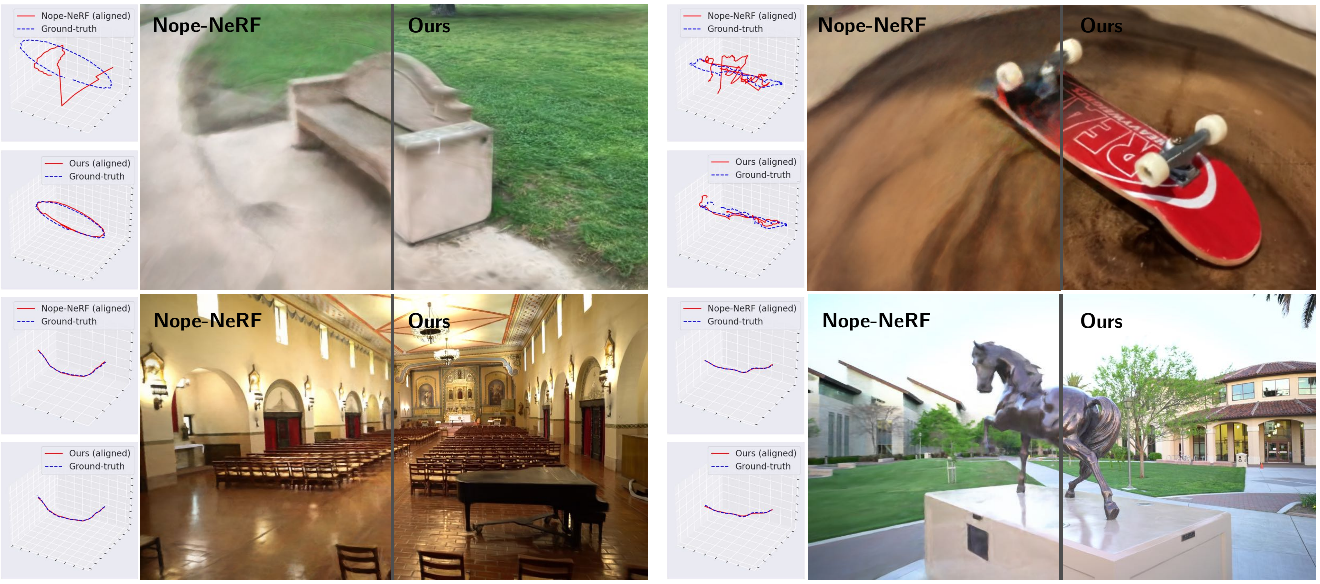

Figure 1. **Novel View Synthesis and Camera Pose Estimation Comparison.** We propose COLMAP-Free 3D Gaussian Splatting (CF-3DGS) for novel view synthesis without known camera parameters. Our method achieves more robustness in pose estimation and better quality in novel view synthesis than previous state-of-the-art methods.

Abstract

While neural rendering has led to impressive advances in scene reconstruction and novel view synthesis, it relies heavily on accurately pre-computed camera poses. To relax this constraint, multiple efforts have been made to train Neural Radiance Fields (NeRFs) without pre-processed camera poses. However, the implicit representations of NeRFs provide extra challenges to optimize the 3D structure and camera poses at the same time. On the other hand, the recently proposed 3D Gaussian Splatting provides new opportunities given its explicit point cloud representations. This paper leverages both the explicit geometric representation and the continuity of the input video stream to perform novel view synthesis without any SfM preprocessing. We process the input frames in a sequential manner and progressively grow the 3D Gaussians set by taking one input frame at a time, without the need to pre-compute the camera poses. Our method significantly improves over previous approaches in view synthesis and camera pose estimation under large motion changes. Our project page is `https://oasisyang.github.io/colmap-free-3dgs`.

1. Introduction

The field of photo-realistic scene reconstruction and view synthesis has been largely advanced with the rise of Neural Radiance Fields (NeRFs [27]). An important initialization step for training NeRF is to first prepare the camera poses for each input image. This is usually achieved by running the Structure-from-Motion (SfM) library COLMAP [39]. However, this pre-processing is not only time-consuming but also can fail due to its sensitivity to feature extraction errors and difficulties in handling textureless or repetitive regions.

Recent studies [6, 23, 49] have focused on reducing the reliance on SfM by integrating pose estimation directly within the NeRF framework. Simultaneously solving 3D scene reconstruction and camera registration has been a chicken-and-egg problem for a long time in computer vi-

[*] This work was done while Yang Fu was a part-time intern at NVIDIA.

sion. This challenge is further amplified in the context of NeRF and its implicit representation, where the optimization process often involves additional constraints. For instance, BARF [23] requires initial poses that are close to their ground truth locations, and NeRFmm [49] is largely limited to face-forwarding scenes. The recently proposed Nope-NeRF [6] takes a long time to train (30 hours) and does not work well when the camera pose changes a lot (e.g., 360 degrees), as shown in the two top cases in Fig. 1. Fundamentally, NeRFs optimize camera parameters in an indirect way, by updating the ray casting from camera positions, which makes optimization challenging.

The arrival of 3D Gaussian Splatting [17] extends the volumetric rendering in NeRFs to accommodate point clouds. While it was originally proposed with pre-computed cameras, we find it offers a new opportunity to perform view synthesis without SfM pre-processing. We propose **C**OLMAP-**F**ree **3D G**aussian **S**platting (CF-3DGS), which leverages two key ingredients: the **temporal continuity from video** and the **explicit point cloud representation**. We summarize out approach below.

Instead of optimizing with all the frames at once, we propose to build the 3D Gaussians of the scene in a continuous manner, "growing" one frame at a time as the camera moves. In this process, we will extract a *local* 3D Gaussians set for each frame, and also maintain a *global* 3D Gaussians set of the whole scene. Assuming we are iterating through $t = \{1, ..., T\}$ frames in a sequential manner, we perform a two-step procedure each time: (i) We construct a local 3D Gaussians set given frame $t-1$, and we sample the next nearby frame t. Our goal is to learn an affine transformation that can transform the 3D Gaussians in frame $t-1$ to render the pixels in frame t. Neural rendering provides the gradients for optimizing the affine transformation, which is essentially the relative camera pose between frames $t-1$ and t. This optimization is not difficult as the **explicit** point cloud representation allows us to directly apply an affine transformation on it which cannot be achieved with NeRFs, and the two frames are close (**temporal continuity**) which makes the transformation relatively small. (ii) Once we have the relative camera pose between frames $t-1$ and t, we can infer the relative pose between the first frame and frame t. This allows us to aggregate the current frame information into the global 3D Gaussians set, where we will perform optimization with the current and all the previous frames and camera poses.

We experiment with two datasets: the Tanks and Temples dataset [19] and videos randomly selected from the CO3D dataset [34]. We evaluate both view synthesis and camera pose estimation tasks, and compare with previous approaches without pre-computed camera poses. Our method performs significantly better than previous approaches on view synthesis in both datasets. For camera pose estimation, we find our method performs on par with the most recent Nope-NeRF [6] when the camera motion is small, and outperforms all approaches by a large margin when the camera changes a lot, such as in the 360-degree videos in CO3D.

2. Related Work

Novel View Synthesis. To generate realistic images from novel viewpoints, several different 3D scene representations have been employed, such as planes [12, 13], meshes [14, 35, 36], point clouds [53, 59], and multi-plane images [22, 45, 61]. Recently, NeRFs [27] have gained prominence in this field due to its exceptional capability of photorealistic rendering. Several efforts have been made on top of the vanilla NeRF for advanced rendering quality. These improvements include enhancing anti-aliasing effects [2–4, 58], refining reflectance [1, 47], training with sparse view [15, 18, 30, 52, 54], and reducing training times [28, 33, 38] and rendering time [10, 24, 43, 57].

More recently, point-cloud-based representation [17, 20, 25, 53, 56, 59] has been widely used for its efficiency during rendering. For instance, Zhang *et al.* [59] propose to learn the per-point position and view-dependent appearance, using a differentiable splat-based renderer, from point clouds initialized from object masks. Additionally, 3DGS [17] enables real-time rendering of novel views by its pure explicit representation and the novel differential point-based splatting method. However, most of these approaches still rely on pre-computed camera parameters obtained from SfM algorithms [11, 29, 39, 44].

NeRF without SfM Preprocessing. Recently, there has been growing interest in trying to eliminate the required preprocessing step of camera estimation in NeRFs. The initial effort in this direction was i-NeRF [55], which predicts camera poses by matching keypoints using a pre-trained NeRF. Following this, NeRFmm [49] introduced a method to jointly optimize the NeRF network and camera pose embeddings. However, despite its successor, SiNeRF [51], employing SIREN-MLP [41] and a mixed region sampling strategy to address NeRFmm's sub-optimal issues, it remains limited to forward-facing scenes. BARF [23] and GARF [8] propose to alleviate the gradient inconsistency issue caused by high-frequency parts of positional embedding. For instance, BARF proposes a coarse-to-fine positional encoding strategy for camera poses and NeRF joint optimization. Though they could handle more complex camera motions, they require a good initialization from the ground-truth cameras.(*e.g.*, within 15° of the ground-truth). More advanced works [6, 7, 26] seek help from some pre-trained networks, *i.e.*, monocular depth estimation and optical flow estimation, to obtain prior knowledge of geometry or correspondence. For example, Nope-NeRF [6] trains a NeRF by incorporating undistorted depth priors which are correct from the monocular depth estimation during training. Addition-

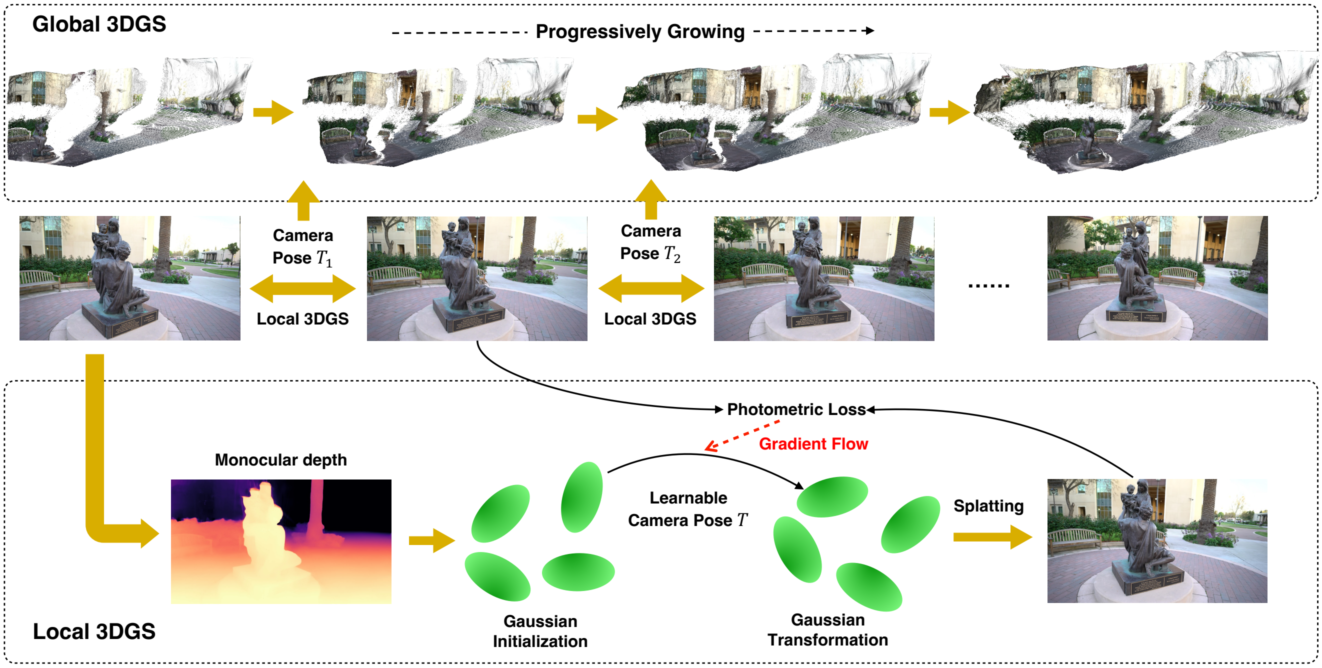

Figure 2. **Overview of proposed CF-3DGS**. Our method takes a sequence of images as input to learn a set of 3D Gaussian that presents the input scene and jointly estimates the camera poses of the frames. We first introduce a local 3DGS to estimate the relative pose of two nearby frames by approximating the Gaussian transformation. Then, a global 3DGS is utilized to model the scene by progressively growing the set of 3D Gaussian as the camera moves.

ally, VideoAE [21], RUST [37], MonoNeRF [9] and FlowCam [42] learn a generalizable scene representation from unposed videos, but their view synthesis performance is unsatisfactory without per-scene optimization.

In summary, although showing some promising results, prior works on NeRFs with unknown poses assume either small perturbations [8, 23], a narrow range of camera motion [49, 51], or prior knowledges [6, 26]. These approaches face difficulties when handling challenging camera trajectories with large camera motion, *e.g.*, 360° scenes in the CO3D [34] dataset. Furthermore, most existing works require quite a long training time, typically exceeding 10 hours. To overcome these limitations, our work proposes a joint optimization of camera parameters and 3D Gaussians, utilizing both local and global 3DGS strategies.

3. Method

Given a sequence of unposed images along with camera intrinsics, our goal is to recover the camera poses and reconstruct the photo-realistic scene. To this end, we propose CF-3DGS to optimize the 3D Gaussian Splatting (3DGS [17]) and camera poses simultaneously. We detail our method in the following sections, starting from a brief review of the representation and rendering process of 3DGS in Sec. 3.1. Then, we propose a *local* 3DGS, a simple yet effective method to estimate the relative camera pose from each pair of nearby frames, in Sec. 3.2. Finally, we introduce a *global* 3DGS, featuring a progressive expansion of the 3D Gaussians from unobserved views in sequential order, in Sec. 3.3.

3.1. Preliminary: 3D Gaussian Splatting

3DGS [17] models the scene as a set of 3D Gaussians, which is an explicit form of representation, in contrast to the implicit representation used in NeRFs. Each Gaussian is characterized by a covariance matrix Σ and a center (mean) point μ,

$$G(x) = e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)} \tag{1}$$

The means of 3D Gaussians are initialized by a set of sparse point clouds(*e.g.*, always obtained from SfM). Each Gaussian is parameterized as the following parameters: (a) a center position $\mu \in \mathbb{R}^3$; (b) spherical harmonics (SH) coefficients $c \in \mathbb{R}^k$ (k represents the degrees of freedom) that represents the color; (c) rotation factor $r \in \mathbb{R}^4$ (in quaternion rotation); (d) scale factor $s \in \mathbb{R}^3$; (e) opacity $\alpha \in \mathbb{R}$. Then, the covariance matrix Σ describes an ellipsoid configured by a scaling matrix $S = \text{diag}([s_x, s_y, s_z])$ and rotation matrix $R = \text{q2R}([r_w, r_x, r_y, r_z])$, where $q2R()$ is the formula for constructing a rotation matrix from a quaternion. Then, the covariance matrix can be computed as follows,

$$\Sigma = RSS^\top R^\top \tag{2}$$

In order to optimize the parameters of 3D Gaussians to represent the scene, we need to render them into images in a differentiable manner. As introduced in [17], the rendering from a given camera view W involves the process of splatting the Gaussian onto the image plane, which is achieved by approximating the projection of a 3D Gaussian along the depth dimension into pixel coordinates. Given a viewing transform W (also known as the camera pose), the covariance matrix Σ^{2D} in camera coordinates can be expressed by

$$\Sigma^{\text{2D}} = JW\Sigma W^{\top} J^{\top} \tag{3}$$

where J is the Jacobian of the affine approximation of the projective transformation. For each pixel, the color and opacity of all the Gaussians are computed using Eq. 1, and the final rendered color can be formulated as the alpha-blending of N ordered points that overlap the pixel,

$$C_{pix} = \sum_{i}^{N} c_i \alpha_i \prod_{j}^{i-1} (1 - \alpha_j) \tag{4}$$

where c_i, α_i represents the density and color of this point computed from the learnable per-point opacity and SH color coefficients weighted by the Gaussian covariance Σ, which we ignore in Eq. 4 for simplicity.

To perform scene reconstruction, given the ground truth poses that determine the projections, we fit a set of initialized Gaussian points to the desired objects or scenes by learning their parameters, i.e., μ and Σ. With the differentiable renderer as in Eq. 4, all those parameters, along with the SH and opacity, can be easily optimized through a photometric loss. In our approach, we reconstruct scenes following the same process, but replacing the ground truth poses with the estimated ones, as detailed in the following sections.

3.2. Local 3DGS for Relative Pose Estimation

Previous studies [6, 16, 23] have demonstrated the feasibility of simultaneously estimating camera parameters and optimizing a Neural Radiance Field (NeRF). This typically involves the integration of various regularization terms and geometric priors. However, rather than directly optimizing camera poses, most existing methods prioritize optimizing the ray casting process from varying camera positions. This is dictated by the nature of implicit representation and the implementation of ray tracing in NeRFs. This indirect approach often results in a complex and challenging optimization under large camera movement scenarios.

On the other hand, 3DGS [17] utilizes an explicit scene representation in the form of point clouds enabling straightforward deformation and movement, as demonstrated in its recent application to dynamic scenes [25, 50]. To take advantage of 3DGS, we introduce a *local* 3DGS to estimate the relative camera pose.

We reveal the relationship between the camera pose and the 3D rigid transformation of Gaussian points, in the following. Given a set of 3D Gaussians with centers μ, projecting them with the camera pose W yields:

$$\mu_{\text{2D}} = K(W\mu)/(W\mu)_z \tag{5}$$

where K is the intrinsic projection matrix. Alternatively, the 2D projection μ_{2D} can be obtained from the orthogonal direction $\mathbb{I}$ of a set of rigidly transformed points, i.e., $\mu' = W\mu$, which yields $\mu_{\text{2D}} := K(\mathbb{I}\mu')/(\mathbb{I}\mu')_z$. As such, estimating the camera poses W is equivalent to estimating the transformation of a set of 3D Gaussian points. Based on this finding, we designed the following algorithm to estimate the relative camera pose.

Initialization from a single view. As demonstrated in Fig. 2 (bottom part), given a frame I_t at timestep t, we first utilize an off-the-shelf monocular depth network, *i.e.*, DPT [32], to generate the monocular depth, denoted as D_t. Given that monocular depth D_t offers strong geometric cues without needing camera parameters, we initialize 3DGS with points lifted from monocular depth, leveraging camera intrinsic and identity camera pose, instead of the original SfM points. After initialization, we learn a set of 3D Gaussian G_t with all attributes to minimize the photometric loss between the rendered image and the current frame I_t,

$$G_t{}^* = \arg \min_{c_t, r_t, s_t, \alpha_t} \mathcal{L}_{rgb}(\mathcal{R}(G_t), I_t), \tag{6}$$

where $\mathcal{R}$ is the 3DGS rendering process. The photometric loss $\mathcal{L}_{rgb}$ is $\mathcal{L}_1$ combined with a D-SSIM:

$$\mathcal{L}_{rgb} = (1 - \lambda)\mathcal{L}_1 + \lambda \mathcal{L}_{\text{D-SSIM}} \tag{7}$$

We use $\lambda = 0.2$ for all experiments. This step is quite lightweight to run and only takes around 5s to fit the input frame I_t.

Pose Estimation by 3D Gaussian Transformation. To estimate the relative camera pose, we transform the pre-trained 3D Gaussian $G_t{}^*$ by a learnable SE-3 affine transformation T_t into frame $t+1$, denoted as $G_{t+1} = T_t \odot G_t$. The transformation T_t is optimized by minimizing the photometric loss between the rendered image and the next frame I_{t+1}

$$T_t{}^* = \arg \min_{T_t} \mathcal{L}_{rgb}(\mathcal{R}(T_t \odot G_t), I_{t+1}), \tag{8}$$

Note that during the aforementioned optimization process, we freeze all attributes of the pre-trained 3D Gaussian $G_t{}^*$ to separate the camera movement from the deformation, densification, pruning, and self-rotation of the 3D Gaussian points. The transformation T is represented in form of quaternion rotation $\mathbf{q} \in \mathfrak{so}(3)$ and translation vector $\mathbf{t} \in \mathbb{R}^3$. As two adjunct frames are close, the transformation is relatively small and easier to optimize. Similar to the initialization phase, the pose optimization step is also quite efficient, typically requiring only 5-10 seconds.

scenes	Ours			Nope-NeRF			BARF			NeRFmm			SC-NeRF		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Church	**30.23**	**0.93**	**0.11**	25.17	0.73	0.39	23.17	0.62	0.52	21.64	0.58	0.54	21.96	0.60	0.53
Barn	**31.23**	**0.90**	**0.10**	26.35	0.69	0.44	25.28	0.64	0.48	23.21	0.61	0.53	23.26	0.62	0.51
Museum	**29.91**	**0.91**	**0.11**	26.77	0.76	0.35	23.58	0.61	0.55	22.37	0.61	0.53	24.94	0.69	0.45
Family	**31.27**	**0.94**	**0.07**	26.01	0.74	0.41	23.04	0.61	0.56	23.04	0.58	0.56	22.60	0.63	0.51
Horse	**33.94**	**0.96**	**0.05**	27.64	0.84	0.26	24.09	0.72	0.41	23.12	0.70	0.43	25.23	0.76	0.37
Ballroom	**32.47**	**0.96**	**0.07**	25.33	0.72	0.38	20.66	0.50	0.60	20.03	0.48	0.57	22.64	0.61	0.48
Francis	**32.72**	**0.91**	**0.14**	29.48	0.80	0.38	25.85	0.69	0.57	25.40	00.69	0.52	26.46	0.73	0.49
Ignatius	**28.43**	**0.90**	**0.09**	23.96	0.61	0.47	21.78	0.47	0.60	21.16	0.45	0.60	23.00	0.55	0.53
mean	**31.28**	**0.93**	**0.09**	26.34	0.74	0.39	23.42	0.61	0.54	22.50	0.59	0.54	23.76	0.65	0.48

Table 1. **Novel view synthesis results on Tanks and Temples**. Each baseline method is trained with its public code under the original settings and evaluated with the same evaluation protocol. The best results are highlighted in bold.

3.3. Global 3DGS with Progressively Growing

By employing the local 3DGS on every pair of images, we can infer the relative pose between the first frame and any frame at timestep t. However, these relative poses could be noisy resulting in a dramatic impact on optimizing a 3DGS for the whole scene (see Table 5). To tackle this issue, we propose to learn a global 3DGS progressively in a sequential manner.

As described in the top part of Fig. 2, starting from the tth frame I_t, we first initialize a set of 3D Gaussian points with the camera pose set as orthogonal, as aforementioned. Then, utilizing the local 3DGS, we estimate the relative camera pose between frames I_t and I_{t+1}. Following this, the global 3DGS updates the set of 3D Gaussian points, along with all attributes, over N iterations, using the estimated relative pose and the two observed frames as inputs. As the next frame I_{t+2} becomes available, this process is repeated: we estimate the relative pose between I_{t+1} and I_{t+2}, and subsequently infer the relative pose between I_t and I_{t+2}.

To update the global 3DGS to cover the new view, we densify the Gassians that are ”under-reconstruction” as new frames arrive. As suggested in [17], we determine the candidates for densification by the average magnitude of view-space position gradients. Intuitively, the unobserved frames always contain regions that are not yet well reconstructed, and the optimization tries to move the Gaussians to correct with a large gradient step. Therefore, to make the densification concentrate on the unobserved content/regions, we densify the global 3DGS every N steps that aligns with the pace of adding new frames. In addition, instead of stopping the densification in the middle of the training stage, we keep growing the 3D Gaussian points until the end of the input sequence. By iteratively applying both local and global 3DGS, the global 3DGS will grow progressively from the initial partial point cloud to the completed point cloud that covers the whole scene throughout the entire sequence, and simultaneously accomplish photo-realistic reconstruction and accurate camera pose estimation.

4. Experiments

4.1. Experimental Setup

Datasets. We conduct extensive experiments on different real-world datasets, including Tanks and Temples [19] and CO3D-V2 [34]. **Tanks and Temples:** Similar to [6], we evaluate novel view synthesis quality and pose estimation accuracy on 8 scenes covering both indoor and outdoor scenes. For each scene, we sample 7 images from every 8-frame clip as training samples and test the novel view synthesis quality on the remaining 1/8 images. The camera poses are estimated and evaluated on all training samples after Umeyama alignment [46]. **CO3D-V2:** It consists of thousands of object-centric videos where the whole object is kept in view while moving a full circle around it. Compared with Tanks and Temples, recovering camera poses from CO3D videos is much harder as it involves large and complicated camera motions. We randomly select 5 scenes[1] of different categories objects and follow the same protocol to split the train/test set.

Metrics. We evaluate the tasks of novel view synthesis and camera pose estimation. For novel view synthesis, we For camera pose estimation, we report the camera rotation and translation error, including the Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) as in [6, 23]. For novel view synthesis, we report the standard evaluation metrics including PSNR, SSIM [48], and LPIPS [60].

Implementation Details. Our implementation is primarily based on the PyTorch [31] framework and we follow the optimization parameters by the configuration outlined in the 3DGS [17] unless otherwise specified. Notably, we set the number of steps of adding new frames equal to the intervals of point densification, in order to achieve progressive growth of the whole scene. Further, we keep resetting the opacity until the end of the training process, which enables us to integrate the new frames into the Gaussian model established from observed frames. Moreover, the camera poses are optimized in the representation of quaternion rotation

[1] We specify all selected scenes in the supplementary material.

scenes	Ours			Nope-NeRF			BARF			NeRFmm			SC-NeRF		
	RPE$_t$ ↓	RPE$_r$ ↓	ATE↓	RPE$_t$	RPE$_r$	ATE	RPE$_t$	RPE$_r$	ATE	RPE$_t$	RPE$_r$	ATE	RPE$_t$	RPE$_r$	ATE
Church	0.008	0.018	0.002	0.034	0.008	0.008	0.114	0.038	0.052	0.626	0.127	0.065	0.836	0.187	0.108
Barn	0.034	0.034	0.003	0.046	0.032	0.004	0.314	0.265	0.050	1.629	0.494	0.159	1.317	0.429	0.157
Museum	0.052	0.215	0.005	0.207	0.202	0.020	3.442	1.128	0.263	4.134	1.051	0.346	8.339	1.491	0.316
Family	0.022	0.024	0.002	0.047	0.015	0.001	1.371	0.591	0.115	2.743	0.537	0.120	1.171	0.499	0.142
Horse	0.112	0.057	0.003	0.179	0.017	0.003	1.333	0.394	0.014	1.349	0.434	0.018	1.366	0.438	0.019
Ballroom	0.037	0.024	0.003	0.041	0.018	0.002	0.531	0.228	0.018	0.449	0.177	0.031	0.328	0.146	0.012
Francis	0.029	0.154	0.006	0.057	0.009	0.005	1.321	0.558	0.082	1.647	0.618	0.207	1.233	0.483	0.192
Ignatius	0.033	0.032	0.005	0.026	0.005	0.002	0.736	0.324	0.029	1.302	0.379	0.041	0.533	0.240	0.085
mean	**0.041**	0.069	**0.004**	0.080	**0.038**	0.006	1.046	0.441	0.078	1.735	0.477	0.123	1.890	0.489	0.129

Table 2. **Pose accuracy on Tanks and Temples**. Note that we use COLMAP poses in Tanks and Temples as the "ground truth". The unit of RPE$_r$ is in degrees, ATE is in the ground truth scale and RPE$_t$ is scaled by 100. The best results are highlighted in bold.

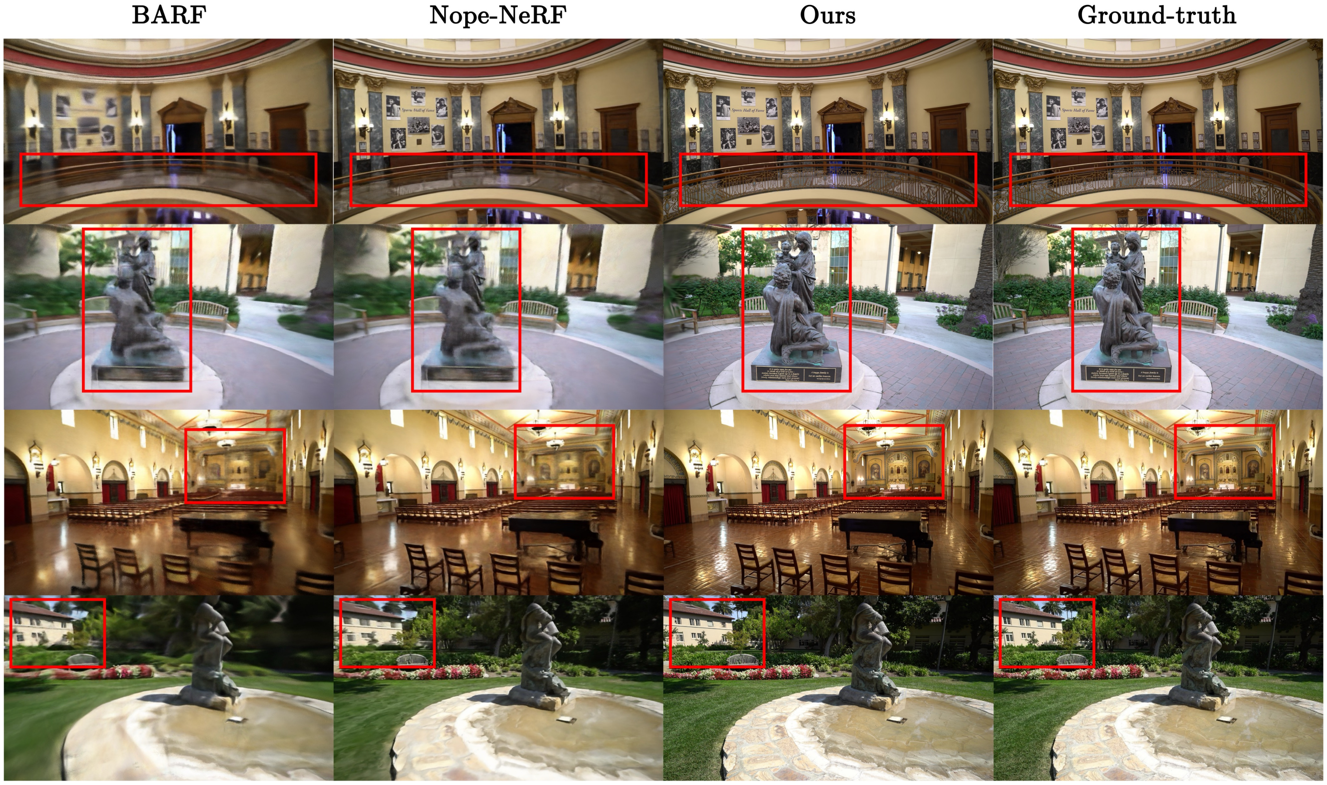

Figure 3. **Qualitative comparison for novel view synthesis on Tanks and Temples.** Our approach produces more realistic rendering results than other baselines. **Better viewed when zoomed in.**

$\mathbf{q} \in \mathfrak{so}(3)$ and translation vector $\mathbf{t} \in \mathbb{R}^3$. The initial learning rate is set to 10^{-5} and gradually decay to 10^{-6} until convergence. All experiments are conducted on a single single RTX 3090 GPU. More details are provided in the supplementary material.

4.2. Comparing with Pose-Unknown Methods

In this subsection, we compare our method with several baselines including, Nope-NeRF [6], BARF [23], NeRFmm [49] and SC-NeRF [16] on both novel view synthesis and camera pose estimation.

Novel View Synthesis. Different from the standard setting where the camera poses of testing views are given, we need to first obtain the camera poses of test views for rendering. Inspired by NeRFmm [49], we freeze the pre-trained 3DGS model that trained on the training views, while optimizing testing views' camera poses via minimizing the photometric error between the synthesised images and the test views. To speed up convergence, the camera pose of each test view is initialised by the closest camera position from the learnt camera poses of all training frames, which are then fine-tuned with the photometric error to obtain the testing camera pose. The same procedure is performed on all baselines for a fair comparison.

Method	Times ↓	110_13051_23361			415_57112_110099			106_12648_23157			245_26182_52130			34_1403_4393		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Nope-NeRF [6]	∼30 h	26.86	0.73	0.47	24.78	0.64	0.55	20.41	0.46	0.58	25.05	0.80	0.49	28.62	0.80	0.35
Ours	**∼2 h**	**29.69**	**0.89**	**0.29**	**26.21**	**0.73**	**0.32**	**22.14**	**0.64**	**0.34**	**27.24**	**0.85**	**0.30**	27.75	**0.86**	**0.20**

Table 3. **Novel view synthesis results on CO3D V2**. Each baseline method is trained with its public code under the original settings and evaluated with the same evaluation protocol. The best results are highlighted in bold.

Method	Times ↓	110_13051_23361			415_57112_110099			106_12648_23157			245_26182_52130			34_1403_4393		
		RPE$_t$ ↓	RPE$_r$ ↓	ATE↓	RPE$_t$	RPE$_r$	ATE	RPE$_t$	RPE$_r$	ATE	RPE$_t$	RPE$_r$	ATE	RPE$_t$	RPE$_r$	ATE
Nope-NeRF [6]	∼30 h	0.400	1.966	0.046	0.326	1.919	0.054	0.387	1.312	0.049	0.587	1.867	0.038	0.591	1.313	0.053
Ours	**∼2 h**	**0.140**	**0.401**	**0.021**	**0.110**	**0.424**	**0.014**	**0.094**	**0.360**	**0.008**	**0.239**	**0.472**	**0.017**	**0.505**	**0.211**	**0.009**

Table 4. **Pose accuracy on CO3D V2**. Note that the camera poses provided by CO3D as the "ground truth". The unit of RPE$_r$ is in degrees, ATE is in the ground truth scale and RPE$_t$ is scaled by 100. The best results are highlighted in bold.

scenes	w.o. growing				Ours			
	PSNR	SSIM	RPE$_t$	RPE$_r$	PSNR	SSIM	RPE$_t$	RPE$_r$
Church	22.01	0.72	0.044	0.122	30.23	0.93	0.008	0.018
Barn	25.20	0.85	0.152	0.232	31.23	0.90	0.034	0.034
Museum	20.95	0.70	0.079	0.212	29.91	0.91	0.052	0.215
Family	22.30	0.77	0.065	0.028	31.27	0.94	0.022	0.024
Horse	23.47	0.81	0.147	0.066	33.94	0.96	0.112	0.057
Ballroom	23.36	0.79	0.056	0.073	32.47	0.96	0.037	0.024
Francis	22.20	0.69	0.147	0.161	32.72	0.91	0.029	0.154
Ignatius	21.05	0.67	0.24	0.058	28.43	0.90	0.033	0.032
mean	22.57	0.75	0.116	0.119	**31.28**	**0.93**	**0.041**	**0.069**

Table 5. **Ablation for Progressively Growing on Tanks and Temples.** Performance on both novel view synthesis and camera pose estimation. The best results are highlighted in bold.

scenes	w. depth				Ours			
	PSNR	SSIM	RPE$_t$	RPE$_r$	PSNR	SSIM	RPE$_t$	RPE$_r$
Church	28.93	0.91	0.008	0.017	30.23	0.93	0.008	0.018
Barn	28.70	0.87	0.029	0.033	31.23	0.90	0.034	0.034
Museum	26.92	0.83	0.049	0.216	29.91	0.91	0.052	0.215
Family	29.05	0.94	0.021	0.024	31.27	0.94	0.022	0.024
Horse	30.86	0.94	0.108	0.054	33.94	0.96	0.112	0.057
Ballroom	30.38	0.94	0.038	0.018	32.47	0.96	0.037	0.024
Francis	29.97	0.88	0.029	0.154	32.72	0.91	0.029	0.154
Ignatius	26.69	0.87	0.032	0.033	28.43	0.90	0.033	0.032
mean	28.94	0.90	0.039	0.069	**31.28**	**0.93**	0.041	0.069

Table 6. **Ablation study of depth loss on Tanks and Temples.** We report the performance on both novel view synthesis and camera pose estimation. The best results are highlighted in bold.

We report the comparison results on Tanks and Temples in Table 1. Our method consistently consistently outperforms the others across all metrics. Notably, compared to Nope-NeRF [6] which takes a significantly longer training time, our approach achieves superior results within a shorter training duration (*e.g.*, 25 hrs vs. 1.5 hrs). We also show the qualitative results in Fig. 3. As illustrated in Fig. 3, The images synthesized through our approach are significantly sharper and clearer than those produced by the other methods, as evidenced by the notably higher scores in terms of SSIM and LPIPS, as shown in Table 1.

Camera Pose Estimation. The learnt camera poses are post-processed by Procrustes analysis as in [6, 23] and compared with the ground-truth poses of training views. The quantitative results of camera pose estimation are summarized in Table 2. Our approach achieves comparable performance with the current state-of-the-art results. We hypothesize that the relatively poorer performance in terms of RPE$_r$ and RPE$_t$ may be attributed to relying solely on photometric loss for relative pose estimation in a local region. In contrast, Nope-NeRF incorporates additional constraints on relative poses beyond photometric loss, including the chamfer distance between two point clouds. As indicated in [6], omitting the point cloud loss leads to a significant decrease in pose accuracy.

4.3. Results on Scenes with Large Camera Motions

Given that the camera motion in scenes from the Tanks and Temples dataset is relatively small, we further demonstrate the robustness of our approach by validating it on the CO3D videos, which present more complex and challenging camera movements. We first evaluate the quality of synthesised images following the same evaluation procedure used in Tanks and Temples data. As demonstrated in Table 3, for novel view synthesis, our approach also significantly outperforms Nope-NeRF which corroborates the conclusions drawn from experiments conducted on the Tanks and Temples dataset. More qualitative results are shown in Fig. 4.

In addition, we evaluated camera pose estimation on the CO3D V2 dataset with the provided ground-truth poses for reference. As detailed in Table 4, different from the on-par results on Tanks and Temples, our approach consistently surpasses Nope-NeRFe across all metrics by a large margin when testing on CO3D V2. This enhanced performance demonstrates the robustness and accuracy of our proposed method in estimating camera poses, especially in scenarios with complex camera movements.

4.4. Ablation Study

In this section, we analyse the effectiveness of different pipeline designs and components that have been added to

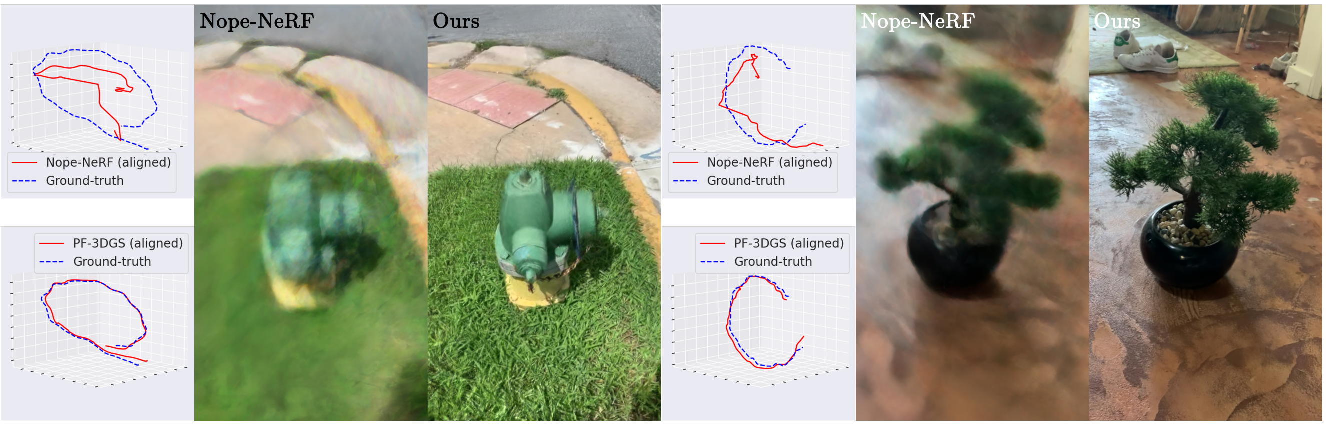

Figure 4. **Qualitative comparison for novel view synthesis and camera pose estimation on CO3D V2.** Our approach estimates camera pose much more robust than Nope-NeRF, and thus generates higher quality rendering images. **Better viewed when zoomed in.**

our approach.

Effectiveness of Progressively Growing. We first validate the effectiveness of progressively growing by removing it from the optimization of the global 3DGS. In other words, we change the current one-stage pipeline to a two-stage process, where the camera pose estimation and 3D Gaussian splatting are learnt in two separate steps. We report the performance on novel view synthesis and camera poses estimation w./w.o progressively growing in Table 5. We observe that the progressively growing is essential for enhancing both novel view synthesis and pose estimation. Without progressive growth, 3DGS is unable to utilize the continuity present in videos, which results in the unstable optimization of the global 3DGS model.

RGB Loss vs Depth Loss. Depth-related losses play a crucial role in some advanced pose-unknown approaches, such as Nope-NeRF [6]. To evaluate the significance of depth-related loss, we employed both RGB and depth loss as the objective function during the optimization. As listed in Table 6, we observe that the depth loss is not as effective as used in NeRFs. The performance on novel view synthesis is even better when merely using the photometric loss.

Comparison with 3DGS with COLMAP poses. We also compare the novel view synthesis quality of our proposed framework against the original 3DGS [17], which was trained using COLMAP-derived poses on the Tanks and Temples dataset. As indicated in Table 7, our joint optimization framework achieves performance comparable to the 3DGS model trained with COLMAP-assisted poses.

5. Conclusion

In this work, we present CF-3DGS, an end-to-end framework for joint camera pose estimation and novel view synthesis from a sequence of images. We demonstrate that previous works either have difficulty handling large camera motions or require extremely long training durations. Diverging from the implicit representation of NeRFs, our approach utilizes explicit point clouds to represent scenes. Leveraging the capabilities of 3DGS and the continuity inherent in video streams, our method sequentially processes input frames, progressively expanding the 3D Gaussians to reconstruct the entire scene. We show the effectiveness and robustness of our approach on challenging scenes like 360° videos. Thanks to the advantages of Gaussian splatting, our approach achieves rapid training and inference speeds.

scenes	Ours			COLMAP + 3DGS		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Church	30.23	0.93	0.11	29.93	0.93	0.09
Barn	31.23	0.90	0.10	31.08	0.95	0.07
Museum	29.91	0.91	0.11	34.47	0.96	0.05
Family	31.27	0.94	0.07	27.93	0.92	0.11
Horse	33.94	0.96	0.05	20.91	0.77	0.23
Ballroom	32.47	0.96	0.07	34.48	0.96	0.04
Francis	32.72	0.91	0.14	32.64	0.92	0.15
Ignatius	28.43	0.90	0.09	30.20	0.93	0.08
mean	**31.28**	**0.93**	**0.09**	30.20	0.92	0.10

Table 7. **Comparison to 3DGS trained with COLMAP poses**. We report the performance of novel view synthesis using ours and vanilla 3DGS. The best results are highlighted in bold

Limitations. Our proposed method optimizes camera pose and 3DGS jointly in a sequential manner, thereby restricting its application primarily to video streams or ordered image collections. Exploring extensions of our work to accommodate unordered image collections represents an intriguing direction for future research.

Acknowledgements. This paper was done as a part-time internship project at NVIDIA LPR. Besides, it was supported, in part, by NSF CAREER Award IIS-2240014, Amazon Research Award, Intel Rising Star Faculty Award, and Qualcomm Innovation Fellowship.

References

[1] Benjamin Attal, Jia-Bin Huang, Christian Richardt, Michael Zollhöfer, Johannes Kopf, Matthew O'Toole, and Changil Kim. Hyperreel: High-fidelity 6-dof video with ray-conditioned sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16610–16620, 2023.

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021.

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022.

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *arXiv preprint arXiv:2304.06706*, 2023.

[5] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth. *arXiv preprint arXiv:2302.12288*, 2023.

[6] 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.

[7] Zezhou Cheng, Carlos Esteves, Varun Jampani, Abhishek Kar, Subhransu Maji, and Ameesh Makadia. Lu-nerf: Scene and pose estimation by synchronizing local unposed nerfs. *arXiv preprint arXiv:2306.05410*, 2023.

[8] Shin-Fang Chng, Sameera Ramasinghe, Jamie Sherrah, and Simon Lucey. Garf: Gaussian activated radiance fields for high fidelity reconstruction and pose estimation. *arXiv e-prints*, 2022.

[9] Yang Fu, Ishan Misra, and Xiaolong Wang. Mononerf: Learning generalizable nerfs from monocular videos without camera pose. In *ICML*, 2023.

[10] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *ICCV*, 2021.

[11] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. 2003.

[12] Derek Hoiem, Alexei A Efros, and Martial Hebert. Automatic photo pop-up. In *ACM SIGGRAPH 2005 Papers*, pages 577–584, 2005.

[13] Youichi Horry, Ken-Ichi Anjyo, and Kiyoshi Arai. Tour into the picture: using a spidery mesh interface to make animation from a single image. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 225–232, 1997.

[14] Ronghang Hu, Nikhila Ravi, Alex Berg, and Deepak Pathak. Worldsheet: Wrapping the world in a 3d sheet for view synthesis from a single image. In *ICCV*, 2020.

[15] Muhammad Zubair Irshad, Sergey Zakharov, Katherine Liu, Vitor Guizilini, Thomas Kollar, Adrien Gaidon, Zsolt Kira, and Rares Ambrus. Neo 360: Neural fields for sparse view synthesis of outdoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9187–9198, 2023.

[16] Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-calibrating neural radiance fields. In *ICCV*, 2021.

[17] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4): 1–14, 2023.

[18] Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *CVPR*, 2022.

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

[20] Georgios Kopanas, Thomas Leimkühler, Gilles Rainer, Clément Jambon, and George Drettakis. Neural point catacaustics for novel-view synthesis of reflections. *ACM Transactions on Graphics (TOG)*, 41(6):1–15, 2022.

[21] Zihang Lai, Sifei Liu, Alexei A Efros, and Xiaolong Wang. Video autoencoder: self-supervised disentanglement of 3d structure and motion. In *ICCV*, 2021.

[22] Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. Mine: Towards continuous depth mpi with nerf for novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12578–12588, 2021.

[23] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *ICCV*, 2021.

[24] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020.

[25] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023.

[26] Andreas Meuleman, Yu-Lun Liu, Chen Gao, Jia-Bin Huang, Changil Kim, Min H Kim, and Johannes Kopf. Progressively optimized local radiance fields for robust view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16539–16548, 2023.

[27] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 2021.

[28] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*

[29] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 2015.

[30] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *CVPR*, 2022.

[31] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[32] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, 2021.

[33] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021.

[34] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10901–10911, 2021.

[35] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *ECCV*, 2020.

[36] Gernot Riegler and Vladlen Koltun. Stable view synthesis. In *CVPR*, 2021.

[37] Mehdi SM Sajjadi, Aravindh Mahendran, Thomas Kipf, Etienne Pot, Daniel Duckworth, Mario Lučić, and Klaus Greff. Rust: Latent neural scene representations from unposed imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17297–17306, 2023.

[38] Sara Fridovich-Keil and Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022.

[39] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016.

[40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[41] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *NeurIPS*, 2020.

[42] Cameron Smith, Yilun Du, Ayush Tewari, and Vincent Sitzmann. Flowcam: Training generalizable 3d radiance fields without camera poses via pixel-aligned scene flow. *arXiv preprint arXiv:2306.00180*, 2023.

[43] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022.

[44] Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. Visual slam algorithms: A survey from 2010 to 2016. *IPSJ Transactions on Computer Vision and Applications*, 9(1):1–11, 2017.

[45] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *CVPR*, 2020.

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

[47] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022.

[48] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 2004.

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

[50] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023.

[51] Yitong Xia, Hao Tang, Radu Timofte, and Luc Van Gool. Sinerf: Sinusoidal neural radiance fields for joint pose estimation and scene reconstruction. 2022.

[52] Dejia Xu, Yifan Jiang, Peihao Wang, Zhiwen Fan, Humphrey Shi, and Zhangyang Wang. Sinnerf: Training neural radiance fields on complex scenes from a single image. In *European Conference on Computer Vision*, pages 736–753. Springer, 2022.

[53] 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.

[54] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8254–8263, 2023.

[55] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *IROS*, 2021.

[56] Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)*, 38(6):1–14, 2019.

[57] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5752–5761, 2021.

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

[59] Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. Differentiable point-based radiance fields for efficient view synthesis. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–12, 2022.

[60] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[61] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. 2018.

COLMAP-Free 3D Gaussian Splatting

Supplementary Material

A. Implementation Details

A.1. Dataset

We select sequences containing dramatic camera motions Tanks and Tamples [19] and CO3D-V2 [34] for training and evaluation. The details of each sequence are listed in Table 8, where *Max rotation* denotes the maximum relative rotation angle between any two frames in a sequence. The sampled images are further split into training and test sets. Starting from the 5*th* image, we sample every 8*th* image in a sequence as a test image. However, this leads to a change in the sampling rate in the temporal domain among training images. In order to study the effect of the sampling rate changes, we follow the experiment setting proposed by [6]. Specifically, for scene *Family* in Tanks and Temples [19], we sample every other image as test images, *i.e*., training on images with odd frame ids and testing on images with even frame ids. For CO3D-V2 [34], we randomly select 10 scenes from 6 categories, *e.g*., apple, bench, hydrant, plant, skateboard, and teddybear. The selected sequence IDs are also shown in Table 8 (bottom part). Compared to Tanks and Temples, most scenes achieve the *Max rotation* of 180° indicating more dramatic and larger camera motions than Tanks and Temples.

	Scenes	Type	Seq. length	Frame rate	Max. rotation (deg)
Tanks and Temples	Church	indoor	400	30	37.3
	Barn	outdoor	150	10	47.5
	Museum	indoor	100	10	76.2
	Family	outdoor	200	30	35.4
	Horse	outdoor	120	20	39.0
	Ballroom	indoor	150	20	30.3
	Francis	outdoor	150	10	47.5
	Ignatius	outdoor	120	20	26.0
CO3D-V2	34_1403_4393	indoor	202	30	180.0
	106_12648_23157	outdoor	202	30	180.0
	110_13051_23361	indoor	202	30	71.6
	219_23121_48537	indoor	202	30	180.0
	245_26182_52130	indoor	202	30	180.0
	247_26441_50907	indoor	202	30	180.0
	407_54965_106262	indoor	202	30	180.0
	415_57112_110099	outdoor	202	30	180.0
	415_57121_110109	outdoor	202	30	180.0
	429_60388_117059	outdoor	202	30	180.0

Table 8. **Details of selected sequences.** We downsample several videos to a lower frame rate. FPS denotes frame per second. *Max rotation* denotes the maximum relative rotation angle between any two frames in a sequence. Our method can handle dramatic camera motion (large maximum rotation angle).

A.2. Training Details.

Local 3DGS. During the training of local 3DGS, we first obtain the monocular depth map of the input image by pre-trained monocular depth estimator, *i.e*., DPT [32], ZeoDepth [5]. Then, the depth map is lifted up with the given camera intrinsic. As the high-resolution input images could lead to a huge amount of point clouds, we downsample the point cloud first before fitting it by 3DGS. Then, the downsampled point cloud is used to initialize the local 3DGS and is further optimized on the input view via photometric loss for 500 iterations. To obtain the transformation of the 3D Gaussian between two views, we freeze the pre-trained local 3DGS including all attributes (*i.e*., position, SH coefficient, opacity, scale, and rotation), and learn the pose parameter of a quaternion vector a translation vector by the photometric loss between the target view and the rendering image. In detail, the freeze local 3D Gaussian is first transformed into the target view coordinate by the learnable pose parameter and then rendered into the target view by the gaussian splatting. The optimization of the camera pose learning process takes 300 steps. The optimization algorithm of local 3DGS is summarized in Algorithm 1

Algorithm 1 Local 3DGS Optimization

```
$\{I_t, I_{t+1}\} \leftarrow$ Two nearby images
DPT $\leftarrow$ Monocular Depth Estimation Model
$D_t \leftarrow$ DPT($I_t$)
$G_t \leftarrow$ InitGauss($I_t$, $D_t$)                ▷ Init Local 3DGS
$T_t \leftarrow$ Identity $\mathbb{I}$                    ▷ Init Pose
while not converged do
    $\hat{I}_t \leftarrow$ Rasterize($G_t$)
    $L \leftarrow$ Loss($I_t, \hat{I}_t$)
    $G_t \leftarrow$ Adam($\nabla L$)                    ▷ Update Local 3DGS
end while
while not converged do
    $\hat{I}_{t+1} \leftarrow$ Rasterize($T_t \odot G_t$)
    $L \leftarrow$ Loss($I_{t+1}, \hat{I}_{t+1}$)
    $T_t{}^* \leftarrow$ Adam($\nabla L$)                ▷ Update Pose
end while
$T_t \leftarrow \prod_{i=1}^{t} T_i$                     ▷ Output Pose
```

Global 3DGS. The optimization process of the global 3DGS starts and initializes from the first frame and its monocular depth estimation. Subsequently, camera poses are estimated in a sequential manner using the local 3DGS, as described in Algorithm 1. Concurrently, the global 3DGS is updated with all the observed images to date (*i.e*., from the first to the current image), in tandem with the camera pose estimation. As each new frame is introduced, the global 3DGS progressively grows and expands through a densification process.

scenes	Nope-NeRF			Ours		
	RPE$_t$	RPE$_r$	ATE	RPE$_t$	RPE$_r$	ATE
189_20393_38136	0.444	2.84	0.034	0.064	0.225	0.007
247_26441_50907	0.34	1.395	0.032	0.395	0.477	0.007
407_54965_106262	0.553	4.685	0.057	0.31	0.243	0.008
429_60388_117059	0.398	2.914	0.055	0.134	0.542	0.018
46_2587_7531	0.426	4.226	0.023	0.095	0.447	0.009
mean	0.432	3.212	0.040	**0.200**	**0.387**	**0.010**

Table 9. **Camera Pose Estimation on CO3D V2.** The best results are highlighted in bold.

scenes	Nope-NeRF			Ours		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
189_20393_38136	29.37	0.85	0.54	32.41	0.92	0.26
247_26441_50907	23.49	0.73	0.54	23.88	0.75	0.36
407_54965_106262	25.53	0.83	0.58	27.80	0.84	0.35
429_60388_117059	22.19	0.62	0.56	24.44	0.68	0.36
46_2587_7531	25.3	0.73	0.46	25.44	0.80	0.21
mean	25.18	0.75	0.54	**26.79**	**0.80**	**0.31**

Table 10. **Novel view synthesis results on CO3D V2.** The best results are highlighted in bold.

A.3. Evaluation Metrics

Novel View Synthesis. We use Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) [48], and Learned Perceptual Image Patch Similarity (LPIPS) [60] to measure the novel view synthesis quality. For LPIPS, we use a VGG architecture [40].

Pose Accuracy. To evaluate pose accuracy, we employ standard visual odometry metrics, including Absolute Trajectory Error (ATE) and Relative Pose Error (RPE). ATE quantifies the discrepancy between estimated camera positions and their ground truth counterparts. RPE, on the other hand, assesses the errors in relative poses between image pairs. This includes both relative rotation error (RPE$_r$) and relative translation error (RPE$_t$).

B. Additional Experiments

The subsequent sections present further quantitative and qualitative results of novel view synthesis and camera pose estimation, conducted on both the Tanks and Temples and CO3D-V2 datasets.

B.1. Camera Pose Estimation

Additional results on CO3D-V2. We conduct experiments on 5 additional scenes of the CO3D-V2 dataset for the task of camera pose estimation. The results are reported in Table 9. We show better performances than Nope-NeRF [6] in both pose accuracy and synthesis quality.

Additional Visualization. Additional qualitative results for camera pose estimation on CO3D-V2 are presented in Fig. 5, following the evaluation procedure outlined in the main paper. In scenarios involving large camera motions, our approach significantly outperforms Nope-NeRF.

B.2. Novel View Synthesis.

Render Novel Views. As mentioned in the main paper, we minimize the photometric error of the synthesized images while freezing the 3DGS model to obtain the testing camera poses. Because the test views are sampled from videos that are close to the training views, these good results may be obtained due to overfitting to the training images. Therefore, we conduct an additional qualitative evaluation on more novel views. Specifically, we fit a bezier curve from the estimated training poses and sample interpolated poses for each method to render novel view videos. The rendered images are shown in Fig. 6 and Fig. 7. Compared to Nope-NeRF [6], our approach renders photo-realistic images with more details (please check the highlighted regions).

Unknown camera intrinsic. We also conduct experiments with heuristic camera intrinsic, where we set the FoV of all scenes to 79° and make the principle points to the image center. The quantitative results are listed in the following table. We find that by setting the camera intrinsic heuristically, the performance on novel view synthesis (NVS) and camera pose estimation slightly degenerates which is reasonable as the intrinsic parameters are also important and could be further optimized along with the camera extrinsic parameters.

Method	PSNR	SSIM	LPIPS	RPE$_t$	RPE$_r$	ATE
Heuristic Intrinsic	30.90	0.92	0.09	0.044	0.072	0.004
G.T. Intrinsic	**31.28**	**0.93**	**0.09**	**0.041**	**0.069**	**0.004**

Table 11. Ablation study of camera intrinsic on Tanks and Temples.

Different monocular depth estimator. We conduct ablation studies on different monocular depth estimation algorithms in the following table. We notice that more accurate monocular depth estimation results could always lead to better performance.

scenes	ZeoDepth				DepthAnything			
	PSNR	SSIM	RPE$_t$	RPE$_r$	PSNR	SSIM	RPE$_t$	RPE$_r$
Church	30.49	0.93	0.012	0.033	30.66	0.93	0.012	0.029
Barn	28.34	0.86	0.039	0.057	30.54	0.88	0.034	0.113
Museum	30.40	0.91	0.052	0.158	30.92	0.92	0.043	0.130
Family	28.79	0.91	0.093	0.037	32.54	0.95	0.037	0.069
Horse	33.32	0.95	0.101	0.035	33.96	0.96	0.108	0.075
Ballroom	32.86	0.96	0.021	0.032	32.54	0.96	0.022	0.030
Francis	31.05	0.89	0.057	0.086	32.73	0.91	0.027	0.126
Ignatius	22.75	0.75	0.172	0.083	28.89	0.89	0.043	0.075
mean	29.75	0.90	0.068	0.065	**31.60**	**0.93**	**0.041**	0.081

Table 12. Ablation study of different depth estimators on Tanks and Temples.

Additional results on CO3D-V2. We conduct experiments on 5 additional scenes of the CO3D-V2 dataset and the novel view synthesis results are summarized in Table 10.

Additional Visualization. We present additional qualitative results for novel view synthesis on Tanks and Temples and

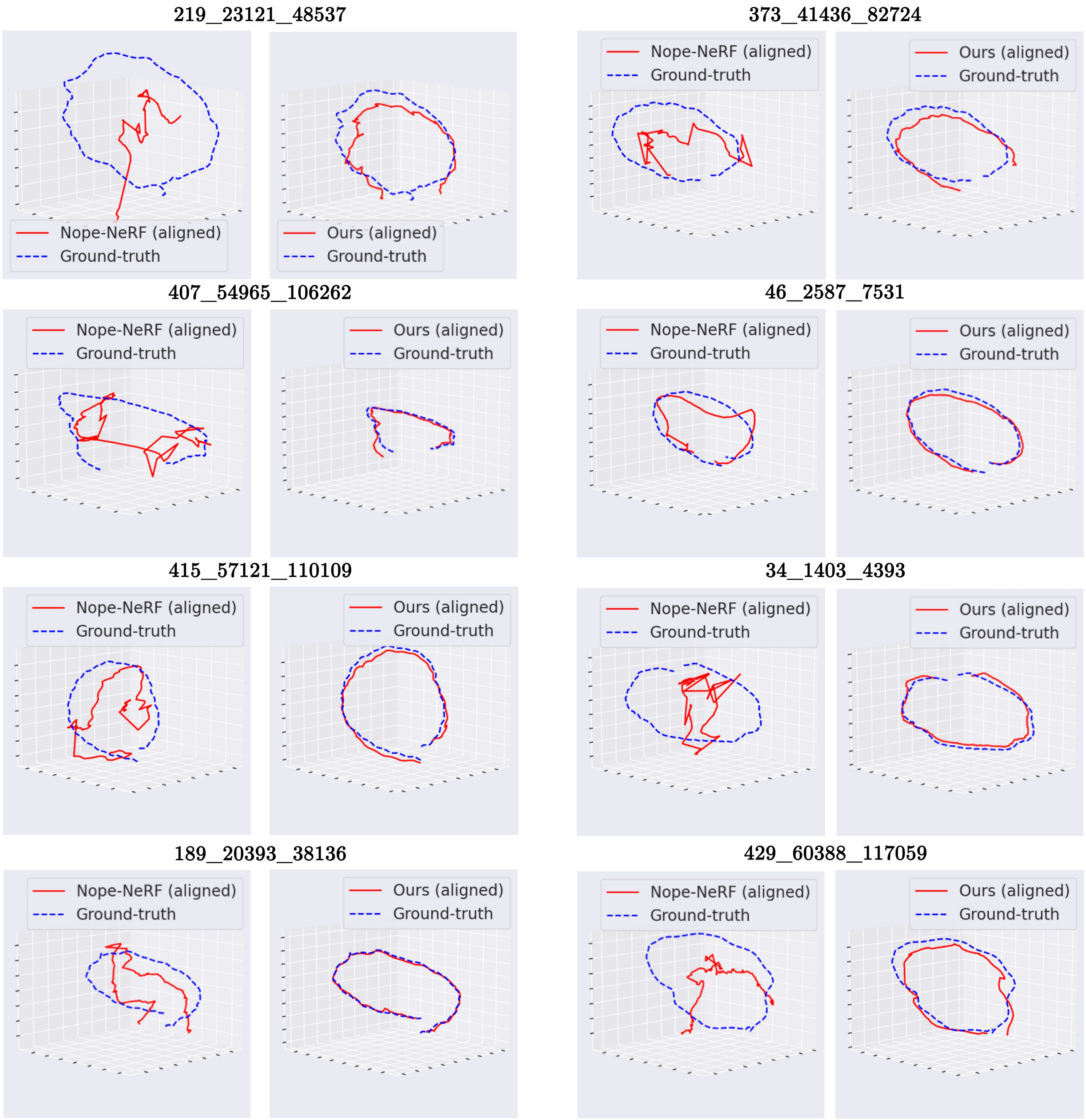

Figure 5. **Qualitative comparison for Camera Pose Estimation on CO3D-V2.** The ground-truth trajectory and the estimated one are shown in blue and red, respectively.

CO3D-V2 in Fig. 8 and Fig. 9 following the same evaluation procedure described in the main paper.

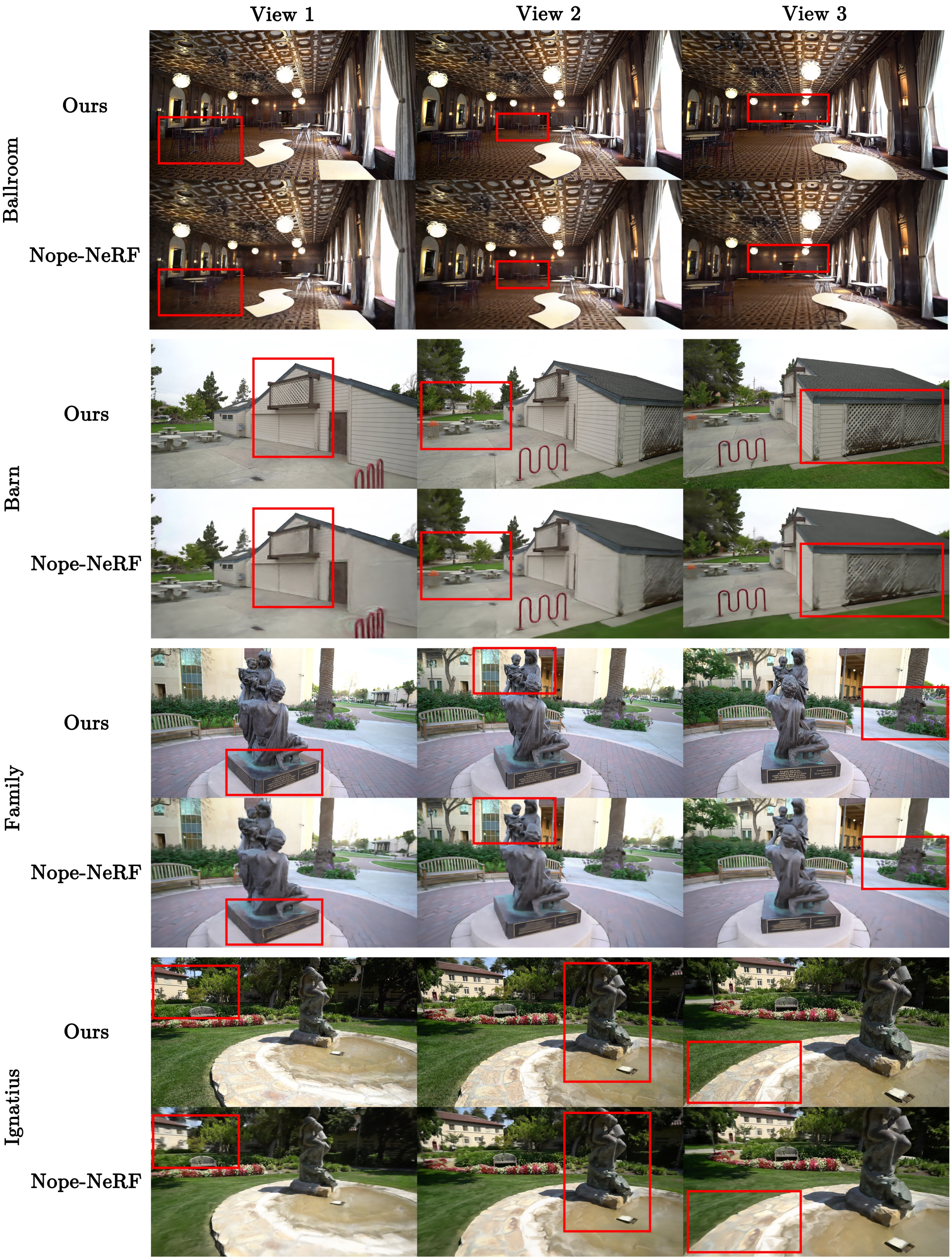

Figure 6. **Qualitative comparison for novel view synthesis on Tanks and Temples.** For each method, we fit the learned trajectory with a bezier curve and uniformly sample new viewpoints for rendering. **Better viewed when zoomed in.**

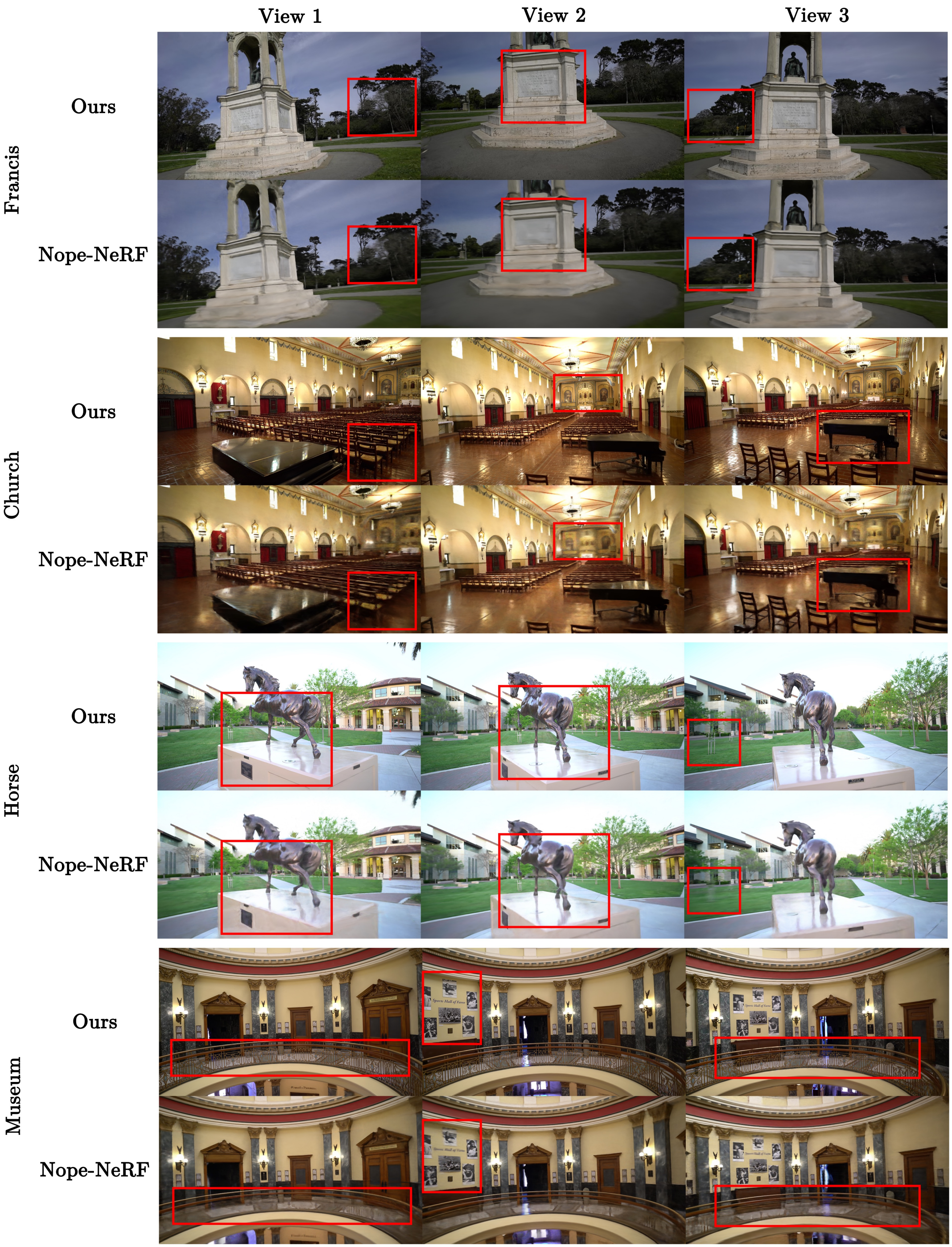

Figure 7. **Qualitative comparison for novel view synthesis on Tanks and Temples.** For each method, we fit the learned trajectory with a bezier curve and uniformly sample new viewpoints for rendering. **Better viewed when zoomed in.**

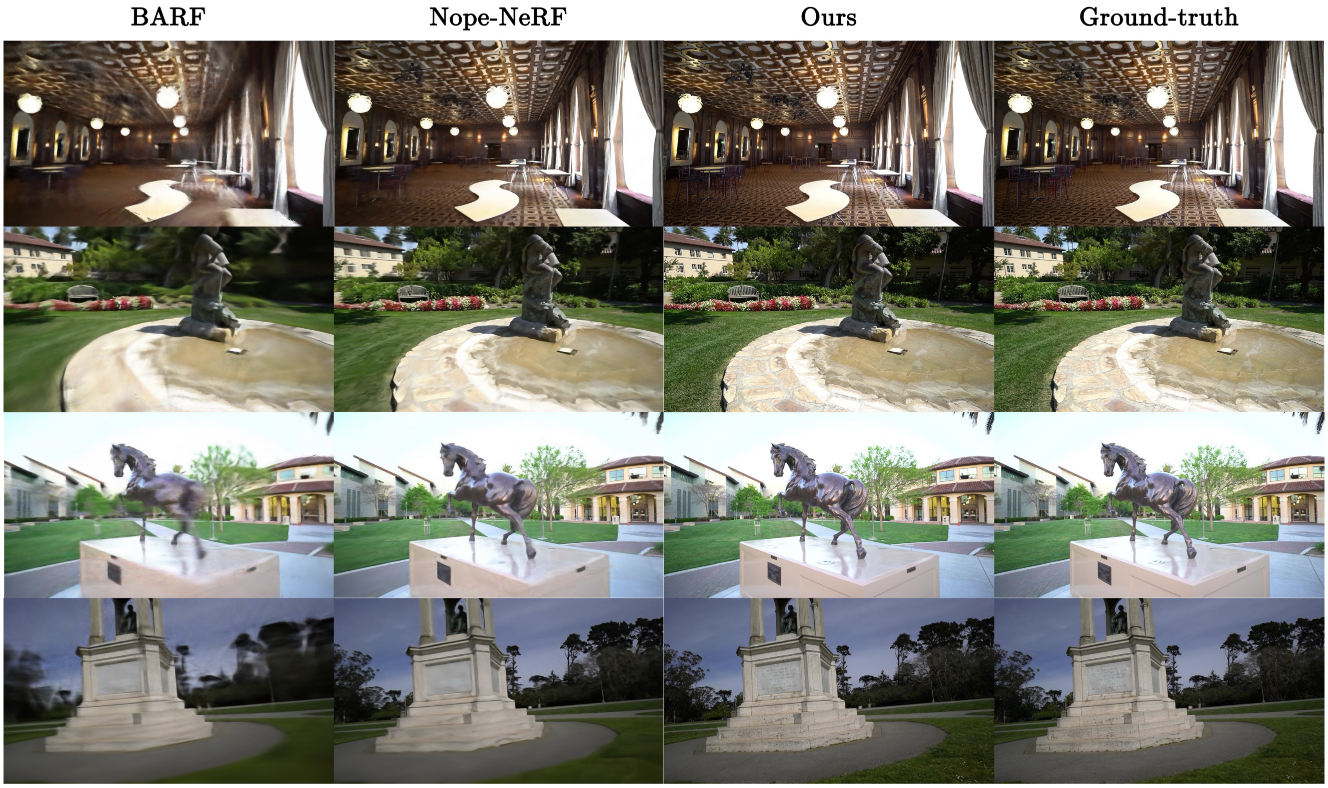

Figure 8. **Qualitative comparison for novel view synthesis on Tanks and Temples.** Our approach produces more realistic rendering results than other baselines. **Better viewed when zoomed in.**

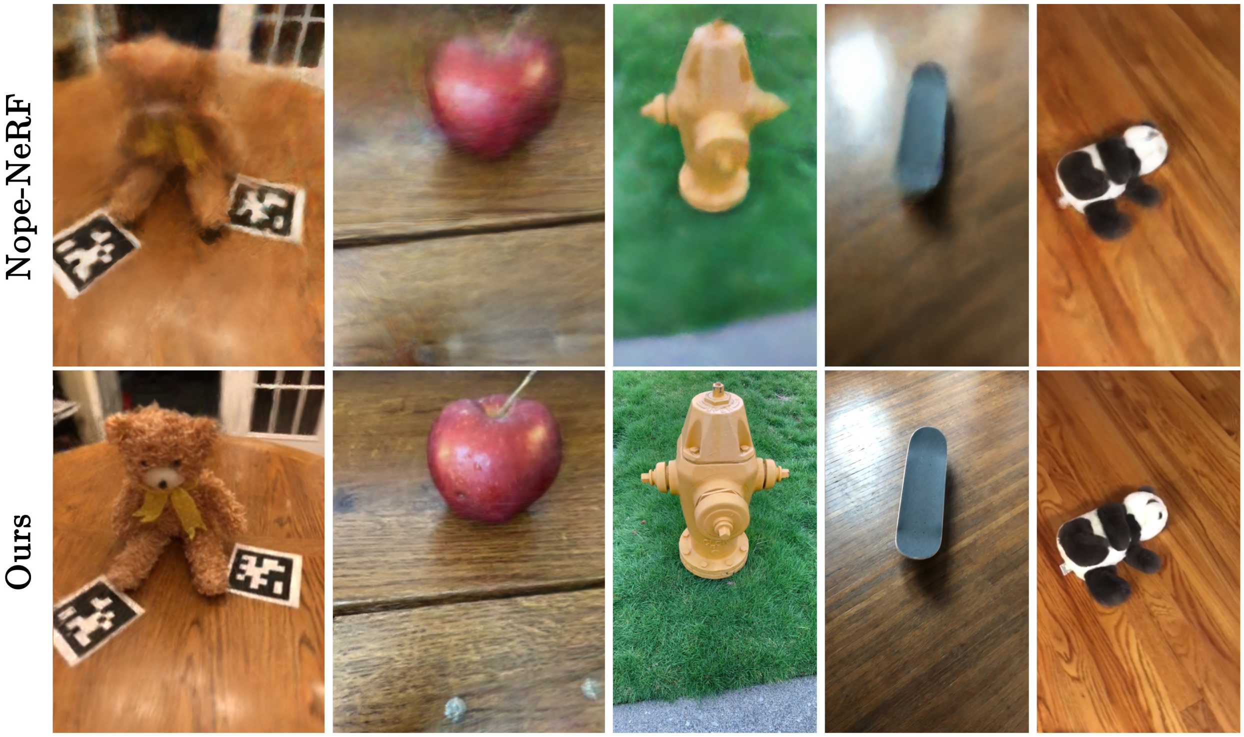

Figure 9. **Qualitative comparison for novel view synthesis on CO3D-V2.**Our approach produces more realistic rendering results than other baselines. Better viewed when zoomed in.