TrackGS: Optimizing COLMAP-Free 3D Gaussian Splatting with Global Track Constraints

Dongbo Shi[1], Shen Cao[2], Lubin Fan[2], Bojian Wu[2], Jinhui Guo[2], Renjie Chen[1], Ligang Liu[1], Jieping Ye[2]
[1]University of Science and Technology of China, [2]Independent Researcher

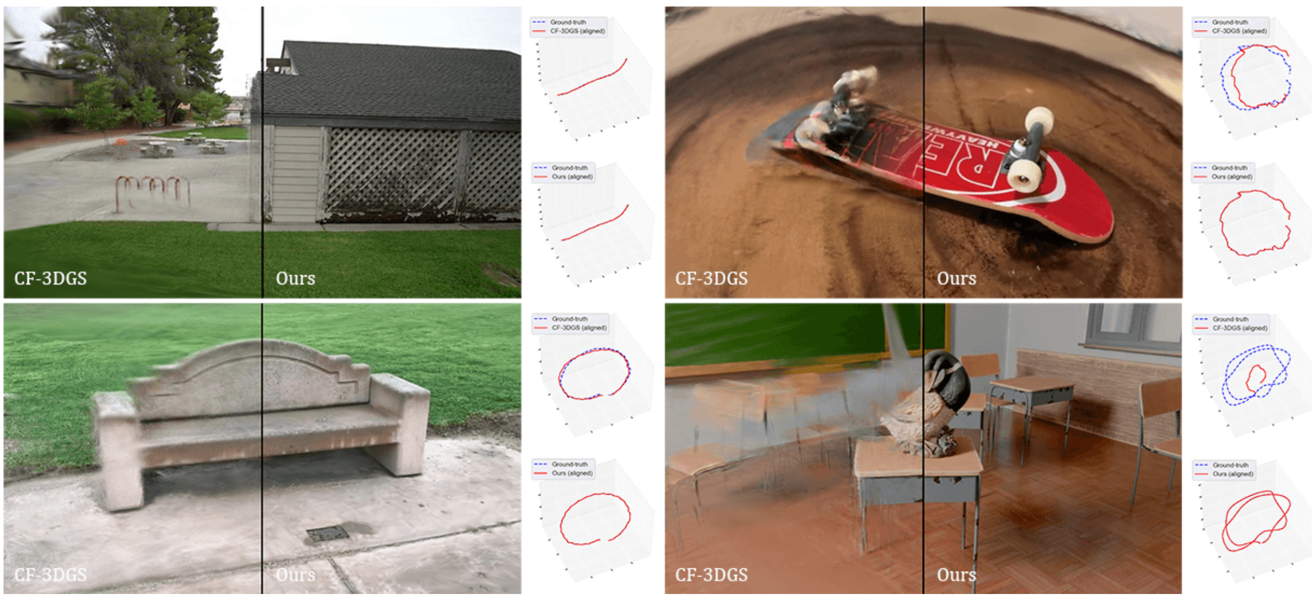

Figure 1. **Comparisons on novel view synthesis and camera poses.** We propose a novel model of 3D Gaussian Splatting without any known camera parameters by leveraging global track information. Compared with state-of-the-art methods, we provide not only higher rendering quality in novel view synthesis, but also more accurate estimation of camera poses on benchmark datasets, including the *challenging* real-world indoor and outdoor or synthetic scenes with *complicated* camera movements (the right column).

Abstract

While 3D Gaussian Splatting (3DGS) has advanced ability on novel view synthesis, it still depends on accurate precomputaed camera parameters, which are hard to obtain and prone to noise. Previous COLMAP-Free methods optimize camera poses using local constraints, but they often struggle in complex scenarios. To address this, we introduce TrackGS, which incorporates feature tracks to globally constrain multi-view geometry. We select the Gaussians associated with each track, which will be trained and rescaled to an infinitesimally small size to guarantee the spatial accuracy. We also propose minimizing both reprojection and backprojection errors for better geometric consistency. Moreover, by deriving the gradient of intrinsics, we unify camera parameter estimation with 3DGS training into a joint optimization framework, achieving SOTA performance on challenging datasets with severe camera movements. Code and datasets will be released.

1. Introduction

Given a collection of images from a 3D scene along with the corresponding camera intrinsic and extrinsic parameters, 3D Gaussian Splatting (3DGS) [11] can effectively represent the scene with a series of 3D Gaussians, and generate high-quality images from novel viewpoints. Due to its efficiency in training and superior performance in testing, 3DGS has become popular for a variety of applications including reconstruction, editing, and AR/VR etc. However, the effectiveness of 3DGS training relies on accurately predetermined camera poses (i.e., camera extrinsics) and camera focal lengths (i.e., camera intrinsics). These parameters are typically derived using COLMAP [24] in advance. This preprocessing step is not only time-consuming but also impacts the training performance of 3DGS, particularly when dealing with complex camera movements and scenes.

Recent COLMAP-Free approaches [1, 5, 6, 8, 9, 16, 32] have tried to address this by adding local constraints. However, it limits these methods to handling only simpler scenes. They typically assume that the input data are sequential and that the focal length is known. When faced with more complex scenarios with very complicated camera movements, these methods generally perform poorly.

To address this problem, we introduce global track information to globally constrain geometric consistency and innovatively integrate it into 3DGS. Specifically, we select the 3D Gaussians that correspond to each track, where the track Gaussians are dynamically adjusted in size during the training process, shrinking to an infinitesimally small size, and approaching spatial points that are distributed near the surface. Using our novel designed 2D and 3D track losses, the reprojection and backprojection errors are explicitly minimized. Moreover, the remaining 3D Gaussians continue to function as before, and all 3D Gaussians are aggregated and constrained by the loss function. Additionally, for the first time, we theoretically derive the focal length gradients to achieve full differentiability of the pipeline, eliminating the pre-computation of all the necessary camera parameters, including both the intrinsics and extrinsics. It enables to unify the learning process for both camera parameters and 3DGS.

In summary, our contributions are as follows:

- We are the first to propose integrating track information with 3DGS, using global geometric constraints to simultaneously optimize camera parameters and 3DGS. To achieve this, we introduce 2D and 3D track losses to constrain reprojection and backprojection errors.
- We propose a joint optimization framework, where for the first time, we derive the gradient of the camera intrinsics. Without relying on any known camera parameters, we achieve full differentiability for the entire pipeline, seamlessly integrating camera parameters estimation, including both intrinsics and extrinsics, with 3DGS training.
- On both *challenging* public and synthetic datasets, our approach outperforms previous methods on both camera parameters estimation and novel view synthesis.

2. Related Work

2.1. Novel View Synthesis

Novel view synthesis is a foundational task in the computer vision and graphics, which aims to generate unseen views of a scene from a given set of images. Numerous methods have been developed to address this problem by approaching it as 3D geometry-based rendering, such as using meshes [7, 21, 22], MPI [13, 25, 35], point clouds [30, 33], etc.

Recently, Neural Radiance Fields (NeRF) [17] provide a novel solution to this problem by representing scenes as implicit radiance fields using neural networks, achieving photo-realistic rendering quality. Although having some works in improving efficiency [15, 18], the time-consuming training and rendering still limit its practicality. Alternatively, 3D Gaussian Splatting (3DGS) [11] models the scene as explicit Gaussian kernels, with differentiable splatting for rendering. Its improved real-time rendering performance, lower storage and efficiency, quickly attract more attentions.

2.2. Optimizing Camera Poses in NeRFs and 3DGS

Although NeRF and 3DGS can provide impressive scene representation, these methods all need accurate camera parameters (both intrinsic and extrinsic) as additional inputs, which are mostly obtained by COLMAP [24]. When the prior is inaccurate or unknown, accurately estimating camera parameters and scene representations becomes crucial.

In earlier studies, scene training and camera pose estimation relied solely on photometric constraints. iNeRF [32] refines the camera poses using a pre-trained NeRF model. NeRFmm [29] introduces a joint optimization approach that simultaneously estimates camera poses and trains the NeRF model. BARF [14] and GARF [2] propose a new positional encoding strategy to address the gradient inconsistency issues in positional embedding, achieving promising results. However, these methods only yield satisfactory optimization when the initial pose is very close to the ground truth, as photometric constraints alone can only enhance camera estimation quality within a limited range. Subsequently, SC-NeRF [8] minimizes a projected ray distance loss based on correspondence between adjacent frames. NoPe-NeRF [1] utilizes monocular depth maps as geometric priors and defines undistorted depth loss and relative pose constraints.

Regarding 3D Gaussian Splatting, CF-3DGS [6] utilizes mono-depth information to refine the optimization of local 3DGS for relative pose estimation and subsequently learns a global 3DGS in a sequential manner. InstantSplat [5] targets sparse view scenes, initially employing DUSt3R [27] to create a densely covered, pixel-aligned point set for initializing 3D Gaussian models, and then implements a parallel grid partitioning strategy to accelerate joint optimization. Jiang *et al.* [10] develops an incremental method for reconstructing camera poses and scenes, but it struggles with complex scenes and unordered images. HT-3DGS [9] interpolates frames for training and splits the scene into local clips, using a hierarchical strategy to build 3DGS model. It works well for simple scenes, but fails with dramatic motions due to unstable interpolation and low efficiency.

However, most existing methods generally depend on sequentially ordered image inputs and incrementally optimize camera parameters and 3DGS, which often leads to drift errors and hinders achieving globally consistent results. Our work seeks to overcome these limitations.

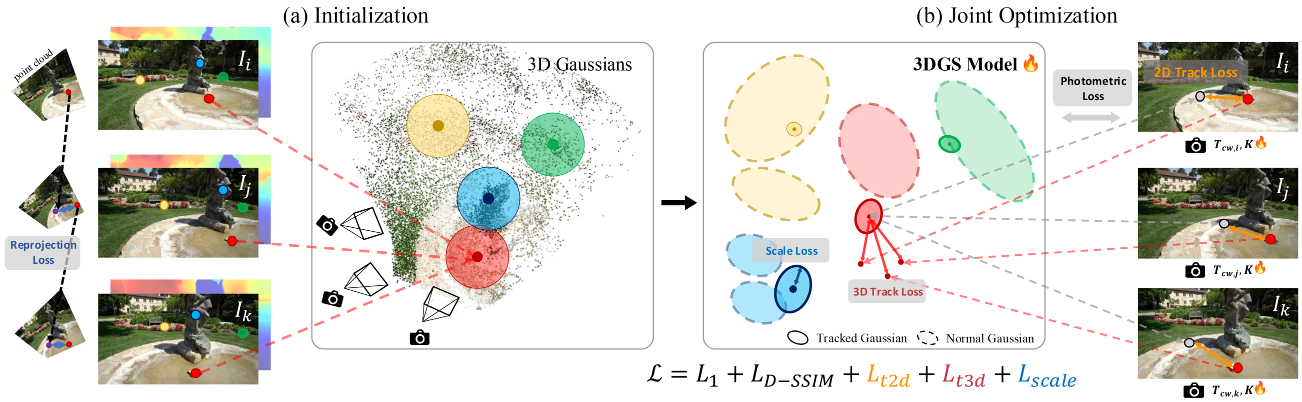

Figure 2. **Overview.** Given a set of images, our method obtains both camera intrinsics and extrinsics, as well as a 3DGS model. During the initialization, we extract the global tracks, and initialize camera parameters and Gaussians from image correspondences and monodepth with reprojection loss. We determine Gaussian kernels with recovered 3D track points, and then jointly optimize the parameters K, T_{cw}, 3DGS through the proposed global track constraints (i.e., L_{t2d}, L_{t3d}, and L_{scale}) and original photometric losses (i.e., L_1 and L_{D-SSIM}).

3. 3D Gaussian Splatting

3DGS models a scene using a set of 3D anisotropic Gaussians. Each Gaussian is parameterized by a centroid $\mu \in \mathbb{R}^3$, a quaternion factor $q \in \mathbb{R}^4$, a scale factor $s \in \mathbb{R}^3$, spherical harmonics (SH) coefficients of color $c \in \mathbb{R}^k$, and opacity $\alpha \in \mathbb{R}$. Donating the rotation matrix of quaternion q and scale matrix of s by $R \in \mathbb{R}^{3\times3}$ and $S = \mathrm{diag}(s)$, the covariance matrix Σ and Gaussian function $G(x)$ are:

$$\Sigma = RSS^\top R^\top, G(x) = \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right). \tag{1}$$

Denoting projection matrix $T_{cw} = [R_{cw}|t_{cw}]$, which transforms points from the *world* to *camera* coordinate space, an image rendered from the specified view can be obtained as follows. First, the covariance matrix in camera coordinates Σ^{2D} is obtained by approximating the projection of 3D Gaussian in pixel coordinates, and can be expressed as:

$$\Sigma^{2D} = JR_{cw}\Sigma R_{cw}^\top J^\top, \tag{2}$$

where J is the Jacobian of the affine approximation of the projective transformation. The final rendered color $\hat{C}$ can be denoted as the alpha-blending of N ordered Gaussians:

$$\hat{C} = \sum_i^N c_i\alpha_i \prod_j^{i-1}(1-\alpha_j), \tag{3}$$

where c_i and α_i are the color and opacity of the Gaussians. Similarly, the depth of the scene preceived of a pixel is,

$$\hat{D} = \sum_i^N d_i\alpha_i \prod_j^{i-1}(1-\alpha_j), \tag{4}$$

where d_i denotes the z-axis coordinate for the transformed Gaussian centers in the camera space.

Usually, the parameters of 3D Gaussians are optimized by rendering and comparing the rendered images with the ground-truths. The loss function $\mathcal{L}$ is defined as follows:

$$\mathcal{L} = (1-\lambda)L_1 + \lambda L_{\text{D-SSIM}}. \tag{5}$$

Typically, 3D Gaussians are initialized with Structure from Motion (SfM) point clouds obtained from the input images.

4. Method

Overview. Given a set of images $\mathcal{I} = \{I_i\}_{i=1}^M$, with unknown extrinsic matrix $T_{cw,i}$ at each view and unknown intrinsic matrix denoted by K, our method aims to build a 3D Gaussian Splatting (3DGS) model while simultaneously estimating both the extrinsic and intrinsic matrices, as shown in Fig. 2. To achieve this goal, our key approach is to leverage the global track constraint to explicitly capture and enforce multi-view geometric consistency, which serves as the foundation for accurately estimating both the 3DGS model and the camera parameters. Specifically, during initialization, we construct Maximum Spanning Tree (Sec. 4.1) based on 2D matched feature points and extract global tracks. Then we initialize both the camera parameters and subsequent 3D Gaussians with the estimated 3D track points. Building on this, we propose an effective joint optimization method with three loss terms: 2D track loss, 3D track loss, and scale loss. The 2D and 3D track losses are minimized to ensure multi-view geometric consistency. The scale loss constrains the track Gaussians remain aligned with the scene's surface while preserving the expressive capability of the 3DGS model. We derive and implement the differentiable components of the camera parameters, including both the extrinsic and intrinsic matrices. This allows us to apply the chain rule, enabling seamless joint optimization of the 3DGS model and the camera parameters.

4.1. Initialization

Global Tracks. We begin by extracting 2D feature points $\{p_i\}$ from each image I and computing feature matches across all images using off-the-shelf algorithms [4, 23]. To organize these matches into global tracks, we first construct a Maximum Spanning Tree (MST) using Kruskal's algorithm, where the node represents each image and the weight of each edge is determined by the number of feature matching pairs between two images. By traversing MST and the feature points, we use the Union-Find algorithm to extract global track $\mathcal{P}$ and remove short tracks for robustness.

Camera Parameters. We assume all cameras share a standard pinhole model with no distortion, and the principal point locates at the center of the image, then the intrinsic matrix K of camera is:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{6}$$

where (c_x, c_y) is the principal point and (f_x, f_y) is focal length. Empirically, we initialize the focal length with a field of view (FoV) of 60° as:

$$f_x = f_y = \frac{\sqrt{c_x^2 + c_y^2}}{\tan(\text{FoV}/2)}. \tag{7}$$

For each edge (i, j) in the MST, we leverage the off-the-shelf monocular depth maps of images (i.e., DPT [19]), and convert correspondences p_i and p_j to the point clouds. Then we define a reprojection loss to optimize the associated transformation as follows:

$$L_{reproj}(i, j) = \|K \cdot (T_{ji} \cdot K^{-1} \cdot p_i) - p_j\|, \tag{8}$$

where T_{ji} is the associated transformation from i to j. By minimizing the overall reprojection loss for all pairs, we can roughly obtain the initial camera's intrinsics and extrinsics.

3D Gaussians. We initialize the 3D Gaussians by 3D track points. We extract a set of tracks $\mathcal{P}$, where each element $(P, \{p_i\}_{i=1}^l) \in \mathcal{P}$ represents a 3D track point P and its corresponding matching points $\{p_i\}_{i=1}^l$ associated with the training images. The 3D track point P is initialized as the centroid of the transformed projections of $\{p_i\}$:

$$P = \frac{1}{l} \sum_{i=1}^{l} K^{-1} p_i. \tag{9}$$

Notably, we use track points solely to initialize 3D Gaussians, as their positions will be refined by global optimization and constraints to accurately represent object surfaces.

4.2. Joint Optimization

4.2.1. Global Track Constraints

We leverage the global tracks obtained during initialization to enforce multi-view geometric consistency in joint optimization, both in 2D and 3D space. Our approach is intuitive and focuses on two key aspects. First, the reprojections of the 3D track points onto each image should closely match the original 2D feature points, ensuring that reprojection relationships are preserved throughout the optimization process. Second, the backprojection of the matched 2D feature points with 3DGS rendering depth from each training image should remain near their corresponding 3D track points in the scene, ensuring spatial consistency across all input images. Based on these two considerations, we introduce the **2D track loss** and **3D track loss**, respectively. Since the 3D track points in the scene should lie on the surface of the object, we address this by using the centroids of the initialized 3D Gaussians to represent the 3D track points. To achieve this, we introduce a **scale loss** to constrain the sizes of these Gaussians. This approach allows us to preserve the original function of the 3D Gaussians while treating them as virtual 3D spatial points. These points serve as the key elements for optimizing camera parameters and enhancing the global geometric consistency.

2D Track Loss. We reproject the 3D track points P (Eq. 9) into the corresponding images using the associated camera parameters and compute the reprojection loss, which will be summed to calculate the total 2D track loss:

$$L_{t2d} = \sum_{P \in \mathcal{P}} \frac{1}{l} \sum_{i=1}^{l} \|p_i - K \cdot T_{cw,i} \cdot P\|. \tag{10}$$

3D Track Loss. We backproject the 2D feature points into 3D scene using the rendered depth and camera parameter associated with each point. The backprojection error is then computed with respect to the 3D tracked point P. Then errors are aggregated to calculate the overall 3D track loss:

$$L_{t3d} = \sum_{P \in \mathcal{P}} \frac{1}{l} \sum_{i=1}^{l} \|d(p_i) \cdot T_{cw,i}^{-1} \cdot K^{-1} \cdot p_i - P\|, \tag{11}$$

where $d(p_i)$ denotes the depth perceived from p_i according to Eq. 4. Note, the 2D track loss relates to the track Gaussians that are mainly used for the optimization of camera parameters, whereas the 3D track loss requires the 3DGS rendered depth values during computation. This indirectly ties the optimization of the 3D track loss to the optimization of the 3DGS model and enhances the capability of multi-view geometric consistency. They are fundamentally different.

Scale Loss. In fact, 3D tracking points reside on object surfaces in the scene. By using 3D Gaussian centroids for tracking, it's crucial to regulate their scale and align centroids closely with the actual object surfaces, ensuring accuracy in tracking point positions and minimizing projection errors. To achieve this, we incorporate a scale loss function to constrain the scale S_{track} of these Gaussians G_{track}:

$$L_{scale} = \sum_{S \in S_{track}} \|\max(S)\|. \tag{12}$$

Overall Objectives. Combined with Eq. 5, our joint optimization can be formulated as:

$$\begin{aligned}\mathcal{L} =&\lambda_1 L_1 + \lambda_{\mathrm{D-SSIM}} L_{\mathrm{D-SSIM}} \\ &+ \lambda_{t2d} L_{t2d} + \lambda_{t3d} L_{t3d} + \lambda_{scale} L_{scale}.\end{aligned} \tag{13}$$

4.2.2. Optimizing Camera Parameters

To optimize the camera parameters of 3D Gaussians simultaneously, the gradient of the loss function $\mathcal{L}$ respect to the camera parameters are needed. We derive these gradients accordingly, where the gradient of extrinsic parameters is:

$$\frac{\partial \mathcal{L}}{\partial T_{cw}} = \frac{\partial \mathcal{L}}{\partial t} q^{\top}, \tag{14}$$

where $q = [\mu, 1]^T$ and $t = T_{cw} q = [t_x, t_y, t_z, t_w]^T$. Further, let $(\mu^{'}, \Sigma^{'})$ be the 2D projection of the centroid and covariance (μ, Σ), the gradient of $\mathcal{L}$ respect to focal length $F = (f_x, f_y)$ can be computed, where $T = JR_{cw}$, as:

$$\begin{cases} \dfrac{\partial \mathcal{L}}{\partial f_x} = \dfrac{t_x}{t_z} \dfrac{\partial \mathcal{L}}{\partial \mu_x^{'}} + < \dfrac{\partial \mathcal{L}}{\partial T} R_{cw}^{\top}, \dfrac{\partial J}{\partial f_x} >, \\ \dfrac{\partial \mathcal{L}}{\partial f_y} = \dfrac{t_y}{t_z} \dfrac{\partial \mathcal{L}}{\partial \mu_y^{'}} + < \dfrac{\partial \mathcal{L}}{\partial T} R_{cw}^{\top}, \dfrac{\partial J}{\partial f_y} > . \end{cases} \tag{15}$$

Please refer to the supplementary materials for more details.

5. Experiments

5.1. Experimental Setup

Datasets. We conduct experiments on two real-world datasets, i.e., *Tanks and Temples* [12] and *CO3D-V2* [20], and a *Synthetic Dataset* created by ourselves. **Tanks and Temples**, adapted from NoPe-NeRF [1], is used for novel view synthesis and pose estimation. It features 8 scenes with mild view changes, both indoors and outdoors. **CO3D-V2** includes thousands of videos of various objects. Following the CF-3DGS [6], we select 8 scenes with significant camera movements to demonstrate our robustness. **Synthetic Dataset** comprises 4 scenes with about 150 frames each created using Blender [3], showcasing complex roaming and object-centric camera motions. It's used to assess camera parameter estimation, providing ground truth for intrinsic and extrinsic parameters. For additional details on the synthetic dataset, see the supplementary material.

Note that, the Tanks and Temples dataset is a relatively simple test case for this task, as the camera motion in this scene is mostly linear or involves small movements, resulting in a very limited solution space for pose estimation. In contrast, the CO3D-V2 and our synthetic datasets contain much more complex camera trajectories. Therefore, as follows, we will emphasis more on the results of such challenging scenes, as they pose a higher level of difficulty.

Metrics. We use standard evaluation metrics, including PSNR, SSIM [28], and LPIPS [34] to evaluate the quality of novel view synthesis. For pose estimation, we rely on the Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) [1, 6, 14]. RPE_r and RPE_t are utilized to measure the accuracy of rotation and translation, respectively. To ensure the metrics are comparable on the same scale, we align the camera poses using Umeyama's method [26] for both estimation and evaluation. For camera focal length, we convert it to the field of view (FoV) and calculate the angular error, following [36].

	Scenes	Ours			HT-3DGS			CF-3DGS			NoPe-NeRF		
		PSNR↑	SSIM↑	LPIPS↓	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Tanks and Temples	Church	29.39	0.92	0.10	31.34	0.94	0.08	30.23	0.93	0.11	25.17	0.73	0.39
	Barn	31.98	0.94	0.08	34.95	0.97	0.05	31.23	0.90	0.10	26.35	0.69	0.44
	Museum	31.92	0.94	0.08	31.59	0.95	0.08	29.91	0.91	0.11	26.77	0.76	0.35
	Family	32.22	0.95	0.09	34.17	0.97	0.05	31.27	0.94	0.07	26.01	0.74	0.41
	Horse	30.33	0.94	0.09	35.82	0.98	0.03	33.94	0.96	0.05	27.64	0.84	0.26
	Ballroom	35.03	0.97	0.03	34.12	0.97	0.04	32.47	0.96	0.07	25.33	0.72	0.38
	Francis	33.39	0.92	0.15	34.09	0.93	0.13	32.72	0.91	0.14	29.48	0.80	0.38
	Ignatius	29.25	0.90	0.11	31.64	0.95	0.06	28.43	0.90	0.09	23.96	0.61	0.47
	mean	31.68	0.94	0.09	33.53	0.96	0.07	31.28	0.93	0.09	26.34	0.74	0.39
CO3D-V2	34_1403_4393	28.68	0.88	0.21	32.52	0.93	0.14	27.75	0.86	0.20	28.62	0.80	0.35
	46_2587_7531	31.83	0.92	0.12	30.65	0.91	0.13	25.44	0.80	0.21	25.30	0.73	0.46
	106_12648_23157	26.18	0.83	0.19	23.43	0.73	0.28	22.14	0.64	0.34	20.41	0.46	0.58
	110_13051_23361	33.44	0.94	0.11	29.95	0.87	0.19	29.69	0.89	0.29	26.86	0.73	0.47
	245_26182_52130	33.82	0.93	0.20	28.59	0.87	0.27	27.24	0.85	0.30	25.05	0.80	0.49
	407_54965_106262	28.73	0.86	0.39	-	-	-	27.80	0.84	0.35	25.53	0.83	0.58
	415_57112_110099	30.37	0.88	0.22	27.23	0.78	0.30	22.14	0.64	0.34	20.41	0.46	0.58
	429_60388_117059	25.70	0.70	0.35	-	-	-	24.44	0.68	0.36	22.19	0.62	0.56
	mean	29.84	0.87	0.22	28.73	0.85	0.22	25.83	0.78	0.30	24.30	0.68	0.51

Table 1. **Novel view synthesis results on Tanks and Temples and 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 gray background.

Implementation Details. Our implementation is primarily based on *gsplat* [31], an accelerated 3DGS library. We implement joint optimization by backpropagating the gradient of camera parameters through modifications to the CUDA operator in the library. All parameters are optimized using Adam optimizer. For initialization, we optimize the relative pose between frames, and focal length. Then, the parameters of 3DGS, the absolute poses of cameras, and focal length are optimized. The camera pose is represented as a combination of an axis-angle representation $\mathrm{q} \in \mathfrak{so}(3)$ and a translation vector $\mathrm{t} \in \mathbb{R}^3$. All experiments are conducted on a single RTX 4090 GPU with 24GB VRAM.

During the training phase, we will clone new Gaussians from those associated with the track points and apply the same training strategy as the original 3DGS (including clone, split, and delete). Note that the track Gaussians still need to be preserved and constrained by a scale loss. We use Eq. 13 for training and set $\lambda_1 = 0.8$, $\lambda_{\mathrm{D-SSIM}} = 0.2$, $\lambda_{t2d} = 0.01$, $\lambda_{t3d} = 0.01$, $\lambda_{scale} = 0.01$.

5.2. Experimental Results and Analysis

Novel View Synthesis. Unlike the standard settings where the camera poses of test views are given, we need to first obtain the camera poses of the test views for rendering. Following the same protocol as CF-3DGS [6], we obtain the camera poses of the test views by minimizing the photometric error between the synthesized images and the test views using the pre-trained 3DGS model. We apply the same procedure to all baseline methods to maintain a consistent bias for a fair comparison.

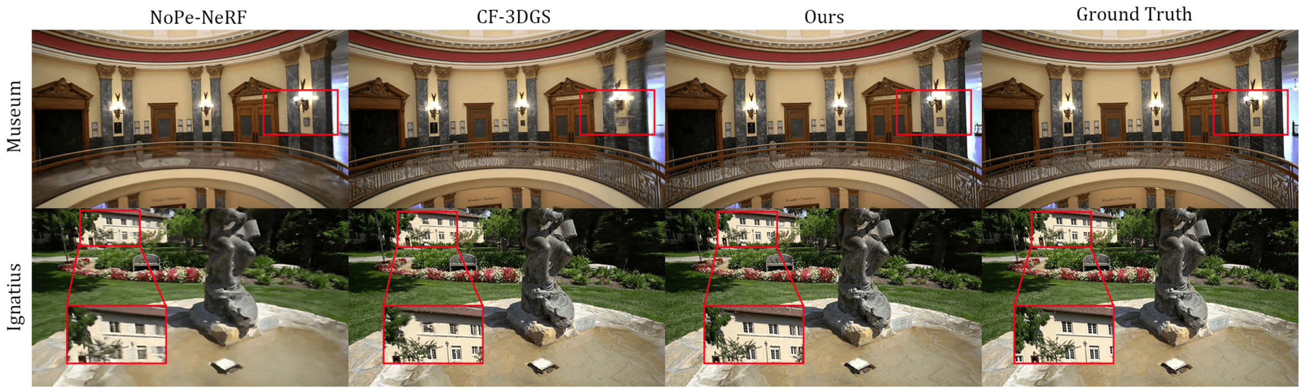

Figure 3. **Qualitative comparison for novel view synthesis on Tanks and Temples.** We achieve better rendering results on details.

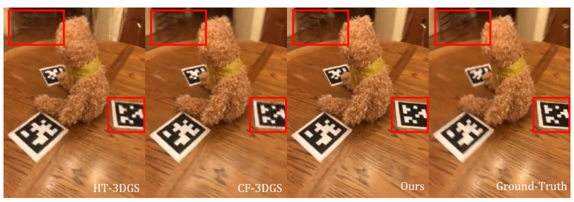

Figure 4. **Qualitative comparison for novel view synthesis on CO3D-V2.** We achieve the best rendering results on details.

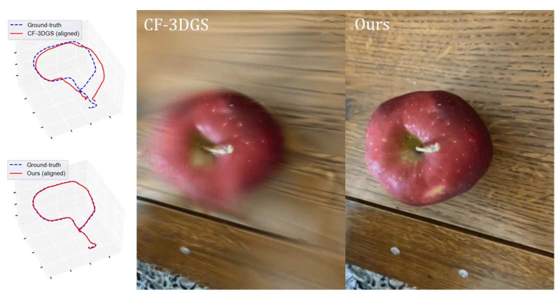

Figure 5. **Qualitative comparison for novel view synthesis and camera pose estimation on CO3D-V2.** Benefit from the accuracy of the camera pose estimation, the rendering quality of novel view synthesis obtained by our method is higher than CF-3DGS.

We report the results on Tanks and Temples and CO3D-V2 in Tab. 1. Our method consistently outperforms NoPe-NeRF across all metrics and slightly outperforms CF-3DGS in overall performance on Tanks and Temples. On CO3D-V2, our results are better than those of all baselines. Interestingly, compared to HT-3DGS [9], our method achieves significantly higher PSNR on the CO3D-V2, which involves large camera motion, but performs worse on the Tanks and Temples, which features smooth motion trajectories. We attribute this to HT-3DGS's reliance on video frame interpolation, which can be unstable under large camera motion. As shown in Fig. 3, the images synthesized by our method are clearer than those obtained by other

	Scenes	Ours			HT-3DGS			CF-3DGS			NoPe-NeRF		
		$\text{RPE}_t \downarrow$	$\text{RPE}_r \downarrow$	ATE↓	RPE_t	RPE_r	ATE	RPE_t	RPE_r	ATE	RPE_t	RPE_r	ATE
CO3D-V2	34_1403_4393	0.099	0.605	0.009	0.041	0.170	0.009	0.505	0.211	0.009	0.591	1.313	0.053
	46_2587_7531	0.013	0.080	0.001	0.025	0.275	0.004	0.095	0.447	0.009	0.426	4.226	0.023
	106_12648_23157	0.009	0.076	0.001	0.045	0.282	0.014	0.094	0.360	0.008	0.387	1.312	0.049
	110_13051_23361	0.012	0.052	0.001	0.093	0.331	0.020	0.140	0.401	0.021	0.400	1.966	0.046
	245_26182_52130	0.005	0.029	0.001	0.064	0.438	0.017	0.239	0.472	0.017	0.587	1.867	0.038
	407_54965_106262	0.062	0.461	0.011	-	-	-	0.310	0.243	0.008	0.553	4.685	0.057
	415_57112_110099	0.004	0.024	0.001	0.049	0.351	0.024	0.110	0.424	0.014	0.326	1.919	0.054
	429_60388_117059	0.052	0.454	0.009	-	-	-	0.134	0.542	0.018	0.398	2.914	0.055
	mean	0.032	0.222	0.004	0.053	0.308	0.017	0.203	0.388	0.013	0.459	2.525	0.047

Table 2. **Quantitative comparison of pose accuracy on CO3D-V2.** Note that, we use COLMAP poses 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 gray background.

Scenes	Ours			CF-3DGS			COLMAP+3DGS		
	PSNR↑	SSIM↑	LPIPS↓	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
classroom	36.26	0.94	0.15	19.69	0.69	0.46	35.81	0.94	0.15
lego_c2	29.36	0.90	0.12	15.93	0.31	0.55	28.77	0.88	0.15
livingroom	33.52	0.88	0.24	16.63	0.57	0.57	32.74	0.87	0.27
bedroom	31.17	0.93	0.13	16.98	0.65	0.45	31.73	0.94	0.13

Table 3. **Novel view synthesis results on our Synthetic dataset.** We show the quantitative results using our method, CF-3DGS and COLMAP+3DGS. The best results are gray background.

methods, as evidenced by higher scores in terms of SSIM and LPIPS, as detailed in Tab. 1. As illustrated in Fig. 4 and 5, the advantages of our algorithm are well demonstrated, especially with large camera motions. Due to global joint optimization, multi-view geometric consistency is better maintained in the trained 3DGS model, leading to high-quality images.

For further comparison, we evaluated our method on the Synthetic Dataset, which features extremely complex camera motions. One result is shown in the bottom-right of Fig. 1. In this case, the camera not only moves in multiple circles around the object but also changes significantly in the vertical direction. Our synthesized image from the novel view remains clear and sharp, whereas the CF-3DGS result is blurry with obvious artifacts.

Camera Parameter Estimation. First, we compare the camera pose estimation with baseline methods. In the comparison, our method only assumes that the camera focal length is fixed, while others additionally input the camera

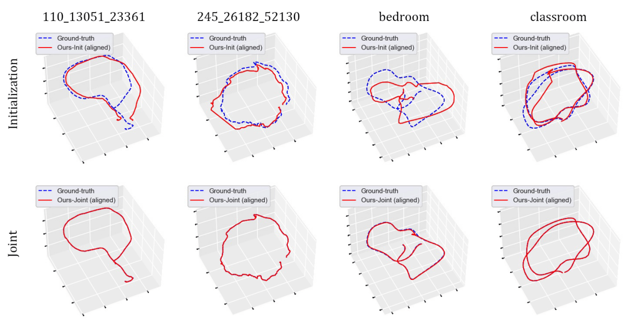

Figure 6. **The trajectory of initial stage and joint stage**. Our joint stage significantly improved the accuracy of camera pose.

focal length. The estimated camera poses are analyzed by Procrusts as in CF-3DGS and compared with the ground-truth of training views. The quantitative results of camera pose estimation on CO3D-V2 datasets are summarized in Tab. 2[1]. The results show that our estimated camera parameters achieve the smallest error among all methods, with the Absolute Trajectory Error (ATE) being only one-fourth of that of the second-best method. This demonstrates that our algorithm excels in scenes with complex camera motions. Compared to the baselines, the global tracking information we use eliminates accumulated errors, leading to more accurate camera pose estimation. Additionally, joint optimization enhances the stability of the estimation results.

Next, we evaluate camera parameter estimation on our Synthetic Dataset. Tab. 4 shows the estimation errors from CF-3DGS, COLMAP, and ours. Note that CF-3DGS uses the camera FoV estimated by COLMAP. We find that our estimated camera FoVs and poses are comparable to those of COLMAP, and the camera pose error is 10 times smaller than that of CF-3DGS. Fig. 6 visualizes our estimated poses in different stages. Thanks to the joint optimization based on global track and the back-propagation of the gradient of the camera parameters, our approach is able to combine these two tasks, thereby reducing the input requirements.

Algorithm Efficiency. All experiments were conducted on a single RTX 4090 GPU. On average across all scenes, the processing times for CF-3DGS and NoPe-NeRF are approximately 1.5 hours and 4 hours, respectively. HT-3DGS reports a runtime of around 4 hours on a professional-grade GPU. In contrast, our approach takes less than 1 hour and uses less than 12GB of GPU memory on average, as we optimize all parameters, including camera parameters.

5.3. Ablation Study

Effectiveness of Different Losses. We ablate each loss of the algorithm on synthetic dataset, since it has ground-truth camera parameters. Tab. 5 reports the average synthesis quality and camera parameter errors across different algorithm variants (see supplementary material for details). In order to better show the role of each loss, we construct variants of the three schemes and remove them one by one. First, we can find that any variant of our algorithm (Variant 3, 4, 5, 6) is better than the CF-3DGS method (Variant 3) in synthesis quality and absolute camera position. Second, from Variant 3, it shows that 2D track loss plays a crucial role in the entire joint optimization. When 2D track loss is not used, compared with the final method (Variant 3 vs. 6), there is a significant decrease in synthesis quality (18.18 vs. 32.58), and the camera parameter error is significantly larger (2.617 vs. 0.015). This shows that the reprojection error constrained by global consistency can significantly enhance the camera parameter estimation, thereby improving the 3DGS training effect and improving the new perspective synthesis ability. In addition, the results of Variant 4 vs. 6 show that 3D track loss can further enhance the geometric consistency of 3DGS. When 3D track loss is used, the PSNR of the novel view synthesis can be further improved by 0.2 dB, and the absolute position error of the camera can be reduced by an order of magnitude. From the experiment, we find that when scale loss is not used, Variant 5 has obvious degradation in all results of the entire scene. This is related to the fact that our method achieves explicit tracking by limiting the size of the tracked Gaussian and using it as a virtual spatial 3D point. Tab. 6 further illustrates the role of scale loss in constraining the size of the tracked Gaussian. It can be seen that after using scale loss, the centroid of the Gaussian is an order of magnitude closer to the surface than those without it.

We further analyze the role of track losses (2D&3D) and scale loss on the CO3D-V2 in Tab. 7. Additionally, we visualize the camera poses and rendered images with and without track losses in Fig. 8. Our observations indicate that global tracks are crucial for improving both novel view synthesis and pose estimation, as the tracks enforce multi-view geometric consistency during 3DGS training.

Effectiveness of Intrinsic Optimization. Accurate camera intrinsics resolve scale ambiguity in 3DGS models, leading to improved novel view synthesis performance. As shown in Tab. 4, our method produces more accurate camera intrinsics (i.e., FoV) compared to COLMAP with original 3DGS. We also performed an ablation study with a fixed camera FoV of 60° and without further optimization. The results, shown in Tab. 9, indicate a 21.4% average decrease in PSNR, due to inaccurate camera intrinsics affecting pose estimation and introducing scale ambiguity.

Comparison with COLMAP-Assisted 3DGS. We compare the synthesis quality from novel views generated by our method against the original 3DGS, where the camera intrinsics and extrinsics are estimated using COLMAP

[1] Due to the hierarchical training of HT-3DGS, it needs significant computational resources and causes OOM errors in our experimental setup. We report results from the paper, only covering partial scenes from CO3D-V2.

Scenes	classroom				lego_c2				livingroom				bedroom			
	FoV(°)↓	RPE_t ↓	RPE_r ↓	ATE↓	FoV(°)	RPE_t	RPE_r	ATE	FoV(°)	RPE_t	RPE_r	ATE	FoV(°)	RPE_t	RPE_r	ATE
CF-3DGS	0.993	0.588	2.436	0.07412	0.021	1.1265	4.946	0.10795	0.029	0.425	2.104	0.07653	0.042	0.366	1.103	0.06284
COLMAP+3DGS	0.993	0.004	0.018	0.00007	0.021	0.009	0.026	0.00019	0.029	0.008	0.026	0.00014	0.042	0.009	0.035	0.00023
Ours	0.012	0.002	0.013	0.00008	0.031	0.002	0.015	0.00011	0.012	0.002	0.013	0.00009	0.003	0.013	0.062	0.00044

Table 4. **Quantitative comparison of parameter accuracy on our Synthetic dataset.** We convert the estimated camera intrinsics focal to FoV and perform the errors of FoV with ground truth (provided by our synthetic datasets) on our method, CF-3DGS and COLMAP+3DGS. As CF-3DGS requires the camera intrinsic parameters as fixed inputs, we set them the same as COLMAP+3DGS.

ID	Variant	PSNR↑	SSIM↑	LPIPS↓	ATE↓	FoV↓
1	COLMAP + 3DGS	32.26	0.91	0.18	0.00020	0.271
2	CF-3DGS	17.30	0.55	0.51	0.08036	0.271
3	w.o. 2D track	18.18	0.56	0.47	0.02020	2.617
4	w.o. 3D track	32.38	0.91	0.17	0.00275	0.063
5	w.o. scale	29.70	0.83	0.18	0.00738	0.069
6	Ours	32.58	0.92	0.16	0.00018	0.015

Table 5. **Ablation study on different losses.**

	classroom	lego_c2	livingroom	bedroom
w. scale	$7.97e^{-5}$	$3.48e^{-3}$	$9.03e^{-4}$	$3.08e^{-3}$
w.o. scale	$1.26e^{-4}$	$8.56e^{-3}$	$3.10e^{-2}$	$2.98e^{-2}$

Table 6. **Effectiveness of scale loss in regulating Gaussian sizes.**

scenes	w.o. 2D&3D track			w.o. scale			Ours		
	PSNR↑	SSIM↑	ATE↓	PSNR	SSIM	ATE	PSNR	SSIM	ATE
34_1403_4393	28.23	0.86	0.0179	28.80	0.87	0.0110	28.68	0.88	0.0088
46_2587_7531	30.10	0.90	0.0110	31.64	0.91	0.0006	31.83	0.92	0.0005
106_12648_23157	19.21	0.47	0.0026	21.97	0.63	0.0014	26.18	0.83	0.0009
110_13051_23361	25.13	0.73	0.0208	33.04	0.94	0.0002	33.44	0.94	0.0002
245_26182_52130	30.38	0.80	0.0099	34.00	0.93	0.0001	33.82	0.93	0.0001
407_54965_106262	27.35	0.83	0.0180	29.20	0.86	0.0090	28.73	0.86	0.0100
415_57112_110099	23.79	0.62	0.0041	23.49	0.61	0.0014	30.37	0.88	0.0010
429_60388_117059	24.57	0.62	0.0108	24.91	0.65	0.0096	25.70	0.70	0.0092
mean	26.10	0.73	0.0119	28.38	0.80	0.0042	29.84	0.87	0.0038

Table 7. **Effectiveness of track losses and scale loss.** Performance on both novel view synthesis and camera pose estimation.

scenes	Ours			COLMAP+3DGS		
	PSNR↑	SSIM↑	LPIPS↓	PSNR	SSIM	LPIPS
Church	29.39	0.92	0.10	29.93	0.93	0.09
Barn	31.98	0.94	0.08	31.08	0.95	0.07
Museum	31.92	0.94	0.08	34.47	0.96	0.05
Family	32.22	0.95	0.08	27.93	0.92	0.11
Horse	30.33	0.94	0.09	20.91	0.77	0.23
Ballroom	35.03	0.97	0.03	34.48	0.96	0.04
Francis	33.39	0.92	0.15	32.64	0.92	0.15
Ignatius	29.25	0.90	0.11	30.20	0.93	0.08
mean	31.68	0.94	0.09	30.20	0.92	0.10

Table 8. **Comparison to 3DGS trained with COLMAP poses.**

	classroom	lego_c2	livingroom	bedroom	mean
Ours	36.26	29.36	33.52	31.17	32.58
COLMAP+3DGS	35.81	28.77	32.74	31.73	32.26
Ours with fixed FoV (60°)	34.76	20.96	26.24	21.20	25.79

Table 9. **Effectiveness of intrinsic optimization.**

on the Tanks and Temples dataset. Tab. 8 shows that our method achieves results that slightly outperform the 3DGS model trained with COLMAP-assisted poses across all scenes. Unlike the original 3DGS, which uses a fixed camera pose for training, our method seamlessly integrates 3DGS training with camera parameter estimation, allowing the two tasks to complement each other and ultimately

Figure 7. **COLMAP failure case on Synthetic dataset.**

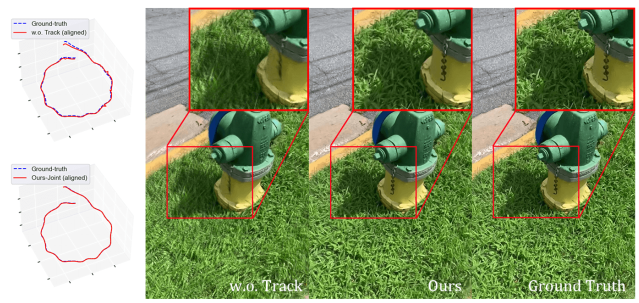

Figure 8. **Visualization of ablation study on track losses in CO3D-V2.** The result without track loss appears blurry in novel view synthesis.

achieve high-quality novel view synthesis. Besides, on a sequence with low-texture areas, COLMAP fails to estimate correct poses, which results in artifacts as shown in Fig. 7.

6. Conclusion

In this work, for the first time, we integrate global tracks with 3DGS and propose a joint optimization framework for COLMAP-Free 3DGS training pipeline. Specifically, this is to avoid the instability, pose drift, and the limitations in handling complex scenes that arise from using local constraints in prior works. In our design, we introduce 2D and 3D track losses to simultaneously constrain geometric consistency in both the 2D and 3D spaces, while also constraining the scale of the track Gaussians to ensure higher spatial accuracy of 3D Gaussians. Additionally, we derive the gradient of the camera intrinsics, an important factor that was previously ignored, allowing the joint optimization of camera parameters and 3DGS to be fully differentiable. We have conducted extensive experiments on challenging datasets, where cameras have severe movements, to validate the effectiveness of our method. However, our current method does not yet support different intrinsic matrices for different views or camera distortion, which we plan to address in the future work.

References

[1] Wenjing Bian, Zirui Wang, Kejie Li, Jiawang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *CVPR*, 2023.

[2] Shin Fang Chng, Sameera Ramasinghe, Jamie Sherrah, and Simon Lucey. Garf: Gaussian activated radiance fields for high fidelity reconstruction and pose estimation. 2022.

[3] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018.

[4] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 224–236, 2018.

[5] Zhiwen Fan, Wenyan Cong, Kairun Wen, Kevin Wang, Jian Zhang, Xinghao Ding, Danfei Xu, Boris Ivanovic, Marco Pavone, Georgios Pavlakos, Zhangyang Wang, and Yue Wang. Instantsplat: Sparse-view sfm-free gaussian splatting in seconds, 2024.

[6] Yang Fu, Sifei Liu, Amey Kulkarni, Jan Kautz, Alexei A. Efros, and Xiaolong Wang. Colmap-free 3d gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20796–20805, 2024.

[7] Ronghang Hu, Nikhila Ravi, Alexander C. Berg, and Deepak Pathak. Worldsheet: Wrapping the world in a 3d sheet for view synthesis from a single image. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021.

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

[9] Bo Ji and Angela Yao. Sfm-free 3d gaussian splatting via hierarchical training, 2024.

[10] 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 *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers '24*, page 1–11. ACM, 2024.

[11] 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.

[12] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4), 2017.

[13] 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 *ICCV*, 2021.

[14] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5721–5731, 2021.

[15] Haotong Lin, Sida Peng, Zhen Xu, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Efficient neural radiance fields for interactive free-viewpoint video. In *SIGGRAPH Asia Conference Proceedings*, 2022.

[16] Jinjie Mai, Wenxuan Zhu, Sara Rojas, Jesus Zarzar, Abdullah Hamdi, Guocheng Qian, Bing Li, Silvio Giancola, and Bernard Ghanem. Tracknerf: Bundle adjusting nerf from sparse and noisy views via feature tracks. In *European Conference on Computer Vision*, 2024.

[17] 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. In *European Conference on Computer Vision*, 2020.

[18] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022.

[19] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12179–12188, 2021.

[20] 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 *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10881–10891, 2021.

[21] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *European Conference on Computer Vision*, 2020.

[22] Gernot Riegler and Vladlen Koltun. Stable view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[23] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4938–4947, 2020.

[24] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[25] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[26] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.

[27] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20697–20709, 2024.

[28] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004.

[29] 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.

[30] 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.

[31] Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for gaussian splatting, 2024.

[32] 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 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1323–1330, 2021.

[33] Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. Differentiable point-based radiance fields for efficient view synthesis. *arXiv preprint arXiv:2205.14330*, 2022.

[34] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

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

[36] Shengjie Zhu, Abhinav Kumar, Masa Hu, and Xiaoming Liu. Tame a wild camera: In-the-wild monocular camera calibration. In *NeurIPS*, 2023.