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

Supplementary Material

1. Derivation of Camera Intrinsic Parameters

With the output image width and height (w, h), as well as the near and far clipping planes (n, f), the extrinsic matrix T_{cw} and the projection matrix P, representing the transformation from camera space to normalized clip space, are denoted as follows:

$$T_{cw} = \begin{bmatrix} R_{cw} & t_{cw} \\ 0 & 1 \end{bmatrix}, P = \begin{bmatrix} \frac{2f_x}{w} & 0 & 0 & 0 \\ 0 & \frac{2f_y}{h} & 0 & 0 \\ 0 & 0 & \frac{f+n}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{1}$$

Fig. 1 demonstrates the computational graph of parameters, which are related to camera intrinsics. In our discussion, the focal length is $F = (f_x, f_y)$ and the principal point is (c_x, c_y). For a 3D Gaussian parameterized by its mean $\mu \in \mathbb{R}^3$ and covariance $\Sigma \in \mathbb{R}^{3\times3}$, the loss $\mathcal{L}$ is formulated by its 2D projected mean $\mu^{'}$ and covariance $\sigma^{'}$. We convert the mean μ into $t = (t_x, t_y, t_z, t_w) \in \mathbb{R}^4$ in camera coordinates, $t^{'} = (t_x^{'}, t_y^{'}, t_z^{'}, t_w^{'}) \in \mathbb{R}^4$ in normalized coordinates (NDC), and finally $\mu^{'} \in \mathbb{R}^2$ in pixel coordinates as follows:

$$t = T_{cw} \begin{bmatrix} \mu & 1 \end{bmatrix}^\top, t^{'} = Pt, \mu^{'} = \begin{bmatrix} \frac{1}{2}(\frac{w \cdot t_x^{'}}{t_w^{'}} + 1) + c_x \\ \frac{1}{2}(\frac{h \cdot t_y^{'}}{t_w^{'}} + 1) + c_y \end{bmatrix}. \tag{2}$$

Notice that the projection of a 3D Gaussian does not result in a 2D Gaussian, the projection of Σ to pixel coordinates is approximated with a first-order Taylor expansion at t in camera space, then the affine transform $J \in \mathbb{R}^{2\times3}$ and the 2D covariance $\Sigma^{'} \in \mathbb{R}^{2\times2}$ [6] are:

$$J = \begin{bmatrix} \frac{f_x}{t_z} & 0 & -\frac{f_x \cdot t_x}{t_z^2} \\ 0 & \frac{f_y}{t_z} & -\frac{f_y \cdot t_y}{t_z^2} \end{bmatrix}, \Sigma^{'} = JR_{cw}\Sigma R_{cw}^\top J^\top. \tag{3}$$

Given the gradients of $\mathcal{L}$ with respect to 2D mean $\mu^{'}$ and covariance $\Sigma^{'}$, we can back-propagate the gradient of focal length F as:

$$\frac{\partial \mathcal{L}}{\partial F} = \frac{\partial \mathcal{L}}{\partial \mu^{'}} \frac{\partial \mu^{'}}{\partial F} + \frac{\partial \mathcal{L}}{\partial \Sigma^{'}} \frac{\partial \Sigma^{'}}{\partial F}. \tag{4}$$

First, we compute the gradient contribution of 2D mean

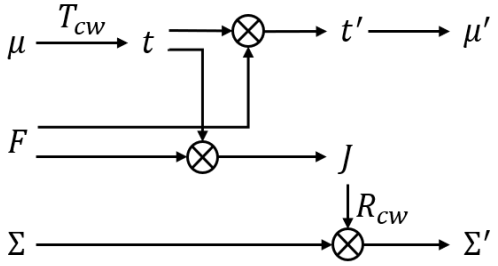

Figure 1. **Computational graph of parameters**.

$\mu^{'}$ to focal length F, $\frac{\partial \mu^{'}}{\partial F}$ can be obtained by the chain rule:

$$\begin{aligned} \frac{\partial \mu^{'}}{\partial F} &= \frac{\partial \mu^{'}}{\partial t^{'}} \frac{\partial t^{'}}{\partial F} \\ &= \begin{bmatrix} \frac{w}{2t_w^{'}} & 0 & 0 & -\frac{wt_x^{'}}{(t_w^{'})^2} \\ 0 & \frac{h}{2t_w^{'}} & 0 & -\frac{ht_y^{'}}{(t_w^{'})^2} \end{bmatrix} \begin{bmatrix} \frac{2t_x}{w} & 0 & 0 & 0 \\ 0 & \frac{2t_y}{h} & 0 & 0 \end{bmatrix}^\top \\ &= \begin{bmatrix} \frac{t_x}{t_z} & 0 \\ 0 & \frac{t_y}{t_z} \end{bmatrix}, \end{aligned} \tag{5}$$

where $t_w^{'} = t_z$ from Eq. 2.

Then, for the second part of Eq. 4, we use another parameter J to compute this component, which means $\frac{\partial \mathcal{L}}{\partial \Sigma^{'}} \frac{\partial \Sigma^{'}}{\partial F} = \frac{\partial \mathcal{L}}{\partial J} \frac{\partial J}{\partial F}$. *gsplat* [5] obtained the gradient of $\mathcal{L}$ to the affine transform J through $T = JR_{cw} \in \mathbb{R}^{2\times3}$ as:

$$\partial \mathcal{L} = < \frac{\partial \mathcal{L}}{\partial T} R_{cw}^\top, \partial J >, \text{where} \frac{\partial \mathcal{L}}{\partial T} = \frac{\partial \mathcal{L}}{\partial \Sigma^{'}} T\Sigma^\top + \frac{\partial \mathcal{L}}{\partial \Sigma^{'}}^\top T\Sigma, \tag{6}$$

with the gradient of J to the focal length F as:

$$\frac{\partial J}{\partial f_x} = \begin{bmatrix} \frac{1}{t_z} & 0 & -\frac{t_x}{t_z^2} \\ 0 & 0 & 0 \end{bmatrix}, \frac{\partial J}{\partial f_y} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{t_z} & -\frac{t_y}{t_z^2} \end{bmatrix}. \tag{7}$$

Finally, the gradient of loss $\mathcal{L}$ with respect to focal length F in Eq. 4 is formulated as:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial f_x} = \frac{t_x}{t_z} \frac{\partial \mathcal{L}}{\partial \mu_x^{'}} + < \frac{\partial \mathcal{L}}{\partial T} R_{cw}^\top, \frac{\partial J}{\partial f_x} >, \\ \frac{\partial \mathcal{L}}{\partial f_y} = \frac{t_y}{t_z} \frac{\partial \mathcal{L}}{\partial \mu_y^{'}} + < \frac{\partial \mathcal{L}}{\partial T} R_{cw}^\top, \frac{\partial J}{\partial f_y} > . \end{cases} \tag{8}$$

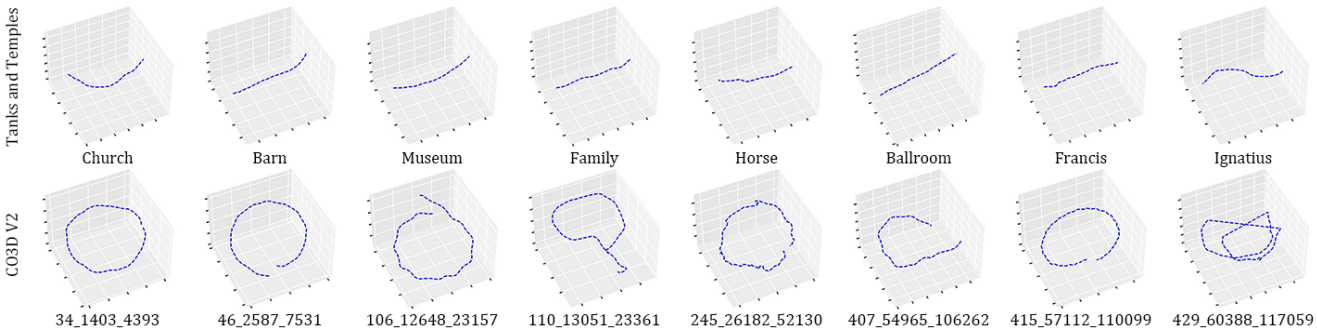

Figure 2. **Ground-truth camera trajectory for Tanks and Temples and CO3D-V2**.

2. Implementation Details

We provide more details about the datasets and training procedure in following sections.

2.1. Dataset

We select three datasets for training and evaluation including existing datasets Tanks and Temples, CO3D-V2, and a virtual Synthetic dataset created by ourself. Tab. 1 shows the details of the scenes in all three datasets, where $Max.\ rot$ is the maximum relative rotation angle between any two frames and the $Avg.\ adj.\ rot$ donates the average relative rotation angle between two adjacent frames. The later represents the magnitude of the relative angle change in a sequence. In comparison, notice that the frame changes in Tanks and Temples are quite small, and our Synthetic datasets are more complex than CO3D-V2, although the $Max.\ rot$ of both datasets is 180 degrees. The visualization of these camera trajectories is shown in Fig. 2.

Synthetic Dataset. Extracting accurate camera parameters from image sequences is challenging. While existing datasets employ COLMAP to derive these parameters, inaccuracies remain. To facilitate more precise comparisons, we have created a Synthetic Dataset using Blender [1]. This dataset includes four indoor scenes. The camera movements in the *classroom* (180 frames) and *lego_c2* (75 frames) scenes are object-centric, whereas those in the *livingroom* (150 frames) and *bedroom* (180 frames) scenes involve roaming. In both types of scenes, the camera navigates complex paths extending over 360 degrees. The visualization of the camera trajectories is illustrated in Fig. 7.

2.2. Training Details

Initialization. To provide the initial values of camera parameters and 3D Gaussians, we first obtain mono-depth maps of images $\mathcal{I}$ by DPT [3]. We extract the feature points $\{p_i\}$ of each image I with SuperPoint [2] and compute the feature matches among all images with SuperGlue [4]. Then we construct the Maximum Spanning Tree (MST) by 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. We extract the set of tracks $\mathcal{P}$ from the MST, where each element $(P, \{p_i\}) \in \mathcal{P}$ is a 3D track point P and its corresponding matching points $\{p_i\}$ associated with the training images. Later on, we define a reprojection loss for the edges of MST, and minimize this objective to optimize the camera intrinsic K and the relative camera extrinsic matrix T_{ji}. This optimization procedure is set as 100 steps in our experiment. Finally, we obtain the initialization of the location of 3D track points P, the camera intrinsic K and extrinsic matrix T_{cw}. The whole algorithm of initialization is summarized in Alg. 1.

	Scenes	Type	Seq. length	Frame rate	Max. rot (deg)	Avg. adj. rot (deg)
Tanks and Temples	Church	indoor	400	30	37.3	0.09
	Barn	outdoor	150	10	47.5	0.32
	Museum	indoor	100	10	76.2	0.76
	Family	outdoor	200	30	35.4	0.13
	Horse	outdoor	120	20	39.0	0.32
	Ballroom	indoor	150	20	30.3	0.20
	Francis	outdoor	150	10	47.5	0.32
	Ignatius	outdoor	120	20	26.0	0.22
CO3D-V2	34_1403_4393	indoor	202	30	180.0	1.53
	46_2587_7531	indoor	202	30	180.0	1.60
	106_12648_23157	outdoor	202	30	180.0	1.33
	110_13051_23361	indoor	202	30	71.6	0.99
	245_26182_52130	indoor	202	30	180.0	1.40
	407_54965_106262	indoor	202	30	180.0	1.46
	415_57112_110099	outdoor	202	30	180.0	1.60
	429_60388_117059	outdoor	202	30	180.0	3.11
Synthetic	classroom	indoor	180	0	180.0	3.84
	lego_c2	indoor	75	0	180.0	3.77
	livingroom	indoor	150	0	180.0	2.63
	bedroom	indoor	180	0	180.0	2.33

Table 1. **Details of the selected sequences.**

lr_μ	lr_q	lr_s	lr_α	$lr_{R_{cw}}$	$lr_{T_{cw}}$
$1.6 * xyz\_scale * 1e^{-2}$	$1e^{-3}$	$5e^{-3}$	$5e^{-2}$	$5e^{-3}$	$1e^{-2}$

Joint optimization. As detailed in Alg. 2, since the initial camera poses and tracking points are very noisy, before training of novel view synthesis, we take a warmup to get

Algorithm 1 Initialization

$\mathcal{I} = \{I_i\}_{i=1}^{M} \leftarrow$ Input images
DPT $\leftarrow$ Monocular Depth Estimation Model
$\{p_i\} \leftarrow \text{SuperPoint}(\mathcal{I})$ ▷ extract feature
$(p_i, p_j) \leftarrow \text{SuperGlue}(\{p_i\})$ ▷ image matching
MST $\leftarrow$ Kruskal Algorithm ▷ construct MST
$\mathcal{P} = (P, \{p_i\}_{i=1}^{l}) \leftarrow \text{Track}(\text{MST})$ ▷ extract track points

loop ▷ Loop 100 iterations
  for edge (i, j) in MST **do**
    $L_{reproj}(i, j) = \|K \cdot (T_{ji} \cdot K^{-1} \cdot p_i) - p_j\|$
    $L_{reproj} += L_{reproj}(i, j)$
  end for
  $K, \{T_{ji}\} \leftarrow \min L_{reproj}$
end loop ▷ Optimize T_{ji}, K
for $(P, \{p_i\})$ in $\mathcal{P}$ **do**
  $d_i \leftarrow \text{DPT}(p_i)$
  $P \leftarrow \text{InitTrackPoint}(K, \{d_i\}, \{p_i\})$
end for ▷ Init Track Points
$T_{cw} = \prod_{(i,j)} T_{ji}$ ▷ Init camera extrinsic

return K, T_{cw}, P

more precise parameters. Here, we draw on the concept of global bundle adjustment, first optimizing these parameters through RGB loss and track loss over 500 epochs. The initial learning rate for each variable is set as above.

For the learning rate of μ, considering the global scale of scene, we introduce the bounding sphere radius of the initial point clouds xyz_scale as a parameter. Additionally, the learning rates for both μ and the camera parameters are decayed using the *ExponentialLR* mechanism. The remaining learning rates are kept constant. Moreover, the leanring rate for the focal length is set differentially: it is set to 0 during the initial 100 epochs, meaning that only the camera's pose and the initial 3DGS will be optimized. After the first 100 epochs, it decreases according to the following formulation:

$$lr_{focal} = \begin{cases} 0.0 & , step \leq 100 \\ \max(1e^{-4}, 5e^{-3} * (1.0 - step/500)) & , step > 100 \end{cases} \tag{9}$$

During warmup, our goal is to achieve a better geometric initialization, at which point the weights of the RGB loss and track loss are all set to 1.0.

During the warmup, we do not perform clone, split, and prune operations on the Gaussian kernels to ensure better geometric constraints on the track points. Afterwards, we will clone new Gaussian kernels from those associated with the track points and apply the same training strategy as the original 3DGS (including clone, split, and prune). However, the tracked Gaussians still need to be preserved without pruning, and should be constrained by a scale loss. The learning rate of μ and camera parameters continue to decay

Algorithm 2 Joint optimization

$lr \leftarrow lr_\mu, lr_q, lr_s, lr_\alpha$ ▷ Initial lr of 3D Gaussians
$lr_{pos} \leftarrow lr_{R_{cw}}, lr_{T_{cw}}$ ▷ Initial lr of camera pose
$G_{track} \leftarrow \text{Gaussian}(P)$ ▷ Initial track 3D Gaussians

procedure WARMUP:
  step $= 0$
  while step < 500 **do**
    $\hat{I}_i, \hat{p}_i \leftarrow \text{Rasterize}(G_{track}, T_{cw,i}, K)$
    $L_i \leftarrow \mathcal{L}(I_i, p_i, \hat{I}_i, \hat{p}_i)$
    $L \leftarrow \sum_i^M L_i$
    $G_{track}, T_{cw}, K \leftarrow \text{Adam}(\nabla L)$ ▷ Update track Gaussians and camera param
    $lr \leftarrow \text{schedule}(lr)$ ▷ Update Gaussian lr
    $lr_{pos} \leftarrow \text{schedule}(lr_{pos})$ ▷ Update pose lr
    if step < 100 **then**
      $lr_{focal} = 0.0$
    else
      $lr_{focal} = \max(1e^{-4}, 5e^{-3} * (1.0 - \frac{\text{step}}{500}))$
    end if ▷ Update focal lr
    step $\leftarrow$ step $+ 1$
  end while
end procedure

procedure JOINT 3DGS:
  step $= 0$
  $\tilde{\mathcal{I}} \leftarrow \text{Queue}(\text{shuffle}(\mathcal{I}))$ ▷ Shuffle images
  while step < 30000 **do**
    $I_i \leftarrow \text{QuePopLeft}(\tilde{\mathcal{I}})$ ▷ Pop first elem in Que
    $G \leftarrow G_{track} + G_{normal}$
    $\hat{I}_i, \hat{p}_i \leftarrow \text{Rasterize}(G, T_{cw,i}, K)$
    $L_i \leftarrow \mathcal{L}(I_i, p_i, \hat{I}_i, \hat{p}_i)$
    $G \leftarrow \text{Adam}(\nabla L_i)$ ▷ Update Gaussians
    $lr \leftarrow schedule(lr)$ ▷ Update Gaussians lr
    if $\tilde{\mathcal{I}} == \emptyset$ **then**
      $L \leftarrow \sum_i^M L_i$
      $T_{cw}, K \leftarrow \text{Adam}(\nabla L)$ ▷ Update all cameras param
      $lr_{pos}, lr_{focal} \leftarrow \text{schedule}(lr_{pos}, lr_{focal})$ ▷ Update all cameras lr
      $\tilde{\mathcal{I}} \leftarrow \text{Queue}(\text{shuffle}(\mathcal{I}))$ ▷ Shuffle images
    end if
    for all $(\mu, \sum, c, \alpha)$ in G **do**
      if $\nabla_p L < \tau_p$ **then** ▷ Densification
        $\text{SplitGaussians}(\mu, \sum, c, \alpha)$
        $\text{CloneGaussians}(\mu, \sum, c, \alpha)$
      end if
      if $\alpha < \epsilon$ or IsTooLarge$(\mu, \sum)$ **then**
        $\text{RemoveGaussian}(G_{normal})$ ▷ Pruning
      end if
    end for ▷ Adaptive control of Gaussians
    step $\leftarrow$ step $+ 1$
  end while
end procedure

Scene	classroom					lego_c2					livingroom					bedroom				
	PSNR	SSIM	LPIPS	ATE	FoV	PSNR	SSIM	LPIPS	ATE	FoV	PSNR	SSIM	LPIPS	ATE	FoV	PSNR	SSIM	LPIPS	ATE	FoV
COLMAP+3DGS	35.81	0.94	0.15	0.00023	0.993	28.77	0.88	0.15	0.00019	0.021	32.74	0.87	0.27	0.00014	0.0291	31.73	0.94	0.13	0.00023	0.042
w.o. 2D Track Loss	24.08	0.78	0.40	0.00973	1.668	17.60	0.35	0.42	0.01969	3.013	20.82	0.62	0.45	0.01735	3.3760	10.22	0.50	0.60	0.03404	2.413
w.o. 3D Track Loss	35.99	0.94	0.14	0.00012	0.083	29.33	0.90	0.13	0.00012	0.031	33.24	0.88	0.26	0.00021	0.1150	30.97	0.93	0.14	0.01056	0.021
w.o. Scale Loss	33.27	0.87	0.14	0.01800	0.066	25.04	0.71	0.16	0.00016	0.041	30.74	0.83	0.26	0.00024	0.1360	29.75	0.91	0.14	0.01111	0.031
Ours	36.26	0.95	0.13	0.00008	0.012	29.36	0.90	0.12	0.00011	0.031	33.52	0.88	0.24	0.00009	0.0121	31.17	0.93	0.13	0.00044	0.003

Table 2. **Ablatation study on different losses.**

using *ExponentialLR* from the end of the warmup and the other learning rates still remain constant. For optimization of the camera parameters,we update the camera parameters after calculating the loss of all cameras like bundle adjustment (BA). Considering the limitations of GPU resources, we optimize the Gaussians separately for each camera.

3. Additional Experiments and Results

We present additional results of novel view synthesis and camera parameter estimation by our method and other baselines, including NoPe-NeRF and CF-3DGS, on Tanks and Temples, CO3D-V2, and Synthetic Dataset.

3.1. Novel View Synthesis

As shown in Fig. 3, 4 and 6, which are evaluated by the same rules as mentioned in the main paper, our method outperforms other baselines by rendering more photo-realistic images, which benefits from the high quality rendering ability of 3DGS model and the accurate camera parameters estimated by our joint optimization.

3.2. Camera Parameter Estimation

The camera movements in the scenes from Tanks and Temples are minimal and predominantly linear, which results in highly accurate estimated camera parameters across all baselines, as detailed in the main paper. We have included comparison results only for CO3D-V2 in Fig. 5 and for Synthetic Dataset in Fig. 7, comparing our method with CF-3DGS. Our method significantly outperforms the baseline, particularly on the Synthetic Datasets, which involve large camera motions and complex scene.

3.3. Rendering Trajectory

To better illustrate the results of novel view synthesis, we have also created several videos showcasing continuous camera motion using these datasets. Fig. 8 shows novel view synthesis results on scenes from the datasets, in which novel views are sampled from new camera trajectories.

3.4. Ablation Study

To exhibit the effectiveness of different losses in our joint optimization, we ablate each loss of the algorithm on synthetic dataset, since it has ground-truth camera parameters. Tab. 2 reports the synthesis quality and camera parameter errors accross different variants on our synthetic dataset.

References

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

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross. Ewa splatting. *IEEE Transactions on Visualization & Computer Graphics*, 8(3):223–238, 2002.

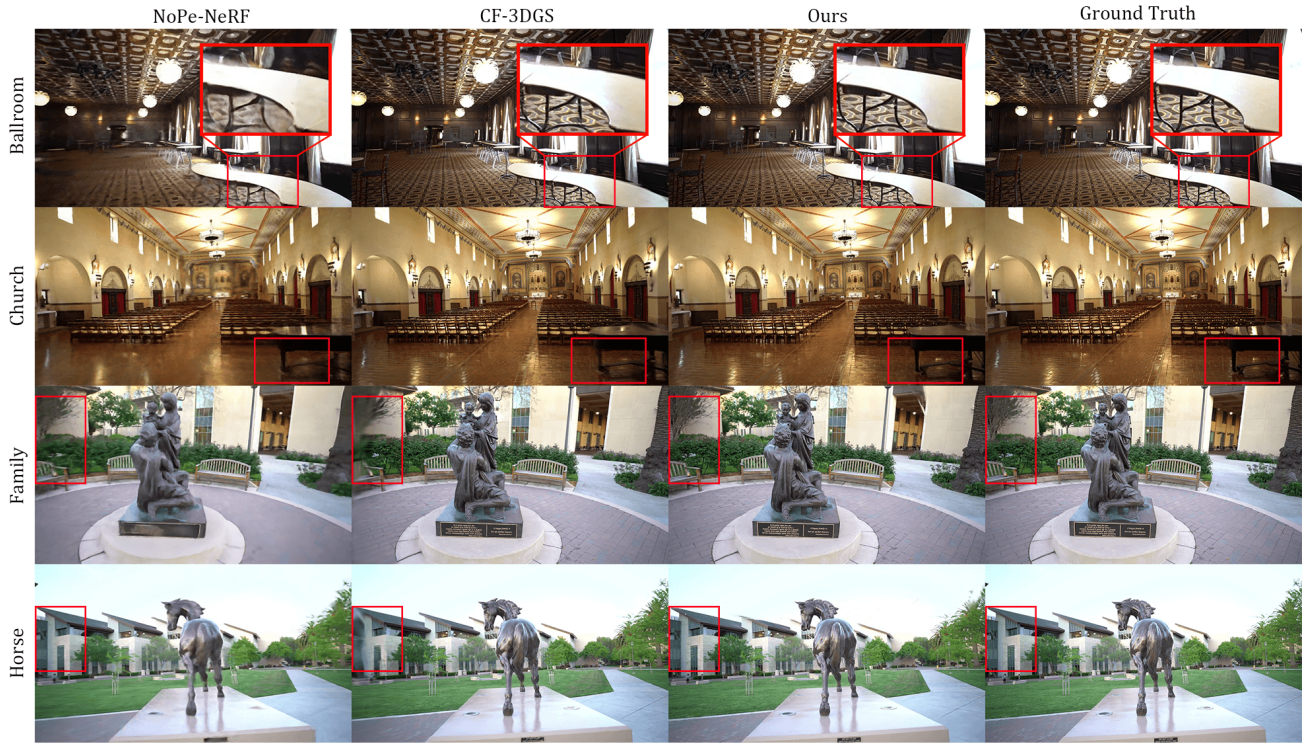

Figure 3. **Comparison of novel view synthesis results on Tanks and Temples.**

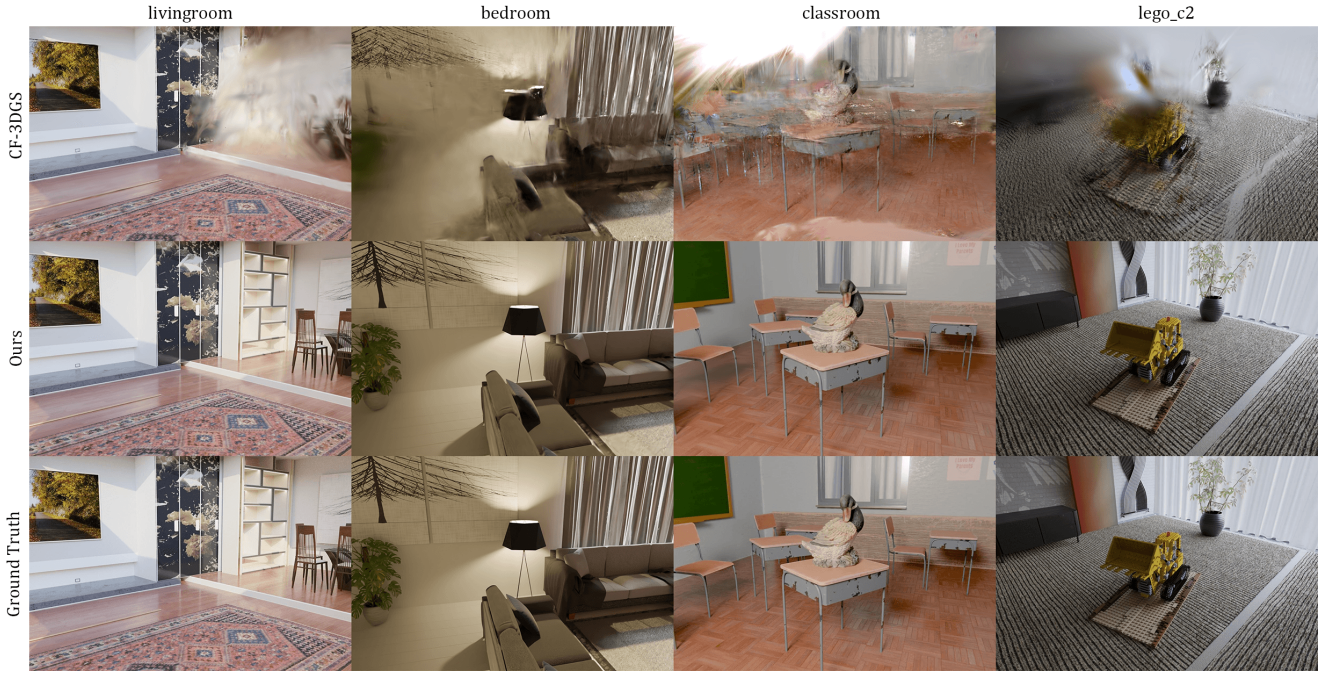

Figure 4. **Comparison of novel view synthesis results on Synthetic dataset.**

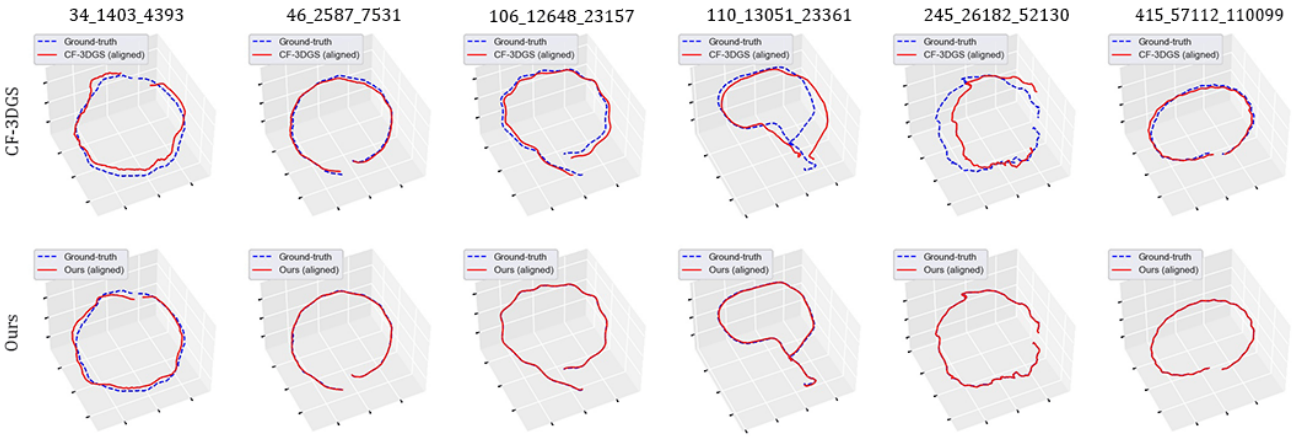

Figure 5. **Comparison of pose estimation on CO3D-V2.**

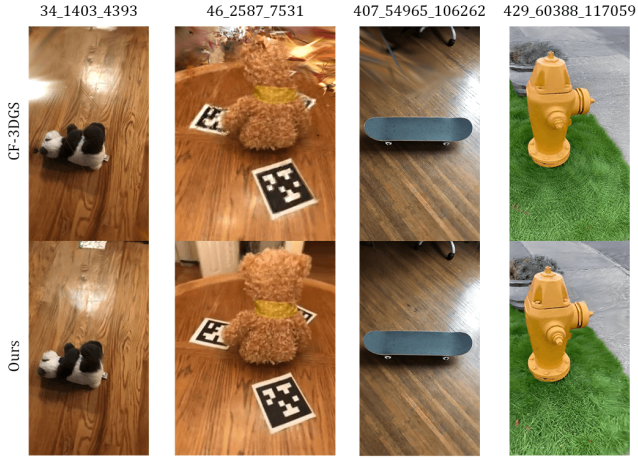

Figure 6. **Comparison of novel view synthesis on CO3D-V2.**

Figure 7. **Comparison of pose estimation on Synthetic dataset.**

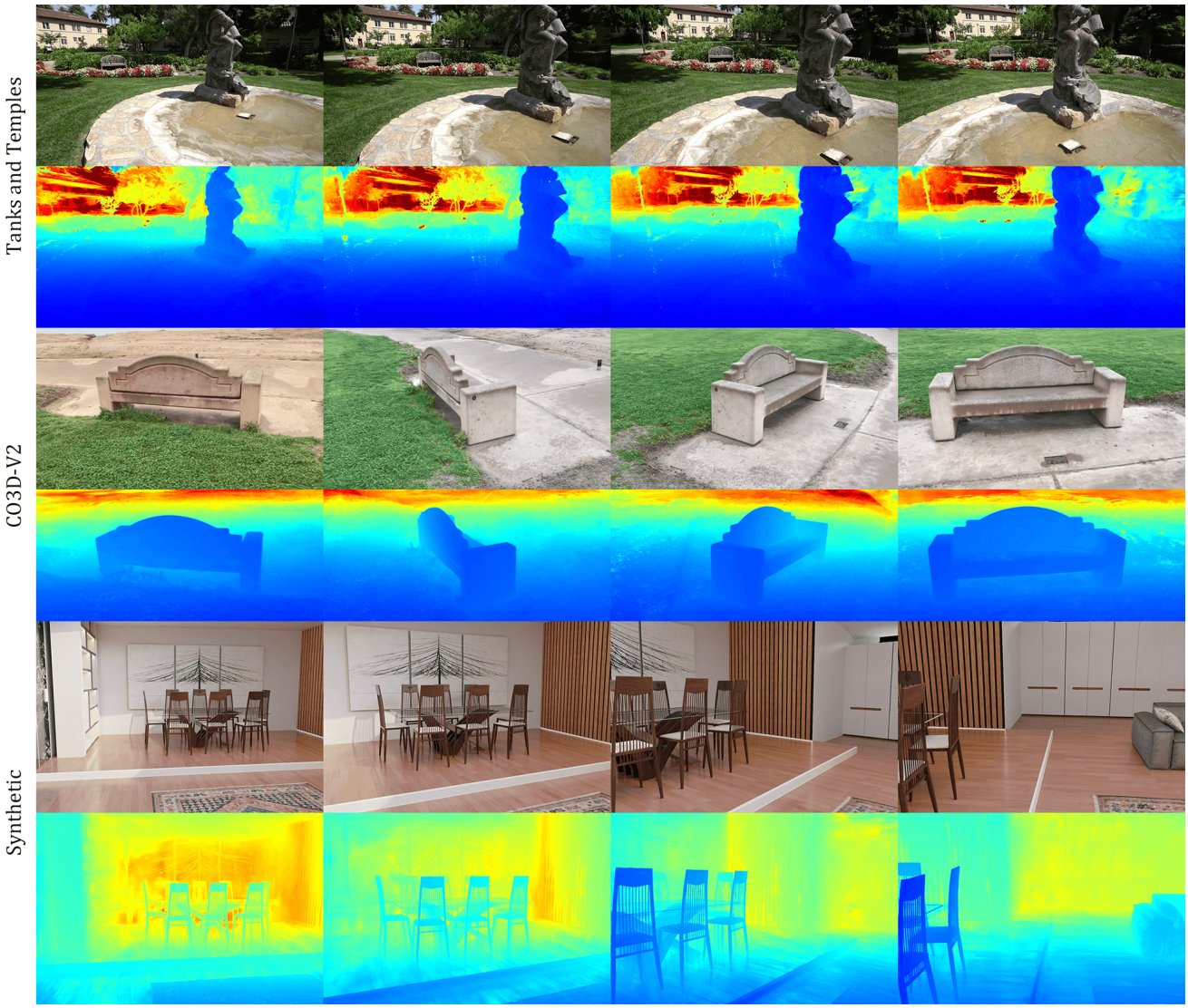

Figure 8. **Novel view synthesis on a new camera trajectory.** We demonstrate the rendering images and the associated depth maps on scenes from three datasets, where the view points are uniformly sampled on a **new** camera trajectory.