SAIL-Recon: Large SfM by Augmenting Scene Regression with Localization

Junyuan Deng[1,2*] Heng Li[1*] Tao Xie[2,3] Weiqiang Ren[2] Qian Zhang[2]
Ping Tan[1†] Xiaoyang Guo[2†]

[1]The Hong Kong University of Science and Technology [2]Horizon Robotics [3]Zhejiang University

Abstract

Scene regression methods, such as VGGT [85], solve the Structure-from-Motion (SfM) problem by directly regressing camera poses and 3D scene structures from input images. They demonstrate impressive performance in handling images under extreme viewpoint changes. However, these methods struggle to handle a large number of input images. To address this problem, we introduce SAIL-Recon, a feed-forward Transformer for large scale SfM, by augmenting the scene regression network with visual localization capabilities. Specifically, our method first computes a neural scene representation from a subset of anchor images. The regression network is then fine-tuned to reconstruct all input images conditioned on this neural scene representation. Comprehensive experiments show that our method not only scales efficiently to large-scale scenes, but also achieves state-of-the-art results on both camera pose estimation and novel view synthesis benchmarks, including TUM-RGBD, CO3Dv2, and Tanks & Temples. We will publish our model and code. Code and models are publicly available at: `https://hkust-sail.github.io/sail-recon/`.

1. Introduction

Structure-from-Motion (SfM) algorithms simultaneously estimate camera poses and scene structures from a collection of unordered images. This problem underlies many computer vision applications, such as novel view synthesis with NeRFs [3, 44], 3DGS [31], multi-view stereo (MVS) reconstruction [94], and visual localization [7]. Traditional SfM methods work either in incremental [59, 64] or global [13, 48] approaches, which rely on crucial components such as feature detection [17], correspondence matching [56], triangulation, and bundle adjustment [76] for joint camera pose and scene structure optimization. However, these individual components are fragile to low-texture, blurred or repeated patterns, which could lead to catastrophic failures in the SfM process.

To overcome the limitation of conventional SfM methods, more recent works [45, 85, 87, 91] develop an end-to-end learning-based SfM pipeline to directly regress scene structures and camera poses from input images. DUSt3R [87] pioneers this scene regression-based approach by training a Transformer [79] to regress the scene coordinate maps (SCM) of two unposed images, which can be used to solve camera poses and correspondences. Some following works [21, 45, 91] extend DUSt3R to multiple input images with 3D constraints, such as scene graph optimization and global alignment. VGGT [85] develops the first large Transformer model to regress almost all 3D results end-to-end with a large dataset and multiple supervisions. Scene regression methods show impressive performance and robustness in handling unposed images with extreme viewpoint changes.

However, many scene regression methods, e.g., VGGT [85], cannot scale up to videos or a large number of input images, as GPU memory usage increases quickly with more images. Some methods [42, 81, 86] tackle video inputs using iteratively updated global memory tokens to fuse the features of each incoming frame, and regress scene coordinate maps conditioned on these memory tokens. Others [21, 43] divide the input video into segments, reconstruct each segment, and align different reconstructions using $Sim(3)$ or $SL(4)$. Both approaches suffer from pose drifting and are heavily dependent on subsequent global alignment to mitigate pose errors.

On the other hand, existing scene regression methods ignore visual localization, a fundamental 3D vision task to solve the camera pose of a query image. Localization can facilitate scaling up an SfM system, a principle commonly employed in simultaneous localization and mapping (SLAM) systems, where mapping is only performed at keyframes and localization is applied to non-keyframes for

*Equal contribution.
†Corresponding authors.
E-mail: pingtan@ust.hk, xiaoyang.guo@horizon.auto

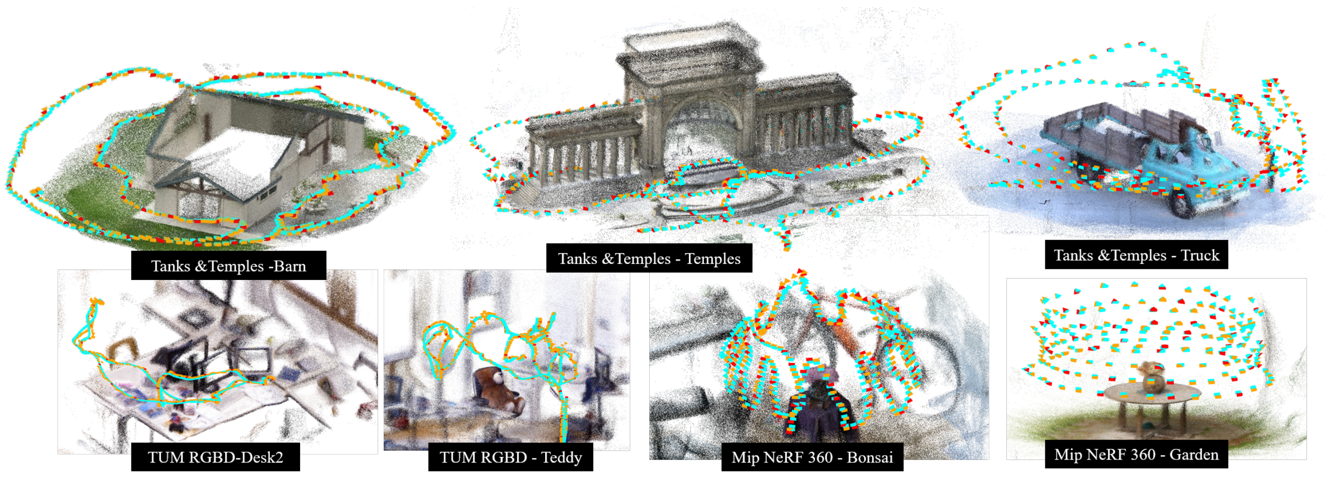

Figure 1. **Regressed Camera Poses and Point Clouds.** We visualize the camera poses and point clouds predicted by SAIL-Recon across various datasets. COLMAP or ground-truth camera poses are shown as blue frustums, while regressed camera poses are shown in yellow, with red indicating anchor images. As illustrated, SAIL-Recon estimates camera poses with accuracy comparable to COLMAP and produces high-quality, geometrically consistent point clouds.

better memory and computation efficiency. Our work seeks to augment scene regression with localization to scale it up in a similar spirit.

Most existing learning-based visual localization works [7–9, 30] require time-consuming per-scene optimization and accurate camera pose annotations for the reference frames. In contrast, we seek to fuse reconstruction and localization into a unified, annotation-free multitask framework for efficient scene regression. Specifically, given a large set of input images or video sequences, we first select a subset of anchor images to generate a global neural scene representation in a single forward pass, thereby avoiding the per-scene training required by previous localization methods. The neural scene representation serves as an implicit neural map of the scene, without relying on explicit 3D points or meshes. Subsequently, the neural scene representation, together with all remaining images, is fed into the same network to jointly recover scene coordinate maps and camera poses for each image. This approach allows for efficient reconstruction of thousands of images in just a few minutes. Unlike the localization process in SLAM systems, we regress map points for all images to produce a more complete 3D map that facilitates robust and dense reconstruction.

Our primary contributions are summarized as follows:

- We introduce SAIL-Recon, a novel feedforward SfM method that generalizes neural scene regression to include localization, resulting in precise and robust reconstruction for thousands of input images in a few minutes.
- We extract a neural scene representation from scene regression network, which serves as a global implicit map for localization.
- We demonstrate through extensive experiments that SAIL-Recon outperforms both traditional and learning-based baselines and achieves state-of-the-art results on SfM and visual localization benchmarks, including TUM-RGBD, CO3Dv2, and Tanks & Temples.

2. Related Works

Geometric Structure-from-Motion (SfM) is a classic computer vision problem [25], which aims to estimate camera poses and 3D scene structures from a collection of unposed images. Traditional SfM solutions are categorized into two main approaches: Incremental SfM [1, 22, 59, 64] initiates the reconstruction with a pair of images and progressively grows it by including images one by one; Global SfM methods [13, 29, 48, 89] determine the global pose of all images simultaneously by motion averaging. Both approaches rely on feature matching, triangulation, and bundle adjustment. Deep learning has significantly advanced these various components, especially in keypoint detection [16, 20, 78, 97] and feature matching [12, 39, 56, 61]. Beyond individual modules, several methods [9, 63, 70, 74, 75, 84, 88] have explored end-to-end differentiable SfM by explicitly enforcing geometric constraints and minimizing reprojection or photometric errors.

Scene Regression-based SfM recovers 3D structures and camera poses from uncalibrated images directly without explicitly enforcing geometric constraints. DUSt3R [87] first employs a transformer model to predict the scene coordinate maps for a pair of images. Subsequent methods employ a global optimization step to expand its result to multiple images [45, 86, 87]. Recent advances have adapted DUSt3R to reconstruct multiple inputs directly [21, 73, 91, 98], and to deal with video inputs with incremental reconstruction [42, 43, 45, 81, 86]. VGGT [85] takes this endeavor further, which addresses nearly all 3D vision tasks in a comprehensive end-to-end fashion with minimum inductive biases while utilizing extensive training data. However, these methods often face challenges when scaling to a large number of input images and may suffer from driftings, even when equipped with additional global alignment.

Visual Localization often relies on a 3D map with reference images of known camera poses. Feature-based approaches extract 2D local features [17, 20, 57, 68] from a query image and match [39, 56] them to 3D points and estimate the query image pose using a perspective-n-point (PnP) algorithm [23]. For large-scale scenes, feature matching focuses on a subset of database images most relevant to the query, improving both accuracy and efficiency [28, 52, 55, 58]. Some learning-based approaches [2, 18, 30, 33, 72, 77, 93] encode the map into a neural network that directly predicts the pose of the query image. The scene coordinate regression approaches [6–8, 62] fit a scene-specific network to predict the 3D coordinates of the image pixels. Most of the localization methods require expensive per-scene training of the localization network. Only a few methods [71, 77, 92] estimate camera pose utilizing a network that simultaneously processes the 3D reconstruction and query image, thus bypassing the need for per-scene fitting. Unlike previous methods, SAIL-Recon circumvents expensive per-scene training by deriving a latent scene representation through scene regression, which is subsequently employed for visual localization.

3. Method

This section presents SAIL-Recon, a unified framework for robust and efficient SfM with thousands of input images from various indoor or outdoor scenes, as shown in Fig. 1. To handle such scenes, we augment *Neural Scene Regression* with *Visual Localization*, which significantly reduces computational costs. We first provide an overview of SAIL-Recon in Sec. 3.1. We then introduce the scene regression backbone in Sec. 3.2, which estimates camera parameters and 3D structures from unordered input images. Building upon neural scene regression, Sec. 3.3 details our approach for constructing a neural scene representation. This representation subsequently serves as the foundation for the visual localization introduced in Sec. 3.4. Sec. 3.5 presents the training methodology, while Sec. 3.6 describes an optional refinement stage.

3.1. Overview

The pipeline of our approach is illustrated in Fig. 2. The core contribution of SAIL-Recon lies in the construction of a neural scene representation from a sparse subset of input images. Instead of relying on explicit geometric prior (e.g., point clouds, posed images), we leverage a neural representation that jointly encodes local visual features and global scene geometry from unposed images. This compact representation enables efficient visual localization across the entire image collection.

Motivated by the impressive performance of VGGT [85], we adopt it as our backbone and augment its scene regression capability by the *Visual Localization* block in Fig. 2, to jointly compute a neural scene representation $\mathcal{R}$ as,

$$(\mathcal{R}, \{T_i, K_i, D_i, S_i\}_{i=1}^{M}) = \mathcal{T}_\theta(\{\mathcal{I}_i\}_{i=1}^{M}), \tag{1}$$

where $\{\mathcal{I}_i\}_{i=1}^{M}$ is the unordered input image set, M is the total number of input images. $\mathcal{T}_\theta$ is a transformer-based network with self-attention blocks parameterized by θ, and $\mathcal{R}$ is the neural scene representation for visual localization. The camera pose of $\mathcal{I}_i$ is specified by the extrinsic and intrinsic matrices $T_i \in \mathbb{R}^{4\times4}$ and $K_i \in \mathbb{R}^{3\times3}$. Furthermore, $D_i \in \mathbb{R}^{H\times W}$ and $S_i \in \mathbb{R}^{H\times W\times3}$ represent the depth map and the scene coordinate map (SCM) for $\mathcal{I}_i$, respectively.

The formulation in Eq. 1 is memory and time consuming. It typically cannot handle more than 100 input images on consumer GPUs. To address this, we use a subset of images, called *Anchor Images* in Fig. 2, to compute the neural scene representation $\mathcal{R}$. Without losing generality, we uniformly sample $N \in [50, 100]$ anchor frames $\{\mathcal{I}_i\}_{i=1}^{N}$ from the full set $\{\mathcal{I}_i\}_{i=1}^{M}$, where M often exceeds 1,000 in large-scale scenes. As illustrated by the *Visual Localization* block in Fig. 2, once we have the scene representation $\mathcal{R}$, we augment the scene regression network $\mathcal{T}_\theta$ to enable localization of a query image $\mathcal{I}^q$ as,

$$\{T^q, K^q, D^q, S^q\} = \mathcal{T}_\theta(\mathcal{I}^q, \mathcal{R}), \tag{2}$$

where T^q and K^q are the extrinsics and intrinsics matrices, and D^q and S^q are the depth and scene coordinate maps of the query image $\mathcal{I}^q$. $\mathcal{T}_\theta$ is the network for both scene regression and localization. We will provide details in Sec 3.4.

Note that we estimate camera parameters and 3D maps of anchor frames during **TRAINING ONLY**, as indicated by the *Operation Flow* in Fig. 2. During inference, we first extract $\mathcal{R}$ from the anchor frames and then process all input images conditioned on $\mathcal{R}$. Although all input images could be reconstructed in a single forward pass in principle, we process them in batches due to GPU memory constraints.

3.2. Neural Scene Regression

The scene regression network takes a set of unposed anchor images $\{\mathcal{I}_i\}_{i=1}^{N}$ as input. Each anchor image $\mathcal{I}_i$ is fed into DINOv2 [47] to extract patchified feature tokens $t^{\mathcal{I}_i} \in \mathbb{R}^{K\times C}$, where $K = (W/14) \times (H/14)$. We follow VGGT [85] to augment $t^{\mathcal{I}_i}$ with an additional camera token $t_g^{\mathcal{I}_i} \in \mathbb{R}^{1\times C}$ and four register tokens $t_r^{\mathcal{I}_i} \in \mathbb{R}^{4\times C}$. The tokens from all anchor frames $\{t^{\mathcal{I}_i}\}$ are subsequently processed through $L = 24$ layers of frame-wise and global

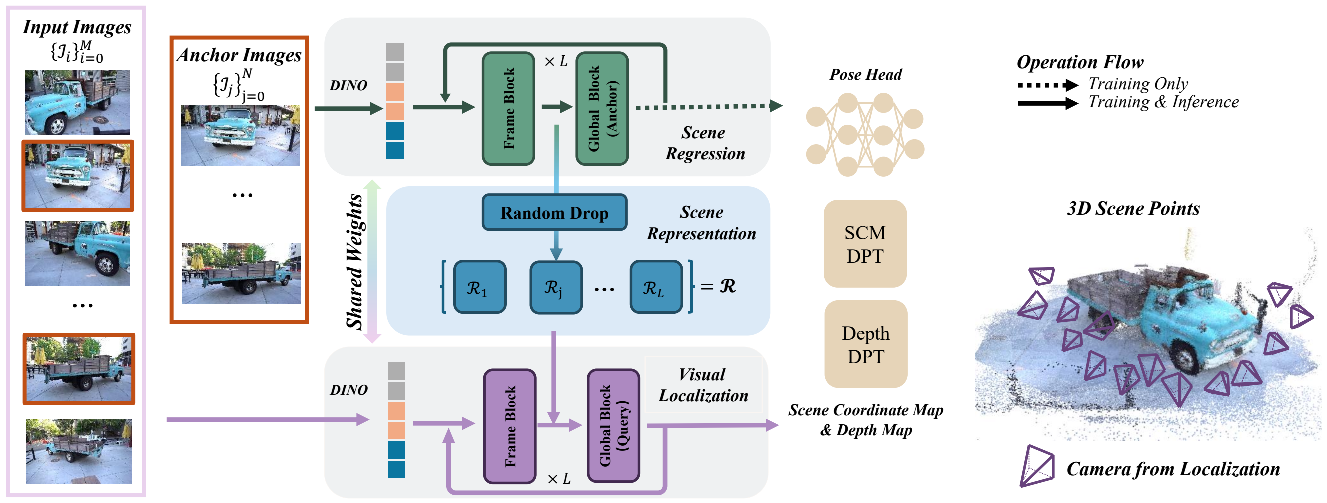

Figure 2. **Architecture Overview.** From a large set of unposed images, we first select a subset as anchor images, which are patchified by DINO [47] with appended camera tokens for scene regression. The *Scene Regression* block extracts a neural scene representation $\mathcal{R}$, which is then used by the *Visual Localization* block to compute the camera poses and 3D scene points for all images.

self-attention as,

$$\{t_j'^{\mathcal{I}_i}\} = \text{attn}_j^{\text{frame}}(\{t_j^{\mathcal{I}_i}\}), \tag{3}$$

$$t_{j+1}^{\mathcal{I}} = \text{attn}_j^{\text{global}}(t_j'^{\mathcal{I}}), \tag{4}$$

where j indexes the attention layers and $\{t_j'^{\mathcal{I}_i}\}$ is the output of frame-wise attention for $\mathcal{I}_i$. $t_j'^{\mathcal{I}} = [\{t_j'^{\mathcal{I}_i}\}_{i=1}^N]$ is the concatenation of all image tokens in the j-th layer. The global attention result $t_{j+1}^{\mathcal{I}}$ is then split along the view dimension into $\{t_{j+1}^{\mathcal{I}_i}\}$ for the frame-wise attention in the next layer.

The image tokens $\{t_L^{\mathcal{I}_i}\}$ in the last layer are then fed into DPT heads [51] to predict depth maps D_i and scene coordinate maps S_i for the input image $\mathcal{I}_i$ as follows,

$$\{D_i, C_i^D, S_i, C_i^S\} = \text{DPT}(\{t_L^{\mathcal{I}_i}\}), \tag{5}$$

where C_i^D and C_i^S are the confidence maps of depth and scene coordinate maps, respectively.

The camera tokens $t_g^{\mathcal{I}_i}$ associated with $\mathcal{I}_i$ on the last layer are processed by a camera head to estimate the intrinsic and extrinsic camera parameters as follows,

$$\{T_i, K_i\} = \text{PoseHead}(\{t_g^{\mathcal{I}_i}\}). \tag{6}$$

3.3. Neural Scene Representation

Scene representation is essential for visual localization. An effective scene representation should encode global 3D scene structure and facilitate correspondences between 3D map points and 2D image pixels. A straightforward choice is to use tokens $t_L^{\mathcal{I}}$ in the final attention layer as scene representation, as these tokens can be used to recover camera poses and dense 3D points map. In other words, we could set $\mathcal{R} = t_L^{\mathcal{I}}$ and then estimate the pose and 3D structure of a query image $\mathcal{I}^q$ by Eq. 2. However, the significant discrepancy between 2D and 3D feature tokens makes it difficult for the network $\mathcal{T}_\theta$ to correlate these feature tokens, even $\mathcal{T}_\theta$ contains a large number of parameters. We empirically find that this design leads to suboptimal results.

Ideally, the scene representation should effectively bridge the gap between 2D and 3D features. Inspired by the fact that the network $\mathcal{T}_\theta$ takes a set of 2D images as input and progressively enhances them to compute 3D structures, we extract intermediate feature tokens from each attention layer of $\mathcal{T}_\theta$ to form the scene representation $\mathcal{R}$. In this way, our scene representation $\mathcal{R}$ captures the gradual evolution from 2D appearance features to 3D coordinate descriptors. Specifically, we compute our scene representation as,

$$\mathcal{R} = [\{\Theta(t_j'^{\mathcal{I}})\}_{j=1}^L], \tag{7}$$

where Θ denotes a downsampling operation applied to intermediate feature tokens $t_j'^{\mathcal{I}}$ to keep the scene representation compact. Specifically, for the anchor frame feature token $t_j'^{\mathcal{I}} \in \mathbb{R}^{N\times K\times C}$, we randomly select a ratio $r \in [0.2, 1.0]$ to sample a subset of tokens in each anchor frame to control the total size of $\Theta(t_j'^{\mathcal{I}}) \in \mathbb{R}^{(N\times\lfloor r\times K\rfloor)\times C}$ in a reasonable range during training. At testing time, we could choose an appropriate r based on the numer of the anchor frames to balance between accuracy and efficency.

3.4. Neural Visual Localization

As described in Eq. 4, the global attention block in $\mathcal{T}_\theta$ aggregates tokens among all anchor frames via self-attention. For visual localization, given a query image $\mathcal{I}^q$, we introduce an attention mask in the global attention blocks, which essentially allows us to compute the cross attention between the query image $\mathcal{I}^q$ and the scene representation $\mathcal{R}_j = \Theta(t_j^{'\mathcal{I}})$:

$$t_{j+1}^q = \text{attn}_j^{\text{global}}(\{t_j^q, \mathcal{R}_j\}). \tag{8}$$

Specifically, tokens from query frames cannot attend to tokens from other query frames; they can only attend to tokens within the same frame and to the scene representation. More details are provided in the Supplementary.

During inference of visual localization, one choice is to compute the camera parameters by applying the PnP algorithm [23] with the scene coordinate map S^q from the DPT head. However, it takes a long time for DPT to up-sample tokens to a high-resolution scene coordinate map. To accelerate pose estimation, we leverage the pose head, which takes only a few camera tokens $t_g^{\mathcal{I}_i}$ as input and directly regresses the camera pose.

$$\{T^q, K^q\} = \text{PoseHead}(t_g^q, \{t_g^{\mathcal{I}_i}\}), \tag{9}$$

where t_g^q and $\{t_g^{\mathcal{I}_i}\}$ are the camera tokens of the query image and the anchor images, respectively. The attention mask is also applied in the pose head. Thus, we can formulate our visual localization for all images as follows,

$$\{T_i, K_i, D_i, S_i\}_{i=1}^M = \mathcal{T}_\theta(\{\mathcal{I}_i\}_{i=1}^M, \mathcal{R}). \tag{10}$$

Note that the anchor images are also processed by this localization block. The scene regression block only extracts the neural scene representation from anchor images.

3.5. Training

Training Losses. During training, we split the input images into the anchor image set and the query image set. We could forward all images in the anchor and query set in a single pass. To this end, we train the SAIL-Recon model $\mathcal{T}_\theta$ end-to-end using a multitask loss on each frame similar to VGGT [85] as,

$$\mathcal{L} = \mathcal{L}_{\text{camera}} + \mathcal{L}_{\text{depth}} + \mathcal{L}_{\text{scm}}. \tag{11}$$

The camera pose loss $\mathcal{L}_{\text{camera}}$ compares the predicted camera parameters $\hat{g}_i$ with the ground truth g_i as

$$\mathcal{L}_{\text{camera}} = \sum_{i=1}^{N} \|\hat{g}_i - g_i\|_1 , \tag{12}$$

where $g_i = [\mathbf{q}, \mathbf{t}, \mathbf{f}]$ includes the camera orientation encoded in a quaternion $\mathbf{q}$, the translation vector $\mathbf{t}$, and the field of view $\mathbf{f}$. We assume that the principal point is at the image center. The depth loss $\mathcal{L}_{\text{depth}}$ follows DUSt3R [87] to measure the discrepancy between the predicted depth $\hat{D}_i$ and the ground truth depth D_i with a predicted uncertainty map $\hat{C}_i^D$. We follow [85] to add a gradient-based smooth term on the depth loss,

$$\mathcal{L}_{\text{depth}} = \sum_{i=1}^{N} \|C_i^D \odot (\hat{D}_i - D_i)\| + \|(\nabla \hat{D}_i - \nabla D_i)\| - \alpha \log C_i^D ,$$

Method	Align.	RRA@5 ↑	RTA@5 ↑	ATE ↓	Reg. ↑	Time [s] ↓
COLMAP [60]	OPT	GT	GT	GT	GT	-
GLOMAP [48]	OPT	75.8	76.7	0.010	100.0	1977
ACE0 [9]	OPT	56.9	57.9	0.015	100.0	5499
DF-SfM [26]	OPT	69.6	69.3	0.014	76.2	-
FlowMap [63]	OPT	31.7	35.7	0.017	66.7	-
VGGSfM [84]	OPT	-	-	-	0.0	2134
MASt3R-SfM [19]	OPT	49.2	54.0	0.011	100.0	2723
DROID-SLAM [75]	OPT	31.3	40.3	0.021	100.0	240
SAIL-Recon-OPT	OPT	71.5	77.7	0.008	100.0	233
Cut3R† [86]	FFD	18.8	25.8	0.017	100.0	42
Spann3R† [82]	FFD	22.1	30.7	0.016	100.0	116
SLAM3R† [42]	FFD	20.3	24.7	0.015	100.0	70
Light3R-SfM [21]	FFD	52.0	52.8	0.011	100.0	63
VGGT-SLAM$^{*\triangle}$ [43]	FFD	57.3	67.9	0.008	100.0	238
SAIL-Recon	FFD	70.4	74.7	0.008	100.0	81

Table 1. **Pose Estimation on Tanks & Temples. [32].** This dataset contains on average over 300 images per scene. We visualize the results using three colors: Best, Second, and Third. A '–' indicates cases where all scenes failed to converge or the running time is unavailable; † denotes that sequential input is required; * indicates evaluation on keyframes; and $\triangle$ denotes that some sequences fail. 'OPT' stands for optimization-based and 'FFD' stands for feedforward-based.

where $\odot$ is an element-wise product. The loss of the scene coordinate map is defined similarly as,

$$\mathcal{L}_{\text{scm}} = \sum_{i=1}^{N} \|C_i^S \odot (\hat{S}_i - S_i)\| + \|(\nabla \hat{S}_i - \nabla S_i)\| - \alpha \log C_i^S .$$

Coordinate Normalization. During training, we randomly select an image $\mathcal{I}_r$ among the anchor images as the reference frame. We then compute the average Euclidean distance from the camera center of $\mathcal{I}_r$ to the 3D points of all anchor frames. We use this scale to normalize the 3D scene and the associated depth and scene coordinate maps.

3.6. Post Refinement

Employing a post-refinement step, such as bundle adjustment (BA) [76], can enhance reconstruction accuracy, particularly when the reconstruction is inferred directly from the network. Previous methods rely on global alignment using depth maps via gradient descent [45, 86, 87], while others use bundle adjustment with pixel correspondences [85]. Both methods are time-consuming and scale poorly to a large number of input images. Since our method provides accurate camera poses in a global coordinate system, we adopt BARF-like methods [37, 69] to optimize camera poses by minimizing a rendering loss. Although this optimization is weaker than bundle adjustment or global alignment, it scales to more than 10K frames and takes only 2-10 minutes. Note that this post-optimization is optional, since the camera poses and scene structures regressed by SAIL-Recon yield strong results for many applications. Further details are provided in Sec. 4.

4. Experiment

In this section, we present a comprehensive evaluation across diverse datasets, covering a wide range of scenarios. We also perform detailed ablation studies to identify the key factors determining the performance of our model. Additional implementation details for both training and inference are provided in the supplementary material.

Training Details. Our training is similar to the setup of VGGT [85]. We train our model by fine-tuning VGGT [85] pre-trained checkpoints. For each batch, we sample 4–48 images, randomly designating 2–24 as anchor frames and treating the remaining images as query frames. We train 30K iterations on 16 NVIDIA A800 GPUs, which takes about four days. We use a diverse mixture of synthetic and real-world datasets, including Co3Dv2 [53], BlendMVS [95], DL3DV [40], MegaDepth [35], WildRGB [90], ScanNet++ [96], HyperSim [54], Mapillary [46], Replica [67], MVS-Synth [27], Virtual KITTI [10], Aria Synthetic Environments, and Aria Digital Twin [49]. These datasets, which represent a subset of those in VGGT [85], are weighted according to their relative sizes to ensure that each dataset contributes proportionally to the overall training process. Please refer to the Supplemental Material for more training details.

4.1. Pose Accuracy Benchmark

We follow [9, 21, 43] to evaluate camera pose accuracy on three benchmarks: Tanks & Temples [32], TUM-RGBD [66], and 7-Scenes [62]. These datasets cover indoor and outdoor environments with image sets and video sequences ranging from 300 to 20k images per scene. We exclude comparisons with methods such as [45, 85, 87, 91] that cannot operate on datasets of this size within a reasonable resource budget. We compute our neural scene representation with 300 tokens from each anchor frame, with a downsample ratio $r \approx 0.2$. All runtime comparisons are conducted under similar GPU throughput conditions, using the Nvidia V100 as the reference hardware.

Tanks & Temples is a dataset includes 21 large-scale indoor and outdoor scenes, with 150–$1,100$ images per scene. We follow [19, 21] to compare our method with optimization-based (OPT) and feedforward-based (FFD) approaches, defined by whether they utilize an explicit optimization on the 3D structure and camera poses. In the OPT category, we compare against DF-SfM [26], GLOMAP [48], PixelSfM [38], VGGSfM [84], ACE-Zero [9], FlowMap [63], and MASt3R-SfM [19]. In the FFD category, we compare with Spann3R [82], Cut3R [86], SLAM3R [42], and Light3R-SfM [21]. We follow [21, 45] to report the proportion of camera pairs with relative rotation error (RRA@5) and relative translation error (RTA@5) below 5^o. The mean value of average translation error (ATE) is calculated between the estimated camera poses and the normalized ground truth poses after Procrustes alignment [24]. As shown in Tab. 1, our method (FFD), significantly outperforms all feedforward-based baselines, including Cut3R [86], Spann3R [82], and SLAM3R [42]. These methods suffer from pose drift due to their incremental reconstruction manner on sequential input. VGGT-SLAM [43] achieves the second best results in average, but only recovers keyframe camera poses and fails to reconstruct *Church*, *Courtroom* and *Palace* due to unstable numerical optimization in the $SL(4)$ manifold. Light3R-SfM [21] is more robust by minimizing pose error on the spanning-tree based scene graph, but with much lower accuracy. Our method achieves SOTA performance while incurring only a marginal computational overhead compared to Light3R-SfM. Furthermore, ours (OPT), which utilized 10K iterations of post-refinement, delivers superior performance in all metrics and is competitive with GLOMAP [48], with only a marginal increase in runtime.

TUM-RGBD is a widely used SLAM benchmark, where each video sequence contains 500–3,000 images. We uniformly select 50 frames as anchor images. Note that our method is an offline SfM method. Here, we show that our method achieves performance similar to that of some SOTA SLAM systems following the standard split in [15, 75]. We exclude Cut3R [86] and SLAM3R [42] from the evaluation because they fail on this benchmark. We evaluate the root mean square error (RMSE) of the absolute trajectory error (ATE) using the `evo` toolkit [24]. As shown in Tab. 2, our method achieves the best results in the uncalibrated setting, without any post-optimization. Results for the calibrated setting are provided in the Supplementary Files. In particular, compared to VGGT-SLAM [43], which employs non-linear factor graph optimization to fuse multiple submaps, our approach achieves higher accuracy without optimization, demonstrating the effectiveness of augmenting scene regression by localization.

7 Scenes is a widely used localization benchmark that provides training and testing split with 2,000–12,000 images per scene. We follow ACE0 [9] to evaluate the localization accuracy. The visual localization methods [8, 9] will train a scene specific localization network with $1,000$-$4,000$ training images, which takes about 10 minutes to 2 hours depending on whether the ground truth camera poses are given. We report the total time of per scene training and localizing all images. We uniformly sample 50 images in the testing split as anchor frames and localize all images in the same split. As shown in Tab. 3, ACE+COLMAP achieves the best performance since it uses ground truth camera poses in training. Without knowing ground truth camera poses, ACE0 and our method achieve the same average localization accuracy. However, ACE0 takes 2 hours to optimize a scene with $4,000$ frames, while SAIL-Recon uses only 8 minutes. We provide more results on 7-Scenes

Method	Sequence									Avg.
	360	desk	desk2	floor	plant	room	rpy	teddy	xyz	
DROID-SLAM* [75]	0.202	0.032	0.091	0.064	0.045	0.918	0.056	0.045	0.012	0.158
MASt3R-SLAM* [45]	0.070	0.035	0.055	0.056	0.035	0.118	0.041	0.114	0.020	0.060
VGGT-SLAM (Sim(3)) [43]	0.123	0.040	0.055	0.254	0.022	0.088	0.041	0.032	0.016	0.074
VGGT-SLAM (SL(4)) [43]	0.071	0.025	0.040	0.141	0.023	0.102	0.030	0.034	0.014	0.053
SAIL-Recon (Offline)	0.070	0.024	0.042	0.107	0.031	0.113	0.020	0.037	0.012	0.051

Table 2. **Root Mean Square Error (RMSE) of Absolute Trajectory Error (ATE) on the TUM RGB-D [66] dataset (unit: m)**. We evaluate methods under an uncalibrated configration following VGGT-SLAM [43], while methods marked with * indicate the intrinsic matrics are provided by GeoCalib [80]. We color result in: Best, Second, and Third.

Prior	ACE KinectFusion	ACE COLMAP	ACE0 OPT.	Ours FFD.
Chess	96.0%	100.0%	100.0%	98.8%
Fire	98.4%	99.5%	98.8%	100.0%
Heads	100.0%	100.0%	100.0%	100.0%
Office	36.9%	100.0%	99.1%	87.4%
Pumpkin	47.3%	100.0%	99.9%	92.8%
Redkitchen	47.8%	98.9%	98.1%	89.9%
Stairs	74.1%	85.0%	61.0%	87.9%
Average	74.1%	97.6%	93.8%	93.8%
Average Time	14min	14min	2h	8min

Table 3. **Localization on 7-Scenes.** Percentage of pose error under (5cm, 5°), compared to pseudo ground truth computed by COLMAP. ACE requires known camera poses during training. Our method achieves comparable localization accuracy to ACE0, where neither approach relies on camera poses in the training set. However, our method is significantly faster than ACE0, which performs self-supervised optimization.

in Supplemantary Files.

4.2. Novel View Synthesis Benchmark

As observed in ACE0 [9], the evaluation of camera poses is sometimes unreliable, as the pseudo ground-truth from COLMAP is only an estimation. Thus, we follow ACE0 [9] to further evaluate camera pose quality through novel view synthesis. Specifically, for each method, we first estimate camera poses for all images of a scene. We then split these images into training and testing sets, and train a Nerfacto [69] model on the training set and render images in the testing view. The rendered images at the testing views are then compared with the ground truth testing images using the Peak Signal-to-Noise Ratio (PSNR) as an indicator of pose accuracy. This evaluation is carried out on the Mip-NeRF 360 [4] and Tanks & Temples [32] datasets. For this evaluation, we enable the post-refinement mentioned in Sec 3.6, since even slight pose noise can prevent the Nefacto model from converging, resulting in poor PSNR. Note that **ALL** baselines in this benchmark are optimization-based methods. Similarly to Sec 4.1, we exclude comparisons with methods [45, 85, 87, 91] that cannot operate on datasets of this size within a reasonable resource budget. We exclude [42, 81] due to poor performance.

Tanks & Temples has two sub-datasets: images and video sequences. For the image set, each scene contains 150–600 images. The video sequence contains $4,000$–$20,000$ frames. We use 100 anchor frames at each scene and evaluate our method in both subsets following ACE0 [9]. We report the results on the image set and leave the comparison on the video sequences in the Supplemantary. Results of compared methods are quoted from ACE0 [9].

		Frames	CMP (D)	Reality Capture	DROID-SLAM† [75]	ACE0 [9]	Ours
Training	Barn	410	24.0	21.2	19.0	16.5	23.5
	Catpr.	383	17.1	15.9	16.6	16.9	16.8
	Church	507	18.3	17.6	14.3	17.2	17.0
	Ignatius	264	20.1	17.7	17.8	19.8	19.5
	MtgRm.	371	18.6	18.1	15.6	18.0	19.5
	Truck	251	21.1	19.0	18.3	20.1	20.9
	Average	364	19.9	18.2	16.9	18.1	19.5
	Time		1h	3min	5min	1.1h	3.5min
Intermediate	Family	152	19.5	18.8	17.6	19.0	20.6
	Francis	302	21.6	20.7	20.7	20.1	21.8
	Horse	151	19.2	19.0	16.3	19.5	20.1
	LightH.	309	16.6	16.5	13.6	17.5	18.2
	PlayGd.	307	19.1	19.2	11.4	18.7	20.3
	Train	301	16.8	15.4	13.8	16.2	16.2
	Average	254	18.8	18.3	15.6	18.5	19.5
	Time		32min	2min	3min	1.3h	3min
Advanced	Audtrm.	302	19.6	12.2	16.7	18.7	20.3
	BallRm.	324	16.3	18.3	13.1	17.9	14.8
	CortRm.	301	18.2	17.2	12.3	17.1	17.4
	Palace	509	14.2	11.7	10.8	10.7	14.3
	Temple	302	18.1	15.7	11.8	9.7	17.8
	Average	348	17.3	15.0	12.9	14.8	16.9
	Time		1h	2min	4min	1h	3.5min

Table 4. **Tanks & Temples.** Pose accuracy via view synthesis with Nerfacto [69]. We report the PSNR in dB and the average reconstruction time. We color code in: Best, Second, and Third. † indicates methods needing sequential inputs.

We use COLMAP with the *default* setting CMP (D) as a reference in Tab. 4. We enable post-refinement with 10K iterations. Our approach achieves the highest PSNR among all baselines, achieving COLMAP-level accuracy while recovering all camera poses in 3-4 minutes. This is significantly faster than COLMAP and ACE0, and is comparable to SLAM systems such as DROID-SLAM, which suffers from pose drifting and produces the lowest PSNR.

	Pseudo GT COLMAP	DROID-SLAM†[75]	BARF [37]	Nope-NeRF[5]	ACE0 [9]	Ours
Bicycle	21.5	10.9	11.9	12.2	18.7	20.50
Bonsai	27.6	10.9	12.5	14.8	25.8	26.76
Counter	25.5	12.9	11.9	11.6	24.5	25.51
Garden	26.3	16.7	13.3	13.8	25.0	24.92
Kitchen	27.4	13.9	13.3	14.4	26.1	27.43
Room	28.0	11.3	11.9	14.3	19.8	27.46
Stump	16.8	13.9	15.0	13.9	20.5	20.83
Average	24.7	12.9	12.8	13.5	22.9	24.77
Average Time	1h	2min	4h	≥24h	8h	5min

Table 5. **Mip-NeRF 360.** Pose accuracy via view synthesis PSNR. Higher is better. We color code in: Best, Second, and Third. † indicates methods needing sequential inputs.

Method	Global Align.	Co3Dv2↑		
		RRA@15	RTA@15	mAA@30
Colmap [59]	OPT	31.6	27.3	25.3
Glomap [48]	OPT	45.9	40.3	37.3
PixSfM [38]	OPT	33.7	32.9	30.1
VGGSfM [84]	OPT	92.1	88.3	74.0
DUSt3R-GA [87]	OPT	96.2	86.8	76.7
MASt3R-SfM [19]	OPT	96.0	93.1	88.0
PoseDiff [83]	FFD	80.5	79.8	66.5
RelPose++ [36]	FFD	82.3	77.2	65.1
Spann3R [82]	FFD	89.5	83.2	70.3
MASt3R * [34]	FFD	94.5	80.9	68.7
Light3R-SfM [21]	FFD	94.7	85.8	72.8
VGGT [85]	FFD	98.4	94.8	88.2
SAIL-Recon $N = 10$	FFD	98.3	94.0	88.1
SAIL-Recon $N = 8$	FFD	98.2	93.6	87.3
SAIL-Recon $N = 5$	FFD	97.7	92.2	85.0
SAIL-Recon $N = 2$	FFD	96.4	89.7	78.5

Table 6. **Ablation on the number of anchor views for pose estimation performance on CO3Dv2 [53].** We evaluate pose estimation accuracy by varying the number of anchor views from 10 input images, randomly sampled from each sequence.

Mip-NeRF 360. Mip-NeRF 360 [4] is a small-scale dataset containing indoor and outdoor scenes with around 150–500 images per scene. We select 50 anchor images per scene and enable post-refinement with 10K iterations. The results are reported in Tab. 5. Again, results of all baseline methods are quoted from ACE0 [9]. Our approach surpasses all baselines, matching PSNR scores with pseudo ground-truth poses from COLMAP. NoPe-NeRF[5] and DROID-SLAM [75] struggle due to wide baselines between images, with NoPe-NeRF[5] needing two days of training. BARF [37] has difficulty starting from scratch. ACE0 [9] is inferior to SAIL-Recon in both accuracy and runtime.

4.3. Ablations

Number of Anchor Images. We evaluate the effect of the number of anchor images on the CO3Dv2 dataset [53]. For each 10 input images, we randomly select $N \in 2, 5, 8, 10$ anchor images to compute the neural scene representation and localize all 10 images. As shown in Tab. 6, our method maintains pose accuracy close to the original VGGT. Its performance drops slowly as the number of anchor images decreases. Remarkably, even with as few as two anchor images, our method still delivers strong performance, underscoring its robustness to sparse anchor images.

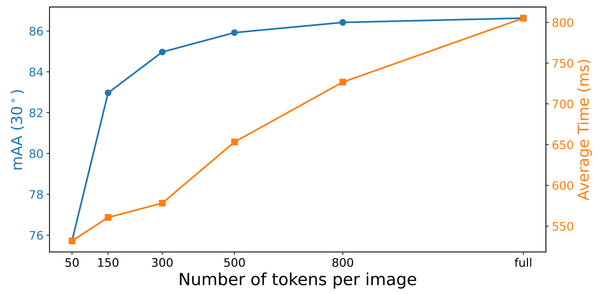

Figure 3. **Pose Accuracy & Runtime vs. Tokens per Image.** We choose 300 tokens per image to balance accuracy and efficiency.

Method	Co3Dv2↑			
	mAA@30	mAA@5	RRA@15	RTA@15
SAIL-Recon	**87.3**	**57.6**	**98.3**	**93.6**
Fixed token	86.5	53.6	98.2	93.6
Avg. Pooling	86.5	53.5	98.2	93.5

Table 7. **Ablation on Training Strategy.** We investigate the different training strategies on pose estimation accuracy.

Number of Tokens/Downsample Ratio r. We investigate the impact of the number of downsampled tokens per anchor image on both pose accuracy and runtime with the number of anchor images fixed at $N = 5$, as shown in Fig. 3. Increasing the number of tokens improves pose accuracy, but also leads to a steady growth in processing time. We selected 300 tokens per image as a trade-off, since it achieves reasonable accuracy with low computation cost.

Training Strategy. We further evaluate the effect of our training strategy. Specifically, during training, our method selects a random number of tokens per image. We compare it with two alternatives: (i) using a fixed number of tokens (300 per image) and (ii) applying average pooling over image tokens to achieve a $4\times$ downsampling (resulting in approximately 340 tokens per image). As shown in Tab. 7, our variant token strategy yields more accurate pose estimation. In particular, our method outperforms the average pooling baseline. We attribute it to dropout [65], where our random selection acts as a regularization mechanism, which improves generalization. Moreover, the variant token strategy offers greater flexibility than pooling, enabling an explicit trade-off between accuracy and efficiency.

5. Conclusion

We introduced SAIL-Recon, a feedforward SfM method that can scale up to thousands of input images. It is achieved by augmenting the scene regression Transformer with localization capabilities. By computing a neural scene representation from a subset of anchor images, we fine-tune the Transformer for localization conditioned on the neural scene representation. In this way, the fine-tuned Transformer can quickly reconstruct camera poses and scene points for all the input images. Experiments on various benchmarks show state-of-the-art results in both pose estimation and novel view synthesis, surpassing traditional and learning-based baselines in accuracy and efficiency.

Future Work & Limitation. While our model demonstrates strong performance, two key limitations remain. First, global pose estimation in a pre-fixed reference coordinate system might lead to a performance drop on some sequences. A better view selection criterion could improve results. Second, uniform anchor image sampling risks missing large or diverse scene regions. We could explore coverage-aware selection that maximizes visibility.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54 (10):105–112, 2011. 2

[2] Vassileios Balntas, Shuda Li, and Victor Prisacariu. Relocnet: Continuous metric learning relocalisation using neural nets. In *Proceedings of the European conference on computer vision (ECCV)*, pages 751–767, 2018. 3

[3] 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. 1

[4] 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. 7, 8

[5] Wenjing Bian, Zirui Wang, Kejie Li, Jiawang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. 2023. 8, 2

[6] Eric Brachmann and Carsten Rother. Visual camera relocalization from rgb and rgb-d images using dsac. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5847–5865, 2021. 3

[7] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC-Differentiable RANSAC for camera localization. In *CVPR*, 2017. 1, 2

[8] Eric Brachmann, Tommaso Cavallari, and Victor Adrian Prisacariu. Accelerated coordinate encoding: Learning to relocalize in minutes using rgb and poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5044–5053, 2023. 3, 6

[9] Eric Brachmann, Jamie Wynn, Shuai Chen, Tommaso Cavallari, Áron Monszpart, Daniyar Turmukhambetov, and Victor Adrian Prisacariu. Scene coordinate reconstruction: Posing of image collections via incremental learning of a relocalizer. In *ECCV*, 2024. 2, 5, 6, 7, 8, 1

[10] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv:2001.10773*, 2020. 6

[11] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE transactions on robotics*, 37(6):1874–1890, 2021. 2, 3

[12] Hongkai Chen, Zixin Luo, Jiahui Zhang, Lei Zhou, Xuyang Bai, Zeyu Hu, Chiew-Lan Tai, and Long Quan. Learning to match features with seeded graph matching network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6301–6310, 2021. 2

[13] Zhaopeng Cui and Ping Tan. Global structure-from-motion by similarity averaging. In *Proceedings of the IEEE international conference on computer vision*, pages 864–872, 2015. 1, 2

[14] Jan Czarnowski, Tristan Laidlow, Ronald Clark, and Andrew J Davison. Deepfactors: Real-time probabilistic dense monocular slam. *IEEE Robotics and Automation Letters*, 5 (2):721–728, 2020. 2, 3

[15] Junyuan Deng, Ling Pei, Qi Wu, Tao Li, Xin Chen, and Wenxian Yu. An adaptive feature optimization strategy for direct visual odometry. In *International Conference on Autonomous Unmanned Systems*, pages 2919–2930. Springer, 2021. 6

[16] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. 2

[17] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. 1, 3

[18] Mingyu Ding, Zhe Wang, Jiankai Sun, Jianping Shi, and Ping Luo. Camnet: Coarse-to-fine retrieval for camera relocalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2871–2880, 2019. 3

[19] Bardienus Duisterhof, Lojze Zust, Philippe Weinzaepfel, Vincent Leroy, Yohann Cabon, and Jerome Revaud. Mast3r-sfm: a fully-integrated solution for unconstrained structure-from-motion. *arXiv preprint arXiv:2409.19152*, 2024. 5, 6, 8

[20] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 8092–8101, 2019. 2, 3

[21] Sven Elflein, Qunjie Zhou, and Laura Leal-Taixé. Light3r-sfm: Towards feed-forward structure-from-motion. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16774–16784, 2025. 1, 2, 5, 6, 8

[22] Jan-Michael Frahm, Pierre Fite-Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, et al. Building rome on a cloudless day. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Her-*

aklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11, pages 368–381. Springer, 2010. 2

[23] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):930–943, 2003. 3, 5

[24] Michael Grupp. evo: Python package for the evaluation of odometry and slam. `https://github.com/MichaelGrupp/evo`, 2017. Accessed: 2025-08-09. 6

[25] Richard Hartley and Andrew Zisserman. Multiple view geometry in computer vision 2nd ed., 4th print, 2006. 2

[26] Xingyi He, Jiaming Sun, Yifan Wang, Sida Peng, Qixing Huang, Hujun Bao, and Xiaowei Zhou. Detector-free structure from motion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21594–21603, 2024. 5, 6

[27] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6

[28] Martin Humenberger, Yohann Cabon, Noé Pion, Philippe Weinzaepfel, Donghwan Lee, Nicolas Guérin, Torsten Sattler, and Gabriela Csurka. Investigating the role of image retrieval for visual localization: An exhaustive benchmark. *International Journal of Computer Vision*, 130(7):1811–1836, 2022. 3

[29] Nianjuan Jiang, Zhaopeng Cui, and Ping Tan. A global linear method for camera pose registration. In *Proceedings of the IEEE international conference on computer vision*, pages 481–488, 2013. 2

[30] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015. 2, 3

[31] 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. 1

[32] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017. 5, 6, 7, 2

[33] Zakaria Laskar, Iaroslav Melekhov, Surya Kalia, and Juho Kannala. Camera relocalization by computing pairwise relative poses using convolutional neural network. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 929–938, 2017. 3

[34] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European Conference on Computer Vision*, pages 71–91. Springer, 2024. 8

[35] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2041–2050, 2018. 6

[36] Amy Lin, Jason Y Zhang, Deva Ramanan, and Shubham Tulsiani. Relpose++: Recovering 6d poses from sparse-view observations. In *2024 International Conference on 3D Vision (3DV)*, pages 106–115. IEEE, 2024. 8

[37] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5741–5751, 2021. 5, 8, 2, 4

[38] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5987–5997, 2021. 6, 8

[39] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. Lightglue: Local feature matching at light speed. *arXiv preprint arXiv:2306.13643*, 2023. 2, 3

[40] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024. 6

[41] Lahav Lipson, Zachary Teed, and Jia Deng. Deep patch visual slam. In *European Conference on Computer Vision*, pages 424–440. Springer, 2024. 2, 3

[42] Yuzheng Liu, Siyan Dong, Shuzhe Wang, Yingda Yin, Yanchao Yang, Qingnan Fan, and Baoquan Chen. Slam3r: Real-time dense scene reconstruction from monocular rgb videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16651–16662, 2025. 1, 3, 5, 6, 7, 2

[43] Dominic Maggio, Hyungtae Lim, and Luca Carlone. Vggt-slam: Dense rgb slam optimized on the sl (4) manifold. *arXiv preprint arXiv:2505.12549*, 2025. 1, 3, 5, 6, 7

[44] 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*, 65(1):99–106, 2021. 1

[45] Riku Murai, Eric Dexheimer, and Andrew J Davison. Mast3r-slam: Real-time dense slam with 3d reconstruction priors. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16695–16705, 2025. 1, 2, 3, 5, 6, 7

[46] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the IEEE international conference on computer vision*, pages 4990–4999, 2017. 6

[47] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 3, 4

[48] Linfei Pan, Daniel Barath, Marc Pollefeys, and Johannes Lutz Schönberger. Global Structure-from-Motion Revisited. In *European Conference on Computer Vision (ECCV)*, 2024. 1, 2, 5, 6, 8

[49] Xiaqing Pan, Nicholas Charron, Yongqian Yang, Scott Peters, Thomas Whelan, Chen Kong, Omkar Parkhi, Richard Newcombe, and Yuheng Carl Ren. Aria digital twin: A new benchmark dataset for egocentric 3d machine perception. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20133–20143, 2023. 6

[50] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 1

[51] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 4

[52] Anita Rau, Guillermo Garcia-Hernando, Danail Stoyanov, Gabriel J Brostow, and Daniyar Turmukhambetov. Predicting visual overlap of images through interpretable non-metric box embeddings. In *European Conference on Computer Vision*, pages 629–646. Springer, 2020. 3

[53] 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 *International Conference on Computer Vision*, 2021. 6, 8

[54] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10912–10922, 2021. 6

[55] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12716–12725, 2019. 3

[56] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 1, 2, 3

[57] Paul-Edouard Sarlin, Ajaykumar Unagar, Mans Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, et al. Back to the feature: Learning robust camera localization from pixels to pose. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3247–3257, 2021. 3

[58] Torsten Sattler, Akihiko Torii, Josef Sivic, Marc Pollefeys, Hajime Taira, Masatoshi Okutomi, and Tomas Pajdla. Are large-scale 3d models really necessary for accurate visual localization? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1637–1646, 2017. 3

[59] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2, 8

[60] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 5

[61] Yan Shi, Jun-Xiong Cai, Yoli Shavit, Tai-Jiang Mu, Wensen Feng, and Kai Zhang. Clustergnn: Cluster-based coarse-to-fine graph neural network for efficient feature matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12517–12526, 2022. 2

[62] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2930–2937, 2013. 3, 6

[63] Cameron Smith, David Charatan, Ayush Tewari, and Vincent Sitzmann. Flowmap: High-quality camera poses, intrinsics, and depth via gradient descent. *arXiv preprint arXiv:2404.15259*, 2024. 2, 5, 6

[64] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. 2006. 1, 2

[65] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. 8

[66] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 573–580. IEEE, 2012. 6, 7, 3

[67] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6229–6238, 2021. 6

[68] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8922–8931, 2021. 3

[69] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, 2023. 5, 7, 1, 3, 4

[70] Chengzhou Tang and Ping Tan. Ba-net: Dense bundle adjustment network. *arXiv preprint arXiv:1806.04807*, 2018. 2

[71] Shitao Tang, Chengzhou Tang, Rui Huang, Siyu Zhu, and Ping Tan. Learning camera localization via dense scene matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1831–1841, 2021. 3

[72] Shitao Tang, Sicong Tang, Andrea Tagliasacchi, Ping Tan, and Yasutaka Furukawa. Neumap: Neural coordinate mapping by auto-transdecoder for camera localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 929–939, 2023. 3

[73] Zhenggang Tang, Yuchen Fan, Dilin Wang, Hongyu Xu, Rakesh Ranjan, Alexander Schwing, and Zhicheng Yan. Mv-dust3r+: Single-stage scene reconstruction from sparse views in 2 seconds. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5283–5293, 2025. 2

[74] Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. *arXiv preprint arXiv:1812.04605*, 2018. 2, 3

[75] Zachary Teed and Jia Deng. DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras. *Advances in neural information processing systems*, 2021. 2, 5, 6, 7, 8, 3

[76] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999. 1, 5

[77] Mehmet Ozgur Turkoglu, Eric Brachmann, Konrad Schindler, Gabriel J Brostow, and Aron Monszpart. Visual camera re-localization using graph neural networks and relative pose supervision. In *2021 International Conference on 3D Vision (3DV)*, pages 145–155. IEEE, 2021. 3

[78] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. *Advances in Neural Information Processing Systems*, 33:14254–14265, 2020. 2

[79] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1

[80] Alexander Veicht, Paul-Edouard Sarlin, Philipp Lindenberger, and Marc Pollefeys. GeoCalib: Single-image Calibration with Geometric Optimization. In *ECCV*, 2024. 7, 1

[81] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. *arXiv preprint arXiv:2408.16061*, 2024. 1, 3, 7

[82] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. In *International Conference on 3D Vision 2025*, 2025. 5, 6, 8

[83] Jianyuan Wang, Christian Rupprecht, and David Novotny. Posediffusion: Solving pose estimation via diffusion-aided bundle adjustment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9773–9783, 2023. 8

[84] Jianyuan Wang, Nikita Karaev, Christian Rupprecht, and David Novotny. Vggsfm: Visual geometry grounded deep structure from motion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21686–21697, 2024. 2, 5, 6, 8

[85] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025. 1, 3, 5, 6, 7, 8, 2

[86] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10510–10522, 2025. 1, 2, 3, 5, 6

[87] 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*, pages 20697–20709, 2024. 1, 2, 5, 6, 7, 8

[88] Xingkui Wei, Yinda Zhang, Zhuwen Li, Yanwei Fu, and Xiangyang Xue. Deepsfm: Structure from motion via deep bundle adjustment. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 230–247. Springer, 2020. 2

[89] Kyle Wilson and Noah Snavely. Robust global translations with 1dsfm. In *European conference on computer vision*, pages 61–75. Springer, 2014. 2

[90] Hongchi Xia, Yang Fu, Sifei Liu, and Xiaolong Wang. Rgbd objects in the wild: Scaling real-world 3d object learning from rgb-d videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22378–22389, 2024. 6

[91] Jianing Yang, Alexander Sax, Kevin J Liang, Mikael Henaff, Hao Tang, Ang Cao, Joyce Chai, Franziska Meier, and Matt Feiszli. Fast3r: Towards 3d reconstruction of 1000+ images in one forward pass. *arXiv preprint arXiv:2501.13928*, 2025. 1, 2, 6, 7

[92] Luwei Yang, Ziqian Bai, Chengzhou Tang, Honghua Li, Yasutaka Furukawa, and Ping Tan. Sanet: Scene agnostic network for camera localization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 42–51, 2019. 3

[93] Luwei Yang, Rakesh Shrestha, Wenbo Li, Shuaicheng Liu, Guofeng Zhang, Zhaopeng Cui, and Ping Tan. Scenesqueezer: Learning to compress scene for camera relocalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8259–8268, 2022. 3

[94] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European conference on computer vision (ECCV)*, pages 767–783, 2018. 1

[95] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. *Computer Vision and Pattern Recognition (CVPR)*, 2020. 6

[96] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023. 6

[97] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: Learned Invariant Feature Transform. In *Proc. ECCV*, 2016. 2

[98] Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. Flare: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21936–21947, 2025. 2

[99] Youmin Zhang, Fabio Tosi, Stefano Mattoccia, and Matteo Poggi. Go-slam: Global optimization for consistent 3d instant reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3727–3737, 2023. 2, 3

SAIL-Recon: Large SfM by Augmenting Scene Regression with Localization

Supplementary Material

In this supplementary material, we provide additional implementation details and experimental setups in Sec.6. We also present further experiments and discussions in Sec.7. More visualization result will be posted in Sec. 8.

6. More Implementation Details

Training Details. As described, we follow the training set of VGGT [85], which we use a cosine learning rate scheduler with a maximum learning rate of 2×10^{-4} and a warm-up of 2K iterations. The input images are resized to a maximum of 518 pixels while preserving the aspect ratios between $[0.33, 1.0]$. Data augmentation includes random color jittering, Gaussian blur, and grayscale conversion. Training is performed with bfloat16 precision, gradient checkpointing, and a mixed anchor–query frame strategy.

Implementation of attn$^{\text{query}}$. We apply an attention mask to realize a cross-attention–like operation between tokens from the query image $\mathcal{I}^q$ and the scene representation $\mathcal{R}_j = \Theta(t_j^{\prime\mathcal{I}})$ from layer j. Specifically, during training, the mask enforces two types of interaction:

- Anchor interaction: tokens from anchor frames are allowed to attend to each other, enabling mutual information exchange across different anchor views
- Query restriction: tokens from query frames cannot attend to tokens from other query frames; they can only attend to tokens within the same frame and to the scene representation $\mathcal{R}_j$.

This design ensures that query tokens extract information primarily from the global scene representation and their own local context, while anchor tokens remain fully connected to maximize cross-view aggregation. The same attention mask is also applied to the attention layer in the pose head.We also develop an alternative version that employs two distinct blocks for the anchor and query, respectively. The query block is initialized from the anchor block and fine-tuned during training. We observe that both versions perform similarly.

Inference Details. As stated in Sec.3.1, we first extract the scene representation $\mathcal{R}$ from the anchor frames and then process query images sequentially. Specifically, we employ a KV-cache[50] to store $\mathcal{R}$ as the keys and values in each global attention layer, which effectively accelerates computation and reduces memory usage. For each subsequent query image, its tokens serve as the queries in the attention mechanism, while the keys/values are formed by concatenating the query tokens with the cached scene tokens. Through the attention operation, the query image tokens are updated by aggregating information from the global scene representation. After passing through all attention layers, we obtain tokens enriched with localization information. These tokens are then fed into the camera head and depth head to predict the corresponding camera parameters, depth and scene coordinate maps, yielding the reconstructed scene from the query viewpoint.

Post Refinement Details We adopt Nerfacto [69] within NeRF Studio, applying its camera pose optimizer for post refinement. For scenes with $\leq 2,000$ images, models undergo 10,000 training iterations with a regularization weight $\lambda = 0.001$ on both camera translation and rotation. The optimizer uses an initial learning rate of 10^{-3}, which decays to 10^{-4} after $1,000$ iterations via cosine annealing. All other parameters follow the defaults of NeRF Studio. For scenes with $\geq 2,000$ images, we perform two separate optimizations with 10000 and 30000 iterations, respectively. The second round uses the poses from the end of the first round as initialization, and the camera optimizer's learning rate begins at 0.0005 and reduces to 0.00001. Other parameters remain constant. Optimizations typically take 2.5 minutes for every 10k iterations, regardless of the number of images.

More experimental setup.

- We denote DROID-SLAM* as the variant that first calibrates intrinsics using GeoCalib [80] on the first image of each sequence and then uses the calibrated parameters in DROID-SLAM.
- All experiments are conducted on an NVIDIA RTX 4090 GPU; To align with V100-based results, runtimes are scaled by a factor of 1.5, reflecting the measured FP16 inference speed gap. For anchor frame selection, we use 50 frames per scene in 7 scenes, with PSNR reported in the combined training and test sets, and the relocalization accuracy evaluated on the test set after ACE0 [9]. For mip-NeRF 360, 50 anchors are selected per sequence. For Tanks and Temples, we select an average of 100 keyframes per sequence to relocalize all remaining images and video frames. For TUM RGB-D, we use 50–100 anchors depending on sequence length: floor, plant, and teddy use 100 frames, and others use 50.
- In cases where the first frame contains limited semantic information, it is replaced with a semantically richer frame as the first anchor.
- For visualization in Fig. 1, we remove points in the lowest 50% confidence on the depth confidence map, corresponding to sky, glass, and other ambiguous surfaces, and apply moderate point cloud downsampling to enhance visual clarity.
- We cite the results in Tabs. 3, 4, 5, 9, 10 from ACE0 [9]. Details on default parameters and configurations for base-

lines such as COLMAP (default), COLMAP (Sparse + Reloc + BA) and Nope-NeRF are available in the supplementary material of ACE0 [9].

7. Additional Results

7.1. Pose Estimation

TUM RGBD. We report the root mean square error (RMSE) of the absolute trajectory error (ATE), comparing our method with a broader set of state-of-the-art approaches [11, 14, 41, 45, 74, 75, 99] under calibrated settings, as summarized in Tab. 8. Our method achieves accuracy on par with the most advanced SLAM systems while remaining robust across diverse sequences without requiring camera calibration. Compared with geometry-based pipelines, such as ORB-SLAM3, our approach exhibits stronger robustness and achieves comparable or superior accuracy to learning-based baselines. The main weakness appears in the floor sequence, where images contain limited visual cues dominated by textureless floor regions. In this case, reference view selection becomes critical: large viewpoint gaps between the query and reference views significantly degrade localization. We further visualize these effects in the trajectory results (Sec. 8).

7.2. Novel View Synthesis

Tanks & Temples. To further evaluate our scalability for large-scale reconstruction, we apply our method to the Tanks & Temples [32] video sequences. For each sequence, we uniformly sample 100 images as anchors and perform localization on all frames.

For clarity, Table 9 reports the results on both the image set and the full video sequences, with the latter shown on the right. As a reference, we include *Sparse COLMAP + Reloc + BA* (CMP (SRB)), which initializes from a sparse COLMAP reconstruction using 150–500 images, registers the remaining frames and performs global bundle adjustment. Our approach consistently outperforms RealityCapture, DROID-SLAM [75], and ACE0 [9] across all splits, with only ACE0 initialized from sparse COLMAP poses (CMP + ACE0) achieving comparable performance. In particular, on the most challenging 'advanced' split, our method achieves the highest PSNR of all methods. Despite each sequence containing more than 10k images on average, our feedforward approach maintains competitive efficiency - only slightly slower than SLAM-based pipelines - while delivering strong reconstruction quality.

7 Scenes. We quote the table from ACE0 [9] and report our results in Tab. 10. For each 7-Scenes sequence, we uniformly sample 50 frames from the train/test splits and estimate poses for all images in the scene. We compare against COLMAP since bundle-adjusted COLMAP poses provide a more accurate reference. Our method attains PSNR comparable to the COLMAP reference and exceeds ACE0. In terms of runtime, even under "fast" settings COLMAP still requires around 13 h per scene; DROID-SLAM returns results quickly but performs poorly on 7-Scenes; ACE0 takes 1 h. In contrast, our approach finishes in 25 min without any pose initialization, achieves higher PSNR than ACE0, and matches the PSNR of ACE0 when initialized from KinectFusion (KF-Init., 7 min).

To further compare against learning-based approaches under constrained memory budgets, we follows [9] to downsample each sequence to 200 frames. Sequential-dependent scene regression models (e.g., Cut3R [86], SLAM3R [42]) require dense temporal input, and VGGT [85] still exceeds memory limits on 200 images. We therefore compare to BARF [37] and NoPe-NeRF [5]: Both BARF [37] and Nope-NeRF [5] fail to recover the scene after a long fitting time. Using our localization of all frames, we consistently obtain the highest PSNR in this 200-frame setting. while remaining faster than these baselines.

8. Visualization

As shown in Fig. 4, 5 and 6, we show the render images of test view in the three different splits of Tank & Temple dataset. We illustrate the test view of *7-Scenes* and *Mip-NeRF 360* dataset in Fig. 7 and 8, respectively. We aslo supplement in Fig. 9 our regressed camera poses and point clouds.

	Method	Sequence									Avg
		`360`	`desk`	`desk2`	`floor`	`plant`	`room`	`rpy`	`teddy`	`xyz`	
Calib.	ORB-SLAM3 [11]	×	0.017	0.210	×	0.034	×	×	×	**0.009**	N/A
	DeepV2D [74]	0.243	0.166	0.379	1.653	0.203	0.246	0.105	0.316	0.064	0.375
	DeepFactors [14]	0.159	0.170	0.253	0.169	0.305	0.364	0.043	0.601	0.035	0.233
	DPV-SLAM [41]	0.112	0.018	0.029	0.057	0.021	0.330	0.030	0.084	0.010	0.076
	DPV-SLAM++ [41]	0.132	0.018	0.029	0.050	0.022	0.096	0.032	0.098	0.010	0.054
	GO-SLAM [99]	0.089	**0.016**	0.028	0.025	0.026	0.052	**0.019**	0.048	0.010	0.035
	DROID-SLAM [75]	0.111	0.018	0.042	**0.021**	**0.016**	**0.049**	0.026	0.048	0.012	0.038
	MASt3R-SLAM [45]	**0.049**	**0.016**	**0.024**	0.025	0.020	0.061	0.027	**0.041**	**0.009**	**0.030**
Uncalib.	DROID-SLAM* [75]	0.202	0.032	0.091	0.064	0.045	0.918	0.056	0.045	0.012	0.158
	MASt3R-SLAM* [45]	0.070	0.035	0.055	0.056	0.035	0.118	0.041	0.114	0.020	0.060
	VGGT-SLAM (Sim(3)) [43]	0.123	0.040	0.055	0.254	0.022	0.088	0.041	0.032	0.016	0.074
	VGGT-SLAM (SL(4)) [43]	0.071	0.025	0.040	0.141	0.023	0.102	0.030	0.034	0.014	0.053
	SAIL-Recon (Offline)	0.070	0.024	0.042	0.107	0.031	0.113	0.020	0.037	0.012	0.051

Table 8. **Root mean square error (RMSE) of absolute trajectory error (ATE) on TUM RGB-D [66] (unit: m)**. Gray rows denote results obtained with calibrated camera intrinsics, while entries marked with * indicate evaluation in the uncalibrated setting. We color result in: Best, Second, and Third. Note that our method is actually a offline SfM method.

		Frames	CMP (D)	Reality Capture	DROID-SLAM† [75]	ACE0	Ours	Frames	CMP (SRB)	CMP+ ACE0	Reality Capture	DROID-SLAM† [75]	ACE0	Ours
Training	Barn	410	24.0	21.2	19.0	16.5	23.5	19.3k	26.3	25.1	16.9	13.5	17.7	25.1
	Catpr.	383	17.1	15.9	16.6	16.9	16.8	11.4k	18.7	18.8	17.9	18.9	18.6	17.5
	Church	507	18.3	17.6	14.3	17.2	17.0	19.3k	18.5	17.3	-	11.5	16.5	15.8
	Ignatius	264	20.1	17.7	17.8	19.8	19.5	7.8k	20.9	20.7	18.6	19.1	20.7	20.7
	MtgRm.	371	18.6	18.1	15.6	18.0	19.5	11.1k	20.8	20.3	18.2	17.1	16.6	20.4
	Truck	251	21.1	19.0	18.3	20.1	20.9	7.5k	23.4	23.1	19.1	20.6	23.0	23.5
	Average	364	19.9	18.2	16.9	18.1	19.5	14.6k	21.4	20.9	18.2	16.8	18.9	20.5
	Time		1h	3min	5min	1.1h	3.5min		8h	1.8h	14h	18min	2.2h	58min
Intermediate	Family	152	19.5	18.8	17.6	19.0	20.6	4.4k	21.3	21.3	19.8	19.8	18.0	21.3
	Francis	302	21.6	20.7	20.7	20.1	21.8	7.8k	22.5	22.7	20.4	21.8	21.7	22.8
	Horse	151	19.2	19.0	16.3	19.5	20.1	6.0k	22.6	22.3	20.7	19.2	21.7	21.9
	LightH.	309	16.6	16.5	13.6	17.5	18.2	8.3k	19.5	20.5	16.6	18.9	18.6	19.7
	PlayGd.	307	19.1	19.2	11.4	18.7	20.3	7.7k	21.2	21.0	16.5	11.3	20.4	21.7
	Train	301	16.8	15.4	13.8	16.2	16.2	12.6k	19.8	18.5	14.4	15.6	18.5	18.5
	Average	254	18.8	18.3	15.6	18.5	19.5	7.8k	21.1	21.0	18.1	17.8	19.8	21.0
	Time		32min	2min	3min	1.3h	3min		5h	1h	11h	14min	2.2h	30min
Advanced	Audtrm.	302	19.6	12.2	16.7	18.7	20.3	13.6k	21.4	19.8	-	16.6	20.0	21.0
	BallRm.	324	16.3	18.3	13.1	17.9	14.8	10.8k	18.0	15.6	-	10.4	18.9	16.9
	CortRm.	301	18.2	17.2	12.3	17.1	17.4	12.6k	18.7	17.8	-	10.2	16.3	17.4
	Palace	509	14.2	11.7	10.8	10.7	14.3	21.9k	15.3	12.3	-	18.6	11.0	13.3
	Temple	302	18.1	15.7	11.8	9.7	17.8	17.5k	19.6	16.1	-	11.9	14.8	18.3
	Average	348	17.3	15.0	12.9	14.8	16.9	15.6k	18.6	16.3	-	11.5	16.2	17.4
	Time		1h	2min	4min	1h	3.5min		10h	2.1h		27min	2.8h	59min

Table 9. **Tanks & Temples.** We show the pose accuracy via view synthesis with Nerfacto [69] as PSNR in dB, and the reconstruction time. We color code in: Best, Second, and Third. †Method needs sequential inputs.

	Frames	Pseudo GT			All Frames				200 Frames			
		Kinect Fusion	COLMAP (default)	COLMAP (fast)	DROID-SLAM†	ACE0 (default)	ACE0 (KF Init.)	Ours (50F)	BARF [37]	NoPE-NeRF	ACE0 (default)	Ours (50F)
Chess	6k	19.6	23.6	23.5	19.3	23.3	23.0	23.4	12.8	12.6	22.7	21.8
Fire	4k	19.2	22.6	22.6	13.0	22.3	22.3	22.8	12.7	11.8	22.1	24.4
Heads	2k	17.0	18.8	18.9	17.6	18.8	19.1	18.5	10.7	11.8	19.9	20.2
Office	10k	18.9	21.4	21.6	failed	21.1	21.5	20.9	11.9	10.9	19.8	19.4
Pumkin	6k	19.9	24.1	23.8	18.3	24.1	23.8	24.5	19.6	14.2	24.7	25.0
RedKitchen	12k	17.6	21.4	21.4	10.9	20.8	20.9	19.9	11.6	11.2	18.9	20.0
Stairs	3k	19.0	16.7	21.0	13	17.7	19.9	20.6	15.8	15.9	18.8	20.8
Average	6.5k	18.7	21.2	21.8	N/A	21.2	21.5	21.5	13.6	12.6	21.0	21.8
Avg. Time		realtime	38h	13h	18min	1h	7min	25min	8.5h	47h	27min	3min

Table 10. **7-Scenes.** We show the pose accuracy via view synthesis with Nerfacto [69] as PSNR in dB, and the reconstruction time. We color code in: Best, Second, and Third. Our method takes 50 frames as anchor images only, achieves SOTA performance. For some competitors, we had to sub-sample the images due to their computational complexity (right side). †Method needs sequential inputs.

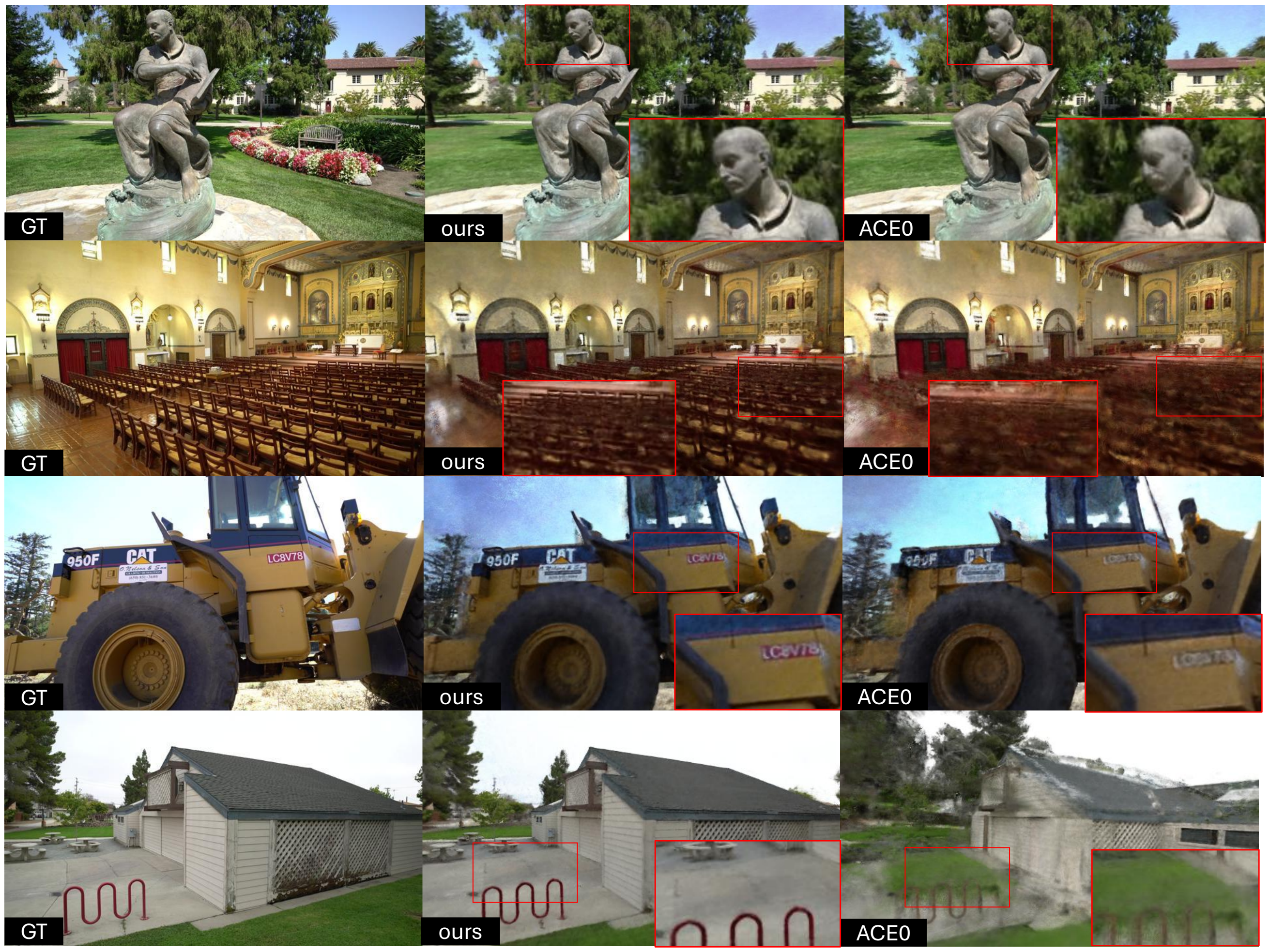

Figure 4. **Visualization on Tank & Temple *training* split.**

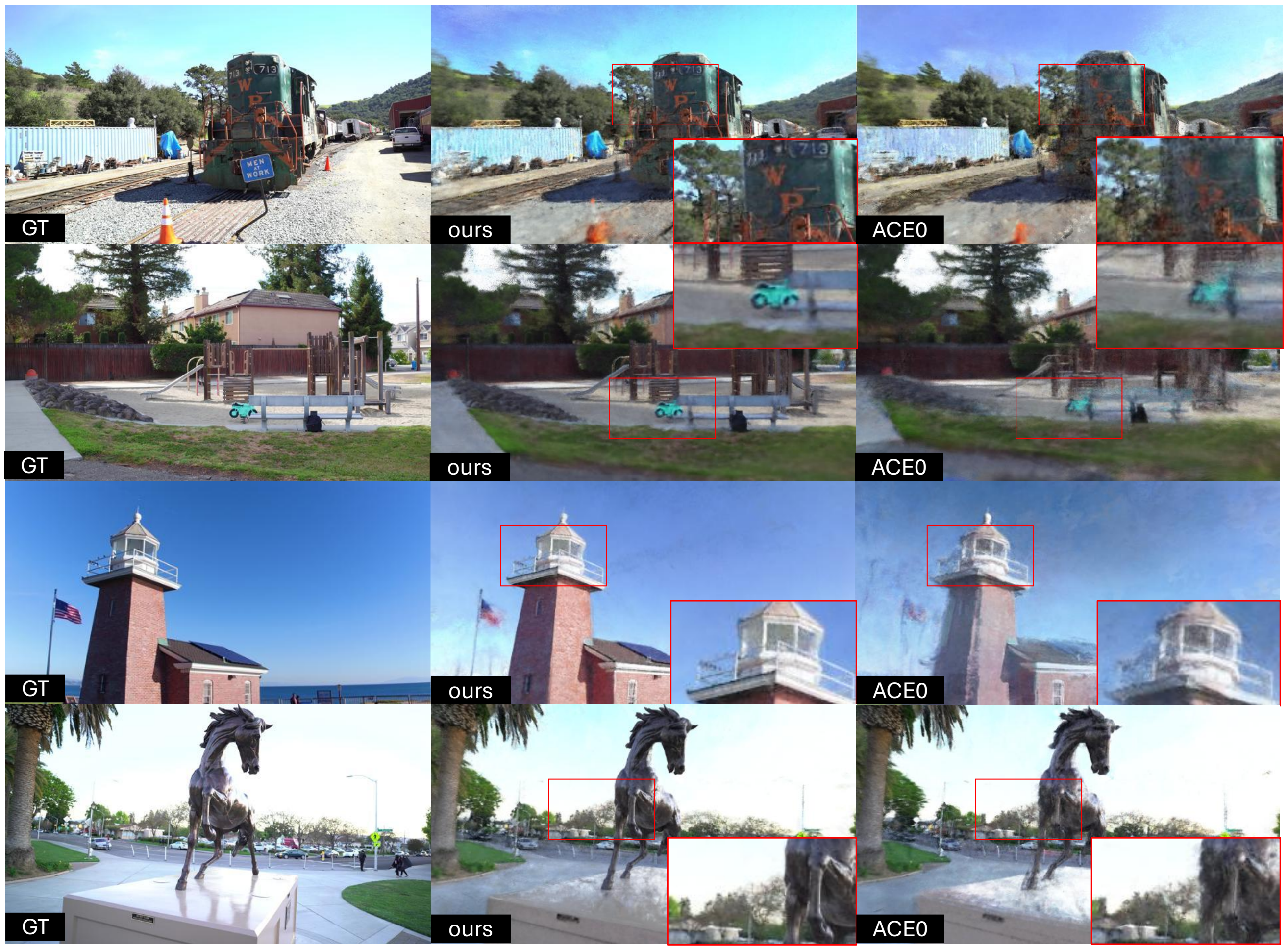

Figure 5. **Visualization on Tank & Temple *intermediate* split.**

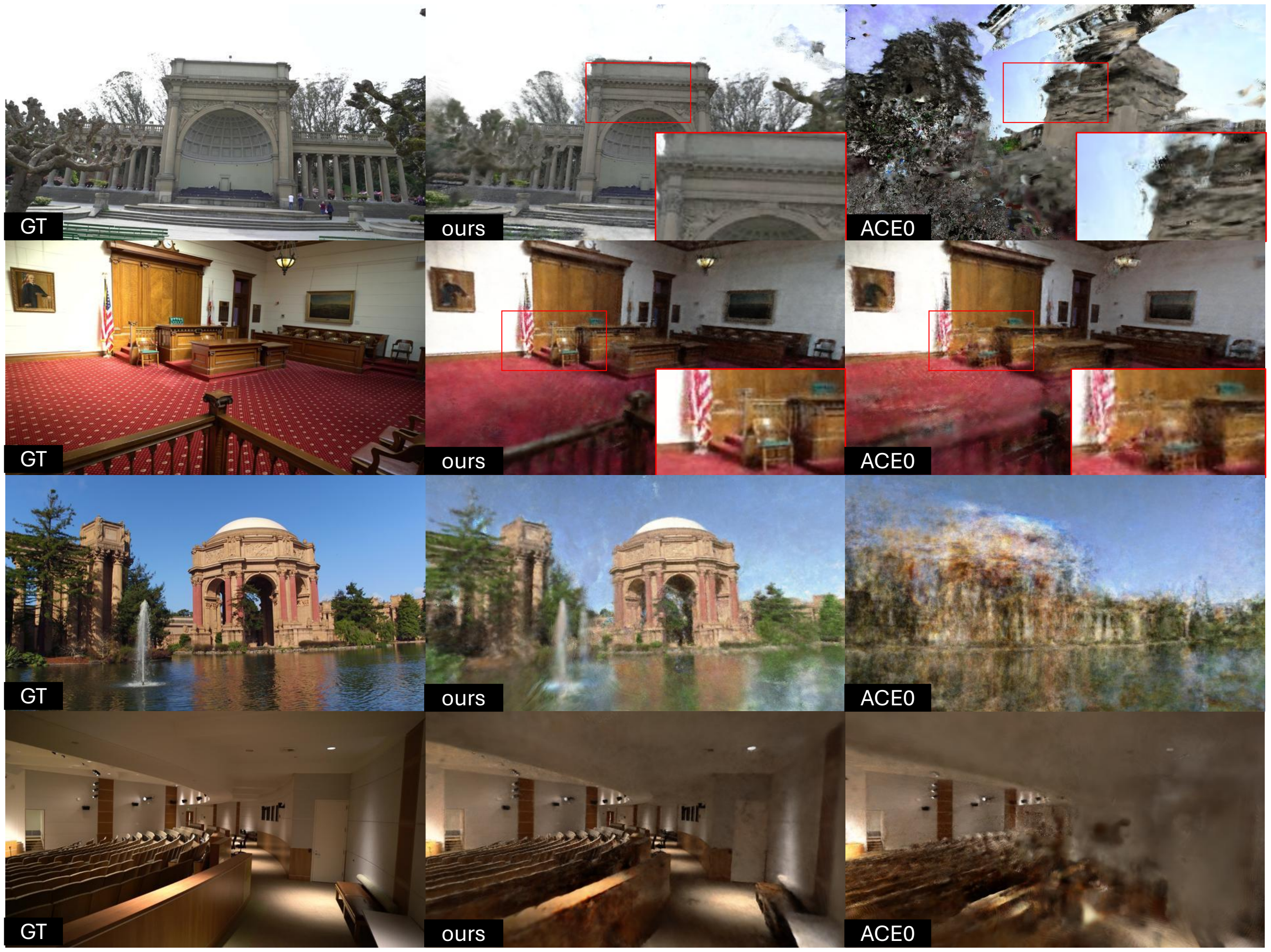

Figure 6. **Visualization on Tank & Temple *advanced* split.**

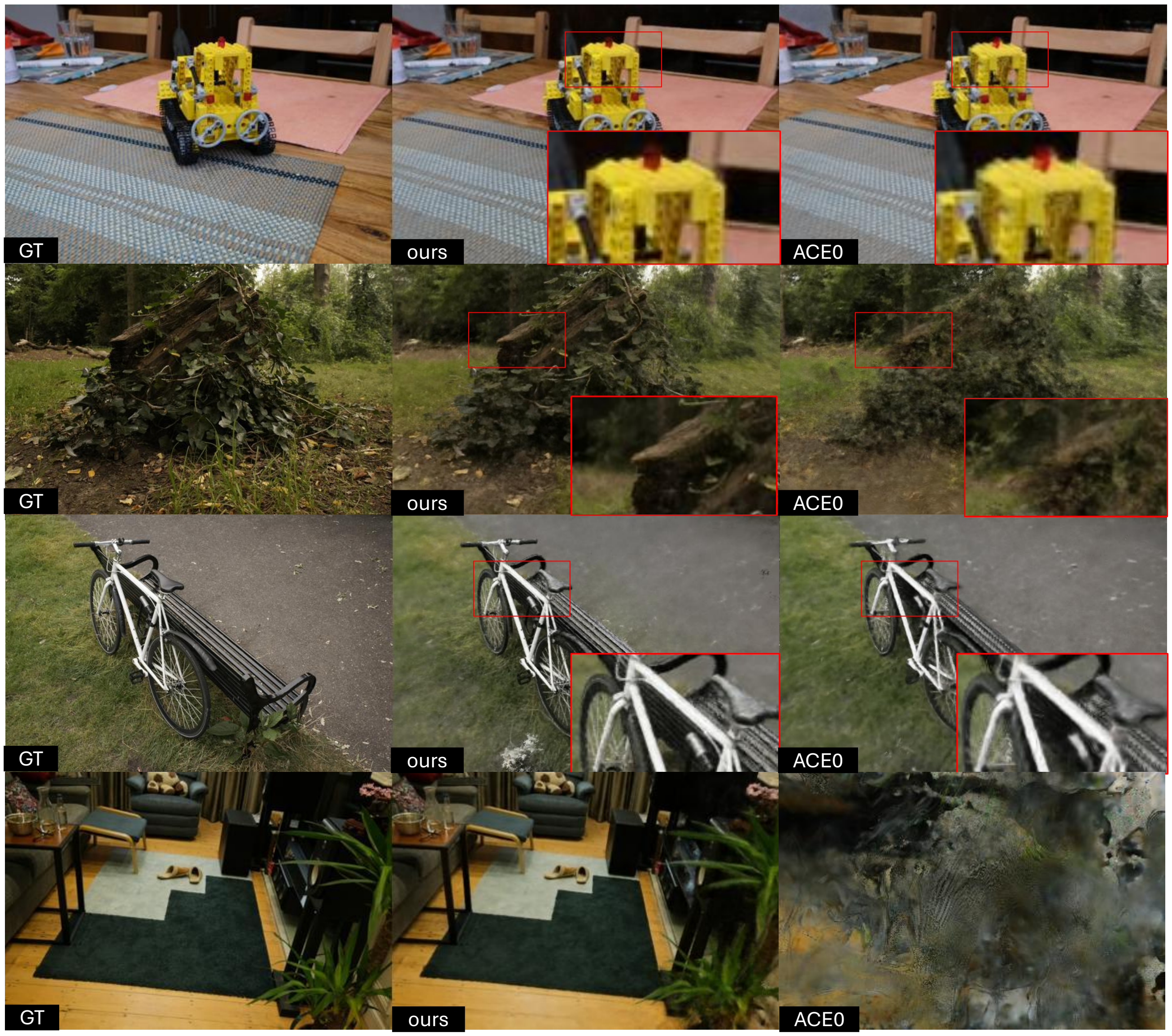

Figure 7. **Visualization on Mip-NeRF 360 dataset.**

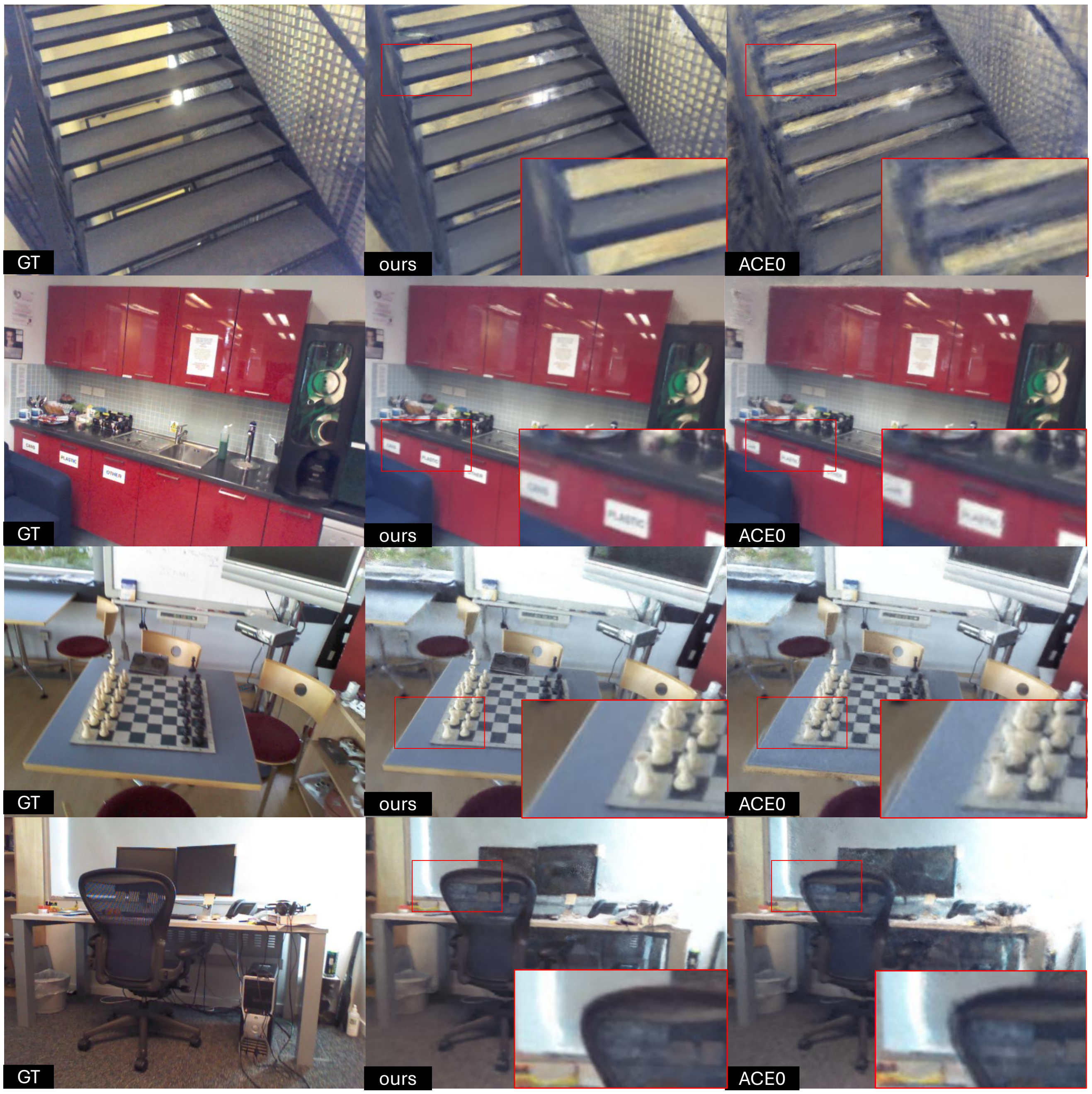

Figure 8. **Visualization on 7-Scenes dataset.**

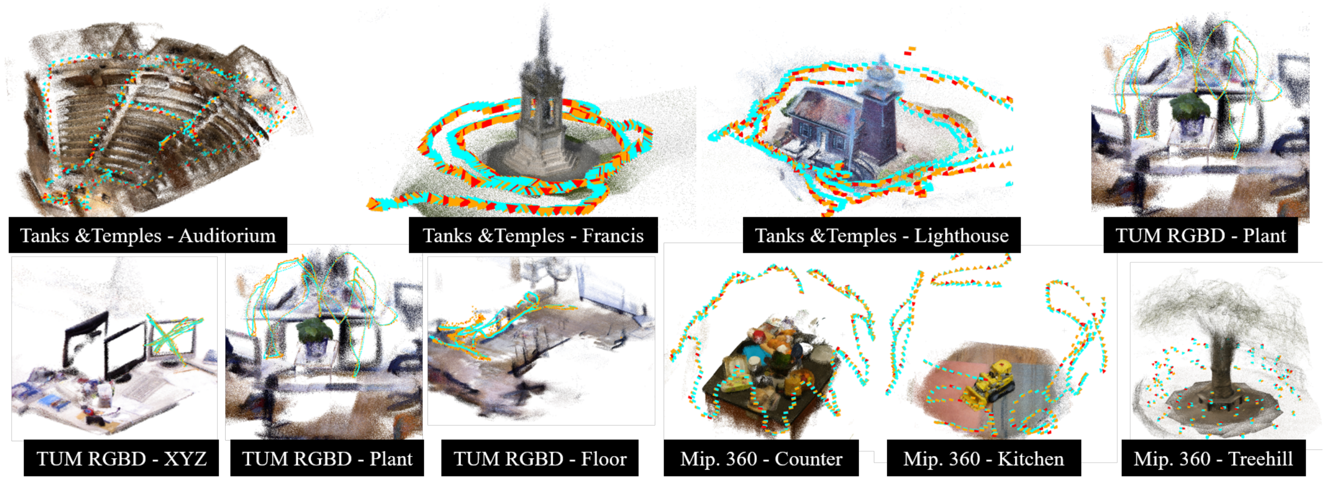

Figure 9. **Regressed Camera Poses and Point Clouds.** We visualize the camera poses and point clouds predicted by SAIL-Recon across various datasets. COLMAP or ground-truth camera poses are shown as blue frustums, while regressed camera poses are shown in yellow, with red indicating anchor images.