Glob3R: Global Structure-from-Motion with 3D Foundation Models

Junyuan Deng[1,2]* **Heng Li**[1]* **Kejie Qiu**[2]† **Lingteng Qiu**[2] **Rui Peng**[2]
Weichao Shen[2] **Weihao Yuan**[3] **Siyu Zhu**[4] **Zilong Dong**[2] **Ping Tan**[1]†

[1]The Hong Kong University of Science and Technology
[2]Alibaba Token Hub, Alibaba Group [3]Nanjing University [4]Fudan University

Abstract

Recent 3D geometric foundation models, such as VGGT, provide robust feed-forward 3D reconstruction by directly predicting camera poses and 3D scene points from input images. However, their results remain inaccurate, and scaling them to long sequences or large unordered image sets typically requires chunk-wise processing, which can introduce drift and inconsistency. We present Glob3R, a global SfM-style reconstruction built on 3D foundation models. Our key idea is to explicitly optimize feed-forward geometric predictions. To this end, we augment a frozen Pi3X backbone with a lightweight dense matching head that predicts image warps between selected reference frames and neighboring views. These dense warps are converted into sparse but reliable multi-view feature tracks, which provide correspondence constraints for global optimization. We further introduce a keyframe-based sliding-window association strategy that propagates tracks and relative poses across overlapping windows, enabling scalable reconstruction. Finally, we perform global motion averaging and bundle adjustment to refine camera poses, reduce scale inconsistencies, and recover dense scene geometry. Extensive experiments on indoor, outdoor, large-scale driving, and unordered SfM benchmarks demonstrate that Glob3R achieves robust and accurate reconstruction. It consistently improves over feed-forward foundation-model baselines and recent scalable reconstruction methods, while being more robust than classical SfM pipelines. The refined poses also lead to higher-quality neural rendering, validating the benefit of combining foundation-model priors with global geometric optimization.

1 Introduction

Reconstructing 3D scenes from image collections remains a fundamental challenge in computer vision, serving as a cornerstone for applications such as augmented reality (AR), robotics, autonomous navigation, and neural rendering. Recently, learning-based 3D geometric foundation models, including DUSt3R [65], VGGT [61], Pi3X [67], and other recent methods [26, 50, 13, 19, 9, 74], have emerged as a new reconstruction paradigm. Departing from traditional pipelines, these models directly predict camera poses and dense per-pixel geometry, namely depth or scene coordinates, from arbitrary image sets in a feed-forward manner. This capability offers a robust and highly efficient initialization for 3D reconstruction from both ordered image sequences and unordered image collections. Despite these advances, existing 3D geometric foundation models still suffer from limited accuracy and scalability. Their feed-forward predictions provide strong global priors, but

*Equal contribution.
†Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

the estimated camera poses and scales are often only approximately correct, which limits their use in high-fidelity applications such as Neural Radiance Fields (NeRF) [39] and other view-synthesis pipelines. A key reason is that many models are trained with Structure from Motion (SfM)-derived supervision, where poses and geometry are generated by tools such as COLMAP [47] rather than measured ground truth, thereby transferring the noise and bias of the reconstruction pipeline to the learned predictions. Meanwhile, GPU memory constraints make it difficult to process long sequences or large image collections in a single forward pass. Recent VGGT-based methods [37, 36, 73, 11] address this by splitting input into chunks and aligning chunk-level predictions with simple $\mathrm{SE}(3)$ or $\mathrm{Sim}(3)$ transformations, but this stitching strategy provides limited cross-chunk constraints and can accumulate pose and scale errors. Other approaches based on test-time training, recurrent memory, or sequence-specific architectures [64, 72, 78, 8] improve scalability, but often require backbone retraining or assume sequential inputs, reducing their flexibility in unordered image collections.

Compared with learning-based feed-forward predictions, classical SfM pipelines can often achieve higher pose accuracy by explicitly establishing correspondences and refining geometry with optimization. Their accuracy mainly comes from explicit geometric constraints, such as triangulation and bundle adjustment (BA). Existing SfM pipelines are commonly divided into incremental and global paradigms. Incremental SfM, such as COLMAP [47], registers images one by one and repeatedly applies BA, making it robust to noisy pairwise geometry but computationally expensive and sensitive to registration order. Global SfM, such as GLOMAP [41], instead builds a pose graph from pairwise relative motions and estimates all camera poses jointly through rotation and translation averaging followed by BA, making it substantially faster than incremental reconstruction. However, its performance still depends on the quality of the pose graph, and unreliable correspondences or outlier relative poses can make the reconstruction brittle in challenging scenarios.

These observations suggest a natural direction: using 3D foundation models to predict multi-view correspondences together with relative motion and dense geometry, and then adopting global SfM principles to optimize the reconstruction. This combination is well-suited to the limitations of both paradigms. The foundation model provides robust geometric initialization, including camera poses, point maps, and confidence estimates, which enables the construction of a reliable pose graph. More importantly, its intermediate features encode cross-view geometric and structural cues, making them effective for correspondence prediction and reducing outlier matches. Given these correspondences and the resulting pose graph, global pose estimation and bundle adjustment can jointly refine camera poses and scene structure, substantially improving reconstruction accuracy.

Specifically, we build our framework on top of Pi3X [67] and introduce a dense matching head that predicts image warps and confidence between selected frames and other views. The dense warps are converted into sparse but reliable multi-view tracks, providing explicit correspondence constraints for global optimization. To efficiently select keyframes and associate long sequences, we further propose a keyframe-based sliding-window strategy that leverages the predicted local point maps, camera poses, and confidence maps. Instead of merely stitching independent chunks, our method uses overlapping windows to propagate tracks and relative poses across the full sequence, while performing subsequent optimization at the frame level. These local predictions and track associations are first converted into an initial global pose graph. We then perform motion averaging (including rotation and translation averaging) followed by bundle adjustment to reduce scale inconsistencies, refine camera poses, and recover dense geometry from the optimized reconstruction. Extensive experiments across various benchmarks demonstrate the effectiveness of our pipeline. Our method improves novel-view synthesis by 2–3 dB PSNR over feed-forward baselines and about 1 dB over COLMAP-based poses. On KITTI, it reduces trajectory RMSE by 10%–50% compared with recent streaming methods [72, 4, 78]. On ETH3D, it substantially improves rotation accuracy and nearly doubles translation accuracy over the latest learning-based SfM baseline [60].

Our main contributions are summarized as follows:

- We propose a foundation-model-guided framework that turns feed-forward 3D predictions into optimizable geometric constraints, combining robust learned priors with global SfM refinement.

- We introduce a dense warping module and a keyframe-based sliding-window association strategy to extract reliable feature tracks across long sequences and unordered image collections.

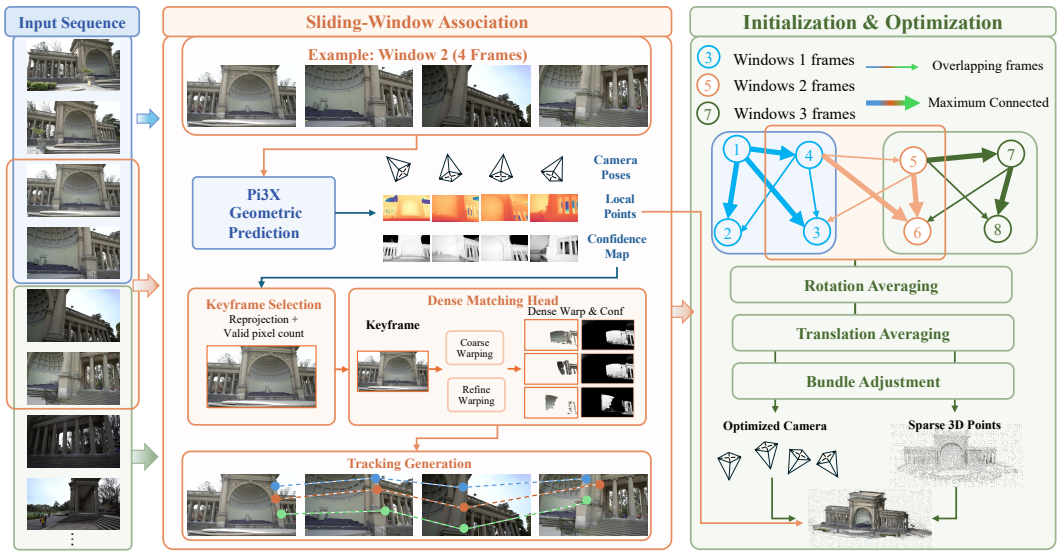

Figure 1: **Overview of the proposed reconstruction framework.** Given an ordered image sequence or a retrieval-based pseudo-sequence, we first predict geometric priors and dense warps within windows. Keyframes are selected according to reprojection coverage, and dense warps are converted into sparse multi-view tracks. These tracks are then merged into a global association graph for rotation averaging, translation averaging, bundle adjustment, and dense reconstruction.

- Extensive experiments on indoor, outdoor, large-scale driving, and unordered SfM benchmarks demonstrate that our method significantly improves geometric accuracy over previous methods.

2 Related Works

2.1 Structure-from-Motion and Geometric Optimization

Structure-from-Motion (SfM) is a fundamental problem in computer vision. Incremental SfM, such as COLMAP [48], progressively registers images starting from an initial image pair. In contrast, global SfM methods, such as GLOMAP [41] and GLUEMAP [42], estimate pairwise relative poses and jointly recover all camera poses via rotation averaging [22, 38, 28], translation averaging [21, 69], or global positioning [41], followed by bundle adjustment [59]. Despite their high reconstruction accuracy, these optimization-based methods remain limited by the cost and reliability of feature matching, especially in large-scale scenes, weakly textured regions, repetitive structures, or forward-moving trajectories. Recent differentiable SfM [55, 68, 56, 57, 2, 51, 63] methods further explore learnable reconstruction by enforcing geometric constraints and minimizing reprojection or photometric errors.

2.2 3D Foundation Models

3D foundation models have recently emerged as a feed-forward paradigm for 3D reconstruction. DUSt3R [65] predicts dense scene coordinate maps for a pair of input views, while VGGT [61], VGGT-Ω [62], Dens3R [18], Pi3 [67], and DA3 [32] predict camera, depth, and other 3D attributes with a single transformer network. These efficient and robust models form part of broader 3D vision research on reconstruction, part segmentation [12], geometry enhancement [10], and multi-view 3D generation [35]. However, their geometric accuracy is often bounded by SfM-derived supervision such as COLMAP poses, and scaling them to large image collections remains challenging. Recent methods such as VGGT-Long [11] and VGGT-SLAM [37] process long sequences by splitting them into chunks and estimating transformations between chunks, but such strategies underuse intra-chunk geometric relations and can accumulate errors across chunk. Some methods [31, 70, 5, 64, 73] maintain a memory bank to extend reconstruction to larger scenes, but the accumulated memory updates can still lead to substantial drift as the scene scale increases. Other approaches improve scalability through test-time training, recurrent memory, or sequence-specific architectures [72, 78, 4], but often require backbone retraining or assume sequential inputs.

2.3 Feature Matching and Dense Correspondence

Feature matching is crucial for both SfM and SLAM. Classical sparse pipelines detect keypoints and then establish correspondences by matching local descriptors, while learned matchers such as SuperGlue [46] and LightGlue [33] improve matching quality through attention-based reasoning. Detector-free methods, including LoFTR [53], DKM [15], and RoMa [17], move toward semi-dense or dense correspondence estimation and are more robust in weakly textured regions. Recently, RoMa v2 [16] further formulates two-view dense matching as image warping, achieving strong matching accuracy. Feature matching has also been explored in 3D foundation models: MASt3R [40] predicts dense descriptors on top of DUSt3R, and VGGT [61] tracks query points across views. Inspired by these works, we construct multi-view correspondences through dense warping and derive accurate feature tracks for subsequent global pose optimization.

3 Method

Fig. 1 illustrates the pipeline of our method. Given an image collection, our goal is to estimate globally consistent camera poses and recover scene geometry. Sec. 3.1 introduces our network architecture based on Pi3X: images are processed by a frozen 3D foundation model to obtain coarse geometric priors, including camera poses, local point maps, and confidence maps. For predicting multi-view feature matching, we further introduce a dense matching head that predicts image warps and confidence between selected keyframes and other frames. Sec. 3.2 describes the sliding-window association strategy, which selects keyframes, converts dense warps into sparse multi-view tracks, and connects local windows into a pose graph. Sec. 3.3 presents the global optimization stage, including motion averaging, bundle adjustment, and dense reconstruction.

3.1 Architecture

We build our model on top of Pi3X [67], which adopts a unified transformer backbone with multiple prediction heads f to infer geometric information from an image set $\mathcal{I} = \left\{ I_i \in \mathbb{R}^{3\times H\times W} \right\}_{i=1}^{N}$:

$$f(\{I_i\}_{i=1}^{N}) = \{\mathbf{T}_i, \mathbf{X}_i, \mathbf{C}_i, m_i\} \tag{1}$$

where for input image I_i, $\mathbf{T}_i \in SE(3) \subset \mathbb{R}^{4\times 4}$ denotes the camera pose, $\mathbf{X}_i \in \mathbb{R}^{3\times H\times W}$ is the 3D point map in the coordinate system of image I_i, $\mathbf{C}_i \in \mathbb{R}^{H\times W}$ is the corresponding confidence map, and m_i denotes the approximate metric scale. We follow the permutation-equivariant design of Pi3X, which removes the need for a designated reference view and enables more flexible multi-view inputs.

To equip the model with dense matching capability, VGGT [61] takes a query point in a reference image as input and employs a tracking head to predict its corresponding points across all other images. Despite its effectiveness, the memory cost grows rapidly with the number of query points, and the resulting point-wise correspondences are less spatially connected than dense matching. Other methods, such as MASt3R [40], predict dense feature descriptors and confidence maps for every pixel, followed by nearest-neighbor search for correspondence estimation. However, dense descriptor prediction is computationally expensive and can be less accurate. Inspired by RoMa V2 [16], we formulate dense matching as image warping, which maps 2D pixels from one image to another through a two-stage process consisting of coarse matching and subsequent refinement. Visualizations of the warping results are available in the supplementary material. Specifically, let $\mathbf{H} \in \mathbb{R}^{N\times L\times C}$ denote the output feature of the Pi3X backbone, where N is the number of input images, L is the token length, and C is the feature dimension. Given a reference image I_a, we treat the remaining images as target views $\mathcal{B} = \{1, \cdots, N\} \setminus \{a\}$. The coarse matching results are predicted by a transformer decoder ($\mathrm{Dec}_{\mathrm{match}}$) followed by a Dense Prediction Transformer ($\mathrm{DPT}_{\mathrm{match}}$):

$$\left(\mathbf{W}^{a\to\mathcal{B}}, \mathbf{p}^{a\to\mathcal{B}}\right) = \mathrm{DPT}_{\mathrm{match}}\left(\mathrm{Dec}_{\mathrm{match}}(\mathbf{H}), a\right), \tag{2}$$

where $\mathbf{W}^{a\to\mathcal{B}} \in \mathbb{R}^{(N-1)\times 2\times \frac{H}{4}\times \frac{W}{4}}$ maps pixels from the reference image to each target image, and $\mathbf{p}^{a\to\mathcal{B}} \in \mathbb{R}^{(N-1)\times \frac{H}{4}\times \frac{W}{4}}$ denotes the corresponding confidence. Both outputs are predicted at a stride of 4 and then refined to full resolution. A refinement module takes the coarse matching results and the original image as input, and progressively predicts residual warps ΔW and confidence residuals Δp at strides $\{4, 2, 1\}$. More details are provided in the supplementary material.

Loss Function and Training. We freeze the encoder, decoder, and existing prediction heads, and fine-tune only the matching head. Since Pi3X is permutation-equivariant, the reference view can be

chosen flexibly during inference, which is important for our subsequence-based pipeline. During training, we simply use the first image as the reference view and supervise its matches to the remaining target images with ground-truth 2D–2D warp. The ground-truth warp is generated by projecting the ground-truth 3D point map with known camera poses, and the confidence supervision is derived from depth consistency. The total loss is the weighted sum of the negative log-likelihood of the best matching patch, the warping loss, and the confidence loss:

$$\mathcal{L} = \sum_{k=2}^{N} \lambda_1 \mathcal{L}_{\mathrm{NLL}}^{k} + \lambda_2 \mathcal{L}_{\mathrm{warp}}^{k} + \lambda_3 \mathcal{L}_{\mathrm{conf}}^{k} \tag{3}$$

where $\mathcal{L}_{\mathrm{NLL}}^{k}$ encourages high cosine similarity between the features of matching patches. Please refer to the supplementary material for more details.

3.2 Sliding-window Sequence Association

Eq. 1 provides a robust geometric prior such as camera pose and point map, while Eq. 2 gives image warps between frames. They can connect a pose graph through sparse but reliable multi-view tracks.

Sequence and Window Construction. Our method operates on an image sequence $\mathcal{S} = \{I_i\}_{i=1}^{M}$. For data with explicit temporal order, we use the original order. For unordered image sets, we build a pseudo-sequence with an image retrieval model such as SALAD [25]. This fits our framework because the network does not require strictly ordered frames. Any frame in the window with sufficient visual overlap can be the reference view. We process the sequence with sliding windows, where each window contains N frames and is shifted by $N/2$ frames along the sequence. The resulting half-window overlap provides shared frames between adjacent windows, enabling correspondences and relative poses to be propagated progressively across the full sequence.

Keyframe Selection. Following the principle of classical SLAM systems, we use a small number of keyframes to establish local mappings, while the remaining frames are associated through the matches to the keyframes. For each window, we initialize its first frame as a keyframe if there is no keyframe. We then run Eq. 1 to obtain the camera pose $\mathbf{T}_t$, the point map $\mathbf{X}_t$ applying the predicted metric scale, and the confidence map $\mathbf{C}_t$ for every frame. For a candidate frame I_t, we reproject its point map onto each existing keyframe $I_r \in \mathcal{K}$ and count the number of valid projected pixels:

$$n_t = \max_{I_r \in \mathcal{K}} \sum_{\mathbf{u}} \mathbf{1}\left[\pi\left(\mathbf{T}_{t\to r}\bar{\mathbf{X}}_t(\mathbf{u})\right) \in \mathcal{D},\ z_{t\to r}(\mathbf{u}) > 0,\ \mathbf{C}_t(\mathbf{u}) > \tau_c\right], \tag{4}$$

where $\mathcal{K}$ is the current keyframe set, $\bar{\mathbf{X}}_t(\mathbf{u})$ denotes the homogeneous 3D point at pixel $\mathbf{u}$, $\mathbf{T}_{t\to r}$ is the relative transform from I_t to I_r, $\pi(\cdot)$ is the projection function, $\mathcal{D}$ is the image domain, $z_{t\to r}(\mathbf{u})$ is the projected depth, and τ_c is the depth confidence threshold. If $n_t < \tau_{\mathrm{proj}}$, we add I_t to the keyframe set. This process is repeated until all frames in the window have been visited. We additionally ensure that at least one keyframe lies in the overlapping half of the window to maintain connectivity between consecutive windows.

Track-based Global Initialization. After keyframe selection, we use the matching head to predict dense warps from each keyframe to all frames in the same window. This step only requires a single forward pass of the frozen backbone to extract features, after which the backbone features are reused and kept fixed for keyframe-to-frame matching, resulting in moderate memory overhead. We sample K high confidence pixels from the keyframe point map, and pick the corresponding pixel on the dense warp map of other frames to build the track. The tracking confidence is obtained by combining the keyframe point confidence and the predicted warp confidence. This produces sparse yet reliable multi-view tracks anchored at keyframes.

The local tracks from the overlapping windows are merged into a pose graph. Each frame is treated as a node, and an edge is added between two frames if they share valid tracking correspondences. The edge weight is defined by the number of valid tracks between the two frames. Since the matching results and local geometry predictions are produced jointly within each window, each valid edge is associated with a relative pose. For edges from different windows, we temporarily use the metric scale predicted by Pi3X [67] to bring their relative transformation into global space. We then compute a maximum spanning tree over the graph and initialize the camera pose with relative poses on the spanning tree. The initialized poses and the associated sparse tracks are then passed to the subsequent optimization stage, where remaining scale inconsistencies and coarse poses are further refined.

This design differs from window-stitching pipelines [11, 37], which mainly merge independent chunk-level predictions. By using frames as the basic association and optimization unit, our method not only connects different windows through overlapping keyframes, but also enables pose refinement of all input images, leading to more accurate and globally consistent results.

3.3 Optimization

The global initialization provides each frame with a reasonable pose estimate, while its scale, inherited from the foundation model, remains approximate and may vary across windows. Moreover, each frame can be connected to multiple keyframes from overlapping windows, resulting in redundant but not always consistent relative pose constraints. We therefore perform global optimization to consolidate local predictions into a coherent reconstruction.

Motion Averaging. We first estimate globally consistent camera poses from the pose graph. Inspired by GLOMAP [41], we decompose this step into rotation averaging followed by translation averaging. Given the relative rotations associated with the graph edges, we perform robust rotation averaging [22] to obtain a globally consistent rotation $\mathbf{R}_i$ for each frame. With rotations fixed, we estimate camera centers and sparse 3D points using multi-view ray consistency. For a track observation $\mathbf{u}_{ij}$ of point j in frame i, we convert it into a normalized camera ray $\mathbf{v}_{ij}$. Instead of optimizing pairwise translation scales, we require the 3D point to lie on the corresponding world-frame viewing ray:

$$\min_{\{\mathbf{c}_i\},\{\mathbf{X}_j\},\{d_{ij}\}} \sum_{(i,j)\in\mathcal{O}} \omega_{ij}\rho\left(\left\|\mathbf{X}_j - \left(\mathbf{c}_i + d_{ij}\mathbf{R}_i^\top \mathbf{v}_{ij}\right)\right\|_2^2\right), \tag{5}$$

where $\mathcal{O}$ is the set of valid track observations, $\mathbf{c}_i$ is the camera center, $\mathbf{X}_j$ is the 3D point, d_{ij} is the depth of point j along the viewing ray of frame i, ω_{ij} is the tracking confidence, and $\rho(\cdot)$ is a robust loss. This formulation is robust to inconsistent local translation scales and recovers globally compatible camera centers and sparse points from multi-view tracks. The resulting poses provide a stable initialization for the subsequent bundle adjustment.

Bundle Adjustment. Starting from the global initialization, we further refine camera poses, sparse points, and camera intrinsics by minimizing reprojection errors over all valid track observations:

$$\min_{\{\mathbf{T}_i\},\{\mathbf{X}_j\},\{\mathbf{K}_i\},\{\boldsymbol{\delta}_i\}} \sum_{(i,j)\in\mathcal{O}} \omega_{ij}\rho\left(\left\|\pi\left(\mathbf{K}_i, \boldsymbol{\delta}_i, \mathbf{T}_i, \mathbf{X}_j\right) - \mathbf{u}_{ij}\right\|_2^2\right), \tag{6}$$

where $\mathbf{T}_i$ is the camera pose, $\mathbf{K}_i$ is the camera intrinsic matrix, $\boldsymbol{\delta}_i$ denotes the distortion parameters, and $\pi(\cdot)$ is the projection function with distortion. When camera calibration is available, we keep $\mathbf{K}_i$ and $\boldsymbol{\delta}_i$ fixed; otherwise, they are jointly optimized with poses and points. The confidence ω_{ij} down-weights uncertain matches, while the robust loss suppresses outliers from incorrect tracks and occlusions. This two-stage optimization first resolves large-scale inconsistencies and then performs accurate reprojection-based refinement, allowing poses to be optimized at the frame level rather than merely stitched at window boundaries.

Dense Reconstruction. During initialization, we also store the predicted depth and confidence map of each frame. After optimization, the sparse 3D tracks provide reliable depth samples at their corresponding pixels. For each keyframe, we estimate a scale factor between the predicted dense depth and the optimized sparse depths using RANSAC, and rescale the dense depth accordingly. The rescaled depth is then back-projected with the optimized intrinsics and distortion parameters to obtain a dense point map in the camera coordinate system. Finally, all dense point maps are transformed by the optimized poses and fused into a globally aligned dense point cloud.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate our method on diverse indoor, outdoor, unordered SfM, and large-scale driving scenes, including 19 Tanks and Temples (T&T) scenes [27], 9 TUM RGB-D sequences [52], 13 ETH3D scenes [49], and 11 KITTI driving sequences [20]. T&T, TUM RGB-D, and KITTI are evaluated as streaming or sequential inputs, while ETH3D is treated as an unordered SfM-style image collection. These datasets cover diverse scene scales, camera motions, and visual conditions.



Figure 2: Qualitative comparison of novel-view synthesis on Tanks and Temples. Our method produces cleaner renderings with fewer pose-induced artifacts.

Table 1: PSNR comparison on selected Tanks and Temples scenes. Best results are in bold.

Scene	Barn	Cat.	Chur.	Court.	C.room	Family	Franc.	Horse	Light.	M.room	Museum	Play.	Temple	Train	Avg.
COLMAP [47]	24.09	**17.04**	18.14	18.04	**18.25**	19.40	21.80	19.47	16.65	18.59	16.87	19.07	18.10	16.70	18.73
GLOMAP [41]	24.26	16.97	**18.20**	**21.08**	18.20	13.08	22.00	14.85	17.76	19.56	11.81	15.72	18.23	12.35	17.43
AMB3R [60]	20.08	15.45	15.98	15.64	15.58	19.50	19.19	19.08	15.00	16.63	15.97	18.04	15.74	15.66	16.97
Scal3R [72]	20.00	14.86	15.08	18.20	13.28	17.21	16.08	16.58	12.09	14.74	13.01	16.79	17.09	13.04	15.58
DA3 [32]	21.78	15.73	16.51	OOM	16.34	18.93	20.38	18.60	17.43	17.78	15.60	19.01	17.21	15.88	17.78
VGGT-Ω [62]	21.43	16.04	15.95	14.47	17.09	18.98	21.06	19.35	17.21	17.92	15.74	18.50	15.73	15.65	17.51
Pi3X [67]	21.22	16.04	17.01	16.74	17.23	18.89	20.27	18.19	17.34	17.63	15.56	18.20	16.83	15.57	17.62
SAIL-R. [7]	23.50	16.80	17.00	15.09	17.40	**20.60**	21.80	20.10	18.20	19.50	15.40	20.30	17.80	16.20	18.55
LingBot [4]	18.54	14.51	15.44	14.43	15.65	17.60	19.01	17.17	15.29	16.08	14.62	17.18	15.28	14.77	16.11
Ours	**24.72**	16.99	17.89	20.72	16.78	20.48	**22.13**	**20.39**	**18.90**	**19.71**	**17.13**	**21.66**	**18.42**	**16.89**	**19.49**

Evaluation Metrics and Inference Settings. We mainly evaluate the camera pose accuracy. For T&T, accurate ground-truth poses are not available, and using COLMAP poses as ground truth would be inappropriate since they are also estimated results [2]. We therefore follow the previous works [2, 7] to evaluate pose quality through novel-view synthesis. For each method, we split the estimated camera pose into training/testing splits, and then train a Nerfacto model [54] on training views and report PSNR on rendered testing views as an indirect measure of pose precision. For TUM RGB-D and KITTI, ground-truth trajectories are available, so we align the estimated trajectories to the ground truth and report pose RMSE. For ETH3D, we report RRA and RTA, the percentages of camera pairs whose relative rotation and translation-direction errors are below 5° or 1°. All benchmark comparisons use a sliding window size of $N = 20$ with a stride of 10 frames (overlap ratio 0.5), without per-scene window-size tuning. Further inference details and parameter ablations are provided in the supplementary material.

4.2 Comparison with State-of-the-art Methods

We compare our method with a broad set of reconstruction and SLAM baselines, including classical SfM methods COLMAP [47], GLOMAP [41], feed-forward 3D foundation models such as DA3 [32], Pi3X [67], and VGGT-Ω [62], streaming SLAM systems such as DROID-SLAM [57], and recent long-sequence or SfM-style learning-based methods, including VGGT-Long [11], VGGT-SLAM [37], LingBot-Map [4], LoGeR [78], SCAL3R [72], SAILRecon [7], MASt3R-SLAM [40], and AMB3R [60]. Please refer to the supplementary for more baseline details and additional comparisons.

Tanks and Temples. Table 1 reports novel-view synthesis results on T&T, where higher PSNR indicates more accurate and consistent camera poses. Our method achieves the best average PSNR, outperforming both the classical SfM systems COLMAP [47] and GLOMAP [41] and the learning-based reconstruction methods on average. Although GLOMAP obtains strong results on several scenes, its performance is less stable due to inaccurate edges and outliers in the pose graph. In contrast, our method benefits from the geometric priors of the foundation model, which provide robust initial poses, point maps for subsequent optimization. Compared with feed-forward methods such as DA3 [32], Pi3X [67], and VGGT-Ω [62], our method achieves better reconstruction quality, suggesting that feed-forward global reasoning alone is insufficient for high-precision pose estimation. Recent

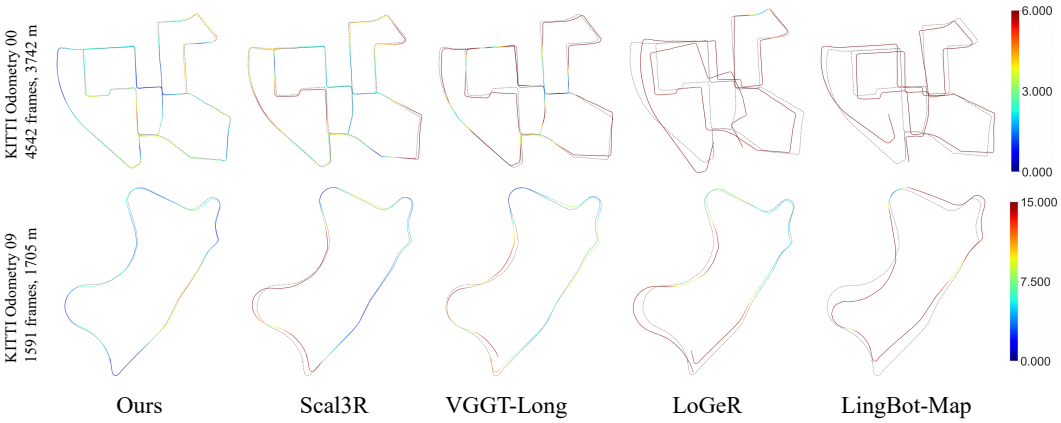

Figure 3: Camera trajectory comparison on KITTI Odometry. Our method better preserves the global trajectory shape and reduces drift on long driving sequences.

streaming or post-optimization pipelines, including LingBot-Map [4], LoGeR [78], Scal3R [72], and SAIL-Recon [7], also remain inferior on this benchmark. This reveals our motivation: 3D foundation models provide robust but coarse pose initialization and geometry-aware features, while classical global SfM can refine geometry but is sensitive to unreliable pose graphs. By converting dense warps into reliable multi-view tracks, our method bridges these two sides and enables stable global SfM-style optimization for high-fidelity reconstruction. The qualitative results in Fig. 2 confirm this observation, where our rendered views are visually closer to the ground-truth images.

Table 2: Pose estimation results on TUM RGB-D sequences. We report trajectory RMSE (cm) ↓.

Method	360	desk	desk2	floor	plant	room	rpy	teddy	xyz	Avg.
DROID-SLAM [57]	20.2	3.2	9.1	6.4	4.5	91.8	5.6	4.5	1.2	16.3
MASt3R-SLAM [40]	7.0	3.5	5.5	5.6	3.5	11.8	4.1	11.4	2.0	6.0
VGGT-SLAM [37]	7.1	2.5	4.0	14.1	2.3	10.2	3.0	3.4	1.4	5.3
AMB3R [60]	**4.6**	1.9	2.8	**3.2**	2.9	**5.8**	2.3	3.7	1.1	**3.2**
LoGeR [78]	10.6	3.2	4.6	10.3	4.6	9.8	3.0	8.4	2.2	6.3
Scal3R [72]	6.6	5.5	2.9	18.9	4.5	11.0	3.2	9.5	5.0	7.4
VGGT-SLAM 2.0 [36]	5.0	2.5	2.9	10.2	2.6	6.3	2.6	3.8	1.4	4.1
SAILRecon [7]	7.0	2.4	4.2	10.7	3.1	11.3	**2.0**	3.7	1.2	5.1
LingBot-Map [4]	6.3	2.8	4.4	5.9	4.4	9.2	2.4	3.6	1.1	4.4
Ours	7.9	**1.8**	**2.7**	3.3	**1.7**	5.9	2.2	**2.7**	**0.9**	**3.2**

TUM RGB-D. Table 2 evaluates pose accuracy in small-scale RGB-D indoor sequences. Our method achieves the best average RMSE among these uncalibrated methods and remains competitive with recent SLAM and reconstruction systems. This benchmark contains short indoor trajectories with frequent rotations, limited baselines, and appearance variations, posing challenges for feature matching followed by geometric optimization. Although our method requires explicit feature matching, it remains stable under these difficult motion settings. These results show that our approach is effective for compact indoor sequences where accurate local pose refinement is required.

Table 3: Pose estimation results on KITTI sequences. We report trajectory RMSE (m) ↓.

Method Frames / km	00 4542 / 3.7	01 1101 / 2.5	02 4661 / 5.1	03 801 / 0.6	04 271 / 0.4	05 2761 / 2.2	06 1101 / 1.2	07 1101 / 0.7	08 4071 / 3.2	09 1591 / 1.7	10 1201 / 0.9	Avg.
DROID-SLAM [57]	92.10	344.60	107.61	2.38	1.00	118.50	62.47	21.78	161.60	72.32	118.70	100.28
VGGT-Long [11]	8.64	61.21	52.72	8.78	4.20	9.88	4.67	2.66	72.98	31.84	27.71	25.94
VGGT-SLAM 2.0 [36]	TL	163.65	TL	50.04	19.38	159.58	46.35	57.80	TL	167.96	76.99	92.72
MASt3R-SLAM [40]	OOM	530.37	OOM	18.87	88.99	159.43	92.00	OOM	263.75	TL	153.07	186.64
AMB3R [60]	167.38	276.12	157.07	22.06	6.91	149.11	55.94	37.25	85.68	134.73	52.15	104.04
LingBot-Map [4]	27.17	70.94	112.02	**2.02**	1.36	26.14	16.61	10.48	**23.82**	17.84	6.48	28.63
LoGeR [78]	30.47	47.91	**36.32**	5.38	1.95	26.34	6.60	5.55	24.41	10.12	10.11	18.65
Scal3R [72]	4.30	**45.29**	42.06	3.36	1.74	3.30	**2.49**	2.03	36.69	12.32	**6.46**	14.55
Ours	**3.10**	49.95	49.81	2.90	**0.42**	**2.61**	4.10	**1.51**	30.48	**3.81**	9.12	**14.35**

KITTI. Table 3 evaluates long-sequence driving scenarios. Our method achieves the best average RMSE across all KITTI sequences. This benchmark is challenging because forward-facing vehicle motion often produces small parallax, weak triangulation, and unstable feature matching. Classical SfM methods [47, 41] are therefore prone to failure or high computational cost in such settings. Recent scalable or streaming reconstruction methods, including VGGT-Long [11], Scal3R [72], LoGeR [78], and LingBot-Map [4], alleviate the memory issue with chunk-wise processing, test-time

Table 4: Pose estimation accuracy comparison on ETH3D at loose (5°) and strict (1°) thresholds. Strict accuracy is reported for AMB3R, VGGT-Ω, and our method.

Scenes	Threshold @5												Threshold @1					
	COLMAP [47]		GLOMAP [41]		DF-SfM [23]		MASt3R-SfM [14]		AMB3R [60]		**Ours**		AMB3R [60]		VGGT-Ω [62]		**Ours**	
	R@5	T@5	R@5	T@5	R@5	T@5	R@5	T@5	R@5	T@5	R@5	T@5	R@1	T@1	R@1	T@1	R@1	T@1
courtyard	56.3	60.0	30.33	26.52	80.7	74.8	89.8	64.4	**100.0**	96.5	**100.0**	**97.2**	86.57	50.28	99.45	**77.29**	**100.00**	73.20
deli. area	34.0	28.1	17.77	8.37	82.5	**82.0**	83.1	81.8	**91.0**	76.6	83.5	81.1	69.63	15.70	**85.54**	57.33	83.06	**67.10**
electro	53.3	48.5	57.23	50.12	82.8	81.2	**100.0**	**95.5**	95.6	81.2	**100.0**	85.1	80.05	36.74	**88.54**	62.22	82.91	**62.52**
facade	92.2	90.0	55.06	49.26	80.9	82.6	74.3	75.3	**100.0**	95.4	**100.0**	**98.2**	86.81	50.12	78.84	62.74	**99.86**	**88.99**
kicker	87.3	86.2	80.65	49.74	93.5	91.0	**100.0**	**100.0**	**100.0**	99.2	**100.0**	95.8	95.42	80.54	94.59	77.94	**99.79**	**85.43**
meadow	0.9	0.9	28.00	7.11	56.2	58.1	58.1	58.1	**100.0**	**95.2**	**100.0**	92.4	80.44	40.00	**100.00**	**87.56**	78.67	68.44
office	36.9	32.3	34.91	31.07	71.1	54.5	**100.0**	**98.5**	**100.0**	53.9	**100.0**	63.2	**92.31**	18.79	78.40	**41.57**	77.81	33.43
pipes	30.8	28.6	74.49	65.31	72.5	61.5	**100.0**	**100.0**	**100.0**	87.9	**100.0**	92.9	67.35	30.61	**100.00**	**79.08**	**100.00**	73.47
p.ground	17.2	18.1	75.90	28.12	70.5	70.1	**100.0**	93.6	98.7	62.2	**100.0**	**96.0**	69.94	19.74	80.89	48.61	**91.69**	**71.26**
relief	16.8	16.8	81.89	78.98	32.9	32.9	34.2	40.2	**100.0**	90.1	**100.0**	**96.6**	68.16	30.70	51.30	27.26	**100.00**	**89.18**
relief 2	11.8	11.8	50.26	47.35	40.9	39.1	57.4	76.1	**100.0**	75.7	**100.0**	**96.3**	45.47	8.64	41.10	25.18	**100.00**	**80.85**
terrace	**100.0**	**100.0**	83.74	79.02	**100.0**	99.6	**100.0**	**100.0**	**100.0**	97.2	**100.0**	95.7	90.17	61.44	96.60	80.72	**100.00**	**93.38**
terrains	**100.0**	**99.5**	55.90	50.17	**100.0**	91.9	58.2	52.5	91.6	53.8	**100.0**	97.3	37.64	9.35	99.66	62.98	**100.00**	**90.53**
Average	49.0	47.8	55.86	43.93	74.2	70.7	81.2	79.7	98.2	81.9	**98.7**	**91.4**	74.61	34.82	84.22	60.81	**93.37**	**75.21**

Table 5: Ablation study of different components in our method.

Method	**Init**	**Motion Avg.**	**Ours**	**Coarse Track.**	**VGGT-Track. [61]**	**RomaV2-Track. [16]**
RRA@1 (↑)	69.70	70.20	**93.37**	67.11	59.29	57.12
RTA@1 (↑)	33.64	34.75	**75.21**	48.56	29.35	44.78
Speed (FPS) (↑)	**3.34**	2.77	2.03	2.54	1.14	0.82

optimization, or streaming updates, but still accumulate trajectory errors over long driving sequences. In contrast, our sliding-window association converts dense warps into sparse tracks and propagates them across overlapping windows, enabling frame-level pose optimization over the full sequence. The qualitative trajectories in Fig. 3 show the same trend: our method closely follows the ground truth on KITTI Odometry 00 and 09, while competing methods exhibit more visible drift or trajectory deformation.

ETH3D. Table 4 reports results on unordered SfM-style image collections. We first compare with GLOMAP [41] using the same view graph connectivity. Although our optimization follows a similar global SfM formulation, GLOMAP drops noticeably in this setting because erroneous edges in the view graph can strongly affect rotation and translation averaging. In contrast, our method benefits from geometric foundation model priors and confidence-aware dense matching, which provide more reliable tracks for global optimization. Our method also significantly outperforms strong learning-based baselines such as AMB3R-SfM [60]. Under the loose threshold of 5°, our method reaches saturated or near-saturated RRA@5 and RTA@5 on most scenes; under the stricter 1° threshold, it still achieves substantially higher RRA@1 and RTA@1 than AMB3R, indicating more accurate relative poses for high-precision SfM evaluation.

4.3 Ablation Study

We perform ablation studies on ETH3D with accurate ground-truth poses. Table 5 compares different variants using RRA@1 and RTA@1. The initialization achieves reasonable accuracy, showing that the system can connect different frames to a meaningful global structure. Motion Averaging further brings a modest improvement by making the initial poses more globally consistent. After complete optimization, our method improves both rotation and translation accuracy, confirming the importance of geometric bundle adjustment.

We further analyze the matching component. Using the coarse matching gives limited improvement over initialization, indicating that the refinement is important for accurate correspondence. Replacing our matching with the VGGT [61] tracking head or RoMaV2 [16] leads to poorer accuracy and lower efficiency. This is because VGGT tracks points from a single reference view, so multiple keyframes require repeated inference; moreover, the resulting tracks are less reliable than the points obtained from global dense warping. RoMaV2 only operates as a two-view matcher and does not exploit the shared multi-view geometric context provided by the foundation model. These results show that our dense matching head provides more accurate and efficient tracks for global pose optimization.

Additional KITTI ablations on frame-level association, global positioning, window size, and overlap ratio are reported in Sec. C.4 of the supplementary material.

5 Conclusion

We present a foundation-model-guided global SfM framework that transforms feed-forward 3D predictions into optimizable geometric constraints. Our method leverages the predicted geometry and cross-view features of a frozen foundation model to build reliable correspondences and an initial pose graph. With a keyframe-based sliding window strategy, these constraints are propagated across long sequences and unordered image collections. Global pose estimation and bundle adjustment then refine the poses, reduce scale inconsistencies, and recover dense geometry. Experiments on diverse benchmarks demonstrate that our framework combines the robustness of 3D foundation models with global SfM-style optimization, achieving efficient and high-fidelity reconstruction.

Future Work. Our current sliding-window strategy adopts a fixed window size. Adapting the window size according to motion type, visual overlap, and matching confidence may further improve efficiency and robustness. In addition, the matching head still depends on the geometric predictions of the underlying foundation model. Reducing this dependence and improving robustness to imperfect initial geometry remain important directions for future research.

Acknowledgment. This work was supported by Alibaba Research Intern Program.

References

[1] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, et al. Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgb-d data. *arXiv preprint arXiv:2111.08897*, 2021.

[2] 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.

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

[4] Lin-Zhuo Chen, Jian Gao, Yihang Chen, Ka Leong Cheng, Yipengjing Sun, Liangxiao Hu, Nan Xue, Xing Zhu, Yujun Shen, Yao Yao, and Yinghao Xu. Geometric context transformer for streaming 3d reconstruction. *arXiv preprint arXiv:2604.14141*, 2026.

[5] Chong Cheng, Xianda Chen, Tao Xie, Wei Yin, Weiqiang Ren, Qian Zhang, Xiaoyang Guo, and Hao Wang. Longstream: Long-sequence streaming autoregressive visual geometry. *arXiv preprint arXiv:2602.13172*, 2026.

[6] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[7] Junyuan Deng, Heng Li, Tao Xie, Weiqiang Ren, Qian Zhang, Ping Tan, and Xiaoyang Guo. Sail-recon: Large sfm by augmenting scene regression with localization. *2026 International Conference on 3D Vision (3DV).*, 2026.

[8] Junyuan Deng, Qi Wu, Xieyuanli Chen, Songpengcheng Xia, Zhen Sun, Guoqing Liu, Wenxian Yu, and Ling Pei. Nerf-loam: Neural implicit representation for large-scale incremental lidar odometry and mapping. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8218–8227, 2023.

[9] Junyuan Deng, Wei Yin, Xiaoyang Guo, Qian Zhang, Xiaotao Hu, Weiqiang Ren, Xiao-Xiao Long, and Ping Tan. Boost 3d reconstruction using diffusion-based monocular camera calibration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7110–7121, 2025.

[10] K Deng et al. Detailgen3d: generative 3d geometry enhancement via data-dependent flow. arxiv. *arXiv preprint arxiv:2411.16820*, 2024.

[11] Kai Deng, Zexin Ti, Jiawei Xu, Jian Yang, and Jin Xie. Vggt-long: Chunk it, loop it, align it – pushing vggt's limits on kilometer-scale long rgb sequences, 2025.

[12] Ken Deng, Yunhan Yang, Jingxiang Sun, Xihui Liu, Yebin Liu, Ding Liang, and Yan-Pei Cao. Geosam2: Unleashing the power of sam2 for 3d part segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6367–6376, 2026.

[13] Tianchen Deng, Wenhua Wu, Kunzhen Wu, Guangming Wang, Siting Zhu, Shenghai Yuan, Xun Chen, Guole Shen, Zhe Liu, and Hesheng Wang. Reloc-vggt: Visual re-localization with geometry grounded transformer. *arXiv preprint arXiv:2512.21883*, 2025.

[14] 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.

[15] Johan Edstedt, Ioannis Athanasiadis, Mårten Wadenbäck, and Michael Felsberg. Dkm: Dense kernelized feature matching for geometry estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17765–17775, 2023.

[16] Johan Edstedt, David Nordström, Yushan Zhang, Georg Bökman, Jonathan Astermark, Viktor Larsson, Anders Heyden, Fredrik Kahl, Mårten Wadenbäck, and Michael Felsberg. Roma v2: Harder better faster denser feature matching. *arXiv preprint arXiv:2511.15706*, 2025.

[17] Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Roma: Robust dense feature matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19790–19800, 2024.

[18] Xianze Fang, Jingnan Gao, Zhe Wang, Zhuo Chen, Xingyu Ren, Jiangjing Lyu, Qiaomu Ren, Zhonglei Yang, Xiaokang Yang, Yichao Yan, and Chengfei Lyu. Dens3r: A foundation model for 3d geometry prediction. In *The Fourteenth International Conference on Learning Representations*, 2026.

[19] Jingnan Gao, Zhe Wang, Xianze Fang, Xingyu Ren, Zhuo Chen, Shengqi Liu, Yuhao Cheng, Jiangjing Lyu, Xiaokang Yang, and Yichao Yan. More: 3d visual geometry reconstruction meets mixture-of-experts. *arXiv preprint arXiv:2510.27234*, 2025.

[20] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[21] Venu Madhav Govindu. Combining two-view constraints for motion estimation. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 2, pages II–II. IEEE, 2001.

[22] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *International journal of computer vision*, 103(3):267–305, 2013.

[23] 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.

[24] 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.

[25] Sergio Izquierdo and Javier Civera. Optimal transport aggregation for visual place recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024.

[26] Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, et al. Mapanything: Universal feed-forward metric 3d reconstruction. *arXiv preprint arXiv:2509.13414*, 2025.

[27] 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.

[28] Heng Li, Zhaopeng Cui, Shuaicheng Liu, and Ping Tan. Rago: Recurrent graph optimizer for multiple rotation averaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15787–15796, 2022.

[29] Yixuan Li, Lihan Jiang, Linning Xu, Yuanbo Xiangli, Zhenzhi Wang, Dahua Lin, and Bo Dai. Matrixcity: A large-scale city dataset for city-scale neural rendering and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3205–3215, 2023.

[30] 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.

[31] Zizun Li, Jianjun Zhou, Yifan Wang, Haoyu Guo, Wenzheng Chang, Yang Zhou, Haoyi Zhu, Junyi Chen, Chunhua Shen, and Tong He. Wint3r: Window-based streaming reconstruction with camera token pool. *arXiv preprint arXiv:2509.05296*, 2025.

[32] Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025.

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

[34] 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.

[35] Jiarui Liu, Heng Li, Weiyu Li, Keng Deng, Junyuan Deng, Zheng Zhongxing, Junyu Huang, Jiahao Chang, Xiaoguang Han, and Ping Tan. Reconplusgen: Injecting reconstruction prior into multi-view 3d generation through noise inversion and modulation. *arXiv preprint arXiv:2609.11129*, 2026.

[36] Dominic Maggio and Luca Carlone. Vggt-slam 2.0: Real-time dense feed-forward scene reconstruction. *arXiv preprint arXiv:2601.19887*, 2026.

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

[38] Daniel Martinec and Tomas Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

[39] 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.

[40] 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.

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

[42] Linfei Pan, Johannes Lutz Schönberger, and Marc Pollefeys. Global structure-from-motion meets feedforward reconstruction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[43] 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.

[44] 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.

[45] 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.

[46] 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.

[47] 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.

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

[49] Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[50] You Shen, Zhipeng Zhang, Yansong Qu, Xiawu Zheng, Jiayi Ji, Shengchuan Zhang, and Liujuan Cao. Fastvggt: Training-free acceleration of visual geometry transformer. *arXiv preprint arXiv:2509.02560*, 2025.

[51] 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.

[52] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[53] 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.

[54] 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*, SIGGRAPH '23, 2023.

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

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

[57] Zachary Teed and Jia Deng. DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras. *Advances in neural information processing systems*, 2021.

[58] Fabio Tosi, Yiyi Liao, Carolin Schmitt, and Andreas Geiger. Smd-nets: Stereo mixture density networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[59] 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.

[60] Hengyi Wang and Lourdes Agapito. Amb3r: Accurate feed-forward metric-scale 3d reconstruction with backend. *arXiv preprint arXiv:2511.20343*, 2025.

[61] 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.

[62] Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. Vggt-ω, 2026.

[63] 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.

[64] 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.

[65] 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.

[66] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4909–4916. IEEE, 2020.

[67] Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Scalable permutation-equivariant visual geometry learning, 2025.

[68] 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.

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

[70] Yuqi Wu, Wenzhao Zheng, Jie Zhou, and Jiwen Lu. Point3r: Streaming 3d reconstruction with explicit spatial pointer memory. *arXiv preprint arXiv:2507.02863*, 2025.

[71] 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.

[72] Tao Xie, Peishan Yang, Yudong Jin, Yingfeng Cai, Wei Yin, Weiqiang Ren, Qian Zhang, Wei Hua, Sida Peng, Xiaoyang Guo, and Xiaowei Zhou. Scal3r: Scalable test-time training for large-scale 3d reconstruction, 2026.

[73] Zhuang Xiong, Chen Zhang, Qingshan Xu, and Wenbing Tao. Vggt-motion: Motion-aware calibration-free monocular slam for long-range consistency. *arXiv preprint arXiv:2602.05508*, 2026.

[74] Zhongmiao Yan, Qi Wu, Songpengcheng Xia, Junyuan Deng, Xiang Mu, Renbiao Jin, Changchun Ye, and Ling Pei. 360recon: An accurate reconstruction method based on depth fusion from 360 images. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 17556–17563. IEEE, 2025.

[75] 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.

[76] 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.

[77] Zitong Zhan, Huan Xu, Zihang Fang, Xinpeng Wei, Yaoyu Hu, and Chen Wang. Bundle adjustment in the eager mode. *IEEE Transactions on Robotics*, 2026.

[78] Junyi Zhang, Charles Herrmann, Junhwa Hur, Chen Sun, Ming-Hsuan Yang, Forrester Cole, Trevor Darrell, and Deqing Sun. Loger: Long-context geometric reconstruction with hybrid memory. *arXiv preprint arXiv:2603.03269*, 2026.

[79] Jiankun Zhong, Zitong Zhan, Quankai Gao, Ziyu Chen, Haozhe Lou, Jiageng Mao, Ulrich Neumann, Chen Wang, and Yue Wang. Instantsfm: Towards gpu-native sfm for the deep learning era. *arXiv preprint arXiv:2510.13310*, 2025.

[80] Yang Zhou, Yifan Wang, Jianjun Zhou, Wenzheng Chang, Haoyu Guo, Zizun Li, Kaijing Ma, Xinyue Li, Yating Wang, Haoyi Zhu, et al. Omniworld: A multi-domain and multi-modal dataset for 4d world modeling. *arXiv preprint arXiv:2509.12201*, 2025.

In this appendix, we provide additional details and results to complement the main paper. Sec. A describes the network architecture, including the Pi3X backbone, matching network, and refinement module. Sec. B presents the loss functions and training protocol. Sec. C provides further experimental details, including inference settings, baseline configurations, additional quantitative and qualitative results, and failure case analysis.

A Network Architecture Details

Pi3X Backbone. Given an input image set $\mathcal{I} = \{I_i\}_{i=1}^{N}$ with resolution $H \times W$, our network takes images of shape $N \times H \times W \times 3$ as input. We first feed the images into the frozen Pi3X [67] backbone. Following Pi3X, each image is encoded by a DINOv2 encoder, producing encoder tokens

$$\mathbf{E} \in \mathbb{R}^{N \times H' \times W' \times C}, \quad H' = \frac{H}{14}, \quad W' = \frac{W}{14}. \tag{7}$$

The encoder processed by alternating frame-wise and global-wise attention blocks [61]. The decoder contains 18 attention blocks in total. The outputs of the last frame-wise and global-wise attention layers are concatenated, yielding the final geometry token

$$\mathbf{H} \in \mathbb{R}^{N \times H' \times W' \times 2C}. \tag{8}$$

The geometry token is further decoded by the original Pi3X prediction heads, including the point decoder, camera decoder, and metric decoder, to predict local point maps, camera poses, and metric scales, respectively. We refer readers to Pi3X [67] for more details of the backbone architecture.

Matching Decoder. Our coarse dense matching branch operates on the geometry token $\mathbf{H}$ produced by the Pi3X decoder. We introduce a lightweight matching decoder, denoted as $\mathrm{Dec}_{\mathrm{match}}$, which consists of 5 attention layers. It transforms the $2C$-dimensional geometry token into a compact matching token:

$$\mathbf{Z} = \mathrm{Dec}_{\mathrm{match}}(\mathbf{H}), \quad \mathbf{Z} \in \mathbb{R}^{N \times H' \times W' \times C}. \tag{9}$$

In our experiments, we find this matching decoder to be important. Directly reducing the channel dimension with a linear projection leads to significantly worse matching quality, suggesting that additional attention layers are needed to adapt the frozen geometry representation to dense correspondence prediction.

During training, we use the first image as the reference image and predict dense warps from it to all other images. During inference, thanks to the permutation-equivariant property of Pi3X, any image can be selected as the reference. Multiple reference images can also be handled by expanding the batch dimension.

Multi-view Match Embedding. We next construct multi-view match embeddings for dense warping. Let $M = H'W'$, and flatten the spatial dimensions of the matching tokens. During training, we use the image I_a as the reference image. Its matching token is denoted as

$$\mathbf{Z}^a = \{\mathbf{z}_n^a\}_{n=1}^{M}, \quad \mathbf{z}_n^a \in \mathbb{R}^C. \tag{10}$$

For each target image I_b, $b \in \{1, \cdots, N\} \setminus \{a\}$, we denote its matching token as

$$\mathbf{Z}^b = \{\mathbf{z}_m^b\}_{m=1}^{M}, \quad \mathbf{z}_m^b \in \mathbb{R}^C. \tag{11}$$

For each target-reference pair (a, b), we compute a patch-level similarity matrix from target patches to reference patches:

$$\mathbf{S}_{mn}^{a \to b} = \exp\left(\frac{1}{\tau}\mathrm{cosim}\left(\mathbf{z}_m^a, \mathbf{z}_n^b\right)\right), \quad \mathbf{S}^{a \to b} \in \mathbb{R}^{M \times M}, \tag{12}$$

where $\tau = 1/10$ is the temperature following RoMa, and $\mathrm{cosim}(\mathbf{x}, \mathbf{y})$ denotes cosine similarity:

$$\mathrm{cosim}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^\top \mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|}. \tag{13}$$

Computing this similarity for all target images yields

$$\mathbf{S}^{a \to \mathcal{B}} = \left\{\mathbf{S}^{a \to b}\right\}_{b \in \mathcal{B}} \in \mathbb{R}^{(N-1) \times M \times M}, \quad \mathcal{B} = \{1, \cdots, N\} \setminus \{a\}. \tag{14}$$

Following RoMa, we convert the similarity matrix into match embeddings using Fourier positional embeddings of the reference image coordinates. For each reference patch coordinate $\mathbf{x}_n^a \in \mathbb{R}^2$, we define

$$\boldsymbol{\chi}_n^a = \cos(2\pi\omega \mathbf{W}\mathbf{x}_n^a) \oplus \sin(2\pi\omega \mathbf{W}\mathbf{x}_n^a) \in \mathbb{R}^C, \tag{15}$$

where $\oplus$ denotes concatenation, $\omega = 1$, and $\mathbf{W}$ is a fixed non-learnable Gaussian matrix with a compatible output dimension. The match embedding for the target patch m in the image I_b is then calculated by aggregating the Fourier embeddings of the reference patches according to the similarity scores:

$$\boldsymbol{\chi}_m^{a\to b} = \sum_{n=1}^{M} \mathbf{S}_{mn}^{a\to b} \boldsymbol{\chi}_n^a, \quad \boldsymbol{\chi}_m^{a\to b} \in \mathbb{R}^C. \tag{16}$$

Therefore, the multi-view match embeddings for all target images are

$$\boldsymbol{\chi}^{a\to\mathcal{B}} = \left\{\boldsymbol{\chi}^{a\to b}\right\}_{b\in\mathcal{B}} \in \mathbb{R}^{(N-1)\times H'\times W'\times C}. \tag{17}$$

This formulation extends the two-view match embedding of RoMa to the multi-view setting by computing reference-to-target dense correspondence embeddings for all images in the local window.

DPT Matching Head. Given the reference-to-target match embeddings $\boldsymbol{\chi}^{a\to\mathcal{B}}$, we build pair-wise matching features for each target image I_b by combining the target matching token with its corresponding match embedding:

$$\mathbf{F}^{a\to b} = \mathrm{Proj}\left(\mathbf{Z}^b \oplus \boldsymbol{\chi}^{a\to b}\right), \quad \mathbf{F}^{a\to b} \in \mathbb{R}^{H'\times W'\times C}, \tag{18}$$

where $\oplus$ denotes channel-wise concatenation and $\mathrm{Proj}(\cdot)$ is a linear projection. Intuitively, $\mathbf{Z}^b$ provides the local image representation of the target frame, while $\boldsymbol{\chi}^{a\to b}$ encodes the soft correspondence distribution from the reference image coordinates patch to each target.

For all target images in the local window, the pair-wise features are stacked as

$$\mathbf{F}^{a\to\mathcal{B}} = \left\{\mathbf{F}^{a\to b}\right\}_{b\in\mathcal{B}} \in \mathbb{R}^{(N-1)\times H'\times W'\times C}. \tag{19}$$

We feed these features, together with the multi-scale encoder features from the Pi3X backbone, into a Dense Prediction Transformer head:

$$\left(\mathbf{W}^{a\to\mathcal{B}}, \mathbf{p}^{a\to\mathcal{B}}\right) = \mathrm{DPT}_{\mathrm{match}}\left(\mathbf{F}^{a\to\mathcal{B}}, \mathbf{E}\right). \tag{20}$$

The output is a dense warp from each target image to the reference image and its confidence:

$$\mathbf{W}^{a\to\mathcal{B}} \in \mathbb{R}^{(N-1)\times 2\times \frac{H}{4}\times\frac{W}{4}}, \quad \mathbf{p}^{a\to\mathcal{B}} \in \mathbb{R}^{(N-1)\times 1\times \frac{H}{4}\times\frac{W}{4}}. \tag{21}$$

We set the finest resolution of the DPT head to one quarter of the input image resolution. Following the standard DPT design, we use a scratch dimension of 256 and output dimensions $[256, 512, 1024, 1024]$ for feature strides $[4, 8, 16, 32]$, respectively. The final coarse warp and confidence are predicted at stride 4, and are then passed to the refinement module to recover full-resolution correspondences.

Refinement Module. The coarse matching head predicts warps at stride 4. To recover full-resolution correspondences, we further employ a lightweight refinement module following the coarse-to-fine design of RoMaV2 [16]. Since our coarse prediction is already produced at stride 4, we only keep refinement stages at strides $\{4, 2, 1\}$, which avoids unnecessary high-level refinement and reduces both feature extraction and inference cost.

We extract fine image features at three resolutions:

$$\boldsymbol{\phi}_4 \in \mathbb{R}^{\frac{H}{4}\times\frac{W}{4}\times 192}, \quad \boldsymbol{\phi}_2 \in \mathbb{R}^{\frac{H}{2}\times\frac{W}{2}\times 48}, \quad \boldsymbol{\phi}_1 \in \mathbb{R}^{H\times W\times 12}. \tag{22}$$

These features are linearly projected before being passed to the corresponding refinement blocks. At each stride $s \in \{4, 2, 1\}$, the refiner takes the current warp estimate $\mathbf{W}^{a\to b}$ and predicts a residual update. Its input is constructed by concatenating the fine features from the reference and target images, a positional displacement embedding, and a local correlation feature around the current matched location:

$$\mathbf{F}_s^{a\to b} = \boldsymbol{\phi}_s^a \oplus \boldsymbol{\phi}_s^b(\mathbf{W}^{a\to b}) \oplus g_s\left(\mathbf{W}^{a\to b} - \mathbf{x}^a\right) \oplus \mathrm{Corr}_s\left(\boldsymbol{\phi}_s^a, \boldsymbol{\phi}_s^b, \mathbf{W}^{a\to b}\right), \tag{23}$$

where $\oplus$ denotes channel-wise concatenation, $\phi_s^b(\mathbf{W}^{a\to b})$ denotes target features sampled at the warped coordinates, $\mathbf{x}^a$ is the reference pixel coordinate, and $g_s(\cdot)$ is a linear projection of the current displacement. The local correlation term computes a small neighborhood correlation in the target feature map centered at the current warp location, providing fine-grained matching evidence for residual correction.

Each refinement block predicts a residual warp and confidence update:

$$(\Delta\mathbf{W}_s^{a\to b}, \Delta p_s^{a\to b}) = \mathrm{Refine}_s\left(\mathbf{F}_s^{a\to b}\right), \tag{24}$$

and the warp is progressively updated from stride 4 to stride 1:

$$\mathbf{W}_s^{a\to b} \leftarrow \mathrm{upsample}\left(\mathbf{W}_{2s}^{a\to b}\right) + \Delta\mathbf{W}_s^{a\to b}. \tag{25}$$

For local correlation, we use window sizes $[k_4, k_2, k_1] = [7, 3, 0]$, where $k_1 = 0$ means that no local correlation is used at full resolution. The internal structure of each refinement block follows a compact convolutional design with depthwise convolution, normalization, non-linearity, and pointwise projection. This refinement stage substantially improves the localization accuracy of the dense warp while keeping the additional computation moderate.

B Loss Function and Training Details

We train the matching branch using three losses: an auxiliary patch-level negative log-likelihood loss on the coarse similarity matrix, a dense warp regression loss, and a confidence supervision loss. During training, we use the first image I_a in each local window as the reference image, and treat the remaining images $\{I_b\}_{b\in\mathcal{B}}$ as target images, where $\mathcal{B} = \{1, \ldots, N\} \setminus \{a\}$.

Ground-truth Warp and Validity. For each reference-target pair (a, b), we generate the ground-truth warp from the reference image I_a to the target image I_b using the ground-truth depth and camera poses. For a reference pixel $\mathbf{x}^a$ with depth $z^a(\mathbf{x}^a)$, its corresponding point is first back-projected to the camera coordinate system of I_a and then transformed to the target view:

$$\tilde{\mathbf{x}}^{a\to b} = \mathbf{K}^b\left(\mathbf{R}^{a\to b}\left(\mathbf{K}^a\right)^{-1}\tilde{\mathbf{x}}^a z^a(\mathbf{x}^a) + \mathbf{t}^{a\to b}\right), \tag{26}$$

where $\tilde{\mathbf{x}}^a$ denotes the homogeneous pixel coordinate, and $(\mathbf{R}^{a\to b}, \mathbf{t}^{a\to b})$ is the relative pose from I_a to I_b. The projected pixel and its corresponding depth in I_b are

$$\mathbf{x}^{a\to b} = \pi\left(\tilde{\mathbf{x}}^{a\to b}\right), \quad z^{a\to b} = \left[\mathbf{R}^{a\to b}\left(\mathbf{K}^a\right)^{-1}\tilde{\mathbf{x}}^a z^a(\mathbf{x}^a) + \mathbf{t}^{a\to b}\right]_3, \tag{27}$$

where $\pi(\cdot)$ denotes the projection from homogeneous coordinates to image coordinates, and $[\cdot]_3$ extracts the depth component. The ground-truth warp is therefore defined as

$$\mathbf{W}^{*\,a\to b}(\mathbf{x}^a) = \mathbf{x}^{a\to b}. \tag{28}$$

We define the confidence label according to target-view depth consistency. A correspondence is assigned positive confidence if the projected pixel lies inside the target image, both the projected depth and the sampled target depth are positive, and their relative depth difference is smaller than a threshold:

$$y^{a\to b}(\mathbf{x}^a) = \mathbf{1}\left[\mathbf{x}^{a\to b} \in \mathcal{D}^b, \quad z^{a\to b} > 0, \quad \frac{\left|z^b\left(\mathbf{x}^{a\to b}\right) - z^{a\to b}\right|}{z^b(\mathbf{x}^{a\to b})} < \tau_z\right], \tag{29}$$

where $\mathcal{D}^b$ is the image domain of I_b, $z^b(\mathbf{x}^{a\to b})$ is the target-view depth sampled at the projected location, and $\tau_z = 0.05$ in our training.

We further define a training mask to specify where the confidence loss and warp loss is applied. The mask is valid when the reference depth is positive and either the projected point falls inside the target image with positive projected and sampled depths, or the projected point falls outside the target image. The latter case is included to explicitly supervise out-of-bound correspondences with zero confidence. Formally,

$$m^{a\to b}(\mathbf{x}^a) = \mathbf{1}\Big[z^a(\mathbf{x}^a) > 0 \quad \text{and} \quad \Big((\mathbf{x}^{a\to b} \in \mathcal{D}^b \quad \text{and} \quad z^{a\to b} > 0 \quad \text{and} \quad z^b\left(\mathbf{x}^{a\to b}\right) > 0) \quad \text{or} \quad \mathbf{x}^{a\to b} \notin \mathcal{D}^b\Big)\Big]. \tag{30}$$

The warp regression loss is evaluated only on pixels with positive confidence label and positive mask label, while the confidence loss is evaluated on pixels selected by this mask. We visualize the original images, warped results, masks, and confidence maps in Fig. 4.

Auxiliary NLL Loss. To stabilize training of the coarse matching head, we add an auxiliary negative log-likelihood loss on the similarity matrix. Recall that for each target-reference pair (b, a), the coarse matching decoder produces a similarity matrix $\mathbf{S}^{a\to b} \in \mathbb{R}^{M\times M}$, where $M = H'W'$. For each target patch m, we determine the nearest reference patch index n_m^* according to the ground-truth warp. We then apply a row-wise Softmax to $\mathbf{S}^{a\to b}$ and minimize

$$\mathcal{L}_{\mathrm{NLL}}^{a\to b} = -\frac{1}{|\Omega_{\mathrm{patch}}^{a\to b}|} \sum_{m\in\Omega_{\mathrm{patch}}^{a\to b}} \log\left(\mathrm{Softmax}\left(\mathbf{S}_{m:}^{a\to b}\right)_{n_m^*}\right), \tag{31}$$

where $\Omega_{\mathrm{patch}}^{a\to b}$ denotes the set of valid target patches. This loss encourages the coarse similarity matrix to assign high probability to the correct reference patch before dense refinement.

Warp Loss. Given the predicted dense warp $\hat{\mathbf{W}}^{a\to b}$, we supervise it with a generalized Charbonnier loss. Let

$$\mathbf{r}^{a\to b}(\mathbf{x}) = \hat{\mathbf{W}}^{a\to b}(\mathbf{x}) - \mathbf{W}^{*\,a\to b}(\mathbf{x}) \tag{32}$$

denote the warp residual at pixel $\mathbf{x}$. The warp loss is defined as

$$\mathcal{L}_{\mathrm{warp}}^{a\to b} = \frac{1}{|\Omega^{a\to b}|} \sum_{\mathbf{x}\in\Omega^{a\to b}} \left(\|\mathbf{r}^{a\to b}(\mathbf{x})\|_2^2 + \epsilon^2\right)^{\alpha/2}, \tag{33}$$

where $\Omega^{a\to b} = \{\mathbf{x} \mid y^{a\to b}(\mathbf{x}) = 1\}$ is the set of valid pixels, ϵ is a small constant, and α controls the robustness of the penalty. We apply this loss to the coarse prediction as well as to all refinement stages.

Confidence Loss. In addition to the warp, the network predicts a confidence map $\hat{p}^{a\to b} \in [0, 1]$. We supervise it using the validity mask defined above. Specifically, we use a binary cross-entropy loss

$$\mathcal{L}_{\mathrm{conf}}^{a\to b} = -\frac{1}{|\mathcal{D}|} \sum_{\mathbf{x}\in\mathcal{D}} \left[y^{a\to b}(\mathbf{x}) \log \hat{p}^{a\to b}(\mathbf{x}) + \left(1 - y^{a\to b}(\mathbf{x})\right) \log\left(1 - \hat{p}^{a\to b}(\mathbf{x})\right)\right]. \tag{34}$$

This loss encourages the predicted confidence to reflect whether a correspondence is geometrically valid and depth-consistent.

Total Loss. The final training objective sums the three terms over all target images in the local window:

$$\mathcal{L} = \sum_{b\in\mathcal{B}} \lambda_{\mathrm{NLL}}\mathcal{L}_{\mathrm{NLL}}^{a\to b} + \lambda_{\mathrm{warp}}\mathcal{L}_{\mathrm{warp}}^{a\to b} + \lambda_{\mathrm{conf}}\mathcal{L}_{\mathrm{conf}}^{a\to b}, \tag{35}$$

where λ_{NLL}, λ_{warp}, and λ_{conf} balance the three losses.

Training Details. To preserve the geometric prior of the foundation model, we freeze the DINOv2 encoder and the original Pi3X geometry heads, and train only the matching decoder, the DPT matching head, and the refinement modules. We train the model on 16 NVIDIA H20 GPUs with gradient accumulation of 2 steps, resulting in an effective batch size equivalent to 32 GPUs. The coarse matching stage and the refinement stage are trained separately, each for 32K iterations. The overall training takes about 4–5 days.

For the coarse stage, we randomly sample 2 to 48 frames from each training sequence. The input frames are resized such that the longer image side is at most 518 pixels, and the aspect ratio is randomly sampled between 0.33 and 1.0. We apply standard data augmentation, including color jittering, Gaussian blur, and random grayscale conversion. The model is optimized with a cosine learning-rate schedule, using a peak learning rate of 1×10^{-4} and 2K warm-up iterations.

The refinement stage follows a similar training protocol, except that we sample 2 to 24 frames from each sequence. We initialize the refinement module from the RoMa v2 refinement checkpoint and train it with a cosine learning-rate schedule using a peak learning rate of 5×10^{-5} and 2K warm-up iterations.

Training Data. We train the matching branch on a diverse mixture of synthetic and real-world multi-view datasets, including Aria Synthetic Environments, Aria Digital Twin [43], DL3DV [34],

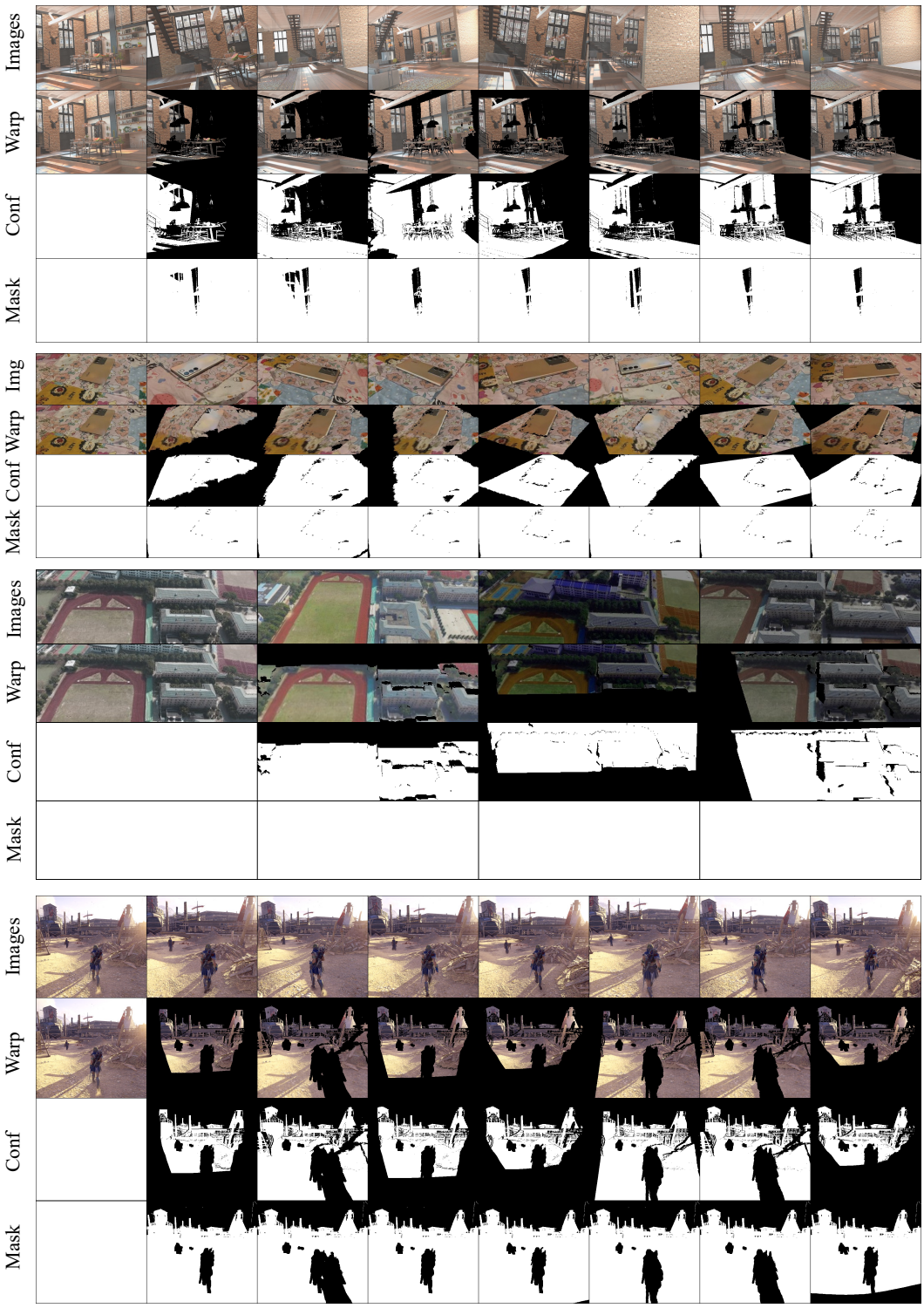

Figure 4: Qualitative results on various datasets. We visualize the original images, warped results, confidence maps, and masks. The rows from top to bottom correspond to the Hypersim [45], WildRGBD [71], BlendedMVS [75], and OmniWorld [80] datasets, respectively.

Table 6: Full PSNR comparison on all 19 Tanks and Temples scenes. Best results are in bold.

Scene	Auditorium	Ballroom	Barn	Caterpillar	Church	Courthouse	Courtroom	Family	Francis	Horse	Ignatius	Lighthouse	Meetingroom	Museum	Palace	Playground	Temple	Train	Truck	Avg.
DA3 [32]	19.40	13.41	21.78	15.73	16.51	OOM	16.34	18.93	20.38	18.60	18.48	17.43	17.78	15.60	12.86	19.01	17.21	15.88	18.17	17.42
VGGT-Ω [62]	19.15	13.65	21.43	16.04	15.95	14.47	17.09	18.98	21.06	19.35	18.76	17.21	17.92	15.74	12.20	18.50	15.73	15.65	19.76	17.30
COLMAP [47]	19.60	16.34	24.09	**17.04**	18.14	18.04	**18.25**	19.40	21.80	19.47	20.07	16.65	18.59	16.87	13.56	19.07	18.10	16.70	21.08	18.57
GLOMAP [41]	11.67	**20.52**	24.26	16.97	**18.20**	**21.08**	18.20	13.08	22.00	14.85	20.12	17.76	19.56	11.81	13.35	15.72	18.23	12.35	21.12	17.41
SAILRecon [7]	20.30	14.80	23.50	16.80	17.00	15.09	17.40	**20.60**	21.80	20.10	19.50	18.20	19.50	15.40	**14.30**	20.30	17.80	16.20	20.90	18.39
Scal3R [72]	16.55	13.56	20.00	14.86	15.08	18.20	13.28	17.21	16.08	16.58	14.82	12.09	14.74	13.01	13.14	16.79	17.09	13.04	14.70	15.31
Pi3X [67]	19.05	13.46	21.22	16.04	17.01	16.74	17.23	18.89	20.27	18.19	18.16	17.34	17.63	15.56	12.47	18.20	16.83	15.57	18.67	17.29
LingBot-Map [4]	18.30	12.90	18.54	14.51	15.44	14.43	15.65	17.60	19.01	17.17	16.59	15.29	16.08	14.62	12.23	17.18	15.28	14.77	16.01	15.87
AMB3R [60]	18.05	12.68	20.08	15.45	15.98	15.64	15.58	19.50	19.19	19.08	18.07	15.00	16.63	15.97	11.77	18.04	15.74	15.66	18.97	16.69
Ours	**20.70**	17.43	**24.72**	16.99	17.89	20.72	16.78	20.48	**22.13**	**20.39**	**20.43**	**18.90**	**19.71**	**17.13**	13.61	**21.66**	**18.42**	**16.89**	**21.14**	**19.27**



Figure 5: Additional qualitative comparisons on Tanks and Temples. Our method produces sharper and more faithful rendered views with fewer pose-induced artifacts.

Co3Dv2 [44], ARKitScenes [1], BlendMVS [75], HyperSim [45], MegaDepth [30], MVS-Synth [24], ScanNet [6], ScanNet++ [76], TartanAir [66], Virtual KITTI [3], Unreal4K [58], WildRGB-D [71], MatrixCity [29], and OmniWorld [80]. These datasets cover indoor and outdoor scenes, object-centric captures, synthetic environments, driving scenarios, and large-scale urban scenes, providing diverse camera motions, appearance variations, and geometric layouts for robust correspondence learning.

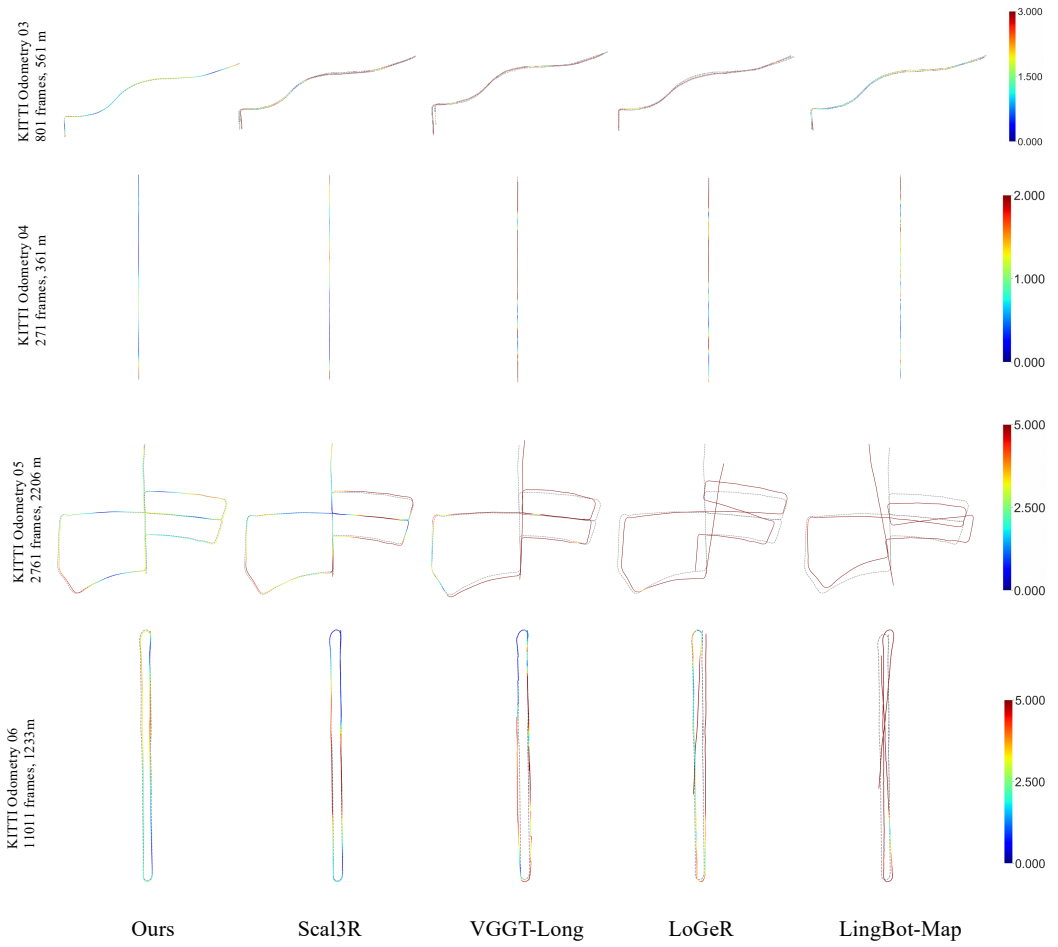

Figure 6: Additional KITTI trajectory visualizations. Our method better preserves global trajectory shape under long forward-driving motion.

C Experiments

C.1 Inference Details

Most experiments are conducted on a single NVIDIA L20 GPU with 48GB memory. All benchmark comparisons use a sliding window size of $N = 20$ and a stride of 10 frames (overlap ratio 0.5), without per-scene window-size tuning. For each keyframe, we sample 512 tracking points by default. For large-scale KITTI sequences, we reduce the number of sampled points to 256 to lower memory consumption. We use a warp confidence threshold of 0.6 and a depth confidence threshold of 0.1 for filtering reliable correspondences.

Sliding-window association. During inference, each window is processed as described in the main paper. We select keyframes according to the valid reprojection ratio. Specifically, if the ratio of valid projected pixels from the current frame to all existing keyframes in the window is below 0.2, the current frame is selected as a new keyframe. In other words, a frame is promoted to a keyframe when fewer than 20% of its pixels can be reliably projected to the current keyframe set. Due to the half-window overlap, most keyframes participate in two neighboring windows. Except for keyframes near the beginning and the end of the sequence, each keyframe can be associated with approximately $20 + 20 - 10 = 30$ frames. For frames appearing in the overlap, we use the matching results from their first inference to avoid duplicate tracks.

Track sampling. After obtaining dense image warps, we sample tracking points from each keyframe. We first filter pixels using the depth confidence threshold, and then randomly sample the required number of pixels from the remaining reliable regions. These points are propagated to other frames

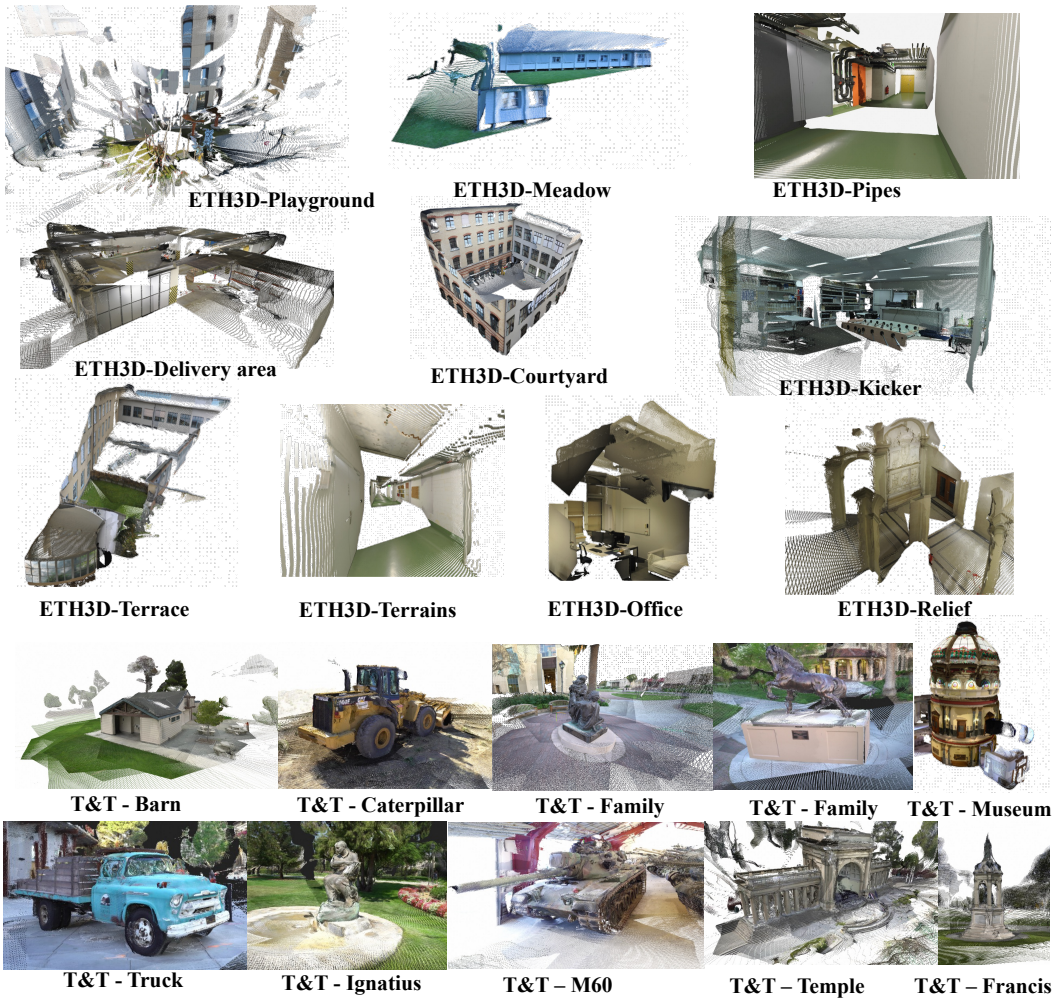

Figure 7: Dense point cloud reconstructions produced by our optimized poses and recovered dense depths.

using the predicted dense warp, and matches with warp confidence lower than 0.6 are discarded. This produces sparse but reliable tracks for subsequent pose initialization and optimization.

Retrieval-based loop association. In addition to local sliding-window associations, we perform image retrieval-based loop candidate discovery over all keyframes. We extract SALAD [25] image features for each keyframe and compute pairwise cosine similarities. Two keyframes are considered to have potential visual overlap if their similarity is larger than 0.5. For each keyframe, we collect all retrieved keyframes whose similarity exceeds this threshold to form an additional matching window, and use the first image in this window as the tracking reference. The resulting long-range matches are added to the optimization graph as loop constraints. This retrieval step is not only useful for explicit loop closure; it can also connect temporally distant but spatially close keyframes, especially in slow-motion regions where adjacent sliding windows may have limited viewpoint change.

Optimization settings. In the optimization stage, we support both shared and per-frame camera intrinsics. By default, we assume that all frames in a sequence share the same intrinsic parameters, while per-frame intrinsics can be enabled when necessary. For large-scale KITTI sequences, after retrieval-based loop detection, we first run pose graph optimization using relative poses from loop keyframe pairs and adjacent-frame constraints from the initialization. The optimized poses are then used to initialize the final global refinement.

GPU-based optimization. The global positioning and bundle adjustment stages can become a computational bottleneck if implemented with conventional CPU-based solvers, especially for long

sequences with many tracks. Following recent GPU-accelerated SfM and bundle adjustment systems [79, 77], we implement the optimization stage in a PyTorch-compatible GPU framework. In particular, the residuals and Jacobians for both global positioning and BA are constructed on GPU, and the sparse normal-equation structure is solved with batched parallel operations. This GPU-based implementation allows our optimization module to better match the efficiency of the feed-forward foundation model inference while preserving the accuracy of global geometric refinement.

C.2 Baseline Details

Unless otherwise specified, we run all baselines using their publicly released code and default configurations.

DA3. For Depth Anything 3 (DA3), we use its strongest released model, GIANT-LARGE1.1, for all evaluations.

VGGT-Ω. For VGGT-Ω [62], we use the original VGGT-Omega-1B-512 checkpoint with balanced preprocessing at resolution 512 and process all images of each T&T scene jointly.

COLMAP. For the T&T PSNR evaluation, we use the COLMAP poses precomputed by ACE0 [2] while for ETH3D evalution, we use the poses from AMB3R [60] and MASt3R-SfM [40].

GLOMAP. We use the default configuration of GLOMAP [41]. For a fair comparison, the image matching stage uses the same image pairs as our pipeline. In a few cases, GLOMAP produces outlier poses with extremely large translations; for these frames, we replace the pose with the last valid estimate.

SAIL-Recon. Following the original SAIL-Recon protocol [7], we refine the estimated poses with photometric bundle adjustment before evaluating PSNR on T&T.

LingBot-Map. For LingBot-Map [4], we use the author-recommended `lingbot-map-long` checkpoint with a keyframe interval of 1 by default. On the Courthouse scene, the default setting fails, so we rerun it with a keyframe interval of 2. For TUM RGB-D, since sequence lengths vary substantially, we evaluate keyframe intervals of 1, 2, and 3, and report the best result for each sequence.

LoGeR. For LoGeR [78], we use LoGeR*, the variant that adds a purely feed-forward alignment step to align raw predictions into a consistent global coordinate system. This setting also includes the chunk-alignment procedure used for long-sequence reconstruction.

VGGT-SLAM 2.0. For VGGT-SLAM 2.0 [36], we do not skip frames during evaluation; all frames are kept and participate in the subsequent optimization.

C.3 Additional Results

Additional T&T qualitative results. Fig. 5 provides additional novel-view synthesis comparisons on Tanks and Temples. Our rendered images are consistently closer to the ground-truth views, preserving sharper object boundaries and more stable scene structures. In contrast, COLMAP [47], DA3 [32], and LingBot-Map [4] often produce visible artifacts, blur, or distorted geometry. These qualitative results further support the PSNR comparison in Table 6, showing that our estimated poses lead to more accurate neural rendering.

Additional KITTI trajectory visualizations. Fig. 6 shows trajectory visualizations on additional KITTI [20] sequences. Our method remains close to the ground-truth trajectories across both short and long driving sequences, while competing methods exhibit larger drift or unstable trajectory shapes. This further validates the robustness of our sliding-window association and track propagation strategy under forward-driving motion, where limited parallax makes long-range pose estimation challenging.

Dense reconstruction results. Fig. 7 visualizes dense point clouds recovered by our method on both ETH3D [49] and Tanks and Temples [27] scenes. The reconstructions preserve coherent scene layouts across indoor, outdoor, and object-centric environments, demonstrating that the optimized poses and recovered dense depths can be fused into globally aligned dense geometry. These results also show that our framework is not limited to sparse pose estimation, but can provide dense reconstruction suitable for downstream 3D applications.

Table 7: Ablations on KITTI. We report trajectory RMSE (m) ↓ for all 11 sequences. The window-stitching comparison uses the same Pi3X backbone and window size $N = 20$. Pairwise trans. avg. replaces the ray-consistent global positioning stage with pairwise translation averaging.

Method	00	01	02	03	04	05	06	07	08	09	10	Avg.
Ours	**3.10**	49.95	**49.81**	2.90	**0.42**	**2.61**	4.10	1.51	**30.48**	3.81	**9.12**	**14.35**
Window stitching	11.41	47.69	73.08	3.87	4.81	16.52	6.96	5.00	30.93	12.07	12.85	20.47
Pairwise trans. avg.	7.96	**44.61**	66.89	**2.56**	0.80	14.05	**2.83**	**1.48**	31.40	**2.67**	9.20	16.77

Table 8: Window-size sensitivity on KITTI. We report trajectory RMSE (m) ↓ with overlap ratio 0.5. The lowest average error is obtained with $N = 20$.

N	00	01	02	03	04	05	06	07	08	09	10	Avg.
4	15.29	51.88	66.18	2.99	4.00	13.91	2.17	3.79	62.18	24.13	12.22	23.52
10	6.51	95.40	62.01	1.65	1.78	9.45	1.83	2.58	31.08	10.71	9.68	21.15
20	3.10	**49.95**	49.81	2.90	**0.42**	2.61	4.10	1.51	30.48	**3.81**	**9.12**	**14.35**
40	**3.01**	62.22	61.18	2.19	1.13	**1.76**	4.43	1.37	26.53	5.77	9.64	16.29
60	4.14	73.45	51.49	2.30	1.63	3.37	2.06	1.14	**24.68**	3.87	10.43	16.23
80	3.09	69.56	51.03	2.38	2.87	1.79	**1.73**	**0.73**	25.36	4.41	11.18	15.83
100	4.41	64.91	43.95	3.53	4.14	2.29	2.80	1.71	25.34	7.01	10.21	15.48
120	4.31	69.50	42.36	**1.62**	1.11	2.58	2.51	0.83	29.48	13.24	14.38	16.54
140	7.19	61.99	41.83	2.76	1.46	2.82	3.51	1.74	28.90	12.21	15.38	16.34
160	5.97	77.91	50.02	2.45	0.82	2.76	3.37	1.14	28.62	19.56	12.74	18.67
180	12.67	63.49	63.84	3.63	1.77	3.80	3.12	2.28	24.92	26.16	14.87	20.05
200	11.31	199.90	**31.71**	2.71	1.13	4.83	4.50	1.61	30.83	20.19	14.83	29.41

Full T&T quantitative results. Table 6 reports the full PSNR results on all 19 Tanks and Temples scenes. Our method achieves the best average PSNR and obtains the highest score on most scenes. Compared with feed-forward models and streaming reconstruction methods, the consistent improvement across diverse scenes indicates that explicit multi-view tracks and global geometric refinement are important for high-fidelity pose estimation and rendering quality.

C.4 Additional Ablation Studies

We evaluate the association strategy, translation-estimation stage, window size, and overlap ratio on all 11 KITTI sequences. Unless varied explicitly, the window size is $N = 20$ and the overlap ratio is 0.5. Each table reports per-sequence trajectory RMSE and its average in meters.

Frame-level association. Table 7 compares our frame-level association with window stitching using the same Pi3X backbone and window size. Frame-level association reduces the average RMSE from 20.47 m to 14.35 m, a relative reduction of 29.9%. It improves 10 of the 11 sequences, supporting the benefit of associating and refining individual frames across windows.

Translation estimation. Replacing ray-consistent global positioning with pairwise translation averaging increases the average RMSE from 14.35 m to 16.77 m (Table 7). The largest improvements occur on sequences 00, 02, and 05. Pairwise translation averaging performs better on some individual sequences, while the ray-consistent formulation achieves lower average error.

Window size. Table 8 varies N from 4 to 200 at an overlap ratio of 0.5. The default $N = 20$ gives the lowest average RMSE of 14.35 m. Larger windows do not consistently improve accuracy: the average RMSE is 16.54 m for $N = 120$ and 29.41 m for $N = 200$. These results support a fixed window of 20 frames for the benchmark comparisons.

Overlap ratio. Table 9 varies the overlap ratio from 0.1 to 0.9 with $N = 20$. An overlap ratio of 0.5 gives the lowest average RMSE, whereas reducing it to 0.1 increases the error to 20.80 m. Increasing the overlap beyond 0.5 also does not consistently improve accuracy. We do not report zero overlap, which removes the shared-frame links used by the local sliding-window association; retrieval-based links alone do not guarantee connectivity between every pair of adjacent windows.

Table 9: Overlap-ratio sensitivity on KITTI with $N = 20$. We report trajectory RMSE (m) $\downarrow$. The default overlap ratio of 0.5 achieves the lowest average error.

Overlap	00	01	02	03	04	05	06	07	08	09	10	Avg.
0.1	10.51	52.37	66.97	**1.57**	2.69	15.50	4.59	3.50	33.54	25.14	12.38	20.80
0.3	6.63	**49.40**	66.92	2.57	2.28	4.69	3.51	1.49	32.81	**3.10**	9.26	16.61
0.5	**3.10**	49.95	49.81	2.90	**0.42**	2.61	4.10	1.51	**30.48**	3.81	9.12	**14.35**
0.7	3.27	54.21	**49.79**	4.30	1.41	3.88	3.73	1.46	31.41	3.39	9.14	15.09
0.9	3.53	56.54	61.56	5.18	2.68	**2.48**	**3.04**	**1.42**	36.09	3.76	**7.97**	16.75

Table 10: Comparison of processing speed on an 800-frame sequence. We report FPS, where higher is faster.

Method	Category	FPS ↑
COLMAP [47]	Traditional SfM	0.14
GLOMAP [41]	Traditional SfM	0.67
Pi3X [67]	Feed-forward	1.64
DA3 [32]	Feed-forward	8.10
FastVGGT [50]	Feed-forward	15.10
VGGT-Long [11]	Streaming / Chunk + Optimization	4.00
VGGT-SLAM [37]	Streaming / Chunk + Optimization	16.87
MASt3R-SLAM [40]	Streaming / Chunk + Optimization	6.06
LingBot-Map [4]	Streaming / Chunk + Optimization	6.80
Scal3R [72]	Streaming / Chunk + Optimization	2.20
Ours	Foundation-model-guided global SfM	2.03

C.5 Runtime Analysis

Table 10 compares processing speed on an 800-frame sequence. Our reported total of 393.91 s corresponds to 2.03 FPS. Table 11 provides the component breakdown. Bundle adjustment is the largest listed stage at 151.80 s, followed by Pi3X inference at 101.55 s and the dense matching head at 68.40 s. Rotation averaging takes 1.92 s and global positioning takes 48.67 s. The listed stages sum to 372.34 s; the remaining 21.57 s is reported as the difference from the total without attributing it to a specific component. The pipeline is faster than the reported classical SfM baselines, while several feed-forward and streaming methods have higher throughput.

C.6 Failure Case Analysis.

Fig. 8 shows a typical failure case on the T&T Ballroom [27] scene. The initial Pi3X [67] prediction already contains strong geometric ambiguity: although the input images observe a single room, the predicted structure is separated into several inconsistent room-like fragments. Since our matching head is built upon the tokens produced by the Pi3X backbone, such ambiguous geometry can also affect the predicted correspondences. As shown in the matching visualization, visually repetitive structures, such as ceiling lamps, are incorrectly matched to different locations. Although our optimization can partially reduce the inconsistency, it cannot fully recover from severely incorrect initial geometry and corrupted matches. Similar failures are also observed in a few other T&T scenes, such as Palace. This limitation suggests that future work should reduce the dependence of dense matching on imperfect foundation-model geometry and make the correspondence estimation more robust under ambiguous scene layouts.

Table 11: Runtime breakdown for an 800-frame sequence. The remaining time is the difference between the reported total and the sum of the five listed stages; it is not a separately profiled component.

Stage	Time (s)
Pi3X geometry inference	101.55
Dense matching head	68.40
Rotation averaging	1.92
Global positioning	48.67
Bundle adjustment	151.80
Subtotal of listed stages	372.34
Remaining time (by subtraction)	21.57
Total	**393.91**



Pi3X Result **Our Result** **Our Matching Result**

Figure 8: Failure case on the T&T Ballroom [27] scene. Ambiguous foundation-model geometry leads to incorrect matching and degraded reconstruction.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main claims, including dense matching with a frozen 3D foundation model, sliding-window association, global optimization, and evaluation on diverse benchmarks.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations in both main paper and supplemental material, including the fixed window size and the dependence of matching quality on the geometric predictions of the underlying foundation model.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present theoretical results, formal theorems, or proof-based claims. The method is an empirical reconstruction framework, and the mathematical formulations are used to define the optimization objectives and losses.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper and supplementary material describe the architecture, losses, training data, inference settings, baselines, datasets, and evaluation protocols needed to reproduce the main results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The code of the paper will be publicly available after acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the training datasets, optimization settings, learning rates, training schedule, inference window size, thresholds, baseline settings, and evaluation metrics.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report the mean results of the evaluation. Since all methods undergo the same evaluation on the benchmarks, the statistical significance of the experiments is properly demonstrated.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The supplementary material reports the main training and inference hardware, including GPU type, memory, training duration, and optimization implementation details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work follows the NeurIPS Code of Ethics and uses public datasets and standard computer vision benchmarks.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The paper discusses the impacts for both community and the social. The principle mentioned in the paper will benefit the 3DV community for future discovery.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: This paper only use open-source datasets for training and testing.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All existing datasets, models, and codebases used in the paper are cited, and their use follows the corresponding public research licenses and terms.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce a new dataset or benchmark asset.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human-subject experiments and therefore does not require IRB approval.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not used as an important, original, or non-standard component of the core method.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.