This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

LASER: Layer-wise Scale Alignment for Training-Free Streaming 4D Reconstruction

Tianye Ding[1*] Yiming Xie[1*] Yiqing Liang[2*] Moitreya Chatterjee[3] Pedro Miraldo[3] Huaizu Jiang[1]
[1] Northeastern University [2] Independent Researcher [3] Mitsubishi Electric Research Laboratories

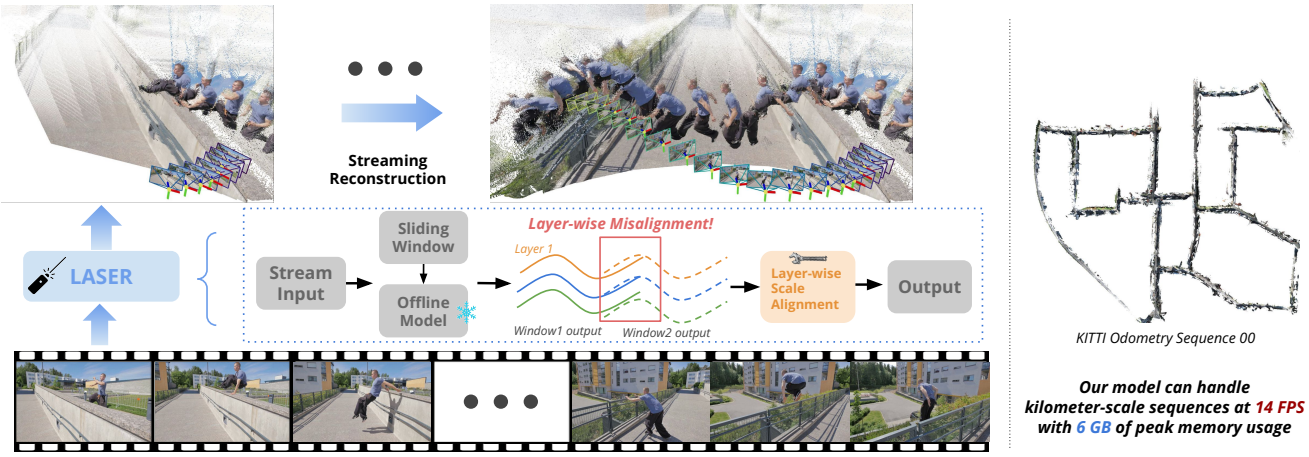


Figure 1. **LASER** transforms offline reconstruction models into streaming systems via a sliding-window approach without retraining. Our submap registration and layer-wise scale alignment modules seamlessly align windows into a globally consistent reconstruction.

Abstract

Recent feed-forward reconstruction models like VGGT and π^3 achieve impressive reconstruction quality but cannot process streaming videos due to quadratic memory complexity, limiting their practical deployment. While existing streaming methods address this through learned memory mechanisms or causal attention, they require extensive retraining and may not fully leverage the strong geometric priors of state-of-the-art offline models. We propose LASER, a training-free framework that converts an offline reconstruction model into a streaming system by aligning predictions across consecutive temporal windows. We observe that simple similarity transformation ($Sim(3)$) alignment fails due to layer depth misalignment: monocular scale ambiguity causes relative depth scales of different scene layers to vary inconsistently between windows. To address this, we introduce layer-wise scale alignment, which segments depth predictions into discrete layers, computes per-layer scale factors, and propagates them across both adjacent windows and timestamps. Extensive experiments show that LASER achieves state-of-the-art performance on camera pose estimation and point map reconstruction while operating at 14 FPS with 6 GB peak memory on a RTX A6000 GPU, enabling practical deployment for kilometer-scale streaming videos. Project website: https://neu-vi.github.io/LASER/

1. Introduction

Recovering 3D scene geometry from images has long been a central pursuit in computer vision, with applications ranging from robotic perception to digital cultural preservation. For decades, this problem was addressed through geometry-centric pipelines: Structure-from-Motion (SfM) [18, 48] and Multi-View Stereo (MVS) [14, 49] systems with handcrated designs. While these classical methods achieve impressive accuracy with careful engineering, they remain sensitive to textureless regions and require known or estimated camera calibration. Recent advent of feed-forward neural approaches have fundamentally changed this landscape. DUSt3R [63] pioneers direct regression of dense pointmaps from uncalibrated image pairs, eliminating the need for explicit correspondence solving. Subsequent work including VGGT [59] and π^3 [64] further handle arbitrary numbers of views, establishing a new paradigm where large-scale transformers trained on diverse data achieve superior reconstruction quality in zero-shot settings.

* Equal contribution

While these offline models achieve excellent reconstruction quality, they struggle with streaming scenarios due to quadratic memory complexity and the need to reprocess all frames when new observations arrive. Several recent works have proposed *streaming variants* that process frames incrementally. Some approaches [7, 57, 62] introduce persistent state or memory mechanisms for continuous 3D perception. Another line of works [24, 27, 73] adapt offline models with causal attention or combine sliding window with camera token pools. Though effective, these methods share a common limitation: they require extensive retraining from scratch or through knowledge distillation to learn streaming-setting reconstruction, which is computationally expensive and may not fully leverage the strong geometric priors of state-of-the-art offline models like VGGT [59] and π^3 [64]. Moreover, recurrent designs like CUT3R [62] can suffer from drift and catastrophic forgetting over long sequences [6], while methods relying on growing memory face scalability constraints. Concurrent work VGGT-Long [9] also pursues a training-free approach by chunking sequences and aligning with $Sim(3)$, but as we show in Sec. 4, simple rigid alignment is insufficient. Given that offline models already encode rich 3D priors, we ask: *can we achieve both training-free conversion for streaming input and robust geometric alignment?*

In this work, we propose **LASER**, a training-free framework that converts offline models into streaming systems by revisiting classical geometric principles. LASER employs a sliding-window strategy, processing overlapping subsets of frames (windows) sequentially with a frozen offline model as the backbone. Modern feed-forward models like VGGT [59] and π^3 [64] excel at producing accurate 3D reconstructions within individual windows. However, *aligning these windows consistently* remains challenging. We observe that simple $Sim(3)$ alignment fails due to *layer depth misalignment*: monocular scale ambiguity causes relative depth scales across scene layers (*e.g.*, foreground *vs*. background) to vary between windows, particularly when camera translation is limited. A global $Sim(3)$ transformation applies uniform scaling to the entire window and cannot resolve such layer-wise scale variations. Drawing on classical insights that scenes naturally decompose into depth-ordered layers with distinct geometric properties [1, 50, 60], we propose *layer-wise scale alignment* adapted to the modern deep learning reconstruction setting. Our approach segments reconstructed point maps into discrete layers using [11], computes per-layer scale factors between consecutive windows, and propagates these scales across the sequence to achieve layer-consistent alignment.

Experimental results show that our design effectively addresses the aforementioned challenges and achieves state-of-the-art performance. LASER outperforms the learned streaming methods while processing image streams at 14 FPS with only 6 GB peak memory on one RTX A6000 GPU. Notably, our training-free approach maintains competitive reconstruction quality with offline models ($0.013m$ vs $0.011m$ mean accuracy on 7-Scenes [51]) while enabling online processing. This shows that when deep learning models provide strong local geometry, classical layer-based geometric reasoning can effectively unify their outputs into consistent long-range reconstructions without retraining. Beyond quantitative gains, LASER offers significant practical advantages: it requires no model retraining, immediately applies to existing offline reconstruction models (as demonstrated with both VGGT [59] and π^3 [64] backbones), and scales to kilometer-long sequences that exceed the memory capacity of offline methods. As new, more powerful offline models emerge, LASER can immediately leverage their improvements without additional training costs, bridging the gap between the quality and efficiency of offline models and the streaming requirements.

Our main contributions are summarized as follows:

- We propose LASER, a training-free framework for converting offline reconstruction models into streaming systems without retraining, (*e.g.*, VGGT [59], π^3 [64]).
- We identify the layer depth misalignment problem arising from monocular scale ambiguity and propose layer-wise scale alignment to address it.
- LASER achieves state-of-the-art performance in streaming pose estimation and 3D reconstruction (-68.6% ATE on Sintel [2] pose estimation, -63.9% Acc on 7-Scenes [51] reconstruction compared to the previous best) while operating at 14 FPS on an A6000 GPU with only 6 GB peak runtime memory.

2. Related Work

Learning-based 3D Reconstruction. Learning-based methods recast 3D reconstruction as a data-driven estimation problem rather than a purely geometric one. Early CNN-based pipelines [19, 20, 70] replace handcrafted correspondence matching with learned feature aggregation and differentiable depth regression, paving the way toward end-to-end multi-view geometry learning. Subsequent methods [40, 53] extend these ideas to online or large-scale reconstruction via recurrent fusion and volumetric integration. Implicit-field formulations [37, 61, 71] achieve photorealistic surface and appearance modeling from sparse or monocular inputs, while explicit representations [4, 23] further improve rendering efficiency and scalability.However, these models typically require per-scene optimization and are bounded by scene complexities [30]. A parallel line of research seeks to eliminate the optimization through feed-forward geometric reasoning. DUSt3R [63] pioneers a paradigm where 3D point clouds and relative poses are directly regressed from image pairs, and VGGT [59] generalizes this idea to variable-sized image sets with the help

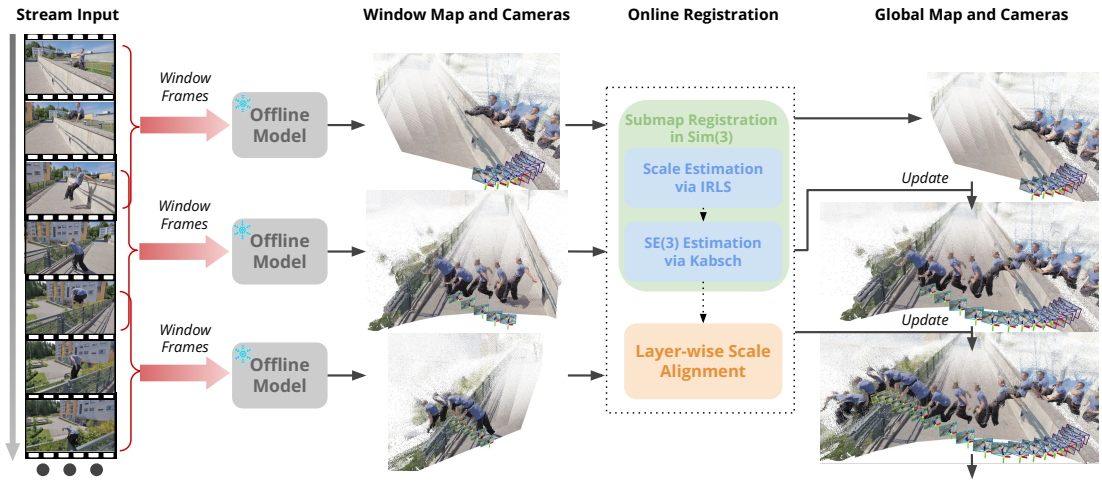


Figure 2. **Overview of LASER.** LASER converts an offline reconstruction model to a streaming version without retraining. Given a video stream, we process frames in overlapping temporal windows with a frozen feed-forward reconstructor. We incrementally register the submap to the global map with $\mathrm{Sim}(3)$ estimation and the proposed layer-wise scale alignment.

of learnable camera tokens. The recent π^3 model [64] further introduces permutation-equivariant attention to unify structure and motion within a single scalable framework. Our proposed LASER builds on this feed-forward foundation, introducing a lightweight streaming formulation that adapts offline reconstruction models for continuous, efficient, kilometer-scale processing without retraining.

Learning-based 4D Reconstruction. Learning-based 4D reconstruction approaches extend geometric and appearance modeling to temporal domain. Early progress built on implicit neural representations [25, 43, 44, 46] introduces temporal conditioning or deformation fields. To improve efficiency, subsequent approaches [3, 10, 13, 29, 32, 34, 65, 69] apply similar extensions to explicit representations. Both implicit and explicit 4D representations require per-scene optimization , limiting usage in streaming settings. Another series of works [5, 17, 33, 72] extend feed-forward regression to dynamic settings. [12, 21, 26, 28] leverage dense video correspondence supervision to generalize to diverse dynamic scenes. π^3 [64] unifies 3D and 4D reasoning through permutation-equivariant attention during large-scale training. Our method shares the goal of scalable 4D reconstruction but focuses on the *streaming* regime: adapting offline feed-forward geometry transformers for causal video processing. With a sliding-window formulation and geometry-aware alignment, LASER achieves temporally consistent reconstruction efficiently without retraining.

Streaming Feed-Forward Reconstruction. Real-world applications such as autonomous driving, robotics, and AR/VR require models that process video streams efficiently and consistently. Recent works have tried to fine-tune offline feed-forward reconstructor to the streaming regime: memory-centric approaches introduce persistent or explicit spatial memory to extend temporal horizons [58, 62, 66]; causal-transformer designs process frames sequentially with token pooling or causal attention [24, 27, 73]; test-time adaptation has also been explored for long videos [6]; parallel efforts bridge feed-forward prediction with SLAM-style optimization [35, 39]. Aside from requiring retraining, these methods either accumulate scale drift over time, fail on long sequences due to memory limits or incur slow inference. On the other hand, [68] pushes feed-forward reconstruction to the kilo-frame regime, but does not support streaming input. LASER addresses these limitations with a general, training-free framework that turns offline feed-forward models (*e.g.*, VGGT or π^3) into a streaming system capable of handling kilo-frame dynamic sequences while preserving their reconstruction quality and efficiency.

3. Method

3.1. Overview

Our goal is to convert an offline 4D reconstruction model to a streaming version *without retraining*. Fig. 2 illustrates our pipeline. Given a video stream, we process frames in overlapping *temporal windows*. Each window is passed through a frozen feed-forward reconstructor to predict dense point maps (local submaps) and camera poses. We incrementally register the submap of the current window to the global map to complete the streaming 4D reconstruction.

4D reconstruction in temporal windows. Let $\{\mathbf{I}_t\}_{t=1}^{T}$ be a monocular RGB video with T frames where each frame has a spatial dimension of $H \times W$. We form overlapping windows $\{\mathcal{W}_i\}_{i=1}^{T/L}$, where each window contains L consecutive frames. Let a_i denote the start index of the frame of the i-th window; then $\mathcal{W}_i = \{t | a_i \leq t < a_i + L\}$, where $a_1 = 1$. Two adjacent windows share an overlap of O frames, *i.e.*, $a_{i+1} = a_i + L - O$.

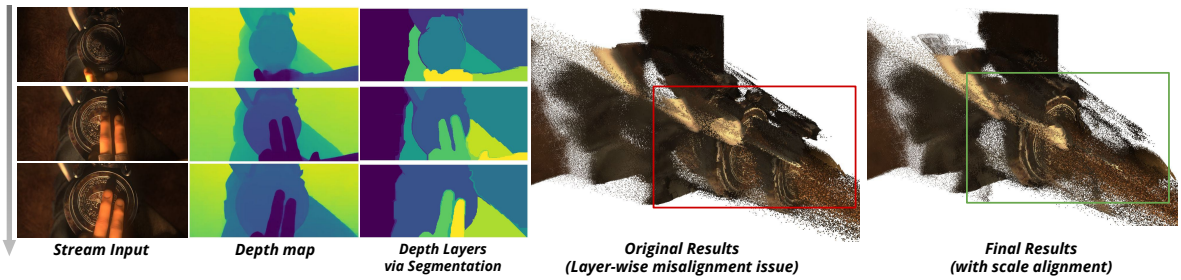


Figure 3. **Layer Depth Misalignment Issue**. After the global $\mathrm{Sim}(3)$ alignment, surfaces at different depths may exhibit *layer-wise scale inconsistency*: foreground regions appear over- or under-scaled relative to background structures across consecutive windows. This anisotropic scaling leads to visible distortions and metric drift in the fused reconstruction. We introduce Layer-wise Scale Alignment (LSA), a geometry-driven refinement that corrects distortions based on a layer graph.

For each window $\mathcal{W}_i$, a pretrained feed-forward reconstructor $f(\cdot)$ (*e.g.*, VGGT [59], π^3 [64]) predicts per-frame point maps and camera poses:

$$f(\{\mathbf{I}_t\}, \mathcal{W}_i) = \{(\mathbf{P}_t^{(i)}, \mathbf{T}_t^{(i)}, \mathbf{C}_t^{(i)}) \mid a_i \leq t < a_i{+}L\}, \quad (1)$$

where $\mathbf{P}_t^{(i)} \in \mathbb{R}^{H\times W\times 3}$ is a dense 3D point map in the window's local coordinates and $\mathbf{T}_t^{(i)} = (\mathbf{R}_t^{(i)}|\mathbf{t}_t^{(i)})$ are the camera poses, consisting of a rotation matrix $\mathbf{R}_t^{(i)} \in SO(3)$ and a translation vector $\mathbf{t}_t^{(i)} \in \mathbb{R}^3$, defined in the window's coordinate system. The reconstructor also outputs pixel-wise confidence scores $\mathbf{C}_t^{(i)}$, which are used to form a set of mutually confident correspondences for scale estimation. Based on $\mathbf{P}_t^{(i)}$ and $\mathbf{T}_t^{(i)}$, we construct the local submap $\mathcal{S}_i$ for window $\mathcal{W}_i$ by transforming all per-frame point maps into the window's coordinate system.

Incremental global map reconstruction in the $\mathrm{Sim}(3)$ space. 4D reconstruction in each window $\mathcal{W}_i$ yields a local submap $\mathcal{S}_i = \{\mathbf{T}_t^{(i)}\mathbf{P}_t^{(i)}\}_{t\in\mathcal{W}_i}$ in the window's own coordinate system. We then estimate a similarity transform $(s_i^w, \mathbf{R}_i^w, \mathbf{t}_i^w) \in \mathrm{Sim}(3)$ between $\mathcal{S}_i$ to $\mathcal{G}_{i-1}$, which are defined in the world coordinate system (in our case, the first temporal window's coordinate system, based on the estimated point maps of the overlapping region. The induced camera pose in the world space for a frame $\mathbf{I}_{t\in\mathcal{W}_i}$ is $\mathbf{T}_t^w = (\mathbf{R}_i^w\mathbf{R}_t^{(i)} \mid s_i^w\mathbf{R}_i^w\mathbf{t}_t^{(i)} + \mathbf{t}_i^w)$ The global map $\mathcal{G}_i$ is then updated progressively as $\mathcal{G}_i = \mathcal{G}_{i-1} \cup \{\mathbf{T}_t^w\mathbf{P}_t^{(i)}\}$, where $\mathcal{G}_0 = \emptyset$ in the initialization.

To estimate the $\mathrm{Sim}(3)$ transform, we first estimate the global scale factor s_i^w via a robust IRLS (Iteratively Reweighted Least Squares) optimization [41], enforcing a shared metric across two adjacent windows. Rotation and translation $(\mathbf{R}_i^w, \mathbf{t}_i^w)$ are then optimized via the Kabsch algorithm [22] under that metric using the *scaled* camera anchors based on the estimated s_i^w. We refer readers to the supplementary material for more details.

3.2. Layer-wise Scale Alignment (LSA)

Although the global $\mathrm{Sim}(3)$ registration aligns each window to a common scale, it assumes isotropic scaling, where the same scale factor applies equally along all spatial axes. In practice, this assumption breaks, *e.g.*, under low-parallax motion, where a monocular reconstructor cannot reliably constrain depth (the Z-axis) relative to lateral axes. As a result, even after the global alignment, surfaces at different depths may exhibit *layer-wise scale inconsistency*: foreground regions appear over- or under-scaled relative to background structures across windows, as shown in Fig. 3. This anisotropic scaling along depth accumulates over time, leading to visible distortions and metric drift in the fused reconstruction. Following classical insights that scenes decompose into depth-ordered layers [1, 50], we introduce *Layer-wise Scale Alignment (LSA)*, a geometry-driven refinement that corrects distortions based on a layer graph.

Depth layer extraction. Inspired by classical layered representations, where a scene is decomposed into depth-ordered surfaces [1, 50], we extract depth layers by segmenting each depth map into spatially coherent regions at similar depths. Specifically, let $\bar{\mathbf{P}}_t^{(i)} = \mathbf{T}_t^w\mathbf{P}_t^{(i)} \in \mathbb{R}^{H\times W\times 3}$ denote the 3D point map after the $\mathrm{Sim}(3)$ registration for frame $\mathbf{I}_t$ in the temporal window $\mathcal{W}_i$. We derive a pseudo-depth map, denoted as $\bar{\mathbf{D}}_t^{(i)}$, from $\bar{\mathbf{P}}_t^{(i)}$ by taking its Z-coordinate components. This pseudo-depth map is partitioned into $M_t^{(i)}$ disjoint depth layers $\{\mathcal{L}_{t,m}^{(i)}\}_{m=1}^{M_t^{(i)}}$ using an efficient segmentation algorithm [11]. Each layer $\mathcal{L}_{t,m}^{(i)}$ corresponds to a continuous geometric surface patch with a coherent depth. Examples are shown in Fig. 3.

Depth layer graph construction. Let $\mathcal{O}_i = \mathcal{W}_{i-1} \cap \mathcal{W}_i$ be the set of overlapping timestamps. To enforce consistent scaling between overlapping windows and across time, we organize all depth layers into a *directed graph* $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, where the vertices correspond to the depth layers $\{\mathcal{L}_{t,m}^{(i-1)}\}_{t\in\mathcal{O}_i}$ and $\{\mathcal{L}_{t,n}^{(i)}\}_{t\in\mathcal{W}_i}$. The edges $\mathcal{E}$ contains both inter-window and intra-window edges:

$$\mathcal{E}_{\text{inter}} = \{(\mathcal{L}_{t,m}^{(i-1)}, \mathcal{L}_{t,n}^{(i)}) \mid \mathrm{IoU}(\mathcal{L}_{t,m}^{(i-1)}, \mathcal{L}_{t,n}^{(i)}) > \tau, t \in \mathcal{O}_i\},$$
$$\mathcal{E}_{\text{intra}} = \{(\mathcal{L}_{t-1,m}^{(i)}, \mathcal{L}_{t,n}^{(i)}) \mid \mathrm{IoU}(\mathcal{L}_{t-1,m}^{(i)}, \mathcal{L}_{t,n}^{(i)}) > \tau, t \in \mathcal{W}_i\},$$

with $\tau = 0.3$. $\mathcal{E}_{\text{inter}}$ link layers at overlapping timestamps

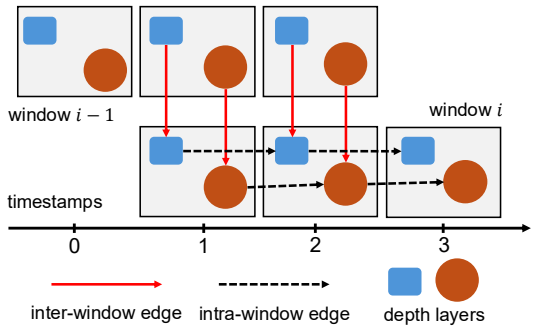


Figure 4. **Layer-wise Scale Alignment (LSA)**. We use a toy example to illustrate our proposed LSA.

between windows $\mathcal{W}_{i-1}$ and $\mathcal{W}_i$, where the same geometric surface patch may appear under different scales due to independent per-window reconstruction. Generally, the depth maps $\bar{\mathbf{D}}_t^{(i-1)}$ and $\bar{\mathbf{D}}_t^{(i)}$ for the same image $\mathbf{I}_t$ in two windows are very close and the depth layers are almost identical. Therefore, a single depth layer in $\mathcal{W}_{i-1}$ will exactly be matched to another layer in $\mathcal{W}_i$. $\mathcal{E}_{\text{intra}}$ connects the same depth layer across adjacent frames within window $\mathcal{W}_i$, encoding geometric continuity over time. An illustration is shown in Fig. 4. Note that the edges are directed, pointing from a parent node to a child node across either the temporal windows or the timestamps. With this graph, we first estimate layer-wise scales based on $\mathcal{E}_{\text{inter}}$ for the layers in the overlapping region. The corrected scales will then be propagated and aggregated across both $\mathcal{E}_{\text{inter}}$ and $\mathcal{E}_{\text{intra}}$.

Layer-wise scale estimation via IRLS across inter-window edges. We estimate the layer-wise scales from point-wise correspondences in the intersection of two consecutive windows. Specifically, we construct a set of correspondences $\mathcal{C}_{t,n}^{(i)} = \{(d_p, d_q)\}$, where $d_p = \bar{\mathbf{D}}_t^{(i-1)}(x)$ and $d_q = \bar{\mathbf{D}}_t^{(i)}(x)$[1], by using all depth values from pixel coordinate x within the intersection of two layers $\mathcal{L}_{t,m}^{(i-1)} \cap \mathcal{L}_{t,n}^{(i)}$. We then find the optimal scale for the layer $\mathcal{L}_{t,n}^{(i)}$ by solving the following objective using IRLS.

$$\hat{s}_{t,n}^{(i)} = \arg\min_{s>0} \sum_{(d_p,d_q)\in\mathcal{C}_{t,n}^{(i)}} \rho\big(\| s\, d_p - d_q \|\big), \qquad (2)$$

where $\rho(\cdot)$ is the Huber loss.

Scale propagation and aggregation along all the edges. After optimizing the layer-wise scales across inter-window edges $\mathcal{E}_{\text{inter}}$, for each layer $\mathcal{L}_{t,n}^{(i)}$, the scales from its parent nodes along both $\mathcal{E}_{\text{inter}}$ and $\mathcal{E}_{\text{intra}}$ are propagated to it. The layer $\mathcal{L}_{t,n}^{(i)}$ may thus receive multiple scales, which will be aggregated in a weighted average manner. The weight for each edge is defined as the IoU score of two connected layers. This procedure is summarized in Algorithm 1. Please refer to the supplementary material for more elaboration.

[1]We omit the symbol x in d_p and d_q to avoid notation clutter.

Algorithm 1 Layer-wise Scale Alignment

Require: Layer graph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ with vertices $\mathcal{V} = \{\mathcal{L}_{t,m}^{(i-1)}\} \cup \{\mathcal{L}_{t,n}^{(i)}\}$ and edges $\mathcal{E} = \mathcal{E}_{\text{inter}} \cup \mathcal{E}_{\text{intra}}$; temporal window $\mathcal{W}_{i-1}$ and $\mathcal{W}_i = \{t\}_{t=a_i}^{a_i+L}$.

Ensure: Final scales $\{s_{t,n}^{(i)}\}$ for layers $\{\mathcal{L}_{t,n}^{(i)}\}$.

Initialize $A_{t,n}^{(i)} \leftarrow 0$ and weight $W_{t,n}^{(i)} \leftarrow 0$ for all $\mathcal{L}_{t,n}^{(i)}$.
# inter-window scale optimization and propagation
for every $(\mathcal{L}_{t,m}^{(i-1)}, \mathcal{L}_{t,n}^{(i)}) \in \mathcal{E}_{\text{inter}}$ **do**
  compute $\hat{s}_{t,n}^{(i)}$ according to Eq.(2)
  $w \leftarrow \text{IoU}(\mathcal{L}_{t,m}^{(i-1)}, \mathcal{L}_{t,n}^{(i)})$
  $A_{t,n}^{(i)} \leftarrow A_{t,n}^{(i)} + w \cdot \hat{s}_{t,n}^{(i)}$
  $W_{t,n}^{(i)} \leftarrow W_{t,n}^{(i)} + w$
end for
# temporal propagation along intra-window edges
for $t = a_i + 1$ **to** $a_i + L$ **do**
  for every $(\mathcal{L}_{t-1,m}^{(i)}, \mathcal{L}_{t,n}^{(i)}) \in \mathcal{E}_{\text{intra}}$ **do**
    if $W_{t-1,m}^{(i)} > 0$ **then**
      $\mu_{t-1,m}^{(i)} \leftarrow A_{t-1,m}^{(i)} / W_{t-1,m}^{(i)}$ # parent mean
      $w \leftarrow \text{IoU}(\mathcal{L}_{t-1,m}^{(i)}, \mathcal{L}_{t,n}^{(i)})$
      $A_{t,n}^{(i)} \leftarrow A_{t,n}^{(i)} + w \cdot \mu_{t-1,m}^{(i)}$
      $W_{t,n}^{(i)} \leftarrow W_{t,n}^{(i)} + w$
    end if
  end for
end for
# weighted average as the final scale for each layer
for every layer $\mathcal{L}_{t,n}^{(i)}$ **do**
  $s_{t,n}^{(i)} \leftarrow A_{t,n}^{(i)} / W_{t,n}^{(i)}$ **if** $W_{t,n}^{(i)} > 0$ **else** 1
end for

Such layer-wise scale propagation and aggregation ensure the consistency across both adjacent windows and the temporal axis. Once the layer-scale optimization is finished, each layer's scale will be propagated to its contained pixels, which will be used to adjust the reconstructed point map $\bar{\mathbf{P}}_t^{(i)}$. As shown in Fig. 3, it can effectively mitigate the distortions in the 4D reconstruction.

4. Experiments

We evaluate our proposed method, LASER, against state-of-the-art approaches across three tasks: video depth estimation (Sec. 4.2), camera pose estimation (Sec. 4.3), and multi-view point map estimation (Sec. 4.4) with details of the experimental setup in Sec. 4.1. Additional qualitative results are shown in Sec. 4.5. We further analyze model efficiency (Sec. 4.6) and perform ablation studies to assess the contribution of each component (Sec. 4.7).

Table 1. **Video Depth Estimation on Sintel [2], Bonn [42] and KITTI [16].** We report Abs Rel and δ<1.25.

Method	Stream	Sintel		Bonn		KITTI	
		Abs Rel ↓	δ<1.25 ↑	Abs Rel ↓	δ<1.25 ↑	Abs Rel ↓	δ<1.25 ↑
VGGT [59]	✗	0.303	**68.5**	0.055	97.1	0.073	96.3
π^3 [64]	✗	**0.245**	68.4	**0.050**	**97.5**	**0.038**	**98.6**
Spann3R [58]	✓	0.622	42.6	0.144	81.3	0.198	73.7
CUT3R [62]	✓	0.421	47.9	0.078	93.7	0.118	88.1
Point3R [66]	✓	0.452	48.9	0.060	96.0	0.136	84.2
VGGT-SLAM [35]	✓	0.424	56.0	0.076	93.2	0.136	81.8
StreamVGGT [73]	✓	0.323	65.7	<u>0.059</u>	<u>97.2</u>	0.173	72.1
STream3Rβ [24]	✓	0.264	**70.5**	0.069	95.2	<u>0.080</u>	<u>94.7</u>
WinT3R [27]	✓	0.374	50.6	0.070	91.2	0.081	94.9
TTT3R [6]	✓	0.404	50.0	0.068	95.4	0.113	90.4
VGGT+Ours	✓	0.297	64.6	0.07	92.6	0.116	88.4
π^3+Ours	✓	**0.247**	<u>68.8</u>	**0.048**	**97.4**	**0.054**	**98.3**

4.1. Experimental Setup

4.1.1. Tasks, datasets, and metrics

Video depth estimation protocol. Following [62, 72], we evaluate on Sintel [2], Bonn [42], and KITTI [16]. Predicted depths are aligned to ground truth via *scale-only* alignment.

Camera pose estimation protocol. Following [62, 72], we compare on small-scale Sintel [2], ScanNet [8], and TUM RGB-D [52]. Predicted trajectories are aligned to ground truth via a *Sim(3)* transformation. For large-scale evaluation, we use KITTI Odometry [16] following [9].

Multi-view point map estimation protocol. We run evaluation on 7-Scenes [51] and NRGBD [67] with a keyframe sampling interval of 10, except for using an interval of 15 on NRGBD for [24, 73] due to their memory limits. Predicted point maps are registered to ground truth using the Umeyama algorithm (coarse *Sim(3)* alignment) followed by Iterative Closest Point (ICP) refinement.

4.1.2. Baselines

Offline feed-forward models. We include DUSt3R [63], Fast3R [68], VGGT [59], and π^3 [64], which process static image batches without temporal constraints.

Streaming or online feed-forward methods. We include Spann3R [58], CUT3R [62], MASt3R-SLAM [39], Point3R [66], VGGT-SLAM [35], StreamVGGT [73], STream3Rβ [24], WinT3R [27], and TTT3R [6], which enable causal inference or maintain persistent memory.

Classical SLAM systems. For camera pose evaluation on KITTI Odometry, we compare to SLAM methods including ORB-SLAM2 [38], LDSO [15], DROID-VO, DROID-SLAM [55], DPV-SLAM, and DPV-SLAM++ [31].

Training-free concurrent work. To further demonstrate the strength of our training-free design and ensure that performance is not dominated by a strong backbone, we also evaluate against VGGT-Long [9], a concurrent training-free streaming framework built on VGGT [59]. For a fair comparison, we re-implement its pipeline on the π^3 [64] backbone, denoted as π^3-Long, enabling a one-to-one comparison under identical base models.

4.1.3. Implementation Details

We instantiate LASER using either VGGT [59] or π^3 [64] as the offline 4D reconstruction backbone. On the kilometer-scale KITTI Odometry, we incorporate loop closure following the VGGT-Long [9] configuration for fair comparison. More details are in the supplemental material.

4.2. Video Depth Estimation

Tab. 1 shows the video depth estimation results. For streaming methods, a slight degradation on performance is reasonable compared with offline methods, since streaming methods operate with limited temporal context which leads to inconsistent depth predictions.

Compared to prior streaming baselines such as CUT3R [62], StreamVGGT [73], and STream3Rβ [24], LASER achieves the lowest Abs Rel across all three datasets when compared to the best-performing baseline on each, as well as the highest δ<1.25 accuracy on Bonn and KITTI , while ranking second on Sintel. Across all datasets, LASER maintains the performance of its offline backbones VGGT [59] and π^3 [64] while operating in the streaming setting. These results demonstrate that LASER delivers high-fidelity depth estimation across diverse domains while operating fully in a streaming manner. Note that many baseline methods, such as StreamVGGT, Stream3Rβ, and VGGT-SLAM, also build upon offline approaches, *yet their performance degrades significantly compared to the offline counterparts.*

4.3. Camera Pose Estimation

Tab. 2 reports results on small-scale datasets. On all three datasets, LASER (π^3) achieves the best results in almost all metrics and even surpasses its offline backbones in several cases. This demonstrates the effectiveness of our framework in pose estimation. LASER (VGGT) consistently ranks second across all metrics, further validating the generality and robustness of our framework across backbones.

On large-scale outdoor sequences (Tab. 3), LASER using π^3 as backbone achieves the second-lowest mean ATE among all methods on both Avg. and Avg.* metrics. It attains accuracy comparable to or better than well-designed SLAM systems such as ORB-SLAM2 [38] and DROID-SLAM [55], which yield only sparse reconstructions and may require camera calibration. Meanwhile, dense offline models like VGGT [59] and π^3 [64] fail to process long sequences due to memory limits, and streaming variants such as CUT3R [62] and MASt3R-SLAM [39] either run out of memory or lose tracking. In contrast, LASER remains stable across all eleven sequences, producing globally consistent trajectories. LASER also outperforms training-free streaming concurrent work VGGT-Long [9] and its variant π^3-Long by a 12–21% reduction in avg ATE. Notably, although π^3-Long shares the same backbone with ours, it per-

Table 2. **Camera Pose Estimation on Sintel [2], ScanNet [8], and TUM [52].** We report ATE, translational RPE, and rotational RPE.

Method	Stream	Sintel			ScanNet			TUM		
		ATE ↓	RPE_{trans} ↓	RPE_{rot} ↓	ATE ↓	RPE_{trans} ↓	RPE_{rot} ↓	ATE ↓	RPE_{trans} ↓	RPE_{rot} ↓
DUSt3R [63]	✗	0.290	0.132	7.869	0.246	0.108	8.210	0.140	0.106	3.286
VGGT [59]	✗	0.171	0.062	0.471	0.035	0.015	0.381	0.012	0.010	0.309
π^3 [64]	✗	**0.073**	**0.037**	**0.287**	**0.030**	**0.012**	**0.346**	**0.014**	**0.009**	**0.307**
Spann3R [58]	✓	0.329	0.110	4.471	0.096	0.023	0.661	0.056	0.021	0.591
CUT3R [62]	✓	0.213	0.066	0.621	0.099	0.022	0.600	0.046	0.015	0.473
Point3R [66]	✓	0.351	0.128	1.822	0.106	0.035	1.946	0.075	0.029	0.642
VGGT-SLAM [35]	✓	0.303	0.128	6.883	0.070	0.049	2.447	0.030	0.020	1.567
StreamVGGT [73]	✓	0.251	0.149	1.894	0.161	0.057	3.647	0.061	0.033	3.209
STream3Rβ [24]	✓	0.213	0.076	0.868	0.052	0.021	0.850	0.026	0.013	0.330
WinT3R [27]	✓	0.225	0.097	1.092	0.062	0.020	0.690	0.074	0.023	0.774
TTT3R [6]	✓	0.201	0.063	0.617	0.064	0.021	0.592	0.028	0.012	0.379
VGGT+Ours	✓	0.131	0.053	0.398	0.035	0.014	0.354	**0.013**	0.010	**0.306**
π^3+Ours	✓	**0.061**	**0.028**	**0.249**	**0.031**	**0.012**	**0.339**	0.016	**0.009**	0.308

Table 3. **Large-scale Camera Pose Estimation on KITTI [16].** We report ATE (lower is better). CF: checkmark (✓) indicates no calibration required; DR: checkmark (✓) indicates dense reconstruction supported. We show metrics for sequence ID; *Avg.* is the mean across sequences. Seq. 01 corresponds to a high-speed driving sequence whose motion differs from others; *Avg.** reports the mean ATE excluding Seq. 01. *OOM*: CUDA out-of-memory, *TL*: tracking lost.

Method	CF	DR	Avg.	Avg.*	00	01	02	03	04	05	06	07	08	09	10
ORB-SLAM2 (w/o LC) [38]	✗	✗	69.73	26.48	40.65	502.20	47.82	**0.94**	1.30	29.95	40.82	16.04	43.09	38.77	**5.42**
ORB-SLAM2 (w/ LC) [38]	✗	✗	54.82	**9.46**	**6.03**	508.34	**14.76**	1.02	1.57	**4.04**	**11.16**	2.19	**38.85**	**8.39**	6.63
LDSO [15]	✗	✗	**22.43**	23.50	9.32	11.68	31.98	2.85	1.22	5.10	13.55	2.96	129.02	21.64	17.36
DROID-VO [55]	✗	✓	54.19	51.19	98.43	84.20	108.80	2.58	0.93	59.27	64.40	24.20	64.55	71.80	16.91
DPVO [56]	✗	✗	53.61	57.70	113.21	12.69	123.40	2.09	**0.68**	58.96	54.78	19.26	115.90	75.10	13.63
DROID-SLAM [55]	✗	✓	100.28	75.85	92.10	344.60	107.61	2.38	1.00	118.50	62.47	21.78	161.60	72.32	118.70
DPV-SLAM [31]	✗	✗	53.03	57.19	112.80	**11.50**	123.53	2.50	0.81	57.80	54.86	18.77	110.49	76.66	13.65
DPV-SLAM++ [31]	✗	✗	25.75	27.14	8.30	11.86	39.64	2.50	0.78	5.74	11.60	**1.52**	110.90	76.70	13.70
VGGT [59]	✓	✓	/	/	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*
π^3 [64]	✓	✓	/	/	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*
MASt3R-SLAM [39]	✓	✓	/	/	*TL*	*TL*	*TL*	*TL*	*TL*	*TL*	*TL*	*TL*	*TL*	*TL*	*TL*
CUT3R [62]	✓	✓	/	/	*OOM*	*OOM*	*OOM*	148.07	22.31	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*
Fast3R [68]	✓	✓	/	/	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*
VGGT-Long [9]	✓	✓	27.64	18.28	8.67	**121.17**	**32.08**	6.12	4.23	8.31	5.34	4.63	53.10	41.99	18.37
π^3-Long	✓	✓	30.72	16.45	6.28	173.45	63.92	4.96	1.66	6.11	4.89	3.99	36.08	**14.78**	21.84
Ours (π^3)	✓	✓	**24.17**	**14.42**	**6.14**	121.61	59.87	**2.64**	**1.36**	**2.73**	**2.92**	**2.28**	**33.14**	17.95	**15.20**

Table 5. **Ablation Studies for Layer-wise Scale Alignment (Sec. 3.2).** *wo/ LSA* denotes not using LSA component. *w/ SAM 2* denotes replacing the efficient segmentation algorithm [11] with a recent counterpart SAM 2. *wo/* $\mathcal{E}_{\text{intra}}$ denotes removing the temporal propagation step from LSA.

	Sintel		Bonn	
	Abs Rel ↓	δ<1.25 ↑	Abs Rel ↓	δ<1.25 ↑
Ours	**0.247**	**68.8**	**0.048**	**97.4**
wo/ LSA	0.328	51.4	0.123	85.6
w/ SAM 2 [47]	0.251	67.8	0.051	97.4
wo/ $\mathcal{E}_{\text{intra}}$	0.261	64.7	0.05	97.1

Table 6. **Ablation on LSA (Sec. 3.2)'s IoU threshold τ.**

	Sintel		Bonn	
τ	Abs Rel ↓	δ<1.25 ↑	Abs Rel ↓	δ<1.25 ↑
0.2	0.249	68.7	0.051	94.7
0.3 (default)	0.247	68.8	0.048	97.4
0.4	0.247	68.4	0.048	97.4
0.5	0.249	68.0	0.048	97.4
0.6	0.248	68.0	0.044	97.1

forms worse, indicating that the improvement is from our streaming algorithm design rather than backbone capacity.

4.4. Multi-View Point Map Estimation

Tab. 4 reports short-term multi-view point map estimation results. LASER consistently improves Acc and Comp over prior streaming baselines. While NC of π^3+Ours is slightly lower than that of StreamVGGT or STream3Rβ, this difference arises from the π^3 backbone's limited surface-normal fidelity. Nevertheless, our formulation, despite training-free, also improves NC over the π^3. A similar pattern is observed between VGGT+Ours and VGGT. These results indicate that our online integration produces smoother, more coherent surface orientations than the backbone.

4.5. Qualitative Results

We present qualitative comparisons in Fig. 6. Across all methods, our approach produces noticeably sharper scene geometry and more accurate camera trajectories. The examples cover conditions of fast-motion videos and large-scale outdoor environments, highlighting robustness under diverse viewpoints and motions. These results demonstrate that LASER generalizes well across datasets, delivering dense and stable reconstructions without any retraining or per-scene optimization.

Table 4. **Indoor, Short-term Multi-view Point Map Estimation on 7 scenes and NRGBD.** We report Accuracy (Acc, lower is better), Completeness (Comp, lower is better), and Normal Consistency (NC, higher is better)'s Mean and Median.

		7 scenes						NRGBD					
		Acc↓		Comp↓		NC↑		Acc↓		Comp↓		NC↑	
Method	Stream	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.
VGGT	✗	0.017	0.005	0.024	0.01	0.586	0.633	0.013	0.006	0.011	0.003	0.705	0.837
Pi3	✗	0.011	0.004	0.019	0.008	0.598	0.652	0.012	0.005	0.01	0.003	0.704	0.826
CUT3R	✓	0.036	0.019	0.029	0.008	0.624	0.695	0.135	0.069	0.055	0.012	0.684	0.825
StreamVGGT	✓	0.043	0.022	0.028	0.008	**0.639**	**0.719**	0.082	0.051	0.046	0.013	0.742	0.918
STream3Rβ	✓	0.042	0.012	0.023	0.008	0.631	0.705	0.042	0.014	0.014	0.005	**0.798**	**0.95**
VGGT+Ours	✓	0.021	0.007	0.021	0.007	0.590	0.639	**0.020**	0.011	**0.012**	**0.004**	0.71	0.857
π^3+Ours	✓	**0.013**	**0.005**	**0.017**	**0.006**	0.607	0.665	**0.020**	**0.010**	**0.012**	**0.004**	0.713	0.856

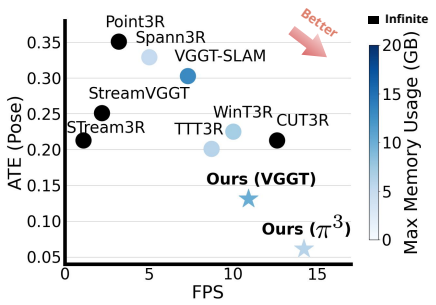
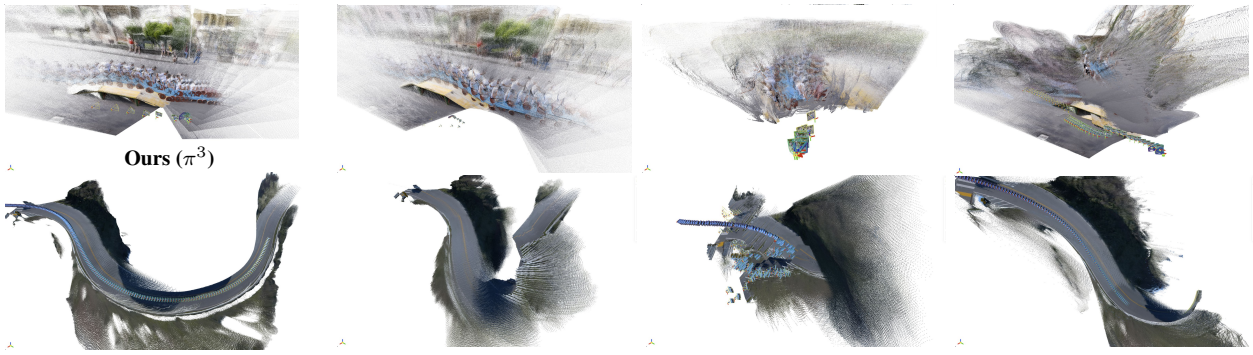


Figure 5. We report the running FPS (on a RTX A6000 GPU), peak memory usage, and pose estimation error (ATE).



Ours (VGGT) **CUT3R** [62] **StreamVGGT** [73] **VGGT-SLAM** [35]

Figure 6. Qualitative comparisons on **DAVIS** [45] and **Hike** [36] (top to bottom). For each sequence, we show the reconstructed global point cloud with the estimated camera trajectory overlaid. *Please zoom in for camera trajectory details*.

4.6. Efficiency Analysis

LASER also demonstrates strong efficiency, as shown in Fig. 5. All experiments are performed on a RTX A6000 GPU. Compared to streaming feed-forward baselines, our method achieves the highest runtime speed (∼14.2 FPS) with only 6 GB of peak memory usage when using π^3 [64] as our offline model, while maintaining superior performance in video depth and camera pose estimation. When using VGGT [59], we also have a competitive inference speed of ∼10.9 FPS and 10 GB peak memory usage.

4.7. Ablation Studies

We evaluate key components of our pipeline through a series of ablations, using π^3 [64] as the backbone.

Layer-wise Scale Alignment (Sec. 3.2). Tab. 5 investigates the components of the LSA module on video depth estimation, as LSA does not affect camera pose estimation. Disabling LSA leads to clear drops in depth accuracy. We also try substituting the segmentation algorithm [11] with SAM 2 [47]. Despite trading speed for better segmentation, SAM 2 does not improve accuracy. Finally, disabling propagation through $\mathcal{E}_{\text{intra}}$ ignores temporal relationships and prevents scale updates in non-overlapping frames, which harms global consistency across long sequences.

Hyperparameters. Fig. 7 and Tab. 6 examine the effect of key hyperparameters: window size L and IoU threshold τ used in LSA. Ours performs robustly under a wide range of settings; the chosen (L=20, τ=0.3) strikes a good balance.

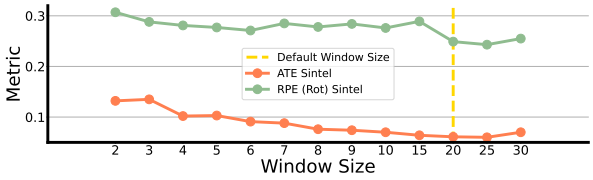


Figure 7. **Ablation on window size L.** In default, we use $L = 20$.

5. Conclusion

We presented LASER, a training-free streaming reconstruction framework that converts an offline 4D reconstruction model into a streaming system. By introducing layer-wise scale alignment, we address the key challenge of inconsistent depth scaling across temporal windows, enabling stable alignment and long-range geometric consistency. Extensive experiments demonstrate that LASER achieves state-of-the-art camera pose estimation accuracy, reconstruction quality, speed and memory budget. We believe this work provides a new angle towards bridging offline and streaming reconstruction. We hope it will inspire future research on integrating classical geometric principles with modern neural architectures for large-scale, continuous 3D perception.

6. Acknowledgment

Tianye Ding and Huaizu Jiang were partially supported by the National Science Foundation under Award IIS-2310254. Yiming Xie was supported by the Apple Scholars in AI/ML PhD fellowship. Pedro Miraldo and Moitreya Chatterjee were supported exclusively by Mitsubishi Electric Research Laboratories.

References

[1] Simon Baker, Richard Szeliski, and P Anandan. A layered approach to stereo reconstruction. In *CVPR*, 1998. 2, 4

[2] DJ Butler, J Wulff, GB Stanley, and MJ Black. A naturalistic open source movie for optical flow evaluation. *ECCV*, 2012. 2, 6, 7

[3] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. *CVPR*, 2023. 3

[4] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *arXiv*, 2021. 2

[5] Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. Easi3r: Estimating disentangled motion from dust3r without training. *arXiv preprint arXiv:2503.24391*, 2025. 3

[6] Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. Ttt3r: 3d reconstruction as test-time training. *arXiv preprint arXiv:2509.26645*, 2025. 2, 3, 6, 7

[7] Zhuoguang Chen, Minghui Qin, Tianyuan Yuan, Zhe Liu, and Hang Zhao. Long3r: Long sequence streaming 3d reconstruction. In *ICCV*, 2025. 2

[8] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 6, 7

[9] 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. 2, 6, 7, 1, 3

[10] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, 2022. 3

[11] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004. 2, 4, 7, 8

[12] Haiwen Feng, Junyi Zhang, Qianqian Wang, Yufei Ye, Pengcheng Yu, Michael J. Black, Trevor Darrell, and Angjoo Kanazawa. St4rtrack: Simultaneous 4d reconstruction and tracking in the world. *arXiv preprint arxiv:2504.13152*, 2025. 3

[13] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *CVPR*, 2023. 3

[14] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009. 1

[15] X. Gao, R. Wang, N. Demmel, and D. Cremers. Ldso: Direct sparse odometry with loop closure. In *iros*, 2018. 6, 7

[16] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 6, 7, 1

[17] Jisang Han, Honggyu An, Jaewoo Jung, Takuya Narihira, Junyoung Seo, Kazumi Fukuda, Chaehyun Kim, Sunghwan Hong, Yuki Mitsufuji, and Seungryong Kim. Enhancing 3d reconstruction for dynamic scenes. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 3

[18] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 1

[19] 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. 2

[20] Mengqi Ji, Juergen Gall, Haitian Zheng, Yebin Liu, and Lu Fang. Surfacenet: An end-to-end 3d neural network for multiview stereopsis. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2307–2315, 2017. 2

[21] Linyi Jin, Richard Tucker, Zhengqi Li, David Fouhey, Noah Snavely, and Aleksander Holynski. Stereo4d: Learning how things move in 3d from internet stereo videos. In *CVPR*, 2025. 3

[22] W. Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 34(5):827–828, 1978. 4, 1

[23] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 2

[24] Yushi Lan, Yihang Luo, Fangzhou Hong, Shangchen Zhou, Honghua Chen, Zhaoyang Lyu, Shuai Yang, Bo Dai, Chen Change Loy, and Xingang Pan. Stream3r: Scalable sequential 3d reconstruction with causal transformer, 2025. 2, 3, 6, 7

[25] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[26] Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. MegaSaM: Accurate, fast and robust structure and motion from casual dynamic videos. In *CVPR*, 2025. 3

[27] 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. 2, 3, 6, 7

[28] Yiqing Liang, Abhishek Badki, Hang Su, James Tompkin, and Orazio Gallo. Zero-shot monocular scene flow estimation in the wild. In *CVPR*, 2025. 3

[29] Yiqing Liang, Numair Khan, Zhengqin Li, Thu Nguyen-Phuoc, Douglas Lanman, James Tompkin, and Lei Xiao. Gaufre: Gaussian deformation fields for real-time dynamic novel view synthesis. In *Proc. IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025. 3

[30] Yiqing Liang, Mikhail Okunev, Mikaela Angelina Uy, Runfeng Li, Leonidas Guibas, James Tompkin, and Adam W Harley. Monocular dynamic gaussian splatting: Fast, brittle, and scene complexity rules. *Transactions on Machine Learning Research*, 2025. Survey Certification. 2

[31] Lahav Lipson, Zachary Teed, and Jia Deng. Deep Patch Visual SLAM. In *European Conference on Computer Vision*, 2024. 6, 7

[32] Xinhang Liu, Yu-Wing Tai, Chi-Keung Tang, Pedro Miraldo, Suhas Lohit, and Moitreya Chatterjee. Gear-nerf: free-viewpoint rendering and tracking with motion-aware spatio-temporal sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19667–19679, 2024. 3

[33] Jiahao Lu, Tianyu Huang, Peng Li, Zhiyang Dou, Cheng Lin, Zhiming Cui, Zhen Dong, Sai-Kit Yeung, Wenping Wang, and Yuan Liu. Align3r: Aligned monocular depth estimation for dynamic videos. In *CVPR*, 2025. 3

[34] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024. 3

[35] Dominic Maggio, Hyungtae Lim, and Luca Carlone. Vggt-slam: Dense rgb slam optimized on the sl (4) manifold. *arXiv preprint arXiv:2505.12549*, 2025. 3, 6, 7, 8, 5

[36] Andreas Meuleman, Yu-Lun Liu, Chen Gao, Jia-Bin Huang, Changil Kim, Min H. Kim, and Johannes Kopf. Progressively optimized local radiance fields for robust view synthesis. In *CVPR*, 2023. 8

[37] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2

[38] Raúl Mur-Artal and Juan D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017. 6, 7

[39] 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 (CVPR)*, pages 16695–16705, 2025. 3, 6, 7

[40] Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-to-end 3d scene reconstruction from posed images. In *ECCV*, 2020. 2

[41] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1977. 4

[42] Emanuele Palazzolo and Stefan Leutenegger. A benchmark for visual-inertial odometry in the presence of motion blur. In *ICRA*, 2019. 6

[43] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021. 3

[44] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Trans. Graph.*, 40(6), 2021. 3

[45] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbelaez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 8

[46] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. *arXiv preprint arXiv:2011.13961*, 2020. 3

[47] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 7, 8

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

[49] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1

[50] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *SIGGRAPH*, 1998. 2, 4

[51] Jamie Shotton et al. Scene coordinate regression forests for camera relocalization in rgb-d images. In *CVPR*, 2013. 2, 6

[52] J"urgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. *IROS*, 2012. 6, 7

[53] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. NeuralRecon: Real-time coherent 3D reconstruction from monocular video. *CVPR*, 2021. 2

[54] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020. 1, 2, 3

[55] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021. 6, 7

[56] Zachary Teed, Lahav Lipson, and Jia Deng. Deep patch visual odometry. *Advances in Neural Information Processing Systems*, 2023. 7

[57] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. *arXiv preprint arXiv:2408.16061*, 2024. 2

[58] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. In *3DV*, 2025. 3, 6, 7

[59] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *CVPR*, 2025. 1, 2, 4, 6, 7, 8, 3

[60] John YA Wang and Edward H Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638, 1994. 2

[61] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *NeurIPS*, 2021. 2

[62] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *CVPR*, 2025. 2, 3, 6, 7, 8, 5

[63] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024. 1, 2, 6, 7, 3

[64] 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. *arXiv preprint arXiv:2507.13347*, 2025. 1, 2, 3, 4, 6, 7, 8

[65] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20310–20320, 2024. 3

[66] Yuqi Wu, Wenzhao Zheng, Jie Zhou, and Jiwen Lu. Point3R: Streaming 3D Reconstruction with Explicit Spatial Pointer Memory, 2025. arXiv:2507.02863 [cs]. 3, 6, 7

[67] Guandao Yang et al. Nrgbd: A large-scale dataset for novel view synthesis and 3d reconstruction from rgb-d images. In *NeurIPS Datasets and Benchmarks*, 2023. 6

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

[69] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023. 3

[70] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European conference on computer vision (ECCV)*, pages 767–783, 2018. 2

[71] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2

[72] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. MonST3r: A simple approach for estimating geometry in the presence of motion. In *ICLR*, 2025. 3, 6

[73] Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming 4d visual geometry transformer. *arXiv preprint arXiv:2507.11539*, 2025. 2, 3, 6, 7, 8, 5