Dense Dynamic Scene Reconstruction and Camera Pose Estimation from Multi-View Videos

Shuo Sun[1], Unal Artan[1], Malcolm Mielle[2], Achim J. Lilienthal[1,3], and Martin Magnusson[1]

[1] Örebro University, Sweden
[2] Schindler EPFL Lab, Lausanne, Switzerland
[3] Technical University of Munich, Germany

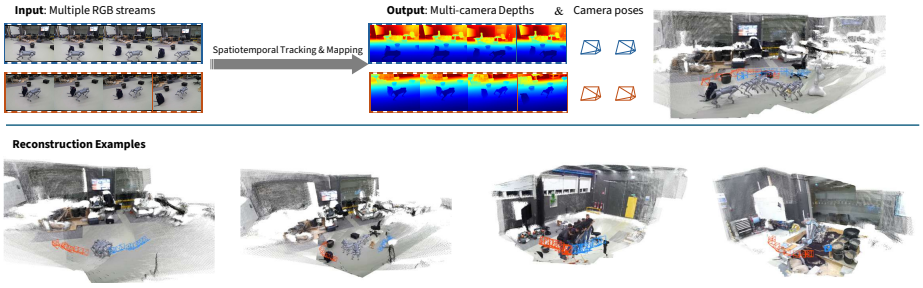


Fig. 1: Given video sequences captured from multiple free cameras, our method can recover dense dynamic scenes consistently and estimate camera poses accurately. We illustrate our complete pipeline (top row) together with reconstruction results on four additional sequences (bottom row), where the red and blue frames indicate cameras.

Abstract. We address the challenging problem of dense dynamic scene reconstruction and camera pose estimation from multiple freely moving cameras—a setting that arises naturally when multiple observers capture a shared event. Prior approaches either handle only single-camera input or require rigidly mounted, pre-calibrated camera rigs, limiting their practical applicability. We propose a two-stage optimization framework that decouples the task into robust camera tracking and dense depth refinement. In the first stage, we extend single-camera visual SLAM to the multi-camera setting by constructing a spatiotemporal connection graph that exploits both intra-camera temporal continuity and inter-camera spatial overlap, enabling consistent scale and robust tracking. To ensure robustness under limited overlap, we introduce a wide-baseline initialization strategy using feed-forward reconstruction models. In the second stage, we refine depth and camera poses by optimizing dense inter- and intra-camera consistency using wide-baseline optical flow. Additionally, we introduce MultiCam-Robolab, a new real-world dataset with ground-truth poses from a motion capture system. Finally, we demonstrate that our method significantly outperforms state-of-the-art feed-forward models on both synthetic and real-world benchmarks, while requiring less memory. Our code is released at https://github.com/ljjTYJR/Multiple-view-Dynamic-Reconstruction.

Keywords: Dynamic Reconstruction · Multi-camera Reconstruction · SLAM in dynamic scenes

1 Introduction

Consider a scenario in which multiple observers simultaneously capture a dynamic scene from diverse viewpoints. Accurately reconstructing dynamic scenes from these observations is challenging due to temporal dynamics and limited view overlap. In this work, we study the problem of dynamic scene reconstruction from multiple free cameras: *Given multiple video inputs, how can we recover dense, dynamic scenes at any time step and estimate camera poses at the same time?*

Multi-camera capture is increasingly common in robotics [12], sports broadcasting [46], and consumer devices where multiple phones or action cameras record a shared event. Applications such as augmented reality, dynamic Gaussian splatting [1, 43], and multi-view video analysis [23] all benefit from accurate, consistent 3D reconstruction across cameras. Yet existing dynamic scene reconstruction methods are limited to single-camera setups [16, 24, 27, 39, 47]. For instance, MegaSAM [24]—a robust monocular visual SLAM—exploits only temporal connections within a single camera, leaving cross-camera consistency unexploited. While multi-camera SLAM methods exist [12, 14, 21, 36, 45], they either assume rigid camera rigs with known calibration or focus on static global map construction, failing to address dynamic scenes with freely moving cameras.

The key technical challenges are threefold: (1) *Scale ambiguity*: monocular depth is inherently scale-ambiguous, and without shared observations, each camera's reconstruction may drift to a different scale; (2) *Limited overlap*: unlike a rigid rig, free-moving cameras may have minimal or intermittent view overlap, complicating inter-camera constraints; (3) *Dynamic content*: moving objects violate the static-world assumption underlying classical multi-view geometry, requiring robust correspondence estimation.

To address these challenges, we introduce a two-stage optimization framework. In the first stage, we extend single-camera SLAM to multiple cameras by constructing a *spatio-temporal connection graph* that links frames across cameras based on view overlap, enabling joint bundle adjustment with consistent scale. To overcome the challenges of multi-camera initialization under limited overlap, we introduce a wide-baseline initialization strategy using a feed-forward reconstruction model. This approach provides a unified scale anchor and provides a reliable starting point for subsequent pose optimization. In the second stage, with coarse camera poses obtained, we refine per-frame depth and camera poses by optimizing dense inter- and intra-camera consistency using dense wide-baseline optical flow correspondences.

Our contributions are as follows: 1) We propose a multi-camera tracking framework that can achieve consistent dynamic scene reconstruction from multiple free-moving cameras. To the best of our knowledge, our work is the first one designed for this task. 2) We collect and make available a new dataset, MultiCamRobolab, enabling quantitative evaluation of methods for multi-camera

dynamic scene reconstruction and camera pose tracking. 3) We demonstrate that our method achieves better tracking and reconstruction results compared to state-of-the-art methods while consuming less memory.

2 Related work

Our method is related to visual SLAM and structure from motion(SfM), two well-established fields for 3D scene reconstruction and camera pose estimation. However, to the best of our knowledge, no existing method explicitly targets the research question addressed in this paper, namely, multi-camera dynamic scene reconstruction. Below, we highlight key advancements and limitations in the state of the art that motivate our contributions.

Visual SLAM and SfM Classical visual SLAM methods [9, 10, 13, 17, 29] are primarily designed for single-view monocular camera configurations. Many works extend this setting to multi-camera configurations. One line of research employs calibrated camera rigs [12,20,26,33], in which multiple cameras are rigidly mounted with known extrinsic parameters. Each camera provides complementary views of the environment, enabling wider field-of-view coverage and improved robustness in challenging scenarios. Another direction leverages multiple cameras in the context of collaborative (multi-agent or multi-session) SLAM [6,8,25,37,45], where each camera (or agent) maintains an independent local mapping process. The individually built static sub-maps are later fused together. Our method differs from the above-mentioned approaches in two primary respects. First, unlike calibrated camera-rig systems, we allow cameras to move freely without requiring known inter-camera extrinsics. Second, in contrast to multi-agent SLAM frameworks, which typically assume static environments, our objective is to reconstruct dense and dynamic scenes.

Structure from motion (SfM) processes unordered image inputs, reconstructing the scene and estimating camera poses at the same time. Though accurate, classical SfM methods [31, 34] only achieve sparse representation and are brittle to dynamic objects in the view. On the other hand, recent feed-forward 3D reconstruction models [38, 41, 44], trained on large-scale datasets, exhibit substantially improved robustness under sparse input conditions compared with classical SfM pipelines. Nevertheless, such models are typically memory-intensive and encounter difficulties when applied to long video sequences. Though some methods [5] utilize such models in a chunk-wise way to handle long videos, their results are not as good as global predictions [7].

Dense dynamic scene reconstruction If given camera poses, the simplest way is to use monocular depth prediction models [3, 16, 32] to predict per-frame depth. However, the monocular depth model alone can often generate flickering and inconsistent results along a video [18]. Previous works solve the problem either via consistent video depth optimization [18, 28] or directly train video depth prediction [4, 15, 19]. Recently, many works reconstructing dynamic scenes rely on "3D reconstruction foundation" models [11, 22, 27, 40, 42, 47] to predict

3D scenes and camera poses at the same time. For example, *Monst3R* [47] fine-tunes the Dust3R model on dynamic videos and optimizes camera poses for long-sequence videos. *CUT3R* [40] uses a stateful recurrent model to process video inputs, achieving significant memory consumption reduction compared to global attention computation. However, existing feed-forward reconstruction models are either memory-intensive or produce low-quality reconstructions. In contrast, our method decouples camera pose estimation from depth optimization, enabling better reconstruction quality and scalability to long video sequences. The most relevant work to ours is [30], which reconstructs a mesh of dynamic objects in the view; however, it requires a fixed camera setup and prior extrinsic calibration. In contrast, our work can reconstruct scenes from freely moving cameras.

3 Methods

3.1 Overview

Problem Definition. As shown in Fig. 1, given a set of time-synchronized monocular video sequences

$$\mathcal{I} = \{(\mathbf{I}_i^t, \mathbf{K}_i) \mid i = 1, \ldots, N;\, t = 1, \ldots, T\},$$

where $\mathbf{I}_i^t \in \mathbb{R}^{H \times W \times 3}$ denotes the image captured by the ith camera at timestamp t, and $\mathbf{K}_i$ is the intrinsic parameters for the ith camera, the objective is to estimate the per-frame camera states

$$\mathcal{X} = \{(\mathbf{T}_i^t, \mathbf{D}_i^t) \mid i = 1, \ldots, N;\, t = 1, \ldots, T\},$$

where $\mathbf{T}_i^t \in SE(3)$ represents the camera pose and $\mathbf{D}_i^t \in \mathbb{R}^{H \times W}$ denotes the corresponding depth map.

In the following section, we first introduce how we track multiple cameras accurately and robustly. Then we demonstrate how to refine the scene consistently given the estimated camera poses. Figure 2 shows an overview of our approach.

3.2 Spatio-temporal multi-camera tracking

Preliminary. We utilize the learned correspondence estimation model from MegaSAM [24]. Given two frames $\mathbf{I}_i$ and $\mathbf{I}_j$, MegaSAM learns to predict the dense optical flow $\mathbf{f}_{i \to j} \in \mathbb{R}^{H' \times W' \times 2}$ and weights $\mathbf{w}_{ij} \in \mathbb{R}^{H' \times W'}$ with lower resolution $(H' \times W')$ in an iterative way. Given a grid of pixel coordinates $\mathbf{u}_i \in \mathbb{R}^{H' \times W' \times 2}$ in frame $\mathbf{I}_i$, we can predict the flow-warped correspondence of $\mathbf{u}_i$ on the image $\mathbf{I}_j$ by $\mathbf{u}_{ij}^{\text{flow}} = \mathbf{u}_i + \mathbf{f}_{i \to j}$. With the current estimated state, we can also project $\mathbf{u}_i$ directly on the image $\mathbf{I}_j$ and achieve the reprojection $\mathbf{u}_{ij}^{\text{reproj}}$:

$$\mathbf{u}_{ij}^{\text{reproj}} = \mathbf{K}_j(\mathbf{T}_{ij} \circ \mathbf{K}_i^{-1}(\mathbf{u}_i, \mathbf{d}_i)), \tag{1}$$

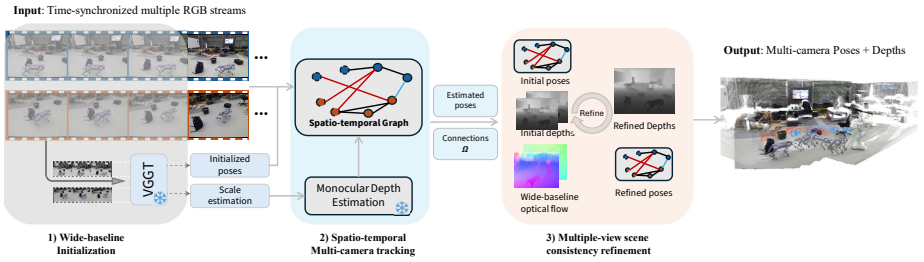


Fig. 2: Method Overview. Given multiple video inputs: Our method first uses a feed-forward model for initialization to achieve a global scale anchor and initialized poses. Then, we build a spatio-temporal connection graph during tracking to estimate camera poses and maintain a consistent scale. At last, we leverage the dense optical flow, estimated poses, and achieved connection graph to refine per-pixel depth to get a consistent scene and refined camera poses.

where $\mathbf{u}_{ij}^{\text{reproj}} \in \mathbb{R}^{H' \times W'}$ is the reprojected correspondence of $\mathbf{u}_i$ on frame $\mathbf{I}_j$; $\mathbf{T}_{ij} = \mathbf{T}_j^{-1} \circ \mathbf{T}_i$; $\mathbf{d}_i$ is the disparity map of frame $\mathbf{I}_i$. Given a group of connected frames $\mathbf{\Omega}$, parameters $\mathbf{T}$ and $\mathbf{d}$ are optimized by minimizing the weighted reprojection error:

$$\mathcal{L}(\mathbf{T}, \mathbf{d}) = \sum_{(i,j) \in \mathbf{\Omega}} ||\mathbf{u}_{ij}^{\text{reproj}} - \mathbf{u}_{ij}^{\text{flow}}||^2_{\Sigma_{ij}} \tag{2}$$

where $\Sigma_{ij} = \texttt{diag}(\mathbf{w}_{ij})$ are weights, where possible dynamic objects will be assigned low weights to reduce their effects.

Spatio-temporal connection graph Ω. To extend the monocular tracking framework to multiple cameras, one of our key contributions is to introduce a *spatio-temporal connection graph*. Our frame connection graph consists of three parts. **Temporal connection $\mathbf{\Omega}^{\text{temp}}$**: For each single camera, since its frames are sequential in time, adjacent frames tend to have enough overlap for reliable correspondence estimation. In this case, we follow common practice in one single-camera setting, maintaining a temporal window to hold the latest keyframes for each camera, which means $\mathbf{\Omega}^{\text{temp}} = \bigcup_i \mathbf{\Omega}_i^{\text{temp}}$ for each i-th camera. **Spatial connection $\mathbf{\Omega}^{\text{spat}}$**: In addition to the temporal intra-connection for each single camera, the spatial inter-connection is also necessary to maintain the scale consistency and improve accuracy. Given the N frames from all cameras at timestamp t, we decide the inter-camera connection by evaluating the overlap across different camera frames: given two RGB frames $\mathbf{I}_i^t$ and $\mathbf{I}_j^t$, we project the grid of coordinates $\mathbf{u}_i$ to the frame $\mathbf{I}_j^t$ by Eq. (1) to achieve the corresponding pixels $\mathbf{u}_{ij}^t$ on the frame $\mathbf{I}_j$'s image plane. If most of the projected pixels (more than 75%) are within the boundary of the image, then we make a spatial connection. This is helpful because even though there are dynamic objects in the scene, at the same timestamp, from different views, the scene is still consistent in this instant. **Spatio-temporal connection $\mathbf{\Omega}^{\text{st}}$**: In addition to the intra-camera temporal connection and inter-camera spatial connection at timestamp t, we

also exploit the historical information across different cameras. At timestamp t_0, keyframe $\mathbf{I}_i^{t_0}$ will try to evaluate $\mathbf{u}_{ij}^{t_0}$ with all other cameras' inactive keyframes $\{\mathbf{I}_j^t | j = [1, N], j \neq i, t = [1, T'], T' < t_0\}$. If there is enough overlap, we also make a connection between two frames. To prevent memory explosion and make connections effectively while running, we implement a *connection balance* strategy: we set a maximum number of edges in the tracking active window and allocate inter-camera connections evenly if they exist. If newly detected connections plus the existing connections are more than the maximum number, we remove the oldest ones from the tracking active window. A graphic demonstration of the spatio-temporal graph can be found in Fig. 3.

In all, during multi-camera tracking, we create the spatio-temporal connection graph

$$\mathbf{\Omega} = \mathbf{\Omega}^{\text{temp}} \cup \mathbf{\Omega}^{\text{spat}} \cup \mathbf{\Omega}^{\text{st}}.$$

We prove the effectiveness of the spatio-temporal connection graph in Sec. 6.2.

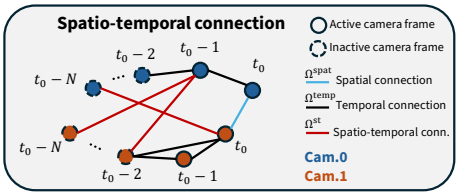


Fig. 3: Demonstration spatio-temporal graph. First, each single camera will estimate temporal connections with its own frames. Second, at the timestamp t_0, **Cam.1** will try to make a spatial connection with **Cam.0** if there is enough overlap. Additionally, the current active keyframe will try to make spatio-temporal connections with those inactive frames from other cameras if there is enough overlap. Ablation studies (Sec. 6.2) show spatio-temporal connections improve tracking accuracy.

Wide-baseline Initialization. In one single-camera setting, tracking initialization can typically rely on selecting frames with sufficient inter-frame overlap, which is naturally satisfied due to the temporal continuity of the image stream. In contrast, in multi-camera configurations, it is common that images captured from different viewpoints have limited or even no overlap at all, which makes conventional overlap-based initialization unreliable.

To address this issue, we adopt the robust feed-forward scene reconstruction model VGGT [38] to initialize our system. Specifically, during initialization, we select the first N_{init} frames from each camera stream and input them into VGGT to obtain initial camera pose estimates along with per-frame depth predictions $\mathbf{D}_{\text{VGGT}}$. These predictions provide a coarse but globally consistent geometric initialization across multiple cameras.

To further enhance tracking robustness and to obtain dense depth priors, similar to MegaSAM, we additionally incorporate monocular depth estimates during tracking. Concretely, we employ a monocular depth estimation model, *UniDepth* [32], to predict per-frame monocular depth maps $\mathbf{D}_{\text{mono}}$. Since monocular depth is only defined up to an affine transformation, we align the predicted depths to the initialization scale by estimating a global scale $s \in \mathbb{R}$ and offset

$o \in \mathbb{R}$ through optimizing

$$s, o = \min \sum_{k} ||s\mathbf{D}^{k}_{\text{mono}} + o - \mathbf{D}^{k}_{\text{VGGT}}||^2, \tag{3}$$

where k indexes the selected initialization frames and $\mathbf{D}^{k}_{\text{VGGT}}$ denotes the corresponding depth predictions from VGGT. The estimated scale and offset are then applied consistently to all monocular depth predictions during tracking.

After obtaining the initial camera poses and depth predictions from the feed-forward model VGGT, we run bundle adjustment to further refine the disparity maps and poses. Through the wide-baseline initialization, the system is able to establish a robust and metrically consistent initialization across multiple cameras from the outset, which significantly stabilizes subsequent tracking and optimization.

Tracking system. After initialization, we incrementally construct the spatio-temporal connection graph $\mathbf{\Omega}$ and jointly optimize camera poses and per-frame depths. As discussed in the Wide-baseline Initialization paragraph, to enhance robustness and enforce scene-scale consistency, we incorporate a prior depth regularization term into the bundle adjustment during tracking.

As described in the Spatio-temporal connection graph, upon the arrival of new frames $\{\mathbf{I}_i\}_i^N$, each frame is associated with previously processed frames to estimate new spatio-temporal connections, which are then added to $\mathbf{\Omega}$. The camera pose of each new frame is initialized under a constant-velocity motion assumption: $\mathbf{T}_i^t \leftarrow (\mathbf{T}_i^{t-1} \cdot (\mathbf{T}_i^{t-2})^{-1})$. To further improve tracking robustness, we integrate aligned monocular depth predictions as priors in the bundle adjustment. The resulting optimization problem is formulated as

$$\mathcal{L}(\mathbf{T}, \mathbf{d}) = \sum_{(i,j)\in\mathbf{\Omega}} ||\mathbf{u}_{ij}^{\text{reproj}} - \mathbf{u}_{ij}^{\text{flow}}||^2_{\Sigma_{ij}} + \lambda||\mathbf{d}_i - \mathbf{D}_i^{\text{s}}||^2 \tag{4}$$

The reference depth $\mathbf{D}_i^{\text{s}}$ is obtained by aligning the monocular depth prediction to the initialization scale via $\mathbf{D}_i^{\text{s}} = s\mathbf{D}_{\text{mono}}^{i} + o$, where the global scale s and offset o are estimated as described in Eq. (3). For online optimization, we maintain an active sliding window consisting of the most recent frames. Within this window, the poses of the earliest frames are fixed to remove the gauge freedom.

3.3 Multiple-view scene consistency refinement

The tracking stage (Sec. 3.2) produces initial camera pose and depth estimates:

$$\mathcal{X}_{\text{init}} = \{(\mathbf{T}_i^t, \mathbf{D}_i^t) \mid i = 1, \ldots, N;\ t = 1, \ldots, T\},$$

where $\mathbf{D}_i^t = s\mathbf{D}_{\text{mono}}^{i,t} + o$ represents the affine-aligned monocular depth from Eq. (3).

While this global affine alignment ensures metric scale consistency at initialization, it is insufficient for achieving dense multi-view geometric consistency throughout the full sequence. Specifically, the simple affine transformation cannot

account for: (1) per-frame scale drift in monocular depth predictions across the video sequence, and (2) per-pixel depth inaccuracies from the monocular depth model prediction. Therefore, we perform a multi-view depth refinement stage that optimizes per-frame scale and per-pixel depth corrections across all cameras simultaneously.

Depth refinement from monocular video has been extensively studied in prior work [18, 24, 28]. In this section, we extend these techniques to the multi-camera setting, exploiting both temporal consistency within each camera and spatial consistency across different cameras.

Dense correspondence estimation. To enable robust multi-view optimization, we compute dense optical flow correspondences that go beyond the low-resolution flow used during tracking. We construct an augmented connection graph $\mathbf{\Omega}_{\text{refine}}$ that includes: (1) the spatio-temporal graph $\mathbf{\Omega}$ from tracking (Sec. 3.2), and (2) additional dense temporal connections within each camera sequence. Specifically, motivated by prior monocular video depth optimization [18, 24, 28], for each frame i in a camera's sequence, we connect it to frames at offsets $\{+2, +4, +8\}$ when they exist, enabling longer-range temporal consistency constraints.

For all frame pairs in $\mathbf{\Omega}_{\text{refine}}$, we estimate dense optical flow using the wide-baseline flow estimation model UFM [48], which provides higher quality correspondences than the tracking-stage flow, particularly for large inter-frame motion and inter-camera baselines.

Optimization objectives. Let (i, j) denote a connected frame pair in $\mathbf{\Omega}_{\text{refine}}$. For brevity, we omit the timestamp superscript t in the following derivations.

Our optimization minimizes the weighted reprojection error:

$$\mathcal{L}_{\text{reproj}} = w_f \mathcal{L}_{\text{flow}} + w_d \mathcal{L}_{\text{disp}} \tag{5}$$

The flow reprojection term $\mathcal{L}_{\text{flow}}$ penalizes misalignment between optical flow correspondences and geometric reprojections. Specifically,

$$\mathcal{L}_{\text{flow}} = \frac{\sum_{(i,j)} \left(\|\mathbf{u}_{ij}^{\text{reproj}} - \mathbf{u}_{ij}^{\text{flow}}\|_1 \cdot \mathbf{c}_i + \lambda \log \frac{1}{\mathbf{c}_i} \right) \cdot m_{ij}}{\sum_{(i,j)} m_{ij}}, \tag{6}$$

where $\mathbf{u}_{ij}^{\text{flow}} = \mathbf{u}_i + \mathbf{f}_{i \to j}$ is the optical flow correspondence, $\mathbf{c}_i \in (0, 1]$ is an optimizable per-flow confidence map, and m_{ij} is the valid flow mask. The $\log(1/\mathbf{c}_i)$ term prevents the confidence from collapsing to zero.

The geometric reprojection $\mathbf{u}_{ij}^{\text{reproj}}$ is computed as:

$$\mathbf{u}_{ij}^{\text{reproj}} = \mathbf{K}_j (\mathbf{T}_{ij} \circ \mathbf{K}_i^{-1} (\mathbf{u}_i, s_i \mathbf{D}_i + \beta_i)), \tag{7}$$

where $s_i \in \mathbb{R}$ and $\beta_i \in \mathbb{R}$ are per-frame scale and shift parameters to be optimized. Note that these per-frame parameters are independent of the global initialization scale (s, o) from Eq. (3). The disparity consistency term $\mathcal{L}_{\text{disp}}$ penalizes inconsistencies between forward-projected and estimated depths, helping to reduce depth flickering [28]:

$$\mathcal{L}_{\text{disp}} = \frac{\sum_{(i,j)} \left(\left| \frac{\max(\mathbf{D}_j^{\text{flow}}, \mathbf{D}_j)}{\min(\mathbf{D}_j^{\text{flow}}, \mathbf{D}_j)} - 1 \right| \cdot \mathbf{c}_i + \lambda \log \frac{1}{\mathbf{c}_i} \right) \cdot m_{ij}}{\sum_{(i,j)} m_{ij}} \tag{8}$$

where $\mathbf{D}_j^{\text{flow}}$ is the depth warped from frame i to frame j via optical flow, while $\mathbf{D}_j$ is the estimated depth at frame j.

To improve the optimization stability, we refine the depth and camera poses in two phases.

Phase 1: Per-frame scale alignment. In the first phase, we fix all camera poses $\{\mathbf{T}_i^t\}$ to their initial estimates from tracking and optimize the per-frame affine parameters $\{s_i, \beta_i\}$ along with the flow confidence maps $\{\mathbf{c}_i\}$ by minimizing $\mathcal{L}_{\text{reproj}}$ (Eq. (5)). This establishes a consistent scale across all frames while down-weighting unreliable flow correspondences through learned confidence.

Phase 2: Iterative pose and depth refinement. Phase 1 establishes frame-level scale consistency but does not correct per-pixel depth errors in the monocular predictions. In Phase 2, we fix per-frame affine parameters (s_i, β_i), and directly optimize the per-pixel depth values $\mathbf{D}_i$ and refine camera poses by minimizing the same reprojection loss $\mathcal{L}_{\text{reproj}}$. Since jointly optimizing camera poses and depths suffers from instability [24], we propose to optimize poses and depths in an alternating iterative manner. For camera pose optimization, we augment the reprojection loss with pose regularization terms to maintain temporal smoothness:

Let the camera pose perturbation be $\boldsymbol{\delta}_i = [\omega_x, \omega_y, \omega_z, t_x, t_y, t_z]^\top \in \mathbb{R}^6$, where $[\omega_x, \omega_y, \omega_z]^\top$ is the rotation in axis-angle form and $[t_x, t_y, t_z]^\top$ is the translation.

The pose update $\Delta_i = \exp(\boldsymbol{\delta}_i) \in SE(3)$ yields the updated pose:

$$\mathbf{T}_i^{\text{new}} = \mathbf{T}_i^{\text{orig}} \cdot \exp\left(\begin{bmatrix} [\boldsymbol{\omega}_i]_\times & \mathbf{t}_i \\ 0 & 0 \end{bmatrix}\right) \tag{9}$$

The pose regularization loss consists of two components: $\mathcal{L}_{\text{pose}} = \mathcal{L}_{\text{prior}} + \mathcal{L}_{\text{smooth}}$, where $\mathcal{L}_{\text{prior}} = w_{\text{prior}} \sum_i ||\boldsymbol{\delta}_i||^2$ penalizes large deviations from the initial pose estimates. The smoothness term $\mathcal{L}_{\text{smooth}} = \mathcal{L}_{\text{smooth}}^{\text{rot}} + \mathcal{L}_{\text{smooth}}^{\text{trans}}$ enforces temporal consistency along each camera trajectory, where $\mathbf{T}_i^t = \begin{bmatrix} R_t^{(i)} & \mathbf{t}_t^i \\ 0 & 1 \end{bmatrix}$ with $R_t^{(i)} \in SO(3)$ and $\mathbf{t}_t^i \in \mathbb{R}^3$ being the rotation and translation components. The smoothness terms are defined as:

$$\mathcal{L}_{\text{smooth}}^{\text{rot}}(\boldsymbol{\delta}) = w_{\text{tem-rot}} \sum_{i=1}^{N_{\text{cam}}} \sum_{t=1}^{N_i - 1} \left\| R_t^{(i)T} R_{t+1}^{(i)} - I \right\|_F^2,$$

and

$$\mathcal{L}_{\text{smooth}}^{\text{trans}} = w_{\text{tem-trans}} \sum_{i=1}^{N_{\text{cam}}} \sum_{t=1}^{N_i - 1} \left\| \mathbf{t}_t^i - \mathbf{t}_{t+1}^i \right\|^2.$$

The overall pose optimization loss is $\mathcal{L} = \mathcal{L}_{\text{reproj}} + \mathcal{L}_{\text{pose}}$.

4 Real-World Multi-Camera Dataset

To assess our performance in real-world settings, we introduce the *MultiCam-Robolab* [4]. This dataset comprises 24 RGB-D sequences acquired in a laboratory

[4] https://github.com/ljjTYJR/multiple-view-dynamic-reconstruction

environment using two or three Microsoft Azure Kinect cameras under diverse motion patterns. Our method only requires RGB images. The depth information from the RGB-D Azure Kinects is only used to quantify performance. Ground-truth camera poses are provided by a Qualisys motion-capture system. Temporal synchronization between the RGB cameras and the Qualisys system is achieved via a time server running on the Qualisys PC. Specifically, timestamps are recorded for image frames and motion-capture measurements, and pairs whose timestamp differences are smaller than the sampling interval are regarded as synchronized observations. Each video clip is captured at 30 FPS and consists of 150–300 frames. The 24 sequences are divided into 5 distinct scenarios. The first 19 sequences are collected with 2 cameras: 1) $\text{Robodog}_{\text{overlap}}$ (4 sequences), where a robot dog is walking in the scene and two cameras have enough view overlap; 2) $\text{Robodog}_{\text{non-overlap}}$ (4 sequences), similar to the first case, but two cameras have little or no overlap; 3) RoboArm (4 sequences), where a 6-DOF manipulator performs digging motions, and 4) DynamicHuman (7 sequences), where a human is walking or working in the scene. The last 5 sequences are collected with 3 cameras: 5) Three-camera (5 sequences), where a human operator is operating, or a robot dog is moving in the scene.

5 Experiments setting

5.1 Baseline methods

We select *COLMAP* [34] and *GLOMAP* [31] as the baseline to represent the classical optimization-based structure-from-motion method. Since COLMAP and GLOMAP only generate sparse point models, we limit our comparison to camera pose evaluation when using them. In experiments, we set the known camera intrinsics for COLMAP and GLOMAP. In addition to the default settings, we also test COLMAP/GLOMAP with two extra settings for fair comparison: "with dynamic masks", which prunes dynamic objects from the views, and "with NetVLAD retrieval", which uses NetVLAD for image retrieval to establish more image connections.

We also evaluate against recent state-of-the-art feed-forward models: *Fast3R* [44], *VGGT* [38], and a memory-efficient version of VGGT *FastVGGT* [35]. These methods demonstrate strong performance in scenarios where traditional methods often fail. Finally, we include *CUT3R* [40] as a baseline due to its specific design for video processing, while it also supports unstructured image inputs. When running CUT3R, we sequentially feed frames from each camera video (i.e., camera-1 followed by camera-2) to fully leverage its temporal processing capabilities.

Fast3R and FastVGGT are evaluated on a single NVIDIA A100 GPU with 40 GiB of memory due to their higher memory requirements, whereas all other methods are tested on a single NVIDIA RTX 4090 GPU. Since VGGT cannot process all frames on a single A100 GPU due to memory limitations, we subsample the input sequence with an interval of 8 during evaluation.

5.2 Metrics

Camera Pose Evaluation. First, camera poses are aligned using the initial ground truth pose to maintain a common reference coordinate system. We then assess the cameras' trajectory accuracy with standard error metrics: Absolute Translation Error (ATE), Relative Translation Error (RTE), and Relative Rotation Error (RRE). Note that we assess the multiple camera trajectories as a single trajectory. The translation error is measured in meters, while the rotation error is measured in degrees.

Depth Quality Assessment. We evaluate depth quality using Absolute Relative Depth (Abs.Rel) and Delta accuracy [24,47]($\delta < 1.25$, i.e., the percentage of predicted depths within a 1.25-factor of true depth). Per-frame depth evaluation is conducted by scaling the predicted depth and offset to the ground truth.

Scene Consistency. Since depth metrics alone do not capture scene-level consistency, we evaluate scene consistency using point-to-point distance as the relevant metric. First, we align the predicted trajectory with the ground truth to compute a global scale factor. The predicted depth maps are subsequently scaled using this factor and re-projected into 3D coordinates. Consistency is then measured by computing the Euclidean distance between corresponding 3D points in the predicted and ground-truth reconstructions. The median Euclidean distance (denoted as M_d) is reported to minimize the impact of outliers.

5.3 Datasets

In addition to our self-collected MultiCamRobolab dataset, we also evaluate our method against the MultiCamVideo-Dataset [2]. MultiCamVideo-Dataset is a multi-camera synchronized video dataset rendered in simulation using Unreal Engine, which includes synchronized multi-camera videos and their corresponding camera trajectories. Each video clip has 81 frames with 10 different camera perspectives. In our evaluation, we randomly extract 50 clips and 3 random cameras for each clip in the MultiCamVideo-Dataset. All images are cropped to 512×384, and trajectories are normalized in $[-1, 1]$.

6 Results

6.1 Quantitative & Qualitative Comparisons

First, we report the **camera pose evaluation** on the MultiCamVideo dataset and MultiCamRobolab in Tabs. 1 to 3. Our method achieves the best results in the MultiCamVideo datasets, as shown in Tab. 1. The right side in Tab. 1 shows the visualization of one reconstruction process of our method. Our method achieves the overall better results in MultiCamRobolab datasets (shown in Tabs. 2 and 3). Specifically, the classical SfM method COLMAP [34] tends to produce noisy pose estimates in the presence of dynamic objects. While GLOMAP [31] yields more accurate estimates, it fails to reconstruct two of the four scenes. Interestingly,

Table 1: Quantitative comparisons of camera trajectories on the **MultiCamVideo** [2] dataset. Average results of all video sequences are reported. †: To avoid memory overflow, we sample images with interval=8 for VGGT [38]; by default, other methods use all frames. The right side shows one reconstruction process of our method.

Method	Interval	ATE↓	RTE↓	RRE↓
COLMAP [34]		0.073	0.004	0.110
VGGT [38]		*OOM*	*OOM*	*OOM*
VGGT† [38]	8	0.027	0.016	0.177
Fast3R [44]		0.144	0.056	1.948
FastVGGT [35]		0.023	0.008	0.080
CUT3R [40]		0.175	0.011	0.164
Ours		0.005	0.001	0.011

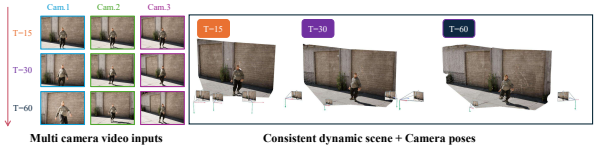


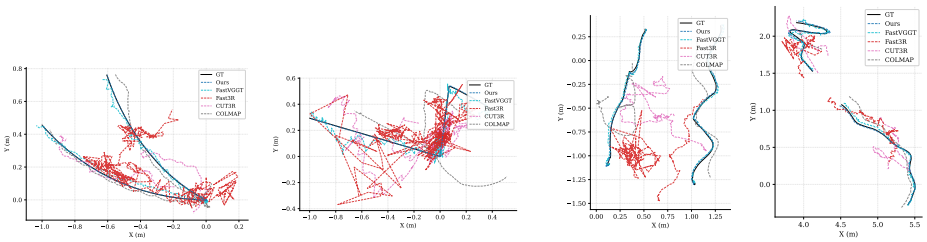
Table 2: Quantitative comparisons of camera trajectories on the **MultiCamRobolab** dataset. Average results of all video sequences are reported. †: To avoid memory overflow, we sample images with interval=8 for VGGT [38]; '×' means fail to reconstruct scenes: not all images are successfully registered. GPU memory reports the peak inference memory consumption.

	Method	Interval	RoboDog$_{\text{overlap}}$			RoboDog$_{\text{non-overlap}}$			RoboArm			DynamicHuman			GPU Mem. (GB)
			ATE↓	RTE↓	RRE↓	ATE↓	RTE↓	RRE↓	ATE↓	RTE↓	RRE↓	ATE↓	RTE↓	RRE↓	
	Classical SfM														
a)	COLMAP [34]		0.134	0.006	0.179	×	×	×	0.008	0.002	0.072	0.133	0.011	0.276	\
b)	GLOMAP [31]		×	×	×	×	×	×	0.006	0.001	0.053	0.020	0.009	0.259	\
	with dynamic masks														
c)	COLMAP [34]		0.045	0.005	0.174	×	×	×	0.008	0.002	0.068	×	×	×	\
d)	GLOMAP [31]		×	×	×	×	×	×	0.005	0.001	0.054	0.028	0.008	0.250	\
	with NetVLAD retrieval														
e)	COLMAP [34]		0.112	0.005	0.176	×	×	×	0.008	0.002	0.071	0.082	0.010	0.270	\
f)	GLOMAP [31]		×	×	×	×	×	×	0.005	0.001	0.054	0.019	0.009	0.244	\
	Feed-Forward														
g)	VGGT [38]		*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*
h)	VGGT† [38]	8	0.024	0.012	0.446	0.019	0.014	0.423	0.006	0.004	0.187	0.032	0.057	1.713	20.45
i)	Fast3R [44]		0.148	0.081	1.510	0.701	0.207	4.723	0.063	0.066	1.118	0.180	0.100	1.258	39.20
j)	FastVGGT [35]		0.021	0.009	0.234	0.020	0.013	0.266	0.006	0.004	0.107	0.020	0.013	0.286	22.08
k)	CUT3R [40]		0.277	0.016	0.299	0.377	0.021	0.326	0.055	0.003	0.112	0.196	0.016	0.330	22.43
l)	Ours		0.011	0.003	0.157	0.020	0.003	0.163	0.005	0.001	0.059	0.013	0.009	0.257	20.04

we find that pruning dynamic objects does not always improve reconstruction. For example, on the DynamicHuman scene, comparing (a),(b) with (c),(d) shows that applying dynamic masks instead degrades performance. We attribute this to three factors: (i) dynamic objects are momentarily *statically consistent* across synchronized cameras, providing valid inter-camera correspondences; (ii) at 30 FPS, object motion is small, so features can remain locally consistent over short time windows; and (iii) in textureless environments, dynamic objects supply critical keypoints. In summary, dynamic objects can themselves contribute to image matching, and removing them simply reduces the number of available correspondences. Unlike these baselines, our method relies on dense neural optical flow; through the weighting mechanism in our optimization, it implicitly accounts for correspondence reliability by down-weighting outliers that violate multi-view consistency. COLMAP and GLOMAP can also fail in the non-overlap scenes (see the RoboDog$_{\text{non-overlap}}$ in Tab. 2). VGGT cannot process all frames, even on a single A100-40GiB machine. For other feed-forward models, Fast3R also can not handle dynamic objects well; though CUT3R is designed for processing videos, it can not achieve satisfactory results (see Fig. 4, for each separate video clip,

Table 3: Quantitative comparisons of camera trajectories on the **MultiCamRobolab-3-cameras dataset**.

Method	ATE↓	RTE↓	RRE↓
COLMAP [34]	0.343	0.013	0.376
Fast3R [44]	0.376	0.154	1.825
FastVGGT [35]	0.032	0.017	0.363
CUT3R [40]	0.474	0.0351	0.467
Ours	0.020	0.011	0.326



(a) MultiCamVideo **(b)** MultiCamRobolab

Fig. 4: Visualization (projected to X-Y plane) of camera trajectories estimated by different methods in the two datasets. Multiple camera trajectories are treated as one and aligned with GT trajectories by SIM(3) alignment.

CUT3R can not generate smooth results). In comparison, FastVGGT's results are the best in all feed-forward models, which reveals the advantage of processing all frames together instead of test-time optimization used in CUT3R. In addition, the results indicate that FastVGGT is robust to dynamic objects in the scene. This observation motivates further investigation into the conditions under which such reconstruction models succeed or fail in dynamic environments. Our method achieves the best overall performance across all scenes (while consuming the least GPU memory), except for $\text{RoboDog}_{\text{non-overlap}}$. In this case, the cameras have few overlapping fields of view, causing the spatio-temporal connections to degenerate into purely temporal connections within each individual camera stream (more discussions can be found in the ablation study Sec. 6.2). The visualization of estimated trajectories compared to ground-truth can be found in Fig. 4.

Second, the **depth and scene consistency evaluation** is shown in Tab. 4. Since the MultiCamVideo does not provide ground-truth depth, we report quantitative evaluation results only on the MultiCamRobolab dataset in Tab. 4. Our method combines the prior depth and spatio-temporal connection refinement, achieving the most consistent depth evaluation. Notably, the scene consistency evaluation in fact combines both pose evaluation and depth evaluation. Though methods such as CUT3R achieve good results on single-frame video depth evaluation, they can not generate consistent scene results. The qualitative reconstruction results can be seen in Fig. 5.

Table 4: Quantitative comparisons of frame depth and scene consistency on the **MultiCamRobolab** dataset. Average results of all video sequences are reported. †:To avoid memory overflow, we sample images with interval=8 for VGGT [38]; by default, other methods use all frames.

Method	Interval	RoboDog$_{\text{overlap}}$			RoboDog$_{\text{non-overlap}}$			RoboArm			DynamicHuman		
		Abs.Rel↓	$\delta_{1.25}$ ↑	Md↓	Abs.Rel↓	$\delta_{1.25}$ ↑	Md↓	Abs.Rel↓	$\delta_{1.25}$ ↑	Md↓	Abs.Rel↓	$\delta_{1.25}$ ↑	Md↓
VGGT [38]	8	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*
VGGT† [38]		0.068	0.939	0.376	0.060	0.947	0.327	0.171	0.772	0.547	0.194	0.900	0.461
Fast3R [44]		0.144	0.788	0.580	0.157	0.749	1.501	0.191	0.714	0.763	0.214	0.735	1.799
FastVGGT [35]		0.026	0.978	0.128	0.018	0.988	0.095	0.060	0.906	0.292	0.030	0.987	0.185
CUT3R [40]		0.048	0.968	0.715	0.040	0.987	0.581	0.158	0.745	0.708	0.060	0.963	0.503
Ours		0.011	0.989	0.091	0.018	0.991	0.092	0.059	0.947	0.289	0.030	0.971	0.112

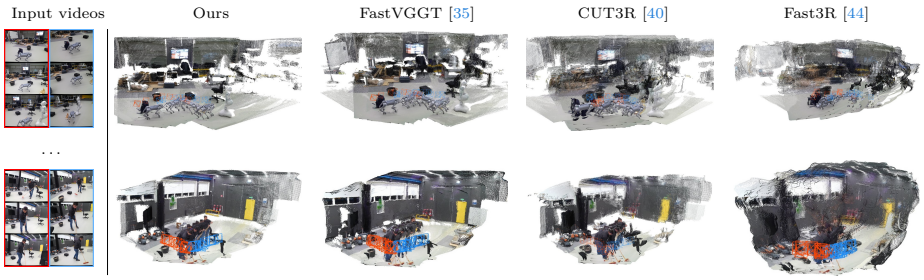


Fig. 5: Qualitative reconstruction results on MultiCamRobolab datasets.

6.2 Ablation Studies

In this section, we perform ablation studies to evaluate the contributions of the key components of our method. In particular, we investigate the effectiveness of (1) the proposed wide-baseline initialization, (2) the spatio-temporal graph used for camera pose tracking, and (3) the refinement stage for improving depth estimation and scene consistency. In Tab. 5, we report the trajectory differences obtained by removing the wide-baseline initialization (W.B. Init.), and the spatio-temporal connection (ST-Graph). The full method is treated as the baseline, and all other results are presented as deviations relative to the full method. Red indicates performance degradation, whereas blue denotes performance improvement. By the results, we can see that the proposed wide-baseline initialization is important for the tracking. That is because, unlike traditional single-camera tracking problems, there is often not enough overlap across different cameras for initialization; it is important to use some reconstruction model to get a prior estimation. But our method is also robust to noisy initialization; we conduct an extra ablation study by adding noise to the initialized poses. In Tab. 6, we add rotational noise of 3° and different levels of translation noise; the results show that our method can recover from the noisy initialized poses under mild conditions. Note that in Tab. 5, "`w/o` ST-Graph" means our method degenerates to running separate video reconstruction and then aligning them via feed-forward predicted poses. The second row in Tab. 5 shows that the spatio-temporal graph can help improve the tracking results because it introduces more constraints during the tracking.

Table 5: Ablation studies on camera pose tracking. The "`Full method`" is used as the baseline, and all other results are reported as relative changes with respect to this baseline. Red entries indicate performance degradation, whereas blue entries indicate performance improvement.

Method	RoboDog$_{\text{overlap}}$			RoboDog$_{\text{non-overlap}}$			RoboArm			DynamicHuman		
	ATE↓	RTE↓	RRE↓	ATE↓	RTE↓	RRE↓	ATE↓	RTE↓	RRE↓	ATE↓	RTE↓	RRE↓
`w/o` W.B. Init.	+0.284	+0.003	-0.002	+0.496	+0.019	+0.026	+0.084	+0.002	+0.001	+0.130	+0.001	+0.009
`w/o` ST-Graph	+0.158	+0.003	-0.002	+0.006	+0.000	-0.002	+0.012	+0.000	-0.001	+0.141	+0.002	+0.007
`Full method`	0.011	0.003	0.157	0.020	0.003	0.163	0.005	0.001	0.059	0.013	0.009	0.257

Table 6: Ablation studies on noisy initializaztion.

Noise level	ATE↓
0	0.013
0.02 m	0.013
0.05 m	0.016
0.10 m	0.044

Note that for the scene RoboDog$_{\text{non-overlap}}$, the spatio-temporal graph helps least, because there is little overlap across cameras. In Tab. 7, we show that optimizing the per-depth-frame scale and offset is not enough; two phases of refinement can help improve the depth accuracy and scene consistency. (More ablation studies on refinement and spatio-temporal connection edge counts can be found in the supplementary materials.)

Table 7: Ablation studies on video depth evaluation and scene consistency. We evaluate the effect of the refinement proposed in Sec. 3.3. The results show that the refinement stage can improve overall scene consistency.

Method	RoboDog$_{\text{overlap}}$			RoboDog$_{\text{non-overlap}}$			RoboArm			DynamicHuman		
	Abs.Rel↓	$\delta_{1.25}$ ↑	Md↓	Abs.Rel↓	$\delta_{1.25}$ ↑	Md↓	Abs.Rel↓	$\delta_{1.25}$ ↑	Md↓	Abs.Rel↓	$\delta_{1.25}$ ↑	Md↓
× Phase1; × Phase2	+0.020	+0.001	+0.066	+0.014	+0.001	+0.060	+0.068	-0.123	+0.226	+0.028	+0.007	+0.063
✓ Phase1; × Phase2	+0.012	+0.001	+0.029	+0.005	+0.001	+0.012	+0.044	-0.083	+0.087	+0.003	+0.007	+0.012
✓ Phase1; ✓ Phase2	0.011	0.989	0.091	0.018	0.991	0.092	0.059	0.947	0.289	0.030	0.971	0.112

7 Conclusion

We introduce the first framework for dense dynamic scene reconstruction and camera pose estimation from multiple free-moving cameras. We use the feed-forward reconstruction model for robust initialization and introduce a spatio-temporal connection graph for multi-camera tracking in a consistent way. Based on the constructed connection graph, we introduce a framework for optimizing depths across multiple cameras. Our method achieves better results while consuming less GPU memory compared to the state-of-the-art feed-forward reconstruction models.

References

1. Azzarelli, A., Anantrasirichai, N., Bull, D.R.: Splatography: Sparse multi-view dynamic gaussian splatting for filmmaking challenges. arXiv preprint arXiv:2511.05152 (2025) 2
2. Bai, J., Xia, M., Fu, X., Wang, X., Mu, L., Cao, J., Liu, Z., Hu, H., Bai, X., Wan, P., et al.: Recammaster: Camera-controlled generative rendering from a single video. arXiv preprint arXiv:2503.11647 (2025) 11, 12
3. Bochkovskii, A., Delaunoy, A., Germain, H., Santos, M., Zhou, Y., Richter, S.R., Koltun, V.: Depth pro: Sharp monocular metric depth in less than a second. arXiv preprint arXiv:2410.02073 (2024) 3
4. Chen, S., Guo, H., Zhu, S., Zhang, F., Huang, Z., Feng, J., Kang, B.: Video depth anything: Consistent depth estimation for super-long videos. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 22831–22840 (2025) 3
5. Deng, K., Ti, Z., Xu, J., Yang, J., Xie, J.: Vggt-long: Chunk it, loop it, align it–pushing vggt's limits on kilometer-scale long rgb sequences. arXiv preprint arXiv:2507.16443 (2025) 3
6. Deng, T., Shen, G., Xun, C., Yuan, S., Jin, T., Shen, H., Wang, Y., Wang, J., Wang, H., Wang, D., et al.: Mne-slam: Multi-agent neural slam for mobile robots. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 1485–1494 (2025) 3
7. Ding, T., Xie, Y., Liang, Y., Chatterjee, M., Miraldo, P., Jiang, H.: Laser: Layer-wise scale alignment for training-free streaming 4d reconstruction. arXiv preprint arXiv:2512.13680 (2025) 3
8. Duhautbout, T., Moras, J., Marzat, J.: Distributed 3d tsdf manifold mapping for multi-robot systems. In: 2019 European Conference on Mobile Robots (ECMR). pp. 1–8. IEEE (2019) 3
9. Engel, J., Koltun, V., Cremers, D.: Direct sparse odometry. IEEE transactions on pattern analysis and machine intelligence **40**(3), 611–625 (2017) 3
10. Engel, J., Schöps, T., Cremers, D.: Lsd-slam: Large-scale direct monocular slam. In: European conference on computer vision. pp. 834–849. Springer (2014) 3
11. Feng, H., Zhang, J., Wang, Q., Ye, Y., Yu, P., Black, M.J., Darrell, T., Kanazawa, A.: St4rtrack: Simultaneous 4d reconstruction and tracking in the world. arXiv preprint arXiv:2504.13152 (2025) 3
12. Heng, L., Choi, B., Cui, Z., Geppert, M., Hu, S., Kuan, B., Liu, P., Nguyen, R., Yeo, Y.C., Geiger, A., et al.: Project autovision: Localization and 3d scene perception for an autonomous vehicle with a multi-camera system. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 4695–4702. IEEE (2019) 2, 3
13. Homeyer, C., Begiristain, L., Schnörr, C.: Droid-splat combining end-to-end slam with 3d gaussian splatting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2767–2777 (2025) 3
14. Hu, J., Mao, M., Bao, H., Zhang, G., Cui, Z.: Cp-slam: Collaborative neural point-based slam system. Advances in Neural Information Processing Systems **36**, 39429–39442 (2023) 2
15. Hu, W., Gao, X., Li, X., Zhao, S., Cun, X., Zhang, Y., Quan, L., Shan, Y.: Depthcrafter: Generating consistent long depth sequences for open-world videos. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 2005–2015 (2025) 3
16. Huang, J., Zhou, Q., Rabeti, H., Korovko, A., Ling, H., Ren, X., Shen, T., Gao, J., Slepichev, D., Lin, C.H., et al.: Vipe: Video pose engine for 3d geometric perception. arXiv preprint arXiv:2508.10934 (2025) 2, 3

17. Klein, G., Murray, D.: Parallel tracking and mapping for small ar workspaces. In: 2007 6th IEEE and ACM international symposium on mixed and augmented reality. pp. 225–234. IEEE (2007) 3
18. Kopf, J., Rong, X., Huang, J.B.: Robust consistent video depth estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1611–1621 (2021) 3, 8
19. Kuang, Z., Zhang, T., Zhang, K., Tan, H., Bi, S., Hu, Y., Xu, Z., Hasan, M., Wetzstein, G., Luan, F.: Buffer anytime: Zero-shot video depth and normal from image priors. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 17660–17670 (2025) 3
20. Kuo, J., Muglikar, M., Zhang, Z., Scaramuzza, D.: Redesigning slam for arbitrary multi-camera systems. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 2116–2122. IEEE (2020) 3
21. Lajoie, P.Y., Ramtoula, B., Chang, Y., Carlone, L., Beltrame, G.: Door-slam: Distributed, online, and outlier resilient slam for robotic teams. IEEE Robotics and Automation Letters **5**(2), 1656–1663 (2020) 2
22. Leroy, V., Cabon, Y., Revaud, J.: Grounding image matching in 3d with mast3r. In: European Conference on Computer Vision. pp. 71–91. Springer (2024) 3
23. Li, L., Gu, R., Wang, C., Xing, J., Yu, X., Zhang, X.P.: Multi-view 3d human pose estimation with weakly synchronized images. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 4833–4841 (2025) 2
24. Li, Z., Tucker, R., Cole, F., Wang, Q., Jin, L., Ye, V., Kanazawa, A., Holynski, A., Snavely, N.: Megasam: Accurate, fast and robust structure and motion from casual dynamic videos. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 10486–10496 (2025) 2, 4, 8, 9, 11
25. Lipson, L., Deng, J.: Multi-session slam with differentiable wide-baseline pose optimization. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 19626–19635 (2024). https://doi.org/10.1109/CVPR52733.2024.01856 3
26. Liu, P., Geppert, M., Heng, L., Sattler, T., Geiger, A., Pollefeys, M.: Towards robust visual odometry with a multi-camera system. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1154–1161. IEEE (2018) 3
27. Lu, J., Huang, T., Li, P., Dou, Z., Lin, C., Cui, Z., Dong, Z., Yeung, S.K., Wang, W., Liu, Y.: Align3r: Aligned monocular depth estimation for dynamic videos. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 22820–22830 (2025) 2, 3
28. Luo, X., Huang, J.B., Szeliski, R., Matzen, K., Kopf, J.: Consistent video depth estimation. ACM Transactions on Graphics (ToG) **39**(4), 71–1 (2020) 3, 8
29. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: A versatile and accurate monocular slam system. IEEE transactions on robotics **31**(5), 1147–1163 (2015) 3
30. Mustafa, A., Kim, H., Guillemaut, J.Y., Hilton, A.: Temporally coherent 4d reconstruction of complex dynamic scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4660–4669 (2016) 4
31. Pan, L., Baráth, D., Pollefeys, M., Schönberger, J.L.: Global structure-from-motion revisited. In: European Conference on Computer Vision. pp. 58–77. Springer (2024) 3, 10, 11, 12
32. Piccinelli, L., Yang, Y.H., Sakaridis, C., Segu, M., Li, S., Van Gool, L., Yu, F.: Unidepth: Universal monocular metric depth estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10106–10116 (2024) 3, 6

33. Schmied, A., Fischer, T., Danelljan, M., Pollefeys, M., Yu, F.: R3d3: Dense 3d reconstruction of dynamic scenes from multiple cameras. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3216–3226 (2023) 3
34. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4104–4113 (2016) 3, 10, 11, 12, 13
35. Shen, Y., Zhang, Z., Qu, Y., Zheng, X., Ji, J., Zhang, S., Cao, L.: Fastvggt: Training-free acceleration of visual geometry transformer. arXiv preprint arXiv:2509.02560 (2025) 10, 12, 13, 14
36. Tian, Y., Chang, Y., Arias, F.H., Nieto-Granda, C., How, J.P., Carlone, L.: Kimera-multi: Robust, distributed, dense metric-semantic slam for multi-robot systems. IEEE Transactions on Robotics **38**(4) (2022) 2
37. Tian, Y., Chang, Y., Herrera Arias, F., Nieto-Granda, C., How, J.P., Carlone, L.: Kimera-multi: Robust, distributed, dense metric-semantic slam for multi-robot systems. IEEE Transactions on Robotics **38**(4), 2022–2038 (2022). https://doi.org/10.1109/TRO.2021.3137751 3
38. Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., Novotny, D.: Vggt: Visual geometry grounded transformer. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 5294–5306 (2025) 3, 6, 10, 12, 14
39. Wang, Q., Ye, V., Gao, H., Zeng, W., Austin, J., Li, Z., Kanazawa, A.: Shape of motion: 4d reconstruction from a single video. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9660–9672 (2025) 2
40. Wang, Q., Zhang, Y., Holynski, A., Efros, A.A., Kanazawa, A.: Continuous 3d perception model with persistent state. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 10510–10522 (2025) 3, 4, 10, 12, 13, 14
41. Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: Dust3r: Geometric 3d vision made easy. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20697–20709 (2024) 3
42. Wang, Y., Zhou, J., Zhu, H., Chang, W., Zhou, Y., Li, Z., Chen, J., Pang, J., Shen, C., He, T.: pi-3: Permutation-equivariant visual geometry learning. arXiv preprint arXiv:2507.13347 (2025) 3
43. Wang, Z., Tan, J., Khurana, T., Peri, N., Ramanan, D.: Monofusion: Sparse-view 4d reconstruction via monocular fusion. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8252–8263 (2025) 2
44. Yang, J., Sax, A., Liang, K.J., Henaff, M., Tang, H., Cao, A., Chai, J., Meier, F., Feiszli, M.: Fast3r: Towards 3d reconstruction of 1000+ images in one forward pass. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 21924–21935 (2025) 3, 10, 12, 13, 14
45. Yugay, V., Gevers, T., Oswald, M.R.: Magic-slam: Multi-agent gaussian globally consistent slam. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 6741–6750 (2025) 2, 3
46. Yus, R., Mena, E., Ilarri, S., Illarramendi, A., Bernad, J.: Multicamba: a system for selecting camera views in live broadcasting of sport events using a dynamic 3d model. Multimedia Tools and Applications **74**(11), 4059–4090 (2015) 2
47. Zhang, J., Herrmann, C., Hur, J., Jampani, V., Darrell, T., Cole, F., Sun, D., Yang, M.H.: Monst3r: A simple approach for estimating geometry in the presence of motion. arXiv preprint arXiv:2410.03825 (2024) 2, 3, 4, 11
48. Zhang, Y., Keetha, N., Lyu, C., Jhamb, B., Chen, Y., Qiu, Y., Karhade, J., Jha, S., Hu, Y., Ramanan, D., et al.: Ufm: A simple path towards unified dense correspondence with flow. arXiv preprint arXiv:2506.09278 (2025) 8