St4RTrack: Simultaneous 4D Reconstruction and Tracking in the World

Haiwen Feng[1,2] * Junyi Zhang[1] * Qianqian Wang[1] Yufei Ye[3] Pengcheng Yu[2]
Michael J. Black[2] Trevor Darrell[1] Angjoo Kanazawa[1]

[1]UC Berkeley [2]Max Planck Institute for Intelligent Systems [3]Stanford University

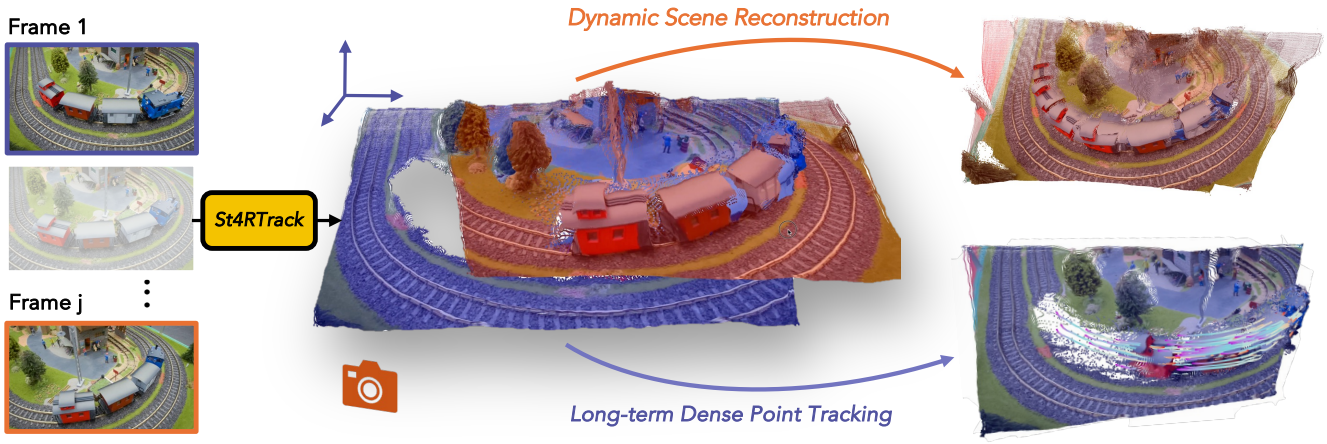

Figure 1. **St4RTrack:** Given an RGB video capturing dynamic scenes, St4RTrack simultaneously tracks the points from the initial frame (visualized in purple) and reconstructs the geometry of the subsequent frames (in orange) in a consistent world coordinate frame. St4RTrack is a *feed-forward* framework that takes a pair of images as input and outputs two pointmaps in the world frame, as the visualization shown in the middle. By iteratively processing the first frame paired with each subsequent frame, St4RTrack achieves simultaneous tracking (right bottom) and reconstruction (right top) for the entire video. Interactive results on our webpage: https://St4RTrack.github.io/.

Abstract

Dynamic 3D reconstruction and point tracking in videos are typically treated as separate tasks, despite their deep connection. We propose St4RTrack, a feed-forward framework that simultaneously reconstructs and tracks dynamic video content in a world coordinate frame from RGB inputs. This is achieved by predicting two appropriately defined pointmaps for a pair of frames captured at different moments. Specifically, we predict both pointmaps at the same moment, in the same world, *capturing both static and dynamic scene geometry while maintaining 3D correspondences. Chaining these predictions through the video sequence with respect to a reference frame naturally computes long-range correspondences, effectively combining 3D reconstruction with 3D tracking. Unlike prior methods that rely heavily on 4D ground truth supervision, we employ a novel adaptation scheme based on a reprojection loss. We establish a new extensive benchmark for world-frame reconstruction and tracking, demonstrating the effectiveness and efficiency of our unified, data-driven framework. Our code, model, and benchmark will be released.*

*Equal contribution, listed alphabetically.

1. Introduction

When asked about the three most important problems in computer vision, Takeo Kanade replied, "Correspondence, Correspondence, Correspondence!" This remark is especially pertinent to multi-view 3D reconstruction, where 3D geometry and 2D correspondence are two sides of the same coin; that is, a 3D point in the physical world naturally brings 2D correspondence across its projections in different views, and conversely, corresponded 2D points across views reconstruct the same 3D point after triangulation. This synergy between 3D geometry and 2D correspondence has long formed the foundation of multi-view geometry [17]. However, when the scene becomes dynamic, this synergy appears to vanish, as existing methods—particularly the recent data-driven ones—tend to treat dynamic reconstruction [22, 58, 67] and correspondence [24, 38, 51] as separate, disconnected tasks. We argue that this is a missed opportunity; the synergy between 3D reconstruction and 2D correspondence is not lost in dynamic scenes—it simply requires an additional element: understanding how the scene content evolves over time. This evolution is captured by 3D motion estimation (*i.e.*, dense 3D point tracking),

which, when computed across a sequence, effectively explains scene motion. Once tracking accounts for scene dynamics, the problem effectively reduces to the rigid case, where the natural interplay between reconstruction and correspondence can once again be leveraged.

We propose St4RTrack, a learning framework that unifies reconstruction and tracking directly from RGB video frames. St4RTrack simultaneously reconstructs and tracks dynamic video content in a single consistent world, achieving world-frame 3D tracking, as demonstrated in Fig. 1. Tracking in the world frame decouples the scene and the camera motion, essential for domains where both the camera and content are in motion. Our approach also reconstructs the 3D geometry of observed image content for both static and dynamic portions of the scene. St4RTrack directly predicts their reconstruction and tracking in the world without requiring an additional alignment stage.

Our key insight stems from the observation that a feedforward static 3D reconstruction method, DUSt3R [61], can be adapted to dynamic scenes simply by changing the pointmaps' annotation [67]. Building on this, we reexamine the pointmaps definition in the 4D scenario and opt to simply redefine its geometric interpretation for both reconstruction and tracking, as illustrated in Fig. 2. Concretely, we achieve it by predicting two pointmaps *at the same timestamp* and *in the same world* from a pair of image frames depicting two different timestamps. More specific, given images $(\mathbf{I}_i, \mathbf{I}_j)$, both pointmaps are predicted in the coordinate frame of $\mathbf{I}_i$, but at the time specified by $\mathbf{I}_j$. Our method is realized through a feed-forward network comprising of two branches: the *reconstruction branch* reconstructs the content of $\mathbf{I}_j$ in the $\mathbf{I}_i$ coordinate frame; and the *tracking branch*, which reconstructs the content of $\mathbf{I}_i$ in the $\mathbf{I}_i$ (its own) coordinate frame, *but* at the time indicated by $\mathbf{I}_j$. Essentially, the tracking branch predicts how the scene content in $\mathbf{I}_i$ evolves to match the moment captured in $\mathbf{I}_j$. This is enabled through a DUSt3R-like dual cross-attention mechanism, where the tracking branch relies on the reconstruction branch to decide how to move points. This minimal change proves sufficient for unifying both dynamic reconstruction and 3D point tracking in the world coordinate system.

Furthermore, unlike existing methods [58, 61, 67] that rely solely on 4D supervision, our approach unlocks 4D reconstruction training on in-the-wild videos via reprojection loss *without* 4D supervision. This is possible because St4RTrack simultaneously establishes camera parameters, 3D geometry and motion. Specifically, based on the outputs of the reconstruction branch, the camera parameters for $\mathbf{I}_j$ can be differentiably computed via PnP. Using these cameras, the pointmap of $\mathbf{I}_i$ is projected into the j-th frame, enabling training with a reprojection loss that leverages 2D correspondences and monocular depth predictions from off-the-shelf approaches [23, 59]. Consequently, the monocular supervisions facilitate effective *test-time adaptation* of St4RTrack to in-the-wild videos, which can differ substantially from the synthetic data used during pretraining.

While prior 3D point tracking benchmarks focus on camera coordinate frames [64], our approach enables world-frame 3D tracking. To evaluate this capability, we establish a novel benchmark, WorldTrack, for both tracking and reconstruction in the world coordinate system. We find that our unified method outperforms the strong baselines that combine several pieces on each individual task. Furthermore, we show that our feedforward results can be improved via test-time adaptation. We believe this is a step towards a unified task-agnostic 4D perception system that can be trained on a large-scale video. Our code, model, and benchmark will be released.

2. Related Works

Camera Estimation and Scene Reconstruction. Jointly estimating camera motion and scene geometry has been studied for decades, often in the context of Structure from Motion (SfM) [1, 47, 48, 56] or Simultaneous Localization and Mapping (SLAM) [7, 10, 37, 52, 60]. However, these methods are primarily designed for static scenes and typically do not model dynamic scene content. Recent advances in learning-based monocular and video depth estimation methods [2, 42, 44, 65] have opened new opportunities to reconstruct dynamic scenes. Notably, R-CVD [25], CasualSAM [68] and MegaSAM [33] jointly optimize camera parameters and per-frame dense depth maps leveraging monocular depth priors, producing consistent depth estimates for dynamic objects and accurate camera parameters even in challenging cases with minimal camera parallax. Another notable recent method, DUSt3R [61], introduces a two-pointmap representation that enables joint estimation of camera motion and scene geometry of a pair of images. While DUSt3R itself primarily focuses on reconstructing static scenes, follow-up effort such as MonST3R [67], demonstrate that this formulation can also effectively handle dynamic scenes with minimal modification on supervisions. Despite these advances, none of the aforementioned methods explicitly estimate 3D scene motion, meaning they do not track the movement of individual 3D points over time. In contrast, our method simultaneously performs joint reconstruction and tracking for dynamic scenes.

2D/3D Point Tracking. Tracking pixel motion over time is a fundamental problem in computer vision. Optical/Scene flow methods [3, 18, 19, 35, 50, 51, 53, 55] produce dense 2D/3D motion vectors but are inherently short-ranged, struggling with large displacements and occlusions. While long-range point tracking [45, 46] has been studied for decades, it has recently been revitalized via supervised learning [8, 9, 16, 23, 24], enabling more robust tracking

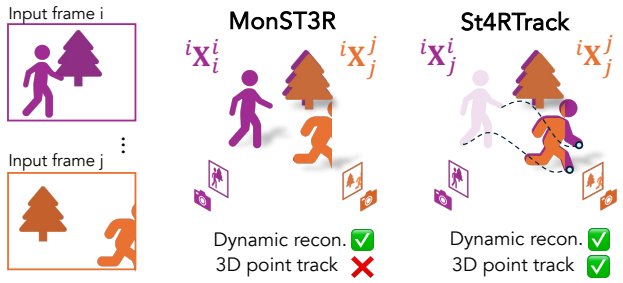

Figure 2. **Pointmap Comparison of MonST3R and St4RTrack.** Given two input frames, MonST3R handles *dynamic* scenes by reconstructing both pointmaps in their own timestamp. St4RTrack predicts where the points in the first frame move in the second frame, and reconstructs the geometry of the second frame. More details of the representation definition are introduced in Sec. 3.1.

over extended time periods and overcoming these limitations. However, these methods still produce only 2D pixel trajectories. More recently, several works [38, 64] achieve 3D tracking by lifting points into 3D space using monocular depth priors and performing tracking in 3D. While closely related to our approach, these methods still operate in the camera frame space, meaning they lack camera motion estimation and do not explicitly separate scene motion from camera motion. In contrast, our method jointly estimates disentangled camera and scene motion, enabling world-space tracking for a more complete understanding of 3D scene dynamics.

Joint Dynamic Reconstruction and Tracking. While traditional non-rigid SfMs have studied joint tracking and reconstruction [4, 6, 12, 39, 70] from 2D correspondences, jointly optimizing them from raw videos is highly challenging and typically requires multi-view synchronized videos as input [13, 30, 36, 41, 63]. With recent advances in neural rendering and data-driven geometric priors, reconstructing and rendering dynamic scenes from monocular videos became possible. However, as Gao et al. [14] point out, many methods [43, 66] focus on "teleporting" input data, which are effectively multi-view and not representative of real-world videos. In addition, since the main focus is view synthesis, motion estimation serves a secondary role in facilitating information fusion between neighboring frames [31, 32]. More recently, several works [27, 34, 57] focus on jointly recovering camera parameters, persistent scene geometry, and long-range 3D tracks from single, causally captured videos. However, these methods take off-the-shelf priors as given and design per-video optimization techniques that optimize a representation from scratch. Most recently, Stereo4D [20]—a concurrent effort to our work—proposed a pipeline for crafting a real-world 4D tracking dataset using internet stereo videos, enabling the supervised regression of 3D trajectories and geometries between frames. In contrast, we propose a feed-forward method that simultaneously performs reconstruction and tracking, while the same architecture also supports test-time adaptation on unlabeled videos to approach the high quality of optimization-based methods.

3. Simultaneous Reconstruction and Tracking

We present a framework that simultaneously reconstructs and tracks dynamic video content in 3D within a single consistent world coordinate frame. The core idea is simple yet powerful: reconstructing and tracking can both be achieved by predicting two appropriately defined pointmaps, where both pointmaps reconstruct the scene content observed in each image *at the same timestamp* in a *consistent coordinate system*. This enables simultaneous reconstruction of both dynamic and static contents, while tracking across a sequence of image frames in a video. Since all geometry, camera, and motion (*i.e.* 3D correspondence over time) can be derived from the representation, it can be adapted to videos without any explicit 4D supervision. Below, we discuss the main insight, how St4RTrack compares to prior works. Then, we discuss the details of the model and how it can be trained and adapted to videos.

3.1. Unified 4D Representation of St4RTrack

Given two images $\mathbf{I}_i, \mathbf{I}_j$ with dynamic content (see Figure 2), how can one devise a single feedforward approach that simultaneously performs reconstruction and tracking? We argue that the underlying representation must (1) capture camera motion to establish a world coordinate frame, (2) reconstruct the 3D geometry of all observed points, and (3) estimate 3D motion that maintain explicit correspondence over time. In this work, we show that just two properly defined pointmaps suffice to fulfill these requirements.

Time-Dependent Pointmap. A pointmap representation assumes that each pixel in an image $\mathbf{I}$ of shape $H \times W$ is associated with a corresponding 3D point, forming a pointmap $\mathbf{X} \in \mathbb{R}^{H \times W \times 3}$. For the case of static scenes, DUSt3R [61] only considers two factors of pointmaps—(1) the source frame of the content and (2) the camera coordinate system in which the points are expressed. However, this definition is insufficient for modeling the dynamic scenario of monocular video. To address this, we introduce a previously overlooked yet decisive factor: *time*.

Specifically, we define a time-dependent pointmap that encodes the 3D positions of the scene points *in a chosen (world) coordinate system at a specific timestamp*. For clarity, we denote this representation as ${}^{a}\mathbf{X}_{t}^{b}$, which denotes the 3D pointmap of physical content from frame b, at time t, expressed in the coordinate system established by frame a." For example, ${}^{i}\mathbf{X}_{j}^{i}$ represents the geometry originally seen in frame i in frame i's own coordinate system, but described at

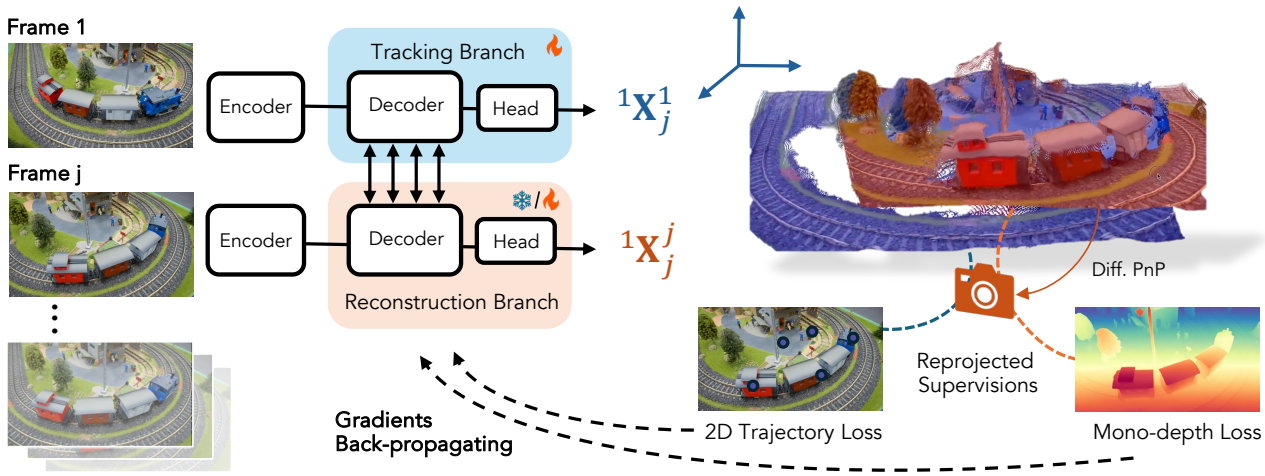

Figure 3. **Overview of St4RTrack.** Given frame 1 and frame j as input, the tracking branch outputs ${}^{1}\mathbf{X}^{1}_{j}$, the pointmap that corresponds to observed content of the first frame at timestep j in its own camera coordinate (*i.e.* world coordinate); the reconstruction branch outputs ${}^{1}\mathbf{X}^{j}_{j}$, the pointmap of the content in frame j at its own timestamp in the world coordinate. To adapt to new videos without any 4D labels, the camera is computed via differentiable PnP from the pointmap, enabling reprojected supervision signals (e.g., 2D trajectories and monocular depth). We finetune both branches during training (Sec. 3.2) with synthetic data, and when adapting to a new video (Sec. 3.3), only the tracking branch is fine-tuned using these reprojected supervision signals.

timestamp j. The time-dependency is achieved by construction without explicit timestamp conditioning, as described in the next section.

Unified 4D Modeling. St4RTrack learns a function f that maps two images $\mathbf{I}_i, \mathbf{I}_j$, captured at timestamp i and j, into two pointmaps:

$$f_\theta(\mathbf{I}_i, \mathbf{I}_j) = {}^{i}\mathbf{X}^{i}_{j}, {}^{i}\mathbf{X}^{j}_{j}. \tag{1}$$

The second image $\mathbf{I}_j$ is reconstructed as the pointmap ${}^{i}\mathbf{X}^{j}_{j}$ in the first image $\mathbf{I}_i$'s coordinate frame. Meanwhile, it predicts ${}^{i}\mathbf{X}^{i}_{j}$, representing the 3D motion of how the content from the first image $\mathbf{I}_i$ moves at timestamp j. Thus, both geometry and motion (tracking) are estimated from this unified prediction.

To handle a full video consisting of T frames, we perform tracking and reconstruction by always selecting the first frame as the anchor frame $\mathbf{I}_i$. Each subsequent frame $\mathbf{I}_j$ is then paired with this initial frame, ensuring that every new frame is consistently aligned to the coordinate system of the first frame. Specifically, $\{f(\mathbf{I}_1, \mathbf{I}_1), f(\mathbf{I}_1, \mathbf{I}_2), \ldots, f(\mathbf{I}_1, \mathbf{I}_T)\}$ are computed in the same reference, $\mathbf{I}_1$, which naturally serves as the world coordinate frame. Thus, world-frame 3D tracking is achieved by explicitly following how points observed in $\mathbf{I}_1$ are placed throughout the sequence, $\{{}^{1}\mathbf{X}^{1}_{1}, {}^{1}\mathbf{X}^{1}_{2}, \ldots, {}^{1}\mathbf{X}^{1}_{T}\}$, while the world frame dynamic reconstruction is obtained by the paired geometry estimation per-frame, $\{{}^{1}\mathbf{X}^{1}_{1}, {}^{1}\mathbf{X}^{2}_{2}, \ldots, {}^{1}\mathbf{X}^{T}_{T}\}$.

Relation to prior works. Our formulation of the 4D modeling generalizes prior works in a unified framework. DUSt3R reconstructs and establishes correspondences but is limited to rigid scenes, as correspondence and reconstruction are dual tasks for static scenes. With this perspective, one can see that if there is no dynamic component (i.e. frozen moment in time or rigid scenes), our formulation is equivalent to DUSt3R, where both images share the same timestamp $t = i$:

$$f_\theta(\mathbf{I}_i, \mathbf{I}_j) = {}^{i}\mathbf{X}^{i}_{i}, {}^{i}\mathbf{X}^{j}_{i}. \tag{2}$$

In a static world, 3D reconstruction from two pointmaps inherently yields the correspondences between them, allowing synergy to arise naturally. However, when objects or the scene are in motion, the dynamic component appears differently in different frames, it becomes crucial to account for 3D scene motion to preserve this synergy. St4RTrack addresses this challenge by predicting the 3D content from the first image at future timestamps.

In the same framework, we see that MonST3R, the dynamic follow-up of DUSt3R can be expressed as such:

$$f_\theta(\mathbf{I}_i, \mathbf{I}_j) = {}^{i}\mathbf{X}^{i}_{i}, {}^{i}\mathbf{X}^{j}_{j}, \tag{3}$$

where each image's 3D geometry is reconstructed in its timestamp, such that the dynamic contents separately align with their frame inputs. While it's sufficient for obtaining dynamic scene geometry, there is no temporal correspondence being established, as illustrated in Fig. 2. Furthermore, both DUSt3R and MonST3R compute the pairwise graphs and perform global alignment.

Since we always designate the first frame as the reference for tracking, the world coordinates are consistently established by the first frame. For simplicity, we omit the explicit notation of the world coordinate in subsequent equations and paragraphs, *i.e.*, $\mathbf{X}^i_j := {}^i\mathbf{X}^i_j$.

3.2. Joint Learning of Tracking and Reconstruction

In this section, we describe how our framework implements equation 1 within a pair-wise framework as DUSt3R. For each pair of frames, $\mathbf{I}_1$ and $\mathbf{I}_j$, we first encode them into token representations using a ViT encoder, then process these tokens through a siamese transformer decoder. The decoder sequentially applies self-attention (allowing tokens within each frame to interact), followed by cross-attention (enabling tokens from one frame to attend to tokens in the other), and finally passes the tokens through an MLP. This continuous information flow between the two branches is crucial for generating spatial-aligned 3D pointmaps in a shared coordinate system, as illustrated in Fig. 3.

Our siamese architecture processes two input views concurrently and generates two 3D pointmaps that are expressed in a common reference frame established by the first view. Although the two branches share the same architectural structure, they serve distinct purposes:

- **Tracking branch** predicts the pointmap $\mathbf{X}^1_j$, which represents the geometry of the first frame at timestamp j in the first frame's coordinates (*i.e.*, the world coordinates).
- **Reconstruction branch** predicts the pointmap $\mathbf{X}^j_j$, which represents the geometry of frame j at its own timestamp, also expressed in the first frame's camera coordinates.

Since this architecture is exactly the same as proposed by DUSt3R and subsequently adopted by MonST3R, with the only difference being the output paired pointmaps (Eq.1-3), our network can be initialized with pretrained 3D knowledge from either DUSt3R or MonST3R.

Pretraining with 4D Synthetic Data. Our proposed representation requires specialized supervision for the Tracking Branch—namely, ensuring that the pointmap from the first frame is correctly positioned in the world across all frames. Achieving this necessitates complete 4D information of the dynamic scene. Therefore, we leverage existing 4D synthetic datasets [22, 69] that provide both the 3D geometry and motion of the rendered content. Specifically, for each dataset, we use the scene mesh vertices (expressed in world coordinates) to provide sparse, masked supervision for the Tracking Branch pointmap representation, and employ per-frame depth maps and camera ground-truth to supervise the Reconstruction Branch. For this fully supervised training process, we use the objectives from DUSt3R. We initialize our dual-branch transformer with weights from MASt3R [29], a DUSt3R variant that has been adapted for 2D correspondence learning. Additional details regarding the 4D synthetic datasets are provided in Sec. 4.1.

3.3. Adapt to Any Video without 4D Label

While the synthetic datasets are small-scale and unrealistic, they are sufficient for our network to learn the newly proposed representations. However, fully supervised training on these datasets presents two key limitations: 1) The 4D synthetic data is limited in scale and does not encompass the full range of motion and geometry present in real-world dynamic scenes; 2) Our proposed pointmap representation requires the capability to freely move the pointmap within the world coordinates—a departure from conventional pixel-aligned geometry predictions, making small-scale training insufficient for achieving fine-grained predictions. These limitations motivate us to further leverage the 3D geometry and motion inherent in the St4RTrack framework to perform domain adaptation on any video without 4D labels. Specifically, we first show how we can derive camera parameters differentially, and with which we can design reprojected 2D trajectory loss and monocular depth loss to supervise the network.

Solving Camera Parameters. The intrinsic matrix $\mathbf{K}$ is first estimated from the tracking branch's first-frame pointmap prediction, following DUSt3R [61]. In this process, the principal point is assumed to be centered, and pixels are treated as square. The focal length is assumed static across frames and estimated using a fast iterative solver based on the Weiszfeld algorithm [62]. Next, the extrinsic parameters $\mathbf{P}^j = [\mathbf{R}^j|\mathbf{T}^j]$ for each frame j are derived using the "reconstruction" pointmap $\mathbf{X}^j_j$. Specifically, each pixel $\mathbf{x}^{j,n}$ in frame j is associated with a 3D coordinate $\mathbf{X}^{j,n}_j$ in the shared world coordinate system (established by the first camera), thus forming 2D-to-3D correspondences. We could then solve for $\mathbf{R}^j$ and $\mathbf{T}^j$ via a Perspective-n-Points (PnP) [28] solver with RANSAC [11] for outlier rejection:

$$\mathbf{R}^j, \mathbf{T}^j = \underset{\mathbf{R},\mathbf{T}}{\operatorname{argmin}} \sum_{n\in\mathcal{I}_j} \left\| \mathbf{x}^{j,n} - \pi\big(\mathbf{K}\,(\mathbf{R}\,\mathbf{X}^{j,n}_j + \mathbf{T})\big) \right\|^2, \tag{4}$$

where $\pi(\cdot)$ is the projection $(x, y, z) \to (x/z, y/z)$.

For differentiability, we adopt a derivative-based Gauss-Newton solver following [5], ensuring that gradients from the reprojection loss can adjust both the camera pose and the 3D pointmaps. Further details are provided in Appendix A.

Reprojection Loss. With the camera pose of frame j derived, the *tracking* pointmap $\mathbf{X}^1_j$ and *reconstruction* pointmap $\mathbf{X}^j_j$ can be transformed from the world coordinate system into the camera coordinate system of frame j. This transformation enables self-supervised training by enforcing two types of consistency: (1) 2D correspondence consistency, which aligns the projected 2D tracks (from $\mathbf{X}^1_j$) with the pseudo-ground truth tracking from CoTracker [23, 24], and (2) geometric consistency, which

aligns the scale-invariant depth (from $\mathbf{X}_j^j$) with the pseudo-ground truth monocular depth from MoGe [59].

More specifically, given the estimated camera pose $(\mathbf{R}^j, \mathbf{T}^j)$ from frame j and the tracking pointmap $\mathbf{X}_j^1$, we reproject these 3D points into the image plane of frame j:

$$\hat{\mathbf{x}}^{j,n} = \pi\big(\mathbf{K}\big(\mathbf{R}^j\,\mathbf{X}_j^{1,n} + \mathbf{T}^j\big)\big). \tag{5}$$

These reprojected points serve as the predicted 2D tracks and are compared with the pseudo-ground truth tracking points $\mathbf{x}_{\text{trk}}^{j,n}$ from CoTracker3 [23].

To mitigate minor focal inaccuracies that may induce scaling shifts, the reprojection loss is computed in a scale-invariant manner. Let $\mathbf{p}_n = \hat{\mathbf{x}}^{j,n}$ and $\mathbf{g}_n = \mathbf{x}_{\text{trk}}^{j,n}$ for $n = 1, \ldots, N$, and denote the image center by c. The scale factor and adjusted predictions are computed together as:

$$\hat{\mathbf{p}}_n = (\mathbf{p}_n - c)\,s + c, \quad s = \frac{1}{N}\sum_{n=1}^{N}\frac{\|\mathbf{g}_n - c\|_2}{\|\mathbf{p}_n - c\|_2}. \tag{6}$$

Then, the scale-invariant 2D reprojection loss is defined as

$$\mathcal{L}_{\text{traj}} = \frac{1}{N}\sum_{n\in\mathcal{I}_j}\|\hat{\mathbf{p}}_n - \mathbf{g}_n\|^2. \tag{7}$$

Similarly, to enforce geometric consistency with the mono-depth predictions from MoGe [59], we use the reconstruction pointmap $\mathbf{X}_j^j$ that transformed to frame j's camera coordinate. The depth of each transformed 3D point is

$$z_{\text{proj}}^{j,n} = \Big(\mathbf{R}^j\,\mathbf{X}_j^{j,n} + \mathbf{T}^j\Big)_z, \tag{8}$$

where $(\cdot)_z$ denotes the third (depth) component. Denote the corresponding mono-depth pseudo ground-truth by $z_{\text{mono}}^{j,n}$. After solving for an optimal scaling factor α^* to align the two depth maps, the scale-invariant mono-depth loss is defined as

$$L_{\text{depth}} = \frac{1}{N}\sum_{n=1}^{N}\Big(\alpha^*\,z_{\text{proj}}^{j,n} - z_{\text{mono}}^{j,n}\Big)^2, \quad \alpha^* = \frac{\sum_i z_{\text{proj}}^{j,n}\,z_{\text{mono}}^{j,n}}{\sum_i\Big(z_{\text{proj}}^{j,n}\Big)^2}. \tag{9}$$

3D Self-Consistency. Beyond the 2D reprojection losses, we introduce a 3D self-consistency term that aligns the tracking pointmap $\mathbf{X}_j^1$ with the reconstruction pointmap $\mathbf{X}_j^j$. Let $\mathcal{I}_1'$ be the set of points in frame 1 that remain visible in frame j, and for each point $n \in \mathcal{I}_1'$, denote its corresponding point (provided by CoTracker) in frame j by n'. We then penalize the distance between their predicted 3D positions:

$$\mathcal{L}_{\text{align}} = \sum_{n\in\mathcal{I}_1'}\big\|\mathbf{X}_j^{1,n} - \mathbf{X}_j^{j,n'}\big\|^2. \tag{10}$$

Minimizing $\mathcal{L}_{\text{align}}$ ensures that both branches produce consistent geometry in the same timestamp.

The overall self-supervision loss is given by:

$$\mathcal{L}_{\text{reproj}} = \mathcal{L}_{\text{traj}} + \lambda_1\,\mathcal{L}_{\text{depth}} + \lambda_2\,\mathcal{L}_{\text{align}}, \tag{11}$$

with λ_1 and λ_2 being the weighting factors. Minimizing $\mathcal{L}_{\text{reproj}}$ aligns the projected 3D structure with the 2D tracking and monocular depth cues and their 3D self-consistency, enabling unsupervised, target-specific refinement of the 3D geometry and point tracking.

Test-Time Adaptation. To address the gap between synthetic pretraining and real-world data, we incorporate reprojection-based losses to enable test-time adaptation in St4RTrack. Our framework supports two adaptation paradigms: **(1) Instance-level adaptation.** During testing, we update St4RTrack on new sequences using only the aforementioned reprojected losses while freezing the *reconstruction branch*. We freeze these weights because both the 2D trajectories and depth are computed under a purely monocular setting, which does not provide view-alignment supervision. This approach preserves the view-alignment capability captured during pretraining. Moreover, since the pretrained network already encodes strong task-relevant representations, this sequence-specific optimization converges rapidly compared to test-time optimization methods that start from scratch. **(2) Domain-level adaptation.** Unlike tabula-rasa approaches such as [27, 57], which require full re-optimization for each new sequence, St4RTrack is an end-to-end learning framework that enables test-time adaptation to align the model from its pretraining data distribution to the target video domain. After adapting to a sparse set of target-domain samples, St4RTrack can directly perform simultaneous reconstruction and tracking on new sequences from the adapted domain without additional optimization.

4. Experiments

St4RTrack performs both dense 3D point tracking and dynamic reconstruction in a unified world coordinate system, all within a single inference. In the following section, we first evaluate our method on 3D tracking and dynamic reconstruction separately, and then present the joint results. We also introduce a new benchmark, *WorldTrack*, for 3D tracking in world coordinates, which is not directly covered by previous methods.

4.1. Experimental Details

Datasets. For fully supervised training, we use three synthetic datasets: Point Odyssey (PO) [69], Dynamic Replica (DR) [22], and Kubric [15]. All three datasets contain scene and camera motion and provide mesh vertex positions as ground-truth 3D point trajectories. We randomly sample

Table 1. **World Coordinate 3D Point Tracking.** We report the performance of average points under distance (APD_{3D}) after global median alignment. We evaluate the accuracy of both all points and dynamic points. The best results are **bold**. See Appendix B.2 for more results.

		All Points				Dynamic Points			
Category	**Methods**	PO	DR	ADT	PStudio	PO	DR	ADT	PStudio
Combinational	SpaTracker+RANSAC-Procrustes	44.03	55.01	50.87	52.05	53.77	58.58	66.49	52.05
	SpaTracker+MonST3R	47.65	55.49	51.95	50.16	58.61	59.21	69.94	50.16
Feed-forward	MonST3R	33.47	58.06	74.35	51.32	39.36	51.86	67.92	51.32
	SpaTracker	38.54	54.85	45.65	62.59	51.20	58.65	67.65	62.59
	St4RTrack (Ours)	**67.95**	**73.74**	**76.01**	**69.67**	**68.72**	**68.13**	**75.34**	**69.67**

Table 2. **World Coordinate 3D Reconstruction.** We report performance on both Point Odyssey (PO) and TUM-Dynamics after global median scaling. The best results are in **bold**.

		Point Odyssey		TUM-Dynamics	
Category	**Methods**	**EPE↓**	**APD↑**	**EPE↓**	**APD↑**
w/ Global Align.	DUSt3R+GA	0.6088	43.90	0.3147	70.49
	MASt3R+GA	0.4030	60.44	0.5186	68.38
	MonST3R+GA	0.2629	72.31	0.3429	63.87
Feed-forward	DUSt3R	0.6386	45.79	0.2891	72.26
	MASt3R	0.4644	56.90	0.5510	66.22
	MonST3R	0.3044	68.25	0.3646	61.38
	St4RTrack (Ours)	**0.2406**	**78.73**	**0.1854**	**83.42**

24 frames with a stride of 1∼6 for each sample sequence. We also filter out less semantically meaningful sequences in PO, resulting in a total of 9.8k sequences for PO, 8.5k for DR, and 5.7k for Kubric dataset.

Training and Inference. During training, we sample 600 sequences from each dataset per epoch. We use the AdamW optimizer with a learning rate of 5×10^{-5} and a mini-batch size of 1 per GPU. The model is trained for 50 epochs on 4 A100 GPUs, which takes about one day. For test-time adaptation, we run 500 optimization steps on a single sequence, taking approximately 5 minutes on 4 A100 GPUs. At inference time, the model runs at 30 FPS on an RTX 4090. Although the model is trained on sequences of 24 frames, our pair-wise approach allows it to operate on arbitrarily long videos during inference. Refer to Appendix C for more details regarding test-time adaptation.

4.2. 3D Tracking in World Coordinates

Datasets. 3D tracking in world coordinates is a critical aspect that has been largely overlooked by previous benchmarks [26], which are limited to camera coordinate systems. To address this limitation, we propose a new benchmark for 3D tracking in world coordinates. Our benchmark leverages two real-world datasets—Aerial Digital Twin (ADT) [40] and Panoptic Studio [21]—by converting the TAPVid-3D [26] sequences to world coordinates using paired extrinsic parameters. However, it is noteworthy that the limitations of these datasets: the ADT sequences exhibit minimal scene motion, while the Panoptic Studio lacks camera motion. To overcome these shortcomings, we include two additional synthetic test sets from Point Odyssey and Dynamic Replica, which have both scene and camera motion. In total, our benchmark comprises four datasets, each containing 50 sequences of 64 frames. Refer to Appendix B.1 for more details and visual examples of the benchmark.

Evaluation Metrics. We follow the TAPVid-3D protocol and use the *Average percent of Points within Delta (APD)* metric for evaluation. Specifically, we first align the predicted 3D point trajectories with the ground truth by normalizing them with their global median. We then compute the prediction error and measure the percentage of points whose error falls below a given threshold δ_{3D} (with $\delta_{3D} \in \{0.1\text{m}, 0.3\text{m}, 0.5\text{m}, 1.0\text{m}\}$) over the first 64 frames. Let $\hat{\mathbf{P}}_t^i$ denote the i-th predicted point at time t and $\mathbf{P}_t^i$ denote its corresponding ground-truth location. The resulting APD_{3D} is then computed as follows:

$$APD_{3D} \equiv \sum_{i,t} \mathbb{1}\Big(\|\hat{\mathbf{P}}_t^i - \mathbf{P}_t^i\| < \delta_{3D}\Big), \tag{12}$$

where $\mathbb{1}(\cdot)$ is the indicator function and $\|\cdot\|$ denotes the Euclidean norm.

Baselines. Since no existing work explicitly performs 3D tracking in world coordinates, for our feedforward baselines, we compare against the camera coordinate 3D tracking method SpatialTracker [64] and a dynamic 3D reconstruction method MonST3R [67] (as a non-tracking baseline). In addition, we implement two combinational baselines for world coordinate 3D tracking. The first baseline applies Procrustes alignment [54] and RANSAC [11] to the camera coordinate 3D tracks predicted by SpatialTracker to offset the camera motion. The second baseline leverages the camera poses predicted by the dynamic SLAM method MonST3R to compensate for camera motion.

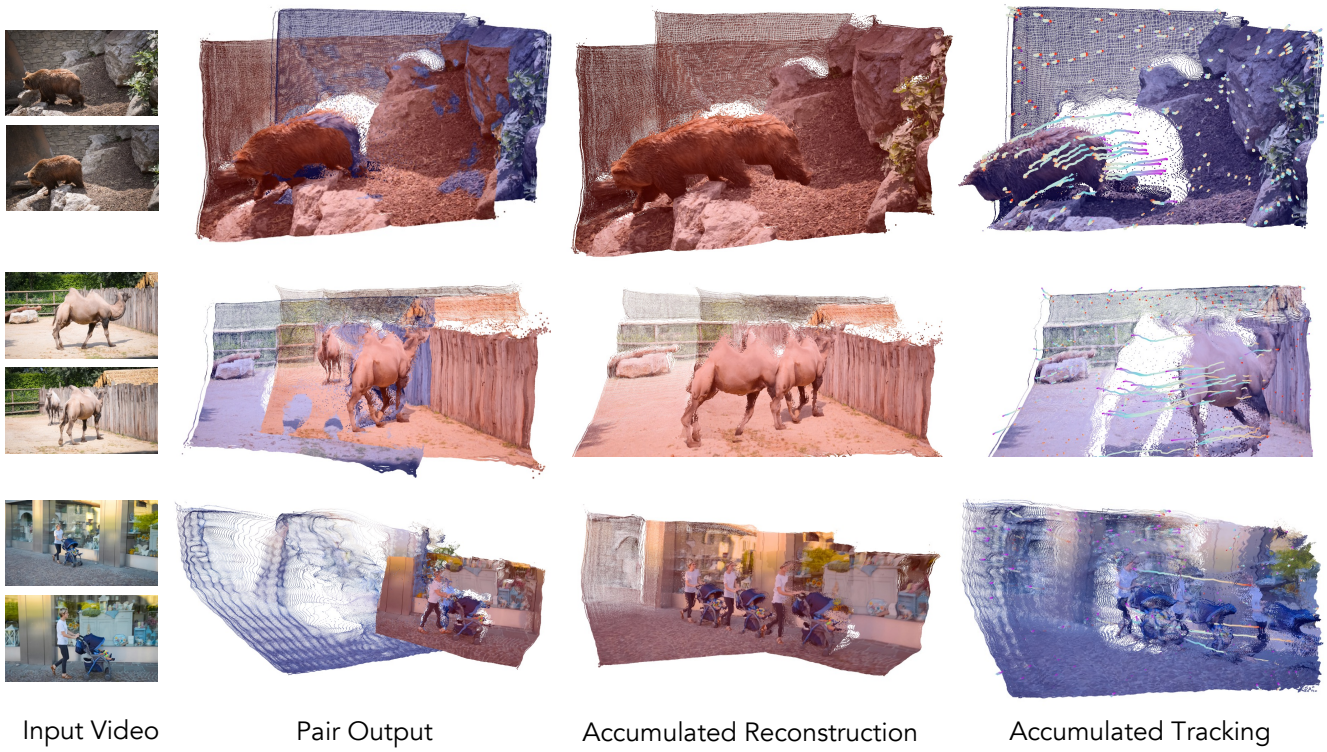

Figure 4. **Qualitative Results.** From left to right, we show our results in *feed-forward inference*: 1) the input video, 2) two pointmaps at frame j overlayed together, 3) the accumulated reconstruction branch result, and 4) the accumulated tracking branch result. The accumulated reconstruction demonstrates a stable reconstruction of the dynamic scene geometry, while the accumulated tracking illustrates long-term, dense tracking of scene motion. More qualitative results in Appendices B.3 and D.

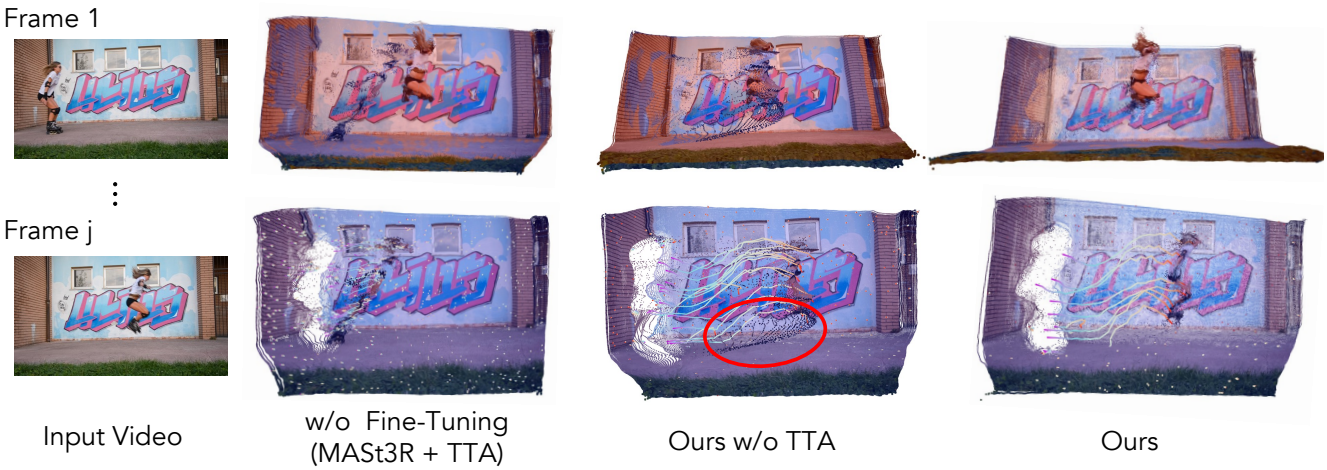

Figure 5. **Ablation Study.** We show the qualitative comparison of our full method and variants that do not pretrain or do not adapt in test time. Predicted pointmaps from two heads are visualized together.

Results. As shown in Tab. 1, we achieve state-of-the-art performance, with test-time adaptation proving particularly beneficial for dynamic points. Notably, on the Panoptic Studio dataset, which is captured with a fixed camera and can be considered a fair benchmark for camera coordinate tracking methods, our approach still outperforms SpatialTracker [64]. It is worth noting that although our model is trained on sequences of 24 frames, it generalizes well to longer sequences, including 64-frame videos. Refer to Appendix B.2 for more results.

4.3. Dynamic 3D Reconstruction

Datasets. We evaluate on both synthetic and real-world data. For the synthetic data, we use Point Odyssey [69]. For real-world evaluation, we employ TUM-Dynamics [49], a subset of a SLAM dataset featuring moving people, dense depth maps, and accurate camera poses.

Evaluation Metrics. Unlike prior works [58, 67] that separately evaluate video depth and camera pose estimation, we directly compare the reconstructed 3D point clouds to the ground truth using the Average percent of Points within Distance (APD) and End-Point Error (EPE) metrics. We filter out ambiguous floating points in the ground truth data and align the point clouds for each sequence using the median scale before evaluation.

Baselines. We compare our method against MonST3R, MASt3R and DUSt3R, both with global alignment (w/ GA) and in feedforward mode. For the feedforward baselines, we construct image pairs of a video in the form of St4RTrack, that align all frames to a common anchor frame.

Results. As in Tab. 2, we also achieve state-of-the-art performance on the task of 3D reconstruction in the world coordinate. Although MonST3R is designed for 3D reconstruction for dynamic scenes, it still underperforms St4RTrack even with global alignment. This further highlights the benefit of jointly tracking and reconstruction. Since we freeze the reconstruction head to preserve the 3D prior, 3D reconstruction results are similar with test-time adaptation.

4.4. Joint Tracking and Reconstruction in the World

Our method simultaneously predicts 3D point trajectories and 3D point clouds in a single feed-forward pass, which we evaluate separately in previous sections. In this section, we present qualitative results that visualize both the raw 3D point trajectories and 3D point clouds within the same world coordinates, as shown in Fig. 4. The "pair output" result demonstrates that the outputs from the tracking and reconstruction branches align well at the same time step. Additionally, the accumulated reconstruction indicates consistency in static regions, while the accumulated tracking shows that our method estimates accurate and smooth 3D tracks over time.

4.5. Ablation Study

We perform an ablation study to evaluate two key design choices of our method and present qualitative results in Fig. 5. First, we assess the effectiveness of our pretraining stage by directly applying test-time adaptation to a pretrained checkpoint from MonST3R [67], without finetuning the base model on our training datasets. As shown in Fig. 5 (column 2), the baseline exhibits unaligned pointmaps between the tracking and reconstruction branches, underscoring the importance of pretraining on synthetic data—even in the presence of a domain gap with real-world data.

Second, we evaluate the impact of our proposed test-time adaptation. As demonstrated in Fig. 5 (column 3), the adapted model successfully corrects drifting points, ensuring that points consistently trace back to their original spatial locations in the first frame. This finding supports our analysis that small-scale training data alone is insufficient for fine-grained prediction, particularly at the boundaries of moving objects. In contrast, St4RTrack produces spatially aligned pointmaps with significantly fewer drifting points. The colorful tails in the visualization indicate the long-term trajectories, while the accurately predicted geometry in dynamic regions results in a crisp and precise rendering.

Refer to Appendix C.2 for more ablation studies.

5. Discussion

Despite St4RTrack presents a promising step toward a unified understanding of dynamic scene geometry and motion in a minimalist way, a challenge arises from the per-frame setting. In particular, issues such as scale misalignment, large camera movements, and occlusions are not fully resolved. Incorporating temporal attention across multiple frames would help capture richer motion priors and alleviate these limitations. Another limitation arises from the pretraining dataset's limited diversity and realism in both geometry and motion, necessitating test-time adaptation to improve St4RTrack 's robustness in out-of-distribution scenarios. However, it still struggles with highly complex motions. Expanding the training set is therefore a key direction for future work. We envision that large-scale pretraining, when compute permits, could significantly boost St4RTrack's performance and enable it to better handle complex, in-the-wild videos.

6. Conclusion

We introduce St4RTrack, a feed-forward framework that *simultaneously* achieves 3D point tracking and dynamic reconstruction *in the world coordinate* from monocular videos using a unified representation. Alongside, we present a novel benchmark, *WorldTrack*, for systematically evaluating dynamic 3D scene geometry and motion estimation in a global reference frame. Our method achieves state-of-the-art performance on both synthetic and real-world datasets, while also extending beyond fully supervised paradigms by enabling test-time adaptation.

7. Acknowledgements

We would like to thank Aleksander Holynski, Yifei Zhang, Chung Min Kim, Brent Yi, and Zehao Yu for helpful discussions. We especially thank Aleksander Holynski for his guidance and feedback.

References

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

[2] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth. *arXiv preprint arXiv:2302.12288*, 2023. 2

[3] Michael J Black and Padmanabhan Anandan. A framework for the robust estimation of optical flow. In *1993 (4th) International Conference on Computer Vision*, pages 231–236. IEEE, 1993. 2

[4] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, pages 690–696. IEEE, 2000. 3

[5] Hansheng Chen, Wei Tian, Pichao Wang, Fan Wang, Lu Xiong, and Hao Li. Epro-pnp: Generalized end-to-end probabilistic perspective-n-points for monocular object pose estimation. *arXiv preprint arXiv:2303.12787*, 2023. 5, 13

[6] Yuchao Dai, Hongdong Li, and Mingyi He. A simple prior-free method for non-rigid structure-from-motion factorization. *International Journal of Computer Vision*, 107:101–122, 2014. 3

[7] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067, 2007. 2

[8] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adria Recasens, Lucas Smaira, Yusuf Aytar, Joao Carreira, Andrew Zisserman, and Yi Yang. TAP-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022. 2

[9] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. TAPIR: Tracking any point with per-frame initialization and temporal refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10061–10072, 2023. 2

[10] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006. 2

[11] Martin A Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 5, 7

[12] Katerina Fragkiadaki, Marta Salas, Pablo Arbelaez, and Jitendra Malik. Grouping-based low-rank trajectory completion and 3d reconstruction. *advances in neural information processing systems*, 27, 2014. 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023. 3

[14] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. *Advances in Neural Information Processing Systems*, 35:33768–33780, 2022. 3

[15] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3749–3761, 2022. 6, 13

[16] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision*, pages 59–75. Springer, 2022. 2

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

[18] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. 2

[19] Junhwa Hur and Stefan Roth. Self-supervised monocular scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7396–7405, 2020. 2

[20] Linyi Jin, Richard Tucker, Zhengqi Li, David Fouhey, Noah Snavely, and Aleksander Holynski. Stereo4d: Learning how things move in 3d from internet stereo videos. *arXiv preprint arXiv:2412.09621*, 2024. 3

[21] Hanbyul Joo, Tomas Simon, Xulong Li, Hao Liu, Lei Tan, Lin Gui, Sean Banerjee, Timothy Godisart, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic studio: A massively multiview system for social interaction capture, 2016. 7

[22] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Dynamicstereo: Consistent dynamic depth from stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13229–13239, 2023. 1, 5, 6

[23] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker3: Simpler and better point tracking by pseudo-labelling real videos. 2024. 2, 5, 6

[24] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *Proc. ECCV*, 2024. 1, 2, 5

[25] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1611–1621, 2021. 2

[26] Skanda Koppula, Ignacio Rocco, Yi Yang, Joe Heyward, João Carreira, Andrew Zisserman, Gabriel Brostow, and Carl Doersch. Tapvid-3d: A benchmark for tracking any point in 3d, 2024. 7, 13

[27] Jiahui Lei, Yijia Weng, Adam W. Harley, Leonidas Guibas, and Kostas Daniilidis. MoSca: Dynamic gaussian fusion from casual videos via 4d motion scaffolds. *arXiv preprint arXiv:2405.17421*, 2024. 3, 6

[28] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. EPnP: An accurate O(n) solution to the PnP problem. *IJCV*, 81:155–166, 2009. 5

[29] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European Conference on Computer Vision*, pages 71–91. Springer, 2024. 5

[30] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5521–5531, 2022. 3

[31] 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*, pages 6498–6508, 2021. 3

[32] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4273–4284, 2023. 3

[33] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 2

[34] Qingming Liu, Yuan Liu, Jiepeng Wang, Xianqiang Lyv, Peng Wang, Wenping Wang, and Junhui Hou. Modgs: Dynamic gaussian splatting from casually-captured monocular videos, 2024. 3

[35] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI'81: 7th international joint conference on Artificial intelligence*, pages 674–679, 1981. 2

[36] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis, 2023. 3

[37] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: A versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 2

[38] Tuan Duc Ngo, Peiye Zhuang, Chuang Gan, Evangelos Kalogerakis, Sergey Tulyakov, Hsin-Ying Lee, and Chaoyang Wang. DELTA: Dense efficient long-range 3d tracking for any video. In *International Conference on Learning Representations (ICLR)*, 2025. 1, 3

[39] David Novotny, Nikhila Ravi, Benjamin Graham, Natalia Neverova, and Andrea Vedaldi. C3dpo: Canonical 3d pose networks for non-rigid structure from motion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7688–7697, 2019. 3

[40] Xiaqing Pan, Nicholas Charron, Yongqian Yang, Scott Peters, Thomas Whelan, Chen Kong, Omkar Parkhi, Richard Newcombe, and Carl Yuheng Ren. Aria digital twin: A new benchmark dataset for egocentric 3d machine perception, 2023. 7

[41] 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. *arXiv preprint arXiv:2106.13228*, 2021. 3

[42] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. Unidepth: Universal monocular metric depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10106–10116, 2024. 2

[43] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10318–10327, 2021. 3

[44] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 2

[45] Michael Rubinstein, Ce Liu, and William T Freeman. Towards longer long-range motion trajectories. 2012. 2

[46] P. Sand and S. Teller. Particle video: Long-range motion estimation using point trajectories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 2195–2202, 2006. 2

[47] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 2

[48] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, pages 519–528. IEEE, 2006. 2

[49] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. pages 573–580, 2012. 9

[50] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 2

[51] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow, 2020. 1, 2

[52] 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. 2

[53] Zachary Teed and Jia Deng. Raft-3d: Scene flow using rigid-motion embeddings. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8375–8384, 2021. 2

[54] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04): 376–380, 1991. 7

[55] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 722–729. IEEE, 1999. 2

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

[57] Qianqian Wang, Vickie Ye, Hang Gao, Weijia Zeng, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. *arXiv preprint arXiv:2407.13764*, 2024. 3, 6

[58] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 1, 2, 9

[59] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision, 2024. 2, 6

[60] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2043–2050. IEEE, 2017. 2

[61] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jérôme Revaud. Dust3r: Geometric 3d vision made easy. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20697–20709, 2023. 2, 3, 5, 13

[62] Endre Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal, First Series*, 43:355–386, 1937. 5

[63] 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*, pages 20310–20320, 2024. 3

[64] Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. Spatialtracker: Tracking any 2d pixels in 3d space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2, 3, 7, 8

[65] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10371–10381, 2024. 2

[66] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20331–20341, 2024. 3

[67] 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 *International Conference on Learning Representations (ICLR)*, 2025. 1, 2, 7, 9

[68] Zhoutong Zhang, Forrester Cole, Zhengqi Li, Michael Rubinstein, Noah Snavely, and William T. Freeman. Structure and motion from casual videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 20–37, 2022. 2

[69] Yang Zheng, Adam W Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19855–19865, 2023. 5, 6, 9

[70] Xiaowei Zhou, Menglong Zhu, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis. Sparseness meets deepness: 3d human pose estimation from monocular video. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4966–4975, 2016. 3

St4RTrack: Simultaneous 4D Reconstruction and Tracking in the World

Supplementary Material

Contents

A. Differentiable Camera Pose Estimation

We seek to backpropagate the projection loss to the 3D pointmaps through the camera pose. To this end, we build upon the RANSAC-PnP approach from DUSt3R [61], which initially solves for pose $\mathbf{P}^*$ (rotation and translation) by matching per-pixel 2D-3D correspondences in the reconstruction pointmap $\mathbf{X}^j_j$. However, RANSAC is inherently non-differentiable.

To enable end-to-end gradients, we adopt the derivative-based Gauss-Newton (GN) solver inspired by EPro-PnP [5]. Specifically, after obtaining a *detached* solution $\mathbf{P}^*$ from RANSAC-PnP, we refine it using one GN step:

$$\Delta\mathbf{P} \;=\; -\big(J^\top J\big)^{-1} J^\top F(\mathbf{P}^*), \tag{13}$$

where $F(\mathbf{P}^*) = [\, f_1^\top(\mathbf{P}^*), \ldots, f_N^\top(\mathbf{P}^*)]^\top$ is the flattened reprojection error for all N points, and $J = \frac{\partial F(\mathbf{P})}{\partial \mathbf{P}}\big|_{\mathbf{P}=\mathbf{P}^*}$ is its Jacobian. The term $J^\top J$ approximates the Hessian of the negative log-likelihood (NLL), while $J^\top F(\mathbf{P}^*)$ is the gradient of the NLL with respect to the pose. This gradient effectively *pushes* the incremental solution $\Delta\mathbf{P}$ toward reducing the reprojection errors. The final *differentiable* pose estimate is:

$$\mathbf{P} \;=\; \mathbf{P}^* \;+\; \Delta\mathbf{P}. \tag{14}$$

Since $\mathbf{P}^*$ is detached, only the GN increment $\Delta\mathbf{P}$ remains differentiable, allowing the reprojection loss to backpropagate through $\mathbf{P}$ and thus refine the 3D pointmaps.

B. Details on the *WorldTrack* Benchmark

B.1. Datasets

Dataset Preparation. For the two real-world datasets, we adopt the 3D camera coordinate tracking annotation of ADT and Panoptic Studio from the TAPVID-3D dataset. Using the paired camera parameters provided, we transform the camera coordinates to the world coordinate system. For the two synthetic datasets, we use the test sets from Point Odyssey and Dynamic Replica Dataset. We uniformly downsample the query points to approximately 1,000 per sequence. Each sequence contains 128 sampled frames, though only the first 64 frames are used for evaluation. This results in 160 and 140 sequences from Point Odyssey and Dynamic Replica, respectively. From these, we randomly sample 50 sequences per dataset for evaluation.

Filtering Criteria. To ensure data quality, we apply several filtering strategies: For TUM, we keep the pixels which associated with depth values within 0.1 - 5 meters, as the depth camera is less accurate at long range. For Point Odyssey, we exclude sequences generated in the Kubric style [15] due to their lack of realism. We also remove scenes with ambiguous depth (e.g., heavy foggy conditions), and any frames where the camera intrinsics are dynamic.

B.2. Additional Quantitative Evaluation

Following TAPVid-3D [26], we adopt global median scale alignment, since both our predictions and the ground truth use the first frame's camera coordinate system as the world coordinate. The Average Percent of Points within Distance (APD_{3D}) measures the overall accuracy of the 3D trajectories in world coordinates, while Euclidean endpoint error (EPE) offers a complementary perspective on localization accuracy. Accordingly, we additionally report EPE re-

Table 3. **World Coordinate 3D Point Tracking (EPE - Global Median) .** We report end-point error (EPE; lower is better) for both all points and dynamic points after global median alignment. The best (lowest) values are in **bold**.

		All Points				Dynamic Points			
Category	**Methods**	PO	DS	ADT	PStudio	PO	DS	ADT	PStudio
Combinational	SpaTracker+RANSAC-Procrustes	0.6408	0.9185	0.5876	0.4266	0.4358	1.0444	0.1600	0.4266
	SpaTracker+MonST3R	0.5917	0.8823	0.5362	0.4837	0.4085	0.9136	0.1511	0.4837
Feed-forward	MonST3R	0.9021	0.4387	0.2721	0.4568	0.6452	0.5313	0.1578	0.4568
	SpaTracker	0.7499	0.9274	0.8530	0.3094	0.4695	1.0828	0.1628	0.3094
	Ours	**0.3140**	**0.2682**	**0.2680**	**0.2637**	**0.2970**	**0.2961**	**0.1212**	**0.2637**

Table 4. **World Coordinate 3D Point Tracking (APD/EPE - SIM(3)).** Each cell shows APT_{3D} (higher is better) / EPE (lower is better) after global IM(3) alignment. The best APT (highest) and the best EPE (lowest) in every column are **bold**.

		All Points				Dynamic Points			
Category	**Methods**	PO	DR	ADT	PStudio	PO	DR	ADT	PStudio
Combinational	SpaTracker+Procrustes	46.20/0.5670	55.10/0.5292	59.40/0.4027	67.82/0.2660	61.00/0.3338	61.65/0.3720	**88.65**/0.0596	67.82/0.2660
	SpaTracker+MonST3R	48.23/0.5388	56.78/0.5069	60.01/0.3910	64.32/0.2971	61.78/0.3290	61.88/0.3681	87.32/**0.0485**	64.32/0.2971
Feed-forward	MonST3R	37.62/0.8073	64.83/0.3725	79.48/0.1881	64.11/0.3015	48.95/0.4768	55.36/0.3872	84.73/0.0720	64.11/0.3015
	SpaTracker	43.17/0.6079	54.65/0.5324	53.96/0.4963	**80.76/0.1650**	60.49/0.3374	61.32/0.3750	87.68/0.0616	**80.76/0.1650**
	Ours	**71.84/0.2774**	**76.28/0.2436**	**83.03/0.1631**	76.97/0.1969	**67.43/0.2870**	**67.90/0.2627**	85.34/0.0688	76.97/0.1969

Table 5. **World Coordinate 3D Reconstruction (APD/EPE - SIM(3)).** Results on Point Odyssey (PO) and TUM-Dynamics after global SIM(3) alignment. Lower is better for EPE, higher is better for APT. The best results are in **bold**.

		Point Odyssey		TUM-Dynamics	
Category	**Methods**	**EPE↓**	**APT↑**	**EPE↓**	**APT↑**
w/ Global Align.	DUSt3R+GA	0.3541	62.42	0.2989	69.23
	MASt3R+GA	0.3717	61.31	0.5294	49.81
	MonST3R+GA	**0.2601**	69.31	0.3173	66.00
Feed-forward	DUSt3R	0.4251	56.70	0.3092	67.48
	MASt3R	0.4473	55.09	0.5862	45.43
	MonST3R	0.3462	62.10	0.3508	62.83
	St4RTrack	0.2741	**69.53**	**0.2413**	**74.14**

sults on the WorldTrack benchmark. As shown in Table 3, St4RTrack attains state-of-the-art EPE on all sub-test sets, consistent with the APD_{3D} results in the main paper.

Beyond alignment to the first camera's pose, we also evaluate under SIM(3) alignment (i.e., SE(3) plus a global scale factor) for both APD_{3D} and EPE to assess performance of 3D tracking (See Tab. 4) and reconstruction (See Tab. 5) under a more flexible registration. Comprehensive evaluations show that St4RTrack achieves state-of-the-art performance in most scenarios.

B.3. Qualitative Evaluation

We present the qualitative results of our fully feed-forward approach on WorldTrack benchmark. Specifically, we show the reconstruction results in Fig. 6 (TUM-Dynamics) and Fig. 7 (Point Odyssey). We show the tracking results of all four datasets in Fig. 8.

C. Details on Test-Time Adaptation

C.1. Implementation Details

We set the weights of different loss factors in Eq. (11) to $\lambda_{\text{traj}} = 1$, $\lambda_{\text{depth}} = 10$, and $\lambda_{\text{align}} = 5$. For WorldTrack evaluation, the two test-time adaptations are set up as follows: **Sequence-Level (Instance) Adaptation:** Fine-tune a separate model for each of the 50 sequences. We sample 300 frames per epoch, train for 3 epochs, and use a batch size of 4. **Dataset-Level (Domain) Adaptation:** Fine-tune a single model on the entire dataset. We sample 100 frames per epoch, train for 15 epochs, and use a batch size of 4.

C.2. Ablation Studies

We ablate (1) the performance gain from the feed-forward St4RTrack, instance-level adaptation, and domain-level

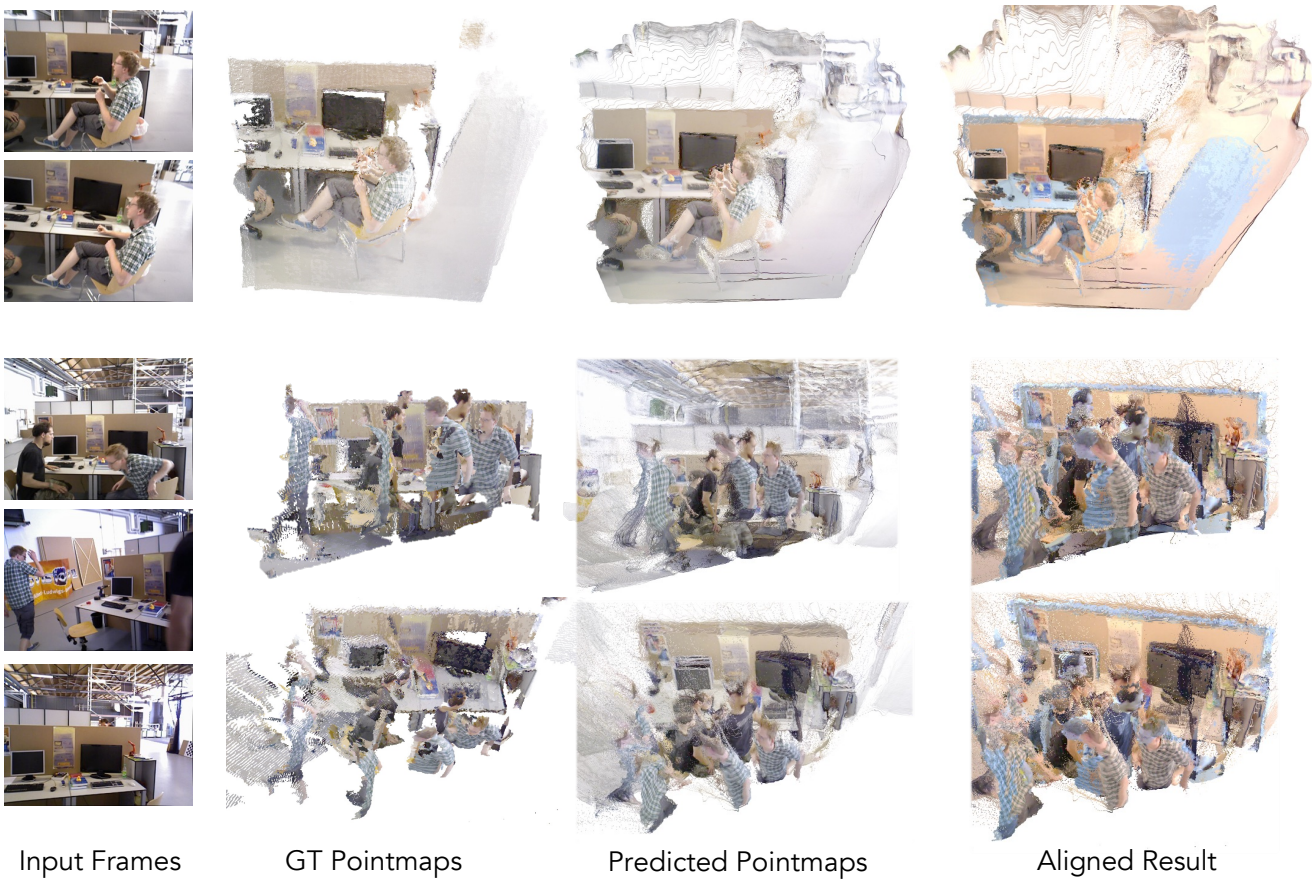

Figure 6. **Reconstruction Results of St4RTrack on TUM-Dynamics Dataset.** From left to right, we show 1) the sampled frames from the input sequence of 64 frames, 2) the subsampled ground truth pointmaps, 3) the predicted pointmaps of our method, and 4) the aligned results of the predicted and GT pointmaps with median-scale. Note that the reconstruction result is inferred in a feed-forward way.

Table 6. **World Coordinate 3D Tracking (Median-Scale).** End-point error (EPE ↓) and APT_{3D} ↑ for DR and PStudio after global median scaling. Best (lowest EPE / highest APT_{3D}) in each column is shown in **bold**.

	DR		PStudio	
Methods	**EPE↓**	**APT↑**	**EPE↓**	**APT↑**
Spatialtracker+Procrustes-RANSAC	0.9185	55.01	0.4266	52.05
St4RTrack	0.2682	73.74	0.2637	69.67
St4RTrack + TTA (per-sequence)	**0.2472**	**76.07**	**0.2243**	**73.71**
St4RTrack + TTA (per-dataset)	0.2547	74.86	0.2280	73.30
w/o trajectory loss	0.2767	72.75	0.2421	72.50
w/o depth loss	0.5524	48.22	0.2975	66.50
w/o alignment loss	0.3263	66.65	0.3357	60.07
w/o pre-training	0.3377	65.50	0.3801	57.71

adaptation, and (2) the contribution of each TTA component by omitting individual elements. Table 6 summarizes our findings. First, both TTA variants yield substantial improvements over the feed-forward mode, with instance-level adaptation achieving the highest accuracy, as it fully specializes to each test sequence. Second, removing any single TTA component—trajectory loss, depth loss, alignment loss, or synthetic pretraining—causes a performance drop in all scenarios, underscoring the necessity of each component.

D. Additional Results

Below, we present additional qualitative results for both feed-forward inference (Fig. 9) and test-time adaptation (Fig. 10).

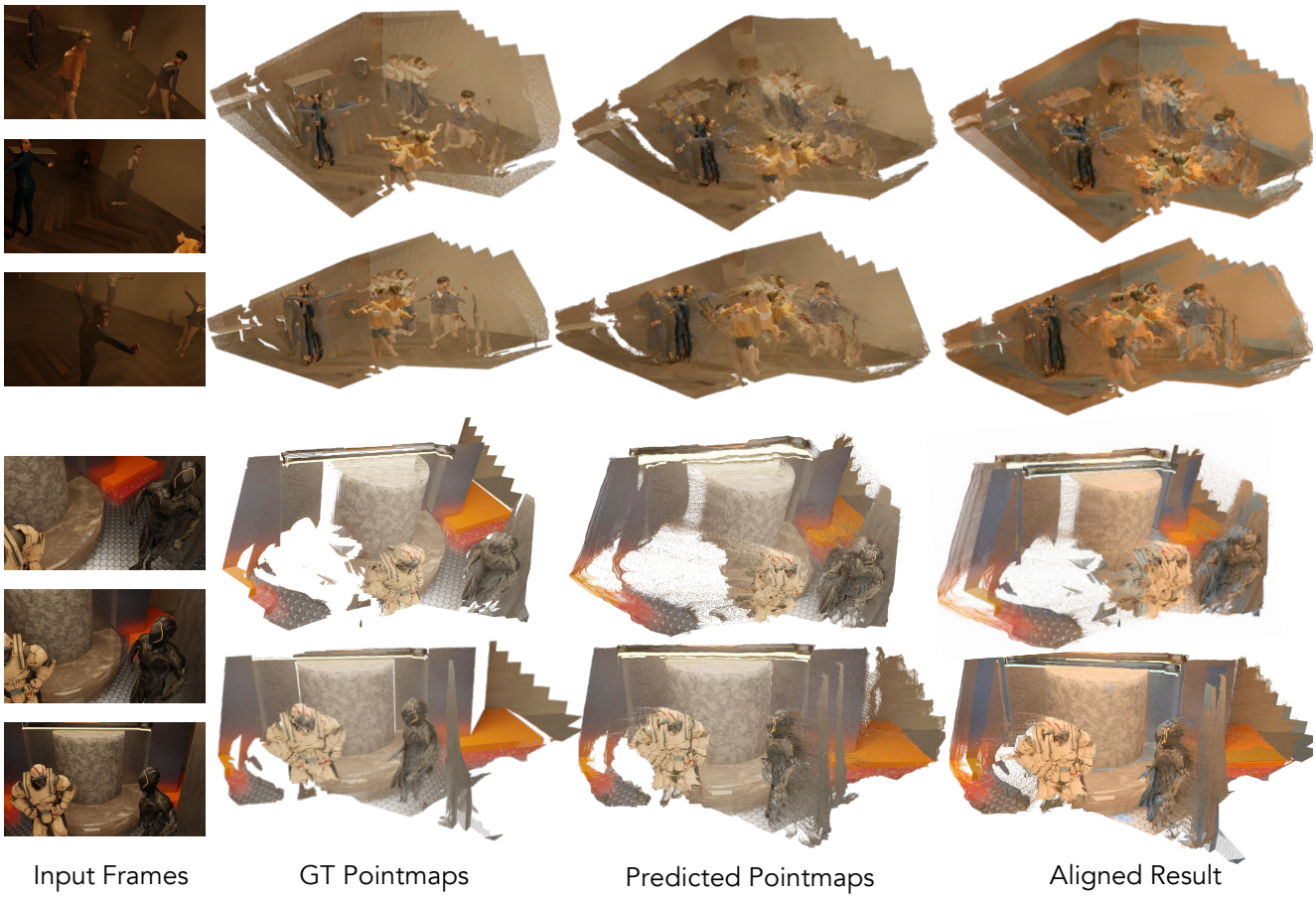

Figure 7. **Reconstruction Results of St4RTrack on Point Odyssey Dataset.** From left to right, we show 1) the sampled frames from the input sequence of 64 frames, 2) the subsampled ground truth pointmaps, 3) the predicted pointmaps of our method, and 4) the aligned results of the predicted and GT (yellow) pointmaps with median-scale. Note that the reconstruction result is inferred in a feed-forward way.

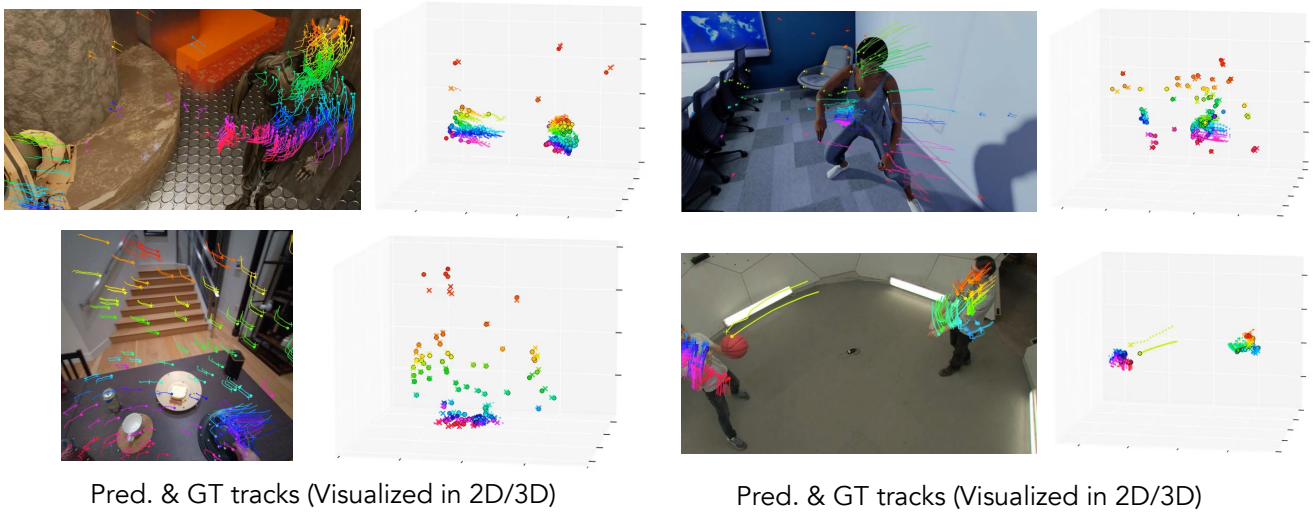

Figure 8. **Tracking Results of St4RTrack on WorldTrack Benchmark.** We show the 2D and 3D visualized results of the predicted tracks (visualized as "+") aligned with the ground truth tracks (visualized as "•"). The corresponding datasets are Point Odyssey (top left), Dynamic Replica (top right), Arial Digital Twin (bottom left), and Pnapotic Studio (bottom right).

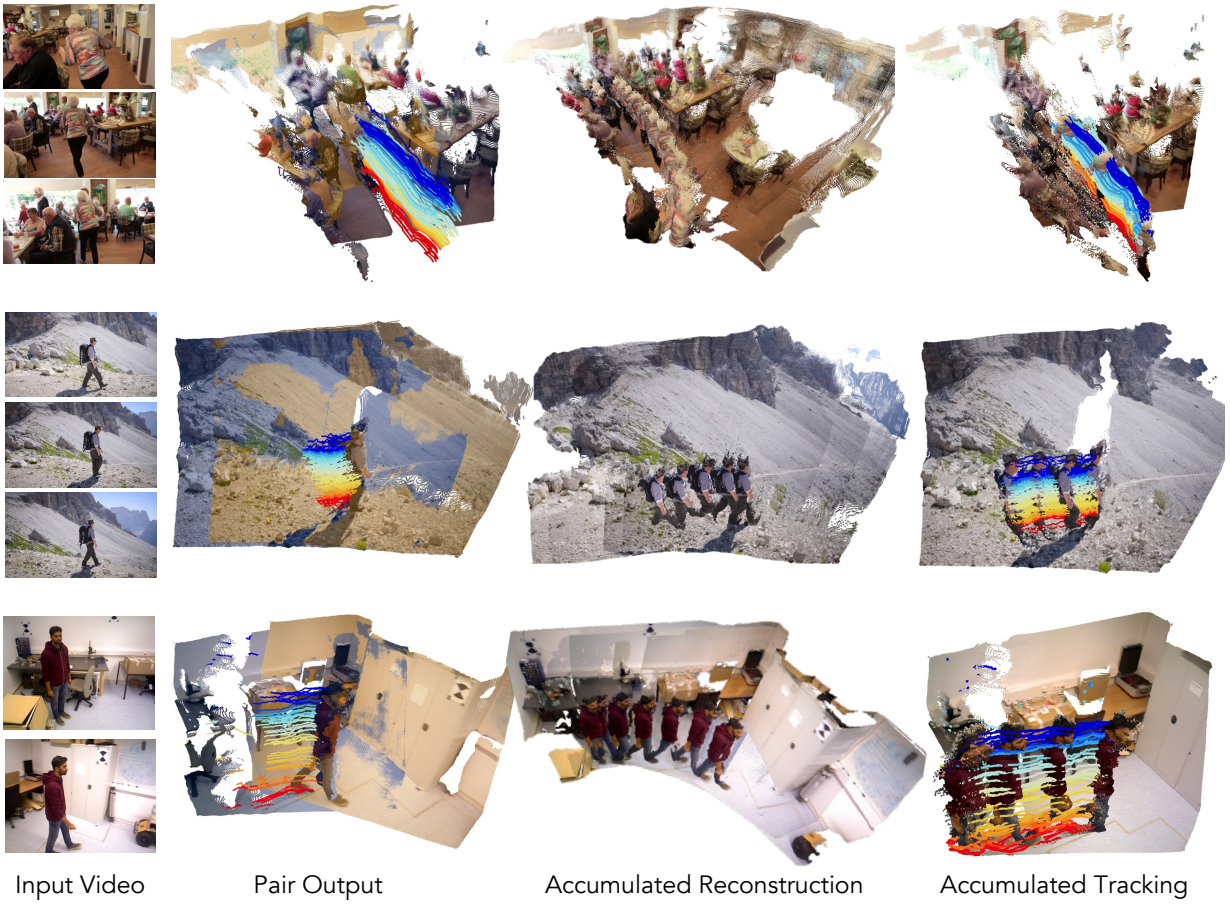

Figure 9. **Fully *Feed-Forward* Inference Results of St4RTrack.** We show from left to right: 1) the input video, 2) the pairwise output for tracking (in blue) and reconstruction (in yellow) of the same frame, 3) the accumulated results of the reconstruction pointmaps, and 4) the accumulated results of the tracking pointmaps. Note that we anchor the *middle frame* as the reference frame for point tracking.

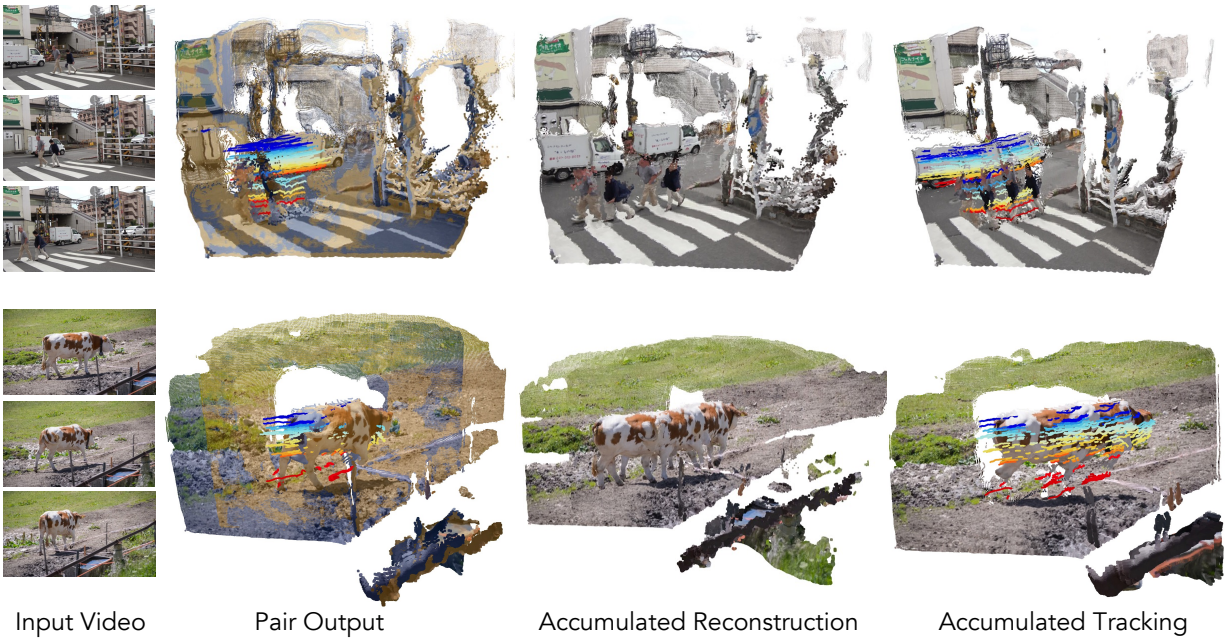

Figure 10. **Test-Time Adaptation Results of St4RTrack.** The first frame is set to be the reference frame.