MV-TAP: Tracking Any Point in Multi-View Videos

Jahyeok Koo* Inès Hyeonsu Kim* Mungyeom Kim Junghyun Park
Seohyeon Park Jaeyeong Kim Jung Yi Seokju Cho Seungryong Kim

KAIST AI

https://cvlab-kaist.github.io/MV-TAP

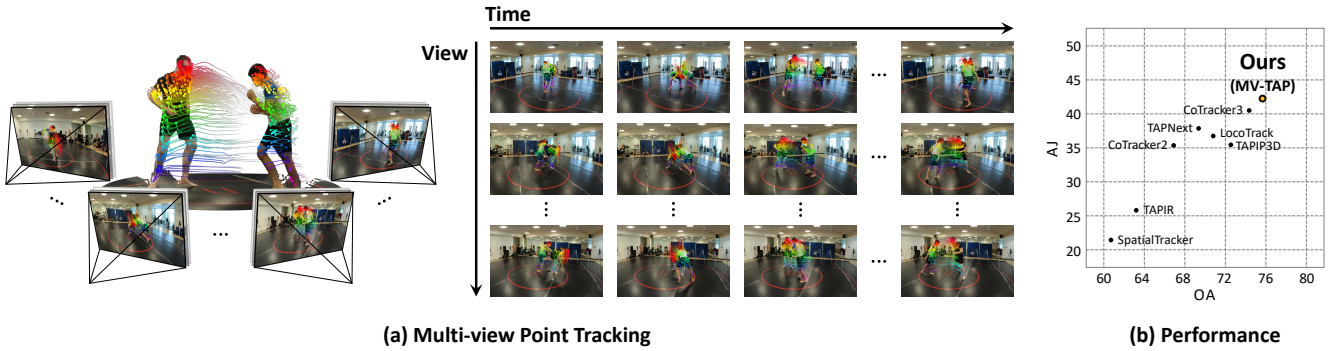

Figure 1: We present **MV-TAP** (**T**racking **A**ny **P**oint in **M**ulti-view **V**ideos), a model designed to effectively integrate information across multiple viewpoints for robust and high-quality point tracking. (a) We visualize the result of MV-TAP on Harmony4D [22]. (b) MV-TAP achieves noticeable gains over other baselines [6, 9, 20, 21, 39, 40, 42], demonstrating its ability to leverage multi-view information.

Abstract

Multi-view camera systems enable rich observations of complex real-world scenes, and understanding dynamic objects in multi-view settings has become central to various applications. Point tracking serves as a key mechanism for capturing dynamic motion. However, conventional single-view approaches often fail due to the limited geometric information available in monocular video, which becomes a critical bottleneck for multi-view extension. In this work, we present MV-TAP, a robust point tracker that tracks points across multi-view videos of dynamic scenes by leveraging cross-view information. MV-TAP utilizes camera geometry and a cross-view attention mechanism to aggregate spatio-temporal information across views, enabling more complete and reliable trajectory estimation in multi-view videos. To support this task, we construct a large-scale synthetic training dataset and real-world evaluation sets tailored for multi-view tracking. Extensive experiments demonstrate that MV-TAP outperforms existing point-tracking methods on challenging benchmarks, establishing an effective baseline for advancing research in multi-view point tracking.

*Equal contribution.

1. Introduction

Multi-view camera systems have advanced significantly in recent years, enabling the understanding of complex real-world scenes. Building upon these advances, understanding dynamic objects in multi-view systems has become crucial for a wide range of applications, including motion capture [5, 12, 19, 22], robot manipulation [17, 33, 41], and autonomous driving [3, 25].

Point tracking [6, 8, 14, 20] has been widely used to understand dynamic scenes by providing fine-grained spatio-temporal correspondences across video frames and indicating point-wise occlusions. Hence, it supports a wide range of downstream tasks, including embodied AI [2, 36], autonomous driving [1], 4D reconstruction [10, 37], and video editing [11, 18]. Despite its strong capability, existing point tracking methods have only been explored in single-view videos. Since the 2D projection of a 3D scene inherently incurs geometric ambiguities, such as frequent occlusion, erratic motion, and depth uncertainty, single-view point tracking methods [8, 14] struggle with these challenges. Consequently, a direct extension of these 2D trackers independently to each viewpoint—without any information exchange across views—fails to leverage the multi-view cues

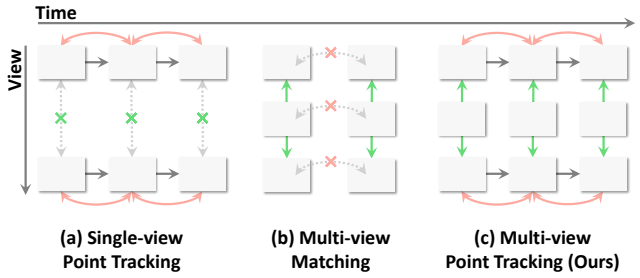

Figure 2: Motivation. We conceptually contrast our (c) multi-view point tracking with (a) single-view point tracking [6, 9, 20, 21, 39, 40, 42] and (b) multi-view matching [7, 27, 30, 31, 34]. Our approach simultaneously models both view- and frame-wise interactions to ensure cross-view and temporal consistency.

necessary for constructing reliable multi-view point trajectories.

Motivated by these observations, we formulate **multi-view point tracking** directly in the 2D pixel space, aiming to track a set of query points throughout multiple videos of a dynamic scene. Our key intuition is that multi-view observations of a dynamic scene provide complementary cues that help resolve ambiguities inherent to monocular settings. For instance, a point that is occluded or motion-blurred in one view may be clearly visible in other views. By reasoning across these viewpoints simultaneously, such ambiguities can be resolved, enabling more robust and accurate point tracking.

While multi-view correspondence has been studied as a promising way [7, 27, 30, 31, 34] to capture geometric relationships, these methods are often ill-suited for dynamic point tracking. They typically focus on static scenes [32], assume rigid geometry, or require geometric priors [40] that are unavailable in casual, in-the-wild videos.

Although a prior approach [29] targets multi-view 3D point tracking in the 3D world coordinate system, it relies on *external depth* inputs which may introduce reprojection errors when reprojecting 3D points to the 2D pixel space. This leaves a critical gap in methodology: there is no established paradigm for leveraging only *multi-view videos* to track points in pixel space through dynamic scenes.

To address these challenges, we present **MV-TAP** (**T**racking **A**ny **P**oint in **M**ulti-view **V**ideos), which builds a holistic understanding of dynamic scenes in multi-view videos. Our framework leverages camera encoding and a cross-view attention mechanism to effectively aggregate information across all views and timesteps. Furthermore, to facilitate research in this direction, we construct a large-scale synthetic dataset specifically designed for training multi-view point-tracking models and propose an evaluation benchmark to assess generalization and robustness across diverse multi-scene settings. Experiments on multi-view benchmarks demonstrate that our approach significantly improves upon existing tracking methods, yielding more complete and accurate predictions. Our code and dataset will be made publicly available.

Our contributions are summarized as follows:

- We propose a **pixel-space formulation** of a multi-view point tracking for establishing robust spatio-temporal correspondences in multi-view dynamic videos.
- We propose MV-TAP, a framework that leverages camera geometry and cross-view information to address inherent limitations of single-view point tracking, such as occlusion and motion ambiguity.
- We demonstrate through extensive experiments that our method achieves competitive performance, providing an effective baseline for this new task.

2. Related Work

Point tracking. The goal of point tracking is to predict the trajectories and visibilities of given query points within a video. Recent methods [6, 9, 14, 15, 20, 21, 23, 24, 39, 42] focus on estimating tracks by finding correspondences across frames while capturing temporal context. Specifically, PIPs [14] proposes iterative refinement on initial query points using local correlation volumes. TAPIR [9] combines this idea with a global correlation approach, which was introduced by TAP-Net [24]. LocoTrack [6] further introduces bidirectional local 4D correlation volumes for enhanced local correspondence. Notably, CoTracker [20, 21] learns the correlation between tracking points by interleaving frame- and track-wise attention modules. This approach achieves superior performance by explicitly learning the correlations between tracking points as well as temporal consistency. Unlike previous approaches, SpatialTracker [39] performs 3D point tracking by combining 2D tracker with depth estimation. TAPNext [42] reframes the point tracking problem as an autoregressive next-token prediction, enabling causal online tracking without specialized inductive biases. Despite these advances, existing methods are primarily designed for monocular videos, and their application to multi-view scenarios remains underexplored. The most relevant work to ours is MVTracker [29], which tracks query points across multi-view synchronized videos in 3D space. However, this approach highly relies on the quality of precomputed depth for lifting to 3D space. Unlike MVTracker, MV-TAP performs directly in the 2D pixel space with a view-wise attention module, showing improved robustness.

Multi-view point matching. Multi-view point matching aims to identify correspondences for a set of query points across images captured from multiple viewpoints. Conventional methods perform pairwise matching using handcrafted local features such as SIFT [27] and ORB [30]. Recent works employ learning-based approaches, such as self-supervised detector–descriptor [7], graph-based matchers [31], and detector-free Transformer matchers [34].

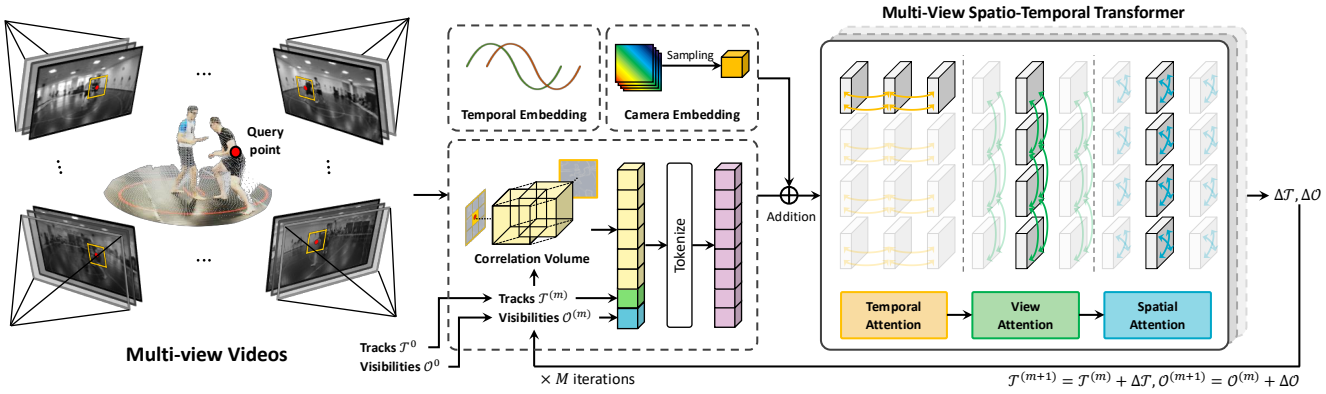

Figure 3: Overall architecture of MV-TAP. Given synchronized multi-view videos, per-view correlation volumes are extracted from a CNN encoder feature for each query point. These correlations are then tokenized and added with camera embedding for relative geometric context across views and temporal embedding. Trajectories and occlusion states are iteratively updated by a Transformer architecture, comprising temporal, spatial, and view attention modules.

These methods can be combined with single-view trackers to extend to the multi-view setting by matching tracks across views from a reference view. However, this naive combination ignores temporal consistency within each view and is inefficient due to redundant per-frame matching.

3. Method

3.1. Motivation and overview

Existing point tracking methods have primarily focused on achieving spatio-temporal consistency in single-view videos [6, 9, 20, 21, 24, 40, 42]. While these methods excel at predicting temporally consistent trajectories, they are not designed to exploit the geometric constraints and complementary information available in a multi-view system. Although tracking points in 3D world space [40] can be an alternative for utilizing multi-view geometry, this approach is highly dependent on the quality of depth estimation for lifting to 3D world space, which often compromises robustness. Motivated by this, we introduce multi-view point tracking in 2D camera space. Our goal is to leverage multi-view information to enhance tracking performance while maintaining the strong spatio-temporal consistency established by 2D trackers.

To achieve this, we present **MV-TAP** (**T**racking **A**ny **P**oint in **M**ulti-view **V**ideos), a model designed to effectively integrate information across multiple viewpoints. Our approach integrates a strong 2D tracking backbone [20] with additional modules designed to leverage multi-view information. Specifically, we introduce a camera encoding module to inject geometric information and a cross view-attention module to aggregate complementary cues across viewpoints. This combination allows our model to achieve robust spatio-temporal consistency across multiple views.

First, we formulate the multi-view point tracking in pixel space (Sec. 3.2). We then describe the construction of our camera-aware representation (Sec. 3.4), followed by multi-view spatio-temporal transformer (Sec. 3.5), which effectively exploits multi-view cues through view-wise attention. Finally, we explain our training strategy (Sec. 3.7). An overview of our architecture is shown in Fig. 3.

3.2. Problem definition

The inputs of multi-view point tracking are multi-view frames $\mathcal{I} = \{I_{v,t} \in \mathbb{R}^{H\times W\times 3}\}$, query points $\mathcal{Q} = \{q_{v,n} \in \mathbb{R}^3\}$, and camera parameters $\mathcal{G} = \{G_{v,t} = K[R_{v,t}|t_{v,t}]\}$ where $K \in \mathbb{R}^{3\times 3}$ indicates the intrinsic matrix shared across views, $R_{v,t} \in SO(3)$ and $t_{v,t} \in \mathbb{R}^3$ denotes rotation and translation parameters. Here, $v = 1, \ldots, V$ denotes the index of camera views, $t = 1, \ldots, T$ refers to the frame index, and $n = 1, \ldots, N$ represents the index of query points. We assume that videos from different views are temporally synchronized. A set of N query points is independently defined for each view, since the visible time step varies across views. Despite being defined independently, the query points across views represent the same scene points, where each query point on view v is represented by a 3-dimensional vector $q_v = (t_q, x_q, y_q)$. Specifically, t_q denotes the queried frame index, and (x_q, y_q) the spatial coordinates in that frame. The goal of this task is to predict a set of trajectories $\mathcal{T} \in \mathbb{R}^{V\times T\times N\times 2}$ and occlusion states $\mathcal{O} \in \mathbb{R}^{V\times T\times N\times 1}$ for the given queries, where $\mathcal{T}$ denotes the 2D pixel locations of N points over T time steps and V views, and $\mathcal{O}$ indicates whether each point is visible or occluded across views and time.

3.3. Local 4D correlation along the temporal axis

From initial or intermediate trajectories, we compute a cost volume as a matching representation. The intermediate trajectories are refined during this process. Following recent single-view point tracking approaches [6, 20], we adopt local 4D correlation to exploit rich appearance cues. For-

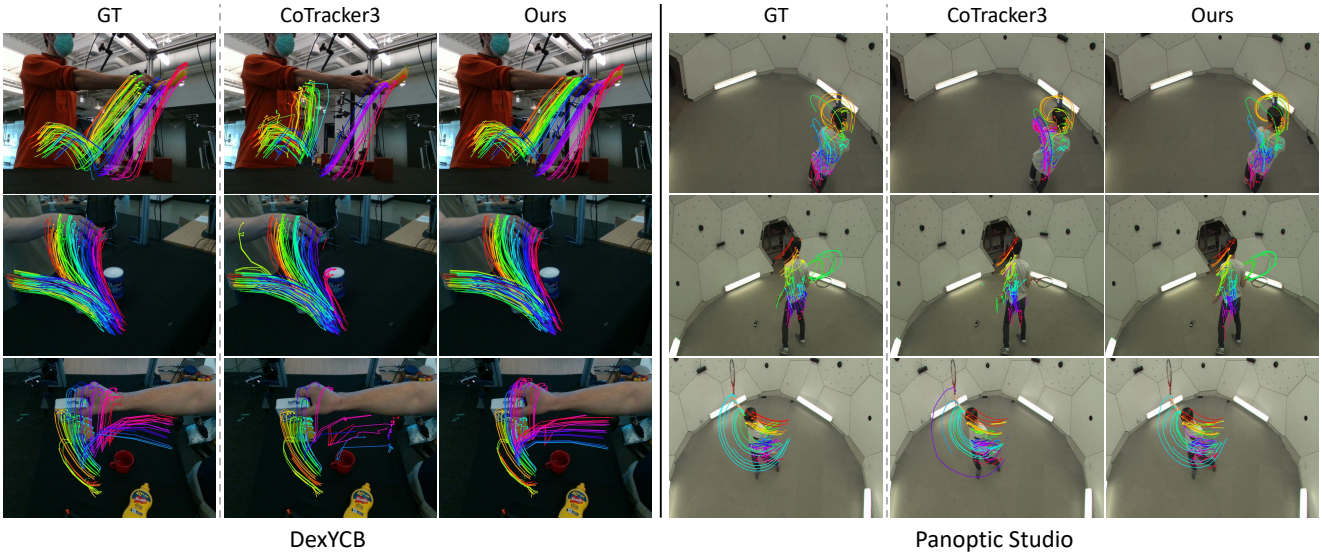

Figure 4: Qualitative comparison. We visualize results of MV-TAP and a single-view baseline [20] on the DexYCB [4] and Panoptic Studio [19] datasets. While the single-view baseline fails under occlusions and large motions resulting in highly fragmented tracks, MV-TAP demonstrates superior robustness, maintaining consistent trajectories in the challenging scenarios.

mally, given an initial or intermediate track hypothesis $\mathcal{T}^0$, we define the local correlation around a query point $q = (t_q, x_q, y_q)$ and its hypothesized match $p = (t_p, x_p, y_p)$. We construct the local neighborhoods around p and q within spatial radii r_p and r_q, respectively. The resulting local 4D correlation tensor is defined as:

$$\mathcal{L}_t(i, j; p, q) = \frac{F_t(i) \cdot F_{t_q}(j)}{\|F_t(i)\|_2 \, \|F_{t_q}(j)\|_2}, \tag{1}$$

where F_t and F_{t_q} denote the feature maps from frame t and the query frame t_q, respectively, and $i \in \mathcal{N}(p, r_p)$ and $j \in \mathcal{N}(q, r_q)$. The local 4D correlation tensor captures pairwise similarities between the two neighborhoods around p and q.

Note that the correlation volume could also be constructed across the view dimension. However, when the baseline between viewpoints is large, the appearance similarity between corresponding local patches degrades significantly. As a result, the correlations become unreliable and may introduce noise into the model. Therefore, we compute the correlations only along the temporal dimension.

At each timestep, the correlation $\mathcal{L}_t$ and the current hypothesis position are encoded into a token. Stacking tokens across frames and query points forms an input token $X \in \mathbb{R}^{V \times T \times N \times d}$.

3.4. View-aware camera encoding

In this section, we describe our view-aware camera parameter embedding. As camera parameters encode the 3D spatial information, we utilize them to provide the model with the relative geometric context of tracking points across views. Specifically, we leverage coordinates [28] to encode the ray corresponding to each tracking point. Each ray $r_{v,t,n} \in \mathbb{R}^6$ is defined by coordinates:

$$r = \begin{bmatrix} \mathbf{d} \\ \mathbf{m} \end{bmatrix}, \qquad \text{where } \mathbf{m} = \mathbf{o} \times \mathbf{d}, \tag{2}$$

where $\mathbf{d} \in \mathbb{R}^3$ is direction of the ray and $\mathbf{m} \in \mathbb{R}^3$ denotes the moment vector computed as the cross product between ray direction $\mathbf{d}$ and ray origin $\mathbf{o} \in \mathbb{R}^3$. The ray direction $\mathbf{d}$ and ray origin $\mathbf{o}$ for a specific pixel are computed as follows:

$$\mathbf{d} = R^\top K^{-1} \mathbf{x}, \qquad \mathbf{o} = -R^\top t, \tag{3}$$

where $\mathbf{x} = (u, v, 1)^\top$ denotes the homogeneous pixel coordinate. The direction $\mathbf{d}$ is normalized to unit length to ensure scale-invariant representation. Based on this definition, we construct the coordinates $\mathcal{R} \in \mathbb{R}^{V \times T \times N \times 6}$ for all trajectories and project them to a feature dimension through an MLP layer. The resulting embedding is then added to the input tokens, along with sinusoidal positional encoding [35]. This camera encoding strategy provides the model with explicit awareness of multi-view camera geometry, allowing it to capture spatial correspondences across different views.

3.5. Multi-view spatio-temporal transformer

The encoded tokens are processed by a Transformer that interleaves *temporal attention*, *spatial attention*, and *view attention*. When applying attention across different axes, the feature dimension d is always preserved, while the other axes are flattened into the batch dimension to enable attention along the selected axis.

Temporal attention. Temporal attention aggregates information along the **time axis** T. Formally, given query, key,

Table 1: Quantitative comparison. We provide quantitative comparison of single and multi-view point trackers on DexYCB [4], Panoptic Studio [19], Kubric [13], and Harmony4D [22] datasets. 'Target' denotes the dimension of the predicted trajectory. 'Space' specifies the coordinate domain: 'Camera' and 'World' denotes pixel and world space, respectively. 'Depth' indicates whether depth input is required. Compared to baselines, MV-TAP achieves superior performance, demonstrating its ability to leverage multi-view information.

Method	Target	Space	Depth	DexYCB			Panoptic Studio			Kubric			Harmony4D		
				AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA
Single-view input															
TAPIR [9]	2D	Camera	✗	29.6	43.9	66.4	22.1	39.3	60.0	66.9	76.9	89.7	27.6	53.1	60.0
CoTracker2 [21]	2D	Camera	✗	37.5	**62.5**	69.4	33.3	59.1	64.4	80.5	91.6	93.7	37.2	71.9	55.7
LocoTrack [6]	2D	Camera	✗	38.7	55.8	74.1	34.9	56.1	67.5	82.5	90.1	93.8	40.8	72.0	64.1
CoTracker3 [20]	2D	Camera	✗	41.5	59.6	76.4	39.6	61.4	72.3	83.5	90.7	94.1	41.4	73.5	63.2
TAPNext [42]	2D	Camera	✗	39.6	57.7	71.9	36.2	60.1	66.8	84.6	93.1	94.3	**42.7**	74.6	**68.4**
SpatialTracker [39]	3D	Camera	✓	23.2	43.3	61.8	19.7	40.5	59.6	65.7	78.5	80.9	25.4	54.4	58.1
TAPIP3D [40]	3D	World	✓	29.1	43.4	66.1	**41.9**	62.2	**78.9**	**88.1**	**95.7**	93.7	5.0	25.6	46.9
Multi-view input															
CoTracker3 + Flat.	2D	Camera	✗	2.7	7.1	35.7	1.0	12.7	38.8	19.6	29.3	34.6	2.1	20.7	46.4
CoTracker3 + Tri.	2D	Camera	✗	39.2	57.1	76.4	37.9	59.5	72.3	70.2	82.6	94.3	39.2	70.4	63.2
MVTracker [29]	3D	World	✓	-	48.4	-	-	62.4	-	-	94.0	-	-	13.3	-
MV-TAP	2D	Camera	✗	**44.2**	61.9	**78.3**	40.3	**62.8**	73.1	87.8	94.0	**96.3**	42.6	**74.9**	65.8

and value projections $Q_T, K_T, V_T \in \mathbb{R}^{T\times d}$ for a fixed point,

$$\texttt{Attn}_{\text{temp}}(X) = \texttt{Softmax}\left(\frac{Q_T K_T^\top}{\sqrt{d}}\right) V_T, \tag{4}$$

which integrates evidence across the frame sequence, ensuring temporally smooth trajectory updates.

Spatial attention. Spatial attention aggregates information along the **point axis** N within a single frame. Formally, given projections $Q_N, K_N, V_N \in \mathbb{R}^{N\times d}$ for a fixed timestep,

$$\texttt{Attn}_{\text{spatial}}(X) = \texttt{Softmax}\left(\frac{Q_N K_N^\top}{\sqrt{d}}\right) V_N, \tag{5}$$

which helps capture rigidity priors by linking points with consistent motion patterns.

View attention. While temporal and spatial attention capture intra-view relationships such as temporal smoothness and local rigidity, they are inherently limited in modeling inter-view relationships. To explicitly align representations across different views, we apply attention along the **view axis** V. Here, $Q_V, K_V, V_V \in \mathbb{R}^{V\times d}$,

$$\texttt{Attn}_{\text{view}}(X) = \texttt{Softmax}\left(\frac{Q_V K_V^\top}{\sqrt{d}}\right) V_V. \tag{6}$$

This attention module allows the model to exchange information between different viewpoints, thereby overcoming view-dependent ambiguities.

3.6. Recurrent trajectory and occlusion updates

With temporal, spatial, and view attention, the Transformer iteratively refines point trajectories and occlusion probabilities. Concretely, at each refinement step, the Transformer predicts incremental updates to both the track position and occlusion state:

$$\Delta\mathcal{T},\ \Delta\mathcal{O} = \texttt{Transformer}(X). \tag{7}$$

These updates are applied to the previous estimates as

$$\mathcal{T}^{(m+1)} = \mathcal{T}^{(m)} + \Delta\mathcal{T}, \quad \mathcal{O}^{(m+1)} = \mathcal{O}^{(m)} + \Delta\mathcal{O}, \tag{8}$$

so that after M refinement steps, the model produces the final trajectory $\mathcal{T}$ and occlusion status $\mathcal{O}$.

3.7. Training loss

To train MV-TAP, we optimize both trajectory regression and occlusion prediction. For trajectory supervision, we use the Huber loss [16]:

$$\mathcal{L}_{\text{track}}(\mathcal{T}, \mathcal{T}^*) = \sum_{m=1}^{M} \gamma^{M-m}\, \ell_{\text{Huber}}\left(\mathcal{T}^{(m)}, \mathcal{T}^*\right), \tag{9}$$

where $\mathcal{T}^{(m)}$ is the predicted trajectory at refinement step m, $\mathcal{T}^*$ is the ground-truth trajectory.

For occlusion supervision, we use a Binary Cross-Entropy (BCE) loss. We apply a sigmoid activation to the occlusion logits $\mathcal{O}^{(m)}$ before evaluating the loss:

$$\mathcal{L}_{\text{occ}}(\mathcal{O}, \mathcal{O}^*) = \sum_{m=1}^{M} \gamma^{M-m}\, \texttt{BCE}\left(\sigma(\mathcal{O}^{(m)}),\ \mathcal{O}^*\right), \tag{10}$$

where $\mathcal{O}^{(m)}$ is the predicted occlusion state at refinement step m and $\mathcal{O}^*$ is the ground truth.

4. Experiments

4.1. Experimental setup

Training. Since existing training datasets for point tracking are available only for single-view scenarios, we generate a synthetic dataset for multi-view point tracking by

Table 2: Ablation on various number of views. We compare MV-TAP with baselines using a varying number of input views. While performance generally improves with more views, the baselines exhibit only marginal gains. In contrast, MV-TAP demonstrates a significantly larger improvement consistently, highlighting its superior ability to leverage multi-view information.

Method	2 views			4 views			6 views			8 views		
	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA
CoTracker3	37.5	56.4	**77.8**	38.9	56.6	75.1	41.0	58.9	75.8	41.5	59.6	76.4
CoTracker3 + Flat.	9.5	19.5	54.3	4.4	8.4	44.5	3.2	8.3	39.1	2.7	7.1	35.7
CoTracker3 + Tri.	37.1	55.6	**77.8**	37.8	55.3	75.1	38.7	56.4	75.8	39.2	57.1	76.4
MVTracker	-	50.8	-	-	50.7	-	-	50.8	-	-	48.4	-
MV-TAP	**39.2**	**56.8**	76.8	**40.3**	**57.7**	**75.2**	**43.3**	**60.7**	**76.9**	**44.2**	**61.9**	**78.3**

Table 3: Can multi-view resolve occlusion ambiguity? We also evaluate the position accuracy on in-frame occluded points. Our model shows robustness on the occlusion, indicating that our model effectively utilizes the multi-view cues. $< \delta^x_{occ}$ denotes point accuracy on in-frame occlusion points.

Method	DexYCB		Panoptic Studio		Harmony4D	
	$< \delta^x_{avg}$	$< \delta^x_{occ}$	$< \delta^x_{avg}$	$< \delta^x_{occ}$	$< \delta^x_{avg}$	$< \delta^x_{occ}$
CoTracker3	59.6	33.9	61.4	46.2	73.5	58.4
CoTracker3 + Flat.	7.1	1.9	12.7	6.3	20.7	15.0
CoTracker3 + Tri.	57.1	34.8	59.5	47.7	70.4	59.1
MVTracker	48.4	34.2	62.4	**61.2**	13.3	8.9
MV-TAP	**61.9**	**38.4**	**62.8**	48.7	**74.9**	**60.3**

Table 4: Effect of additional training. Although initialized from the same pretrained model, MV-TAP attains consistently higher performance across all metrics. This shows that its gains primarily come from the architectural design, not merely extended training.

Method	DexYCB			Panoptic		
	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA
CoTracker3	41.8	59.0	73.8	39.6	61.6	71.8
MV-TAP	**44.2**	**61.9**	**78.3**	**40.3**	**62.8**	**73.1**

leveraging Kubric generation engine [13]. Our generated dataset consists of synchronized multi-view videos of 5,000 scenes, along with annotations that include point trajectories, their corresponding occlusion states, and camera parameters, both intrinsic and extrinsic. We provide further details in Supplementary material. Our model is trained on the generated multi-view dataset for 50K steps on 4 NVIDIA A6000 GPUs with a batch size of 1 per GPU. We employ AdamW optimizer [26] with a learning rate of 10^{-4} and a weight decay of 10^{-4}. We utilize a cosine learning rate scheduler with a 1,000 step warm-up stage and apply gradient clipping with a threshold of 1.0 for stable convergence. As MV-TAP is based on CoTracker3 [20], we adopt the pretrained weights to initialize the feature encoder and transformer layers. During training, we freeze only the feature extraction network, while all other parameters are updated. The number of input views is randomly selected between 1 and 4. The input resolution is 384×512, and the number of trajectories is 384. We set the number of refinement iterations to $M = 4$, and the spatial radii for local 4D correlation to $r_p = r_q = 3$.

Evaluation protocol. We evaluate our method and baselines on the DexYCB dataset [4], the Panoptic Studio dataset [19], the Kubric dataset [13], and Harmony4D dataset [22]. For DexYCB and Panoptic Studio, we utilize the point tracking annotations provided by [24] and [29], respectively, while for Kubric and Harmony4D, we construct point tracking annotations using the generation engine and human-mesh-recovery based annotation pipeline [23]. In the case of DexYCB, we sample dynamic points on the hand and the interacted object to focus on dynamic point tracking evaluation. To ensure consistent evaluation, we conduct our main experiments under an 8-view setup for all datasets, where we use all 8 available views in DexYCB and Kubric, and sample 8 views from Panoptic Studio and Harmony4D. For view sampling, we design three sampling strategies *nearest, random, and farthest* which are defined based on the distance between cameras. In the main paper, we adopt the *farthest* sampling strategy for our main results, while the results of other sampling strategies are provided in Supplementary Material.

We use the standard point tracking metrics from TAP-Vid [8], including position accuracy ($< \delta^x_{avg}$), occlusion accuracy (OA), and Average Jaccard (AJ). $< \delta^x_{avg}$ represents the average Percentage of Correct Keypoints (PCK) to evaluate the accuracy of the predicted keypoint position. Concretely, it is computed by averaging PCK over error thresholds of 1, 2, 4, 8, and 16 pixels for visible points in ground-truth. OA denotes the accuracy of the binary prediction for occlusion. AJ is a composite score that jointly evaluates position and occlusion prediction of each point.

Baselines. We compare our method against recent state-of-the-art point tracking methods, including single-view and multi-view approaches, covering both 2D and 3D formulations. We employ TAPIR [9], CoTracker2 [21], LocoTrack [6], CoTracker3 [20], TAPNext [42], SpatialTracker [39], and TAPIP3D [40] for single-view baselines and MVTracker [29] for multi-view baselines. In particular, SpatialTracker, TAPIP3D and MVTracker require a depth map as input, we provide ground-truth depth maps when available. Otherwise, we utilize an off-the-shelf depth estimator [38]. Moreover, MVTracker targets 3D tracking in world space and predicts aggregated visibility across all viewpoints and thus cannot provide per-view visibility.

Table 5: Comparison on various number of points. We measure tracking performances under varying numbers of query points. Our model consistently outperforms shows better robustness compared to the baselines across both sparse and dense settings.

Method	50 Points			100 Points			300 Points			500 Points		
	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA
CoTracker3	42.0	59.9	74.6	41.9	60.1	77.2	41.5	59.6	76.4	41.5	59.8	76.7
CoTracker3 + Flat.	2.7	7.4	34.4	2.5	6.8	35.6	2.7	7.1	35.7	2.6	7.0	35.7
CoTracker3 + Tri.	39.4	53.4	74.6	39.4	57.1	77.2	39.2	57.1	76.4	39.5	57.3	76.7
MVTracker	-	43.6	-	-	47.8	-	-	48.4	-	-	51.6	-
MV-TAP	**44.3**	**62.0**	**77.5**	**44.7**	**62.5**	**78.3**	**44.2**	**61.9**	**78.3**	**44.3**	**62.1**	**78.7**

Table 6: Ablation on model architecture. We present an ablation study on our model components for multi-view awareness. The performance consistently improves as each component is added, demonstrating that each module significantly contributes to leveraging multi-view information.

Method	DexYCB			Panoptic Studio		
	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA
CoTracker3	41.5	59.6	76.4	39.6	61.4	72.3
+ View attn.	43.6	61.5	77.4	38.6	61.6	69.4
+ Cam embed.	42.2	60.6	78.0	39.9	60.9	73.0
MV-TAP	**44.2**	**61.9**	**78.3**	**40.3**	**62.8**	**73.1**

Table 7: Comparison under frequently occluded trajectories. We evaluate methods on trajectories with high occlusion frequency measured by visibility-transition rate. MV-TAP leverages cross-view cues to remain robust on frequently occluded points, improving AJ, δ^x_{avg} and OA.

Method	DexYCB			Panoptic Studio		
	AJ	$< \delta^x_{avg}$	OA	AJ	$< \delta^x_{avg}$	OA
CoTracker3	26.2	43.4	66.6	37.4	60.6	69.0
CoTracker3 + Flat.	0.5	1.8	41.2	0.8	13.4	40.6
CoTracker3 + Tri.	26.0	43.6	66.6	36.6	59.4	69.0
MVTracker	-	21.4	-	-	59.4	-
MV-TAP	**29.7**	**47.3**	**70.5**	**38.0**	**61.9**	**69.9**

Therefore we report only $< \delta^x_{avg}$ for MVTracker by projecting its world-space trajectories into each camera's pixel space. For the single-view tracking baselines, which are designed for a monocular video setting, we perform tracking independently on each view and then aggregate the per-view results to compute the final metrics for fair comparison with multi-view methods.

Additionally, we include a naïve multi-view extension of the single-view tracker. The variant flattens the view and temporal dimensions, allowing the tracker to process multi-view videos as a single sequence. We implement this extension with CoTracker3 as an additional multi-view baseline (**CoTracker3 + Flat.**). Furthermore, to examine whether simple geometric cues can resolve the monocular ambiguities, we utilize multi-view geometry with provided camera parameters. We consider two approaches, optimization with epipolar constraints or triangulation. However, the epipolar-based approach requires a reference point to define the epipolar line. Since the reference point is already noisy due to prediction inaccuracies, it leads to significant error propagation. Consequently, we exclude this for our experiments. Instead, we adopt triangulation-based optimization. For each timestep, we compute a 3D point via triangulation from the 2D outputs of the single-view tracker, and then reproject it into each view to refine 2D trajectories. The triangulation-based approach is also built upon CoTracker3 as an additional multi-view baseline (**CoTracker3 + Tri.**).

4.2. Main Results

Quantitative results. We compare our approaches with recent state-of-the art point trackers on DexYCB [4], Panoptic Studio [19], Kubric [13] and Harmony4D [22]. As shown in Table 1, MV-TAP achieves consistently strong performance across all benchmarks, achieving noticeable performance. While TAPIP3D [40] and TAPNext [42] slightly outperform our methods on a few metrics, TAPIP3D relies on ground-truth depth and TAPNext adopts a substantially heavier architecture based on SSM and ViT blocks, but both methods underperform on specific benchmarks. Given that, performance of MV-TAP indicates strong multi-view point tracking capability.

Qualitative results. We present qualitative comparison in Figure 4 and additional qualitative results in Figure 5. We visualize the results from the DexYCB, Panoptic Studio, and Harmony4D. MV-TAP shows superior robustness to large and non-rigid motions, demonstrating the effectiveness of multi-view information for point tracking.

4.3. Ablation and Analysis

Ablation on various number of views. Table 2 presents a comparison of MV-TAP against a selected subset of baselines on the DexYCB[4] dataset, evaluating performance across various numbers of views. Although MV-TAP is trained only with 1 to 4 views due to resource limitations, owing to the attention mechanism, it can handle an arbitrary number of views even larger than 4. MV-TAP consistently outperforms the baselines across various view settings and the performance of metrics steadily improves as the number of views increases. This indicates that additional viewpoints provide richer spatial information for multi-view tracking.

Can multi-view information resolve occlusion ambiguity? To investigate whether multi-view information helps resolve occlusion ambiguity, we evaluate $< \delta^x_{\text{avg}}$ on particular occlusion points. In conventional point tracking evalua-

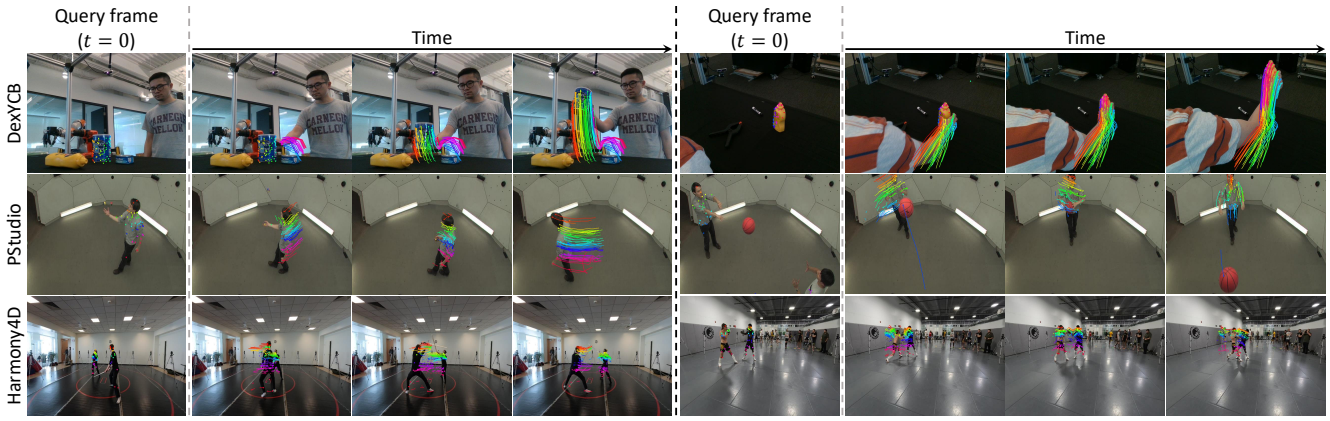

Figure 5: Visualization of point trajectories obtained by MV-TAP across diverse datasets. We showcase predictions of our model on the DexYCB [4], Panoptic Studio [19], and Harmony4D [22] datasets.

tion pipelines, the $< \delta^x_{\text{avg}}$ is measured only at visible points due to the limitation of the human annotated dataset [8], where occluded points are difficult to annotate. However, automatically annotated datasets [4, 13, 19, 22] provide accurate coordinates even for occluded points that remain in the frame. Therefore, we categorize occlusion into *in-frame occlusion* and *out-of-frame occlusion*, and evaluate point accuracy on in-frame occluded points. As shown in Table 3, incorporating multi-view information helps with occlusion handling. The triangulation-based optimization method shows strong robustness on in-frame occlusion points, as explicitly leveraging camera geometry helps to resolve occlusion ambiguity. However, its performance degrades on visible points due to its dependency on single-view tracker estimations. In contrast, MV-TAP maintains competitive stability by utilizing multi-view information, even without explicit geometric optimization.

Effect of additional training. While MV-TAP is initialized from the pretrained CoTracker3 [20], we further analyze the effect of additional training. In Table 4, MV-TAP shows clear performance improvements over baseline. This result suggests that the performance gain of MV-TAP is not simply due to additional training, but rather benefits from its well-designed architecture which effectively leverages multi-view information to enhance generalization and robustness.

Comparison on various numbers of query points. Downstream applications [1, 2, 10, 11, 18, 36, 37] vary in their requirements, ranging from sparse to dense correspondences estimation depending on the task. To account for this diversity, we evaluate the tracking performance under various number of query points. As shown in Table 5, MV-TAP overall achieves higher results than the baselines, highlighting its general applicability.

Ablation on model components. In Table 6, we present an ablation study on MV-TAP architecture. We examine the effect of applying camera embedding and view attention individually and in combination. These results show that both modules individually improve performance over the baseline. Notably, the combined version, which incorporates both view attention and camera embedding, achieves the best performance, demonstrating their complementary effects.

Comparison on frequent occlusion scene. In Table 7, we evaluate MV-TAP against baselines specifically on frequently occluded trajectories. For this evaluation, we quantify the occlusion frequency, which is defined as the number of visibility state transitions per trajectory. Then, we filter trajectories with high occlusion frequency by thresholding. The full quantification and thresholding details of the occlusion frequency statistics are provided in the Supplementary materials. As discussed earlier, single-view point tracking models show limited robustness in frequently occluded trajectories due to the absence of cross-view information. In contrast, MV-TAP maintains strong and consistent performance even under frequent occlusions, demonstrating the effectiveness of our well-designed approach in exploiting multi-view information for robust point tracking.

5. Conclusion

This work establishes multi-view 2D point tracking as a new and important task for advancing reliable spatio-temporal correspondence in dynamic, real-world scenes. By introducing MV-TAP, a model that aggregates cross-view information through camera embedding and view-attention, we demonstrate how leveraging multi-view inputs can overcome key limitations of monocular trackers such as occlusion and motion ambiguity. Together with a large-scale synthetic dataset and a real-world evaluation dataset specifically designed for this task, our contributions provide both a principled formulation of the problem and a strong baseline method, paving the way for future research in robust multi-view point tracking.

Acknowledgment

This research was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2019-II190075, RS-2024-00509279, RS-2025-II212068, RS-2023-00227592, RS-2025-02214479, RS-2024-00457882, RS-2025-25441838, RS-2025-25441838, RS-2025-02214479, RS-2025-02217259) and the Culture, Sports, and Tourism R&D Program through the Korea Creative Content Agency grant funded by the Ministry of Culture, Sports and Tourism (RS-2024-00345025, RS-2024-00333068, RS-2023-00222280, RS-2023-00266509), and National Research Foundation of Korea (RS-2024-00346597).

References

[1] Arjun Balasingam, Joseph Chandler, Chenning Li, Zhoutong Zhang, and Hari Balakrishnan. Drivetrack: A benchmark for long-range point tracking in real-world videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22488–22497, 2024. 1, 8

[2] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision*, pages 306–324. Springer, 2024. 1, 8

[3] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 1

[4] Yu-Wei Chao, Wei Yang, Yu Xiang, Pavlo Molchanov, Ankur Handa, Jonathan Tremblay, Yashraj S Narang, Karl Van Wyk, Umar Iqbal, Stan Birchfield, et al. Dexycb: A benchmark for capturing hand grasping of objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9044–9053, 2021. 4, 5, 6, 7, 8

[5] Anargyros Chatzitofis, Leonidas Saroglou, Prodromos Boutis, Petros Drakoulis, Nikolaos Zioulis, Shishir Subramanyam, Bart Kevelham, Caecilia Charbonnier, Pablo Cesar, Dimitrios Zarpalas, et al. Human4d: A human-centric multimodal dataset for motions and immersive media. *IEEE Access*, 8:176241–176262, 2020. 1

[6] Seokju Cho, Jiahui Huang, Jisu Nam, Honggyu An, Seungryong Kim, and Joon-Young Lee. Local all-pair correspondence for point tracking. In *European Conference on Computer Vision*, pages 306–325. Springer, 2024. 1, 2, 3, 5, 6

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

[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. 1, 6, 8

[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. 1, 2, 3, 5, 6

[10] Haiwen Feng, Junyi Zhang, Qianqian Wang, Yufei Ye, Pengcheng Yu, Michael J. Black, Trevor Darrell, and Angjoo Kanazawa. St4rtrack: Simultaneous 4d reconstruction and tracking in the world. *arXiv preprint arxiv:2504.13152*, 2025. 1, 8

[11] Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Yusuf Aytar, Michael Rubinstein, Chen Sun, et al. Motion prompting: Controlling video generation with motion trajectories. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1–12, 2025. 1, 8

[12] Saeed Ghorbani, K Mahdaviani, Anne Thaler, K Kording, DJ Cook, G Blohm, and NF Troje. Movi: A large multipurpose motion and video dataset. arxiv 2020. *arXiv preprint arXiv:2003.01888*, 9, 2020. 1

[13] 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. 5, 6, 7, 8

[14] 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. 1, 2

[15] Adam W Harley, Yang You, Xinglong Sun, Yang Zheng, Nikhil Raghuraman, Yunqi Gu, Sheldon Liang, Wen-Hsuan Chu, Achal Dave, Pavel Tokmakov, et al. Alltracker: Efficient dense point tracking at high resolution. *arXiv preprint arXiv:2506.07310*, 2025. 2

[16] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics: Methodology and distribution*, pages 492–518. Springer, 1992. 5

[17] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020. 1

[18] Hyeonho Jeong, Chun-Hao P Huang, Jong Chul Ye, Niloy J Mitra, and Duygu Ceylan. Track4gen: Teaching video diffusion models to track points improves video generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 7276–7287, 2025. 1, 8

[19] Hanbyul Joo, Hao Liu, Lei Tan, Lin Gui, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic studio: A massively multiview system for social motion capture. In *Proceedings of the IEEE international conference on computer vision*, pages 3334–3342, 2015. 1, 4, 5, 6, 7, 8

[20] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. *arXiv preprint arXiv:2410.11831*, 2024. 1, 2, 3, 4, 5, 6, 8

[21] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *European Conference on Computer Vision*, pages 18–35. Springer, 2024. 1, 2, 3, 5, 6

[22] Rawal Khirodkar, Jyun-Ting Song, Jinkun Cao, Zhengyi Luo, and Kris Kitani. Harmony4d: A video dataset for in-the-wild close human interactions. *Advances in Neural Information Processing Systems*, 37:107270–107285, 2024. 1, 5, 6, 7, 8

[23] Inès Hyeonsu Kim, Seokju Cho, Jahyeok Koo, Junghyun Park, Jiahui Huang, Joon-Young Lee, and Seungryong Kim. Learning to track any points from human motion. *arXiv preprint arXiv:2507.06233*, 2025. 2, 6

[24] Skanda Koppula, Ignacio Rocco, Yi Yang, Joe Heyward, Joao Carreira, Andrew Zisserman, Gabriel Brostow, and Carl Doersch. Tapvid-3d: A benchmark for tracking any point in 3d. *Advances in Neural Information Processing Systems*, 37: 82149–82165, 2024. 2, 3, 6

[25] Fuhao Li, Huan Jin, Bin Gao, Liaoyuan Fan, Lihui Jiang, and Long Zeng. Nugrounding: A multi-view 3d visual grounding framework in autonomous driving. *arXiv preprint arXiv:2503.22436*, 2025. 1

[26] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[27] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 2

[28] Julius Plücker. *Analytisch-geometrische Entwicklungen*. GD Baedeker, 1828. 4

[29] Frano Rajič, Haofei Xu, Marko Mihajlovic, Siyuan Li, Irem Demir, Emircan Gündoğdu, Lei Ke, Sergey Prokudin, Marc Pollefeys, and Siyu Tang. Multi-view 3d point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 59–68, 2025. 2, 5, 6

[30] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011. 2

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

[32] 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

[33] Younggyo Seo, Junsu Kim, Stephen James, Kimin Lee, Jinwoo Shin, and Pieter Abbeel. Multi-view masked world models for visual robotic manipulation. In *International Conference on Machine Learning*, pages 30613–30632. PMLR, 2023. 1

[34] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8922–8931, 2021. 2

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, A Gomez, Ł Kaiser, and Illia Polosukhin. Attention is all you need in advances in neural information processing systems, 2017. *Search PubMed*, pages 5998–6008, 2017. 4

[36] Mel Vecerik, Carl Doersch, Yi Yang, Todor Davchev, Yusuf Aytar, Guangyao Zhou, Raia Hadsell, Lourdes Agapito, and Jon Scholz. Robotap: Tracking arbitrary points for few-shot visual imitation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5397–5403. IEEE, 2024. 1, 8

[37] Qianqian Wang, Vickie Ye, Hang Gao, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. *arXiv preprint arXiv:2407.13764*, 2024. 1, 8

[38] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20697–20709, 2024. 6

[39] 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*, pages 20406–20417, 2024. 1, 2, 5, 6

[40] Bowei Zhang, Lei Ke, Adam W Harley, and Katerina Fragkiadaki. Tapip3d: Tracking any point in persistent 3d geometry. *arXiv preprint arXiv:2504.14717*, 2025. 1, 2, 3, 5, 6, 7

[41] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 1

[42] Artem Zholus, Carl Doersch, Yi Yang, Skanda Koppula, Viorica Patraucean, Xu Owen He, Ignacio Rocco, Mehdi SM Sajjadi, Sarath Chandar, and Ross Goroshin. Tapnext: Tracking any point (tap) as next token prediction. *arXiv preprint arXiv:2504.05579*, 2025. 1, 2, 3, 5, 6, 7