

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

– Supplementary Materials –

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

KAIST AI

Table 1: Feature-based and depth-based initialization. We compare query initialization using feature-based and depth-based methods in terms of $< \delta_{avg}^x$.

Method	DexYCB	Panoptic Studio
<i>Feature-based initialization</i>		
ResNet	<u>7.2</u>	<u>30.2</u>
VGGT [14]	1.5	17.3
VGGT-DINO [18]	5.2	29.4
DINOv3 [13]	11.8	36.5
<i>Depth-based initialization</i>		
MapAnything [8]	<u>6.6</u>	<u>32.9</u>
DA3 [11]	41.6	47.7

Table 2: Robustness to query init.

Query Initialization	Method	DexYCB	
		$< \delta_{avg}^x$	$< \delta_{occ}^x$
GT Depth	MVTracker	48.4	34.2
	MV-TAP	55.8	36.1
DUST3R Depth	MVTracker	34.9	23.5
	MV-TAP	53.3	34.5

0. Overview

In this supplementary material, we first explore different strategies for constructing queries for multi-view point tracking in Sec. 1. We show an illustration of the camera encoding module in Sec. 2. Next, we present additional experiments in Sec. 3. We also provide detailed descriptions of the training and evaluation datasets in Sec. 4. We then show additional qualitative results in Sec. 5. Finally, we describe limitations and future work of MV-TAP in Sec. 6.

1. Query initialization

We assume that the queries are given for all views. However, this assumption is impractical for specific settings.

*Equal contribution.

Algorithm 1 Feature-based query initialization

Require: Multi-view videos V , query point $q = (x_q, y_q, t_q, v_q)$
Ensure: Cross-view correspondences $\{(x_v^*, y_v^*, t_v^*, v)\}_{v \neq v_q}$

```

1: Extract query feature  $\mathbf{f}_q \leftarrow \Phi(V_{v_q, t_q})(x_q, y_q)$ 
2: for each view  $v \neq v_q$  do
3:   for each frame  $t$  do
4:      $C_{v, t}(x, y) \leftarrow \langle \mathbf{f}_q, \mathbf{F}_{v, t}(x, y) \rangle$ 
5:   end for
6:    $t_v^* \leftarrow \arg \max_t \max_{x, y} C_{v, t}(x, y)$ 
7:    $(x_v^*, y_v^*) \leftarrow \arg \max_{x, y} C_{v, t_v^*}(x, y)$ 
8: end for
```

Algorithm 2 Depth-based query initialization

Require: Multi-view videos V , query point $q = (x_q, y_q, t_q, v_q)$
Ensure: Cross-view correspondences $\{(x_v^*, y_v^*, t_v^*, v)\}_{v \neq v_q}$

```

1: Extract depth  $\mathbf{d}_{v_q, t_q} \leftarrow \mathbf{D}(V_{v_q, t_q})(x_q, y_q)$ 
2: Unproject query point to 3D
3:  $\mathbf{x}_q \leftarrow [x_q, y_q, 1]^\top$ 
4:  $\mathbf{p}_q^{3D} \leftarrow R_{v_q, t_q}(\mathbf{d}_{v_q, t_q} K^{-1} \mathbf{x}_q - t_{v_q, t_q})$ 
5: Reproject to other views
6: for each view  $v \neq v_q$  do
7:    $\tilde{\mathbf{x}}_{v, t_q} \leftarrow K(R_{v, t_q} \mathbf{p}_q^{3D} + t_{v, t_q})$ 
8:    $\mathbf{x}_{v, t_q} \leftarrow [\tilde{x}_{v, t_q} / \tilde{z}_{v, t_q}, \tilde{y}_{v, t_q} / \tilde{z}_{v, t_q}]^\top$ 
9:    $(x_v^*, y_v^*, t_v^*) \leftarrow (\mathbf{x}_{v, t_q}, t_q)$ 
10: end for
```

Thus, we explore query initialization methods by assuming that a query is given for an arbitrary view and searching for its corresponding position in the other views. Unlike the other experiments, where queries are defined for all views as $q = (x_q, y_q, t_q)$, in this section the query is specified for a particular view. We therefore extend the definition to $q = (x_q, y_q, t_q, v_q)$ to explicitly indicate the view index.

We assume that the queries are given for all views. While determining corresponding query points across views from

Table 3: Threshold-wise PCK analysis on the DexYCB and Panoptic Studio datasets.

Method	DexYCB						Panoptic Studio					
	$< \delta_1^x$	$< \delta_2^x$	$< \delta_4^x$	$< \delta_8^x$	$< \delta_{16}^x$	$< \delta_{avg}^x$	$< \delta_1^x$	$< \delta_2^x$	$< \delta_4^x$	$< \delta_8^x$	$< \delta_{16}^x$	$< \delta_{avg}^x$
<i>Single-view input</i>												
TAPIR [3]	13.4	27.4	43.3	61.3	74.0	43.9	5.5	17.0	37.7	59.9	76.5	39.3
CoTracker2 [7]	30.8	44.1	63.2	82.2	92.0	62.5	21.9	40.8	62.4	79.9	90.5	59.1
LocoTrack [2]	28.5	39.7	55.6	72.0	83.0	55.8	21.4	38.4	57.5	75.3	88.1	56.1
CoTracker3 [6]	29.9	42.2	59.9	77.7	88.5	59.6	<u>23.2</u>	42.6	65.0	83.0	93.3	61.4
TAPNext [19]	<u>30.6</u>	41.9	58.2	73.7	84.0	57.7	23.5	42.5	64.0	80.8	89.7	60.1
SpatialTracker [16]	24.2	31.5	41.4	53.3	66.0	43.3	8.0	19.4	37.7	59.0	78.6	40.5
TAPIP3D [17]	25.2	36.2	46.0	52.9	56.7	43.4	21.2	43.3	<u>67.2</u>	85.2	94.2	62.2
<i>Multi-view input</i>												
CoTracker3 + Flat.	3.1	4.4	6.1	8.5	13.3	7.1	3.6	6.5	10.5	16.8	25.9	12.7
CoTracker3 + Tri.	23.7	40.6	58.2	75.9	87.0	57.1	19.7	41.5	63.7	81.2	91.5	59.5
MVTracker [12]	24.0	31.6	42.7	60.6	83.3	48.4	15.8	39.7	69.6	90.0	96.8	62.4
MV-TAP	30.2	<u>43.5</u>	<u>63.0</u>	<u>81.1</u>	<u>91.8</u>	<u>61.9</u>	<u>23.2</u>	<u>42.9</u>	66.3	<u>85.8</u>	<u>95.6</u>	62.8

Table 4: Threshold-wise PCK analysis on the Kubric and Harmony4D datasets.

Method	Kubric						Harmony4D					
	$< \delta_1^x$	$< \delta_2^x$	$< \delta_4^x$	$< \delta_8^x$	$< \delta_{16}^x$	$< \delta_{avg}^x$	$< \delta_1^x$	$< \delta_2^x$	$< \delta_4^x$	$< \delta_8^x$	$< \delta_{16}^x$	$< \delta_{avg}^x$
<i>Single-view input</i>												
TAPIR [3]	43.5	71.4	84.8	90.8	94.1	76.9	11.1	30.2	56.5	77.7	90.1	53.1
CoTracker2 [7]	82.1	88.8	93.4	96.2	97.7	91.6	34.7	59.8	78.9	89.9	96.3	71.9
LocoTrack [2]	80.2	87.2	91.7	94.6	96.7	90.1	35.2	59.7	79.0	90.0	96.1	72.0
CoTracker3 [6]	81.6	88.2	92.8	96.0	98.0	91.3	34.8	60.8	81.9	92.4	97.3	73.5
TAPNext [19]	85.8	91.2	94.5	96.4	97.5	93.1	36.9	63.0	<u>82.8</u>	<u>92.9</u>	<u>97.6</u>	<u>74.6</u>
SpatialTracker [16]	59.4	70.8	80.8	88.1	93.2	78.4	15.7	33.8	57.2	76.0	89.1	54.4
TAPIP3D [17]	<u>85.5</u>	95.8	<u>98.4</u>	<u>99.4</u>	<u>99.7</u>	95.7	3.7	8.5	17.6	35.4	63.0	25.6
<i>Multi-view input</i>												
CoTracker3 + Flat.	21.2	25.4	29.0	32.9	38.2	29.3	4.9	10.3	18.5	28.3	41.3	20.7
CoTracker3 + Tri.	68.7	87.3	93.0	96.1	98.1	88.7	29.5	57.6	78.9	90.0	95.9	70.4
MVTracker [12]	72.8	<u>93.7</u>	99.0	99.8	99.9	93.0	1.6	3.7	8.0	17.1	30.8	12.2
MV-TAP	84.2	91.4	96.0	98.7	99.6	<u>94.0</u>	<u>36.6</u>	<u>62.8</u>	83.2	93.4	98.2	74.9

a single-view query is a critical challenge in practical multi-view point tracking. In our main experiments, we chose to focus specifically on the tracking component, treating initialization as a separate preprocessing step in order to isolate and evaluate tracking performance.

Feature-based initialization. First, we consider a feature matching approach, which is commonly used in conventional point tracking methods [2, 3] to obtain initial trajectories. We adopt a variety of architectures as baselines, including ResNet pretrained on CoTracker3 [6], VGGT [14], DINOv3 [13], and VGGT-DINO, which follows an SD-DINO [18] strategy to combine features from VGGT and DINOv3 as baselines for feature matching. Specifically, we extract VGGT features from its aggregator output.

Given a query point $q = (x_q, y_q, t_q, v_q)$, we extract its

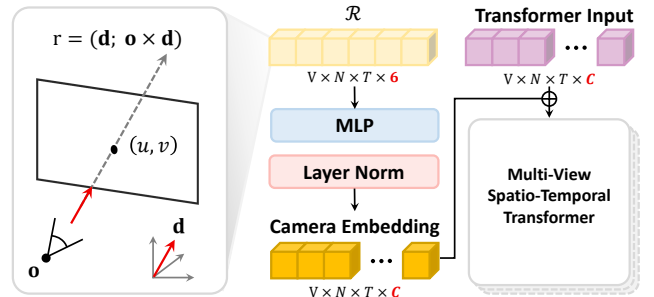


Figure 1: View-aware camera encoding. MLP-projected coordinates are added to transformer input tokens to inject view-wise information.

feature vector \mathbf{f}_q from the backbone feature map $\Phi(V_{v_q, t_q})$ at (x_q, y_q) . For each candidate view $v \neq v_q$ and frame t , we

Table 5: Triangulation-based refinement. ‘Tri’ denotes whether triangulation is applied. ‘Final’ applies triangulation to the final output, and ‘Window’ applies it after each temporal window. ‘RANSAC’ denotes whether outlier filtering is applied.

Method	Tri.	RANSAC	DexYCB		Panoptic Studio		Kubric		Harmony4D	
			$< \delta_{avg}^x$	$< \delta_{occ}^x$	$< \delta_{avg}^x$	$< \delta_{occ}^x$	$< \delta_{avg}^x$	$< \delta_{occ}^x$	$< \delta_{avg}^x$	$< \delta_{occ}^x$
CoTracker3	X	X	59.6	33.9	61.4	46.2	90.7	64.6	73.5	58.4
CoTracker3	Final	X	52.1	32.5	58.4	47.2	85.8	66.2	69.7	58.9
CoTracker3	Final	✓	57.1	<u>34.8</u>	59.5	47.7	88.7	<u>66.6</u>	70.4	59.1
CoTracker3	Window	X	53.8	33.9	59.5	<u>48.1</u>	86.2	65.8	71.2	<u>60.4</u>
CoTracker3	Window	✓	<u>58.5</u>	35.5	<u>61.2</u>	48.9	<u>88.9</u>	67.4	<u>72.2</u>	61.3
MV-TAP	X	X	61.9	38.4	62.8	48.7	94.0	70.4	74.9	60.3
MV-TAP	Final	X	60.6	39.7	61.2	50.6	90.9	<u>71.5</u>	72.7	61.8
MV-TAP	Final	✓	60.5	39.4	61.6	50.2	91.2	71.2	73.1	62.3
MV-TAP	Window	X	<u>61.8</u>	40.1	63.8	52.2	91.1	71.8	74.8	<u>64.4</u>
MV-TAP	Window	✓	60.9	39.2	<u>63.2</u>	<u>51.8</u>	<u>91.4</u>	71.3	75.1	64.7

Table 6: Occlusion Frequency. We evaluate MV-TAP and baselines on various type of frequent occlusion tracks. ‘Top-25’, ‘Top-50’, ‘Top-75’, and ‘Top-100’ denote the top 25%, 50%, 75%, and 100% most frequently occluded tracks, respectively.

Method	DexYCB			Panoptic Studio			Kubric			Harmony4D		
	AJ	$< \delta_{avg}^x$	OA	AJ	$< \delta_{avg}^x$	OA	AJ	$< \delta_{avg}^x$	OA	AJ	$< \delta_{avg}^x$	OA
<i>Top-25</i>												
CoTracker3	25.3	42.7	<u>65.9</u>	<u>37.3</u>	<u>60.5</u>	68.9	<u>65.7</u>	79.7	61.5	<u>33.4</u>	<u>64.9</u>	59.2
CoTracker3 + Flat.	0.5	1.8	42.3	0.7	13.2	40.3	11.9	23.1	38.2	1.7	17.4	49.9
CoTracker3 + Tri.	<u>25.7</u>	<u>43.1</u>	<u>65.9</u>	36.4	59.0	<u>69.1</u>	65.2	78.8	<u>85.7</u>	31.9	62.7	<u>59.2</u>
MVTracker	-	17.2	-	-	59.5	-	-	89.8	-	-	10.6	-
MV-TAP	28.7	46.3	69.7	37.5	61.3	69.9	74.4	<u>86.6</u>	90.9	34.6	67.1	61.7
<i>Top-50</i>												
CoTracker3	<u>29.3</u>	<u>47.4</u>	<u>68.6</u>	<u>38.3</u>	60.7	<u>70.9</u>	<u>71.2</u>	83.2	67.0	<u>36.1</u>	<u>67.7</u>	<u>60.8</u>
CoTracker3 + Flat.	0.8	2.8	40.3	0.7	12.5	39.7	0.7	12.5	39.7	1.9	18.0	48.4
CoTracker3 + Tri.	29.0	46.9	<u>68.6</u>	37.1	59.3	70.4	70.2	82.0	<u>88.4</u>	34.3	65.3	<u>60.8</u>
MVTracker	-	30.0	-	-	<u>60.9</u>	-	-	91.1	-	-	9.6	-
MV-TAP	32.8	50.9	72.5	38.9	62.0	71.8	79.6	<u>89.5</u>	93.3	36.8	69.4	63.0
<i>Top-75</i>												
CoTracker3	<u>34.2</u>	<u>52.8</u>	<u>72.3</u>	<u>38.7</u>	60.8	<u>71.6</u>	<u>73.7</u>	84.8	89.6	<u>39.1</u>	<u>71.0</u>	<u>62.3</u>
CoTracker3 + Flat.	1.4	4.4	38.6	0.8	13.1	39.3	14.6	25.9	35.6	2.0	20.3	46.9
CoTracker3 + Tri.	34.0	52.0	<u>72.3</u>	37.7	59.4	71.5	72.6	83.5	<u>89.6</u>	37.1	68.4	<u>62.3</u>
MVTracker	-	40.0	-	-	<u>61.4</u>	-	-	91.6	-	-	11.5	-
MV-TAP	37.5	55.7	74.9	39.6	62.3	72.6	81.3	<u>90.5</u>	94.1	40.4	72.6	65.1
<i>Top-100</i>												
CoTracker3	<u>41.5</u>	<u>59.6</u>	<u>76.4</u>	<u>39.6</u>	61.4	<u>72.3</u>	<u>83.5</u>	90.7	94.1	<u>41.4</u>	<u>73.5</u>	<u>63.2</u>
CoTracker3 + Flat.	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.	39.2	57.1	<u>76.4</u>	37.9	59.5	<u>72.3</u>	70.2	82.6	<u>94.3</u>	39.2	70.4	<u>63.2</u>
MVTracker	-	48.4	-	-	<u>62.4</u>	-	-	<u>93.0</u>	-	-	12.2	-
MV-TAP	44.2	61.9	78.3	40.3	62.8	73.1	87.8	94.0	96.3	42.6	74.9	65.8

compute a correlation map by dot product as:

$$C_{v,t}(x, y) = \langle \mathbf{f}_q, \mathbf{F}_{v,t}(x, y) \rangle, \quad (1)$$

where $\mathbf{F}_{v,t} = \Phi(V_{v,t})$. We then select the most relevant frame for each view by

$$t_v^* = \arg \max_t \max_{x,y} C_{v,t}(x, y), \quad (2)$$

and localize the 2D correspondence using argmax operation

as:

$$(x_v^*, y_v^*) = \arg \max_{x,y} C_{v,t_v^*}(x, y). \quad (3)$$

The algorithm is provided in Algorithm 1.

As shown in Tab. 1, DINOv3 and VGGT-DINO show relatively strong performances compared to the other baselines, but their performance remains inadequate for query initialization.

Depth-based initialization.

Table 7: Ablation on various number of views.

Method	2 views			4 views			6 views			8 views		
	AJ	$< \delta_{avg}^x$	OA	AJ	$< \delta_{avg}^x$	OA	AJ	$< \delta_{avg}^x$	OA	AJ	$< \delta_{avg}^x$	OA
<i>Panoptic Studio</i>												
CoTracker3	39.7	61.8	72.4	39.4	60.9	72.3	38.6	59.8	72.8	39.6	61.4	72.3
CoTracker3 + Flat.	9.9	21.4	56.6	2.4	11.4	48.8	2.4	11.1	39.7	1.0	12.7	38.8
CoTracker3 + Tri.	39.6	61.6	72.4	38.1	59.5	72.3	37.2	58.1	72.8	37.9	59.5	72.3
MVTracker	-	57.0	-	-	61.2	-	-	60.2	-	-	62.4	-
MV-TAP	39.6	61.7	72.3	39.8	61.6	72.5	39.0	60.7	73.1	40.3	62.8	73.1
<i>Harmony4D</i>												
CoTracker3	36.1	69.8	59.5	38.7	72.0	61.3	40.5	72.8	62.5	41.4	73.5	63.2
CoTracker3 + Flat.	14.5	34.1	51.1	4.3	20.5	49.8	3.2	18.2	47.5	2.1	20.7	46.4
CoTracker3 + Tri.	36.3	70.1	59.5	36.8	69.4	61.3	37.7	69.0	62.5	39.2	70.4	63.2
MVTracker	-	15.9	-	-	11.1	-	-	12.2	-	-	12.2	-
MV-TAP	36.7	70.1	64.3	39.4	72.2	65.0	41.9	74.2	65.6	42.6	74.9	65.8

Alternatively, we utilize off-the-shelf metric depth estimators, which take camera parameters as input for query initialization. Using the depth of the query point \mathbf{d}_{v_q, t_q} , we unproject the query coordinate to obtain its 3D location \mathbf{p}_q^{3D} using the given camera parameters:

$$\mathbf{p}_q^{3D} = R_{v_q, t_q}^\top (\mathbf{d}_{v_q, t_q} K^{-1} \mathbf{x}_q - t_{v_q, t_q}), \quad (4)$$

where $\mathbf{x}_q = [x_q, y_q, 1]^\top$ is the homogeneous coordinate of the query point, K denotes the intrinsic parameters, and R_{v_q, t_q} and t_{v_q, t_q} are the rotation and translation parameters. Then, we reproject the 3D coordinate of query point into the remaining views $v \neq v_q$ using their corresponding intrinsic and extrinsic parameters to obtain query initialization for multi-view point tracking.

$$\tilde{\mathbf{x}}_{v, t_q} = K (R_{v, t_q} \mathbf{p}_q^{3D} + t_{v, t_q}), \quad \text{for } v \neq v_q, \quad (5)$$

$$\mathbf{x}_{v, t_q} = [\tilde{x}_{v, t_q} / \tilde{z}_{v, t_q}, \tilde{y}_{v, t_q} / \tilde{z}_{v, t_q}]^\top, \quad (6)$$

where $\tilde{\mathbf{x}}_{v, t_q} = [\tilde{x}_{v, t_q}, \tilde{y}_{v, t_q}, \tilde{z}_{v, t_q}]^\top$ denotes the homogeneous representation of reprojected point. The algorithm is provided in Alg. 2.

We evaluate this strategy using MapAnything [8] and DepthAnything 3 (DA3) [11]. Tab. 1 presents the PCK of these baselines. While DA3 shows impressive performance, the initialized queries still contain noise and are insufficient for perfect multi-view initialization.

Robustness to initialization noise. Precise depth estimation is often unavailable in real-world scenarios. Therefore, practical trackers must be robust to initialization imperfections. Importantly, our 2D-based tracking design exhibits robustness to initialization noise, which helps mitigate practical concerns. To demonstrate this, we evaluate our method using the same initialization strategy as MVTracker: lifting 2D observations to 3D using depth and reprojecting them to all views. Beyond the ground-truth depth setting used in MVTracker, we also test with depth estimates from DUST3R

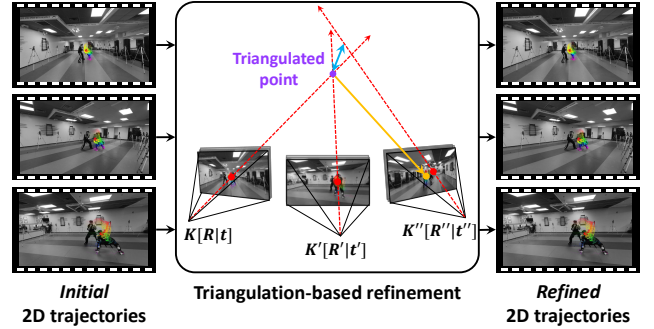


Figure 2: **Triangulation-based refinement.** For each point from initial 2D trajectories predicted by point trackers, at a given timestep, we perform triangulation with their respective camera parameters and obtain corresponding 3D point. This 3D point is then reprojected into each image planes to refine initial point trajectories, resolving monocular ambiguities using simple geometric cues.

to reflect realistic scenarios where perfect depth is unavailable. As shown in Tab. 2, our method maintains stability under both ground-truth and estimated depth initialization. This indicates that despite the inherent challenges of single-view query initialization, our method paired with off-the-shelf depth estimators remains highly applicable in practice.

2. Camera encoding

In Fig. 1, we illustrate the camera encoding introduced in Sec. 3.4 of the main paper.

3. Additional ablation and analysis

Threshold-wise PCK analysis. In Tabs. 3 and 4, we show a detailed comparison of PCK at thresholds of 1, 2, 4, 8, and 16 pixels. For this comparison, we use the same baselines and datasets as in the main comparison. Across all thresholds and datasets, MV-TAP demonstrates consistently competitive performance.

Table 8: Ablation on view sampling methods.

Method	Panoptic Studio			Harmony4D		
	AJ	$< \delta_{avg}^x$	OA	AJ	$< \delta_{avg}^x$	OA
<i>Nearest Sampling</i>						
CoTracker3	37.9	58.9	<u>73.8</u>	37.9	69.6	<u>61.8</u>
CoTracker3 + Flat.	1.9	17.2	43.2	10.7	31.9	51.5
CoTracker3 + Tri.	36.4	56.5	<u>73.8</u>	36.5	67.5	<u>61.8</u>
MVTracker	-	57.1	-	-	12.0	-
MV-TAP	39.8	60.3	75.9	40.4	71.3	66.5
<i>Random Sampling</i>						
CoTracker3	40.2	62.5	<u>72.7</u>	39.4	72.1	<u>62.0</u>
CoTracker3 + Flat.	1.5	10.5	49.3	6.5	20.7	50.5
CoTracker3 + Tri.	38.8	60.7	<u>72.7</u>	37.6	69.4	<u>62.0</u>
MVTracker	-	60.3	-	-	12.1	-
MV-TAP	40.5	62.8	73.2	40.6	73.0	65.3

Table 9: The design of MV-TAP can be applied to various point-tracking methods.

Method	DexYCB		
	AJ	$< \delta_{avg}^x$	OA
CoTracker2 [21]	37.5	62.5	69.4
CoTracker2 + MV-TAP	37.9	63.2	69.2
Improvements over baseline	+0.4	+0.7	-0.2
LocoTrack [6]	38.7	55.8	74.1
LocoTrack + MV-TAP	41.8	57.9	74.5
Improvements over baseline	+3.1	+2.1	+0.4
CoTracker3 [20]	41.5	59.6	76.4
CoTracker3 + MV-TAP	44.2	61.9	78.3
Improvements over baseline	+2.7	+2.3	+1.9

Table 10: Computation cost.

Method	300 points						8 views			
	2 view		4 view		8 view		50 points		100 points	
	FPS \uparrow	GB \downarrow	FPS \uparrow	GB \downarrow	FPS \uparrow	GB \downarrow	FPS \uparrow	GB \downarrow	FPS \uparrow	GB \downarrow
CoTracker3	78.6	0.2	41.0	0.4	21.2	0.8	<u>37.4</u>	0.8	32.8	0.8
TAPNext	10.9	2.1	7.7	3.0	4.6	4.7	5.9	4.7	5.9	4.7
TAPIP3D	11.3	<u>1.8</u>	5.1	<u>2.0</u>	2.5	<u>2.3</u>	6.5	<u>2.3</u>	5.1	<u>2.3</u>
MVTracker	63.8	2.3	<u>37.5</u>	4.5	21.2	10.1	23.3	10.1	22.7	10.1
MV-TAP	<u>67.0</u>	2.9	36.5	5.7	<u>19.7</u>	11.3	37.7	5.8	<u>28.8</u>	6.9

Triangulation-based refinement. In Tab. 5, we investigate the effect of triangulation-based optimization. We evaluate using CoTracker3 [6] and MV-TAP, examining how different optimization strategies influence performance. Specifically, we compare applying triangulation after each temporal window or after the entire sequence, along with the optional use of RANSAC. This process is illustrated in Fig. 2. We measure the performance using the average PCK of visible points and invisible in-frame points, as triangulation-based optimization directly refines the estimated 2D coordinates.

This experiment demonstrates that triangulation-based

optimization tends to improve PCK ($< \delta_{occ}^x$) of the in-frame invisible points. While both single-view and multi-view trackers struggle to accurately estimate the positions of in-frame invisible points, triangulation-based optimization exploits reliable visible correspondences to infer more accurate estimates in these invisible regions. However, this optimization assumes that the visible correspondences are accurate, which may lead to noisy predictions when the tracker outputs contain errors.

Comparison on frequent occlusion scene. We extend the frequent occlusion analysis in Tab. 7 of main paper to further verify the generality of our approach by additionally sampling trajectories with various occlusion frequencies. Tab. 6 demonstrates that MV-TAP exhibits strong robustness to frequently occluded points.

Analysis on various number of views. Tab. 7 presents further analysis for various numbers of views on the Panoptic Studio [5] and Harmony4D [9] datasets. Similar to the experiment for the DexYCB [1] dataset, MV-TAP achieves superior performance over baselines and shows steady improvements across all evaluation metrics.

Analysis on sampling strategies. As we mentioned earlier, the Panoptic Studio [5] and Harmony4D [9] datasets are sampled using the *farthest* strategy based on inter-camera distance. We additionally evaluate MV-TAP and the baselines under the *nearest*, and *random* sampling strategies to analyze the robustness with respect to view selection. As shown in Tab. 8, MV-TAP consistently outperforms baselines across the other sampling strategies.

Additional backbones. Tab. 9 demonstrates the architectural generalizability of MV-TAP. We apply our method to two additional monocular point tracking baselines, CoTracker2 and LocoTrack. These results indicate that our proposed modifications are simple and generalize well across various monocular tracking architectures. We emphasize this architectural generalizability and simplicity as a conceptual contribution of our work.

Computational cost. Tab. 10 shows the computational cost of MV-TAP against existing baselines. We measure the inference time (FPS) and peak memory of GPU VRAM. MV-TAP achieves competitive performance to baselines. Notably, despite utilizing multi-view information, MV-TAP achieves competitive speed and memory efficiency compared to the baselines.

4. Dataset details

4.1. Training dataset

As no large-scale training dataset exists for multi-view point tracking, we introduce a new synthetic dataset generated via Kubric [4]. Our dataset comprises 5,000 dynamic scenes with synchronized rendered videos from four distinct viewpoints. To facilitate effective cross-view learning, camera

positions are sampled in a chained manner to ensure sufficient overlap. Specifically, camera positions are sampled on hemispheres with radii ranging from 10 to 12, and adjacent viewpoints are spaced with an angular separation between 10° and 45° . To generate globally shared ground-truth tracks, we sample query points from each viewpoint and project them onto the remaining views.

4.2. Evaluation dataset detail

To evaluate tracking performance on complex real-world human motion, we constructed a benchmark using the Harmony4D [9] dataset, which provides synchronized multi-view videos and SMPL meshes. We obtained per-view ground-truth trajectories and visibility by projecting SMPL mesh vertices onto each image plane. To ensure the reliability of these trajectories, we employed the filtering method proposed in AnthroTAP [10]. Following the default configuration, we filtered out erratic trajectories by comparing them with optical flow predictions [15], retaining only high-confidence tracks.

5. Additional qualitative results

We provide additional qualitative results in Fig. 3, Fig. 4, and Fig. 5, on the DexYCB [1], Panoptic Studio [5], and Harmony4D [9] datasets, respectively.

6. Limitation and future work

While MV-TAP achieves strong performance for multi-view point tracking within 2D pixel space, it still has notable limitations. First, we assume that query points are provided in all views. However, this requirement is often impractical. As shown in our query initialization experiments, finding correspondences for a single-view query in other views remains unreliable for robust multi-view point tracking. Developing more advanced strategies for multi-view query initialization is therefore needed to broaden the applicability of MV-TAP.

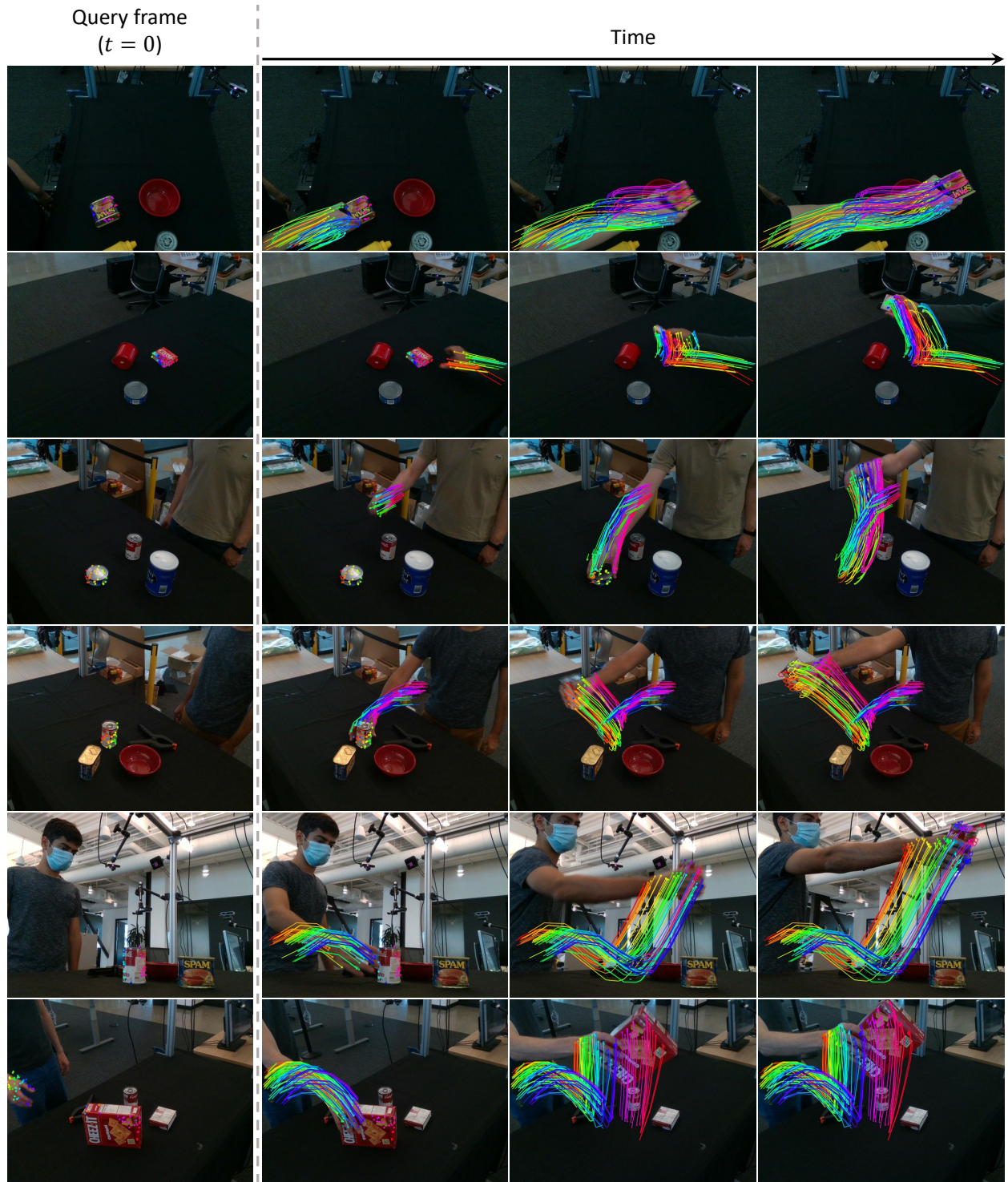


Figure 3: Additional qualitative results of our model. We provide further visualizations on the DexYCB [1] dataset.

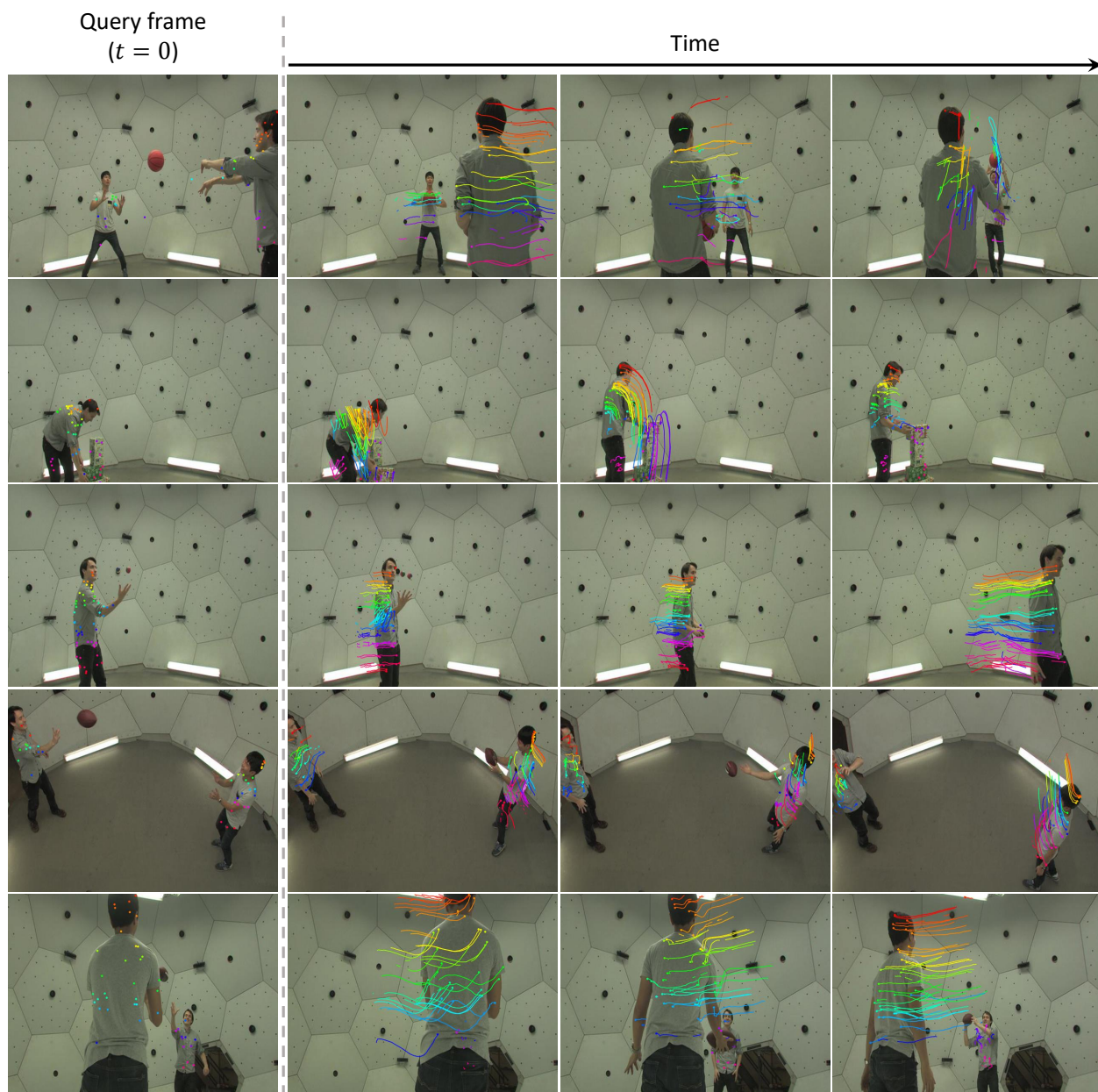


Figure 4: Additional qualitative results of our model. We provide further visualizations on the Panoptic Studio [5] dataset.

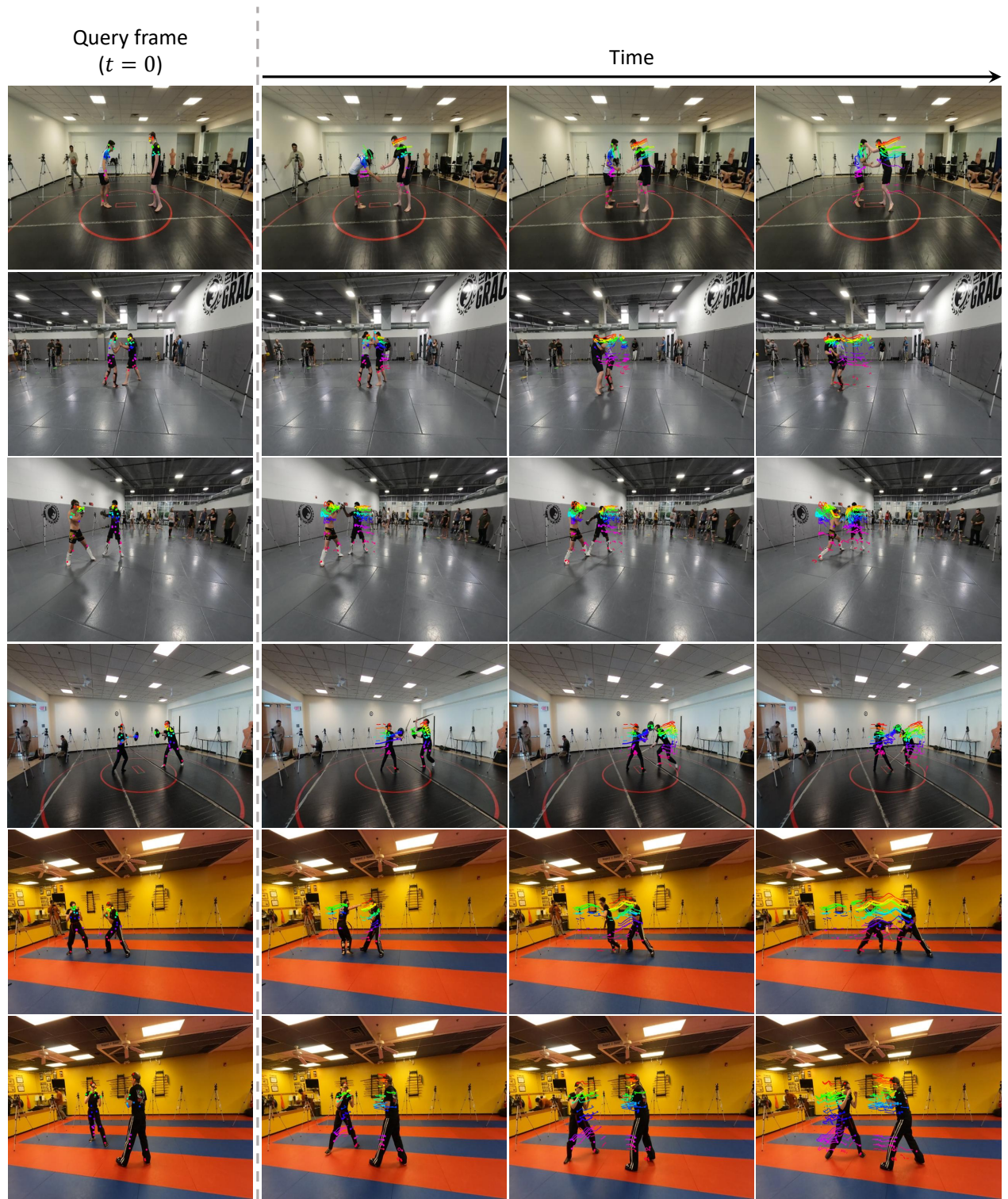


Figure 5: Additional qualitative results of our model. We provide further visualizations on the Harmony4D [9] dataset.

References

- [1] 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. 5, 6, 7
- [2] 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. 2
- [3] 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
- [4] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanaprasgam, 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
- [5] 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. 5, 6, 8
- [6] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Co-tracker3: Simpler and better point tracking by pseudo-labelling real videos. *arXiv preprint arXiv:2410.11831*, 2024. 2, 5
- [7] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Co-tracker: It is better to track together. In *European Conference on Computer Vision*, pages 18–35. Springer, 2024. 2
- [8] Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zaus, Ethan Weber, Nelson Antunes, et al. Mapanything: Universal feed-forward metric 3d reconstruction. *arXiv preprint arXiv:2509.13414*, 2025. 1, 4
- [9] 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. 5, 6, 9
- [10] 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. 6
- [11] Haotong Lin, Sili Chen, Junhao Liew, Donny Y Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025. 1, 4
- [12] 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
- [13] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025. 1, 2
- [14] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025. 1, 2
- [15] Yihan Wang, Lahav Lipson, and Jia Deng. Sea-raft: Simple, efficient, accurate raft for optical flow. In *European Conference on Computer Vision*, pages 36–54. Springer, 2024. 6
- [16] 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. 2
- [17] Bowei Zhang, Lei Ke, Adam W Harley, and Katerina Fragkiadaki. Tapip3d: Tracking any point in persistent 3d geometry. *arXiv preprint arXiv:2504.14717*, 2025. 2
- [18] Junyi Zhang, Charles Herrmann, Junhwa Hur, Luisa Polania Cabrera, Varun Jampani, Deqing Sun, and Ming-Hsuan Yang. A tale of two features: Stable diffusion complements dino for zero-shot semantic correspondence. *arXiv preprint arXiv:2305.15347*, 2023. 1, 2
- [19] Artem Zhoolus, 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. 2