Tracking Everything Everywhere across Multiple Cameras

Li-Heng Wang[1], **YuJu Cheng**[2], **Tyng-Luh Liu**[1]

[1]Institute of Information Science, Academia Sinica, Taiwan
[2]National Taiwan University
{lihengw, liutyng}@iis.sinica.edu.tw

Abstract

Pixel tracking in single-view video sequences has recently emerged as a significant area of research. While previous work has primarily concentrated on tracking within a given video, we propose to expand pixel correspondence estimation into multi-view scenarios. The central concept involves utilizing a canonical space that preserves a universal 3D representation across different views and timesteps. This model allows for precise tracking of points even through prolonged occlusions and significant deformations in appearance between views. Moreover, we show that our model, through the use of an efficient training strategy incorporating distillation loss, is capable of performing incremental pixel tracking, a process often seen as complex in test-time optimization techniques. Comprehensive experiments validate the method's ability to accurately establish point correspondences across cameras. Furthermore, our method achieves promising results of multi-view pixel tracking without requiring the entire video sequences to be provided at once.

1 Introduction

Pixel-level matching has posed a significant challenge in computer vision. Notable approaches for addressing this task include optical flow estimation (Dosovitskiy et al. 2015; Teed and Deng 2020; Shi et al. 2023a,b; Huang et al. 2022) and feature matching (Sarlin et al. 2020; Sun et al. 2021; Pautrat et al. 2023). However, optical flow supports accurate estimation primarily on subsequent frames and tends to fail in tracking through occlusions, while feature matching only identifies significant features from pairs of images instead of matching all points. To tackle both challenges concurrently, several studies have recently been dedicated to dealing with long-term pixel tracking. These methods not only track every single pixel through a video sequence but also provide accurate long-term motion estimation. However, these approaches have mainly concentrated on finding dense and long-term trajectories within a single video sequence.

An additional aspect of the pixel-matching problem involves finding dense correspondences between pairs of images captured from distinct viewpoints. Although several approaches, e.g., (Edstedt et al. 2023b; Ni et al. 2023; Edstedt et al. 2023a; Nam et al. 2023; Shen et al. 2020; Truong, Danelljan, and Timofte 2020) manage to perform dense pixel-level matching, they are not effective for matching all pixels and result in inconsistent motion estimation between frames. Furthermore, these techniques often lose track of pixels when they become occluded. A contributing factor to these issues is the independent processing of frames, neglecting the temporal relationships between them. The task of matching pixels across multi-view video frames, which requires consistent and reliable feature matching across different cameras and timesteps, has yet to be explored.

The proposed formulation concerns a more challenging scenario where we aim to track every single point across multiple video sequences captured by different cameras. Given any query pixel from an arbitrary timestep and view, the model should predict accurate correspondences in every other frame across cameras. This scenario not only involves predicting accurate positions and visibilities across long sequences, similar to single-view pixel tracking, but also requires integrating information across cameras. An advantage of the multi-camera setup is that it provides different perspectives of the same scene, reducing the likelihood of losing track of pixels due to occlusions, extreme angles, or poor lighting conditions. For instance, if a pixel is occluded in one camera, the information from another camera can compensate, leading to more accurate and robust tracking.

To achieve this, we propose a unified multi-view pixel tracking model capable of performing pixel-to-pixel matching across cameras and time frames. The key insight of our method is leveraging a global canonical space, which models a universal 3D representation of the entire scene, invariant to view and time changes. To obtain correspondences across different views and timesteps, the model learns a bijective mapping to deform points from different timesteps and views to the canonical space, and vice versa. A per-view latent code is learned along with the timestep to condition the bijective mapping model, allowing it to capture observational differences without needing camera poses. In addition, we add a spatial loss to encourage rigidity among neighboring pixels, addressing the issue of some pixels lagging behind fast-moving objects.

Our approach also considers the situation where full multi-view video sequences are not available in training, necessitating test-time training of the model. A caveat of the test-time optimization frameworks (e.g., NeRF) is their limi-

Figure 1: Our proposed multi-view pixel-level tracking method can identify pixel correspondences across image frames captured at different time instances and from multiple cameras.

tation in adapting to new data. Simply fine-tuning the trained model with new data could suffer from catastrophic forgetting. In our task, these phenomena cause the model to be incapable of reconstructing the global 3D scene, resulting in tracking failure. We introduce an efficient incremental learning strategy to tackle the issue. First, by utilizing the prior tracking model as supervision, the update process can implicitly retain relevant properties of the canonical space, thus stabilizing the training process. Additionally, since color information is already encoded in the pre-trained model, freezing MLP layers that contribute to this information leads to faster convergence without sacrificing tracking quality.

We evaluate our method on two different benchmarks: the Plenoptic Video Dataset (Li et al. 2022) and the Immersive Video Dataset (Broxton et al. 2020), both of which are manually labeled for our experiments. The results demonstrate that our approach provides accurate point correspondences through long sequences across cameras, outperforming feature matching using DINO (Oquab et al. 2023). Furthermore, we show that our model effectively updates pixel-level tracking incrementally while mitigating the negative effects of catastrophic forgetting. Figure 1 illustrates qualitative examples of pixel-tracking performance by our method.

2 Related Work

Point Tracking. Point tracking has gained significant attention in recent years for its vital role in computer vision applications. The TAP-Vid benchmark by Doersch et al. (Doersch et al. 2022) serves as a pivotal foundation for Tracking-Any-Point (TAP) advancements. Enhancing point tracking in videos, TAPIR (Doersch et al. 2023) employs global trajectory refinement with convolution layers, while CoTracker (Karaev et al. 2023) and Context-TAP (Weikang et al. 2023) utilize optical flow estimation with spatial context for optimized TAP performance. SpatialTracker (Song et al. 2024) adopts the framework of CoTracker while operating in 3D space instead. Addressing long-term tracking, MFT (Neoral, Šerỳch, and Matas 2023; Jelínek, Šerỳch, and Matas 2024) and Leap-VO (Chen et al. 2024) focus on extrapolating long-term trajectories from short-term point interactions. DINO-Tracker leveraged pre-trained DINO features as a prior and proposed a self-supervised test-time training strategy for point tracking (Tumanyan et al. 2024). Inspired by Omnimotion (Wang et al. 2023a), who proposed a 3D canonical space for video feature analysis, our research seeks to address the unresolved challenges of occlusions and fast-moving object tracking in this field. Recently, CaDeX++ introduced a test-time optimization method that could track pixels faster and more robustly (Xiao et al. 2024).

Feature Matching. Feature matching techniques can estimate dense correspondences between distant pairs of video frames. Several methods, such as SuperGlue (Sarlin et al. 2020), directly compare the feature of two images to filter out the dense correspondence, while LoFTR(Sun et al. 2021) uses a two-phase process to filter correspondence from coarse level to fine level. In addition, PATS (Ni et al. 2023) enables the model to handle scale differences in images when matching and setting many-to-many relationships between images, which shows superior performance in downstream tasks.

Optical Flow. Optical flow has been traditionally defined as an optimization problem that seeks to get the motion in a sequence or pair of images. Recently, many works have improved the performance of neural network to directly predict optical flow. From the primitive work FlowNet (Dosovitskiy et al. 2015), which uses a simple convolution network to predict the flow, to one leading method RAFT (Teed and Deng 2020), estimating flow through iterative updates of a flow field based on 4D correlation volumes. VideoFlow (Shi et al. 2023a) proposes MOtion Program (MOP) to handle optical flow for multiple frames. While optical flow methods allow for precise motion estimation between consecutive frames, they are not suited to long-range motion estimation, which is a topic that remains unsolved.

Incremental Learning. Incremental learning involves progressively training a model to incorporate new knowledge while retaining previously learned information, with a key challenge being the avoidance of catastrophic forgetting (French 1999). Several works have addressed this task in the area of Neural Radiance Fields (NeRF) (Wu et al. 2024; Yan et al. 2024; Cai and Müller 2023; Zhang et al. 2023; Guo et al. 2022; Po et al. 2023; Chung et al. 2022; Wu and Tuytelaars 2023). These methods primarily utilize knowledge distillation to alleviate catastrophic forgetting. Previous data are distilled by either storing keyframes from seen data or generated by pre-trained models, and optimized with new image frames. INV (Wang et al. 2023b) analyzed the function of MLP layers in the NeRF network and proposed an approach to accelerate the incremental training process by freezing certain layers of the network without losing structural or color information.

Multi-Camera Tracking. Multi-camera tracking is a challenging task since it requires strong feature matching abilities to track numerous targets in various views. Most works focus on multiview object detection and tracking (Amosa et al. 2023; Xu et al. 2017; Huang et al. 2023; Gan et al. 2021; Feng et al. 2024; Han et al. 2021), leaving multiview point tracking an unsolved challenge. Our research endeavors aim to address this challenge by enhancing the ability of a single-view point tracking model.

3 Our Method

3.1 Overview

Finding pixelwise correspondences across time frames is essential for the pixel tracking problem. Omnimotion (Wang et al. 2023a) addresses this by representing the input video sequence as a canonical 3D volume. To construct this volume, the method begins by querying a pixel p_i from frame i of the input video. The 3D information of p_i is obtained by sampling points along the ray $\mathbf{x}_i(z) = \mathbf{o}(p_i) + z\mathbf{d}$, where $\mathbf{o}(p_i)$ denotes the ray's origin, $\mathbf{d}$ is the direction of the camera, and z represents the depth along the ray. Each pixel p_i is assigned K sampled 3D points, denoted as $\{\mathbf{x}_i^k\}$. These sampled points are then mapped to their corresponding canonical 3D point $\{\mathbf{u}^k\}$ by an invertible mapping network $T_i(\cdot) = T_\theta(\cdot\,;\psi_i)$ conditioned on a time-dependent latent vector ψ_i. The same mapping network is used to map $\mathbf{u}^k$ back to a target frame j to locate its corresponding $\mathbf{x}_j^k$:

$$\begin{aligned} T_j^{-1} \circ T_i(\mathbf{x}_i^k) = T_j^{-1} \circ T_\theta(\mathbf{x}_i^k;\psi_i) = T_j^{-1}(\mathbf{u}^k) \\ = T_\theta^{-1}(\mathbf{u}^k;\psi_j) = \mathbf{x}_j^k. \end{aligned} \tag{1}$$

Latent vectors $\{\psi_i\}$ are modeled by a latent MLP L_θ. An additional MLP is introduced to predict the color and density of the 3D canonical point, denoted as $F_\theta(\mathbf{u}^k) = (\mathbf{c}_k, \sigma_k)$. The corresponding 3D point $\mathbf{x}_j$ of frame j can be computed by alpha blending all the predicted samples $\mathbf{x}_j^k$. We have

$$\hat{\mathbf{x}}_j = \sum_{k=1}^{K} \pi_k \alpha_k \mathbf{x}_j^k, \text{ where } \pi_k = \prod_{\ell=1}^{k-1}(1-\alpha_\ell) \text{ and } \alpha_\ell = 1 - e^{-\sigma_\ell}. \tag{2}$$

The image-space color C_j can be obtained using the same equation as above, replacing $\mathbf{x}_j^k$ with color $\mathbf{c}_k$. The resulting $\hat{\mathbf{x}}_j$ from (2) is then mapped to the pixel p_j of frame j using the same projection parameters.

3.2 Problem Formulation

Given V time-synchronized video sequences, denoted as $\{I_t^v\}_{t=1,v=1}^{T,V}$, our objective is to track any 2D pixels across V cameras and T time frames. For each query pixel p_t^v, the model predicts its corresponding position $\hat{p}_{t'}^{v'} = (\hat{x}_{t'}^{v'}, \hat{y}_{t'}^{v'})$ and occlusion state $\hat{O}_{t'}^{v'} \in \{0,1\}$, where $v, v' \in \{1,...,V\}$ and $t, t' \in \{1,...,T\}$.

3.3 Multi-Camera Pixel Tracking

Omnimotion employs an additional latent MLP to derive the time-dependent latent code ψ_t over time. We see from (1) that the mapping network T_θ will take ψ_t as input and map the local 3D coordinates to time-independent canonical coordinates, namely $\mathbf{u} = T_\theta(\mathbf{x}_t;\psi_t)$. In our proposed multiview scenario, illustrated in Figure 2, it is essential to model the latent code considering both view and temporal aspects. Given that the camera parameters for each view are typically unknown, it may not be suitable to assign a random view index to each view for generating latent codes, because such random indices offer no meaningful information to the model. We instead introduce *learnable per-view vectors* $\{\mathbf{s}_v\}_{v=1}^V$ to generate the global latent code alongside the time index. Specifically, we have $\psi_{v,t} = L_\theta(\mathbf{s}_v, t)$ and the invertible network is now denoted as $T_{v,t}(\cdot) = T_\theta(\cdot\,;\psi_{v,t})$. The per-view vectors are optimized jointly with the latent-code MLP L_θ, allowing the model to adapt to view-specific content without requiring prior knowledge of camera poses.

3.4 Loss Functions

Pixel Loss. We minimize the mean absolute error (MAE) between the predicted flow supervision points for frames within the same camera view and correspondence supervision points for frames across different cameras. Let Ω_p include the set of all the filtered correspondences $(p_{i,v}, p_{j,v'})$ between the image frame i of view v and the image frame j of view v'. We define the *pixel loss* as

$$\mathcal{L}_p = \sum_{(p_{i,v},p_{j,v'})\in\Omega_p} \left\|\hat{p}_{i,v\to j,v'} - p_{j,v'}\right\|_1. \tag{3}$$

Photometric Loss. For visual consistency, the *photometric loss* minimizes the mean squared error (MSE) between the predicted pixel color $\hat{C}_{j,v'}$ and the observed color $C_{j,v'}$ in the source video frame. It is formulated as

$$\mathcal{L}_c = \sum_{(p_{i,v},p_{j,v'})\in\Omega_p} \left\|C(\hat{p}_{i,v\to j,v'}) - C(p_{j,v'})\right\|_2^2. \tag{4}$$

Temporal Loss. To ensure temporal smoothness of the 3D motion between frames, we predict the forward and backward motions of an input 3D point $\mathbf{x}_t$ at time t, yielding the *temporal loss*:

$$\mathcal{L}_t = \sum_{(\mathbf{x},t)\in\Omega_\mathbf{x}} \left\|\mathbf{x}_{t+1} + \mathbf{x}_{t-1} - 2\mathbf{x}_t\right\|_1, \tag{5}$$

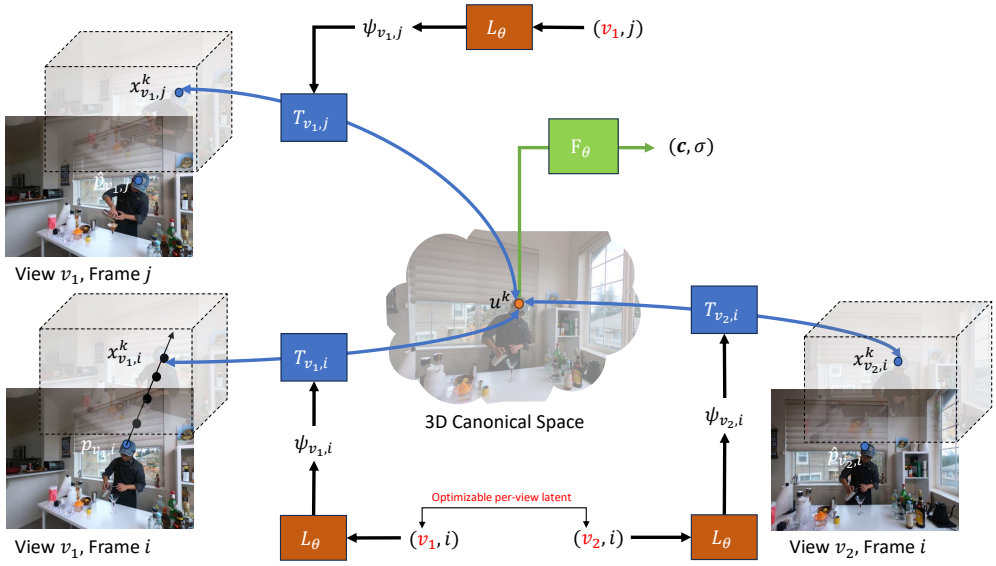

Figure 2: The proposed multi-view framework for tracking every pixel across different cameras.

where $\Omega_{\mathbf{x}}$ includes all sampled 3D local points from all frames of each camera view.

Spatial Loss. Observe that pixels often lag behind the moving object at the edges when the object undergoes occlusion or large motion. To overcome this challenge, we add an additional spatial smoothness constraint that propagates information between neighboring pixels. Specifically, we encourage pixels belonging to the same local part to move together over time:

$$\mathcal{L}_s = \sum_{(p_{i,v}, p_{j,v'}) \in \Omega_p} \sum_{q_i \in \mathcal{N}(p_i)} w_{p,q} \left\| d(p_{i,v}, q_{i,v}) - d(\hat{p}_{j,v'}, \hat{q}_{j,v'}) \right\|_1, \quad w_{p,q} = e^{\mathrm{sim}(\mathbf{f}(p_i), \mathbf{f}(q_i))}, \tag{6}$$

where $d(\cdot)$ gives the l_1 distance between two pixels, $\mathcal{N}(p_i)$ includes the four neighboring pixels of p_i, and $\mathbf{f}(p)$ represents the feature of pixel p, obtained by bilinear interpolation of the pre-trained DINO feature maps, and $\mathrm{sim}(\cdot, \cdot)$ is the cosine similarity measure. With (3–6), we arrive at the total loss for learning multi-view pixel tracking, defined as their weighted sum:

$$\mathcal{L} = \mathcal{L}_p + \lambda_c \mathcal{L}_c + \lambda_t \mathcal{L}_t + \lambda_s \mathcal{L}_s, \tag{7}$$

where $\lambda_c, \lambda_t, \lambda_s$ are the respective loss balancing weights.

3.5 Incremental Tracking

Our method for multi-view pixel tracking can be generalized to perform incremental tracking. Given a multi-view pixel tracking model, characterized by the three MLPs: $(L_\theta, F_\theta, T_\theta)$, learned off-line from a set of training images $\{I^v_1, ..., I^v_t\}^V_{v=1}$, our goal is to perform pixel-tracking for future frames $\{I^v_{t+1}, ..., I^v_T\}^V_{v=1}$ by incrementally updating the trained model.

As outlined in Algorithm 1, we employ a sliding window scheme to achieve incremental tracking. For tracking on frames $\{I^v_{t+1}\}^V_{v=1}$, we freeze the last two layers of the color MLP F_θ, as the MLP can be divided into two parts: structure layers and color layers. Structural details are constructed in the earlier layers, while color information is stored in the later layers. With the pre-trained model provided, the color/later layers have already been trained. Therefore, we only train the first two layers of F_θ to model the structural change. The model is updated using correspondences collected from the image set $\{I^v_{t-m+2}, ..., I^v_{t+1}\}^V_{v=1}$, where m is the length of the sliding window.

Algorithm 1: Incremental Tracking Algorithm

Input: $L_\theta, F_\theta, T_\theta$, m = sliding window size, T = total number of frames.
Initialize: $t = m$, Freeze last two layers of F_θ, $L_{\theta,t} = L_\theta, F_{\theta,t} = F_\theta, T_{\theta,t} = T_\theta$
while $t + 1 \leq T$ **do**
  collect pairwise correspondences Ω_p from frames $(t - m + 2$ to $t + 1)$
  $L_{\theta,t+1}, F_{\theta,t+1}, T_{\theta,t+1} \leftarrow L_{\theta,t}, F_{\theta,t}, T_{\theta,t}$
  while *step < number of iteration* **do**
    sample pairwise correspondences from Ω_p
    update $L_{\theta,t+1}, F_{\theta,t+1}, T_{\theta,t+1}$ using $L_{\theta,t}, F_{\theta,t}, T_{\theta,t}$
  end
  $L_{\theta,t}, F_{\theta,t}, T_{\theta,t} \leftarrow L_{\theta,t+1}, F_{\theta,t+1}, T_{\theta,t+1}$
  $t \leftarrow t + 1$
end

Moreover, we add a regularization loss to enforce the canonical coordinates predicted by the current training model remain unchanged, i.e.,

$$\mathcal{L}_{\mathrm{Reg}} = \sum_{(\mathbf{x},t) \in \Omega_x} \left\| T_{\theta,t}(\mathbf{x}_t; \psi_{v(\mathbf{x}),t}) - T_{\theta,t+1}(\mathbf{x}_{t+1}; \psi_{v(\mathbf{x}),t+1}) \right\|_1, \tag{8}$$

where $\Omega_{\mathbf{x}}$ is now reduced to 3D sampled points within the sliding window, and $v(\mathbf{x})$ is the underlying camera view

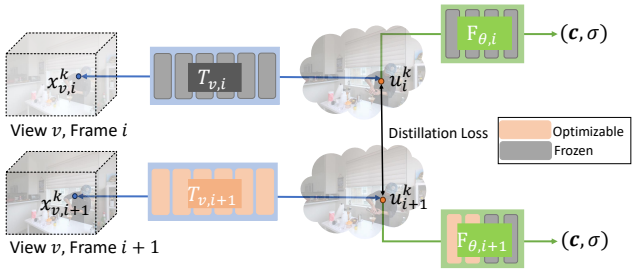

Figure 3: Multi-view incremental pixel tracking can be efficiently performed via optimizing L_θ, partially updating F_θ and enforcing model distillation on T_θ based on information in a sliding window.

for point $\mathbf{x}$. Besides the total loss in (7), the regularization loss encourages model distillation in adapting to new scenes while keeping the constructed 3D canonical space *unchanged*, thereby further accelerating and stabilizing the incremental training process.

3.6 Collecting Cross-View Input Data

Dense correspondences within the same camera view are obtained following the work of (Wang et al. 2023a). For cross-camera correspondences, PATS (Ni et al. 2023) is employed. PATS is a dense feature-matching algorithm capable of performing pixel-level matching under severe scale variations and in indistinctive regions. Furthermore, PATS is trained in a self-supervised manner, making it less sensitive to generalization gaps between training and unseen images. To enhance the quality of collected correspondences, pre-trained DINO feature maps are applied to filter out inaccurate correspondences. We extract dense features for each pixel from the feature map and exclude correspondences whose features' cosine similarity is less than 0.5.

4 Experiments

4.1 Experiment Setup

Training Details. We select four different views and train for a total of 200,000 iterations for each scene. View selection is done by ensuring that each video has enough variance in terms of view angle and scene coverage. We employ a two-phase training process. In phase 1, we perform the warm-up training for 100,000 iterations by selecting a random view and reducing the problem to pixel correspondence learning, akin to the single-view scenario. Phase 2 involves learning pixel correspondences both within a single view and between different views. For the incremental tracking experiments, we pre-train our model for the first 10 frames with 50,000 iterations via the same procedure. For each timestep thereafter, we train for 2,000 iterations with incremental settings. For details on the model architecture, please refer to the supplementary materials.

Datasets. We collect multi-view data from the Plenoptic Video Dataset (Li et al. 2022) and the Immersive Video Dataset (Broxton et al. 2020). Both datasets consist of time-synchronized multi-view camera videos with significant view variations. The Plenoptic Video Dataset includes six scenes captured with 21 cameras at a resolution of 2704×2028. The Immersive Video Dataset is a synthetic dataset captured with 46 cameras at 4K resolution. We select eight scenes that have a salient object centered in the video, making them suitable for our task. We downsample both datasets to a resolution of 640×480 for our experiments and reduce the frame rate to one-fifth of the original. To obtain ground truth for quantitative comparisons, we label the correspondences between the first frame of the query view and the target view. The predicted points of subsequent frames for each view are then compared with the results of a pre-trained single-view tracking model, which is trained on each camera independently. Details of the labeling process are provided in the supplemental material.

Evaluation Metrics. We evaluate our performance using the same metrics as the TAP-Vid benchmark (Doersch et al. 2022). $(\mathbf{1}) < \delta^{\mathbf{x}}_{\mathbf{avg}}$: evaluate the average position accuracy where the points are visible. Position accuracy is defined as the ratio of points that lie within a specified threshold of their ground truth positions. $< \delta^x_{avg}$ is calculated by averaging the ratios at five different thresholds: 1, 2, 4, 8, and 16 pixels. **(2) Average Jaccard (AJ):** consider both the accuracy of the predicted location and visibility. The metric is calculated as the fraction of true positives (i.e., points within the threshold of visible ground truth points) over all points. The threshold is the same as that used for $< \delta^x_{avg}$. **(3) Median Trajectory Error (MTE):** finds the median distance discrepancy between the given estimated tracks and the ground truth tracks. **(4) Occlusion Accuracy (OA):** computes the ratio of predicted points that are accurately classified as either visible or occluded.

4.2 Comparison

Although numerous prior studies have attempted to address multi-view tracking problems, their approaches are limited to object-level tracking, making them unsuitable for direct application to our pixel-level setup. Hence, we employ DINO (Oquab et al. 2023), a self-supervised correspondence matching method, as the baseline for our multi-camera experiments. Note that there are several methods, such as SuperGlue (Sarlin et al. 2020) and LoFTR (Sun et al. 2021), that support pixel-level matching. However, these methods do not guarantee matching for every pixel. In contrast, DINO provides a matching outcome for the target frame whenever a pixel in the query frame is selected.

DINO. (Oquab et al. 2023) uses pre-trained DINO feature maps to establish correspondences between the query frame and all target frames across cameras. The matching process mimics the video object tracking algorithm provided in the DINO source code. The main algorithm involves restricting neighborhood predictions based on previous matchings and utilizing nearest-neighbor between frames to facilitate temporal information.

Results on Multi-Camera Tracking

Method	Plenoptic				Immersive			
	AJ ↑	$< \delta^x_{avg}$ ↑	MTE ↓	OA ↑	AJ ↑	$< \delta^x_{avg}$ ↑	MTE ↓	OA ↑
DINO	-	35.8	52.2	-	-	37.1	50.8	-
Ours	76.5	87.8	8.3	97.4	62.6	78.5	14.2	91.1
Ours incremental	68.6	80.5	11.9	91.3	53.9	65.7	26.8	84.2

Table 1: **Quantitative comparison with the baseline on the Plenoptic and the Immersive benchmarks.** We report four metrics for our method. We do not include AJ and OA in the DINO experiments, as these two metrics involve classifying pixels occlusion/visibility, which DINO cannot support. Our method performs well in both offline and incremental training scenarios.

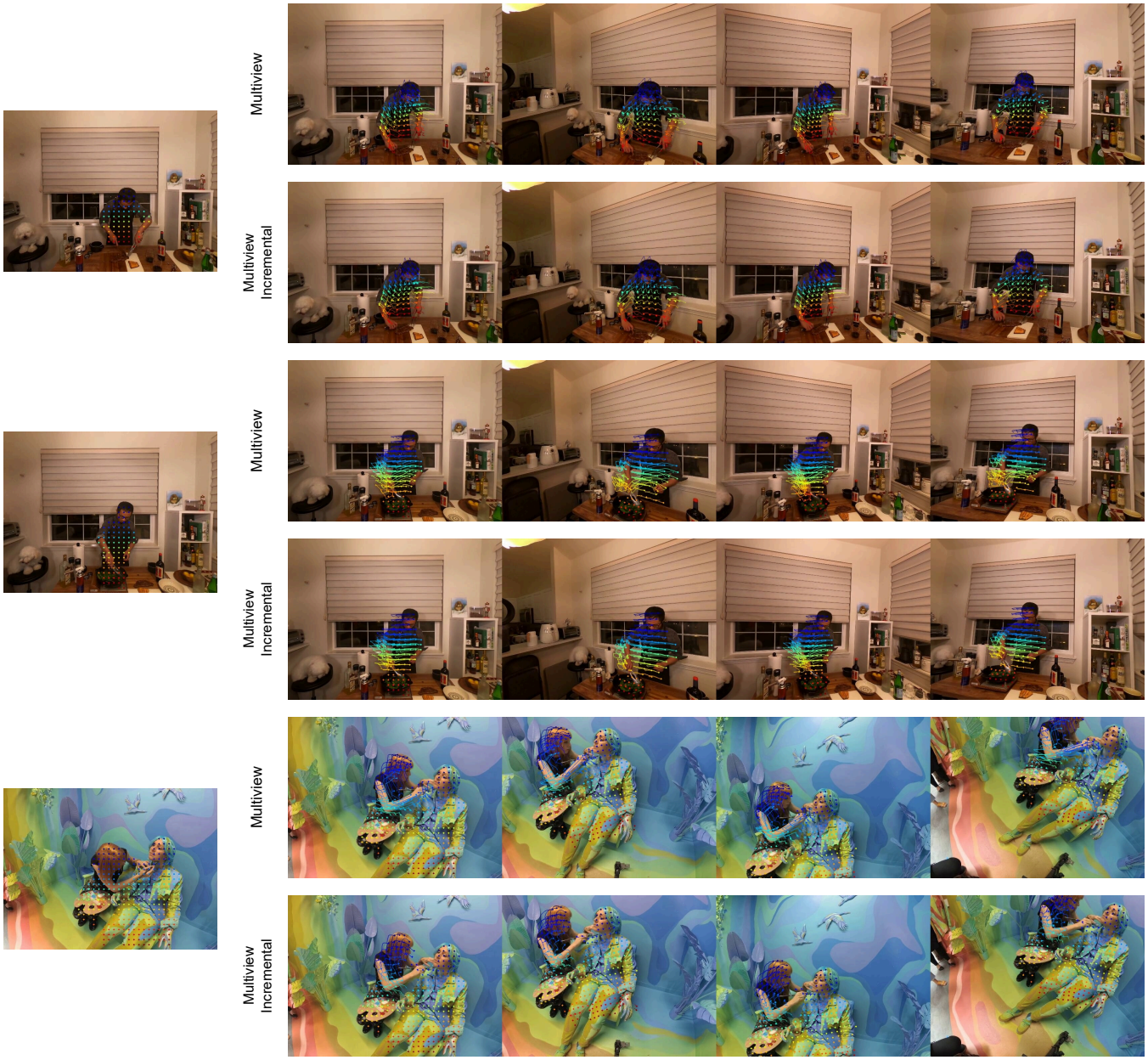

Figure 4: **Qualitative examples of our method.** The leftmost image shows the query pixels we intend to track. Each row represents different views of the same scene. We illustrate our qualitative results by visualizing the trajectories for each pixel in its own view. Our method successfully provides accurate correspondences and smooth trajectories across views and timesteps.

Method	Plenoptic				Immersive			
	AJ ↑	$< \delta^x_{avg}$ ↑	MTE ↓	OA ↑	AJ ↑	$< \delta^x_{avg}$ ↑	MTE ↓	OA ↑
w/o distillation loss	65.6	77.7	20.6	88.9	52.1	66.0	37.1	85.1
w/o parameter freezing	66.3	76.8	19.6	84.6	52.3	64.5	35.5	78.4
Full	**66.4**	**77.9**	**15.4**	**89.5**	**53.3**	**66.8**	**29.0**	**85.3**

Table 2: Ablation study on incremental tracking

Method	Plenoptic				Immersive			
	AJ ↑	$< \delta^x_{avg}$ ↑	MTE ↓	OA ↑	AJ ↑	$< \delta^x_{avg}$ ↑	MTE ↓	OA ↑
w/o spatial loss	75.8	84.3	13.6	96.5	64.3	78.8	16.3	**93.0**
w/o per-view latent	76.6	84.5	9.8	95.3	68.4	82.8	13.9	91.5
Full	**78.7**	**87.0**	**8.6**	**96.9**	**69.5**	**83.2**	**12.8**	92.3

Table 3: Ablation study on spatial loss and per-view latent

Qualitative Comparison. We show qualitative examples of multi-view tracking in Figure 4. DINO produces undesirable trails due to its inability to effectively utilize temporal information and account for occlusion during matching, leading to tracking failures when a pixel becomes occluded. In contrast, our method accurately maps correspondences across cameras and performs tracking in long video sequences. The visualizations show smooth and temporally coherent trajectories for each pixel. Incremental tracking results demonstrate comparable performance tracking results and do not suffer from catastrophic forgetting.

Quantitative Comparison. As shown in Table 1, our method outperforms the DINO baseline by a large margin across all metrics, indicating that our model achieves higher accuracy and robustness in tracking. In the incremental experiments, metrics are computed from the frame when incremental updating begins (i.e., the 11th frame in our experiments). The performance closely approaches the results obtained from the offline training process, illustrating the effectiveness of our approach.

4.3 Ablation Studies

We run ablation studies to evaluate the effectiveness of our proposed methods on a subset of the two datasets. We choose three scenes from both datasets, respectively.

Incremental Tracking. We ablate various components of our approach for incremental tracking. Quantitative results are shown in Table 2. *w/o distillation loss* removes the distillation loss from the loss function, and *w/o parameter freezing* tunes the entire model without freezing any parameters. Without either of these components, the model struggles to provide stable and consistent trajectories. The drop in occlusion accuracy without parameter freezing occurs because the color information already stored in the model is altered during the incremental updating process, leading to less accurate visibility predictions.

Spatial Loss and Per-View Latent. We perform ablations on two designs for multi-view tracking in Table 3. *w/o spatial loss* disables the spatial loss during training, resulting in a performance drop, particularly in scenes with rapidly moving objects. The result highlights the importance of spatial loss in preserving the local geometric relationships between pixel pairs. We provide a qualitative example in Figure 5. *w/o per-view latent* replaces the optimizable per-view latent with a one-dimensional pre-determined view index. This substitution limits the model's ability to capture view-dependent information, leading to poorer performance.

Figure 5: **Qualitative comparison of results with and without the use of spatial loss.**

5 Conclusions

Our work tackles the challenge of tracking individual points across multiple video sequences captured from diverse camera views. By leveraging a unified global canonical space, our method establishes accurate correspondences across both cameras and time frames. To further enhance cross-view tracking, we introduce a per-view latent code to encode view information and a spatial loss to promote consistency and rigidity among neighboring pixels. To adapt to new data, we provide an incremental learning strategy that incorporates parameter-efficient training and a novel distillation loss to overcome catastrophic forgetting and ensure stability. Experimental results demonstrate the superiority of our proposed method, showcasing its effectiveness in providing accurate point correspondences and robust long-term pixel tracking across varying views and scenes.

Acknowledgments

This work was supported in part by NSTC grants 111-2221-E-001-015-MY3, 111-2221-E-001-011-MY2, 113-2221-E-001-010-MY3 and 113-2634-F-007-002 of Taiwan. We thank National Center for High-performance Computing for providing computing resources.

References

Amosa, T. I.; Sebastian, P.; Izhar, L. I.; Ibrahim, O.; Ayinla, L. S.; Bahashwan, A. A.; Bala, A.; and Samaila, Y. A. 2023. Multi-camera multi-object tracking: a review of current trends and future advances. *Neurocomputing*, 552: 126558.

Broxton, M.; Flynn, J.; Overbeck, R.; Erickson, D.; Hedman, P.; Duvall, M.; Dourgarian, J.; Busch, J.; Whalen, M.; and Debevec, P. 2020. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (TOG)*, 39(4): 86–1.

Cai, Z.; and Müller, M. 2023. Clnerf: Continual learning meets nerf. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 23185–23194.

Chen, W.; Chen, L.; Wang, R.; and Pollefeys, M. 2024. LEAP-VO: Long-term Effective Any Point Tracking for Visual Odometry. *arXiv preprint arXiv:2401.01887*.

Chung, J.; Lee, K.; Baik, S.; and Lee, K. M. 2022. Meilnerf: Memory-efficient incremental learning of neural radiance fields. *arXiv preprint arXiv:2212.08328*.

Doersch, C.; Gupta, A.; Markeeva, L.; Recasens, A.; Smaira, L.; Aytar, Y.; Carreira, J.; Zisserman, A.; and Yang, Y. 2022. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35: 13610–13626.

Doersch, C.; Yang, Y.; Vecerik, M.; Gokay, D.; Gupta, A.; Aytar, Y.; Carreira, J.; and Zisserman, A. 2023. TAPIR: Tracking Any Point with per-frame Initialization and temporal Refinement. *arXiv preprint arXiv:2306.08637*.

Dosovitskiy, A.; Fischer, P.; Ilg, E.; Hausser, P.; Hazirbas, C.; Golkov, V.; Van Der Smagt, P.; Cremers, D.; and Brox, T. 2015. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, 2758–2766.

Edstedt, J.; Athanasiadis, I.; Wadenbäck, M.; and Felsberg, M. 2023a. DKM: Dense kernelized feature matching for geometry estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17765–17775.

Edstedt, J.; Sun, Q.; Bökman, G.; Wadenbäck, M.; and Felsberg, M. 2023b. RoMa: Revisiting Robust Losses for Dense Feature Matching. *arXiv preprint arXiv:2305.15404*.

Feng, W.; Wang, F.; Han, R.; Gan, Y.; Qian, Z.; Hou, J.; and Wang, S. 2024. Unveiling the Power of Self-supervision for Multi-view Multi-human Association and Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

French, R. M. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4): 128–135.

Gan, Y.; Han, R.; Yin, L.; Feng, W.; and Wang, S. 2021. Self-supervised multi-view multi-human association and tracking. In *Proceedings of the 29th ACM international conference on multimedia*, 282–290.

Guo, M.; Li, C.; Chen, H.; and Lee, G. H. 2022. Incremental Neural Implicit Representation with Uncertainty-Filtered Knowledge Distillation. *arXiv preprint arXiv:2212.10950*.

Han, R.; Feng, W.; Zhang, Y.; Zhao, J.; and Wang, S. 2021. Multiple human association and tracking from egocentric and complementary top views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9): 5225–5242.

Huang, H.-W.; Yang, C.-Y.; Jiang, Z.; Kim, P.-K.; Lee, K.; Kim, K.; Ramkumar, S.; Mullapudi, C.; Jang, I.-S.; Huang, C.-I.; et al. 2023. Enhancing Multi-Camera People Tracking with Anchor-Guided Clustering and Spatio-Temporal Consistency ID Re-Assignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5238–5248.

Huang, Z.; Shi, X.; Zhang, C.; Wang, Q.; Cheung, K. C.; Qin, H.; Dai, J.; and Li, H. 2022. Flowformer: A transformer architecture for optical flow. In *European conference on computer vision*, 668–685. Springer.

Jelínek, T.; Šerỳch, J.; and Matas, J. 2024. Dense Matchers for Dense Tracking. *arXiv preprint arXiv:2402.11287*.

Karaev, N.; Rocco, I.; Graham, B.; Neverova, N.; Vedaldi, A.; and Rupprecht, C. 2023. CoTracker: It is Better to Track Together. *arXiv:2307.07635*.

Li, T.; Slavcheva, M.; Zollhoefer, M.; Green, S.; Lassner, C.; Kim, C.; Schmidt, T.; Lovegrove, S.; Goesele, M.; Newcombe, R.; et al. 2022. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5521–5531.

Nam, J.; Lee, G.; Kim, S.; Kim, H.; Cho, H.; Kim, S.; and Kim, S. 2023. Diffmatch: Diffusion model for dense matching. *arXiv preprint arXiv:2305.19094*.

Neoral, M.; Šerỳch, J.; and Matas, J. 2023. MFT: Long-Term Tracking of Every Pixel.

Ni, J.; Li, Y.; Huang, Z.; Li, H.; Bao, H.; Cui, Z.; and Zhang, G. 2023. Pats: Patch area transportation with subdivision for local feature matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17776–17786.

Oquab, M.; Darcet, T.; Moutakanni, T.; Vo, H.; Szafraniec, M.; Khalidov, V.; Fernandez, P.; Haziza, D.; Massa, F.; El-Nouby, A.; et al. 2023. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*.

Pautrat, R.; Suárez, I.; Yu, Y.; Pollefeys, M.; and Larsson, V. 2023. Gluestick: Robust image matching by sticking points and lines together. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 9706–9716.

Po, R.; Dong, Z.; Bergman, A. W.; and Wetzstein, G. 2023. Instant continual learning of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3334–3344.

Sarlin, P.-E.; DeTone, D.; Malisiewicz, T.; and Rabinovich, A. 2020. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4938–4947.

Shen, X.; Darmon, F.; Efros, A. A.; and Aubry, M. 2020. Ransac-flow: generic two-stage image alignment. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, 618–637. Springer.

Shi, X.; Huang, Z.; Bian, W.; Li, D.; Zhang, M.; Cheung, K. C.; See, S.; Qin, H.; Dai, J.; and Li, H. 2023a. Videoflow: Exploiting temporal cues for multi-frame optical flow estimation. *arXiv preprint arXiv:2303.08340*.

Shi, X.; Huang, Z.; Li, D.; Zhang, M.; Cheung, K. C.; See, S.; Qin, H.; Dai, J.; and Li, H. 2023b. Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1599–1610.

Song, Y.; Lei, J.; Wang, Z.; Liu, L.; and Daniilidis, K. 2024. Track Everything Everywhere Fast and Robustly. *arXiv preprint arXiv:2403.17931*.

Sun, J.; Shen, Z.; Wang, Y.; Bao, H.; and Zhou, X. 2021. LoFTR: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8922–8931.

Teed, Z.; and Deng, J. 2020. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, 402–419. Springer.

Truong, P.; Danelljan, M.; and Timofte, R. 2020. GLU-Net: Global-local universal network for dense flow and correspondences. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 6258–6268.

Tumanyan, N.; Singer, A.; Bagon, S.; and Dekel, T. 2024. DINO-Tracker: Taming DINO for Self-Supervised Point Tracking in a Single Video. *arXiv preprint arXiv:2403.14548*.

Wang, Q.; Chang, Y.-Y.; Cai, R.; Li, Z.; Hariharan, B.; Holynski, A.; and Snavely, N. 2023a. Tracking Everything Everywhere All at Once. *arXiv preprint arXiv:2306.05422*.

Wang, S.; Supikov, A.; Ratcliff, J.; Fuchs, H.; and Azuma, R. 2023b. INV: Towards Streaming Incremental Neural Videos. *arXiv preprint arXiv:2302.01532*.

Weikang, B.; Huang, Z.; Shi, X.; Dong, Y.; Li, Y.; and Li, H. 2023. Context-PIPs: Persistent Independent Particles Demands Context Features. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Wu, M.; and Tuytelaars, T. 2023. NeVRF: Neural Video-based Radiance Fields for Long-duration Sequences. *arXiv preprint arXiv:2312.05855*.

Wu, X.; Dai, P.; Deng, W.; Chen, H.; Wu, Y.; Cao, Y.-P.; Shan, Y.; and Qi, X. 2024. CL-NeRF: Continual Learning of Neural Radiance Fields for Evolving Scene Representation. *Advances in Neural Information Processing Systems*, 36.

Xiao, Y.; Wang, Q.; Zhang, S.; Xue, N.; Peng, S.; Shen, Y.; and Zhou, X. 2024. SpatialTracker: Tracking Any 2D Pixels in 3D Space. *arXiv preprint arXiv:2404.04319*.

Xu, Y.; Liu, X.; Qin, L.; and Zhu, S.-C. 2017. Cross-view people tracking by scene-centered spatio-temporal parsing. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Yan, Q.; Wang, Q.; Zhao, K.; Chen, J.; Li, B.; Chu, X.; and Deng, F. 2024. CF-NeRF: Camera Parameter Free Neural Radiance Fields with Incremental Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 6440–6448.

Zhang, L.; Li, M.; Chen, C.; and Xu, J. 2023. IL-NeRF: Incremental Learning for Neural Radiance Fields with Camera Pose Alignment. *arXiv preprint arXiv:2312.05748*.