Track Everything Everywhere Fast and Robustly

Yunzhou Song[1*], Jiahui Lei[1*], Ziyun Wang[1],
Lingjie Liu[1], and Kostas Daniilidis[1,2]

[1] University of Pennsylvania
[2] Archimedes, Athena RC
{timsong,leijh,ziyunw,lingjie.liu,kostas}@cis.upenn.edu

https://timsong412.github.io/FastOmniTrack/

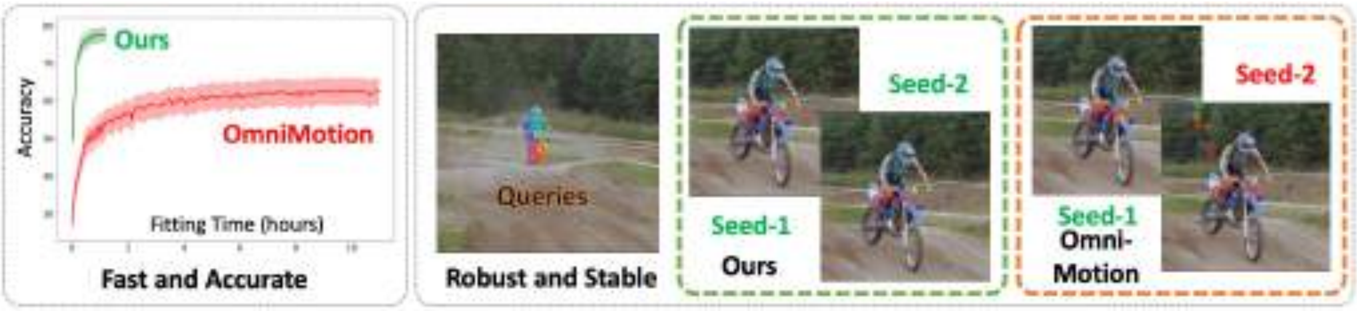

Fig. 1: Our optimization-based approach achieves fast and robust long-term tracking

Abstract. We propose a novel test-time optimization approach for efficiently and robustly tracking any pixel at any time in a video. The latest state-of-the-art optimization-based tracking technique, OmniMotion [34], requires a prohibitively long optimization time, rendering it impractical for downstream applications. OmniMotion [34] is sensitive to the choice of random seeds, leading to unstable convergence. To improve efficiency and robustness, we introduce a novel invertible deformation network, CaDeX++, which factorizes the function representation into a local spatial-temporal feature grid and enhances the expressivity of the coupling blocks with non-linear functions. While CaDeX++ incorporates a stronger geometric bias within its architectural design, it also takes advantage of the inductive bias??? provided by the vision foundation models. Our system utilizes monocular depth estimation??? to represent scene geometry and enhances the objective by incorporating DINOv2 long-term semantics to regulate the optimization process. Our experiments demonstrate a substantial improvement in training speed (more than **10 times** faster), robustness, and accuracy in tracking over the SoTA optimization-based method OmniMotion [34].

prior knowledge on preference

run the DINO v2 later

1 Introduction

The association of visual information from continuous observations across long time horizons lays the foundation for modern spatial intelligence. In computer vision, one of the key tasks that provides this association is the long-term tracking of pixels, which serves as the backbone for a wide spectrum of tasks, from 3D reconstruction to video recognition.

[*] Authors contributed equally to this work.

Previously, methods for estimating the correspondence can be divided into two categories based on their track representations. Feature-based methods represent points as local descriptors [1,20,28], which can be matched over a long time horizon, due to the various invariance properties built into their design. However, feature descriptors are often sparsely matched due to the quadratic matching cost between every pair of images. On the other hand, optical flow methods estimate the motion of pixels in a dense manner [12,30,31,33,35,38]. Due to the instantaneous nature of optical flow methods, they tend to perform poorly with long-range motion estimation and suffer from occlusion. Recently, several methods have been proposed to solve the problem via learning-based methods [8,9,11,14,43]. These methods learn strong prior knowledge by training on large synthetic datasets. In complement to learning-based methods, a new class of methods has emerged to optimize point tracks using test-time optimization on single scenes. A representative test-time optimization method is OmniMotion [34], which is optimized to reconstruct a dynamic scene with a NeRF [22,27] deformed by a global RealNVP [7,17], a normalizing flow network representing deformation. A major benefit of OmniMotion is that the optimization does not rely on strong prior knowledge, and is, thus, not susceptible to generalization gaps between training and testing. However, due to the losses being only photometric, OmniMotion converges slowly when the training data do not provide enough constraints due to object and view occlusions. Moreover, the quality of reconstruction is often unpredictable because of the unconstrained random network initialization.

Hope to learn by Model

data -driven

???

(x,y,z) -> (x',y',z')

In this paper, we focus on advancing the computational efficiency, robustness, and accuracy of test-time optimization tracking methods [34] by introducing inductive bias through visual foundation models and network architecture. One computational bottleneck of OmniMotion [34] is the cost of querying a global MLP-like NVP deformation network proposed first in CaDeX [17]. In Sec. 3.2, we introduce CaDeX++, a novel local feature-grid factorization of **invertible** deformation field, whose expressivity is further improved via a non-linear 1-D homeomorphism instead of the 1-D affine function in the NVP [17,34]. This design is inspired from NSVF [19], Instant-NGP [23] and TensoRF [5], which exploit local factorized representations to boost global MLP-based NeRFs [22]. Another time-consuming and under-constrained factor of OmniMotion [34] is the geometry reconstruction through volume rendering losses [22]. Instead, (Sec. 3.3) we regularize the optimization by initializing the optimizable per-frame depth map geometry based on monocular metric depth estimation, powered by the recent advances of 2D visual foundational models [2]. Finally, OmniMotion [34] only fits the short-term local optical flows [33], resulting in the lack of long-term association information. In Sec. 3.4, we incorporate this missing information via incorporating the foundational DINOv2 [25] feature correspondence into the fitting losses. Leveraging a novel factorization of an invertible deformation field and vision foundation models as regularizers yields a novel method that achieves tracking accuracy and robustness improvement over OmniMotion [34], while significantly reducing training time by more than **90%**. We outline our contributions as follows.

- An efficient and expressive novel invertible deformation network, CaDeX++, with local feature grid and non-linear interpolation.
- A novel depth-based geometry representation and the incorporation of DINOv2 [25] long-term semantics, which boosts and regularizes the tracking optimization process.
- Significant speed up, stabilization, and performance improvement over OmniMotion [34] in long-term tracking task.

2 Related Work

Pixel Tracking: Classical methods for estimating pixel correspondence can be divided into two categories: **keypoint tracking** and **optical flow**. For keypoint tracking methods, sparse feature descriptors are computed on local patches. Some common feature descriptors in visual odometry and SLAM methods include ORB [28], SIFT [20], and SURF [1]. Recently, a new class of methods has been proposed to learn feature descriptors using deep neural networks [6,42]. The correspondence between two sets of feature descriptors can be matched using pairwise difference or using learned matching networks [18]. Despite the different flavors of feature descriptors and matching algorithms, the correspondence is defined with respect to a predefined set of interest points. Detector-free methods [32] learn to match between all pairs of image locations without running feature detectors while having a global field of view. On the other hand, optical flow provides a dense correspondence field. In traditional optical flow computation, we often jointly optimize a data term and a regularization term. Horn and Schunck [12] optimize a global flow field through gradient descent while adding a smoothness term to solve the classical aperture problem. Lucas and Kanade [21] use the least squares criterion to optimize photoconsistency between flow-warped image patches and the new patches. This model solves an overdetermined system by assuming a parametric motion model. Later, learning-based approaches convert optimization terms into loss functions, allowing self-supervised optical flow training [31]. RAFT [33] uses recurrent neural networks to simulate optimization steps in traditional optical flow. This architecture has been widely used to address optical problems due to its superior performance in handling different object and flow scales [13,37,41].

Dense Long-range Tracking In the previous section, we provide an overview of two distinct types of tracking methods. Keypoint-based methods often provide correspondence over a longer time horizon, but the tracked points are usually sparse to save computational time. On the other hand, optical flow methods provide dense displacement of the pixels but fall short in long-term tracking consistency. Particle Video [29] is proposed to optimize long-range motion while preserving the density of optical flow estimation, by connecting short-term flow and regularizing the distortion between particles. Recently, PIPs [11] built on the original particle videos by proposing a deep MLP-Mixer module to iteratively update the long-term tracks. TAP-Net [8] uses a small neural network to directly

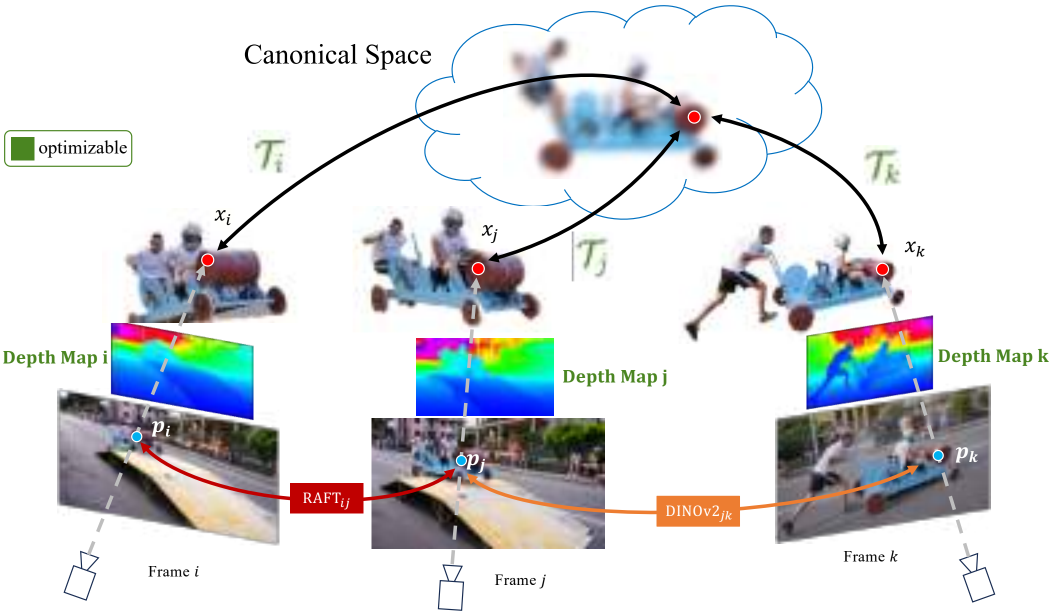

Fig. 2: Method Overview: To track a query pixel p_i, we first lift the pixel to 3D with an optimizable depth map (Sec. 3.3). The 3D point is deformed into the shared canonical space and back to another time frame j with a novel efficient and expressive invertible deformation field $\mathcal{T}$ (Sec. 3.2). The depth maps and the deformation $\mathcal{T}$ are optimized with both short-term dense RAFT [33] optical flow and long-term sparse DINOv2 [25] correspondence (Sec. 3.4).

RMN

regress point locations. TAPIR [9] starts with the TAP-Net initialization and refines the point trajectories with the MLP-Mixer architecture of PIPs. MFT [24] estimates the flow uncertainty and occlusion maps, which are used to select high-confidence flow chains to generate long-term tracks. In PointOdyssey [43], Zheng et al. design PIPs++ to increase the temporal field of view using a convolution over time and memorize the most recent appearance templates. OmniMotion [34] optimizes the per-pixel point tracks by lifting the 2D pixels into 3D and fitting optimizeable invertible warping functions. This representation allows for flexible tracking of long videos. The modeled scene is represented as a canonical 3D volume with its bijective mappings to each quasi-3D local scene. Both the canonical space and the mapping functions are jointly optimized.

3 Method

3.1 Preliminaries

Given an RGB video sequence $\{I_t\}_{t=1}^{T}$ with T frames and an arbitrary query pixel $p_t \in \mathbb{R}^2$ from a video frame I_t, our goal is to predict its long-term sequential trajectories $(\hat{p}_1, ..., \hat{p}_T)$ as well as the visibility $(\hat{v}_1, ..., \hat{v}_T) \in \{0, 1\}^T$. In the following sections, we will use subscripts to indicate the time or frame, and superscripts to indicate the track identity. An overview of our method is in Fig. 2.

A key consideration for an optimization-based long-term tracker is designing the parameterization of long-term tracks $(\hat{p}_j, \hat{v}_j) = \mathcal{F}(p_i, j)$, where $\mathcal{F}$ is the model of long-term tracks that takes input any query pixel p_i and destination

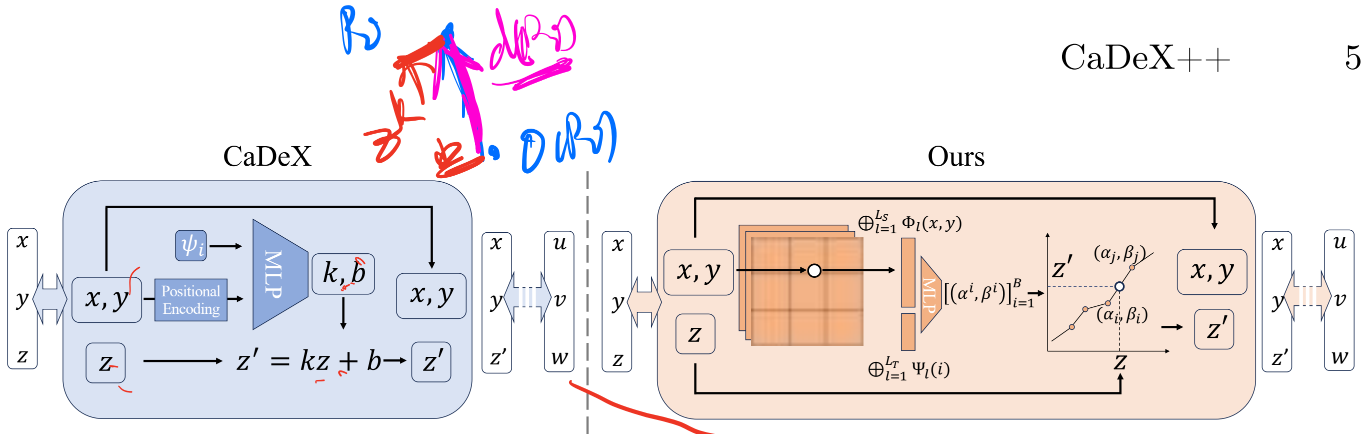

Fig. 3: Architecture of CaDeX++ (right). The deformation network has a stack of coupling blocks and gradually changes one coordinate dimension per block (For difference Sec. 3.2).

there are several candidates of z ⇒ Xi

time j, predicts the position $\hat{p}_j$ and visibility $\hat{v}_j$ at time j. The current SoTA OmniMotion [34] parameterizes this tracking function $\mathcal{F}$ as a rendering process [34]. First, the query pixel p_i at time i is marched along a line of points on the ray $x_i^k = o(p_i) + z^k d(p_i)$ where o, d and z are the ray center, direction, and marching depth, respectively. An invertible deformation field [17] $u = \mathcal{T}_i(x_i)$ is used to deform each ray-marching position at time i to the shared global canonical space position u, where the geometry and appearance of the scene are modeled as a canonical radiance field $(\text{color}, \sigma) = G(u)$. Finally, all the canonical positions on the bent ray are mapped back to the target time frame j with the inverse of the deformation field $\mathcal{T}_j^{-1}$:

the

$$x_j^k = \mathcal{T}_j^{-1}(u^k) = \mathcal{T}_j^{-1} \circ \mathcal{T}_i(x_i^k). \tag{1}$$

The prediction of the target pixel position can be formulated as a rendering process:

predicted position of the pixel at time j

$T_k\alpha_k = (1-\alpha_1)\cdots(1-\alpha_{k-1})\alpha_k$

$$\hat{p}_j = \pi\left(\sum_{k=1}^{K} T_k \alpha_k x_j^k\right), \quad T_k = \prod_{l=1}^{k-1}(1-\alpha_l), \quad \alpha_k = 1 - \exp(1-\sigma_k), \tag{2}$$

where π is the camera projection function and σ is the opacity predicted by the canonical radiance field G. The noisy short-term optical flow pairs $\mathcal{P}_{\text{RAFT}} = \{(p_i, p_j)\ i, j \in [1, ..., T]\}$ are usually assumed given as optimization targets, which are predicted by well-established networks like RAFT [33]. OmniMotion composes the tracking function above with a set of learnable $G, \mathcal{T}$ networks, and fits it against the noisy local optical flow pairs $\mathcal{P}$, while minimizing the rendering photometric errors. Similarly, the visibility can be found in the rendering process. For further information, readers are directed to Wang et al. [34]. We will see in the next sections why Eqs. 1 2 are inefficient and result in high-variant fittings and how we address these issues.

3.2 CaDeX++: Non-linear and Local Invertible NVPs

The expressivity and efficiency of $\mathcal{T}$ in Eq. 1 are critical when we extensively query the deformation field to model the long-track function $\mathcal{F}$. As shown in Fig. 3-Left, OmniMotion [34] uses CaDeX [17], a **global** NVP to parameterize

$\mathcal{T}$, which consists of a stack of coupling blocks. During each coupling iteration, a single dimension of the coordinates, such as z, is modified by a global MLP queried by the other two coordinates (x and y when z is modified) and a global latent code. To ensure the invertibility of this coupling step, z is changed by a simple 1-D affine mapping predicted by the MLP in the following coupling block:

$$(k,b) = \mathrm{MLP}\left([x,y];\psi_i\right), \quad z' = kz + b \tag{3}$$

where ψ_i is a time-dependent global latent code. A stack of such coupling blocks parameterized by per-block MLPs alternatingly changes the coordinates gradually. Please see CaDeX [17] and OmniMotion [34] for details.

However, the NVP [17] formulation in Eq. 3 has two main drawbacks, which we overcome with CaDeX++. First, the MLP and the latent codes are all global, which requires large networks for sufficient capacity. Inspired by global MLP-based NeRF versus local feature-grid-based representations, such as Instant-NGP [23] and TensoRF [5], we ask the question: Can we factorize the **invertible** deformation field [17] into **local** representations as well? At first sight, achieving invertibility may seem challenging due to the need for a specific network structure. We propose a novel approach to exploiting the desired locality by factorizing the latent code ψ while significantly reducing the *MLP* network size. Specifically, the latent code ψ that controls the coordinate deformation of each coupling block can be factorized into a multi-resolution lookup function. For example, we can factorize ψ in Eq. 3, indexed by unchanged coordinates, x, y and the time index i as

$$\psi(x,y,i) = \overset{\text{Time}}{\left(\oplus_{l=1}^{L_T}\Psi_l(i)\right)} \oplus \overset{\text{Spatial}}{\left(\oplus_{l=1}^{L_S}\Phi_l(x,y)\right)}, \tag{4}$$

where $\oplus$ denotes feature concatenation and L_T, L_S are the spatial and temporal feature grid resolution levels, respectively. $\Psi_l(i)$ and $\Phi_l(x,y)$ are bi-linearly querying a 1-D or 2-D feature grid at resolution l, respectively. Eq. 4 is a local spatial-temporal factorization that decouples time and space. Note that when we replace ϕ_i in Eq. 3 with $\psi(x,y,i)$ from Eq. 4, the invertibility still holds since $\psi(x,y,i)$ does not depend on the changing z coordinate.

Another drawback of CaDeX [17] in Eq. 3 is the insufficient expressivity of the affine function applied to the changing z dimension. The only requirement for invertibility is to ensure that the function that changes z is invertible and the affine function of the form $kz+b$ is the simplest among all such functions. To increase the expressivity within the limited number of coupling blocks, we propose to use the monotonic piece-wise functions as non-linear deformation. Specifically, the 1D function is parameterized by a list of B control points $[(\alpha^1,\beta^1),\dots(\alpha^B,\beta^B)]$ with piece-wise linear interpolation:

$$z' = \frac{z-\alpha^i}{\alpha^j-\alpha^i}(\beta^j-\beta^i)+\beta^i,\ z\in[\alpha^i,\alpha^j),\ j-i=1. \tag{5}$$

To guarantee the monotonicity of the control points, we make the network predict the positive delta values as:

$$[(\Delta\alpha^1,\Delta\beta^1)\dots,(\Delta\alpha^B,\Delta\beta^B)] = \mathrm{TinyMLP}\left([x,y];\psi(x,y,i)\right), \tag{6}$$

where $\Delta\alpha^b > 0$ and $\Delta\beta^b > 0$. For further information regarding the interpolation and network structures, please refer to our supplementary document. By incorporating locality and non-linearity inductive bias, we enhance both efficiency and expressiveness, all while preserving the essential guarantees of invertible characteristics.

3.3 Optimization with Depth Prior

Although we model the deformation field efficiently with CaDeX++, the optimization process of OmniMotion [34] can often be unstable and slow. The undesirable optimization performance arises from the scene's geometry being optimized using a volume rendering loss as described in Eq. 2. Moreover, with the small camera baseline in many casual videos, a standard NeRF [22]- may lead to a reconstruction that is highly ambiguous because the accuracy of the "triangulation" of photometric loss is compromised by the limited parallax. Therefore, we avoid such a NeRF-like reconstruction process by explicitly exploiting recent advances in foundational monocular metric depth estimation, i.e. ZoeDepth [2], which estimates a reasonably accurate and consistent geometry for each frame. Note that we use the **metric** depth models [2,10,39] as opposed to a scale-invariant depth models [4,15,36] to avoid inconsistency of scale within a video.

Given an initial depth map D_i estimated from ZoeDepth for every video frame, the tracking function $\mathcal{F}$ in Eq. 1 simply reduces to back-projection, deformation, and projection:

$$\hat{p}_j = \pi\left(\mathcal{T}_j^{-1} \circ \mathcal{T}_i(\pi^{-1}(D_i[p_i], p_i))\right), \tag{7}$$

where π^{-1} is the back-projection function that lifts the query pixel p_i with its depth $D_i[p_i]$ into 3D. Note that the projection distortion can be effectively absorbed into $\mathcal{T}$ because the deformation is learnable. We follow OmniMotion [34] to use a fixed pin-hole camera with a FOV of 40 degrees. Given the inaccuracy of the depth maps D_i obtained from ZoeDepth, we set all D_i **optimizable**, regularized by a smoothness term, as detailed in Section 3.5. In summary, the inefficient and under-constrained radiance field G in Eq. 2 is replaced with a list of optimizable depth maps $\{D_i\}_{i=1}^T$ to boost and stabilize the optimization process.

3.4 Incorporation of Long-term Semantics

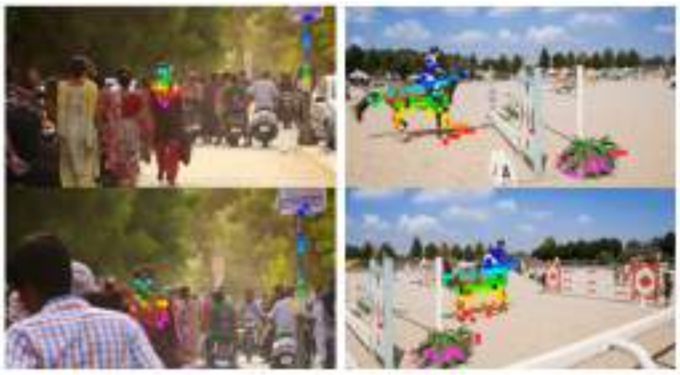

Fig. 4: Filtered long-range semantic correspondences based on DINOv2 [25].

To optimize $\mathcal{T}$, OmniMotion [34] relies solely on short-term optical flow as the fitting target. Inspired by recent progress in 2D visual foundational features, we incorporate sparse long-term semantic correspondence into the optimization targets, by using image features pre-trained on large image datasets. Specifically, we utilize and filter the DINOv2 [25] features to establish long-term correspondences different frames

(i,j) (j,i)

that are sparse but reliable. Given two DINOv2 feature maps F_i, F_j, we first compute the inter-frame pairwise cosine similarity for every two patch features between i, j and choose the mutually consistent nearest neighbor matches as candidates for long-term correspondence. This cycle consistency criterion means that a patch i whose best match is patch j must also be the closest match of Q6 patch j. We then filter the candidates by evaluating the self-similarity within each frame of the feature map. This helps us avoid any ambiguity in matching due to the absence of texture and noise in the feature map. Please refer to the additional materials for further information on the filtering process. In summary, through the utilization of DINOv2 [25], we augment the initial optical flow optimization objectives in OmniMotion [34] by incorporating a broader range that encompasses long-term sparse correspondence.

3.5 Training and Inference

During training (test-time optimization), given a pair of 2D correspondence from the target sets $(p_i, p_j) \in \mathcal{P} = \mathcal{P}_{\text{RAFT}} \bigcup \mathcal{P}_{\text{DINOv2}}$, we randomly choose one pixel as the query and another as the target. For the query pixel p_i, we back-project the p_i into 3D by looking up its depth from the optimizable depth map D_i as $x_i = \pi^{-1}(D_i[p_i], p_i)$. We then map x_i directly to time j by $\hat{x}_{i\to j} = \mathcal{T}_j^{-1}(\mathcal{T}_i(x_i))$ and project it to 2D screen to get the prediction pixel coordinate $\hat{p}_{i\to j} = \pi(\hat{x}_{i\to j})$ as in Eq. 7. We define the losses between $\hat{p}_{i\to j}$ and p_j as follows:

→ ground-truth

- **Pixel Position Loss**: We minimize the mean absolute error for both flow supervision points and long-term matching supervision points denoted as $\mathcal{L}_p$:

$$\mathcal{L}_p = \frac{1}{|\mathcal{P}|} \sum_{(p_i,p_j)\in\mathcal{P}} ||\hat{p}_{i\to j} - p_j||_1 \tag{8}$$

  where $\mathcal{P} = \mathcal{P}_{\text{RAFT}} \bigcup \mathcal{P}_{\text{DINOv2}}$ is the set of all correspondence generated by optical flow and long-term semantics.
- **Depth Consistency Loss**: Since the depth maps initialized from ZoeDepth [2] are not perfectly accurate, we additionally supervise the deformed point $\hat{x}_{i\to j}$ depth consistency with the target pixel's optimizable depth $D_j[p_j]$:

$$\mathcal{L}_d = \frac{1}{|\mathcal{P}|} \sum_{(p_i,p_j)\in\mathcal{P}} ||z(\hat{x}_{i\to j}) - D_j[p_j]||_1 \tag{9}$$

- **Depth Regularization Loss**: To ensure stability for depth optimization, we restrict the depth maps that can be optimized to remain near the initially set depth map. Given the initial ZoeDepth [2] depth map predictions D_i^{init} and their spatial gradients $\nabla_p D_i^{\text{init}}$, we regularize the optimized depth maps to stay close to the initialization:

$$\mathcal{L}_{\text{reg}} = \frac{1}{|\mathcal{P}|} \sum_{(p_i,p_j)\in\mathcal{P}} ||\nabla_p D_j^{\text{init}}[p_j] - \nabla_p D_j[p_j]||_2 + ||D_j^{\text{init}}[p_j] - D_j[p_j]||_1 \tag{10}$$

The final total loss is the weighted sum of the loss terms above:

$$\mathcal{L} = \mathcal{L}_p + \lambda_d \mathcal{L}_d + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}, \tag{11}$$

where $\lambda_p, \lambda_d, \lambda_{reg}$ are the loss balancing weights.

During inference, the long-term trajectory is efficiently predicted by Eq. 7 given any query position. For the visibility $\hat{v}_j$ at target time j, we simply compare the z value of the warped 3D point $\hat{x}_{i \to j}$ from query time i with $D_j[\hat{p}_{i \to j}]$, the depth value at frame j. Occlusion is detected if the warped 3D point is behind the depth value than a small threshold ϵ_d.

4 Experiments

4.1 Experiment Setup

Dataset: Following OmniMotion [34], we evaluated our method on the following datasets from TAP-Vid [8]:

- **DAVIS** [26], a real scene dataset of 30 videos from the DAVIS 2017 validation set. Each video contains 34 to 104 RGB frames. In this dataset, we observe both camera and scene motions.
- **RGB-Stacking** [16], a synthetic robot manipulation dataset. The dataset is composed of 50 videos, each with 250 RGB frames. The videos are rendered with only object motion with a static camera.

Metrics:

- $\delta^x_{\textbf{avg}}$ The average position precision percentage of tracked points that fall within x absolute pixel error of their targets. The metric is defined for all points that are visible in the ground truth. It has 5 thresholds δ^x, $x \in \{1, 2, 4, 8, 16\}$, where δ^x is the fractions of points that lie within x pixels of their ground truth position.
- **Average Jaccard (AJ)** The joint accuracy of points that are ground-truth visible. It measures the mean proportion of points that both lie within x pixels of their ground truth position and are predicted as visible.
- **Occlusion Accuracy (OA)** The fraction of the correct visibility prediction for all points in a frame. The numerator is the number of correct predictions including both visible and occluded points.
- **Temporal Coherence (TC)** The mean L_2 distance between the acceleration of actual tracks and predicted tracks is determined by calculating the difference in flow between three consecutive frames i, j, k for visible points, denoted as $f_{j \to k} - f_{i \to j}$. $i \to j \to k$,

We conducted all experiments on 480p images and evaluated metrics on 256x256 images following the training and evaluation protocols of OmniMotion [34].

4.2 Comparison with SoTA Methods

Baselines We compare our method with feed-forward methods and optimization-based methods. Some of the representative baselines are: **1) PIPs** [11] is a method that iteratively updates the position and visibility of a trajectory point within 8 frames. The long-term trajectories are obtained by zipping overlapping

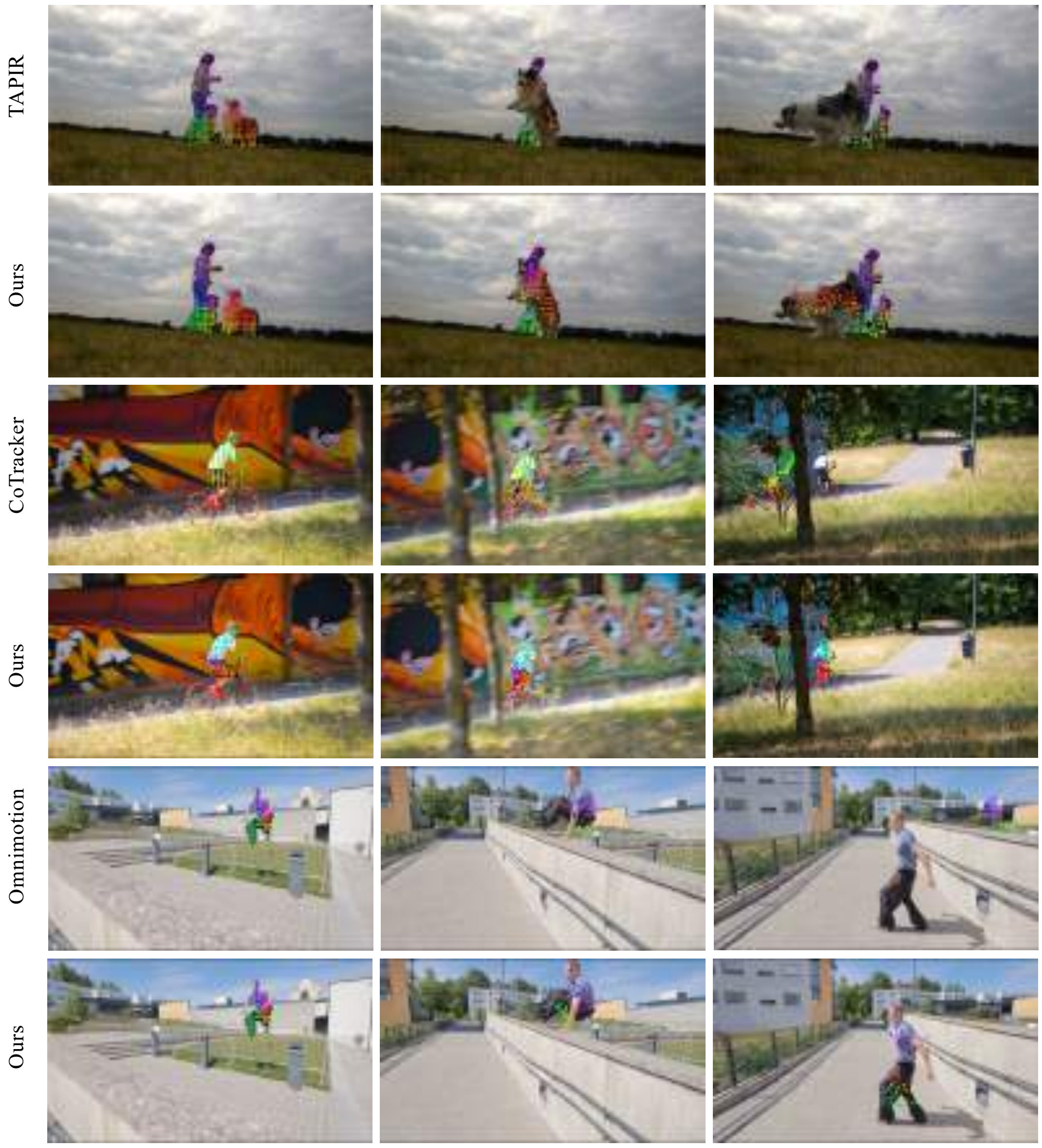

Fig. 5: We compare the tracking performance our method with TAPIR [9], Cotracker [14] and OmniMotion [34] on DAVIS scenes *dogs-jump, bmx-trees*, and *parkour* from top to bottom. The leftmost column shows the initial query points. Our method performs better on these scenes than the other method.

windows. **2) TAP-Net** [8] is a simple baseline that computes correspondence by directly querying the feature cost volume of a pretrained visual backbone. **3) TAPIR** [9] initializes a trajectory using an exhaustive global matching process and refines the point location, occlusion, and uncertainty iteratively with local features. **4) CoTracker** [14] is the state-of-the-art long-term tracking method. Cotracker updates several trajectories jointly by computing cross-track/time attention, allowing trajectory prediction with a global receptive field over all tracked points. **5) OmniMotion** [34] is a test-time optimization method that optimizes point correspondences using a set of invertible mapping functions between each frame to canonical space. The underlying representation is an optimizable NeRF volume.

Quantitative comparisons We present the quantitative evaluation results in Tab. 1. Our method achieves the best temporal coherence among all methods on DAVIS and has better position precision than other optimizable methods, which

Table 1: Quantitative comparison of our method and baselines. We categorized all methods into two categories: the feedforward methods which first train a network and then inference trajectories on testing videos, and the optimization-based methods which fuse pairs of pixel correspondence into trajectories for each testing scene without a pre-trained tracking network.

Method		DAVIS				RGB-Stacking			
		AJ↑	δ^x_{avg} ↑	OA↑	TC ↓	AJ ↑	δ^x_{avg} ↑	OA↑	TC ↓
Feed-forward	PIPs [11]	39.9	56.0	81.3	1.78	37.3	50.6	89.7	0.84
	Flow-Walk [3]	35.2	51.4	80.6	**0.90**	41.3	55.7	92.2	**0.13**
	MFT [24]	56.1	70.8	86.9	-	-	-	-	-
	TAP-Net [8]	38.4	53.4	81.4	10.82	61.3	73.7	91.5	1.52
	TAPIR [9]	59.8	72.3	87.6	-	**66.2**	77.4	**93.3**	-
	CoTracker [14]	**65.1**	**79.0**	**89.4**	0.93	65.9	**80.4**	85.4	0.14
Opti-mization	Connect RAFT [33]	30.7	46.6	80.2	0.93	42.0	56.4	91.5	0.18
	Deformable Sprites [40]	20.6	32.9	69.7	2.07	45.0	58.3	84.0	0.99
	OmniMotion [34]	51.7	67.5	85.3	0.74	**77.5**	87.0	93.5	**0.13**
	Ours	**59.4**	**77.4**	**85.9**	**0.68**	75.4	**87.1**	**93.6**	0.15

is comparable with other feed-forward methods. On RGB-stacking our method performs better than other feed-forward methods.

Compared to pure flow-based optimization approaches, our method achieves significantly better precision and temporal coherence on complex motions over the real scene dataset. Our method incorporates long-term supervision with short-term ones, which simultaneously corrects the global trajectory coarsely and refines the detailed motion locally. Compared with feed-forward approaches, our method achieves better on the textureless synthetic videos. Feature-based methods rely on visual textures to track contrastive points, which are prone to fail when tracking multiple identical points. More analyses are specified in section 4.4.

Without neural rendering for depth or color and equipped with the novel CaDeX++, our method accelerates the convergence more than 10 times faster than OmniMotion on DAVIS and 5 times faster on RGB-stacking approximately as shown in Fig. 6. We conduct the experiments on NVIDIA V100 GPUs.

Qualitative comparison Fig. 5 reveals that compared with baselines, our method can track points against long-term occlusion. Our method can also handle complex object motion and large camera motion.

4.3 Ablation Study

We perform ablations to verify our design decisions listed in Tab. 2 on a subset of the DAVIS [26] dataset. *No depth* indicates replacing the optimizable depth maps with photometric neural rendering to predict depth. *No long-term* is a version that excludes long-term supervision in the training dataset. *No CaDeX++* is the model that downgrades the local invertible mapping into the baseline global MLP ones.

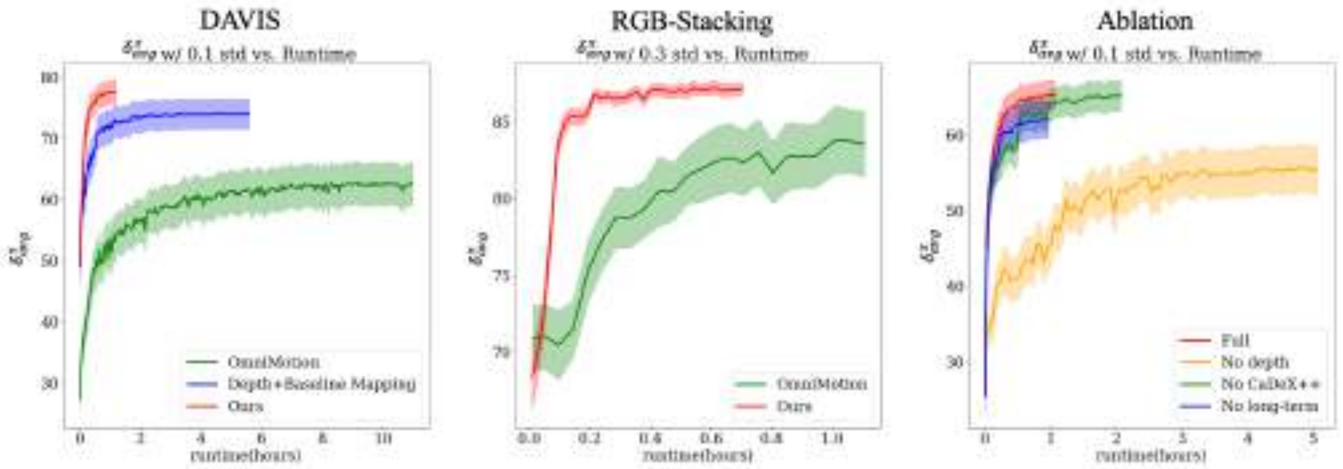

Fig. 6: Runtime Comparisons for DAVIS, subset of RGB-Stacking, and ablation experiments

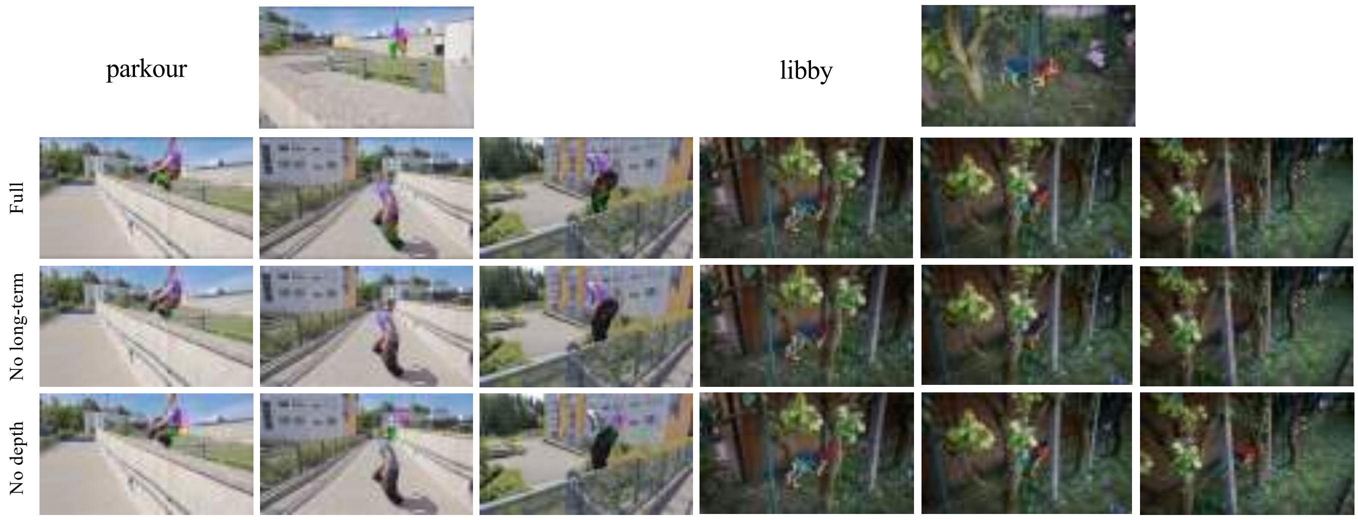

Fig. 7: Qualitative comparison of ablation configurations.

As shown in Tab. 2 and Fig. 6, the introduction of the optimizable depth maps significantly improves the tracking precision and converging speed by leveraging ordinal information from the depth priors to cluster depth semantically. Long-term supervision enhances the trajectory precision considerably and CaDeX++ accelerates convergence speed.

Qualitative results demonstrated in Fig. 7 prove that the introduction of the depth prior makes the tracking of points within the same instance more concentrated and less prone to dispersion. Besides, without long-term supervision, our method fails to handle large and frequent occlusions across time.

4.4 Further Comparison with CoTracker and OmniMotion

CoTracker [14] As a learned method, Cotraker works well when the expectation of the learned distribution aligns with that of the testing distribution. Nevertheless, this is not always the case when evaluating videos that have not been previously seen. In Fig. 8, we show several cases where Cotracker fails. In Fig. 8-Left, when the frame rate is low and significant relative motion exists, we observe that the pixels representing the ground are inaccurately tracked as moving along with the vehicle. In Fig. 8-Right, we demonstrate that the background points, despite having rich textures, are still inaccurately tracked.

Table 2: Ablation study on a subset of DAVIS. We ablate loss two loss terms and CaDex++ architecture.

Method	AJ ↑	δ^x_{avg} ↑	OA ↑	TC ↓
No depth	42.0	56.8	73.3	1.42
No long-term	45.6	61.3	75.5	1.32
No CaDeX++	48.2	65.4	80.1	**0.97**
Full	**48.6**	**65.7**	**80.1**	1.14

Table 3: Disagreement between tracking trajectory and optical flow. Lower is better, indicating better consistency.

Method	DAG↓	
	car-turn	plane
CoTracker	40.3	32.5
Ours	**14.9**	**12.8**

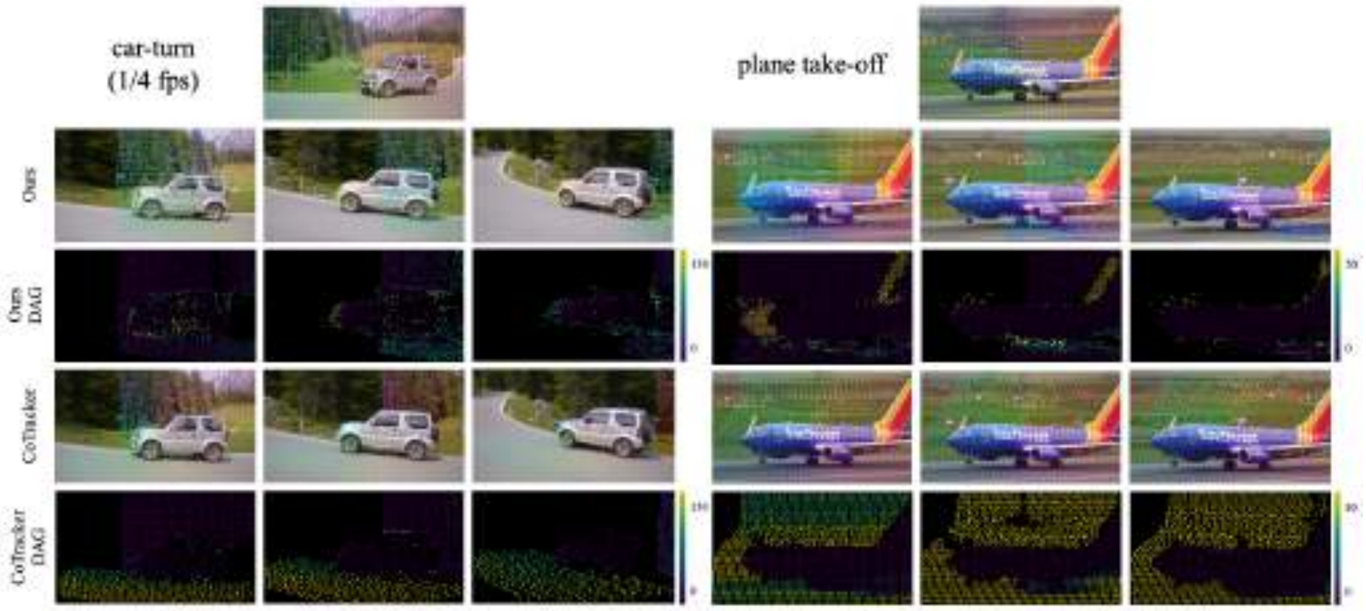

Fig. 8: Failure case of CoTracker and visualization of DAG. We track both the foreground and the background pixels. The error map shows the error magnitude of all trajectory points, where bright yellow equals a large error and dark purple equals a small error.

The background failure instances are not adequately represented in the DAVIS benchmark because the ground-truth points are labeled as foreground majorly. In these two sequences, we observe that the local optical flow is significantly more accurate than long-term tracks. Therefore, we further quantitatively measure these failures by computing the average disagreement with the trajectory and optical flow by

$$DAG = \frac{1}{|P|} \sum_{(p_i,p_j)\in P} ||(p_j - p_i) - f_{i\rightarrow j}(p_i))||_2, \; j - i = 1 \tag{12}$$

where P is the set of all visible trajectory points, (p_i, p_j) is the two adjacent points on a trajectory and $f_{i\rightarrow j}$ is the flow computed between frame i, j. We report these accuracies in Tab 3. In this case, the less the disagreement, the more precise the track is. We observe that ours still tracks reasonably well. In contrast, Cotraker is not able to predict accurate point tracks in both cases.

OmniMotion [34] We further verify one of our important arguments of robustness. When the network is optimized on the same scene with different random

Table 4: Comparison of convergence robustness of OmniMotion [34] and ours.

Method	$\delta^x_{avg} \uparrow$							
	motocross-jump				libby			
	min	max	mean	std	min	max	mean	std
Omnimotion	4.7	60.5	26.3	26.1	2.3	18.0	8.86	5.9
Ours w/o depth	4.4	65.5	44.3	23.5	1.8	20.2	12.7	6.6
Ours	75.2	76.4	75.6	0.5	40.1	48.5	45.7	3.0

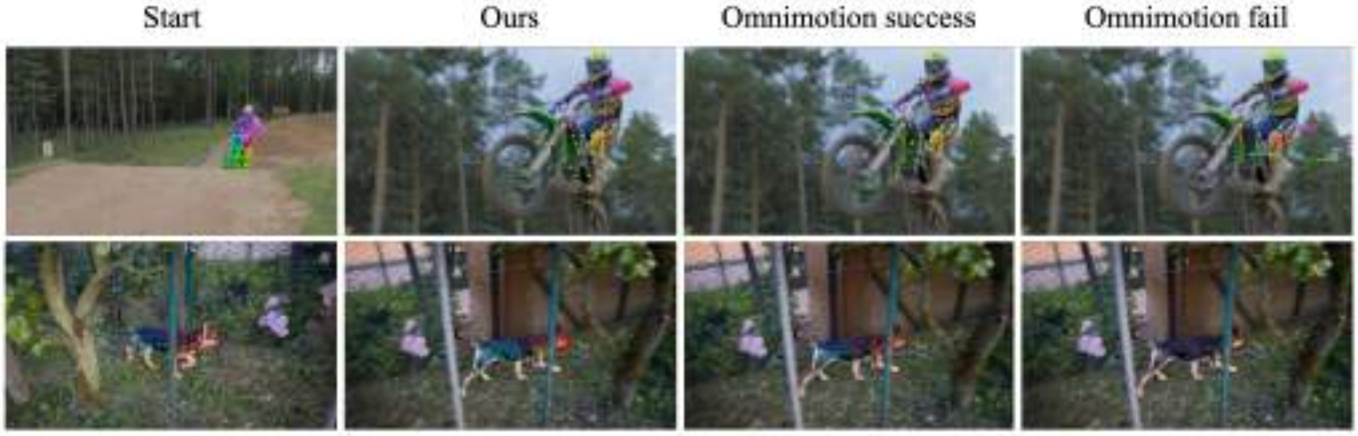

Fig. 9: Robustness: running OmniMotion [34] with different random seeds will result in highly variant fitting results shown on the right while ours is stable.

seeds, OmniMotion [34] often results in fitting errors with high variance, as shown in Tab. 4 and Fig. 9. In contrast, our approach demonstrates stability even with varying random seeds, as illustrated in Table 4. Our robustness is primarily attributed to the incorporation of the Depth prior (Sec. 3.3), which acts as a regularization technique and restricts the optimization space. This decision is supported by the findings in Tab. 4 when we ablate the impact of removing the depth from our model.

5 Conclusion

We present a novel approach to computing the long-term trajectories of pixels from a video. Our approach aims to maximize computational efficiency and robustness, which are key shortcomings of the previous work. By proposing a novel invertible block with local grid, we boost the expressivity of the mapping functions. Additionally, we take advantage of recent foundational models and bootstrap long-term semantic consistency with short-term flow consistency. Our model achieves state-of-the-art performance in optimizing the tracking test time, while significantly reducing computational time by 90%. Compared with Omnimotion, the previous SoTA, our method tracks everything everywhere faster and more robsustly.

Q9

Acknowledgement We gratefully acknowledge the financial support through the NSF IIS-RI 2212433 grant, and a gift from AWS AI to Penn Engineering's ASSET Center for Trustworthy AI.

References

1. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (surf). Computer vision and image understanding **110**(3), 346–359 (2008)
2. Bhat, S.F., Birkl, R., Wofk, D., Wonka, P., Müller, M.: Zoedepth: Zero-shot transfer by combining relative and metric depth. arXiv preprint arXiv:2302.12288 (2023)
3. Bian, Z., Jabri, A., Efros, A.A., Owens, A.: Learning pixel trajectories with multiscale contrastive random walks. In: CVPR. pp. 6508–6519 (2022)
4. Birkl, R., Wofk, D., Müller, M.: Midas v3. 1–a model zoo for robust monocular relative depth estimation. arXiv preprint arXiv:2307.14460 (2023)
5. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. In: European Conference on Computer Vision. pp. 333–350. Springer (2022)
6. DeTone, D., Malisiewicz, T., Rabinovich, A.: Superpoint: Self-supervised interest point detection and description. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops. pp. 224–236 (2018)
7. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real nvp. arXiv preprint arXiv:1605.08803 (2016)
8. Doersch, C., Gupta, A., Markeeva, L., Recasens, A., Smaira, L., Aytar, Y., Carreira, J., Zisserman, A., Yang, Y.: Tap-vid: A benchmark for tracking any point in a video. Advances in Neural Information Processing Systems **35**, 13610–13626 (2022)
9. Doersch, C., Yang, Y., Vecerik, M., Gokay, D., Gupta, A., Aytar, Y., Carreira, J., Zisserman, A.: Tapir: Tracking any point with per-frame initialization and temporal refinement. arXiv preprint arXiv:2306.08637 (2023)
10. Guizilini, V., Vasiljevic, I., Chen, D., Ambruș, R., Gaidon, A.: Towards zero-shot scale-aware monocular depth estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9233–9243 (October 2023)
11. Harley, A.W., Fang, Z., Fragkiadaki, K.: Particle video revisited: Tracking through occlusions using point trajectories. In: European Conference on Computer Vision. pp. 59–75. Springer (2022)
12. Horn, B.K., Schunck, B.G.: Determining optical flow. Artificial intelligence **17**(1-3), 185–203 (1981)
13. Jiang, S., Campbell, D., Lu, Y., Li, H., Hartley, R.: Learning to estimate hidden motions with global motion aggregation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9772–9781 (2021)
14. Karaev, N., Rocco, I., Graham, B., Neverova, N., Vedaldi, A., Rupprecht, C.: Cotracker: It is better to track together. arXiv preprint arXiv:2307.07635 (2023)
15. Ke, B., Obukhov, A., Huang, S., Metzger, N., Daudt, R.C., Schindler, K.: Repurposing diffusion-based image generators for monocular depth estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
16. Lee, A.X., Devin, C.M., Zhou, Y., Lampe, T., Bousmalis, K., Springenberg, J.T., Byravan, A., Abdolmaleki, A., Gileadi, N., Khosid, D., et al.: Beyond pick-and-place: Tackling robotic stacking of diverse shapes. In: 5th Annual Conference on Robot Learning (2021)
17. Lei, J., Daniilidis, K.: Cadex: Learning canonical deformation coordinate space for dynamic surface representation via neural homeomorphism. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6624–6634 (2022)
18. Li, W.: Superglue-based deep learning method for image matching from multiple viewpoints. In: Proceedings of the 2023 8th International Conference on Mathematics and Artificial Intelligence. pp. 53–58 (2023)

19. Liu, L., Gu, J., Lin, K.Z., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. NeurIPS (2020)
20. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**, 91–110 (2004)
21. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI'81: 7th international joint conference on Artificial intelligence. vol. 2, pp. 674–679 (1981)
22. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
23. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (ToG) **41**(4), 1–15 (2022)
24. Neoral, M., Šerỳch, J., Matas, J.: Mft: Long-term tracking of every pixel. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 6837–6847 (2024)
25. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al.: Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193 (2023)
26. Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., Van Gool, L.: The 2017 davis challenge on video object segmentation. arXiv preprint arXiv:1704.00675 (2017)
27. Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F.: D-nerf: Neural radiance fields for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10318–10327 (2021)
28. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: Orb: An efficient alternative to sift or surf. In: 2011 International conference on computer vision. pp. 2564–2571. Ieee (2011)
29. Sand, P., Teller, S.: Particle video: Long-range motion estimation using point trajectories. International journal of computer vision **80**, 72–91 (2008)
30. Shi, J., Tomasi, C.: Good features to track. In: Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on. pp. 593–600. IEEE (1994)
31. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8934–8943 (2018)
32. Sun, J., Shen, Z., Wang, Y., Bao, H., Zhou, X.: Loftr: Detector-free local feature matching with transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8922–8931 (2021)
33. Teed, Z., Deng, J.: 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. pp. 402–419. Springer (2020)
34. Wang, Q., Chang, Y.Y., Cai, R., Li, Z., Hariharan, B., Holynski, A., Snavely, N.: Tracking everything everywhere all at once. arXiv preprint arXiv:2306.05422 (2023)
35. Wang, X., Jabri, A., Efros, A.A.: Learning correspondence from the cycle-consistency of time. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2566–2576 (2019)
36. Xian, K., Shen, C., Cao, Z., Lu, H., Xiao, Y., Li, R., Luo, Z.: Monocular relative depth perception with web stereo data supervision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)

37. Xu, H., Yang, J., Cai, J., Zhang, J., Tong, X.: High-resolution optical flow from 1d attention and correlation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10498–10507 (2021)
38. Xu, H., Zhang, J., Cai, J., Rezatofighi, H., Tao, D.: Gmflow: Learning optical flow via global matching. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8121–8130 (2022)
39. Yang, L., Kang, B., Huang, Z., Xu, X., Feng, J., Zhao, H.: Depth anything: Unleashing the power of large-scale unlabeled data. In: CVPR (2024)
40. Ye, V., Li, Z., Tucker, R., Kanazawa, A., Snavely, N.: Deformable sprites for unsupervised video decomposition. In: CVPR. pp. 2657–2666 (2022)
41. Zhang, F., Woodford, O.J., Prisacariu, V.A., Torr, P.H.: Separable flow: Learning motion cost volumes for optical flow estimation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 10807–10817 (2021)
42. Zhang, M.L., Wu, L.: Lift: Multi-label learning with label-specific features. IEEE transactions on pattern analysis and machine intelligence **37**(1), 107–120 (2014)
43. Zheng, Y., Harley, A.W., Shen, B., Wetzstein, G., Guibas, L.J.: Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19855–19865 (2023)

Appendix

A CaDeX++

We implemented the temporal feature grid $\Psi_l(i)$ in three resolutions: $T/20$, $T/4$, and $13T/20$, where T is the number of frames. Each resolution has a feature dimension of 16. For the spatial feature grid $\Phi_l(x, y)$, we implemented 2 resolutions 12 and 96, with feature dimensions 32 for each resolution. 2 hidden layers are set for the tiny MLP. We perform ablation studies on DAVIS [26] scenes: breakdance, bmx-trees, libby, parkour, and blackswan.

The tiny MLP predicts the positive incremental bias of the control points as $[(\Delta\alpha^1, \Delta\beta^1)...(\alpha^B, \Delta\beta^B)]$ together with the positive outlier slope k_l, k_r. We divide the incremental bias into two sets $\{(\Delta\alpha_N^i, \Delta\beta_N^i)\}_{i=1}^{B/2}$ and $\{(\Delta\alpha_P^i, \Delta\beta_P^i)\}_{i=1}^{B/2}$ to generate the control points with negative and positive α values. For the control points with negative α values, their coordinates are computed as:

$$(\alpha_N^k, \beta_N^k) = -(\sum_{i=1}^{k} \Delta\alpha_N^i, \sum_{i=1}^{k} \Delta\beta_N^i) \tag{13}$$

While the control points with positive α values are aggregated as:

$$(\alpha_P^k, \beta_P^k) = (\sum_{i=1}^{k} \Delta\alpha_P^i, \sum_{i=1}^{k} \Delta\beta_P^i) \tag{14}$$

For the input that lies outside the left-most or right-most control point (α_m, β_m), we compute the output as:

$$z' = k_m(z - \alpha_m) + \beta_m \tag{15}$$

where k_m is the outlier slope.

B Preparing Long-term Correspondence

During training, we sample flow for each query frame among a neighbourhood of 12 frames, and search long-term correspondence outside a neighborhood of 10 frames. Coarse correspondences are computed on the low-resolution feature maps of DINOv2 [25]. We applied three strong filters to remove noisy and keep representative matches.

- **Mutual Maximum**. For a matched pair (p_i, p_j) of two frames F_i, F_j, the best matching of p_i in frame F_j should be p_j and vice versa:

  $$\operatorname*{argmax}_{p_i \in F_i} S\langle \operatorname*{argmax}_{p_j \in F_j} S\langle p_k, p_j\rangle, p_i\rangle = p_k,\ p_k \in F_i \tag{16}$$

  where $S\langle p_i, p_j\rangle$ denotes the cosine similarity between the feature of points p_i, p_j. We only choose the pairs that have similarity over $\theta_m = 0.75$.
- **Background Filter**. For a point p_k in a matched pair, we compute the similarity between p_k with all other points in its feature map. Then we count the number of similar points beyond a threshold of θ_s. We keep the points that have less then N_s similar points. We set $\theta_s = 0.55$ and $N_s = 100$.

  $$\sum_{p_i \in F_i} \mathbf{1}(S\langle p_k, p_k\rangle > \theta_s) < N_s \tag{17}$$

- **Local Noise Filter**. For a point p_k in a matched pair, we compute the similarity among its 11 × 11 neighbor points $M(p_k)$ and sum up all the similarity. We choose the points with total local similarity larger than $\theta_l = 30$.

  $$\sum_{p_i \in M(p_k)} S\langle p_i, p_k\rangle > \theta_l \tag{18}$$

C Optimization Based on CoTraker

We utilizes the output of CoTracker as part of our training supervision for each scene. The optimization result on DAVIS dataset is shown in Tab. 5.

Table 5: Result of optimization on DAVIS with CoTracker output.

Method	DAVIS [26]			
	AJ↑	δ^x_{avg} ↑	OA↑	TC ↓
CoTracker [14]	**65.1**	79.0	**89.4**	0.93
Ours	62.2	**80.0**	86.8	**0.69**

D Limitation and Future Research

Like other optimization-based methods, the efficacy of our tracking performance is dominated by the precision and quality of the input depth and the pixel correspondence.

Moreover, current network architecture primarily addresses 2D pixel tracking task. It is imperative to investigate its potential capabilities in other tasks, including 3D reconstruction, object pose estimation, and content generation.