FlowTrack: Revisiting Optical Flow for Long-Range Dense Tracking

Seokju Cho[1] Jiahui Huang[2] Seungryong Kim[1] Joon-Young Lee[2]

[1]Korea University [2]Adobe Research

Abstract

In the domain of video tracking, existing methods often grapple with a trade-off between spatial density and temporal range. Current approaches in dense optical flow estimators excel in providing spatially dense tracking but are limited to short temporal spans. Conversely, recent advancements in long-range trackers offer extended temporal coverage but at the cost of spatial sparsity. This paper introduces FlowTrack, a novel framework designed to bridge this gap. FlowTrack combines the strengths of both paradigms by 1) chaining confident flow predictions to maximize efficiency and 2) automatically switching to an error compensation module in instances of flow prediction inaccuracies. This dual strategy not only offers efficient dense tracking over extended temporal spans but also ensures robustness against error accumulations and occlusions, common pitfalls of naive flow chaining. Furthermore, we demonstrate that chained flow itself can serve as an effective guide for an error compensation module, even for occluded points. Our framework achieves state-of-the-art accuracy for long-range tracking on the DAVIS dataset, and renders 50% speed-up when performing dense tracking.

1. Introduction

Understanding the intricate dynamics of objects as they move, deform, and occlude within video sequences has been a longstanding challenge in the field of computer vision [45]. This challenge is significantly amplified when it comes to long-range understanding [3, 13, 23, 25], where the temporal and spatial extents of these interactions extend over periods and distances.

In addressing the intricate challenge of *dense and long-range tracking*, a recent development, OmniMotion [41] approaches this by finding dense trajectories through the optimization of dynamic neural fields [24]. While OmniMotion successfully generates reliable dense trajectories, it demands extensive optimization for each video and often encounters training instability, especially in complex video sequences. Recent developments in point tracking methods [5, 6, 11, 20] offer an alternative for long-range dense tracking, by processing densely sampled query points. Yet, this requires exhaustive processing of each point across the entire video sequence. Also, these point tracking methods often disregard the spatial context [6, 11], leading to spatial incoherence.

In pursuit of a practical approach for dense and long-range tracking, we revisit optical flow [7, 15, 17, 37, 43].

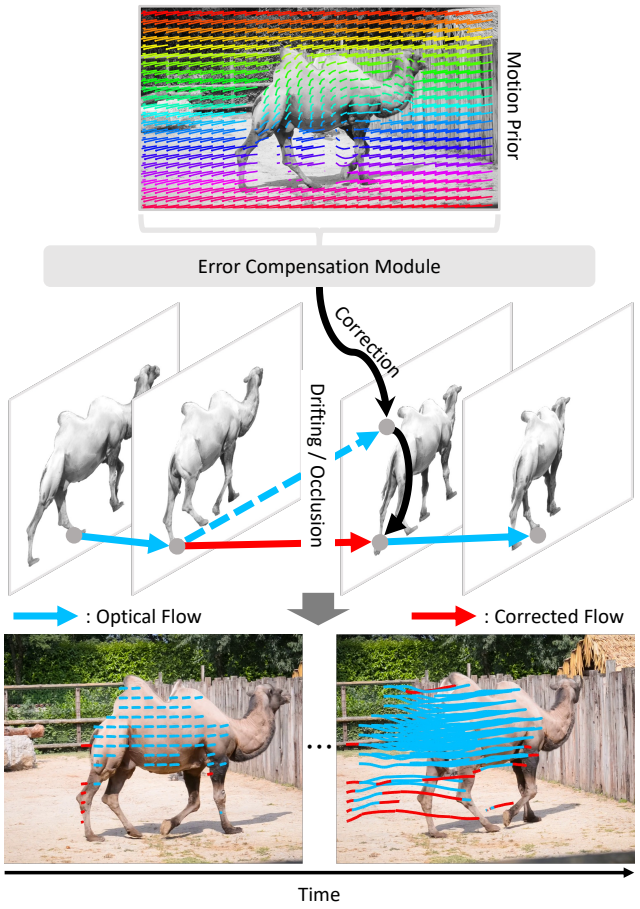


Figure 1. **We revisit optical flow to tackle dense long-range tracking problem.** We address the drifting and occlusion issues in optical flow using our error compensation module. This module rectifies the error in the flow by leveraging confident motion extracted through flow chaining.

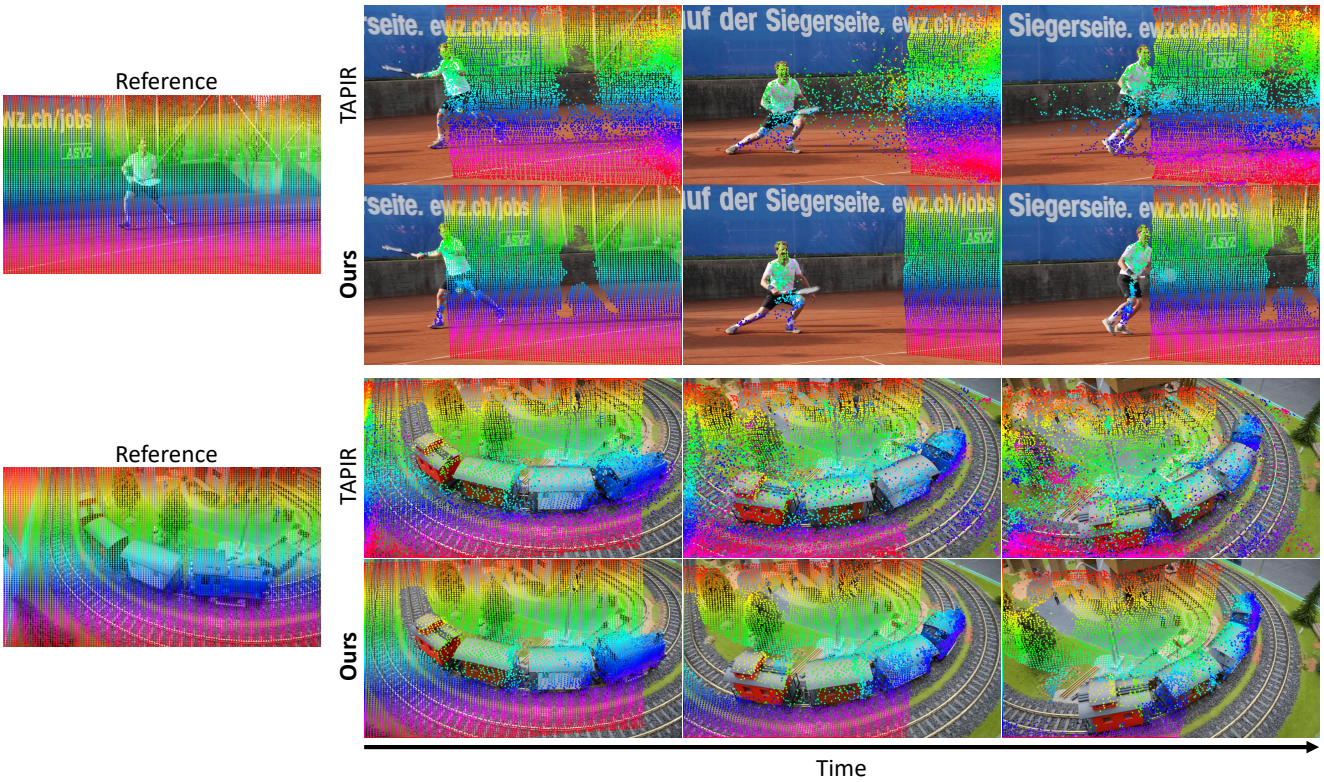


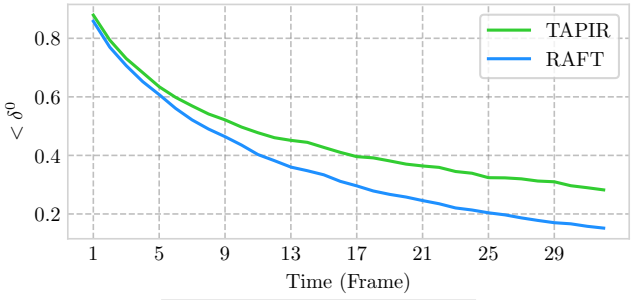
Figure 2. **Visualization of dense long-range tracking.** Our method successfully predicts *dense long-range tracking* with manageable computation and offers better spatial smoothness compared to point-wise tracking methods [6]. Best viewed in color.

Optical flow is a well-established problem in computer vision that focuses on tracking every point across consecutive video frames. Despite significant advancements in dense optical flow, their effectiveness in long-range tracking remains underexplored. Although we can easily extend optical flow to long-range by sequentially chaining flow estimates, this straightforward extension encounters two significant challenges. First, small errors in flow estimation can accumulate over time, leading to trajectory drift. Second, in situations where the point of interest becomes occluded or the trajectory has drifted, it is challenging to recover it to the correct tracking path. Despite these drawbacks, we found dense optical flow exhibits exceptional accuracy in short sequences, comparable to state-of-the-art point tracking methods [6]. Furthermore, the use of optical flow offers a significant computational advantage, as the cost of tracking a single point becomes trivial once the flow has been computed. This raises a compelling question: Can we extend off-the-shelf dense optical flow estimators to achieve long-range tracking by harnessing their powerful short-term tracking capabilities and computational efficiency?

In this context, we present an error compensation module designed to address the challenges of optical flow in the long-range tracking task, which is illustrated in Fig. 1. This module is activated to correct the error whenever the framework detects unconfident flow predictions. Here, we leverage the *spatio-temporal motion prior* of readily obtained optical flow. We observe that a confident trajectory obtained from flow can serve as an effective signal to compensate for the error of unconfident flow. The correlation of motion and the spatial smoothness present in the confident motion allows us to discern the co-movement of objects [4, 8] directly from the optical flow. This enables inference of movement in occluded areas by referencing the movement of nearby objects or the background. Additionally, the spatial smoothness of optical flow is beneficial in processing areas with minimal texture or repetitive patterns [39]. It aids in maintaining spatial consistency during error correction, yielding consistent dense prediction, as shown in Fig. 2. From these insights, we extract motion priors from optical flow and process them through a motion encoder. The processed motion is then used as an input to an error compensation module.

In summary, our contributions are as follows:

- We revisit optical flow methods for *long-range tracking*, and introduce a framework that effectively leverages their powerful short-term tracking capabilities and efficiency.
- We introduce an error compensation module that addresses the challenges of trajectory drift and occlusion from optical flow.
- We demonstrate that our framework yields significant improvements on long-range tracking tasks in terms of both accuracy and speed.



	TAPIR	RAFT
Infer. Time (s)	177.6	6.0

Figure 3. **Optical flow estimator vs. long-range tracker.** (top) Within a short time span and in the absence of occlusion, RAFT [37], an off-the-shelf optical flow estimator, achieves comparable performance against TAPIR [6], a long-range tracker. The scores are measured on the TAPVid-DAVIS dataset [5]. (bottom) When measuring inference time for extracting dense trajectories, RAFT is approximately $\times 30$ faster than TAPIR. Inference time is measured in *horsejump-high* video in DAVIS.

2. Related Work

Optical flow. Optical flow is a fundamental problem in the field of computer vision that aims to estimate correspondence between adjacent frames in videos [16, 17, 36]. RAFT [37] is a seminal work that iteratively refines the flow estimate by processing the local cost volume. Recent methods employ Transformer [15] architecture or Masked Autoencoder [35] to find better feature representation upon the iterative refinement paradigm. Although these methods can predict precise displacement on two nearby frames, they suffer from predicting long-range trajectories. Recent AccFlow [42] explores long-range optical flow with a backward accumulation strategy in synthetic setups, while our method focuses on real-world scenarios with more severe occlusions. Multi-frame optical flow methods [9, 27, 32, 34] explore to improve flow prediction by leveraging multiple nearby frames. Our work is complementary to multi-frame optical flow methods, and advancements in this field can be directly integrated into our framework.

Long-term point tracking. PIPs [11] suggests predicting each trajectory independently while leveraging long-range context, distinct from optical flow methods that predict dense two-frame displacements for all pixels. Similarly, TAPIR [6] predicts long-term trajectory given a good initial prediction from TAPNet [5]. However, these works overlook the spatial correlation between the points, which could give a strong prior for dense tracking. Concurrently, CoTracker [20] leverages the spatial relations with the Transformer architecture. However, they track additional points for each query point to introduce spatial redundancy, thus requiring a large amount of extra computation and hindering its applicability. Other distinct works [28, 41] aim to predict long-term dense trajectory. MFT [28] finds flow with the distant frames and chooses the most confident flow within the candidates, which may inherit the problem of optical flow methods on handling distant frames. OmniMotion [41], which optimizes a dynamic neural field for each video [24], suffers from unstable and slow training, requiring hours per video.

In this paper, we bridge the gap between short-range optical flow and long-range point tracking. We develop a method for *long-range dense tracking* that achieves high-precision tracking over extended periods while maintaining computational tractability.

3. Method

3.1. Motivation and Preliminaries

We propose a novel framework that revisits the effectiveness of dense optical flow estimators for long-range point tracking in videos. While the most simple extension for applying optical flow to long-range tracking involves chaining adjacent flow predictions, this approach often leads to error accumulation and struggles to handle occlusions effectively. Meanwhile, we recognize that dense flow estimators offer fast and accurate tracking over short sequences, as illustrated in Fig. 3. To leverage this advantage, we selectively use the flow estimators for intervals where it remains accurate and effectively switch to an error compensation module, whenever the flow prediction is inaccurate. Additionally, we found that confident motion extracted from optical flow can aid in correcting the flow itself. In this regard, we introduce a motion prior into the error correction module, which helps the model to understand motion structure and local-global spatial similarities. Fig. 4 illustrates the error compensation module.

For the long-range tracking, we start with a query point, denoted as q, in the first frame of a video. The objective is to predict its trajectory $\mathcal{T} = \{p_t\}_{t=1}^{T}$ across the video frames $\mathcal{I} = \{I_t\}_{t=1}^{T}$, where p_t denotes a point in frame t, $I_t \in \mathbb{R}^{H \times W \times 3}$ represents a frame at time t, and T is the total number of frames. Additionally, we predict the visibility status $\mathcal{V} = \{v_t\}_{t=1}^{T}$ of the tracked point, where v represents probability for visibility. For simplicity, we assume a single query point is given in the first frame, and this can be easily generalized to dense tracking by densely sampling the point.

3.2. Long-Range Tracking with Flow Chaining

Given an off-the-shelf dense optical flow [15, 37, 43] model $\mathcal{F}(\cdot)$, we compute the forward and backward flows between adjacent frames as follows:

$$F_t = \mathcal{F}(I_t, I_{t+1}), \quad F_t^{\text{rev}} = \mathcal{F}(I_{t+1}, I_t), \tag{1}$$

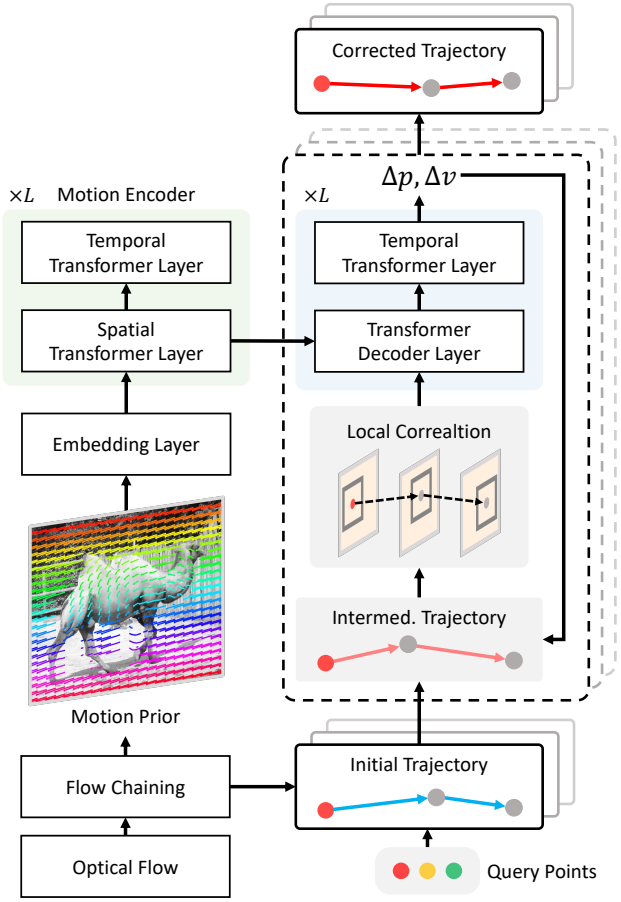


Figure 4. **Overview of the error compensation module.**

where $F_t \in \mathbb{R}^{H\times W\times 2}$ and $F_t^{\text{rev}} \in \mathbb{R}^{H\times W\times 2}$ represent the forward and backward flows, respectively. We utilize these flows to calculate the trajectory of the query point by chaining the flows as follows:

$$\begin{aligned} p_1 &= q, \\ p_{t+1} &= \text{Interp}(F_t, p_t) + p_t, \end{aligned} \tag{2}$$

where $\text{Interp}(\cdot)$ denotes a sampling process using bicubic interpolation, which computes an interpolated value based on the position of a given point.

Flow filtering. To address occlusion and drifting, we first filter inaccurate predictions using cycle-consistency check [18], which is done by comparing the forward and backward flows. This process can be defined as follows:

$$v_{t+1}^{\text{cycle}} = \mathbb{1}\left[\text{Interp}(F_{t+1}^{\text{rev}}, p_{t+1}) + \text{Interp}(F_t, p_t) < \tau_{\text{cycle}}\right]. \tag{3}$$

Here, $\mathbb{1}[\cdot]$ denotes an indicator function, and τ_{cycle} represents the cycle consistency threshold. Additionally, we compare the features of the predicted point to the features of the query point to enhance robustness against drifting:

$$\begin{aligned} D_q &= \text{Interp}(\mathcal{E}(I_1), q), \\ D_{t+1} &= \text{Interp}(\mathcal{E}(I_{t+1}), p_{t+1}), \\ v_{t+1} &= v_t * v_{t+1}^{\text{cycle}} * \mathbb{1}\left[\frac{D_{t+1}\cdot D_q}{\|D_{t+1}\|\|D_q\|} > \tau_{\text{feat}}\right]. \end{aligned} \tag{4}$$

In the above equations, $\mathcal{E}(\cdot)$ denotes a feature encoder, D_q and D_{t+1} represent feature vectors for q and p_{t+1}, respectively. By iterating Eq. 2 and Eq. 4 from $t = 1$ to $T-1$, we obtain the trajectory and occlusion information. This chaining process can be formally represented as:

$$\mathcal{T}, \mathcal{V} = \text{FlowChain}(q, \{I_t\}_{t=1}^{T}), \tag{5}$$

Flow chaining efficiently produces a dense trajectory within the video once the flow has been extracted. However, it lacks the ability to recover from errors, as it heavily relies on accurate prediction of the previous flow predictions.

3.3. Error Compensation Module

To address the limitations of the flow chaining process, we introduce a module designed to recover from errors by directly comparing the features of the query point to the features of the current frame image. Similar to RAFT [37], we iteratively update the initial prediction p obtained from the flow chaining by predicting updates Δp and Δv, refining the intermediate estimate with $p^{k+1} = p^k + \Delta p$ and $v^{k+1} = v^k + \Delta v$. To estimate the update, we first construct a local correlation map between the query point and local neighborhood of predicted position from flow chaining, defined as follows:

$$\mathcal{N}(p) = \{p + \delta \mid \delta \in \mathbb{Z}^2, \|\delta\|_\infty \leq r\}. \tag{6}$$

Here, r represents the radius of the local neighborhood. We then calculate the correlation map by performing a dot product between the feature of the query point and the features extracted from the neighborhood $\mathcal{N}(p)$. To capture hierarchical information, we create a set of correlation maps from a 4-layered pyramidal feature map [37] and these correlation maps are flattened and stacked to form $\mathcal{C}(p)$, which is then fed into a model predicting the error residuals for the erroneously predicted point and its visibility status:

$$\Delta p, \Delta v = \Phi(p, \mathcal{C}(p)), \tag{7}$$

where Φ represents a learnable function.

Although this formulation allows for error compensation, the correlation map may contain noise and ambiguity, especially in the presence of repetitive patterns or textureless areas [33]. To address this ambiguity, we leverage the temporal smoothness of the trajectory. Instead of updating

Figure 5. **Visualization of attention map for motion prior.** We visualize the attention map of visible and occluded points attending to motion priors. Even when a point is occluded, motion priors from optical flow can guide the reasoning for the occluded point.

Δp and Δv on one frame at a time, we update them on multiple frames together in a window:

$$\{\Delta p_t\}_{t\in\mathcal{W}}, \{\Delta v_t\}_{t\in\mathcal{W}} = \Phi\left(\{[p_t, \mathcal{C}(p_t)]\}_{t\in\mathcal{W}}\right), \quad (8)$$

where $[\cdot]$ indicates concatenation operation and $\mathcal{W}$ denotes the temporal window. The model iteratively updates the initial prediction from the erroneous flow chaining result with the predicted residuals, iterating K times, effectively compensating for drifting or occlusion errors.

In practice, we implement $\Phi(\cdot)$ using a Transformer architecture [40]. Also, in addition to p and $\mathcal{C}(p)$, we provide the feature of the query point and the embedded relative position within the local temporal window $\mathcal{W}$ [20]. All these embeddings are combined and aggregated across the temporal dimension within the Transformer.

3.4. Leveraging Spatio-Temporal Motion Prior from Optical Flow

With optical flow information readily available, we can capture local motion patterns at almost no additional computational cost. By incorporating spatial context into our model, we enable it to leverage objectness [8] and spatial smoothness [19], such as the co-movement of points belonging to the same object, which can be seen in Fig. 5. This enhances the training process by providing valuable information for occluded points and aiding the model in understanding local-global similarities among tracks.

Motion extraction with optical flow. We begin by extracting trajectories within a temporal window $\mathcal{W}$, using optical flow chaining. Our trajectory estimation starts from the first frame of the temporal window, and we sample initial positions using a regular grid with a specified stride. This process can be formally defined as follows:

$$\mathcal{G} = \{(x, y) \mid x = [0 : W : s], y = [0 : H : s]\},$$
$$\mathcal{T}_\mathcal{W}, \mathcal{V}_\mathcal{W} = \text{FlowChain}(\mathcal{G}, \{I_t\}_{t\in\mathcal{W}}), \quad (9)$$

where s denotes stride and $\mathcal{T}_\mathcal{W} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{W}|\times 2}, \mathcal{V}_\mathcal{W} \in \mathbb{R}^{|\mathcal{G}|\times|\mathcal{W}|}$ represent the arrays of trajectories and their visibility, respectively. When training, we use randomly sampled N_s points rather than a regular grid for regularization. Subsequently, we process these trajectories using a specially designed motion encoder.

Motion encoder. The purpose of the motion encoder is to identify coarse motion structures and capture spatial smoothness that can aid in tracking. To effectively uncover its motion structure, we employ the Transformer architecture [40]. It's worth noting that the trajectory has both spatial and temporal dimensions. Therefore, we decompose the process into a spatial transformer layer and a temporal transformer layer similar to [1, 2, 25], as follows:

$$\mathbf{x} = \text{Embedding}([\mathcal{T}_\mathcal{W}, \mathcal{V}_\mathcal{W}])$$
$$\mathbf{x}'_{i+1} = \mathcal{E}_i^{\text{spat}}\left(\mathcal{E}_i^{\text{temp}}\left(\mathbf{x}_\text{i}\right)\right). \quad (10)$$

Here, Embedding represents an embedding layer that projects the input to the dimensions of the transformer layer, and $\mathbf{x}$ and $\mathbf{x}'$ represent intermediate results. Along with $\mathcal{T}_\mathcal{W}$, we also add visibility status obtained from flow chaining. The spurious trajectories lacking valid motion information can mislead the model's motion reasoning [38], thus visibility status can guide the model to decide the *what motion to trust*. In practice, to provide better contextual information, we additionally add features corresponding to the points $\mathcal{G}$ and the similarity between a feature from $\mathcal{G}$ and a feature from position $\mathcal{T}$ to the input. We then alternately apply $\mathcal{E}^{\text{temp}}(\cdot)$ and $\mathcal{E}^{\text{spat}}(\cdot)$ L times to the embedding, resulting in a set of intermediate embeddings $\mathcal{X}_\mathcal{W} = \{\mathbf{x}'_i\}_{i=1}^{L}$. Finally, we condition an error compensation module Φ with the extracted motion embeddings $\mathcal{X}_\mathcal{W}$.

Decoding motion prior on error compensation module. We condition the error compensation module Φ on the motion embeddings $\mathcal{X}_\mathcal{W}$ using cross-attention [40]. Specifically, we modify the architecture of Φ similarly to a Transformer decoder. The decoder consists of two alternating components: a self-attention layer and a cross-attention layer, where self-attention layer models temporal interactions, while the cross-attention layer learns to leverage spatial priors learned from the motion encoder. We redefine Eq. 8 as follows:

$$\{\Delta p_t\}_{t\in\mathcal{W}}, \{\Delta v_t\}_{t\in\mathcal{W}} = \Phi\left(\{[p_t, \mathcal{C}(p_t)]\}_{t\in\mathcal{W}}, \mathcal{X}\right). \quad (11)$$

Note that $\mathcal{X}$ is calculated only once and shared across all iterations, leading to efficiency. Finally, we correct the error in flow chaining by progressively improving it through the predicted update outlined in Eq. 11.

Algorithm 1: FlowTrack.

Input: A video sequence $\{I_1, \dots, I_T\}$; Extracted forward flow $\{F_1, \dots F_{T-1}\}$ and backward flow $\{F_1^{\text{rev}}, \dots F_{T-1}^{\text{rev}}\}$; Query set $\mathcal{Q}$; Strided grid $\mathcal{G}$;

Output: Trajectory $\mathcal{T}$ and visibility $\mathcal{V}$ of the points.

begin
- Initialization: $\mathcal{T}, \mathcal{V} \leftarrow \emptyset$
- **for** $t = 1 : T : \lfloor \frac{|\mathcal{W}|}{2} \rfloor$ **do**
  - $\mathcal{W} \leftarrow \{t, \dots, t + |\mathcal{W}|\}$
  - `/* Extract motion prior */`
  - $\mathcal{T}_\mathcal{W}, \mathcal{V}_\mathcal{W} \leftarrow \text{FlowChain}(\mathcal{G}, \{I_t\}_{t \in \mathcal{W}})$
  - $\mathcal{X}_\mathcal{W} \leftarrow \text{MotionEncoder}(\mathcal{T}_\mathcal{W}, \mathcal{V}_\mathcal{W})$
  - `/* In practice, multiple queries are processed concurrently. */`
  - **for** $q \in Q$ **do**
    - $\mathcal{T}', \mathcal{V}' \leftarrow \text{FlowChain}(q, \{I_t\}_{t \in \mathcal{W}})$
    - Find confident track $\mathcal{T}_{\text{conf}} \subset \mathcal{T}', \mathcal{V}_{\text{conf}} \subset \mathcal{V}'$
    - Update: $\mathcal{T}, \mathcal{V}$ with $\mathcal{T}_{\text{conf}}, \mathcal{V}_{\text{conf}}$
    - $\mathcal{T}_{\text{u}}, \mathcal{V}_{\text{u}} \leftarrow \mathcal{T} \setminus \mathcal{T}_{\text{conf}}, \mathcal{V} \setminus \mathcal{V}_{\text{conf}}$
    - `/* Iterative refinement */`
    - **for** $i = 1, \dots, K$ **do**
      - $\Delta p, \Delta v \leftarrow \Phi(\mathcal{T}_{\text{u}}, \mathcal{V}_{\text{u}}; \mathcal{X}_\mathcal{W})$
      - Update: $\mathcal{T}_{\text{u}}, \mathcal{V}_{\text{u}}$ with $\Delta p, \Delta v$
    - Update: $\mathcal{T}, \mathcal{V}$ with $\mathcal{T}_{\text{u}}, \mathcal{V}_{\text{u}}$

3.5. Integrating Flow Chaining and Error Compensation Module

Once we have corrected the errors in flow chaining using the proposed error compensation module, we switch back to flow chaining and continue tracking until the next occlusions occur, as depicted in Fig. 1. Finally, we generalize this framework to dense tracking by sharing the motion encoder across the multiple dense queries, as illustrated in Fig. 4. However, extending this strategy to a setup where multiple query points are provided introduces challenges in parallel processing of motion encoder, as the points may become occluded at different times for each query. To address this problem, we synchronize the temporal window $\mathcal{W}$ across the query points. We compute the trajectory within the window, then slide the window by a stride of $\lfloor \frac{|\mathcal{W}|}{2} \rfloor$, and iterate this process until reaching the end of the video. The overall process, dubbed *FlowTrack*, is described in Alg. 1.

4. Experiments

4.1. Evaluation Dataset

In our study, we utilize the TAP-Vid [5] benchmark to test our method, a benchmark specifically created for assessing tracking performance in long video sequences. This benchmark encompasses a mix of real and synthetic videos, with the real videos having accurately annotated point tracks and the synthetic ones having perfect ground-truth tracks. The DAVIS [30] dataset features 30 real video clips, including challenges such as large scale change. The Kinetics [21] dataset consists of 1,189 real videos from YouTube, which includes various real-world difficulties such as severe motion blur or scene cuts. The RGB-Stacking [22] dataset contains a series of 50 synthetic videos, each 250 frames long, with homogeneous backgrounds, making it difficult to find accurate correspondences.

We employ several evaluation metrics in line with the TAP-Vid benchmark to assess both the position and occlusion accuracy of our predicted tracks. Average Position Accuracy ($< \delta^x_{avg}$) calculates the precision of visible points across thresholds of 1, 2, 4, 8, and 16 pixels. Occlusion Accuracy (OA) gauges the correctness of model's visibility predictions for each frame. Average Jaccard (AJ) is a dual measure of occlusion and position accuracy. Temporal Coherence (TC) measures the $L2$ distance between the ground-truth acceleration and the acceleration of the predicted trajectory [41].

4.2. Implementation Details

We train our model with batch size of 8, on 8 RTX 3090 GPUs. We use the Kubric MOVi-F dataset [10] as a training dataset. We train our model for 100,000 iterations with AdamW [26] optimizer, with a learning rate of $5 \cdot 10^{-4}$ and linear decay learning rate scheduler. We train our model with $L1$ loss for trajectory and binary cross entropy loss for visibility. Along with the visibility status, we also predict uncertainty to determine whether the error of the predicted position is below a certain threshold [6]. This prediction is then used to adjust the visibility status for inference, following Doersch et al. [6]. We use a modified RAFT [37] that uses 4-stride feature map for extracting optical flow. We use window size $|\mathcal{W}| = 8$, motion sampling stride $s = 8$, and the number of sampled motion $N_s = 256$. We use $\tau_{\text{cycle}} = 0.2$, $\tau_{\text{feat}} = 0.2$, local correlation radius $r = 3$, and the number of layer $L = 6$. When training, we randomly sample 256 query points. We utilize a modified ResNet [12] as our feature encoder $\mathcal{E}$, where the outputs of all blocks are resized and concatenated and then projected onto a 128-channel dimension. For the error compensation module, we set the number of iterations K to 4. All inference times were measured on a single RTX 3090 GPU.

4.3. Main Results

As shown in Tab. 1, we evaluate FlowTrack on the TAP-Vid [5] dataset. We primarily focus on the TAPVid-DAVIS dataset, as it contains real-world scenes without scene cuts, which are most relevant for tracking applications. Overall, our method shows constant performance gain upon RAFT, highlighting the effectiveness of our error compensation module. On the DAVIS [30] dataset, we achieve the highest scores across nearly all metrics for both 256×256 and 384×512 resolutions. Specifically, FlowTrack exhibits su-

Method	DAVIS AJ ↑	DAVIS $< \delta^x_{avg}$ ↑	DAVIS OA ↑	DAVIS TC ↓	Kinetics AJ ↑	Kinetics $< \delta^x_{avg}$ ↑	Kinetics OA ↑	Kinetics TC ↓	RGB-Stacking AJ ↑	RGB-Stacking $< \delta^x_{avg}$ ↑	RGB-Stacking OA ↑	RGB-Stacking TC ↓
384×512 Input Resolution												
PIPs [11]	42.0	59.4	82.1	1.78	35.3	54.8	77.4	-	37.3	51.0	91.6	-
CoTracker [20]	64.8	79.1	**88.7**	0.92	-	-	-	-	-	-	-	-
FlowTrack (Ours)	**66.0**	**79.8**	87.2	**0.50**	-	-	-	-	-	-	-	-
256×256 Input Resolution												
COTR [18]	35.4	51.3	80.2	-	19.0	38.8	57.4	-	6.8	13.5	79.1	-
RAFT [37]	30.0	46.3	79.6	0.93	34.5	52.5	79.7	-	44.0	58.6	90.4	-
TAP-Net [5]	38.4	53.1	82.3	10.82	46.6	60.9	85.0	-	59.9	72.8	90.4	-
OmniMotion [41]	51.7	67.5	85.3	0.74	55.1*	69.6*	89.6*	-	**77.5**	**87.0**	93.5	-
TAPIR [6]	61.3	73.6	88.8	1.01	**57.2**	**70.1**	87.8	-	62.7	74.6	91.6	-
FlowTrack (Ours)	**63.2**	**76.3**	**89.2**	**0.53**	56.4	68.9	**88.0**	0.80	66.2	76.9	**94.3**	0.09

Table 1. **Comparison to prior works on TAPVid.** The best scores are highlighted in **bold**, while the second-best are denoted with underline. *: The scores are evaluated in the subset of the dataset.

	Method	DAVIS AJ	DAVIS $< \delta^x_{avg}$	DAVIS OA
(I)	Full model	**63.2**	**76.3**	**89.2**
(II)	**(I)** - Motion Prior	60.5	73.8	88.7
(III)	**(II)** - Matching Module	38.2	56.8	59.5

Table 2. **Ablation of main components.** We conduct an ablation study by gradually excluding components to our full model.

Methods	1 point	10 points	10^2 points	10^3 points	10^4 points	10^5 points
OmniMotion [41]	>36,000	>36,000	>36,000	>36,000	>36,000	>36,000
PIPs [11]	4.61	39.50	389.04	3869.67	>36,000	>36,000
CoTracker-Single† [20]	8.44	77.56	769.29	7492.02	>36,000	>36,000
TAPIR [6]	**0.55**	**0.55**	**0.56**	**2.92**	28.13	284.69
FlowTrack (Ours)	6.67	6.97	7.01	8.09	**19.81**	**145.37**

Table 3. **Inference time (s) comparison on varying number of query points.** All the inference times are measured using the 50-frame *horsejump-high* sequence in the DAVIS dataset [29]. We indicate times exceeding 10 hours as $> 36,000$. †: Infer one point at a time, which is the setup used for the main evaluation.

perior temporal coherence (TC), which we attribute to the inherent smoothness of optical flow. This also implies that our error correction module produces temporally smooth predictions. Also, our method demonstrates competitive performance on the TAP-Vid-Kinetics dataset containing videos with various disruptive effects, such as scene cuts and caption-only sequences. While our approach falls short of the state-of-the-art [6] on this dataset, we hypothesize that these disruptive effects (*i.e.*, scene cuts) make the motion extracted from optical flow unreliable, hindering accurate trajectory inference. On the synthetic RGB-Stacking dataset, our method surpassed recent point tracking approaches [5, 6, 11], while OmniMotion [41] demonstrated strong performance on the same dataset. We conjecture that the static camera in the RGB-Stacking dataset eases the optimization process for dynamic neural fields [31].

4.4. Ablation Study and Analysis

Ablation on core components. We provide ablation of core components in Tab. 2. We demonstrate the performance of our full model in **(I)**, and progressively exclude components. **(II)** presents our model without the motion prior. The significant performance gap highlights the benefit of leveraging motion extracted from optical flow. This also suggests that confident motion from optical flow can serve as an effective signal for tracking. In **(III)**, we demonstrate the effectiveness of the matching module, which is outlined in Sec. 3.3, where it successfully compensates for the flow error. Note that the score in **(III)** reflects the performance of RAFT baseline, where we use a modified version of RAFT that utilizes a 4-stride feature.

Method	DAVIS AJ	DAVIS $< \delta^x_{avg}$	DAVIS OA
RAFT [37]	**63.2**	**76.3**	**89.2**
FlowFormer [15]	62.4	75.7	88.8
GMFlow [43, 44]	57.7	71.2	87.7
GMFlow* [43, 44]	62.5	75.7	89.0

Table 4. **Ablation on state-of-the-art optical flow models.** *: We apply the method at its training resolutions, since using it at a resolution of 256×256 leads to degraded results.

τ_{cycle}	DAVIS AJ	DAVIS $< \delta^x_{avg}$	DAVIS OA	Flow Ratio	Inference Time (s)
0.1	**63.8**	**76.8**	89.1	34.7	354.47
0.2	63.2	76.3	**89.2**	60.1	219.42
0.3	62.6	75.8	89.0	66.3	188.15
0.5	61.8	75.0	89.0	73.4	149.34
0.7	61.3	74.6	89.1	78.3	122.59
0.9	61.0	74.3	89.1	**80.2**	**113.60**

Table 5. **Analysis of the relationship between cycle consistency threshold and performance and efficiency in dense tracking.** We demonstrate the trade-off between performance and efficiency in relation to the cycle consistency threshold.

Analysis on inference time. In Tab. 3, we present a comparison of inference times. Note that we evaluated these times using the public code released by the respective authors. Our model demonstrates significantly faster inference in dense tracking setups compared to point tracking

Figure 6. **Long-range dense tracking application with edit propagation.** We conduct edit propagation using our method, which results in consistent propagation even in cases of occlusion, while maintaining spatial smoothness.

methods [11, 20], especially when dealing with $> 10^4$ points. This is noteworthy considering that even predicting dense trajectories at 256×256 resolution requires 65,536 inferences, highlighting our model's substantial advantage. However, our model does require the feed-forward processing of a motion encoder, introducing some overhead even for a single feed-forward pass of points. Despite this, we want to emphasize the practicality of our model in dense tracking applications, such as pose estimation [46] or editing propagation [14].

Ablation on optical flow model. We ablate on various optical flow models in Tab. 4. We employ RAFT [37] as a baseline and compare it to FlowFormer [15] and GMFlow [43, 44]. However, we found that GMFlow suffers at 256×256 resolution, thus we present additional scores using the original training resolution. Overall, our model shows similar scores across these optical flow models.

Analysis on cycle consistency threshold τ_{cycle}. We show the trade-off between accuracy and efficiency in dense tracking in Tab. 5. We evaluate performance using the TAPVid-DAVIS dataset. We measure the Flow Ratio and Inference Time as metrics for efficiency in dense prediction. The Flow Ratio denotes the ratio at which optical flow is utilized for tracking; a higher flow ratio leads to more efficient tracking. Our findings reveal that a tighter cycle-consistency threshold necessitates more error compensation, resulting in slower inference times but better accuracy. We balance between performance and efficiency, providing flexibility for real-world applications. For applications requiring high precision, a low cycle-consistency threshold could be a good choice. Conversely, for applications demanding speed, a high threshold could be beneficial.

4.5. Application on Edit Propagation

We visualize application of dense long-range tracking in Fig. 6. Video editing requires spatially smooth propagation; otherwise, the edits would appear scattered. It's noteworthy that our method can smoothly propagate edits even in the presence of occlusions. Additionally, we emphasize that these edits were completed within a minute, demonstrating the practicality of our framework.

5. Conclusion

In this paper, we present a novel framework that revisits and extends optical flow methods for effective long-range, dense tracking. By introducing an innovative error compensation module, we address the primary challenges associated with optical flow, namely trajectory drifting and occlusions. This module harnesses the spatio-temporal motion priors from confident flow estimations to correct inaccuracies in the flow, enabling the module to maintain spatial coherence and track objects accurately over extended periods. Through extensive experiments, we demonstrate that our framework not only mitigates the limitations of optical flow in long-range scenarios but also outperforms recent point tracking methods.

Acknowledgement. This research was supported by the MSIT, Korea (IITP-2024-2020-0-01819, ICT Creative Consilience Program, RS-2023-00227592), and National Research Foundation of Korea (NRF-2021R1A6A1A03045425).

References

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021. 5

[2] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, page 4, 2021. 5

[3] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *European Conference on Computer Vision*, pages 640–658. Springer, 2022. 1

[4] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *Proceedings of the IEEE international conference on computer vision*, pages 686–695, 2017. 2

[5] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adrià Recasens, Lucas Smaira, Yusuf Aytar, João Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022. 1, 3, 6, 7

[6] 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. *arXiv preprint arXiv:2306.08637*, 2023. 1, 2, 3, 6, 7

[7] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015. 1

[8] Katerina Fragkiadaki, Pablo Arbelaez, Panna Felsen, and Jitendra Malik. Learning to segment moving objects in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4083–4090, 2015. 2, 5

[9] Pierre Godet, Alexandre Boulch, Aurélien Plyer, and Guy Le Besnerais. Starflow: A spatiotemporal recurrent cell for lightweight multi-frame optical flow estimation. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2462–2469. IEEE, 2021. 3

[10] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3749–3761, 2022. 6

[11] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision*, pages 59–75. Springer, 2022. 1, 3, 7, 8

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[13] Miran Heo, Sukjun Hwang, Jeongseok Hyun, Hanjung Kim, Seoung Wug Oh, Joon-Young Lee, and Seon Joo Kim. A generalized framework for video instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14623–14632, 2023. 1

[14] Jiahui Huang, Leonid Sigal, Kwang Moo Yi, Oliver Wang, and Joon-Young Lee. Inve: Interactive neural video editing. *arXiv preprint arXiv:2307.07663*, 2023. 8

[15] Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Ka Chun Cheung, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Flowformer: A transformer architecture for optical flow. In *European Conference on Computer Vision*, pages 668–685. Springer, 2022. 1, 3, 7, 8

[16] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8981–8989, 2018. 3

[17] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017. 1, 3

[18] Wei Jiang, Eduard Trulls, Jan Hosang, Andrea Tagliasacchi, and Kwang Moo Yi. Cotr: Correspondence transformer for matching across images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6207–6217, 2021. 4, 7

[19] Rico Jonschkowski, Austin Stone, Jonathan T Barron, Ariel Gordon, Kurt Konolige, and Anelia Angelova. What matters in unsupervised optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 557–572. Springer, 2020. 5

[20] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023. 1, 3, 5, 7, 8

[21] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 6

[22] Alex X Lee, Coline Manon Devin, Yuxiang Zhou, Thomas Lampe, Konstantinos Bousmalis, Jost Tobias Springenberg, Arunkumar Byravan, Abbas Abdolmaleki, Nimrod Gileadi, David Khosid, et al. Beyond pick-and-place: Tackling robotic stacking of diverse shapes. In *5th Annual Conference on Robot Learning*, 2021. 6

[23] Yu Li, Zhuoran Shen, and Ying Shan. Fast video object segmentation using the global context module. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pages 735–750. Springer, 2020. 1

[24] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference*

on Computer Vision and Pattern Recognition, pages 6498–6508, 2021. 1, 3

[25] Yongqing Liang, Xin Li, Navid Jafari, and Jim Chen. Video object segmentation with adaptive feature bank and uncertain-region refinement. *Advances in Neural Information Processing Systems*, 33:3430–3441, 2020. 1, 5

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

[27] Daniel Maurer and Andrés Bruhn. Proflow: Learning to predict optical flow. *arXiv preprint arXiv:1806.00800*, 2018. 3

[28] Michal Neoral, Jonáš Šerỳch, and Jiří Matas. Mft: Long-term tracking of every pixel. *arXiv preprint arXiv:2305.12998*, 2023. 3

[29] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016. 7

[30] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. 6

[31] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 7

[32] Zhile Ren, Orazio Gallo, Deqing Sun, Ming-Hsuan Yang, Erik B Sudderth, and Jan Kautz. A fusion approach for multi-frame optical flow estimation. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2077–2086. IEEE, 2019. 3

[33] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic. Convolutional neural network architecture for geometric matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6148–6157, 2017. 4

[34] Xiaoyu Shi, Zhaoyang Huang, Weikang Bian, Dasong Li, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Videoflow: Exploiting temporal cues for multi-frame optical flow estimation. *arXiv preprint arXiv:2303.08340*, 2023. 3

[35] Xiaoyu Shi, Zhaoyang Huang, Dasong Li, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1599–1610, 2023. 3

[36] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 3

[37] Zachary Teed and Jia Deng. 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*, pages 402–419. Springer, 2020. 1, 3, 4, 6, 7, 8

[38] Prune Truong, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning accurate dense correspondences and when to trust them. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2021. 5

[39] Prune Truong, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning accurate dense correspondences and when to trust them. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5714–5724, 2021. 2

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 5

[41] Qianqian Wang, Yen-Yu Chang, Ruojin Cai, Zhengqi Li, Bharath Hariharan, Aleksander Holynski, and Noah Snavely. Tracking everything everywhere all at once. *arXiv preprint arXiv:2306.05422*, 2023. 1, 3, 6, 7

[42] Guangyang Wu, Xiaohong Liu, Kunming Luo, Xi Liu, Qingqing Zheng, Shuaicheng Liu, Xinyang Jiang, Guangtao Zhai, and Wenyi Wang. Accflow: Backward accumulation for long-range optical flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12119–12128, 2023. 3

[43] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8121–8130, 2022. 1, 3, 7, 8

[44] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, Fisher Yu, Dacheng Tao, and Andreas Geiger. Unifying flow, stereo and depth estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 7, 8

[45] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13–es, 2006. 1

[46] Wang Zhao, Shaohui Liu, Hengkai Guo, Wenping Wang, and Yong-Jin Liu. Particlesfm: Exploiting dense point trajectories for localizing moving cameras in the wild. In *European Conference on Computer Vision*, pages 523–542. Springer, 2022. 8