Dense Optical Tracking: Connecting the Dots

Guillaume Le Moing[1, 2, *] Jean Ponce[2, 3] Cordelia Schmid[1, 2]

[1]Inria [2]Département d'informatique de l'ENS (CNRS, ENS-PSL, Inria) [3]Courant Institute and Center for Data Science New York University

Abstract

*Recent approaches to point tracking are able to recover the trajectory of **any** scene point through a large portion of a video despite the presence of occlusions. They are, however, too slow in practice to track **every** point observed in a single frame in a reasonable amount of time. This paper introduces DOT, a novel, simple and efficient method for solving this problem. It first extracts a small set of tracks from key regions at motion boundaries using an off-the-shelf point tracking algorithm. Given source and target frames, DOT then computes rough initial estimates of a dense flow field and visibility mask through nearest-neighbor interpolation, before refining them using a learnable optical flow estimator that explicitly handles occlusions and can be trained on synthetic data with ground-truth correspondences. We show that DOT is significantly more accurate than current optical flow techniques, outperforms sophisticated "universal" trackers like OmniMotion, and is on par with, or better than, the best point tracking algorithms like CoTracker while being at least two orders of magnitude faster. Quantitative and qualitative experiments with synthetic and real videos validate the promise of the proposed approach. Code, data, and videos showcasing the capabilities of our approach are available in the project webpage.*[1]

1. Introduction

A fine-grain analysis of motion is crucial in many applications involving video data, including frame interpolation [27, 40], inpainting [39, 74], motion segmentation [11, 69], compression [1, 44], future prediction [37, 65], or editing [34, 70]. Historically, most systems designed for these tasks have heavily relied on optical flow algorithms [18, 59, 60] with an inherent lack of robustness to large motions or occlusions [19], confining their use of context to a handful of neighboring frames and limiting long-term reasoning.

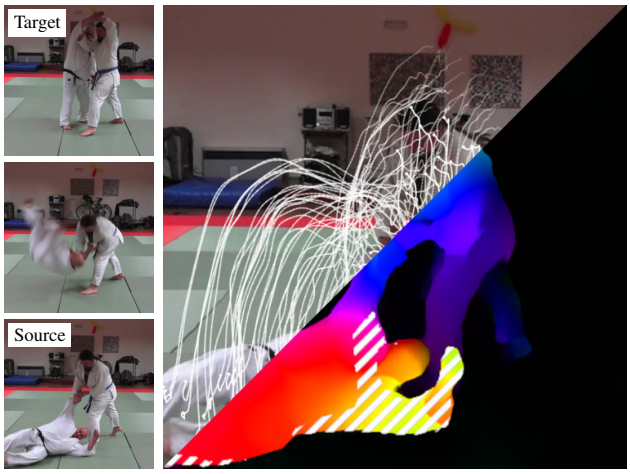

Figure 1. **DOT** unifies point tracking and optical flow techniques. From a few initial tracks, it predicts dense motions and occlusions from source to target frames. We represent tracks in white, occlusions with stripes, and motion directions using distinctive colors.

Lately, point tracking methods [16, 17, 22, 33, 72] have emerged as a promising alternative. They are, in most cases, able to track **any** specific point through a large portion of a video, even in the presence of occlusions, successfully retaining more than 50% of initial queries after thousands of frames [72]. Yet, these methods remain too slow and memory intensive to track **every** individual point in a video, that is, to obtain a set of tracks dense enough to cover every pixel location at every time step. This computational hurdle has thus far limited the widespread adoption of point tracking as a viable replacement for optical flow in downstream tasks.

The inefficiency of point tracking methods [16, 17, 22, 72] arises from their independent processing of individual tracks. In a recent study [33], Karaev *et al*. shed light on a related issue: tracking individual queries independently lacks spatial context. Their approach, CoTracker, tracks multiple points together to exploit the correlations among their trajectories. In some cases, performance drops when the number of simultaneous tracks increases too much, thus preventing solutions which are both fast and accurate.

*corresponding author: guillaume.le-moing@inria.fr
[1]https://16lemoing.github.io/dot

In this paper, we take this concept one step further by tracking every point in a frame simultaneously. Our approach, DOT, connects the dots (hence its name) between optical flow and point tracking methods (Figure 1), enjoying the spatial coherence of the former and the temporal consistency of the latter. Our contributions are as follows:

- We introduce DOT, a novel, simple and efficient approach that unifies point tracking and optical flow, using a small set of tracks to predict a dense flow field and a visibility mask between arbitrary frames in a video.
- We extend the CVO benchmark [64] with 500 new videos to enhance the assessment of dense and long-term tracking. The new videos are longer and have a higher frame rate than existing ones, for more challenging motions.
- We use extensive experiments on the CVO and TAP benchmarks [16, 64], with quantitative and qualitative results, to demonstrate that DOT significantly outperforms state-of-the-art optical flow methods and is on par with, or better than, the best point tracking algorithms while being much faster at dense prediction ($\times 100$ speedup).

2. Related work

Optical flow estimation has traditionally been addressed using variational methods [23] to minimize an energy function enforcing spatial smoothness and brightness constancy constraints. Several works have focused on adapting this to long-range motions, including coarse-to-fine warping strategies [2, 48, 51], a quadratic relaxation of the original problem [57], non-local regularization [36, 53], nearest-neighbor fields [3, 4, 10, 24], and methods relying on local feature matching [5, 25, 55, 63, 66].

DOT builds on these classical methods, also refining dense motions from sparse correspondences, *but using point tracking instead of local feature matching*. Recently, supervised approaches trained on synthetic data [7, 18, 21, 47] have emerged as a powerful alternative to variational methods. *DOT also benefits from these advances*.

Optical flow in the deep learning era. FlowNet [18, 28] was the first convolutional neural network used to directly predict optical flow from image pairs. Several classical ideas have been used to enhance accuracy. For instance, SPyNet [54] incorporates a feature pyramid [6], and DCFlow [68] uses 4D cost volumes to compute correlations between every pair of patches across successive frames, similar to [5]. PWC-Net [59] combines the two ideas. SF-Net [73] builds upon sparse-to-dense methods. RAFT [60] uses a recurrent neural network to progressively refine the flow by iterative look-up in the cost volumes. Lately, transformer-based approaches [26, 29, 31, 58, 67] have been used to model long-range dependencies and identify similarities between remote patches in pairs of frames.

Contrary to these methods that only consider pairs of isolated frames, *DOT takes full videos as input*. We show that temporal information helps handling occlusions, enhances robustness in the presence of large displacements, and disentangles the motions of objects with similar visual characteristics, in particular for distant frames.

Optical flow across distant frames has been approached as an interpolation problem given a sparse set of trajectories extracted from a video [20]. Another early attempt [41] involved computing the flow between consecutive frames using the Lucas-Kanade algorithm [45], then deducing long-term motions using forward flow accumulation. Since then, various approaches following the idea of chaining flows estimated between consecutive (or distant) frames have emerged [13, 14, 30, 50, 64]. Among them, AccFlow [64] introduces a backward accumulation technique, more robust to occlusions than forward strategies. MFT [50] identifies the most reliable chain of optical flows by predicting uncertainty and occlusion scores. Iterative methods, such as RAFT and its variants [31, 58, 60], can be employed to warm-start the estimation of optical flow between distant frames. Recently, OmniMotion [62], an optimization technique that relies on a volumetric representation, akin to a dynamic neural radiance field [49], have been proposed to estimate motion across every pixel and every frame in a video. Similar ideas have also been applied to tracking points in 3D in a multi-camera setting [46].

DOT uses a different strategy, relying on off-the-shelf point tracking algorithms to predict dense and long-term motions, much faster than the per-video optimization technique in OmniMotion [62] and less prone to error accumulation than chaining methods like AccFlow [64] or MFT [50].

Point tracking is closely related to optical flow. Given a pixel in a frame, it predicts the position and visibility of the corresponding world surface point in every other frame of the video. This task has gained significant attention recently, particularly with the introduction of the TAP benchmark and the associated TAPNet baseline [16]. Numerous methods have since emerged in this domain. Persistent independent particles (PIPs) [22] is a method that tracks points even under occlusion, drawing inspiration from the concept of particle video [56], which lies midway between optical flow and local feature matching. TAPIR [17] combines the global matching strategy of TAPNet with the refinement step of PIPs. PIPs++ [72] significantly extends the temporal window of PIPs, enhancing its robustness to long occlusion events. In contrast to previous methods which track one query at a time, CoTracker [33] uses a few additional tracks as context, resulting in improved performance.

DOT takes this concept a step further by predicting the motion of all points in a frame simultaneously, making it significantly more efficient (at least $\times 100$ speedup) than point tracking methods applied densely, while matching or surpassing their performance on individual queries.

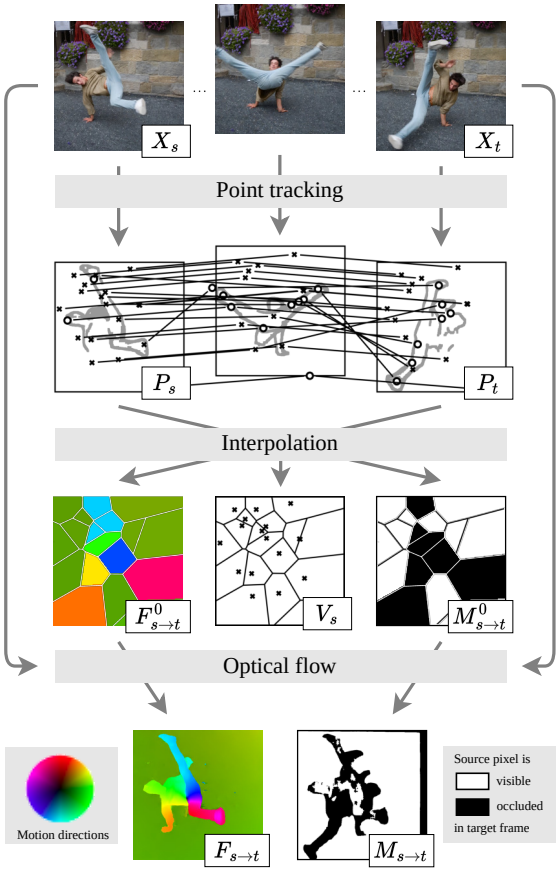

Figure 2. **Dense optical tracking.** Our approach, DOT, takes a video as input and produces dense motion information between any pair of source and target frames X_s / X_t as an optical flow map $F_{s\to t}$ and a visibility mask $M_{s\to t}$. We first track the 2D position and the visibility (×: visible, ○: occluded) of a small set of physical points throughout the video. These are sampled preferably from key regions at motion boundaries (shown in grey). We deduce motion estimates $F^0_{s\to t}$ / $M^0_{s\to t}$ by using all the tracks whose associated point is visible at s, noted V_s, to initialize their nearest neighbors, forming Voronoi cells. We finally refine these estimates with optical flow techniques, using the frames X_s and X_t.

3. Method

Overview. We consider a sequence of RGB frames X_t (t in $1 \dots T$) in $\mathbb{R}^{W\times H\times 3}$. Given source and target time steps (s,t), our goal is to predict for each pixel position (x,y) in the source frame X_s the visibility v (0 if occluded and 1 if visible) and the 2D location $(x+\Delta x, y+\Delta y)$ of the corresponding physical point in the target frame X_t. We represent these *dense* correspondences between source and target as a flow $F_{s\to t}$ in $\mathbb{R}^{W\times H\times 2}$ where $F_{s\to t}(x,y) = (\Delta x, \Delta y)$ and a mask $M_{s\to t}$ in $\{0,1\}^{W\times H}$ where $M_{s\to t}(x,y) = v$.

Our approach to *dense optical tracking* (DOT), illustrated in Figure 2, is composed of the following modules:

- **Point tracking.** To accommodate long-range motions, we first compute N tracks using an off-the-shelf point tracking method [17, 33, 72] on all frames of the video. The number N is kept low (~10^3) compared to the number HW of pixels (~10^5) to limit computation and allow fast inference. A point i in $1 \dots N$ tracked at time t in $1 \dots T$ is denoted by $p^i_t = (x^i_t, y^i_t, v^i_t)$, with position (x^i_t, y^i_t) in $\mathbb{R}^2$, and visibility v^i_t in $\{0,1\}$. We also denote by $P_t = \{p^i_t \mid i \in [1 \dots N]\}$ the set with all the points at t.
- **Interpolation.** Tracks offer *sparse* correspondences between s and t. We deduce initial flow and mask estimates, $F^0_{s\to t}$ and $M^0_{s\to t}$, using nearest-neighbor interpolation: we associate with every pixel in the source frame X_s the nearest track i among those visible at s, noted V_s, and use the position and visibility of the correspondence (p^i_s, p^i_t) to initialize the flow and the mask respectively.
- **Optical flow.** We refine these estimates into final predictions, $F_{s\to t}$ and $M_{s\to t}$, with an optical flow method inspired by RAFT [60] which uses the source and target frames, X_s and X_t, to account for the local geometry of objects. We obtain dense tracks by considering all frames as target for a given source, *e.g.*, setting s to 1 and using t in $\{2 \dots T\}$ to get all tracks from the first frame.

These components and the corresponding training processes are described in more detail in the following paragraphs.

3.1. Overall approach

Point tracking. Not all tracks are equally informative and point tracking is computationally intensive, so we sample tracks more densely in regions undergoing significant motions or likely to become occluded. These are often localized around the edges of moving objects, at the boundary between pixels with significant motions and others which are either just dis-occluded or soon to be occluded. We find these regions by running a pre-trained optical flow model on consecutive frames of the video and then applying a Sobel filter [32] to detect discontinuities in the flow.

Given a budget of N tracks, our sampling strategy consists in initializing half of the tracks randomly near flow edges (up to 5 pixels from an edge) and sampling the remaining ones in the entire image. It is important to emphasize that our approach is agnostic to the choice of a specific point tracking method, so it will also benefit from ongoing advancements in this rapidly evolving research field.

Interpolation. We initialize coarse motion and visibility estimates, $F^0_{s\to t}$ and $M^0_{s\to t}$, between source and target at a reduced spatial resolution $W/P \times H/P$ where P is some constant (P=4 in practice). These are derived from the N input tracks using the nearest visible track for every position (x,y) in $[P, 2P, .., W] \times [P, 2P, .., H]$ in the source frame:

$$\begin{cases} F^0_{s\to t}(x,y) & = (x^i_t, y^i_t) - (x^i_s, y^i_s), \\ M^0_{s\to t}(x,y) & = v^i_t, \end{cases} \tag{1}$$

with $i = \arg\min_{j \in V_s} \|(x_s^j, y_s^j) - (x, y)\|_2$ and where $V_s = \{j \in 1..N \mid v_j^s = 1\}$ designates all the points which are visible at time s. Nearest neighbors of these points form Voronoi cells in the image plane, see Figure 2.

Optical flow. The estimates $F^0_{s\to t}$ and $M^0_{s\to t}$ are inherently imprecise, particularly for points distant from the input tracks. We refine them using an optical flow method.

As in RAFT [60], we extract coarse features Y_s (resp. Y_t) in $\mathbb{R}^{W/P\times H/P\times C}$ for the source (resp. target) frame X_s (resp. X_t) using a convolutional neural network. We then compute the correlation between features for all pairs of source and target positions at different feature resolution levels, yielding a 4D correlation volume for each resolution level. The flow is progressively refined by repeating K times the following procedure (K=4 in practice): Let $F^k_{s\to t}$ be the estimate at the iteration k. For each source position, we sample the correlation volume at different target positions laid on a regular grid centered at the position indicated by $F^k_{s\to t}$. The similarity of each source point with a local neighborhood around its current target correspondence is then fed to a recurrent neural network [12] to predict $F^{k+1}_{s\to t}$. We obtain the flow $\widehat{F}_{s\to t}$ from the final estimate $F^K_{s\to t}$ using the upsampling operation introduced in RAFT [60].

Our model differs from RAFT [60] in that we have a meaningful initialization for the flow $F^0_{s\to t}$ instead of a zero motion. Moreover, we handle occlusions by refining a mask $\widehat{M}_{s\to t}$ from the coarse estimate $M^0_{s\to t}$ along with the flow. We note that the predicted mask $\widehat{M}_{s\to t}$ is a soft estimate which may be turned into a binary one by thresholding with a fixed scalar τ for every position (τ=0.8 in practice).

3.2. Training process

We use an off-the-shelf model for point tracking (*e.g.*, [17, 33, 72]) and freeze the corresponding parameters during training. The interpolation is a parameter-free operation. Therefore, among the three components of our approach, only the optical flow module requires training.

Objectives. The parameters of DOT are optimized by minimizing the sum of two objective functions: a motion reconstruction objective which is the L_1 distance between the predicted and the ground-truth flows, and a visibility prediction objective which is the binary cross entropy between the predicted and the ground-truth masks.

Optimization. DOT is trained on frames at res. 512×512 for 500k steps with the ADAM optimizer [35] and a learning rate of 10^{-4} using 4 NVIDIA V100 GPUs. Given the practical challenge of gathering and storing dense ground truth across different time horizons, *e.g.*, a single flow map represents 262,144 correspondences at this resolution, we only compute the training objectives on a few of these correspondences. Specifically, we train using N=2048 tracks from CoTracker [33] as input (see ablations with other methods in Table 3), and use another N ground-truth tracks (synthetically generated) for supervision. At inference, we adjust N to trade motion prediction quality for speed (Table 4). Further implementation details are presented in Appendix A-B. For reproducibility, code, data and pretrained models are publicly available in our project webpage.

4. Experiments

Optical flow baselines include RAFT [60] and GMA [31], methods which directly predict the motion between pairs of distant frames. We also employ these methods as warm starts, represented by (🔥) in our tables and figures, where the flow between neighboring frames serves as an initialization for estimating the flow between more distant frames. Another strategy, represented as (🔗), is to chain optical flows computed for adjacent frames. We also compare to AccFlow [64], a backward accumulation method for estimating long-range motions, MFT [50], an advanced method for chaining optical flows, and OmniMotion [62], a method which regularizes optical flow by constructing a volumetric representation. All three methods build upon RAFT [60].

Point tracking baselines. We explore the direct extension of sparse methods to predict dense motions by applying them at every pixel. Specifically, we use PIPs [22], PIPs++ [72], TAP-Net [16] and TAPIR [17], all of which track individual point independently. We also consider CoTracker [33], which enables the simultaneous tracking of multiple points in two different modes: The first processes batches of points spread over the image and is hence efficient. The second one, which we denote as (★), is specifically optimized for individual point queries, by incorporating a context of local and global points solely for inference, discarding them afterwards. While the latter improves precision, it does so at the expense of considerably slower processing, making it impractical for dense prediction. Karaev *et al*. use the first version for visualizations and the second for quantitative evaluations [33]. Given their significant differences, we treat them as distinct approaches.

Datasets. Like others, we train and evaluate our method using data generated with Kubric [21], a simulator for realistic rendering of RGB frames along with motion information, featuring scenes with objects falling to the ground and colliding with one another. We consider different datasets:

- **MOVi-F Train.** This set is composed of approximately 10,000 videos, each containing 24 frames rendered at 12 frames per second (FPS). The videos are equipped with point tracks, primarily sampled from objects and, to a lesser extent, from the background. Our method and point tracking baselines [16, 17, 22, 33] train on this data, or close variations thereof. We note that PIPS++ [72] uses data from a different simulator called PointOdyssey, focusing on long videos with naturalistic motion.

Table 1. **Motion prediction on the CVO benchmark.** We evaluate dense predictions between the first and last frames of videos. We report the end point error (EPE) of flows for all pixels (all), visible (vis) and occluded ones (occ), the intersection over union (IoU) of occluded regions, and the average inference time per video (in seconds). We also indicate the number N of tracks for different methods.

	Method	N	CVO (*Clean*)		CVO (*Final*)			CVO (*Extended*)		
			EPE ↓ (all / vis / occ)	IoU ↑	EPE ↓ (all / vis / occ)	IoU ↑	Time* ↓	EPE ↓ (all / vis / occ)	IoU ↑	Time ↓
Optical flow	RAFT [60]	-	2.82 / 1.70 / 8.01	58.1	2.88 / 1.79 / 7.89	57.2	**0.166**	28.6 / 21.6 / 41.0	61.7	**0.166**
	GMA [31]	-	2.90 / 1.91 / 7.63	60.9	2.92 / 1.89 / 7.48	60.1	0.186	30.0 / 22.8 / 42.6	61.5	0.186
	RAFT (🔥) [60]	-	2.48 / 1.40 / 7.42	57.6	2.63 / 1.57 / 7.50	56.7	0.634	21.8 / 15.4 / 33.4	65.0	4.142
	GMA (🔥) [31]	-	2.42 / 1.38 / 7.14	60.5	2.57 / 1.52 / 7.22	59.7	0.708	21.8 / 15.7 / 32.8	65.6	4.796
	MFT [50]	-	2.91 / 1.39 / 9.93	19.4	3.16 / 1.56 / 10.3	19.5	1.350	21.4 / 9.20 / 41.8	37.6	18.69
	AccFlow [64]	-	1.69 / 1.08 / 4.70	48.1	1.73 / 1.15 / 4.63	47.5	0.746	36.7 / 28.1 / 52.9	36.5	5.598
Point tracking	PIPs++ [72]	262144	9.05 / 6.62 / 21.5	33.3	9.49 / 7.06 / 22.0	32.7	974.3	18.4 / 10.0 / 32.1	58.7	1922.
	TAPIR [17]	262144	3.80 / 1.49 / 14.7	73.5	4.19 / 1.86 / 15.3	72.4	131.1	19.8 / 4.74 / 42.5	68.4	848.7
	CoTracker [33]	262144	1.51 / 0.88 / 4.57	75.5	1.52 / 0.93 / 4.38	75.3	173.5	5.20 / 3.84 / 7.70	70.4	1645.
Hybrid	*Dense optical tracking* (DOT)	1024	1.36 / 0.76 / 4.26	80.0	1.43 / 0.85 / 4.29	79.7	0.864	5.28 / 3.78 / 7.71	70.8	5.234
		2048	1.32 / 0.74 / 4.12	80.4	1.38 / 0.82 / 4.10	80.2	1.652	5.07 / 3.67 / 7.34	71.0	9.860
		4096	**1.29 / 0.72 / 4.03**	**80.4**	**1.34 / 0.80 / 3.99**	**80.4**	3.152	**4.98 / 3.59 / 7.17**	**71.1**	19.73

"∗": the time is the same for *Clean* and *Final* sets.

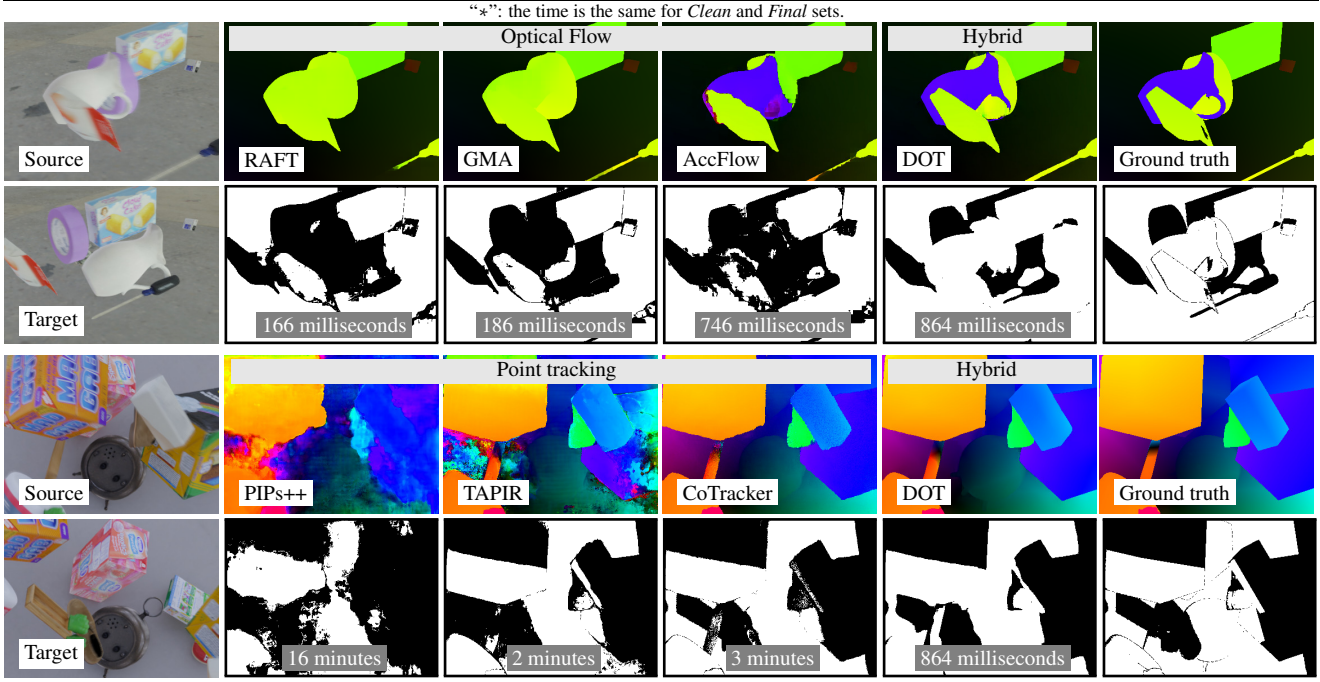

Figure 3. **Qualitative samples on the CVO benchmark.** We show the predicted flow (1st / 3rd rows) and visibility mask (2nd / 4th rows) between the first and last frame of different videos of the *Final* test set. We also report inference time. Optical flow methods produce smooth motion estimates but miss important object regions. Point tracking methods, PIPS++ and TAPIR, are more accurate but tend to produce noisy estimates. CoTracker improves on this aspect by processing multiple point tracks simultaneously instead of one at a time, but we still observe some artifacts when zooming in. DOT combines the benefits of both optical flow and point tracking approaches.

- **CVO Train [64].** This set has around 10,000 videos, with 7 frames rendered at 60 FPS. The videos come with bidirectional optical flow information for adjacent frames and between the first frame and every other frame. Our optical flow baselines [31, 60, 64] use this data for training.
- **CVO Test [64].** There were originally two test sets, *Clean* and *Final*, with the latter incorporating motion blur. Each of them contains around 500 videos of 7 frames at 60 FPS.[2] We also introduce the *Extended* CVO set, with another 500 videos of 48 frames rendered at 24 FPS. This new set is designed to assess longer videos with more challenging motions. All sets provide the optical flow and visibility mask between the first and last frame of videos.

[2]We use a curated version of this dataset since in its original release it contained a few scenes with erroneous optical flows (for 25 videos). Our full data curation pipeline is detailed in Appendix H.

Table 2. **Motion prediction on the TAP benchmark.** We evaluate trajectories spanning the whole video for specific points. We report average jaccard (AJ), position accuracy ($<\delta^x_{\text{avg}}$), and occlusion accuracy (OA). For completeness we include single-point methods optimized for individual test queries, but report their performance separately since they are not strictly comparable when evaluating dense predictions.

	Method	DAVIS (*First*)			DAVIS (*Strided*)			RGB-S. (*First*)			RGB-S. (*Strided*)			Kinetics (*First*)		
		AJ ↑	$<\delta^x_{\text{avg}}$ ↑	OA ↑	AJ ↑	$<\delta^x_{\text{avg}}$ ↑	OA ↑	AJ ↑	$<\delta^x_{\text{avg}}$ ↑	OA ↑	AJ ↑	$<\delta^x_{\text{avg}}$ ↑	OA ↑	AJ ↑	$<\delta^x_{\text{avg}}$ ↑	OA ↑
Single	TAP-Net [16]	33.0	48.6	78.8	38.4	53.1	82.3	53.5	68.1	86.3	59.9	72.8	90.4	38.5	54.4	80.6
Single	PIPs [22]	42.2	64.8	77.7	52.4	70.0	83.6	-	-	-	-	-	-	31.7	53.7	72.9
Single	TAPIR [17]	56.2	70.0	86.5	61.3	73.6	**88.8**	55.5	69.7	88.0	62.7	74.6	91.6	**49.6**	64.2	85.0
Single	CoTracker (★) [33]	**60.6**	**75.4**	**89.3**	**64.8**	**79.1**	88.7	**65.4**	**79.1**	**91.0**	**73.6**	**84.5**	**94.5**	48.7	**64.3**	**86.5**
Dense	RAFT (🔗) [60]	-	-	-	30.0	46.3	79.6	-	-	-	44.0	58.6	90.4	-	-	-
Dense	OmniMotion [62]	-	-	-	51.7	67.5	85.3	-	-	-	77.5	87.0	93.5	-	-	-
Dense	MFT [50]	47.3	66.8	77.8	56.1	70.8	86.9	-	-	-	-	-	-	39.6	60.4	72.7
Dense	CoTracker [33]	56.9	74.1	84.4	61.1	77.4	85.8	67.7	80.7	90.8	74.1	85.2	92.3	44.8	63.2	81.2
Dense	DOT (*ours*)	**60.1**	**74.5**	**89.0**	**65.9**	**79.2**	**90.2**	**77.1**	**87.7**	**93.3**	**83.4**	**91.4**	**95.7**	**48.4**	**63.8**	**85.2**

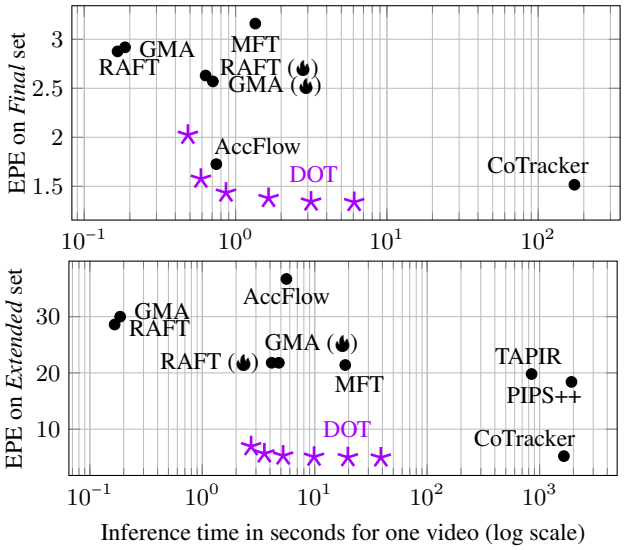

Figure 4. **Performance *vs* speed on the CVO benchmark.** DOT reaches different trade-offs by setting the number N of initial tracks to different values in $[256, 512, 1024, 2048, 4096, 8192]$. We observe that our method improves over all approaches while keeping a speed similar to state-of-the-art optical flow techniques.

We further evaluate DOT and other approaches on the TAP benchmark [16]. It provides ground-truth tracks for various types of scenes, including real-world videos: **DAVIS [52],** with 30 videos (~100 frames each) featuring one salient object; **Kinetics [9],** with over 1,000 videos (250 frames each) representing various human actions; **RGB-Stacking [38],** with 50 synthetic videos (250 frames each) in a robotic environment with textureless objects and frequent occlusions.

Evaluation metrics. We measure computational efficiency and the quality of dense motion predictions between the first and last frames of videos using the following metrics:

- End point error (EPE) between predicted and ground-truth flows. We give mean EPE for all pixels, as well as separately for visible (vis) and occluded (occ) pixels.
- Intersection over union (IoU) between predicted and ground-truth occluded regions in visibility masks.
- Computational efficiency as the average inference time required to produce the mask and flow between the first and last frame of one video using an NVIDIA V100 GPU.

Some methods may exclusively produce flow predictions so we estimate the visibility mask by doing forward-backward consistency checks on the predicted flow. This involves processing videos a second time by flipping their temporal axis.

We follow the standard evaluation protocol for TAP [16] and report occlusion accuracy (OA), position accuracy for visible points averaged over different threshold distances ($<\delta^x_{\text{avg}}$), and average jaccard (AJ) which combines both.

4.1. Comparison with the state of the art

CVO. Results in terms of EPE and IoU on the CVO test sets in Table 1 show that DOT significantly improves over optical flow baselines. In particular, on the *Extended* set, with large motions and long occlusion events, DOT reduces by a factor 4 the EPE and yields relative improvements of more than 8% in IoU compared to these methods. DOT also outperforms point tracking baselines while being at least two orders of magnitude faster. These methods are slow, even when parallelizing computations on GPU, as they are applied to every of the $N = 512 \times 512 = 262144$ pixels in a frame. DOT, in contrast, is flexible as it may trade motion prediction quality for speed by adjusting the number N of initial point tracks. See also Appendix E for visual comparisons by taking different values for N. Figure 4 shows that DOT yields the best possible trade-offs. DOT even improves over input tracks. By extracting 1024 tracks with CoTracker, we obtain 10.6% (resp. 7.1%) relative improvements on EPE (resp. IoU) on the *Final* set on the same tracks after refining them with DOT. Qualitative samples in Figure 3 show the superiority of DOT over prior works.

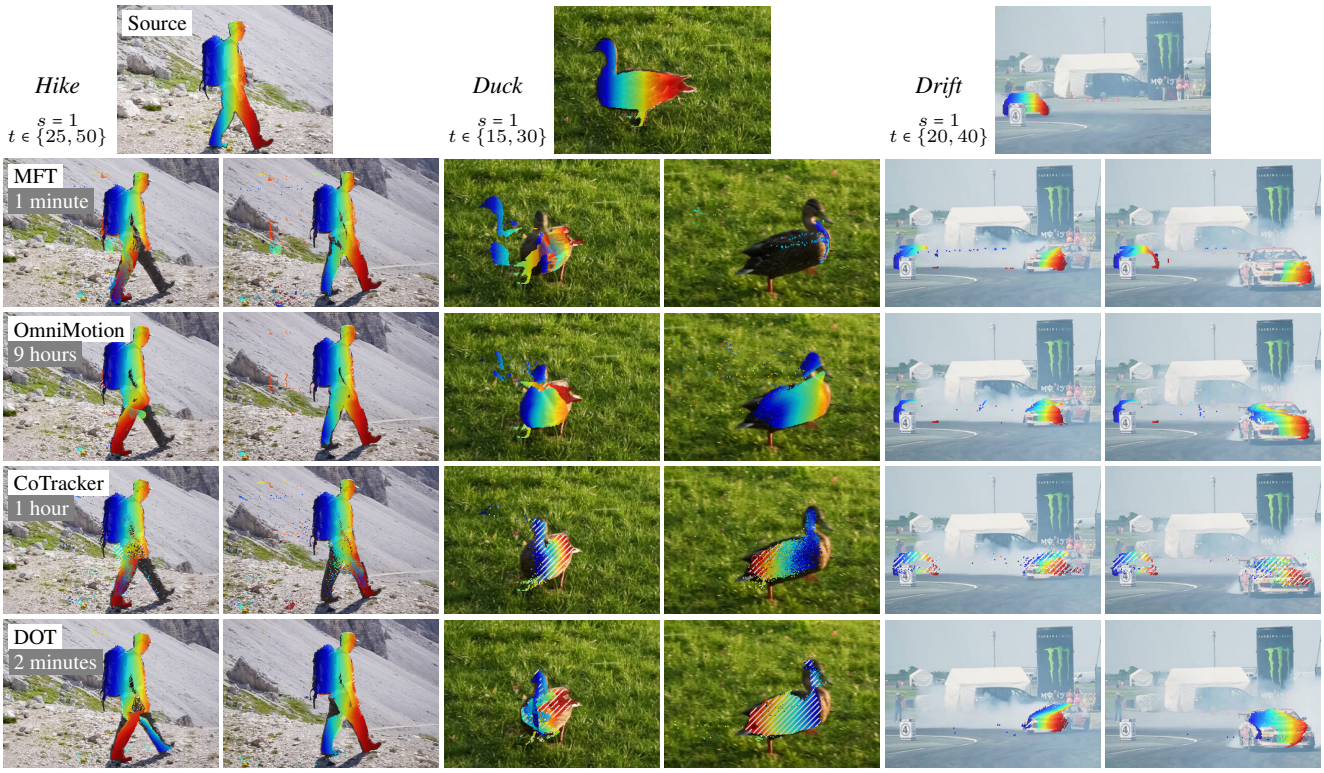

Figure 5. **Qualitative samples on the TAP benchmark.** We compare various methods by tracking all points in the first frame of videos from the DAVIS dataset. Only foreground points are visualized, each depicted with distinctive colors, and overlayed with white stripes when occluded. We also indicate the time for each method to process a 480p video of 50 frames on an NVIDIA V100 GPU. In the "*Hike*" video, our method, DOT, stands out by successfully tracking both legs as the person walks. DOT has robust performance under occlusion, as shown in the "*Duck*" video where the animal changes sides. In contrast, MFT lose sight of the object, showing the limitations of optical flow methods under occlusion. OmniMotion does not account for the rotation of the object. CoTracker successfully tracks the object but fails to predict occlusions, showing the limitations of point tracking methods overly reliant on local features, especially when different parts of an object look similar. DOT handles videos with small objects or atmospheric effects like smoke, like in the "*Drift*" video. Other methods tend to miss object parts in similar conditions. Please zoom in for details and refer to the videos in the supplemental materials.

TAP. DOT predicts dense motions, without knowing which points will be used for testing, as opposed to single-point tracking techniques, optimized for specific test queries. Remarkably, even under this challenging setting, DOT is competitive with the best-performing single-point tracking algorithms across all of the TAP benchmark test sets, see Table 2. DOT even slightly improves over the state of the art on DAVIS (*strided*), and achieves a substantial advantage on RGB-Stacking, with over 13% relative increase in average jaccard (AJ) compared to single-point methods. Its success lies in its ability to handle textureless objects effectively, whose lack of distinctive local features make them challenging for point tracking approaches. The optical flow component in our approach allows us to rely on a broader context, thereby significantly enhancing motion predictions for such objects. Furthermore, DOT performs significantly better than dense methods which, like our method, are not optimized for specific test queries. The relative improvements range from 6% to 15% in average jaccard (AJ), up to 9% in position accuracy (δ^x_{avg}) and up to 5% in occlusion accuracy (OA) compared to the previous state of the art. Qualitative results on real data in Figures 5-6, in Appendix C and G, and videos in our project webpage show the significant improvements achieved by our method in terms of spatial consistency, robustness to occlusions and appearance changes compared to the state of the art.

4.2. Ablation studies

Effect of the method used to extract sparse correspondences. We compare the performance of DOT when different methods are used to extract input correspondences in Table 3. We explore local feature matching techniques which produce correspondences between pairs of images. Although they allow fast computations, such methods may produce incorrect matches when parts of different objects present important similarities [8]. Moreover, they do not handle occlusions, with points required to be visible in both images, and are not very robust to motion blur. We found

Table 3. **Effect of the method used to extract sparse correspondences.** Feature matching is done with LightGlue [42] using different types of local features. We compare the performance of DOT on the CVO (*Final*) test set when fed with N=1024 correspondences extracted with different methods. For fair comparison, we do not do in-domain training (no specialization to any particular input), and we use the same densification model and the same sampling strategy (*i.e.*, the one from Section 3.1) for all methods.

	Method	EPE ↓ (all / vis / occ)	IoU ↑	Time ↓
Feature matching	SIFT [43]	10.9 / 9.15 / 19.5	31.9	1.022
	DISK [61]	10.9 / 8.81 / 21.2	32.9	0.362
	ALIKED [71]	9.55 / 7.17 / 21.0	34.4	0.282
	SuperPoint [15]	8.76 / 6.20 / 21.0	36.7	**0.244**
Point tracking	PIPs++ [72]	7.30 / 4.92 / 19.5	48.0	4.102
	TAPIR [17]	4.00 / 1.82 / 14.3	70.2	0.668
	CoTracker [33]	**1.89 / 1.27 / 4.99**	**70.9**	0.864

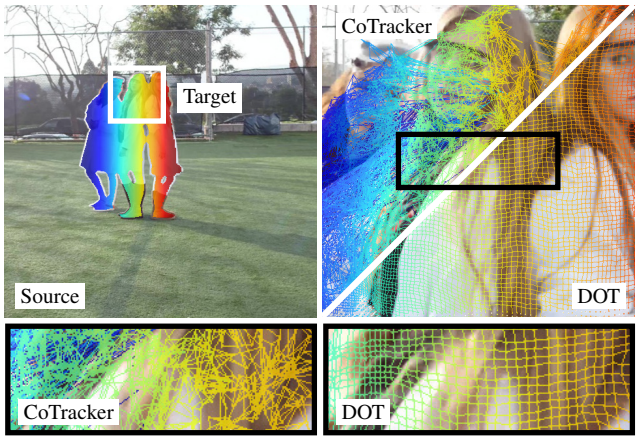

Figure 6. **Preservation of local structure.** We compare CoTracker and DOT on the "*Lab-coat*" video from the DAVIS dataset. We represent foreground points in the source frame (s=1) using distinctive colors, and visualize predicted tracks for the target frame (t=40) with the same colors: we show the mesh obtained by connecting each point at time t to its cardinal neighbors from time s. DOT preserves local structure much better than CoTracker.

that resulting correspondences are not well distributed, *i.e.*, essentially on the background, as illustrated in Appendix F. Conversely, point tracking methods solve this issue by leveraging temporal information, leading to notable improvements. Among these methods, we have opted for the efficient variant of CoTracker [33] since it provides superior motion reconstructions at a reasonable processing speed.

Ablations of the core components of our approach are in Table 4, with one component removed at a time. Using visibility information from tracks slightly improves motion predicition (EPE) and visibility prediction (IoU). We compare randomly sampling tracks to our motion-based strategy and observe that the latter yields more informative tracks.

Table 4. **Ablations of the core components of our approach.** We remove components one at a time and rank them from the least to the most significant in terms of flow and mask reconstruction quality on the CVO (*Final*) test set. We use N=1024 input tracks.

Method	EPE ↓ (all / vis / occ)	IoU ↑
Dense optical tracking (DOT)	**1.43 / 0.85 / 4.29**	**79.7**
- No use of visibility info from tracks	1.46 / 0.87 / 4.31	78.9
- No motion-based sampling of tracks	1.50 / 0.90 / 4.45	79.2
- Patch size of 8 instead of 4	1.60 / 1.02 / 4.45	76.8
- No in-domain training	1.90 / 1.27 / 5.00	70.8
- No track-based estimates	2.88 / 1.79 / 7.89	57.2
- No optical-flow refinement	3.19 / 2.48 / 6.79	54.5

We find that changing the patch size for flow refinement from P=8 to P=4, effectively doubling the resolution of the features, improves performance. A more detailed analysis is in Appendix D. In-domain training, *i.e.*, specializing our densification model to noisy estimates from a specific point tracking model, as opposed to training with ground-truth tracks as input, is also helpful. Moreover, relying solely on the optical flow component of our approach for a pair of source and target frames, without incorporating track-based estimates, results in much worse performance. Similarly, maintaining the same number of input tracks but omitting optical flow refinement does not yield satisfactory results.

5. Conclusion

We have introduced DOT, an approach for dense motion estimation which unifies optical flow and point tracking techniques, effectively leveraging the strengths from both: reaching the accuracy of the latter with the speed and spatial coherence of the former. Like any other approach, it, of course, may fail due to extreme occlusions, fast motions, or rapid changes in appearance. DOT is agnostic to the choice of a specific point tracking algorithm, so future advances in this field will directly benefit our approach. We believe that the efficiency of DOT holds the potential to drive substantial progress across various downstream applications.

Acknowledgements

This work was granted access to the HPC resources of IDRIS under the allocation 2021-AD011012227R2 made by GENCI. It was funded in part by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute), and the ANR project VideoPredict, reference ANR-21-FAI1-0002-01. JP was supported in part by the Louis Vuitton/ENS chair in artificial intelligence and a Global Distinguished Professorship at the Courant Institute of Mathematical Sciences and the Center for Data Science at New York University.

References

[1] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Balle, Sung Jin Hwang, and George Toderici. Scale-space flow for end-to-end optimized video compression. In *CVPR*, 2020. 1

[2] Padmanabhan Anandan. A computational framework and an algorithm for the measurement of visual motion. *IJCV*, 1989. 2

[3] Christian Bailer, Bertram Taetz, and Didier Stricker. Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. In *ICCV*, 2015. 2

[4] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *SIGGRAPH*, 2009. 2

[5] Thomas Brox and Jitendra Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *TPAMI*, 2010. 2

[6] Peter J Burt and Edward H Adelson. The laplacian pyramid as a compact image code. *Transaction on Communications*, 1983. 2

[7] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012. 2

[8] Ruojin Cai, Joseph Tung, Qianqian Wang, Hadar Averbuch-Elor, Bharath Hariharan, and Noah Snavely. Doppelgangers: Learning to disambiguate images of similar structures. In *ICCV*, 2023. 7

[9] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 6

[10] Zhuoyuan Chen, Hailin Jin, Zhe Lin, Scott Cohen, and Ying Wu. Large displacement optical flow from nearest neighbor fields. In *CVPR*, 2013. 2

[11] Xuelian Cheng, Huan Xiong, Deng-Ping Fan, Yiran Zhong, Mehrtash Harandi, Tom Drummond, and Zongyuan Ge. Implicit motion handling for video camouflaged object detection. In *CVPR*, 2022. 1

[12] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *EMNLPW*, 2014. 4, 12

[13] Tomas Crivelli, Pierre-Henri Conze, Philippe Robert, and Patrick Pérez. From optical flow to dense long term correspondences. In *ICIP*, 2012. 2

[14] Tomas Crivelli, Matthieu Fradet, Pierre-Henri Conze, Philippe Robert, and Patrick Pérez. Robust optical flow integration. *TIP*, 2014. 2

[15] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *CVPRW*, 2018. 8, 15

[16] 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. *NeurIPS*, 2022. 1, 2, 4, 6

[17] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. TAPIR: Tracking any point with per-frame initialization and temporal refinement. In *ICCV*, 2023. 1, 2, 3, 4, 5, 6, 8

[18] 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 *ICCV*, 2015. 1, 2

[19] Denis Fortun, Patrick Bouthemy, and Charles Kervrann. Optical flow modeling and computation: A survey. *CVIU*, 2015. 1

[20] David Gibson and Michael Spann. Robust optical flow estimation based on a sparse motion trajectory set. *TIP*, 2003. 2

[21] 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 *CVPR*, 2022. 2, 4

[22] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *ECCV*, 2022. 1, 2, 4, 6

[23] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 1981. 2

[24] Yinlin Hu, Rui Song, and Yunsong Li. Efficient coarse-to-fine patchmatch for large displacement optical flow. In *CVPR*, 2016. 2

[25] Yinlin Hu, Yunsong Li, and Rui Song. Robust interpolation of correspondences for large displacement optical flow. In *CVPR*, 2017. 2

[26] 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 *ECCV*, 2022. 2

[27] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *ECCV*, 2022. 1

[28] 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 *CVPR*, 2017. 2

[29] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. In *ICLR*, 2022. 2

[30] Joel Janai, Fatma Guney, Jonas Wulff, Michael J Black, and Andreas Geiger. Slow flow: Exploiting high-speed cameras for accurate and diverse optical flow reference data. In *CVPR*, 2017. 2

[31] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning to estimate hidden motions with global motion aggregation. In *ICCV*, 2021. 2, 4, 5

[32] Nick Kanopoulos, Nagesh Vasanthavada, and Robert L Baker. Design of an image edge detection filter using the sobel operator. *Journal of solid-state circuits*, 1988. 3

[33] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together. *arXiv preprint*, 2023. 1, 2, 3, 4, 5, 6, 8, 14, 15

[34] Yoni Kasten, Dolev Ofri, Oliver Wang, and Tali Dekel. Layered neural atlases for consistent video editing. *TOG*, 2021. 1

[35] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 4

[36] Philipp Krähenbühl and Vladlen Koltun. Efficient nonlocal regularization for optical flow. In *ECCV*, 2012. 2

[37] Guillaume Le Moing, Jean Ponce, and Cordelia Schmid. WALDO: Future video synthesis using object layer decomposition and parametric flow prediction. In *ICCV*, 2023. 1

[38] 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 *CoRL*, 2021. 6

[39] Zhen Li, Cheng-Ze Lu, Jianhua Qin, Chun-Le Guo, and Ming-Ming Cheng. Towards an end-to-end framework for flow-guided video inpainting. In *CVPR*, 2022. 1

[40] Zhen Li, Zuo-Liang Zhu, Ling-Hao Han, Qibin Hou, Chun-Le Guo, and Ming-Ming Cheng. AMT: All-pairs multi-field transforms for efficient frame interpolation. In *CVPR*, 2023. 1

[41] SukHwan Lim, John G Apostolopoulos, and AE Gamal. Optical flow estimation using temporally oversampled video. *TIP*, 2005. 2

[42] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. LightGlue: Local feature matching at light speed. In *ICCV*, 2023. 8, 15

[43] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 8, 15

[44] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. DVC: An end-to-end deep video compression framework. In *CVPR*, 2019. 1

[45] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, 1981. 2

[46] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024. 2

[47] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 2

[48] Etienne Mémin and Patrick Pérez. Hierarchical estimation and segmentation of dense motion fields. *IJCV*, 2002. 2

[49] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2

[50] Michal Neoral, Jonáš Šerỳch, and Jiří Matas. MFT: Long-term tracking of every pixel. In *WACV*, 2024. 2, 4, 5, 6

[51] Nils Papenberg, Andrés Bruhn, Thomas Brox, Stephan Didas, and Joachim Weickert. Highly accurate optic flow computation with theoretically justified warping. *IJCV*, 2006. 2

[52] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 6

[53] René Ranftl, Kristian Bredies, and Thomas Pock. Non-local total generalized variation for optical flow estimation. In *ECCV*, 2014. 2

[54] Anurag Ranjan and Michael J. Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017. 2

[55] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. EpicFlow: Edge-preserving interpolation of correspondences for optical flow. In *CVPR*, 2015. 2

[56] Peter Sand and Seth Teller. Particle video: Long-range motion estimation using point trajectories. *IJCV*, 2008. 2

[57] Frank Steinbrücker, Thomas Pock, and Daniel Cremers. Large displacement optical flow computation without warping. In *ICCV*, 2009. 2

[58] Xiuchao Sui, Shaohua Li, Xue Geng, Yan Wu, Xinxing Xu, Yong Liu, Rick Goh, and Hongyuan Zhu. Craft: Cross-attentional flow transformer for robust optical flow. In *CVPR*, 2022. 2

[59] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 1, 2

[60] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 1, 2, 3, 4, 5, 6, 12

[61] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. DISK: Learning local features with policy gradient. *NeurIPS*, 2020. 8

[62] Qianqian Wang, Yen-Yu Chang, Ruojin Cai, Zhengqi Li, Bharath Hariharan, Aleksander Holynski, and Noah Snavely. Tracking everything everywhere all at once. In *ICCV*, 2023. 2, 4, 6

[63] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. Deepflow: Large displacement optical flow with deep matching. In *ICCV*, 2013. 2

[64] 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 *ICCV*, 2023. 2, 4, 5

[65] Yue Wu, Rongrong Gao, Jaesik Park, and Qifeng Chen. Future video synthesis with object motion prediction. In *CVPR*, 2020. 1

[66] Jonas Wulff and Michael J Black. Efficient sparse-to-dense optical flow estimation using a learned basis and layers. In *CVPR*, 2015. 2

[67] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. GMFlow: Learning optical flow via global matching. In *CVPR*, 2022. 2

[68] Jia Xu, René Ranftl, and Vladlen Koltun. Accurate optical flow via direct cost volume processing. In *CVPR*, 2017. 2

[69] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by motion grouping. In *ICCV*, 2021. 1

[70] Vickie Ye, Zhengqi Li, Richard Tucker, Angjoo Kanazawa, and Noah Snavely. Deformable sprites for unsupervised video decomposition. In *CVPR*, 2022. 1

[71] Xiaoming Zhao, Xingming Wu, Weihai Chen, Peter CY Chen, Qingsong Xu, and Zhengguo Li. ALIKED: A lighter keypoint and descriptor extraction network via deformable transformation. *TIM*, 2023. 8

[72] Yang Zheng, Adam W Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J Guibas. PointOdyssey: A large-scale synthetic dataset for long-term point tracking. In *ICCV*, 2023. 1, 2, 3, 4, 5, 8, 15

[73] Junsheng Zhou, Yuwang Wang, Kaihuai Qin, and Wenjun Zeng. Moving indoor: Unsupervised video depth learning in challenging environments. In *ICCV*, 2019. 2

[74] Shangchen Zhou, Chongyi Li, Kelvin CK Chan, and Chen Change Loy. ProPainter: Improving propagation and transformer for video inpainting. In *ICCV*, 2023. 1