Dense Trajectory Fields: Consistent and Efficient Spatio-Temporal Pixel Tracking

Marc Tournadre[1,2], Catherine Soladié[1], Nicolas Stoiber[2], and Pierre-Yves Richard[1]

[1] CentraleSupélec, France
{catherine.soladie,pierre-yves.richard}@centralesupelec.fr
[2] Dynamixyz - TakeTwo Interactive, France
{marc.tournadre,nicolas.stoiber}@take2games.com

Abstract. In this paper, we present Dense Trajectory Fields (DTF), a novel low-level holistic approach inspired by optical-flow and trajectory methods, focusing on both spatial and temporal aspects at once. DTF contains the dense and long-term trajectories of all pixels from a reference frame, over an entire sequence. We solve it with DTF-Net, a fast and lightweight neural network, comprising 3 main components: (1) a joint iterative refinement of image and motion features over residual layers, (2) token-based Reciprocal Attention clusters and, (3) a Refinement Network that builds patch-to-patch cost-volumes around salient centroid trajectories. We extend the recent Kubric dataset to provide dense ground-truth over all pixels, to train DTF-Net. Experiments show that optical-flow and trajectory methods exhibit either temporal or spatial inconsistencies. Conversely, DTF-Net provides a better compromise while keeping faster, giving a coherent motion over the entire sequence. Code is available at `https://github.com/MTournadre/DTFNet.git`

Keywords: Optical-Flow · Trajectory · Tracking · Video Analysis

1 Introduction

Optical-Flow is an essential vision task in low-level image analysis. It provides a dense pixel-wise match between a pair of images. Usually applied to consecutive frames, it describes an instantaneous motion that can be of many uses, like action recognition [40, 47], video compression [30, 48], or supervising higher-level 3D reconstruction [53, 67]. By nature (variational and CNN-based), it gives smooth and spatially consistent results. Often limited to two frames, it cannot leverage long-term information. As such, multi-frame optical-flow [32, 41] has shown that a larger temporal context can improve the estimation.

On the other hand, recent advances in vision [36, 52, 53] have shown a growing interest in determining point trajectories, leading to *persistent particle tracking* [17], or *Tracking Any Point* [11], over multiple frames. These methods give well structured trajectories over time and are robust to occlusions, providing a precious long-term motion information. Nonetheless, most of them work on a

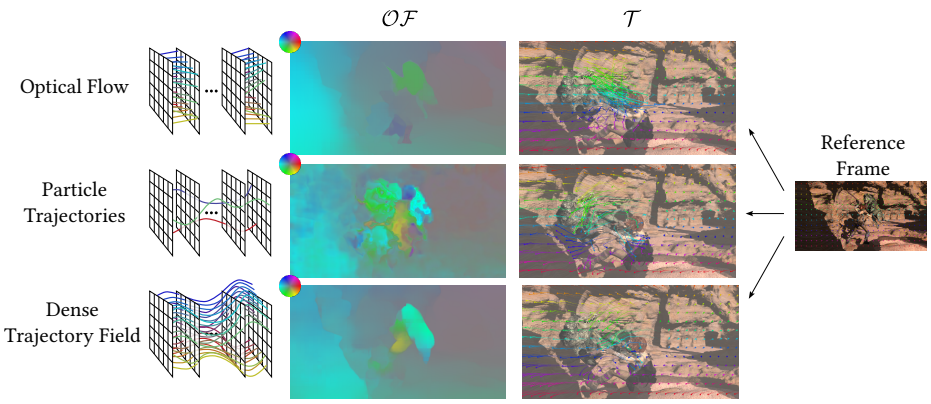


Fig. 1: We introduce *Dense Trajectory Fields*, a low-level holistic approach for tracking all pixels of a reference frame, at once. Usual optical-flow and trajectory algorithms (here, RAFT and PIPs) suffer from either temporal or spatial consistency. Our method focuses on both aspects at once, resulting in a more coherent motion overall.

sparse set of independent particles, which discards the spatial cohesion. This is why some recent works [8, 24, 36] have focused on densifying this approach.

Following these observations, we propose in this work to process an entire input sequence at once to determine the motion of every pixel from one reference frame, resulting in a *Dense Trajectory Field* (DTF). DTF is *low-level* and *holistic*, giving fast and consistent results. We introduce DTF-Net, a lightweight architecture built on a comprehensive use of transformers, that have shown great performance at image and temporal signal analysis [56,66]. Our method successively refines image and motion embeddings through attention clusters, from which we extract localized centroids. Their motion is updated based on multi-resolution patch-to-patch cost-volumes, and restored to the whole sequence thanks to *reciprocal attention*. Repeating this process along several residual layers enables to build a rich and consistent motion over all pixels.

To train our model, we extend the synthetic dataset Kubric [16] to provide dense trajectories ground-truth, and share the code to allow replication of this extension. We compare trajectory and optical-flow methods on both tasks, and show that DTF-Net is overall more coherent.

In summary, we show the following contributions:

- A new low-level holistic approach to dense pixel tracking called *Dense Trajectory Fields*, expressing dense and long-term motion over an entire sequence
- A lightweight and efficient attention-based architecture for processing videos and motion
- An overview on how current trajectory and optical-flow methods behave spatially and temporally
- An extension of the Kubric [16] dataset providing dense ground-truth

2 Related Work

2.1 Optical-Flow

Optical-flow consists in estimating the motion of every pixel from one frame to another. It has first been solved as a variational method by Horn and Schunk [18]. Back then, it was already essential for these optimization methods to keep a smoothness constraint (*e.g.* a Laplacian prior) to ensure the spatial consistency [5]. Since the arrival of deep learning in computer vision, most methods now use CNN-based approaches, starting with DeepFlow [58]. FlowNet [14] introduced more complex architectures, showing the importance of correlations as input features to motion estimation. Following works have quickly established standard datasets [6,15,27,34,62], mostly synthetic, and optimal training strategies [20]. PWC-Net [46] considered applying a coarse-to-fine approach, by using successive feature warping and correlations at different resolutions.

The RAFT [50] architecture has overtaken previous approaches [44] by introducing a more structured multi-resolution 4D cost-volume, an additional context encoder, and a GRU that mimics an optimization algorithm. Despite other successful approaches [21, 33, 68, 70], most recent works are still based on RAFT [2, 10, 31, 64]. Some of them focus on occlusion [28], others use transformers [19, 23, 42, 43, 63] to improve global consistency. On top of that, recent works [16, 45, 57, 61, 69] built datasets with more state-of-the-art shading and dynamics.

To overcome the two-frame limitation, some works propose to tackle Multi-Frame Optical-Flow [7, 22, 32, 41]. They show how long-term information can improve their result. Yet, they do not consistently track points over time, but still evaluate the instantaneous movement from t to $t+1$, losing track of occluded pixels. Recently, AccFlow [61] proposed to tackle Long-Range Optical-Flow by backward accumulating local flows to mitigate occlusion.

2.2 Particle Trajectories

Likewise, recent advances focus on leveraging long-term information to estimate the trajectories of query points [11, 12, 17, 69], following the seminal work of Sand and Teller [39]. They also build cost-volumes on video features, and iteratively update query, motion and visibility – and for some, uncertainty.

Unlike optical-flow, they focus on the temporal context and are robust to occlusions, but they discard the spatial smoothness of optical-flow. This is why some recent works have focused on densifying this processing: MFT [36] computes chains of optical-flows with uncertainty prediction to track every pixels. FlowTrack [8] further adds an error compensation module. CoTracker [24] proposes to entangle several trajectories with transformers. DOT [29] densifies the result by mixing both trajectory and optical-flow approaches.

OmniMotion [53] introduced a test-time optimization to track all pixels using a dynamic implicit neural representation. Though it achieves impressive results, it is very heavy and higher-level: it can turn unpractical, and relies on an off-the-shelf optical-flow or trajectory algorithm.

2.3 Transformers in Vision

The transformer architecture has been introduced by [51] to initially solve NLP problems. It builds pair-wise relationships between two signals (Cross-Attention) – possibly the same signal (Self-Attention) – using multi-head attention. It has quicly been extended to images [9, 13, 25, 37, 66] and even videos [1, 3, 26, 35], reaching new state-of-the-art results, thanks to their powerful generic formulation. Due to its original quadratic complexity, many works have tried to alleviate the attention computation using sparse attention [38], linear transformations [54], quad-tree attention [49], additive attention [60], pyramidal attention [55, 56], or tokenization [4, 59].

In this work, we also use tokenization to reduce the cost of video processing to linear complexity ($N_{\text{tokens}} \times N_{\text{pixels}}$, where $N_{\text{tokens}} \ll N_{\text{pixels}}$ is constant). We leverage the resulting semantical attention clusters to build localized centroids. We then use the reciprocity of this pixel-to-token relationship to go back and forth between sparse and dense feature spaces.

3 Dense Trajectory Fields (DTF)

In the following, we use the bold mathematic notation to denote matrices and tensors $\mathbf{T}$. We index them using $\mathbf{T}_{tij}$ and slice them with $\mathbf{T}_{*ij}$. Lowercase expressions like $\mathbf{m}$ refer to the sparse centroid homologous of an original dense entity $\mathbf{M}$ expressed in image space.

Let's start by describing the nature of the data we aim to predict, and show how it impacts our architecture. Formally, we consider an input sequence $\mathbf{I} \in \mathbb{R}^{T \times H \times W \times 3}$ of T consecutive RGB images of size $H \times W$. Estimating the *Dense Trajectory Field* (DTF) consists in determining the 2D motion $\mathbf{\Delta X} \in \mathbb{R}^{T \times H \times W \times 2}$ of all pixels from one reference frame $t_{\text{ref}} \in [1, T]$ to all frames t in $[1, T]$. Note that there is no constraint on t_{ref}, meaning that the DTF is not necessarily causal.

This definition unifies all optical flows and all pixel trajectories into a dense 3D motion volume as illustrated in Fig. 2. According to this definition, we can get the *optical-flow* from the reference frame to any other frame t by taking a temporal slice:

$$\mathcal{OF}\left(\mathbf{I}_{t_{\text{ref}}} \rightarrow \mathbf{I}_t\right) = \mathbf{\Delta X}_t \qquad \forall t \tag{1}$$

Consequently, $\mathbf{\Delta X}_{t_{\text{ref}}}$ should be $\mathbf{0}$. We also obtain the *trajectory* of any pixel $\mathbf{p}_{ij}$ of $\mathbf{I}_{t_{\text{ref}}}$, by slicing along (i, j):

$$\mathcal{T}\left(\mathbf{p}_{ij}\right) = \mathbf{\Delta X}_{*ij} \qquad \forall (i, j) \tag{2}$$

We draw the reader's attention on the fact that this motion tensor does not operate in video space, in the sense that the 2D motion $\mathbf{\Delta X}_{tij}$ sampled at (t, i, j) is not related to the content of the sequence at (t, i, j). It is instead the optical-flow from t_{ref} to t, hence it shares the spatial layout of $\mathbf{I}_{t_{\text{ref}}}$ for all t, and not

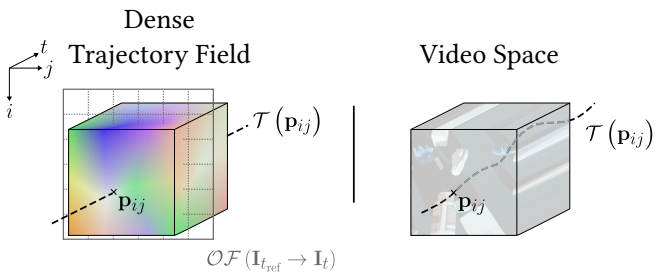


Fig. 2: The Dense Trajectory Field is a dense motion volume of all pixels from a reference frame $\mathbf{I}_{t_{\text{ref}}}$. Taking a slice over t gives the optical-flow $\mathcal{OF}$ from t_{ref} to t, while a slice at (i, j) gives the trajectory $\mathcal{T}$ of the pixel $\mathbf{p}_{ij}$ from reference frame. While a trajectory of a pixel $\mathbf{p}_{ij}$ is a moving point in video space, it is represented as a straight line in DTF (motion space).

$\mathbf{I}_t$. In other words, in video space a trajectory is a moving point, but in motion space it is a fixed point at (i, j), as illustrated in Fig. 2. In the following, we take care of treating separately any *image*- and *motion*-related features.

4 Centroid-Refinement Architecture

We call our architecture DTF-Net. It is based on 3 main components:

- A residual architecture that refines the image and motion features over successive layers
- Each layer summarizes the information over sparse centroids, processes them, and restores the result to the original space
- These centroids are processed through a Refinement Network, that works on multi-resolution patch-to-patch cost-volumes

4.1 Overall Architecture

We depict an overview of our architecture in Fig. 3. Similar to most optical-flow architectures, it first encodes each image through a CNN encoder $\mathcal{E}$. This encoder is analog to the original RAFT [50] and reduces the resolution to $\frac{1}{8}$.

Over L successive residual layers, we refine in parallel two sets of features: image features $\mathbf{F}^l$, where $\mathbf{F}^0$ is the result of the encoder $\mathcal{E}$, and motion features $\mathbf{M}^l$, where $\mathbf{M}^0$ is initialized to $\mathbf{0}$. Unlike PIPs [17] we use a single *implicit* feature space for both motion and visibility. This space is meant to intricate motion and visiblity, and contain a richer representation than a simple $\Delta(x, y)$.

We design a Motion Head $\mathcal{H}_m$ (and Visibility Head $\mathcal{H}_v$), a simple 2-layer MLP, that transforms these implicit motion features into the DTF (resp. into visibility scores). These heads are shared across all layers. Finally, similar to the original RAFT, we apply a bilinear upsampling mask determined from the reference image features $\mathbf{F}^l_{t_{\text{ref}}}$, that restores the DTF back to the original resolution.

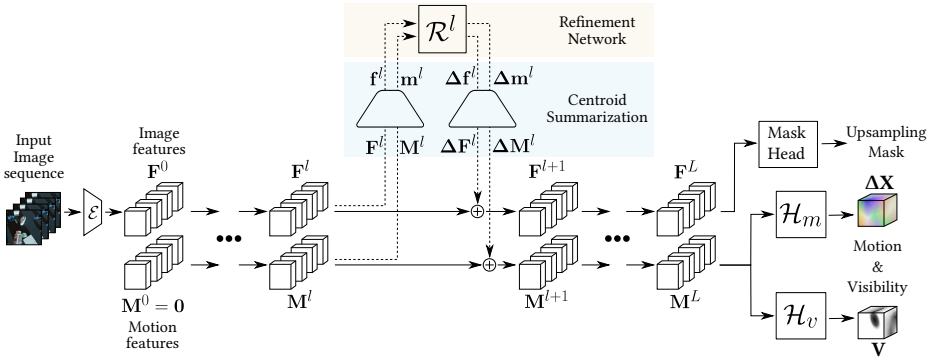


Fig. 3: Our overall architecture processes a full sequence of images, by iteratively refining image features $\mathbf{F}$ and motion features $\mathbf{M}$ altogether through a residual process. At each layer l, both are reduced to smaller sets $\mathbf{f}$ and $\mathbf{m}$. We estimate an upadte through our Refinement Network $\mathcal{R}$, and apply it back to the full original space. Motion features are finallly converted into the DTF $\mathbf{\Delta X}$ and visibility $\mathbf{V}$, and upsampled to the original resolution.

4.2 Centroid Summarization

Doing an extensive treatment of all pixels in the sequence would be too greedy. We take profit from the fact that, perceptually, most of the motion information is very diffused over the image, and can be well approximated from a small set of salient keypoints. To reflect this, we try to summarize the content of the sequence by applying tokenization.

For each layer l, we learn a set of tokens $\mathbf{T^l} \in \mathbb{R}^{N_T \times D_T}$, that will query the content of the whole sequence. Similarly to the original transformer paper [51], we construct a multi-head attention by projecting $\mathbf{T^l}$ and $\mathbf{F^l}$ into queries and keys, split them into N_H heads, and compute their dot product:

$$\mathbf{Q} = \mathbf{T}^l = \left(\mathbf{Q}_{(1)} \cdots \mathbf{Q}_{(N_H)}\right)^\intercal \tag{3}$$

$$\mathbf{K} = \mathbf{W}_K^l \mathbf{F^l} = \left(\mathbf{K}_{(1)} \cdots \mathbf{K}_{(N_H)}\right)^\intercal \tag{4}$$

$$\mathbf{A}_{(h)} = softmax\left(\frac{\mathbf{Q}_{(h)}\mathbf{K}_{(h)}^\intercal}{\sqrt{D_T}}\right) \qquad \forall h \in [1, N_H] \tag{5}$$

We interpret these attention maps as *attention clusters*, from which we decide to build *centroids*, *i.e.* a representative for each cluster of pixels. To proceed, we compute the argmax of the attention weights to obtain the position of the centroids, and extract features at these locations.

Unlike the usual transformer approach, we do not build implicit values $\mathbf{V}$. Instead, we simply apply the attention weights to the original feature sets. This way, our Refinement Network keeps working in the original $\mathbf{F}$ and $\mathbf{M}$ spaces,

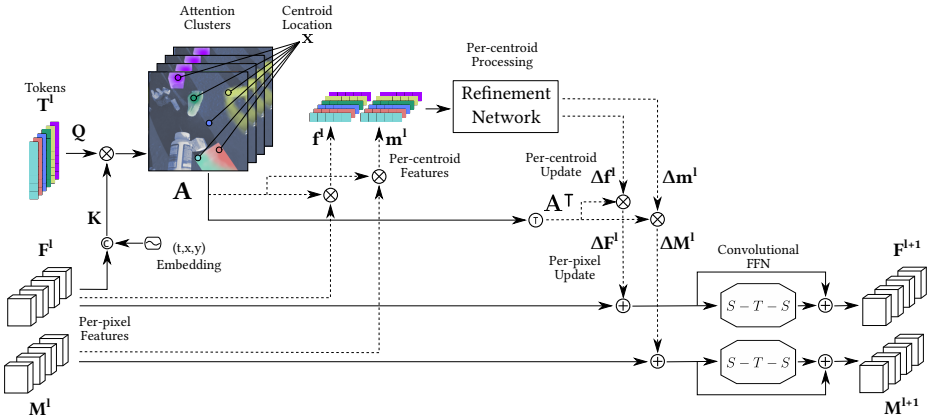


Fig. 4: Our summarization uses tokens to extracts centroids from attention clusters. These centroids allow us to process the same feature sets $\mathbf{f}$ and $\mathbf{m}$, but only on sparse keypoints, which is much lighter. We use the same relationship to compress and restore the information, by building a reciprocal attention $\mathbf{A}^\intercal$. We finally apply a complementary FFN, made of spatial and temporal convolutions.

and is able to apply the Motion Head $\mathcal{H}_m$ on centroids. This is illustrated in Fig. 4.

This summarization may seem coarse, but each layer holds its own set of tokens, meaning that they will all process different sets of $N_T \times N_H$ centroids. By successively refining motion and image features over several layers, more and more details are covered. Moreover, we still benefit from the *linear* complexity of tokenization.

Once the centroids have been treated, we restore their sparse result to the original dense images: instead of creating another set of query/keys, we keep the same pixel-centroid affinities, and apply the *reciprocal attention* $\mathbf{A}^\intercal$:

$$\mathbf{A}^\intercal_{(h)} = softmax\left(\frac{\mathbf{K}_{(h)}\mathbf{Q}^\intercal_{(h)}}{\sqrt{D_T}}\right) \qquad \forall h \in [1, N_H] \tag{6}$$

We concatenate back all heads and apply an output projection matrix $\mathbf{W}_O$. The result is added to the original feature sets. We finally apply a small FFN, composed of spatial 2D convolutions and an inner 1D temporal convolution. Like the standard transformer, these are meant to add local control on the information, which is complementary to the global attention mechanism.

Discussion. Note that this method can easily be applied to other tasks on video sequences, and is particularly suited to a diffuse information like motion. Yet, it comes at the price of working only on a sparse data, which motivated several choices of architecture meant to compensate this potentially poor motion: a wide implicit motion feature space, patch-to-patch correlations instead of pixel-to-patch, and multi-head motion prediction.

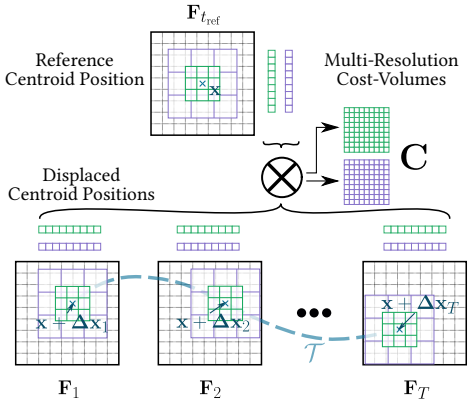


Fig. 5: The Refinement Network processes centroids localized at $\mathbf{x}$. We use the shared Motion Head to reconstruct the centroid trajectory $\mathbf{x} + \mathbf{\Delta x}$, around which we extract feature patches at several resolutions. We build a cost-volume $\mathbf{C}$ made of patch-to-patch correlations.

4.3 Refinement Network

Each layer l is built upon a Refinement Network $\mathcal{R}^l$, designed to update the motion of the centroids $\mathbf{m}^l = \mathbf{A}_{t_{\text{ref}}}\mathbf{M}^l$ and their image features $\mathbf{f} = \mathbf{AF}$, as trajectory methods do [12, 17]. But unlike them, ours is capable to update the whole content of the sequence as well, not only the queried points. This way, we propagate the information all over the sequence, up to the next layers.

Similar to previous works [12, 50], we base our motion processing on multi-resolution cost-volumes $\mathbf{C}$ of image features. To this end, we first apply the shared Motion Head $\mathcal{H}_m$ on each centroid motion feature, and hence obtain their corresponding 2D positions over time $\mathbf{x} + \mathbf{\Delta x}^l = \mathbf{x} + \mathcal{H}_m(\mathbf{m^l})$, around which we extract patches of pixels. This is illustrated in Fig. 5. These patches are of size P^2, and extracted at N_P levels of resolutions, so the resulting multi-resolution patch-to-patch cost-volume results in a pool of $N_P \times P^4$ features, for each frame, for each head, for each token. After applying some linear projections to reduce the feature size, we concatenate it to the centroid motion and image embeddings, and feed that to a simple residual temporal CNN.

This Refinement Network returns an update of motion features $\mathbf{\Delta m}^l$ and image features $\mathbf{\Delta f}^l$. By applying the (per-head) reciprocal attention $\mathbf{A}^\intercal$, we restore this information to the original sequence, and finally use the FFN. As explained in Sec. 3, motion features are recovered by applying the same reference attention for all t, unlike image features:

$$\mathbf{M}_t^{l+1} = \text{FFN}_M \left(\mathbf{M}_t^l + \mathbf{A}_{t_{\text{ref}}}^\intercal \mathbf{\Delta m}_t^l \right) \tag{7}$$

$$\mathbf{F}_t^{l+1} = \text{FFN}_F \left(\mathbf{F}_t^l + \mathbf{A}_t^\intercal \mathbf{\Delta f}_t^l \right) \tag{8}$$

4.4 Dataset

We want to train our network on sequences with extensive sets of trajectories, to benefit from both spatial and temporal coherence from data. We choose to adapt the existing Kubric [16] dataset, to track all pixels of a random reference frame t_{ref} in each sequence. We use the provided per-pixel object coordinates and per-frame object transformations to reconstruct the 3D trajectory of every reference pixel, and project it back to 2D. We use the segmentation and depth data to check for consistency, to detect occlusions – thanks to the depth we also consider self-occlusions.

We take the MOVi-E subset at 256x256 resolution, which provides 10k sequences of 24 consecutive frames, displaying moving simulated objects, with camera movement. We use a similar data augmentation as RAFT, *i.e.* random color jittering, stretching, cropping (spatially, and temporally around the reference frame), occlusion, and flip (spatial and temporal).

4.5 Training

We fully supervise our training on the predicted motion and visibility. Both losses are computed at all intermediate layers l, with a fading coefficient $\gamma \in [0, 1]$, as RAFT and PIPs did. The resulting total loss is:

$$\mathcal{L}_{\text{tot}} = w_{\text{mot}}\mathcal{L}_{\text{mot}} + w_{\text{vis}}\mathcal{L}_{\text{vis}} \tag{9}$$

The motion loss $\mathcal{L}_{\text{mot}}$ is computed using a Huber loss over the 2D trajectories of all pixels. We use a cross-entropy loss to supervise the visiblity prediction:

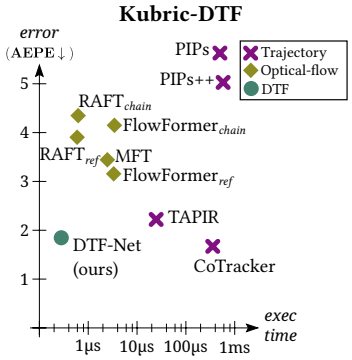
$$\mathcal{L}_{\text{mot}} = \frac{1}{THW}\sum_{l=1}^{L}\gamma^{L-l}\text{Huber}\big(\|\boldsymbol{\Delta}\mathbf{X} - \boldsymbol{\Delta}\mathbf{X}^*\|_2^1\big) \tag{10}$$

$$\mathcal{L}_{\text{vis}} = -\frac{1}{THW}\sum_{l=1}^{L}\gamma^{L-l}\Big[\mathbf{V}^*\log(\mathbf{V}) + (1-\mathbf{V}^*)\log(1-\mathbf{V})\Big] \tag{11}$$

Implementation details. We use $w_{\text{mot}} = 1$ and $w_{\text{vis}} = 3$ to balance losses. Our model is light to inference, but heavy to train. For this reason, we first train our model with 60k iterations on sequences of 8 frames at 256x256 resolution, and finetune it with 40k iterations on sequences of 24 frames at 192x192. The full training takes about 48h on 4x A100 GPUs with 48GB each. More details are provided in the supplemental.

5 Experiments

We compare our approach to other methods for optical-flow and trajectories, each applied to both problems and the DTF task, to highlight their temporal and spatial behaviors.



Method	Nb. of params	Exec. time	VRAM usage	$\Delta_{\mathcal{OF}}$	$\frac{\partial^2 \mathcal{T}}{\partial t^2}$
PIPs [17]	28.8M	490µs	3.3GB	0.14	0.29
PIPs++ [69]	17.6M	580µs	2.2GB	0.16	0.28
TAPIR [12]	54.7M	24µs	2.2GB	0.39	0.23
CoTracker [24]	24.1M	290µs	9.7GB	0.26	0.07
RAFT$_{chain}$ [50]	5.3M	0.68µs	2.8GB	0.24	0.17
RAFT$_{ref}$ [50]	5.3M	0.65µs	2.8GB	0.07	0.48
FlowFormer$_{chain}$ [19]	17.6M	3.5µs	4.4GB	0.28	0.16
FlowFormer$_{ref}$ [19]	17.6M	3.4µs	4.4GB	0.07	0.38
MFT [36]	6.9M	2.5µs	1.0GB	0.03	0.32
DTF-Net (ours)	40.1M	**0.32µs**	2.3GB	0.10	0.12

Fig. 6 & Table 1: Dense Trajectory Fields comparison on the Kubric-DTF validation set. We show the average execution time per-pixel and per-frame. We give the 2^{nd} derivatives on spatial axis (Laplacian of optical-flows) and temporal axis, to illustrate the smoothness of the different methods. More metrics are given in appendix.

Among trajectory-based methods, we show PIPs [17], PIPs++ [69], TAPIR [12], and CoTracker [24]. We use RAFT [50], FlowFormer [19], and MFT [36] as optical-flow-based methods. We expect CoTracker to show more spatial consistency than the other trajectory methods as it introduces spatial dependencies among queries. Similarly, MFT should give more temporal consistency for its capacity to build efficient chains of optical-flows over time.

We run experiments on a GTX Titan X 12GB, with a i7-6700K 4GHz. Unfortunately, we could not test OmniMotion [53] due to its unreasonable computation time and VRAM usage.

5.1 Dense Trajectory Fields

We compare our method to other approaches on the DTF task, which can be considered as an extension of optical-flow and trajectory tasks. To build the DTF from trajectories, one can estimate the trajectory of every pixels in the reference frame. For practical reasons (reasonable runtime or VRAM usage), we had to track only a sparse grid of stride 4 and bilinearly interpolate. Concerning pure optical-flow methods, we consider two strategies: chaining optical-flows on consecutive frames, which implies image warping, can lead to drift and cannot handle occlusion, that we call X_{chain} ; and estimating the optical-flow from the reference frame to all others, which introduces large displacements, but can handle occlusion and recover from bad estimations, marked as X_{ref}. Previous trajectory works [11, 69] only considered the first option.

We use our Kubric-DTF validation set for benchmark: it contains similar shading and dynamics as our training set, but with different objects and backgrounds. To our knowledge, it is the only dataset providing a dense and sufficiently long-term ground-truth – CVO [61] is analogous, but with too short sequences. Note that MFT, TAP-Net, TAPIR and CoTracker have also been trained on the (sparse) Kubric [16] training set.

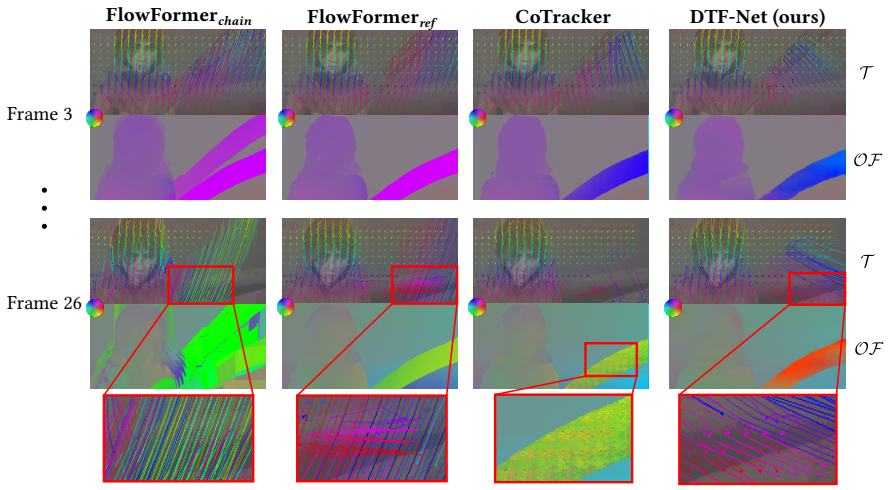


Fig. 7: Qualitative comparison to optical-flow and trajectory methods, on the *wall* sequence from Sintel [6] test set (clean pass). DTF-Net is the only one to show both spatial and temporal consistencies, and to successfully recover the stick after occlusions.

We show in Fig. 6 a runtime comparison of each method aside with its performance on Kubric-DTF. On this dataset, DTF-Net is only beaten by CoTracker, but is about 3 orders of magnitude faster. The memory usage is also shown [3]. Despite having a lot of parameters, our method is the fastest: this is because its architecture is residual, unlike other models that are recurrent. Optical-flow methods are faster than trajectory ones because they are designed to track entire images, when the latter is meant to focus on a few points. Yet, we show, aside with CoTracker, that tracking densely is greatly beneficial. The *chaining* strategy for optical-flow is less accurate than the *ref* one, which trajectory approaches should consider. On this aspect, MFT [36] takes advantage of both.

Fig. 6 also shows the mean norm of the Laplacian on the obtained optical-flows, as well as the second derivative over time. These 2^{nd} order derivatives give the smoothness of each method, to assess the cohesion of the prediction. As expected, trajectory methods are temporally smooth, and optical-flow methods are spatially smooth. DTF-Net shows a good compromise on both aspects, demonstrating its ability to structurally learn smooth spatio-temporal motion from data. We refer the reader to the supplemental video for qualitative evaluation.

Fig. 7 gives qualitative visuals of different methods on the 'wall' sequence from Sintel test set, where the character holds a stick, on a background of a large wall. Here, FlowFormer$_{chain}$ ends up confounding the stick with the background,

[3] We batch methods on 1024 trajectories, or 24 frames for optical-flow methods. The MFT implementation does not support batching.

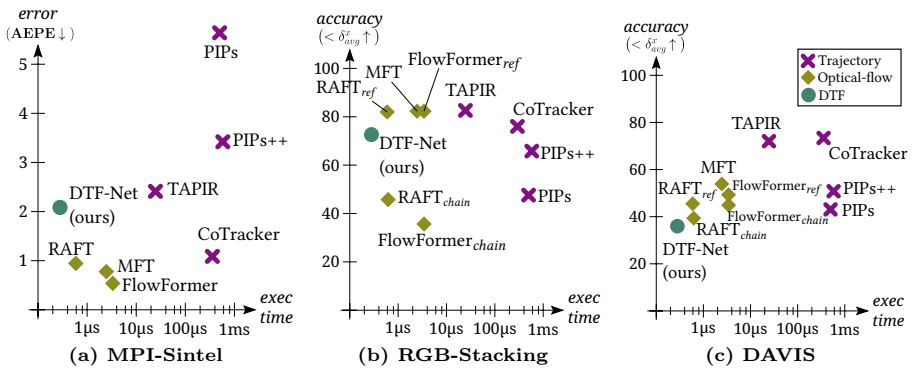


Fig. 8: Comparison of optical-flow, trajectory and DTF methods, all applied to (a) **optical-flow** and (b)(c) **trajectory** tasks. Execution time is measured per pixel and per frame. Detailed metrics are shown in appendix.

whereas FlowFormer$_{ref}$ gives very edgy trajectories. CoTracker [4] misses the spatial coherence of large zones. On the other hand, even though DTF-Net is not the sharpest one, it is the only one depicting both spatial and temporal consistencies. Being holistic makes it able to recover the stick swinging in and out of the frame.

5.2 Optical-Flow & Trajectory

To evaluate on optical-flow, we benchmark all models on the Sintel [6] dataset on the *clean* pass. This dataset provides ground-truth for instantaneous optical-flow and occlusion over long sequences. It enables us to run trajectory methods on small temporal windows around each frame t. By estimating the trajectory of all pixels, we obtain the DTF $\mathbf{\Delta X}$, and extract the forward optical-flow $\mathcal{OF} = \mathbf{\Delta X}_{t+1}$.

Because trajectory methods can be slow when used extensively, we run all methods at half resolution. It is worth noting that RAFT and MFT greatly improve as the resolution increases, which would likely enlarge the gap with trajectory methods.

For evaluation on trajectories, we use the TAP-Vid [11] benchmark, with the 'query-first' strategy on images at 256x256.

Fig. 8 shows results of all methods applied to both problems. Naturally, specialized methods excel in its respective task. We still note on RGB-Stacking that the optical-flow methods with the *ref* strategy gives the best performance: this is because these sequences are very long (250 frames) and contain a very flat shading, for which optical-flow are better suited.

[4] To track all pixels, CoTracker infers independent grids of 30x30 trajectories at different offsets, for practical matters. When merged, these grids might not be consistent.

Architecture	Optical-Flow metrics				Traj. metrics		
	AEPE↓	AAE↓	F-1↓	F-5↓	AJ↑	$< \delta^x_{avg}$↑	OA↑
DTF-Net w/o img upd.	4.51	7.68	44.1	14.1	81.1	85.0	92.9
DTF-Net pixel-to-patch	4.12	**6.16**	<u>38.1</u>	13.9	<u>82.6</u>	85.7	<u>93.6</u>
DTF-Net explicit motion	<u>4.00</u>	6.79	39.4	<u>13.6</u>	82.2	<u>85.8</u>	93.0
DTF-Net (full)	**3.60**	<u>6.26</u>	**35.0**	**11.8**	**84.5**	**87.7**	**94.1**

Table 2: Ablation results, on the validation set of our extended version of Kubric [16]. Shown experiments are: removing the update of image features, using pixel-to-patch correlations, and using explicit motion and visibility. Models are only trained through the first training stage (60k iterations), which may differ from previous results.

Overall, DTF-Net offers a fast compromise for both tasks. DAVIS is the most challenging, it contains complex scenes and motion. In this scenario, as DTF-Net has learned a simple motion manifold, it faces a larger domain gap than trajectory methods: one can deduce complex *point* trajectories from Kubric, but we cannot deduce complex *cluster* motions.

6 Ablation Study

We present ablation experiments to justify our choices of architecture:

- Removing the update of image features $\mathbf{\Delta F}$, and keep the one coming from the encoder, like previous optical-flow and particle methods [11, 12, 17, 50]
- Using pixel-to-patch correlation features instead of patch-to-patch, giving $N_P \times P^2$ features instead of $N_P \times P^4$
- Using explicit motion and visibility, instead of a wide latent motion space

Another considered ablation was to replace the centroid summarization by a dense pixel-wise processing. Yet, this is unfeasible for two reasons: first, because this discards the reciprocal attention $\mathbf{A}_t^\intercal$, making $\mathbf{\Delta f}$ inapplicable to frames other than t_{ref}; second, because the memory consumption would explode.

Results are shown in Tab. 2. The most effective contribution is the image feature update. It shows that the joint refinement of image and motion is very beneficial to the video analysis. Trajectory methods could not do it because they work locally around queries, so they cannot update the rest of the image. At the opposite, thanks to the reciprocal attention, DTF-Net can restore the update to the full image.

The patch-to-patch correlations and wide motion space are also contributing. The intuition for both is to be able to restore a more 'complex' motion (*e.g.* homography) of the sparse centroids to spread over all pixels, and also to share more information with the next residual layers.

We also study the behavior of DTF-Net with different numbers of layers L, to assess the potential of a bigger network. Results for 4, 6, 8 (used in this paper) and 10 layers are shown in Tab. 3. It appears that DTF-Net would benefit from more residual layers, and can still be improved. Yet, the 10-layers model is significantly heavier to train, reaching our 48GB VRAM limitation.

Architecture	Optical-Flow metrics				Traj. metrics		
	AEPE↓	AAE↓	F-1↓	F-5↓	AJ↑	$< \delta^x_{avg}$↑	OA↑
DTF-Net 4 layers	4.92	6.64	45.6	17.8	78.6	81.9	92.8
DTF-Net 6 layers	4.20	6.54	37.4	13.8	82.7	85.9	<u>93.7</u>
DTF-Net 8 layers	<u>3.60</u>	<u>6.26</u>	<u>35.0</u>	<u>11.8</u>	<u>84.5</u>	<u>87.7</u>	**94.1**
DTF-Net 10 layers	**3.40**	**6.03**	**33.6**	**10.9**	**85.2**	**88.5**	**94.1**

Table 3: Results of DTF-Net with different number of layers L, on the validation set of our extended version of Kubric [16]. Models are only trained through the first training stage. Shown experiments have used the model with 8 layers.

7 Limitations & Future Work

Our method relies on positional embeddings: like FlowFormer [19], it does not extend well to higher resolutions and larger temporal windows, but a tiling strategy like theirs should help. Also, our method puts a lot of pressure on the centroids' ability to accurately summarize the *reference* frame: missing an important zone at t_{ref} is fatal for its motion estimation for all frames t. Experiments have shown that DTF-Net offers an efficient for both dense and long-term problems, but does not beat other specialized methods on their respective tasks.

We introduced foundations for low-level holistic motion analysis, but kept DTF-Net as basic as possible. It can be improved in many ways: upgrading the encoder [12, 19], explicitly handling occlusions [28], adding cross-trajectory dependencies [24], using domain adaptation [65], or extending it to online applications [12]. Its generalization capacity could be improved by enhancing its motion manifold with a more diversified dataset with higher resolution [69].

8 Conclusion

We introduce *Dense Trajectory Fields*, a novel approach to low-level, dense and long-term motion estimation, that aims to track all pixels of a reference frame over an entire sequence. We propose DTF-Net, a neural architecture consisting of an iterative refinement of image and motion features. To efficiently alleviate the video processing, we use a mechanism of reciprocal attention to build salient centroids, and refine their trajectories using patch-to-patch cost-volumes. We extend the existing Kubric dataset to give pixel-wise ground-truth, to train our model on. We test our method against existing trajectory and optical-flow algorithms, and show how both behave on each other's task. DTF-Net offers a fast solution that exhibits a good spatial and temporal consistency.

Acknowledgements

This work was possible thanks to the resources shared by Mésocentre Paris Saclay, and thanks to the devoted support of the Dynamixmyz team: V. Barrielle, F. Arrestier, O. Roupin, N. Stoiber & S. Kirthi Kumaraswamy.

References

1. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6836–6846 (2021)
2. Bai, S., Geng, Z., Savani, Y., Kolter, J.Z.: Deep Equilibrium Optical Flow Estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 620–630 (2022)
3. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 813–824. PMLR (18–24 Jul 2021), https://proceedings.mlr.press/v139/bertasius21a.html
4. Bolya, D., Fu, C.Y., Dai, X., Zhang, P., Feichtenhofer, C., Hoffman, J.: Token merging: Your vit but faster. In: The Eleventh International Conference on Learning Representations (2023), https://openreview.net/forum?id=JroZRaRw7Eu
5. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In: European conference on computer vision. pp. 25–36. Springer (2004)
6. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: A. Fitzgibbon et al. (Eds.) (ed.) European Conf. on Computer Vision (ECCV). pp. 611–625. Part IV, LNCS 7577, Springer-Verlag (Oct 2012)
7. Chen, Y., Zhu, D., Shi, W., Zhang, G., Zhang, T., Zhang, X., Li, J.: Mfcflow: A motion feature compensated multi-frame recurrent network for optical flow estimation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 5068–5077 (2023)
8. Cho, S., Huang, J., Kim, S., Lee, J.Y.: Flowtrack: Revisiting optical flow for long-range dense tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19268–19277 (2024)
9. Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., Shen, C.: Twins: Revisiting the design of spatial attention in vision transformers. Advances in Neural Information Processing Systems **34**, 9355–9366 (2021)
10. Deng, C., Luo, A., Huang, H., Ma, S., Liu, J., Liu, S.: Explicit motion disentangling for efficient optical flow estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9521–9530 (2023)
11. Doersch, C., Gupta, A., Markeeva, L., Recasens, A., Smaira, L., Aytar, Y., Carreira, J., Zisserman, A., Yang, Y.: Tap-vid: A benchmark for tracking any point in a video. Advances in Neural Information Processing Systems **35**, 13610–13626 (2022)
12. Doersch, C., Yang, Y., Vecerik, M., Gokay, D., Gupta, A., Aytar, Y., Carreira, J., Zisserman, A.: TAPIR: Tracking Any Point with per-frame Initialization and temporal Refinement (2023), _eprint: 2306.08637
13. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=YicbFdNTTy
14. Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 2758–2766 (2015)

15. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2012)
16. Greff, K., Belletti, F., Beyer, L., Doersch, C., Du, Y., Duckworth, D., Fleet, D.J., Gnanapragasam, D., Golemo, F., Herrmann, C., Kipf, T., Kundu, A., Lagun, D., Laradji, I., Liu, H.T.D., Meyer, H., Miao, Y., Nowrouzezahrai, D., Oztireli, C., Pot, E., Radwan, N., Rebain, D., Sabour, S., Sajjadi, M.S.M., Sela, M., Sitzmann, V., Stone, A., Sun, D., Vora, S., Wang, Z., Wu, T., Yi, K.M., Zhong, F., Tagliasacchi, A.: Kubric: A Scalable Dataset Generator. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3749–3761 (Jun 2022)
17. Harley, A.W., Fang, Z., Fragkiadaki, K.: Particle video revisited: Tracking through occlusions using point trajectories. In: ECCV (2022)
18. Horn, B.K., Schunck, B.G.: Determining optical flow. In: Techniques and Applications of Image Understanding. vol. 281, pp. 319–331. International Society for Optics and Photonics (1981)
19. Huang, Z., Shi, X., Zhang, C., Wang, Q., Cheung, K.C., Qin, H., Dai, J., Li, H.: FlowFormer: A transformer architecture for optical flow. ECCV (2022)
20. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: Flownet 2.0: Evolution of optical flow estimation with deep networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2462–2470 (2017)
21. Jaegle, A., Borgeaud, S., Alayrac, J.B., Doersch, C., Ionescu, C., Ding, D., Koppula, S., Zoran, D., Brock, A., Shelhamer, E., Henaff, O.J., Botvinick, M., Zisserman, A., Vinyals, O., Carreira, J.: Perceiver IO: A general architecture for structured inputs & outputs. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=fILj7WpI-g
22. Janai, J., Guney, F., Ranjan, A., Black, M., Geiger, A.: Unsupervised learning of multi-frame optical flow with occlusions. In: Proceedings of the European conference on computer vision (ECCV). pp. 690–706 (2018)
23. Jiang, S., Campbell, D., Lu, Y., Li, H., Hartley, R.: Learning to Estimate Hidden Motions with Global Motion Aggregation (2021), 00006 _eprint: 2104.02409
24. Karaev, N., Rocco, I., Graham, B., Neverova, N., Vedaldi, A., Rupprecht, C.: Cotracker: It is better to track together. arXiv preprint arXiv:2307.07635 (2023)
25. Khan, S., Naseer, M., Hayat, M., Zamir, S.W., Khan, F.S., Shah, M.: Transformers in vision: A survey. ACM computing surveys (CSUR) **54**(10s), 1–41 (2022), publisher: ACM New York, NY
26. Kim, T.H., Sajjadi, M.S., Hirsch, M., Scholkopf, B.: Spatio-temporal transformer network for video restoration. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 106–122 (2018)
27. Kondermann, D., Nair, R., Meister, S., Mischler, W., Güssefeld, B., Hofmann, S., Brenner, C., Jähne, B.: Stereo ground truth with error bars. In: Asian Conference on Computer Vision, ACCV 2014 (2014)
28. Kong, L., Yang, X., Yang, J.: OAS-Net: Occlusion Aware Sampling Network for Accurate Optical Flow. In: ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 2475–2479. IEEE (2021). https://doi.org/10/gn298w, 00002
29. Le Moing, G., Ponce, J., Schmid, C.: Dense optical tracking: Connecting the dots. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 19187–19197 (June 2024)

30. Li, J., Li, B., Lu, Y.: Deep contextual video compression. Advances in Neural Information Processing Systems **34**, 18114–18125 (2021)
31. Li, J., Niu, Y.: Cgcv: Context guided correlation volume for optical flow neural networks. arXiv e-prints pp. arXiv–2212 (2022)
32. Lu, Y., Wang, Q., Ma, S., Geng, T., Chen, Y.V., Chen, H., Liu, D.: Transflow: Transformer as flow learner. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18063–18073 (2023)
33. Luo, A., Yang, F., Luo, K., Li, X., Fan, H., Liu, S.: Learning optical flow with adaptive graph reasoning. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI) (2022)
34. Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) (2016), `http://lmb.informatik.uni-freiburg.de/Publications/2016/MIFDB16`
35. Neimark, D., Bar, O., Zohar, M., Asselmann, D.: Video transformer network. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3163–3172 (2021)
36. Neoral, M., Šerých, J., Matas, J.: Mft: Long-term tracking of every pixel (2023)
37. Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., Tran, D.: Image transformer. In: International Conference on Machine Learning. pp. 4055–4064. PMLR (2018), 00000
38. Press, O., Smith, N., Lewis, M.: Train short, test long: Attention with linear biases enables input length extrapolation. In: International Conference on Learning Representations (2021)
39. Sand, P., Teller, S.: Particle Video: Long-Range Motion Estimation Using Point Trajectories. Int J Comput Vis **80**, 72–91 (2008)
40. Sevilla-Lara, L., Liao, Y., Güney, F., Jampani, V., Geiger, A., Black, M.J.: On the integration of optical flow and action recognition. In: Pattern Recognition: 40th German Conference, GCPR 2018, Stuttgart, Germany, October 9-12, 2018, Proceedings 40. pp. 281–297. Springer (2019)
41. Shi, X., Huang, Z., Bian, W., Li, D., Zhang, M., Cheung, K.C., See, S., Qin, H., Dai, J., Li, H.: Videoflow: Exploiting temporal cues for multi-frame optical flow estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12469–12480 (2023)
42. Shi, X., Huang, Z., Li, D., Zhang, M., Cheung, K.C., See, S., Qin, H., Dai, J., Li, H.: Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1599–1610 (2023)
43. Sui, X., Li, S., Geng, X., Wu, Y., Xu, X., Liu, Y., Goh, R., Zhu, H.: CRAFT: Cross-Attentional Flow Transformer for Robust Optical Flow. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17602–17611 (2022)
44. Sun, D., Herrmann, C., Reda, F., Rubinstein, M., Fleet, D.J., Freeman, W.T.: What Makes RAFT Better Than PWC-Net? (Disentangling Architecture and Training for Optical Flow). In: ECCV (2022)
45. Sun, D., Vlasic, D., Herrmann, C., Jampani, V., Krainin, M., Chang, H., Zabih, R., Freeman, W.T., Liu, C.: Autoflow: Learning a better training set for optical flow. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10093–10102 (2021)

46. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume (2018), 01207 _eprint: 1709.02371
47. Sun, S., Kuang, Z., Sheng, L., Ouyang, W., Zhang, W.: Optical flow guided feature: A fast and robust motion representation for video action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1390–1399 (2018)
48. Tang, C., Sheng, X., Li, Z., Zhang, H., Li, L., Liu, D.: Offline and online optical flow enhancement for deep video compression. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 5118–5126 (2024)
49. Tang, S., Zhang, J., Zhu, S., Tan, P.: Quadtree attention for vision transformers. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=fR-EnKWL_Zb
50. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: European Conference on Computer Vision. pp. 402–419. Springer (2020), 00000
51. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, \., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017), 34320
52. Wang, C., Eckart, B., Lucey, S., Gallo, O.: Neural Trajectory Fields for Dynamic Novel View Synthesis (2021), _eprint: 2105.05994
53. Wang, Q., Chang, Y.Y., Cai, R., Li, Z., Hariharan, B., Holynski, A., Snavely, N.: Tracking everything everywhere all at once. In: International Conference on Computer Vision (2023)
54. Wang, S., Li, B.Z., Khabsa, M., Fang, H., Ma, H.: Linformer: Self-attention with linear complexity. arXiv e-prints pp. arXiv–2006 (2020)
55. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 568–578 (2021)
56. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: PVT v2: Improved baselines with Pyramid Vision Transformer. Computational Visual Media **8**(3), 415–424 (Mar 2022). https://doi.org/10.1007/s41095-022-0274-8, https://doi.org/10.1007%2Fs41095-022-0274-8, publisher: Springer Science and Business Media LLC
57. Wang, W., Zhu, D., Wang, X., Hu, Y., Qiu, Y., Wang, C., Hu, Y., Kapoor, A., Scherer, S.: Tartanair: A dataset to push the limits of visual slam. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4909–4916. IEEE (2020)
58. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: DeepFlow: Large displacement optical flow with deep matching. In: Proceedings of the IEEE international conference on computer vision. pp. 1385–1392 (2013), 00000
59. Wu, B., Xu, C., Dai, X., Wan, A., Zhang, P., Yan, Z., Tomizuka, M., Gonzalez, J., Keutzer, K., Vajda, P.: Visual transformers: Token-based image representation and processing for computer vision. arXiv e-prints pp. arXiv–2006 (2020)
60. Wu, C., Wu, F., Qi, T., Huang, Y., Xie, X.: Fastformer: Additive attention can be all you need. arXiv e-prints pp. arXiv–2108 (2021)
61. Wu, G., Liu, X., Luo, K., Liu, X., Zheng, Q., Liu, S., Jiang, X., Zhai, G., Wang, W.: Accflow: Backward accumulation for long-range optical flow. arXiv preprint arXiv:2308.13133 (2023)
62. Wulff, J., Butler, D.J., Stanley, G.B., Black, M.J.: Lessons and insights from creating a synthetic optical flow benchmark. In: A. Fusiello et al. (Eds.) (ed.) ECCV

Workshop on Unsolved Problems in Optical Flow and Stereo Estimation. pp. 168–177. Part II, LNCS 7584, Springer-Verlag (Oct 2012)
63. Xu, H., Yang, J., Cai, J., Zhang, J., Tong, X.: High-Resolution Optical Flow from 1D Attention and Correlation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10498–10507 (2021), 00001
64. Xu, H., Zhang, J., Cai, J., Rezatofighi, H., Yu, F., Tao, D., Geiger, A.: Unifying flow, stereo and depth estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2023)
65. Yoon, J., Kim, S., Kwak, S., Cho, M.: Optical flow domain adaptation via target style transfer. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 2111–2121 (2024)
66. Zhai, X., Kolesnikov, A., Houlsby, N., Beyer, L.: Scaling vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12104–12113 (2022)
67. Zhang, H., Li, F., Rawlekar, S., Ahuja, N.: S3O: A Dual-Phase Approach for Reconstructing Dynamic Shape and Skeleton of Articulated Objects from Single Monocular Video. arXiv preprint arXiv:2405.12607 (2024)
68. Zhao, S., Zhao, L., Zhang, Z., Zhou, E., Metaxas, D.: Global Matching with Overlapping Attention for Optical Flow Estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17592–17601 (2022)
69. Zheng, Y., Harley, A.W., Shen, B., Wetzstein, G., Guibas, L.J.: PointOdyssey: A Large-Scale Synthetic Dataset for Long-Term Point Tracking (2023), _eprint: 2307.15055
70. Zheng, Z., Nie, N., Ling, Z., Xiong, P., Liu, J., Wang, H., Li, J.: DIP: Deep Inverse Patchmatch for High-Resolution Optical Flow. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8925–8934 (2022)