Learning Fine-Grained Features for Pixel-wise Video Correspondences Supplementary Material

Rui Li Shenglong Zhou Dong Liu
University of Science and Technology of China, Hefei, China
{liruid,slzhou96}@mail.ustc.edu.cn, dongeliu@ustc.edu.cn
https://github.com/qianduoduolr/FGVC

The supplementary material contains: 1) the detailed network architecture used in our method; 2) more details about the evaluation; 3) more training details; 4) the demo video consisting of several qualitative examples for point tracking.

1. Network Architecture

Backbone. Following the prior studies [4, 9, 11], we use the modified ResNet [3] architecture as our backbone, which is illustrated in Table 1. The res_5 layer along with the stride in res_4, is removed to get the feature maps with a resolution of 1/8 of the original video resolution. We set the stride of the encoder θ to 2 by further removing the pool_1 and the stride in res_3 to get more fine-grained features.

Self-attention and Cross-attention Layers. In our coarse-to-fine framework, after the coarse-grained features extraction, we send the coarse features $F_1^{\downarrow}, F_2^{\downarrow}$ to the self-attention and cross-attention layers, inspired by the recent studies [7,10] introducing the transformer modules to transform the features into representations that are easy to match. Formally, the attention layer is denoted as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(QK^T\right)V. \quad (1)$$

For self-attention layer, the input features are the same for $Q/K/V$, while in the cross attention layer, the input features for Q can be either $F_1^{\downarrow}$ or $F_2^{\downarrow}$, and the input features for K/V are $F_1^{\downarrow}$ (when Q is $F_2^{\downarrow}$) or $F_2^{\downarrow}$ (when Q is $F_1^{\downarrow}$). We follow the Linear Transformer [5] proposed to reduce the computational cost of transformer by substituting the exponential kernel, and we set the number of the head to 8 for the feature dimension of 256. Besides, we sequentially interleave the self and cross attention layers in our coars-to-fine framework by 2 times.

Up-sampling. In Figure 1, we illustrate the detailed architecture of the up-sampling layer which consists of several residual blocks [3] and interpolation modules. Note the first interpolation module is performed in the channel dimension, and the latter one is performed over space.

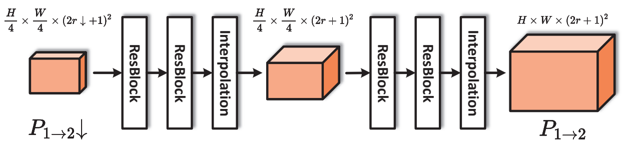
$\frac{H}{4} \times \frac{W}{4} \times (2r\downarrow+1)^2$
ResBlock
ResBlock
Interpolation
$\frac{H}{4} \times \frac{W}{4} \times (2r+1)^2$
ResBlock
ResBlock
Interpolation
$H \times W \times (2r+1)^2$
$P_{1\to2\downarrow}$
$P_{1\to2}$

Figure 1: Illustration of the up-sampling layer.

Table 1: Network Architecture of the modified ResNet-18 [3]. The residual blocks are shown in brackets and the kernel size of each convolution is presented followed by the output channels.

Stage	Network	Output size
data	-	3×256^2
conv_1	7×7, 64, stride 2	64×128^2
pool_1	3×3, 64, stride 2	64×64^2
res_2	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$, stride 1	64×64^2
res_3	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$, stride 2	128×32^2
res_4	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$, stride 1	256×32^2

2. More details about evaluation

Following previous works [4, 9, 11], all evaluation tasks are cast as video label propagation. This involves making predictions for target pixels in current frames, based solely on the ground-truth annotation for the first frame. We leverage the model as a similarity function to make predictions using k-nearest neighbors. This is a natural choice given the learned representations to align with prior research for ensuring fair comparisons. Specifically, given the labels $L_i \in \mathbb{R}^{N \times D}$ at the frame i (D represents the number of target pixels or semantic masks), and encoded features $F_i, F_j \in \mathbb{R}^{N\times C}$. We obtain the probabilistic map $P_{j\to i} \in \mathbb{R}^{N\times(2r+1)^2}$ within a local range r. We take the common practice in [4, 9, 11] to utilize the k-nearest neighbor for selecting top-k indexes of values in $P_{j\to i}(\cdot|q) \in \mathbb{R}^{(2r+1)^2}$ for each query q. Then the we obtain the selected $\widehat{P}_{j\to i} \in \mathbb{R}^{N\times k}$ and labels $\widehat{L}_i \in \mathbb{R}^{k\times D}$. The predictions $L_j \in \mathbb{R}^{N\times D}$ on the frame j can be computed as $L_j = \widehat{P}_{j\to i}\widehat{L}_i$. Moreover, we adopt a queue of context frames (memory bank) for propagation as is commonly done in previous studies, where the set of pixels of the previous m frames, are utilized for auto-regressive propagation. In our experiments of point tracking on three newly included datasets [1, 2], we set the r, k, and m to 24, 10, and 5 for all feature-matching-based methods with a stride of 2, and reduce the r to 18/12 for stride of 4/8.

3. More training details

The training of the coarse-to-fine framework is only performed on the real-world dataset YouTube-VOS [12] with a batchsize of 32, which takes 90k iterations to converge. The initial learning rate is set to 1e-3 with a cosine (half-period) learning rate schedule. In the training process, we regard the encoder θ learned before as the teacher, and we leverage the fine-grained probabilistic maps produced by the encoder θ as pseudo labels for the objective function $\mathcal{L}_{\text{KL}}$. We perform all experiments on 4 GTX-3090 GPUs.

4. Qualitative examples for point tracking

We select several representative video clips on TAP-Vid-DAVIS [2] to verify the effectiveness of our method compared with state-of-the-art methods which provide the code and pre-trained models, e.g., VFS [11], MAST [6] and RAFT [8]. Please refer to the video demo in https://www.youtube.com/watch?v=2ZCVUoiyM0U for video-wise qualitative comparisons. Without fine-tuning our pre-trained model on any additional dataset, we propagate the points of the first frame to the current frame. As observed in the enclosed video, the results produced by our method show clear improvements over state-of-the-art methods. The predictions of our method tend to have more accurate and smooth trajectories even facing severe temporal discontinuity, e.g., appearance changes, large motion, and deformations. These examples again verify the effectiveness of our method.

References

[1] Benjamin Biggs, Thomas Roddick, Andrew Fitzgibbon, and Roberto Cipolla. Creatures great and smal: Recovering the shape and motion of animals from video. In *ACCV*, pages 3–19, 2019.

[2] 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. In *NeurIPS*, 2022.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[4] Allan Jabri, Andrew Owens, and Alexei Efros. Space-time correspondence as a contrastive random walk. In *NeurIPS*, pages 19545–19560, 2020.

[5] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, pages 5156–5165, 2020.

[6] Zihang Lai, Erika Lu, and Weidi Xie. Mast: A memory-augmented self-supervised tracker. In *CVPR*, pages 6479–6488, 2020.

[7] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *CVPR*, pages 8922–8931, 2021.

[8] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, pages 402–419, 2020.

[9] Xiaolong Wang, Allan Jabri, and Alexei A Efros. Learning correspondence from the cycle-consistency of time. In *CVPR*, pages 2566–2576, 2019.

[10] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *CVPR*, pages 8121–8130, 2022.

[11] Jiarui Xu and Xiaolong Wang. Rethinking self-supervised correspondence learning: A video frame-level similarity perspective. In *ICCV*, pages 10075–10085, 2021.

[12] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327*, 2018.