Supplementary materials for MFT: Long-Term Tracking of Every Pixel

Michal Neoral, Jonáš Šerých, Jiří Matas
CMP Visual Recognition Group, Department of Cybernetics,
Faculty of Electrical Engineering, Czech Technical University in Prague
{neoramic,serycjon,matas}@fel.cvut.cz

1. Speed estimation

In Tab. 3 in the paper, we show inference speeds in a scenario of dense tracking (every pixel) on 512×512 video of 50 frames. Here, we describe how we computed the numbers. The 2.32 FPS for MFT was directly measured, including the computation time for RAFT optical flows. We have also measured the official CoTracker [4] implementation in the dense tracking configuration on 50 frame 512x512 video, resulting in 0.04 FPS. Note that CoTracker achieves better results in its default setting — tracking a single query accompanied by an auxiliary query grid at a time. This would result in even lower FPS. Both MFT and CoTracker experiments were conducted on a single GeForce RTX 2080 Ti GPU.

We did not measure the speeds of the other methods, and instead estimated them as follows. OmniMotion [5], is trained for approximately 9 hours on each sequence using an A100 GPU[1]. We have estimated the runtime by dividing the 50 frames by those 9 hours. This does not include the pre-processing time, which includes among other steps computing RAFT flows between all pairs of frames, making our estimate of 0.002 FPS optimistic.

For TAP-Net [1], PIPs [3] and TAPIR [2], we have used the timing info in the Table 9 in the appendix of [2]. This table lists the execution time of all three methods with varying number of queries and varying sequence length. All the measurements were performed on a 256×256 resolution video using a V100 GPU. As we want to access the inference speed on dense tracking on an arbitrary 512×512 video (of length 50), we have extrapolated the timing on 50 frames long video with 50 query points by multiplying the reported runtime by $512^2/50$, as if the methods would track densely in batches of 50 query points. While somewhat better parallelization should be possible, tracking all the queries at the same time is not possible due to high GPU RAM usage. Also this estimate does not include the increased computation needed to process 512×512 videos.

2. Visualisations

The supplementary video `1717-sup.mp4` demonstrates MFT performance qualitatively on video editing and style transfer applications. For the video editing examples we have inserted a logo into the first frame of each sequence. We have used the dense MFT tracks to propagate the logo pixels from the first frame throughout the sequence.

In the style transfer example we have obtained a stylized version of a single video frame. We have then transferred the stylized image using the MFT outputs to produce a temporally stable stylization of the whole video. All the examples use the MFT per-pixel trajectories directly, without any smoothing, regularization, or other post-processing techniques.

[1] https://github.com/qianqianwang68/omnimotion/tree/ad080e750a8b67cc568a79b0e7f049e420fa895a#training accesed 2023-08-30

References

[1] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adria Recasens Continente, Lucas Smaira, Yusuf Aytar, Joao Carreira, Andrew Zisserman, and Yi Yang. TAP-Vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 2022. 1

[2] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10061–10072, October 2023. 1

[3] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*, pages 59–75. Springer, 2022. 1

[4] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023. 1

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