ACVM - Correlation filter tracking

Sebastijan Trojer

I. Introduction

In this project we were tasked with implementing the MOSSE tracker and integrating it into the VOT toolkit. We were given two options for the toolkit and we used the lite version, since it's easier to integrate, so we could focus on a more detailed tracker evaluation. We evaluated the tracker on the VOT2013 dataset.

II. Experiments

A. Tracker evaluation on entire dataset.

We utilized the toolkit-lite to evaluate the tracker on the entire VOT2013 dataset [1]. Table I shows the parameters, that were used in this specific experiment and the results reported by the toolkit.

σ	α	Average overlap	Total Failures	Average speed
1	0.1	0.42	92	572
	0.2	0.43	91	649
	0.25	0.43	89	624
	0.3	0.43	92	632
1.41	0.1	0.38	84	610
	0.2	0.42	84	617
	0.25	0.41	82	593
	0.3	0.40	85	602
1.73	0.1	0.42	92	633
	0.2	**0.40**	**70**	**606**
	0.25	**0.44**	**88**	**619**
	0.3	0.43	90	626
2	0.1	0.40	95	613
	0.2	0.39	81	566
	0.25	0.42	85	597
	0.3	0.40	83	616
3	0.1	0.41	80	601
	0.2	0.39	82	621
	0.25	0.40	78	619
	0.3	0.40	85	629

Table I
Performances across different parameters. The best performing runs in terms of average overlap and total failures are shown in bold.

The best performance was achieved with σ set to 1.73 and α set to 0.2 and 0.25. There is a tradeoff between the average overlap and total failures. The λ parameter was fixed at 10^{-8} during the experiments. Figure 3 shows the performance of the tracker with respect to the parameter values. Notice that there is a big peak for the average overlap at the sigma value of 1.74, corresponding with the results in the table. The other two images attempt to show a heatmap of number of failures and average speed, respectively. An interesting insight is that the average speed seems to be very dependant on the choince of both parameters. At low learning rates, the speed is significantly lower than at middle values. Similarly, the average speed drops as we use higher σ values, which is surprising, as the gaussian filter is only computed once, therefore this parameter was not expected to have a large effect on speed. Note that the visualizations are just informative based on our results, so there might be some variance if we ran the experiments repeatedly. The most important takeaway here is the distinct peak in performance in the leftmost image, which corresponds to the drop of failures at the same σ value in the middle point, indicating that this parameter value is indeed a suitable choice.

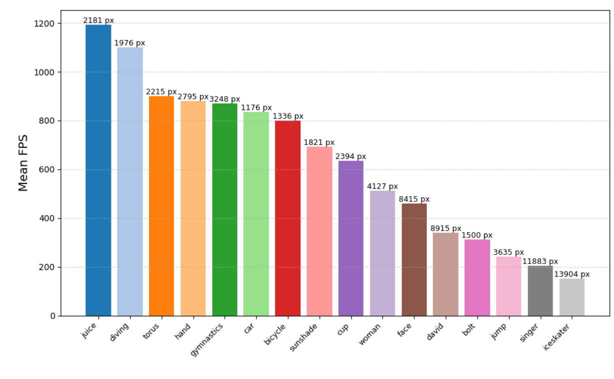

Figure 1. Mean FPS per sequence. The numbers above the bar indicate the average (initial + failed) bounding box size in pixels.

B. Tracking speed

We also evaluated the average FPS per sequence. Since we suspected that the speed is largely dependant on the bounding box size, we also recorded the mean bounding box size, when the tracker failed. The FPS per sequence and corresponding bounding box sizes are shown in Figure 1. Notice that the slowest sequences also contain the largest bounding boxes, which is expected, since there is more data to process each iteration. Figure **??** Furthermore we analyzed the processing time for the initialization and tracking, and notice that there are no significant changes, although the initialization is slightly faster, since it only constructs the filter once, whereas the tracking iteration has to construct it twice - once on the search region and once to update the filter after finding a match.

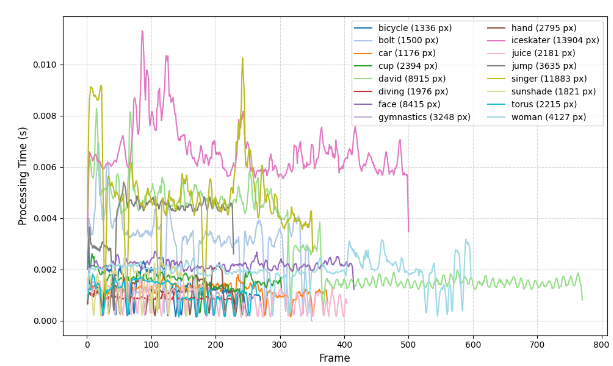

Figure 2. Per-frame tracking speed (shown in seconds) for all sequences. Notice that the re-initialization parts are indeed lower than the regular tracking times.

C. Increasing the filter region

Additionally we experimented with constructing the filter on a region bigger than the initial target. We tested multiple region multipliers, from 0.9 to 1.3. The results are given in Table II, σ and α were fixed to 1.73 and 0.2 respectively, since that gave the best results in terms of average overlap and total failures tradeoff in the first part.

If we take a closer look at the results, we notice that the increase factor seems to have the biggest impact on the processing

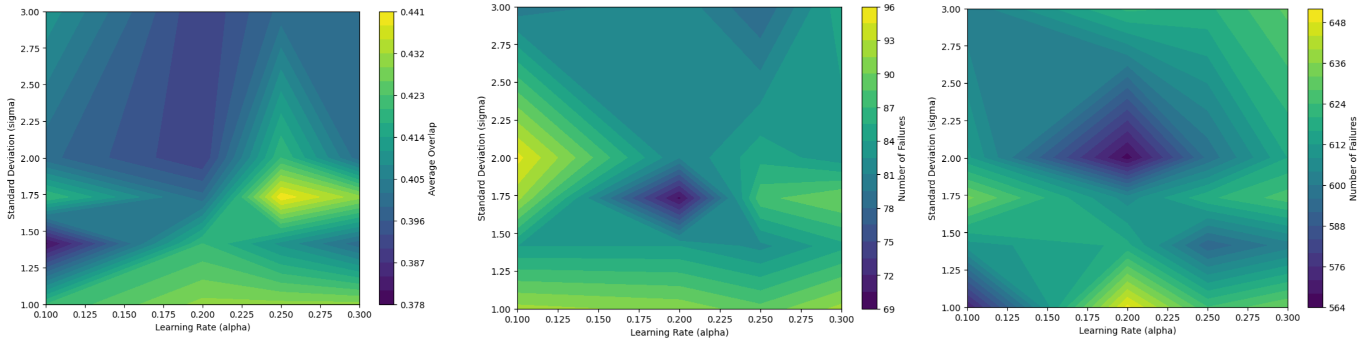

Figure 3. Visualization of the effect of parameters σ and α on different performance metrics. The leftmost image shows the average overlap with respect to the parameter choices, and higher value (brighter color) indicates better performance. Similarly, the middle image shows the number of failures and the rightmost image shows the average speed.

Increase factor	Average overlap	Total failures	Average speed
0.9	0.40	80	770
1	0.40	70	636
1.05	0.40	92	621
1.1	0.36	85	645
1.2	0.35	103	582
1.3	0.30	99	508

Table II
Results of the experiments with respect to different increase factors.

speed, rather than the overlap and failure count, however both still increase when we increase the filter size, which is in a way expected, as we also capture more noise, not just background information. We also tested an increase factor of 0.9, just to see if the performance would drop significantly, but surprisingly it performed better than even a 5% increase in the filter. We also tested different learning rates and gauss filter deviations to adjust for the new filter size, however that had no significant positive effects on the performance.

III. Conclusion

We implemented an adapted MOSSE filter and tested in on an established VOT dataset. We also integrated it into the toolkit and utilized it for robust evaluation. We considered different parameter values and compared the outcomes to determine the optimal ones. As an upgrade, we also considered scaling the filter size, however that failed to improve the performance.

References

[1] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, and G. Nebehay, "The visual object tracking vot2014 challenge results," in *Computer Vision - ECCV 2014 Workshops*, L. Agapito, M. M. Bronstein, and C. Rother, Eds. Cham: Springer International Publishing, 2015, pp. 191–217.