Correlation filter tracking

Radoslav Atanasoski

I. Introduction

This assignment introduces the basic Correlation filter Tracker (simplified version of MOSSE) and MOSSE (Minimum Output Sum of Squared Error) Tracker, which stores the numerator and denominator of the filter separately and calculates the localization using circular shift. The trackers are further integrated with a provided toolkit for evaluation for the whole VOT14 dataset, using average overlap, number of fails and frames per second for performance evaluation.

II. Experiments

For the improvement of the trackers performance, preprocessing is applied as stated in the original paper [1], where the frame (patch) is converted to grayscale and logarithmically scaled. Finally, the outputted results are subtracted by the average of the transformed frame and divided by the standard deviation. Using this pre-processing technique, the trackers performance significantly improves with slight speed (FPS) decrease.

(a) Grayscale of initialization frame (david sequence).

(b) Preprocessing of initialization frame (david sequence).

The results of the pre-processing are further multiplied with the provided cosine window (hanning), to reduce the weights around the edges and amplify the center weights.

Furthermore, for MOSSE, PSR is used, where if the PSR score drops bellow a certain threshold, the trackers learning rate (alpha) is halved, lowering the chances of the tracker updating to occlusions. Several other preventative methods we tested such as not updating the filter at all, reducing alpha by some factor. However, halving the learning rate yielded better results, thus is used throughout the experiments.

A. Results

In tables I, II and III we can see a few interesting examples of how the trackers performed under a variety of different parameter values:

- Scale factor - SF
- Epsilon/Lambda - E/L
- Learning rate - α
- PSR threshold - PSR
- Average overlap - AO
- Frames per second - FPS

The results in the tables are extracted from the provided toolkit, taking the average overlap, number of fails and frames per second for the entire sequence of data as performance measurements. The results alternate for each tracker, in order to easily see their differences in performance.

Tracker	SF	σ	E/L	α	PSR	**AO**	**Fails**	**FPS**
MOSSE	1.0	2.0	0.001	0.3	7.0	0.48	**50.0**	493.49
CF	1.0	2.0	0.001	0.3	/	0.47	69.0	564.18
MOSSE	1.0	2.0	0.001	0.0	7.0	0.48	185.0	311.19
CF	1.0	2.0	0.001	0.0	/	0.49	167.0	838.00
MOSSE	1.1	2.0	0.01	0.1	3.0	0.48	55.0	698.23
CF	1.1	2.0	0.01	0.1	/	0.46	65.0	740.01
MOSSE	1.1	2.0	0.1	0.2	5.0	0.47	58.0	648.51
CF	1.1	2.0	0.1	0.2	/	0.46	**56.0**	405.72
MOSSE	1.1	5.0	0.1	0.2	5.0	0.47	71.0	663.50
CF	1.1	5.0	0.1	0.2	/	0.47	70.0	857.76

Table I: Testing results for evaluating how σ and α affect the tracker.

From table I, we can see that the learning rate (alpha) affects the trackers performance drastically. In the first example by using an alpha of 0.3, we get 0.48 AO and 50 fails for MOSSE which are the best of all other test results. However, if we set alpha to 0, the tracker simply fails to update and produces 185 fails for MOSSE and 167 for CF. From this we can see the importance of the learning rates in the trackers. Depending on the environment, better test results were always achieved by having a learning rate (alpha) in the ranges 0.1 - 0.3. Furthermore, if sigma is too high the tracker starts performing poorly. Having sigma of 2, yields optimal results in most cases.

III. Conclusion

Tracker	SF	σ	E/L	α	PSR	**AO**	**Fails**	**FPS**
MOSSE	1.1	2.0	0.001	0.2	3.0	0.47	56.0	301.00
CF	1.1	2.0	0.001	0.2	/	0.47	64.0	651.90
MOSSE	1.3	1.0	0.01	0.1	5.0	0.47	75.0	573.28
CF	1.3	1.0	0.01	0.1	/	0.45	85.0	476.82
MOSSE	1.0	2.0	0.01	0.2	5.0	0.47	61.0	876.26
CF	1.0	2.0	0.01	0.2	/	0.47	64.0	842.13
MOSSE	1.2	2.0	0.01	0.2	5.0	0.44	59.0	559.7
CF	1.2	2.0	0.01	0.2	/	0.43	63.0	800.56
MOSSE	1.5	2.0	0.01	0.2	5.0	0.45	75.0	368.96
CF	1.5	2.0	0.01	0.2	/	0.43	79.0	355.72

Table II: Testing results for evaluating how the scale factor affects the tracker.

From table II, we can see the effect of scale factor to the trackers performance. The larger the scale, the more background is introduced into the patch. We can see that if we increase the scale factor too much, we get worse results. In some cases, by using a scale factor in the ranges 1.0 - 1.2 (low scale factor) produces the best results overall, where 1.0 means the patch is not scaled.

Model	Seq.	Fails	**Init. time**	**Track time**	FPS
MOSSE	david	0	0.00303	0.00803	124.5
CF	david	0	0.00287	0.00662	151.0
MOSSE	jogging	0	0.00116	0.00253	394.2
CF	jogging	1	0.00157	0.00230	440.2
MOSSE	bolt	1	0.00150	0.00180	561.2
CF	bolt	7	0.00146	0.00155	697.1

Table III: Comparing the initialization stage and tracking stage average execution time.

In table III, we can see the average speed to initialization frame and tracking frame. In every example, the initialization frame computation is faster then the tracking computation. In the initialization stage, the patch is extracted, pre-proccesed and the filter is constructed using Furrier transform which is very fast. In the tracking stage, the patch is extracted from the frame, localization is calculated, and the tracker is updated using a learning rate. Both stages are executed very fast due to the conversion and calculation in Furrier domain, however the initialization stage is slightly faster then the tracking stage. Also, we can see that we get very low FPS for david then the rest, the reason is that the extracted patch is much larger for david, thus requires more computation.

IV. Conclusion

In this assignment, a simplified version of MOSSE is implemented (CF) and the actual MOSSE tracker from the original paper. Furthermore, both trackers are integrated with a provided toolkit and evaluated on the VOT14 dataset, measuring their performance. The trackers are tested using a variety of different parameters, of which MOSSE achieved best results with 50 fails and CF with 56 fails throughout all sequences in the dataset. The parameters are evaluated and their affect on the trackers performance is further discussed.

References

[1] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 2544–2550.