Exercise 3: Correlation filter tracking

Veljko Dudić

I. Introduction

In the third assignment, a simplified version of the MOSSE correlation filter was implemented. The algorithm learns a filter that generates a high response for the template and a low response for the background. Additionally, computing the filter in the Fourier domain delivers an extra boost in performance.

II. Experiments

The tracker was implemented following the instructions provided in lectures and laboratory sessions. Additional improvements include converting the input patch to logarithmic space and normalizing it to achieve a mean of 0 and a standard deviation of 1. The simplified MOSSE tracker was evaluated using the pytracking-toolkit-lite [1] on the VOT2014 dataset. In the implementation, experimental results indicated that optimal performance was achieved with a fixed regularization factor of $\lambda = 0.1$. The parameters α (update rate) and σ (Gaussian parameter) were then fine-tuned.

α	σ	λ	SF	#Failures	AO	FPS
0	0.5	0.1	1.1	252	0.51	574.93
0.1	0.5	0.1	1.1	84	0.44	645.03
0.2	0.5	0.1	1.1	78	0.46	634.44
0.3	0.5	0.1	1.1	68	0.45	616.97
0.4	0.5	0.1	1.1	70	0.46	614.9
0.9	0.5	0.1	1.1	84	0.45	593.25
0.3	1	0.1	1.1	66	0.47	609.9
0.3	2	0.1	1.1	56	0.48	613.29
0.3	3	0.1	1.1	64	0.49	617.69
0.4	1	0.1	1.1	61	0.48	592.78
0.4	2	0.1	1.1	66	0.49	602.59
0.2	1	0.1	1.1	55	0.49	606.37
0.2	2	0.1	1.1	58	0.49	594.01
0.2	3	0.1	1.1	63	0.50	633.67

Table I: Performance metrics on VOT2014 with different parameters

Due to time constraints, a greedy search strategy was adopted to optimize the parameters. The process began with an initial configuration, and each parameter was individually adjusted to determine the value that yielded the best performance. Once a parameter was optimized, it was fixed while the other parameters continued to be tuned. Interestingly, although the initial configuration experiments indicated that $\alpha = 0.3$ yielded the best performance, further tuning of the remaining parameters produced superior results with $\alpha = 0.2$. In the Table I, the results for various parameter configurations are summarized. The optimal performance was achieved with a failure count of 55, an average overlap of 0.49, and a processing speed of 606.37 FPS. The experiments indicate that the parameter α exerts a greater influence on performance than σ. When $\alpha = 0$, the algorithm utilizes only the previous frame as the template for the object, leading to increased miss rates due to high variations in object appearance. Although σ does not affect performance to the same extent as α, selecting values that are either too small or too large can still degrade performance. For the experiments conducted, the optimal values for α and σ were determined to be 0.2 and 1, respectively.

With α and σ established, further experiments were carried out to determine the optimal value for the scaling factor (SF).

α	σ	λ	SF	#Failures	AO	FPS
0.2	1	0.1	1.0	62	0.46	684.71
0.2	1	0.1	1.1	55	0.49	606.37
0.2	1	0.1	1.2	67	0.50	564.14
0.2	1	0.1	1.5	81	0.51	383.44
0.2	1	0.1	1.8	84	0.50	364.28

Table II: Performance metrics on VOT2014 with different scaling factor

Table II illustrates that the optimal performance was obtained with a scaling factor (SF) of 1.1. Increasing the SF to larger values degrades the FPS performance, as larger image patches require additional computations to calculate the filters. Given that the optimal parameters depend on the specific use case, applying the best parameters for the entire dataset resulted in 2 failures for the **fernando** sequence. However, by increasing the SF to 1.5 and σ to 2, the number of failures was reduced to 1. Also, on videos where the target deforms, a lot of failures were detected.

III. Conclusion

For this assignment, a discriminative correlation filtering method essentially a simplified version of the MOSSE algorithm was implemented. Parameter evaluation and tuning were conducted on the VOC2014 dataset, yielding a configuration that resulted in 55 total failures. Future improvements may be realized by fully implementing the MOSSE paper and incorporating scale estimation into the tracking process.

References

[1] A. Lukezic, "pytracking-toolkit-lite," https://github.com/alanlukezic/pytracking-toolkit-lite, gitHub repository. Accessed: 2025-04.