Correlation filter tracking

Biljana Vitanova

I. Introduction

In this project, we implement a Correlation Filter Tracker for object tracking. We integrate our implementation into the toolkit-lite framework and evaluate its performance on the VOT2014 dataset. We also analyze the influence of key parameters on tracking accuracy, explore the impact of increasing the search region, and report tracking speed.

II. Experiments

A. Correlation Filter Tracker

In the first part, we implemented a simple Correlation Filter Tracker, where a filter is trained based on the initial appearance of the target object. This filter is constructed to produce a sharp response at the target location when applied to new frames. The filter is learned in the frequency domain and is continuously updated during tracking to adapt to appearance changes.

We integrated our tracker with the toolkit-lite framework and evaluated it on the VOT2014 dataset. The results, obtained using parameters $\sigma = 1.5$, $\lambda = 500$, and $\alpha = 0.2$, are presented in Table I.

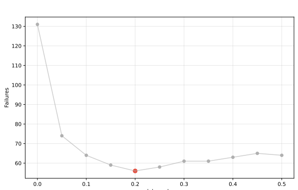
Table I: Performance of the Correlation Filter Tracker on VOT14 Dataset.

Average Overlap	**Total Failures**	**Average Speed [FPS]**
0.49	56	1201.45

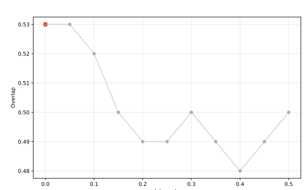
B. Parameter Analysis

We evaluated the performance of the tracker by varying different parameters. For parameters not currently under evaluation, we used the default values from the previous section.

The first parameter we tested was the update rate α, which controls how much the correlation filter H is updated with new information from each frame. A lower α makes the filter rely more on earlier frames, while a higher α allows it to adapt more quickly to recent changes in the target's appearance.



(a) Failures over α values

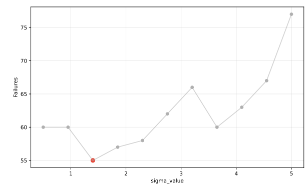


(b) Overlap over α values

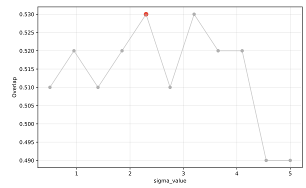
Figure 1: Effect of alpha parameter on tracking performance

From the Figure we 1. we can see that number of failures is minimum aroung 0.2 and is 56, and after that stars increasing. In this case the average overalp is not so reliable, because its maximum with the update rate 0.0 but then number of failures is 130.

The next parameter we tested was σ, which defines the Gaussian kernel or the spread of the desired response. We tested it over a range of values between 0.5 and 5.



(a) Failures over σ values



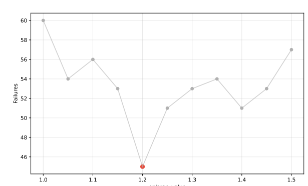
(b) Overlap over σ values

Figure 2: Effect of σ parameter on tracking performance

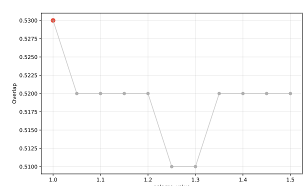
The optimal value was $\sigma = 1.5$, which resulted in 55 failures. When σ is set too high, the response becomes too wide, and the tracker is less certain about the exact position of the object which prooves the 2a. This makes it harder to localize the target accurately, which leads to more tracking failures, as seen from 2a.

C. Effect of Search Region Size

Next, we aim to improve the tracker's performance by increasing the search region. To do this, we evaluate scaling the search region from 1.0 up to 1.5.



(a) Failures over σ values



(b) Overlap over σ values

Figure 3: Effect of scale factor on tracking performance

From Figure 3a, we can see that increasing the search region does affect the tracker's performance. The best result is achieved with a scale factor around 1.2, where the number of failures drops to 44. The average overlap does not change significantly across different scale values.

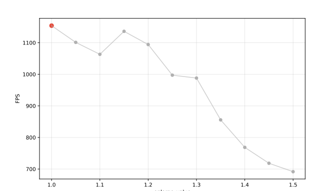


Figure 4: Average FPS across different scale factors

However, increasing the search region negatively impacts the speed of the tracker. As shown in Figure 6, the average FPS drops from around 1100 at scale 1.0 to about 700 at scale 1.5.

D. Tracking speed

We analyzed the tracking speed for each sequence to see how it varies across different videos. As shown in Figure 5, the FPS ranges from around 250 to 2250, with an average of approximately 1201.45 FPS.

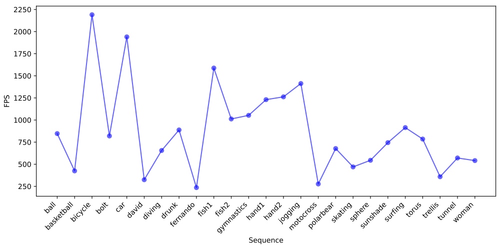


Figure 5: Average FPS across different scale factors

There is no clear pattern between the FPS and the sequence content, such as object type or motion complexity. The variation in speed is likely influenced by frame resolution or object size.

Figure 6 shows the tracking speed, measured in FPS, for each sequence, separately for the initialization and tracking steps. Intuitively, we expect the initialization step to be faster, as it usually performs a fixed set of operations (e.g., extracting a patch, computing the initial filter), while the tracking step includes additional computations like correlation response evaluation and model update. As a result, tracking tends to take longer, which means lower FPS.

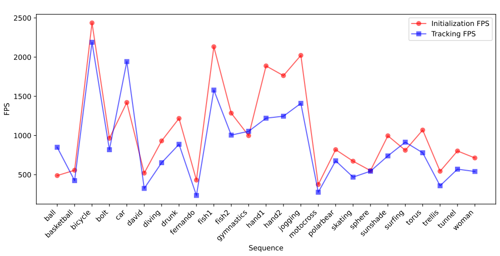


Figure 6: Average FPS across different scale factors

However, there are sequences like ball, car, surfing, and sphere where tracking FPS is higher than initialization. This might happen when the initialization includes more computationally heavy steps (e.g., filter creation in the frequency domain). Overall, the difference between initialization and tracking time varies depending on the sequence, but in most cases, initialization is faster.

III. Conclusion

In this project, we successfully implemented a correlation filter tracker and integrated it with the toolkit. We evaluated its performance under different parameters and found that using appropriate values improves performance. We showed that increasing the search region improves tracking and compared the tracking speed between initialization and regular frames across sequences, observing that initialization usually takes less time compared to tracking.