Detecting Tracking Failures From Correlation Response Maps

Ryan Walsh, Henry Medeiros

Department of Electrical and Computer Engineering, Marquette University
1551 W. Wisconsin Ave, Milwaukee WI, 53233
{ryan.w.walsh, henry.medeiros}@marquette.edu

Abstract. Tracking methods based on correlation filters have gained popularity in recent years due to their robustness to rotations, occlusions, and other challenging aspects of visual tracking. Such methods generate a confidence or response map which is used to estimate the new location of the tracked target. By examining the features of this map, important details about the tracker status can be inferred and compensatory measures can be taken in order to minimize failures. We propose an algorithm that uses the mean and entropy of this response map to prevent bad target model updates caused by problems such as occlusions and motion blur as well as to determine the size of the target search area. Quantitative experiments demonstrate that our method improves success plots over a baseline tracker that does not incorporate our failure detection mechanism.

1 Introduction

Visual object tracking is an important aspect of computer vision. Much work has gone into designing methods to improve the robustness of visual trackers against a variety of challenging conditions. Some of these obstacles include partial and full occlusion, illumination variations and pose changes as well as deformation and blurring, all of which significantly alter the target's appearance. Recent developments, particularly with the popularization of correlation filters [1,2,3] and convolutional neural networks (CNN) [4,5], have vastly improved the robustness of these systems. However, most of these approaches do not address the long-term tracking problem in which a tracker must automatically detect whether it has lost track of the target and take remedial actions. Our work attempts to improve the long-term robustness of correlation filter-based trackers against temporary target losses caused by occlusion and motion blur. Although in this work we focus on the Correlation Filter with Convolutional Features algorithm (CF2) [4], a state-of-the art visual tracker, our method is applicable to any other approach that generates a confidence map of the target position at each image frame.

We propose a solution that attempts to preclude bad updates from occurring and gives the tracker a means of recovery from hard occlusions. Using the properties of the response map (see Figure 1), we generate confidence scores from

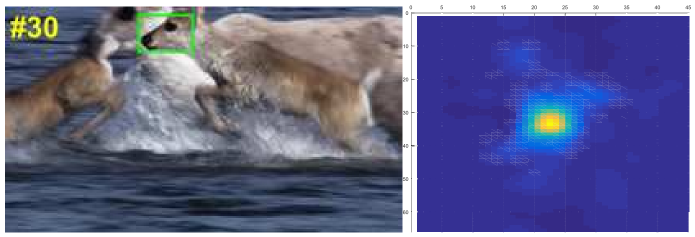


Fig. 1. Correlation filter tracker response map from CF2 over the OTB-2015 deer sequence

which we discriminate good frames from bad frames and enable or disable the target model update accordingly. We additionally use these confidence scores to readjust the search area within the image. This process is summarized in Algorithm 1. A significant observation that led to this approach was CF2's inability to handle partial occlusions without learning and having to unlearn the occluding object. While CF2 handles the majority of these events well, occlusions of extended duration hamper its performance. In certain circumstances, CF2 learns and begins to track the occluding object. By analyzing the mean and entropy of the response map over time, we were able to detect changes in the target's appearance and surroundings that may harm the tracker's ability. In addition, while CF2 is relatively insensitive to the size of target search area, dynamic variation of this parameter was critical to allow it to recover from temporary occlusions in which the position of the target changes significantly before and after the occlusion. Utilizing these concepts, our tracker is able to detect hard occlusions as well as significant appearance changes caused by motion blur.

Algorithm 1 Proposed long-term tracking algorithm with failure detection

```
    Input: Correlation filter response map
    Output: Tracker state

 1: repeat
 2:     Compute mean $\mu$ and entropy $s$ from response map according to Eqs. 1 and 2
 3:     Formulate confidence scores $D(\mu)$ and $D(s)$ according to Eqs. 3 and 4
 4:     Evaluate tracker state according to Fig. 3
 5:     Do tracker state operations according to Alg. 2
 6: until End of video sequence
```

2 Related Work

Tracking approaches based on correlation filters are often accompanied with modules to detect or mitigate tracker failures. One general approach to failure detection is to explore the consistency among multiple detectors with complementary characteristics. Multiple kernel correlation filters [2] and ensemble methods such as [6] are examples of such approaches applied to correlation filters. Deformable parts model approaches such as [7,8] also fall in this category.

Another effective failure detection mechanism is to analyze the temporal consistency along the target's trajectory. In [9], for example, the trajectory of the target is computed on a forward and on a backward pass, and the error between both trajectories is used to identify incorrect estimates. A similar approach is proposed in [10], where the distributions of a forward and a backward tracker are compared to determine reliable target features. In [11], Cordes et al. use temporal consistency to keep track of image features and improve scene structure reconstruction using bundle adjustment.

Methods that combine these two approaches are also very successful. The Multiple Experts using Entropy Minimization tracker (MEEM) [12] uses samples from several time instants to compute the cumulative confidence level of multiple classifiers and choose one among them to estimate the target position. The MULti-Store Tracker (MUSTer) [13] uses a keypoint-based approach for long term tracking and a correlation filter for short term tracking, and failures are detected according to the level of consistency among both trackers.

Finally, some approaches attempt to recognize abrupt variations in the target model to detect or avoid failures. In [14], the authors attempt to avoid tracking failures caused by motion blur by explicitly modeling the target appearance in the presence of blur of varying magnitudes and along multiple directions. Siena and Kumar [15] detect occlusions by monitoring color variations in the target model. While our approach may be considered to belong to this category, it differs from these methods in that it is the first to utilize the intrinsic characteristics of correlation filters to address tracking challenges.

3 Proposed Approach

In this section, we introduce the response map features utilized in our approach as well as the methods employed in the generation of our confidence scores. We also discuss how our proposed method handles target model learning and position update. Finally, we discuss the proposed search area scaling behavior.

3.1 Confidence Score Generation

Let f_i be an $N \times M$ matrix representing the response map from the ith frame. The mean and the negative entropy of f_i are given by

$$\mu_i = \sum_{x=1}^{N} \sum_{y=1}^{M} f_i^{(x,y)}, \tag{1}$$

$$s_i = \sum_{x=1}^{N} \sum_{y=1}^{M} f_i^{(x,y)} \log\left(f_i^{(x,y)}\right), \tag{2}$$

where $f_i^{(x,y)}$ corresponds to the yth element of the xth row of f^i. We then compute a moving weighted average over the last N frames of the column vector $x_i = [\mu_i, s_i]^T$, which includes the mean and the entropy

$$W(x_i) = \frac{\sum_{i=1}^{N} x_i i}{\sum_{i=1}^{N} i}, \tag{3}$$

where i is the frame number. Note that older frames are represented by lower values of i and more recent frames by higher values. Our confidence score vector $D(x_i) = [D(\mu_i), D(s_i)]^T$ is given by the difference between the current value of x_i and its moving average normalized by the mean of these two values:

$$D(x_i) = \frac{W(x_i) - x_i}{{}^{(W(x_i)+x_i)}\!/_{2}} \tag{4}$$

The state of the tracker can then be determined by comparing $D(\mu_i)$ and $D(s_i)$ with threshold values μ_t, s_{pt}, s_{ft} as explained in the next section.

3.2 Learning and Position Management

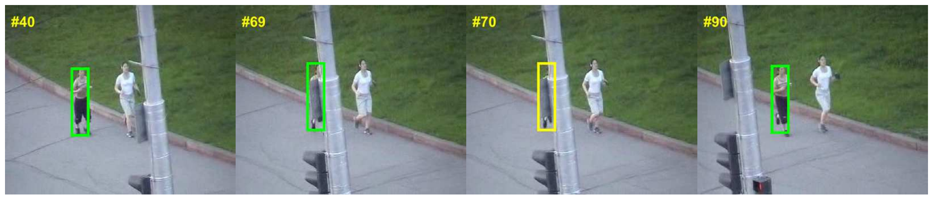
Depending on the value of confidence score $D(x_i)$, our method either i) allows the tracker to operate normally, ii) disables the target model learning mechanisms, or iii) prevents the tracker from performing bounding box updates altogether. To do so, we designed a finite state machine that toggles state based upon thresholding confidence values. The finite state machine has three primary states: *Target Found*, *Partial Loss*, and *Full Loss*. In the *Target Found* state, the tracker has a high degree of confidence and it performs its normal operations. In the *Partial Loss* state, the tracker has lost enough confidence to assume that the target is sufficiently altered or occluded to disable learning. In the *Full Loss* state, the tracker has lost a severe amount of confidence, and hence the tracker avoids learning as well as updating the bounding box position. The state transitions conditions are expressed in Figure 3.

The actions performed by our approach in each state are summarized in Algorithm 2. It is important to note that whenever the finite state machine is not in the *Target Found* state, the weighted averages are and corresponding confidence scores are not updated. Doing so prevents the expected mean and entropy from converging to the new values that the tracker should disregard. An example of our algorithm in action is shown in Figure 2.

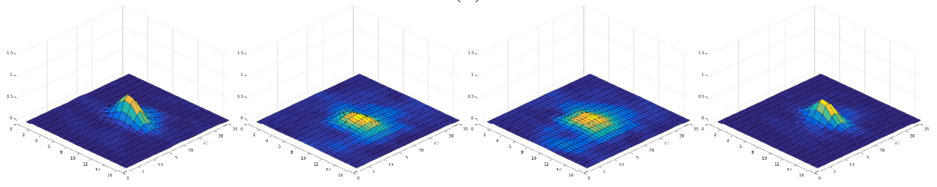
3.3 Search Area Scaling Behavior

The magnitude of the confidence score vector $D(x_i)$ is also used to determine the size of the search area M_i at a given frame as follows

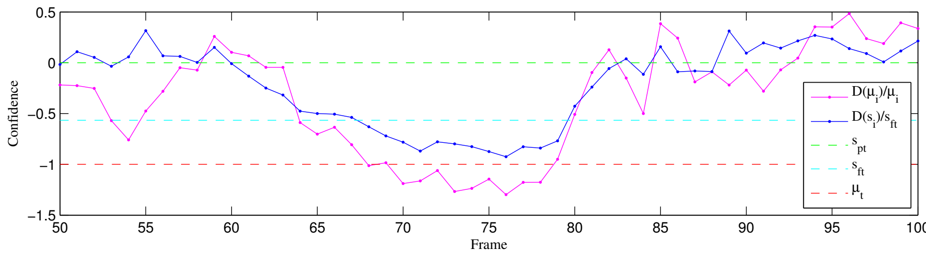
$$M_i = m_s \left[\left(\frac{\sqrt{s_m \mu_m}}{2} \right) D(x_i)^T T \right] + m_o, \tag{5}$$



(a)



(b)



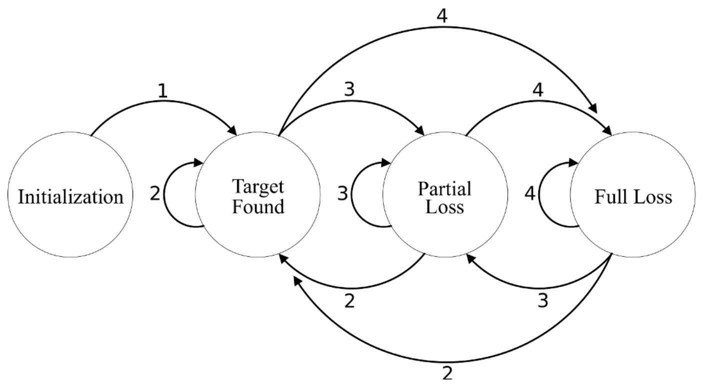
(c)

Fig. 2. Evolution of the confidence over the OTB-2015 *jogging-1* sequence. In (a), for each frame, a green bounding box shows the *Target Found* state and a yellow bounding box corresponds to a *Partial Loss* state (see Fig. 3 for a description of the states). In (b), the corresponding response maps are given. In (c) the confidence scores $D(\mu_i)$ and $D(s_i)$ are given as well as the corresponding thresholds μ_t, s_{pt}, and s_{ft}

where $T = \left[\mu_t^{-1}, s_{ft}^{-1}\right]$, μ_m and s_m are normalization constants such that the dot product $0 \le D(x_i)^T T \le 1$, and m_s and m_o specify the range of values desired for scaling the search window. That is, $m_o \le M_i \le m_o + m_s$. By multiplying the confidence vector by T, we make the search window size proportional to ${}^{D(\mu_i)}/_{\mu_t}$ and ${}^{D(s_i)}/_{s_{ft}}$. Since both thresholds are negative, the lower the confidence is with respect to its corresponding threshold, the larger the search window size.

4 Results

We evaluate our algorithm using two large and well known tracker benchmarks: OTB-2015 [16], and VOT2015 [17]. The OTB-2015 benchmark extends the traditional OTB-2013 benchmark [18] to 100 data sequences that are annotated with 11 attributes which represent challenging aspects of tracking. It benchmarks



1. Always True
2. $D(\mu) > \mu_t \vee D(s) > s_{pt}$
3. $D(\mu) \leq \mu_t \wedge D(s) \leq s_{pt} \wedge D(s) > s_{ft}$
4. $D(\mu) \leq \mu_t \wedge D(s) \leq s_{pt} \wedge D(s) \leq s_{ft}$

Fig. 3. State transition diagram for the tracker. The number next to each arrow corresponds to the transition conditions described in the corresponding item below the figure

Algorithm 2 Tracker state evaluation

```
    Input: Mean weighted average, Entropy weighted average
    Output: Tracker state

1: if State == Target Found then
2:     Update weighted averages of the mean $\mu$ and entropy $s$
3:     Estimate new target position
4:     Update correlation filters
5: else if State == Partial Loss then
6:     Estimate new target position
7: else state = Full Loss
8:     Update search area according to Eq. 5
9: end if
```

trackers against a one-pass evaluation (OPE), a spatial robustness evaluation (SRE), and a temporal robustness evaluation (TRE).[1] The VOT2015 benchmark contains 60 short sequences taken from popular and challenging datasets. Each frame of every sequence is annotated according to 5 attributes.

In OTB-2015, our method outperforms CF2 on a subset of attributes while maintaining reliability in the ones it does not. Improvements are most noticeable in sequences with low resolution and out of view portions. The four attributes in which our method shows improved performance are visualized in Figures 4

[1] Because of implementation difficulties, our evaluation excludes the *redTeam* sequence and covers only 99 of the original 100 sequences.

and 5 for the OPE and SRE metrics. The improvement on the TRE metric was negligible (0.21%) and the corresponding graphs are omitted. The corresponding improvements in the overall dataset are modest, however, as sequences marked with low resolution and out of view attributes account for only 8% and 14% respectively of the entire benchmark. As a result, these gains are diluted in the overall results, as shown in Table 1.

We found that in the OTB evaluation, our method improves performance against several tracking challenges without negatively affecting the general performance of the tracker. Specifically, our tracker improves the SRE performance of most sequences and increases the chance that when the tracker loses track of the target, it will successfully recover later in the sequence. This performance gain is best shown in the *Biker*, *Bird1*, *Car1*, *Jogging-2*, and *Freeman1* sequences, where our scaling method helps prevents a target loss. These improvements extend to scenes where a quick occlusion or change in motion moves the target off track or blurs the target. Some of the sequences in which our method generated significant performance improvements are illustrated in Figure 6.

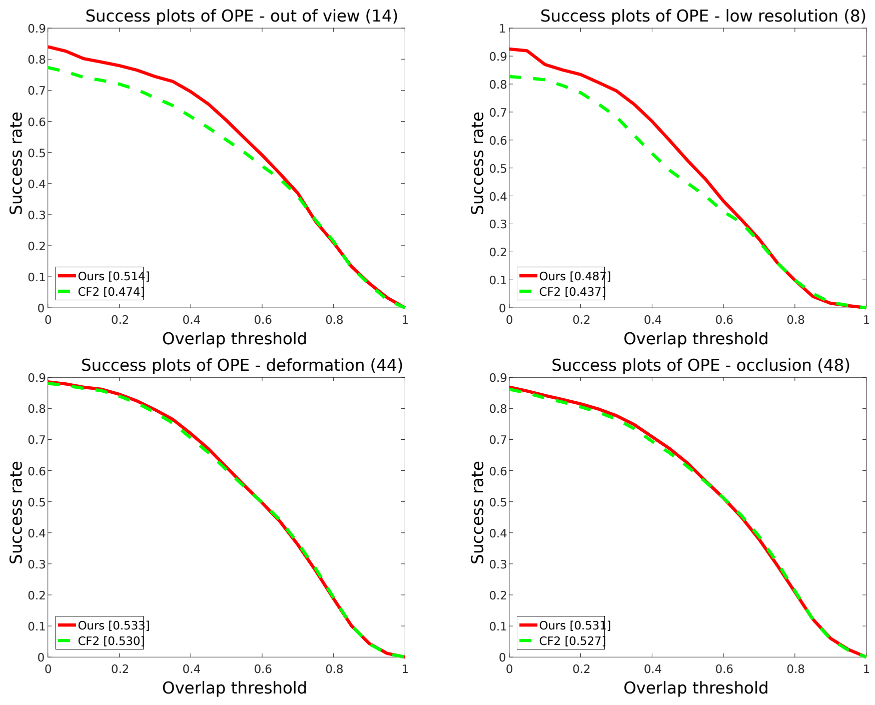


Fig. 4. OPE evaluation in the OTB-2015 dataset showing attributes in which our method outperforms the baseline tracker

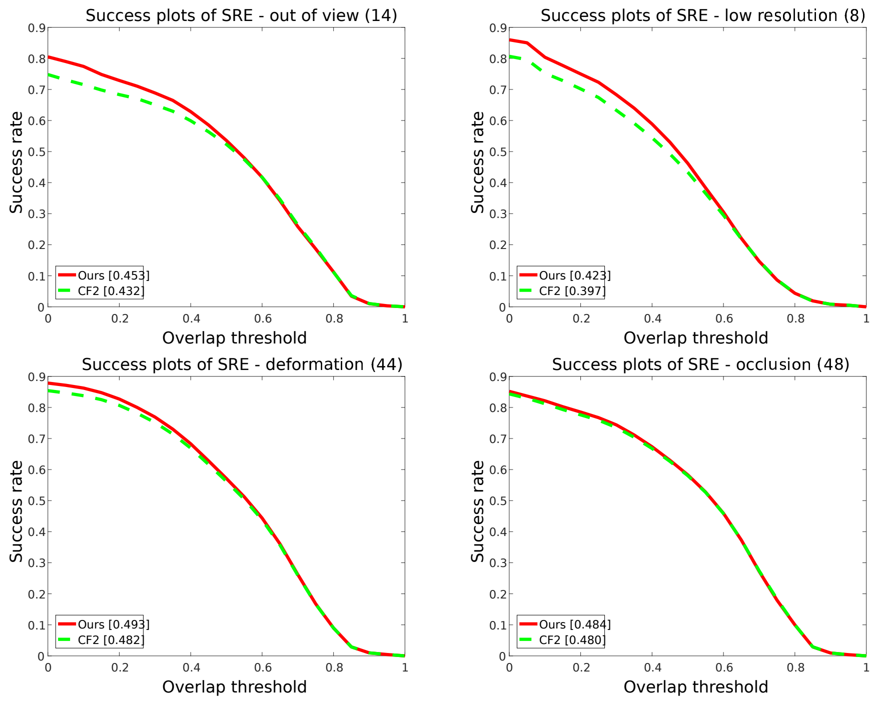


Fig. 5. SRE evaluation in the OTB-2015 dataset showing attributes in which our method outperforms the baseline tracker

Table 1. Relative overall performance change in the OTB-2015 dataset with respect to the baseline method. The modest gains are caused by the fact that the sequences in which our method shows the greatest performance improvements correspond to a small portion of the overall dataset

	OPE (%)	SRE (%)	TRE (%)
Precision	1.6	0.56	0.13
Success	0.36	0.39	-0.17

In the VOT2015 evaluations, both methods perform similarly among most sequences, but our method shows slight reductions in accuracy, robustness, and expected overlap of 0.07, 0.01, and 0.008, respectively. We found, however, that these reductions are sometimes due to the evaluation method used by the VOT2015 testbench, which reinitializes trackers whenever a tracking failure occurs. In some sequences, these reinitializations penalize our method's ability to keep track of a target longer, even if with a lower accuracy. An example of this occurs in the *fish3* sequence where our tracker is able to track the target for the entire duration of the sequence, but with some drift towards the end. The baseline tracker fails early in the sequence and is accurately reinitialized with the

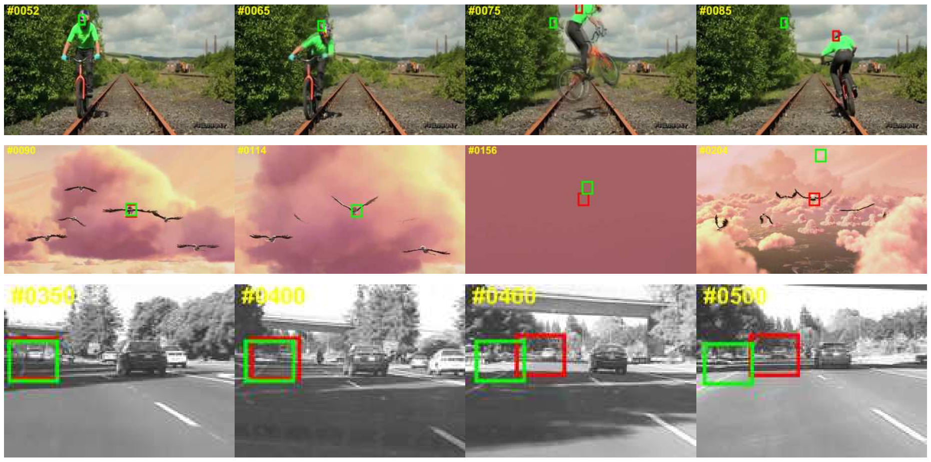


Fig. 6. Select OTB-2015 sequences where our method significantly outperforms the baseline tracker in SRE and or TRE evaluation: From top to bottom, the sequences are *biker*, *bird1*, and *car1*. CF2 is represented by the green bounding box and our approach by the red one. Other sequences in which our tracker shows substantial performance improvement such as *faceOcc1*, *freeman1*, *jogging-2*, and *twinnings* and not shown here due to space constraints

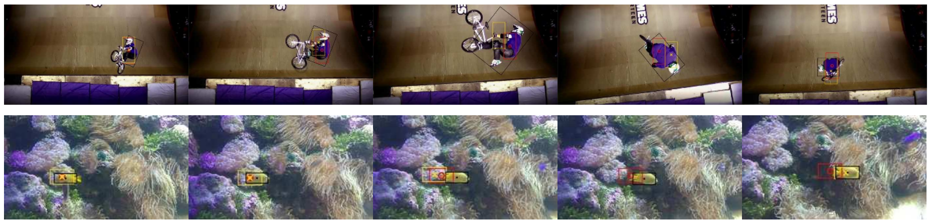


Fig. 7. Select VOT2015 sequences where our method outperforms the base tracker. From top to bottom, the sequences are *bmx* and *fish3*. CF2 is represented by the yellow bounding box, our approach by the red bounding box, and the ground truth by the black one

ground truth allowing it to achieve a better precision score and essentially penalizing our tracker for not losing track of the target. We provide relevant examples in Figure 7, which illustrate highlights and instances where the benchmark made it difficult make divisive conclusions.

4.1 Failure Cases

In Figure 8, we provide examples where our tracker performs worse than the baseline tracker. In the *Bolt2* sequence, low confidence scores in the initial frames cause the tracker's search space to grow until it finds a similar object nearby

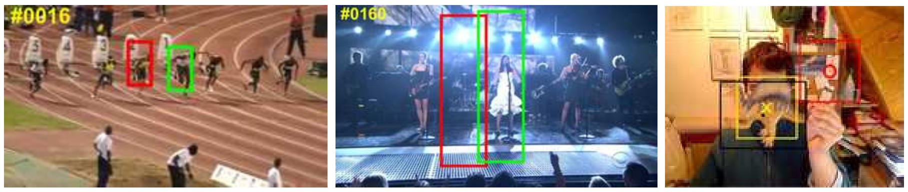


Fig. 8. Snapshots of the OTB and VOT sequences in which CF2 outperforms our method: The leftmost and the center sequences are OTB's *bolt2* and *singer1* sequences. CF2 is again the green bounding box and our approach is the red box. The leftmost figure corresponds to the VOT2015 *dinosaur* sequence where CF2 is shown in yellow, our method in red, and the groundtruth in black

which it ultimately begins to track. For the *Singer1* sequence, a large bounding box, which includes a substantial portion of the background, coupled with a scale change lowers our method's confidence and consequently causes it to expand its search space. The combination of these features spreads the convolution sampling thin and loses the target. In VOT2015's *dinosaur* sequence, the initial target also includes a significant portion of the background, and its clutter causes large swings in the response map. As a result, our method generates low confidence scores and consequently loses the target.

5 Conclusions

This work presents a failure detection module applied to a correlation filter tracker. This module utilizes the mean and entropy of the response map to generate confidence scores, which in turn, are used to scale the search space and control the learning behavior of the base tracker. Our results show that our method improves the base tracker's reliability with respect to blurry motion and abrupt occlusions.

Overall we see that there is room for improvement in how we generate our confidence scores. While the mean and the entropy are good indicators of tracking failures, additional metrics of the spread of the correlation maps, which are more robust to background clutter, are needed to address the issues discussed in Section 4.1. One possibility is to additionally weight our metrics by a radial function that makes it less susceptible to background clutter around the target.

References

1. Chen, Z., Hong, Z., Tao, D.: An experimental survey on correlation filter-based tracking. arXiv preprint arXiv:1509.05520 (2015)
2. Tang, M., Feng, J.: Multi-kernel correlation filter for visual tracking. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 3038–3046

3. Danelljan, M., Hager, G., Shahbaz Khan, F., Felsberg, M.: Learning spatially regularized correlation filters for visual tracking. In: The IEEE International Conference on Computer Vision (ICCV). (2015)
4. Ma, C., Huang, J.B., Yang, X., Yang, M.H.: Hierarchical convolutional features for visual tracking. In: The IEEE International Conference on Computer Vision (ICCV). (2015)
5. Nam, H., Han, B.: Learning multi-domain convolutional neural networks for visual tracking. arXiv preprint arXiv:1510.07945 (2015)
6. Zhu, G., Wang, J., Lu, H.: Clustering based ensemble correlation tracking. Computer Vision and Image Understanding (2016)
7. Lukežič, A., Čehovin, L., Kristan, M.: Deformable parts correlation filters for robust visual tracking. arXiv preprint arXiv:1605.03720 (2016)
8. Akin, O., Erdem, E., Erdem, A., Mikolajczyk, K.: Deformable part-based tracking by coupled global and local correlation filters. Journal of Visual Communication and Image Representation **38** (2016) 763–774
9. Kalal, Z., Mikolajczyk, K., Matas, J.: Forward-backward error: Automatic detection of tracking failures. In: Pattern recognition (ICPR), 2010 20th international conference on, IEEE (2010) 2756–2759
10. Biresaw, T.A., Alvarez, M.S., Regazzoni, C.S.: Online failure detection and correction for bayesian sparse feature-based object tracking. In: Advanced Video and Signal-Based Surveillance (AVSS), 2011 8th IEEE International Conference on, IEEE (2011) 320–324
11. Cordes, K., Müller, O., Rosenhahn, B., Ostermann, J.: Feature trajectory retrieval with application to accurate structure and motion recovery. In: International Symposium on Visual Computing, Springer (2011) 156–167
12. Zhang, J., Ma, S., Sclaroff, S.: Meem: robust tracking via multiple experts using entropy minimization. In: European Conference on Computer Vision, Springer (2014) 188–203
13. Hong, Z., Chen, Z., Wang, C., Mei, X., Prokhorov, D., Tao, D.: Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 749–758
14. Wu, Y., Hu, J., Li, F., Cheng, E., Yu, J., Ling, H.: Kernel-based motion-blurred target tracking. In: International Symposium on Visual Computing, Springer (2011) 486–495
15. Siena, S., Kumar, B.V.: Detecting occlusion from color information to improve visual tracking. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE (2016) 1110–1114
16. Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence **37** (2015) 1834–1848
17. Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Cehovin, L., Fernandez, G., Vojir, T., Hager, G., Nebehay, G., Pflugfelder, R.: The visual object tracking VOT2015 challenge results. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. (2015) 1–23
18. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2013) 2411–2418