REAL-TIME TRACKER WITH FAST RECOVERY FROM TARGET LOSS

Alessandro Bay[⋆]*, Panagiotis Sidiropoulos*[⋆†]*, Eduard Vazquez*[⋆]*, Michele Sasdelli*[⋆♯]

[⋆]Cortexica Vision System Ltd, London, UK
[†]Space and Climate Physics Department, University College London, UK
[♯]Australian Institute for Machine Learning, The University of Adelaide, Australia

ABSTRACT

In this paper, we introduce a variation of a state-of-the-art real-time tracker (CFNet), which adds to the original algorithm robustness to target loss without a significant computational overhead. The new method is based on the assumption that the feature map can be used to estimate the tracking confidence more accurately. When the confidence is low, we avoid updating the object's position through the feature map; instead, the tracker passes to a single-frame failure mode, during which the patch's low-level visual content is used to swiftly update the object's position, before recovering from the target loss in the next frame. The experimental evidence provided by evaluating the method on several tracking datasets validates both the theoretical assumption that the feature map is associated to tracking confidence, and that the proposed implementation can achieve target recovery in multiple scenarios, without compromising the real-time performance.

***Index Terms*—** Real-time tracking, Siamese convolutional neural networks, Correlation filters, Target loss recovery, Census transform

1. INTRODUCTION

The recent advances in deep learning research has significantly improved the state-of-the-art in a multitude of computer vision and multimedia processing applications such as object detection [1], semantic segmentation [2], action recognition [3], etc. However, in tracking applications deep learning initially struggled to exhibit performance improvements [4]. Only recently deep learning has significantly improved the state-of-the-art, both for real-time [5, 6] and non-real-time tracking [7, 8].

A methodological modification that greatly benefitted deep-learning trackers was the replacement of the off-line training on large classification datasets with either on-line training (e.g. ECO [9]) or with off-line training on an image retrieval (and not classification) setup (e.g. CFNet [6]). Each of these two classes of algorithms achieve state-of-the-art performance in one of the two main video tracking sub-categories. More specifically, algorithms using on-line training are optimal in non-real-time tracking and algorithms using image retrieval training in real-time tracking.

However, several significant challenges remain unresolved, including the recovery from target loss [10] and the sensitivity to distractors [11]. This work aims to reduce the sensitivity of the state-of-the-art CFNet tracker from target loss without significantly reducing its speed. The difficulty of tracker recovery originates from its main design principle, i.e. its ability to accumulate correct object positions for a substantial amount of time. This ability becomes a severe flaw in the cases that the tracker would temporarily lose the position of the object due to an abrupt camera movement, an unexpected and abrupt object movement, a technical problem, a compression error, etc. Once the sampling drift [10] causes the tracker bounding box to not intersect with the object, the tracker capability to accumulate correct retrievals "locks" the object in the background with little possibility of recovery.

The tracking-through-similarity paradigm that is adopted by CFNet [6] provides a mechanism that can be used for on-line identification of target loss. More specifically, in this work we follow the common (e.g. [12]) nearest neighbour distance ratio (NNDR) to declare ambiguous or unambiguous tracking updates. If the CFNet output is declared ambiguous the tracker is entering a 1-frame "failure mode", during which (1) the tracker is updated using a backup tracker and (2) the search area is doubled to facilitate the object redetection in the next frame. The implementation of the architecture introduced in this paper is named CFNet-FTLR (Fast Target Loss Recovery) and uses a simple low-level visual feature correlation scheme as a backup tracker. Apart from the original architecture, the main contributions of the paper include (1) the ambiguity measure that estimates the confidence on the tracker output, (2) the enlargement of the search area when the tracker is on failure mode, (3) the use of a simple low-level representation as a short-term backup tracker, and (4) an improved running average equation for the query model.

2. RELATED WORK

Due to its major significance, the tracking of moving objects in videos has a long history of research and development.

The matureness of the domain partially explains the failure of deep-learning trackers to outperform "classical" trackers [4]. Perhaps a more important reason is the inherent characteristics of the tracking setup that undermined a straightforward transfer of deep-learning techniques to this task [13]. More specifically, (1) maximising heatmaps corresponding to semantic classes is not necessarily the optimal strategy to locate a specific object [14] (especially in the presence of distractors), (2) off-line training is hampered by the lack of large-volume annotated datasets for video tracking, and (3) on-line training is prohibitively slow for applications requiring real-time tracking, especially if the model update is conducted in each and every frame [9, 10].

On the other hand, the small temporal window between two consecutive frames implies a visual similarity of the tracked object. Based on this rationale, a tracking-by-similarity tracker has been introduced [5], in which similarity is learned through a Siamese deep network that is trained offline, while the localisation is conducted through a correlation filter [6]. This method achieved state-of-the-art real-time performance, despite its simple architecture.

One of the main issues with such a tracker is the sampling drift [10], from which the tracker often fails to recover, that occurs in the case of abrupt object motion, camera motion, etc. Two main solutions have been recently proposed: (1) the heatmap generated in the final stage of the algorithm can incorporate the localisation of the ambiguity by not exhibiting a clear peak and in this case it rejects the position update [15], and (2) the model drift can be avoided through a global similarity search (in the whole frame) every N frames (N being a hard-coded constant parameter) [10].

In this work, we carefully combine the above solutions in order to optimise both the accuracy and the computational time. The localisation ambiguity is detected using the Nearest Neighbour Distance Ratio (NNDR), instead of through a peak comparison to a hard-coded threshold. After rejecting an ambiguous position update, tracking is conducted in this frame using correlation of low-level patch representations as a backup tracker. This is inspired by the recent improvements on tracker performance that was achieved by including the first network layers, which model the low-level image content [8, 9]. Finally, in the next frame, the search window is expanded, but without covering the whole frame, as in [10], so as to reduce the computational cost.

3. CFNET-FTLR TRACKER

The main concept of our CFNet-Fast Target Loss Recovery (CFNet-FTLR, Figure 1) tracker lies in the assumption that the CFNet feature map can be used to evaluate the confidence on the bounding box update. The confidence is modelled through the ratio of the two most dominant peaks, which are estimated by projecting the 3D map onto yz and xz planes, before being differentiated twice in order to identify the local maxima, which determine the two most dominant peaks. Since CFNet is estimating patch similarity, these peaks correspond to the nearest neighbour and the second nearest neighbour of the current bounding box.

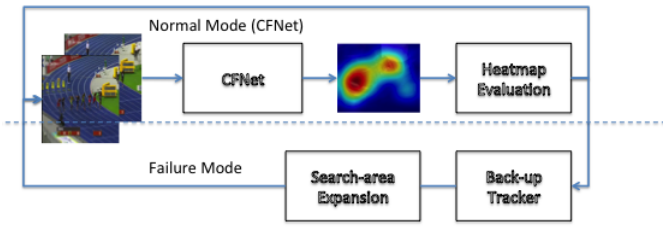

Fig. 1. The architecture of the CFNet-FTLR tracker

Their ratio (i.e. NNDR) is thresholded to evaluate the confidence on the tracker output. More specifically, if the ratio is above the confidence threshold, then the tracker output is considered safe. Therefore, the top peak is followed to update the object position and the tracking continues following the standard CFNet algorithm in the next frame. On the other hand, if the ratio is below the confidence threshold, the tracker output is considered ambiguous and the tracker passes to failure mode.

The following measures are taken during the time that the tracker is on failure mode: (1) CFNet output is not used to update the object position (the top peak position is ignored), (2) the object position is updated following the estimation of the backup tracker, and (3) in the following frame the object is searched in an area which is wider than the original one.

The backup tracker is based on correlating Census-transformed [16] image patches. Census transform is a simple and powerful low-level representation of the image content that holds a set of desirable features: (1) it has linear computational complexity, (2) it preserves the object edges, (3) it is robust to radiometric differences that are not uncommon in videos [17], and (4) it generates robust optical flow estimations [18]. Moreover, from a deep-learning point of view, Census transform could be considered as a hand-crafted filter of the first layer of a neural network, i.e. it fits to the recent results implying that low-level visual information can contribute to the overall tracker performance. The 8-bit binary strings that the Census transform generates for each pixel are converted into 4 decimal numbers by iteratively applying a circular shift of 2 positions before conversion. The result is correlated and the position of the maximum value is followed to update the object bounding box.

Additionally, the standard approach used in CFNet [5] to create the query model from the previously seen feature map is a simple running average:

$$Q_1 = F_1, \qquad Q_n = (1-\alpha)Q_{n-1} + \alpha F_n, \qquad (1)$$

where Q_n is the n-th query model, F_n is the n-th feature map, and the update factor α is empirically set to $\alpha = 0.005$.

An unintentional result of Eq. (1) is that during the initial video frames (in which case $n \gg 1/\alpha$), Q_n is dominated by the first feature map F_1. For example, if $\alpha = 0.005$ and the frame ratio is 25 fps, then 8 seconds after the video started F_1 would determine 37% of the generated query feature map. In the general case, such a strong dependence from F_1 is expected to be suboptimal, especially in practical applications, where the tracked object would be initialised from an arbitrary pose, and frequently from a frame of poor visible quality.

Therefore, it is suggested to reduce the dependence from the first frame, by replacing Eq. (1) with the following running average formula:

$$Q_n = (1 - 0.5/n - \alpha)Q_{n-1} + (0.5/n + \alpha)F_n. \quad (2)$$

Eqs. (1) and (2) converge asymptotically. Their main difference is that Eq. (2) generates a "smooth average" (SA) over the first frames, by creating a bootstrap model which uses a large number of initial frames ($\sim \frac{1}{5\alpha}$), instead of a single frame. This improves significantly the performance in the more challenging TRE evaluation, a benchmark that measures the robustness of the tracker on the initial object pose. The introduced algorithm (CFNet-FTLR_SA) is analytically presented in Algorithm 1.

Algorithm 1 CFNet-FTLR_SA tracker($video$, b_0, $NNDR$, $DefaultImageArea$)

1: (Initialisation) $ImageArea = DefaultImageArea$
2: Select the next frame and crop a patch of size equal to $ImageArea$ around object region
3: Resize the patch to fit the input size of the Neural Network (NN)
4: Forward pass the patch to the NN and estimate the feature map
5: Estimate the correlation map from the feature map and the existing running average
6: Estimate the two highest peaks in the correlation map, P_1 and P_2
7: **if** $P_1/P_2 > NNDR$ **then**
8: (a) update the position
9: (b) the running average using Eq. (2)
10: (c) put $ImageArea = DefaultImageArea$
11: **else**
12: (a) update the position after running the backup tracker
13: (b) put $ImageArea = 2 * DefaultImageArea$
14: **if** this is the last video frame **then return**

4. EXPERIMENTAL RESULTS

4.1. Implementation Details and details

Using the CFNet implementation provided by the authors, we have gradually augmented it with more modifications in order to confirm the contribution of all parts of the algorithm. Therefore, apart from the final algorithm CFNet-FTLR_SA, as well as the original CFNet method, there are three more variations experimentally evaluated: (1) CFNet-FTLR_0, in which the position of the bounding box is not updated until the NNDR is above the confidence threshold, but the search area for the subsequent frames is twice the original one, (2) CFNet-FTLR_1, in which the new position is extrapolated by the position of the object in the past two frames using bilinear interpolation, and (3) CFNet-FTLR, in which the smooth average is not included in the algorithm.

We additionally tested the validity of our basic assumption, i.e. that the feature map NNDR is associated with tracking ambiguity, by evaluating also a "theoretic tracker", which achieves the best performance of a tracker using the normal mode/failure mode architecture. This "theoretic tracker" is modelled by replacing the backup tracker with a "perfect" one that always returns the ground truth, i.e. by feeding the ground truth as the new object position whenever an ambiguity is identified. The performance obtained by this "tracker" (named CFNet-FTLR_GT) is the upper limit that can be reached by the introduced architecture, while CFNet-FTLR_SA is a first implementation towards this direction.

For evaluating our algorithm, we use three tracking benchmarks: (a) OTB-100, OTB-50 and OTB-2013 [6] datasets created from the Object Tracking Benchmark (OTB) [19], (b) UAV-123 [20], which contains videos collected from from low-altitude Unmanned Aerial Vehicles, and (c) a unified tracking benchmark on drone platforms (DTB-70) [21]. For all benchmarks we report the results for One-Pass Evaluation (OPE) and Temporal Robustness Evaluation (TRE), following the standard literature approach (e.g. [11]), while the tracker speed (in fps) is also reported for each method.

All of the calculations were performed with MATLAB R2017a, MatConvNet 1.0-beta25, Cuda-8.0, Cudnn-5.1, on a i7-6800K CPU @ 3.40GHz $\times$ 12 workstation with 32 GB RAM and a single nVidia GeForce GTX 1080Ti GPU.

4.2. Results

Tables 1, 2 and 3 summarise the performance for both OPE and TRE evaluation, comparing our methods with the subset of literature algorithms that also perform real-time on-line tracking.Note that the Tables include the theoretic boundary of the performance, which is modelled by CFNet-FTLR_GT, while for the OTB dataset, CFNet-FTLR_0 and CFNet-FTLR_1 are also evaluated.

A first conclusion that can be reached from these results is related to the validity of the assumption that the object ambiguity is correlated with a feature map presenting multiple peaks. More specifically, CFNet-FTLR_GT implies that a CFNet variation exploiting this property using a fast and accurate backup tracker could achieve accuracy and precision improvement as high as 13% and 24.5%, respectively (OPE, DTB-70 dataset). The median accuracy and precision absolute improvements are 5.7% and 10.3%, respectively, a substantial increase which would bring CFNet near the performance currently achieved only by non-real-time trackers.

On the other hand, comparing CFNet-FTLR_SA with the original CFNet algorithm shows that our variation consistently outperforms the original CFNet at a computational

		OTB-2013				OTB-50				OTB-100			
		OPE		TRE		OPE		TRE		OPE		TRE	
Method	fps	IoU	prec	IoU	prec	IoU	prec	IoU	prec	IoU	prec	IoU	prec
CFNet [6]	71.1	57.0	74.1	59.9	76.1	49.4	64.3	52.9	69.0	55.7	71.6	58.2	73.7
CFNet-FTLR_0	64.2	57.4	74.4	60.2	76.5	49.1	64.1	53.0	69.2	54.6	70.3	58.0	73.4
CFNet-FTLR_1	64.0	57.9	75.2	60.2	76.5	48.9	63.8	53.0	69.0	54.8	69.9	58.1	73.6
CFNet-FTLR	61.6	**60.0**	**78.2**	60.6	76.9	51.3	68.1	53.9	70.7	**57.3**	74.1	58.7	74.4
CFNet-FTLR_SA	62.3	58.7	76.8	**61.9**	**79.8**	**51.5**	**68.5**	**55.0**	**73.2**	57.1	**74.6**	**60.3**	**77.8**
CFNet-FTLR_GT	65.0	62.7	83.6	65.3	84.6	54.6	74.6	58.7	79.3	59.3	78.2	62.5	80.8

Table 1. The results on OTB dataset for our baseline (CFNet) along with the 4 tested variations of FTLR: FTLR0, FTLR1, FTLR and FTLR with smooth average (FTLR_SA). The best performance is highlighted in bold.

	OPE		TRE	
Method	IoU	prec	IoU	prec
KCF [22]	33.1	52.3	–	–
DSST [23]	35.6	58.6	–	–
CFNet [6]	47.0	66.5	52.4	72.2
CFNet-FTLR	47.2	67.0	**53.3**	73.7
CFNet-FTLR_SA	**47.6**	**67.6**	52.9	**73.8**
CFNet-FTLR_GT	54.9	80.7	59.2	84.5

Table 2. The results on UAV-123 dataset for our baseline (CFNet) along with two FTLR variations. Two more literature, on-line and real-time, methods are included in the comparison. The best performance is highlighted in bold.

	OPE		TRE	
Method	IoU	prec	IoU	prec
DSST [23]	26.4	40.2	–	–
KCF [22]	28.0	46.8	–	–
CFNet [6]	39.4	57.9	48.1	67.2
CFNet-FTLR	41.2	61.3	49.1	68.7
CFNet-FTLR_SA	**43.0**	**64.5**	**50.7**	**72.2**
CFNet-FTLR_GT	52.4	82.2	56.3	81.9

Table 3. The results on DTB-70 dataset for our baseline (CFNet) along with two FTLR variations. Two more literature, on-line and real-time, methods are included in the comparison. The best performance is highlighted in bold.

overhead (CFNet-FTLR_SA runs at 62.3 fps while CFNet at 71.1 fps) that for most applications would be considered negligible. The median accuracy and precision increase is 2.1% and 4.1%, respectively, while for the OPE evaluation in the DTB-70 dataset it is 3.6% and 6.6%.

By comparing the achieved improvement with the theoretic upper boundary, it can be deduced that CFNet-FTLR_SA exploits approximately 40% of the additional accuracy and precision that the use of NNDR allows. This is far from optimal, but it should be contrasted with simple solutions such as CFNet-FTLR_0 and CFNet-FTLR_1, which completely fail to improve the results. As a matter of fact, both CFNet-FTLR_0 and CFNet-FTLR_1 median accuracy and precision is almost identical to CFNet, exhibiting performance deterioration for half of the evaluations that have been tested.

Tables 1, 2 and 3 include not only CFNet but also all real-time on-line trackers that we could find to report results on the same datasets. In all datasets included in this work, the CFNet-FTLR_SA variation exhibits state-of-the-art performance, since it outperforms all included on-line real-time literature techniques. On the other hand, if this method is compared with non-real-time state-of-the-art trackers, it seems that it exhibits worse performance than the best current non-real-time trackers (such as MDNet [4] in the DTB-70 dataset). However, the performance gap between MDNet and the original CFNet was 6.2% and 11.1% in success and precision, respectively. By using CFNet-FTLR_SA, instead, this performance gap (which can be viewed as the performance gap between non-real-time and real-time trackers) is reduced to 2.6% and 4.5%.

Finally, the comparison between CFNet-FTLR_SA and CFNet-FTLR validates the analysis conducted in the previous section. CFNet-FTLR_SA outperforms CFNet-FTLR in all but four (OPE and TRE) evaluations, while it shows a slightly better performance in TRE evaluation than in OPE evaluation. This is aligned with the dependence of the TRE from the object viewing angle (in the original frame), thus signifying that in most practical applications (in which the object viewing angle in the first frame is not generally known) CFNet-FTLR_SA should be preferred.

5. CONCLUSIONS

In this work, we examined the hypothesis that in tracking-through-similarity algorithms the output heatmap that determines the object position in the next frame could be used to detect possible sampling drift. Moreover, we introduced an algorithm based on this hypothesis that employs a 1-frame backup tracker to temporarily update the object position, thus allowing the tracker to recover from target loss in subsequent frames. The experimental results in three distinct datasets confirm the potential of the introduced method.

6. REFERENCES

[1] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[2] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[3] Karen Simonyan and Andrew Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in neural information processing systems*, 2014, pp. 568–576.

[4] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[5] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr, "Fully-convolutional siamese networks for object tracking," in *European conference on computer vision*. Springer, 2016, pp. 850–865.

[6] Jack Valmadre, Luca Bertinetto, João Henriques, Andrea Vedaldi, and Philip HS Torr, "End-to-end representation learning for correlation filter based tracking," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 5000–5008.

[7] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Y. Choi, "Action-decision networks for visual tracking with deep reinforcement learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[8] M. Danelljan, A. Robinson, F. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *IEEE European conference on computer vision*, 2016.

[9] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "Eco: Efficient convolution operators for tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 6931–6939.

[10] R. Tao, E. Gavves, and A. W.M. Smeulders, "Tracking for half an hour," in *arXiv preprint arXiv:1711.10217*, 2017.

[11] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *IEEE International conference on computer vision*, 2015.

[12] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[13] D. Zhang, H. Maei, X. Wang, and Y.-F. Wang, "Deep reinforcement learning for visual object tracking in videos," in *arXiv preprint arXiv:1701.08936*, 2017.

[14] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *IEEE International conference on computer vision*, 2015.

[15] J. Valmadre, L. Bertinetto, J. F. Henriques, R. Tao, A. Vedaldi, A. Smeulders, P. Torr, and E. Gavves, "Long-term tracking in the wild: A benchmark," in *arXiv preprint arXiv:1803.09502*, 2018.

[16] R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondence," in *IEEE European conference on computer vision*, 1994, pp. 151–158.

[17] H. Hirschmuller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 9, pp. 1582–1599, 2009.

[18] D. Hafner, O. Demetz, and J. Weickert, "Why is the census transform good for robust optic flow computation," in *Scale Space and Variational Methods in Computer Vision*, 2013.

[19] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang, "Online object tracking: A benchmark," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[20] Matthias Mueller, Neil Smith, and Bernard Ghanem, "A benchmark and simulator for uav tracking," in *European conference on computer vision*. Springer, 2016, pp. 445–461.

[21] Siyi Li and Dit-Yan Yeung, "Visual object tracking for unmanned aerial vehicles: A benchmark and new motion models," in *AAAI*, 2017.

[22] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

[23] Martin Danelljan, Gustav Häger, Fahad Khan, and Michael Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014.