JOINT LIGHTWEIGHT OBJECT TRACKING AND DETECTION FOR UNMANNED VEHICLES

Paraskevi Nousi *Danai Triantafyllidou* *Anastasios Tefas* *Ioannis Pitas*

Department of Informatics, Aristotle University of Thessaloniki, Greece

ABSTRACT

In this paper, we address the problem of lightweight and effective visual object tracking and we present a real-time tracking system suitable for integration in embedded autonomous platforms. We propose a novel tracking framework for classification-based re-detection and tracking, with learnable management of tracking and detection results. The proposed framework includes a novel, very efficient object re-identification method, which filters the detection candidates and systematically corrects the tracking results. In our experiments, we demonstrate the effectiveness of the proposed system by comparing its performance against several other state-of-the art trackers and report the results on the UAV123 and UAV20L datasets. The results indicate that the proposed method is significantly more robust and accurate against recent state-of-the-art trackers, surpassing problems caused by real-world scenarios, while maintaining fast tracking speeds, making it suitable for use in real-time vision applications for autonomous robots, such as Unmanned Aerial Vehicles (UAVs).

Index Terms— Visual Object Tracking, Lightweight Tracking, Re-detection

1. INTRODUCTION

Single object tracking is a crucial task in computer vision pertaining to a vast range of applications, including surveillance, robotic and drone vision. Although much progress has been made in the filed of object tracking, developing robust solutions in terms of heavy occlusions, abrupt motion, deformations and illumination variations remains a challenging problem. In many cases trackers can drift from the original object and never recover it, in which case re-identification is required [1, 2]. Real world scenarios present several challenges towards the re-identification task, mainly due to inter-class variations, i.e., different objects may look alike across different video frames, or intra-class variations, i.e., the same object may look different as a consequence of changes in illumination, scale or pose, indicated by appropriate attribute labels in modern tracking benchmarks [3].

In most typical trackers, a re-identification process would consist of exhaustively evaluating all possible locations in order to find the region that best corresponds to the most recent model of the target. This is extremely inefficient and, as a result, quite ineffective as several input frames may be lost during the exhaustive evaluation leading to more tracker failures. Thus, what is required is a fast way to filter possible locations of the target as soon as a tracking failure is recognized, as well as an efficient way to select the actual target from a list of potential candidates. Recognizing when a tracking failure has or is about to occur is also a very crucial component to a robust tracking system.

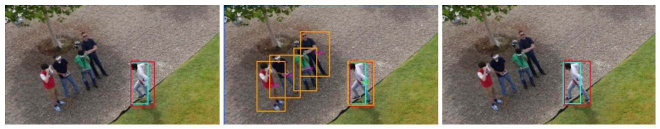


Fig. 1. Example of the proposed joint tracking and tracking framework. During tracker's $\mathcal{T}$ normal function, the reported bounding box (red) is evaluated by the classifier $\mathcal{C}$, and a decision is made on whether the reported bounding box corresponds to the correct target (cyan). Based on the classifier's output, the detector $\mathcal{D}$ predicts possible locations of the target (yellow). The classifier chooses the correct target and tracking continues.

In this paper, we propose a novel tracking framework for classification based target re-detection and tracking (CRT). The proposed framework incorporates three basic modules: a tracker $\mathcal{T}$, a detector $\mathcal{D}$, and a classifier $\mathcal{C}$. The tracker estimates the position of a given target in subsequent video frames. The detector outputs possible locations of the target in case of failure, while the classifier is responsible for re-detecting the target by evaluating the bounding boxes produced by the tracker and detector. Thus, the proposed framework automatically decides if and when to re-initialize the tracking procedure. An example of the proposed framework's function is illustrated in Figure 1.

This paper is structured as follows. In Section 2, we describe the related work in long-term visual object tracking, while in Section 3, we analyze the proposed framework and discuss the role of each component in depth while providing insight into the system's intricate details. The conducted experiments on object tracking are analyzed in Section 4 and finally, conclusions are drawn in Section 5.

2. RELATED WORK

Convolutional Neural Networks (CNNs) have met great success in many modern computer vision problems, such as object recognition and detection [4]. Trackers based on convolutional neural networks, have recently started to attract research attention. The first line of research was focused on combining discriminative correlation filters (DCF) based methods with CNNs by replacing hand crafted features with deep features. CNN-based trackers like MDNet [5], ECO [6], have achieved state-of-the-art performance on various object tracking benchmarks, inspiring a plethora of works.

Recently, a spatially supervised, recurrent CNN coined ROLO [7], was proposed, to regress detections produced by a typical object detector from frame to frame. In [8], a fully convolutional Siamese network (SiamFC) was used to predict the location of a given target in subsequent frames. In [9], the authors proposed a tracker named ADNet, controlled by sequentially pursuing actions learned by Deep Reinforcement Learning. In [10], the GOTURN tracker uses a deep regression network, trained to learn a generic relationship between object motion and appearance, drawing from the ability of deep CNNs to model complex functions from large amounts of data. The last two methods do not use any fine tuning during test time, which fundamentally increases their computational efficiency.

The aforementioned tracking approaches have multiple drawbacks, such as their inability to handle variations arising from long-term tracking scenarios, as well as their computational complexity. Inspired by the considerable speed-up potential of offline training, we show that it is possible to integrate state-of-the-art object detectors that are trained entirely offline to the tracking task and achieve real-time performance, even on embedded systems, while maintaining state-of-the art tracking accuracy.

3. PROPOSED FRAMEWORK

Our proposed system architecture consists of three parts: a detector $\mathcal{D}$, responsible for predicting possible targets, a tracker $\mathcal{T}$, and a classifier $\mathcal{C}$, which coordinates the tracking and detection results and manages the final output.

The detector $\mathcal{D}$ is a submodule capable of predicting the locations of possible targets given an input frame $\mathbf{I}_f \in \mathbb{R}^{H\times W\times C}$, $f = 1,\dots,F$, with spatial dimensions H, W and C channels. It output should be five values for each detected box, representing coordinates, dimensions, as well as the detector's confidence for the predicted box. Formally, the detector is formulated as a function $f_{\mathcal{D}}$ of the input image:

$$f_{\mathcal{D}}(\mathbf{I}_f) = \{\mathbf{d}_i^f\}_{i=1}^{N_f} \tag{1}$$

where N_f is the number of detected objects for the f-th frame, and $\mathbf{d}_i^f = [x_i^f, y_i^f, w_i^f, h_i^f, c_i^f]^T$ is the i-th predicted box for the same frame, represented as a vector containing the box coordinates in top-right format, the box size, and the detector's confidence for this box. The detector may, for example, be implemented as a state-of-the-art two-stage or single-stage CNN-based object detector, such as any variant of Faster R-CNN [4], YOLO [11] or SSD [12], all of which are suitable candidates for the specified detection function.

The tracker $\mathcal{T}$ may be any traditional visual object tracker, which given an input frame $\mathbf{I}_f$ and the previous target location and size $\mathbf{t}^{f-1} = [x_t^{f-1}, y_t^{f-1}, w_t^{f-1}, h_t^{f-1}]^T$, predicts the target's position in the current frame $\mathbf{t}^f$:

$$f_{\mathcal{T}}(\mathbf{I}_f, \mathbf{t}^{f-1}) = \mathbf{t}^f, \tag{2}$$

where $\mathbf{t}^0$ is the groundtruth bounding box of the target to be tracked. In addition, the tracker should provide access to its internal scoring function $g_{\mathcal{T}}$, which evaluates a candidate bounding box and produces a *tracking score* $q \in \mathbb{R}$, roughly corresponding to the similarity between the candidate box and the target. Recent state-of-the-art trackers based on correlation filtering, such as SiamFC [13] or KCF [14], fit ideally into the specified system. Formally, given an input frame $\mathbf{I}_f$ as well as candidate box $\mathbf{d}_i^f$ predicted by the detector, the tracker outputs a tracking score q_i^f:

$$g_{\mathcal{T}}(\mathbf{I}_f, \mathbf{d}_i^f) = q_i^f. \tag{3}$$

Each of the candidate bounding boxes d_i^f for the f-th frame is then evaluated and augmented with the tracker's score:

$$\hat{\mathbf{d}}_i^f = [x_i^f, y_i^f, w_i^f, h_i^f, c_i^f, q_i^f]^T. \tag{4}$$

Finally, the classifier $\mathcal{C}$ must take into consideration features $\mathbf{x}_i^f \in \mathbb{R}^d$ extracted from the scored bounding boxes $\hat{\mathbf{d}}_i^f$ and the previous target position $\mathbf{t}^{f-1}$, and predict which, if any, of the potential bounding boxes correspond to the original target, so as to re-detect it in case of failure. Formally, the classifier should be formulated as a function $f_{\mathcal{C}}(\mathbf{x}_i^f)$ which evaluates the features extracted for a bounding box and outputs a score $p_i^f \in \mathbb{R}$ corresponding to the object's probability of being the actual tracked object:

$$f_{\mathcal{C}}(\mathbf{x}_i^f) = p_i^f, \tag{5}$$

where $\mathbf{x}$ represents the features extracted from a scored candidate box $\hat{\mathbf{d}}$. The tracker's prediction $\mathbf{t}^f$ is also considered as a candidate box for the classifier to verify as corresponding to the actual target, enhanced by the tracker's score $g_{\mathcal{T}}(\mathbf{I}_f, \mathbf{t}^f)$ for this box, forming the evaluation set $\mathbf{D}^f \in \mathbb{R}^{(n_f+1)\times 6}$:

$$\mathbf{D}^f = [\hat{\mathbf{d}}_1^f, \hat{\mathbf{d}}_2^f, \dots, \hat{\mathbf{d}}_{n_f}^f | \hat{\mathbf{t}}^f]^T \tag{6}$$

where $\hat{\mathbf{t}}^f = [x_t^f, y_t^f, w_t^f, h_t^f, 1, q_t^f]$ is the score-augmented 6-dimensional vector representing the tracker's prediction on the position of the target, where a value of 1 is inserted to the fifth position, differentiating this box from the ones extracted by the detector, and q_t^f is the tracker's score on the predicted target itself. The box with the highest probability is the system's final output.

The classifier inherently acts as the system coordinator, by first detecting a failure when all of the candidate boxes are unlikely to depict the tracked target. This functionality can be controlled by a threshold p_{th} for the output of the classifier, such that any box with a predicted probability under this value is considered as a negative sample. As an alternative, a threshold t_{th} may be chosen for the tracker's score on the predicted location $\mathbf{t}^f$, by which to decide when a failure has occurred, i.e., when the score is below that threshold.

The tracking procedure begins with the tracker estimating the new location of the target given the input image and its previous location. Possible tracking failures are detected by the classifier, i.e., by evaluating the box predicted by the tracker. If the classifier outputs a high probability, then no extra effort must be exerted, as the prediction is considered correct. Otherwise, the detector must detect possible targets which are then evaluated by the classifier. If a high probability is predicted for any of those targets, the target with the highest probability of them all is chosen as the system's output and the tracker is re-initialized at this position.

4. EVALUATION

We evaluate the proposed tracking framework on the UAV123 and UAV20L [3] tracking benchmarks and compare its performance against several state-of-the art methods. Following the evaluation method in [15], we measure the performance of the compared trackers in one-pass evaluation (OPE). For this purpose, we used the two standard metrics: (i) *overlap success rate*, defined as the percentage of frames where the bounding box overlap surpasses a given threshold, and (ii) *center location error*, defined as the percentage of frames where the Euclidean distance between the centers of the ground-truth and estimated bounding boxes is under than a given threshold. We use the toolkit provided by [3].

For the tracker $\mathcal{T}$, we choose the KCF tracker [14], for its speed and CPU implementation, leaving the GPU to be exclusively used by the detector. For the detector $\mathcal{D}$, we employ the state-of-the-art YOLOv3 detector [11] trained on the COCO dataset, capable of detecting 80 object classes. The detector is limited to run only in frames where the tracking score produced by $g_{\mathcal{T}}$ for the tracked bounding box is below a threshold t_{th}, set to 0.5.

For the classifier $\mathcal{C}$, we use a simple Multilayer Perceptron (MLP) with one hidden layer of 100 neurons, followed by a ReLU non-linearity and a final binary classification layer. To train the classifier we use the VisDrone2018-SOT train set [16]. We consider a sample $\mathbf{x}_i^f$ to be positive if it has an overlap of at least 0.5 with the annotated, groundtruth box. All other samples are considered as negatives, and we maintain a 1:5 ratio for positive to negative samples, by only keeping the five samples which are the closest to the groundtruth box.

We evaluate the proposed method with all the tracking results reported in [3], including SRDCF [17], MEEM [18], SAMF [19], MUSTER [20], DCF [14], DSST [21], OAB [22], TLD [1], STRUCK [23], MOSEE [24], CSK [25] as well as the baseline tracker KCF [14] and ECO [6]. Furthermore, we evaluate several trackers on this dataset and compare their results to the proposed framework: MDNet [5], SiamFC [13], ADNet [9], CREST [26], BACF[27] and GOTURN[10].

The overall comparison of all trackers on UAV123 using success and precision plots is illustrated in Figure 2a. For each plot, the 10 best performing trackers as well as the baseline KCF are shown with corresponding representative measure i.e., AUC in success plots and precision at 20 pixel in precision plot. In terms of precision, the proposed method (CR+KCF) achieves a score of 71.4%, coming in third place after MDNet (1fps) and SiamFC while being much faster than both [1]. In terms of success, CR+KCF surpasses all real-time trackers, while providing a huge boost ($+20\%$) to the baseline performance of KCF and maintaining its speed.

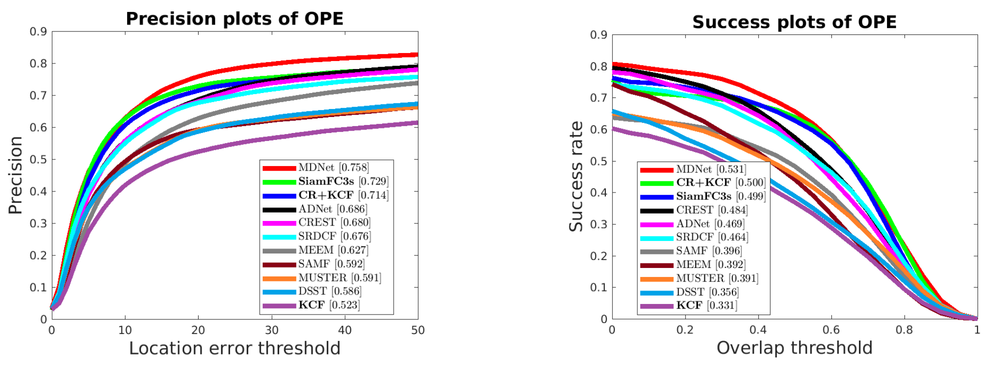
For the UAV20L dataset, we also compare our tracker with the recently released PTAV tracker [2], which is a parallel tracking and verifying system. We compare the results for the class-agnostic case in Figure 2b. Both in precision and in success plots, CR+KCF clearly outperforms all the other trackers, with an overlap measure of 52.4% in the success rate plot and a score of 72% at the precision plot.

4.1. Speed and Deployment on Embedded Devices

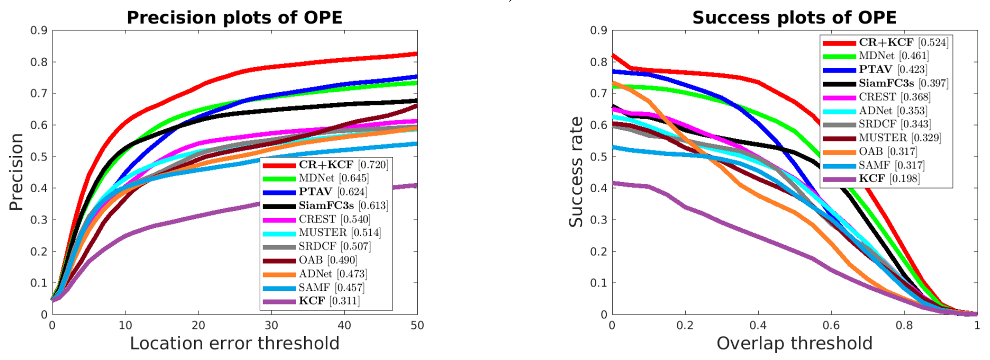
We have implemented our proposed tracking framework in Robot Operation System (ROS) to a) allow for the efficient communication of the framework's submodules and control by a central visual analysis node, and b) to facilitate the system's deployment on robotic devices. More specifically, we have deployed the proposed system on an NVIDIA Jetson TX2 module. To maintain real-time speed even on this low-computational power device, we use the KCF tracker and a YOLOv2 detector, alongside a buffering mechanism to alleviate the effect of the detector's running time on the overall system performance.

Overall, the implemented system is capable of running at about the same speed as the traditional KCF tracker, while achieving much higher overlap scores. At standard resolution images (640×480) our system can run at an average of 24fps on a TX2 module, using the multiscale version of KCF. By using the non-adaptive version, higher speeds can be achieved while the detector can handle the scale variations.

[1]See comparison in https://github.com/foolwood/benchmark_results



a)



b)

Fig. 2. Success (right) and precision plots (left) for OPE (one pass evaluation) on the a) UAV123 and b) UAV20L benchmarks. Real-time trackers are shown in bold.

5. CONCLUSIONS

We have presented a general tracking framework, named CRT, which achieves high tracking performance, comparable to state-of-the-art trackers while maintaining real-time processing capabilities even on embedded systems. Our results in widely accepted tracking benchmarks show that CRT is suitable for long-term tracking scenarios and has the ability to efficiently and effectively handle severe occlusions, viewpoint changes and scale variations. A novel re-initialization module that can be integrated in different tracking algorithms and deals with target loss is introduced. Extensive experiments results demonstrate the ability of the proposed framework to significantly enhance the performance of the basic KCF tracker, achieving state-of-the-art results, while operating in real-time even on embedded devices.

Acknowledgments

This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 731667 (MULTIDRONE). This publication reflects the authors views only. The European Commission is not responsible for any use that may be made of the information it contains.

6. REFERENCES

[1] K. Mikolajczyk Z. Kalal and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012.

[2] Heng Fan and Haibin Ling, "Parallel tracking and verifying: A framework for real-time and high accuracy visual tracking," in *Proc. IEEE Int. Conf. Computer Vision, Venice, Italy*, 2017.

[3] J. Lim Y. Wu and M.-H. Yang, "A benchmark and simulator for uav tracking," in *IEEE European Conference on Computer Vision (ECCV 2016)*, 2016.

[4] Ross Girshick Jian Sun Shaoqing Ren, Kaiming He, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[5] Han B. Nam, H., "Learning multi-domain convolutional neural networks for visual tracking," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[6] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg, "Eco: Efficient convolution operators for tracking," in *CVPR*, 2017.

[7] Guanghan Ning, Zhi Zhang, Chen Huang, Zhihai He, Xi-aobo Ren, and Haohong Wang, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *IEEE International Symposium on Cir- cuits and Systems (ISCAS)*, 2016.

[8] Luca Bertinetto, Jack Valmadre, Stuart Golodetz, Ondrej Miksik, and Philip H. S. Torr, "Staple: Complementary learners for real-time tracking," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[9] Sangdoo Yun, Jongwon Choi, Youngjoon Yoo, Kimin Yun, and Jin Young Choi, "Action-decision networks for visual tracking with deep reinforcement learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[10] Silvio Savarese David Held, Sebastian Thrun, "Learning to track at 100 fps with deep regression networks," in *Conference Computer Vision (ECCV)*, 2016.

[11] R Girshick A Farhadi J Redmon, S Divvala, "You only look once: Unified, real-time object detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[12] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[13] J. F. Henriques A. Vedaldi L. Bertinetto, J. Valmadre and P. H. Torr, "Fully- convolutional siamese networks for object tracking," in *European Conference on Computer Vision (ECCV)*, 2016.

[14] 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.

[15] J. Lim Y. Wu and M.-H. Yang, "Online object tracking: A benchmark," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2013.

[16] Pengfei Zhu, Longyin Wen, Xiao Bian, Haibin Ling, and Qinghua Hu, "Vision meets drones: A challenge," *arXiv preprint arXiv:1804.07437*, 2018.

[17] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4310–4318.

[18] Jianming Zhang, Shugao Ma, and Stan Sclaroff, "MEEM: robust tracking via multiple experts using entropy minimization," in *Proc. of the European Conference on Computer Vision (ECCV)*, 2014.

[19] Jianke Zhu Yang Li, "A scale adaptive kernel correlation filter tracker with feature integration," in *European Conference on Computer Vision, Workshop VOT2014 (ECCVW)*, 2014.

[20] Chaohui Wang Xue Mei Danil Prokhorov Dacheng Tao Zhibin Hong, Zhe Chen, "Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking," in *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[21] P. Martins J. Henriques, R. Caseiro and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *European Conference on Computer Vision (ECCV)*, 2012.

[22] Grabner M. Bischof H. Grabner, H., "Real-time tracking via on-line boosting," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2006.

[23] Philip H. S. Torr Sam Hare, Amir Saffari, "Struck: Structured output tracking with kernels," in *International Conference on Computer Vision (ICCV)*, 2011.

[24] Beveridge J.R. Draper B.A. Lui Y.M. Bolme, D.S., "Visual object tracking using adaptive correlation filters," in *The IEEE Conference on Computer Vision and Pattern Recognition CVPR) Year = 2010*.

[25] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *European conference on computer vision*. Springer, 2012, pp. 702–715.

[26] Yibing Song, Chao Ma, Lijun Gong, Jiawei Zhang, Rynson Lau, and Ming-Hsuan Yang, "Crest: Convolutional residual learning for visual tracking," in *IEEE International Conference on Computer Vision*, 2017, pp. 2555 – 2564.

[27] Simon Lucey Hamed Kiani Galoogahi, Ashton Fagg, "Learning background-aware correlation filters for visual tracking," .