EMBEDDED UAV REAL-TIME VISUAL OBJECT DETECTION AND TRACKING

Paraskevi Nousi, Ioannis Mademlis, Iason Karakostas, Anastasios Tefas, Ioannis Pitas

Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece

ABSTRACT

The use of camera-equipped Unmanned Aerial Vehicles (UAVs, or "drones") for a wide range of aerial video capturing applications, including media production, surveillance, search and rescue operations, etc., has exploded in recent years. Technological progress has led to commercially available UAVs with a degree of cognitive autonomy and perceptual capabilities, such as automated, on-line detection and tracking of target objects upon the captured footage. However, the limited computational hardware, the possibly high camera-to-target distance and the fact that both the UAV/camera and the target(s) are moving, makes it challenging to achieve both high accuracy and stable real-time performance. In this paper, the current state-of-the-art on real-time object detection/tracking is overviewed. Additionally, a relevant, modular implementation suitable for on-drone execution (running on top of the popular Robot Operating System) is presented and empirically evaluated on a number of relevant datasets. The results indicate that a sophisticated, neural network-based detection and tracking system can be deployed at real-time even on embedded devices.

***Index Terms*—** Drone video analysis, object detection, object tracking, real-time computing, Robot Operating System

1. INTRODUCTION

The popularization of commercial, battery-powered, camera-equipped, Vertical Take-off and Landing (VTOL) Unmanned Aerial Vehicles (UAVs, or "drones") during the past decade, has significantly affected aerial video capturing operations, in varying domains such as media production [1–8], search and rescue, surveillance, inspection, etc. UAVs are affordable, agile and flexible, having, for instance, the ability to hover above points interests and access narrow spaces.

Traditional UAVs are teleoperated fully manually, but technological progress has led to the commercial release of drones with a significant degree of cognitive autonomy, relying on advanced sensors, embedded computing boards and artificial intelligence/robotics algorithms. An illuminating example would be current high-end cinematography UAVs, such as the DJI Phantom IV Pro, or the more recent Skydio R1, which already provide a number of autonomous capabilities for both safe flying and filming, such as obstacle detection and avoidance, automated landing, physical target following/target orbiting enabled by visual target tracking (for low-speed, manually pre-selected targets), as well as automatic central composition framing, i.e., continuously rotating the camera so as to always keep the pre-selected target properly framed at the center.

Clearly, near-future holds the promise of fully autonomous UAVs that only require high-level supervision from a human operator. Automated video analysis [9–14], such as on-line video detection and tracking of targets, lies at the heart of these developments. However, the limited computational hardware, the possibly high camera-to-target distance and the fact that both the UAV/camera and the target(s) are moving, makes it difficult to achieve both accuracy and stable real-time performance in these tasks. Relevant state-of-the-art algorithms, e.g., based on deep neural networks, are impressively precise and optimized for parallel execution on General-Purpose Graphical Processing Units (GP-GPUs). Such high-performance hardware has recently been commercialized in small, power-efficient form factor for embedded systems, ideal for on-board inclusion in UAVs[1]. However, current processing power and energy consumption restrictions limit what is possible on a UAV, in comparison to desktop computers.

2D visual target detection is necessary for localizing the desired target's image (i.e., the Region-of-Interest, or ROI) on a video frame. Additionally, visual target detectors can be exploited for identifying a possible obstacle or an on-ground UAV landing site. The extracted ROI is a rectangle (described in pixel coordinates) that encloses the target's image. In current commercially available drones, similar methods are already employed to better adjust a manually pre-specified ROI, based on the video content. In the future, more automated UAVs are expected to rely solely on automatic visual target detection.

2D visual target tracking tracks a pre-specified ROI on the consecutive frames of a video sequence, by taking advantage of spatiotemporal locality constraints, and updates the ROI pixel coordinates at each video frame. Although track-

The research leading to these results has received funding from the European Union's European Union Horizon 2020 research and innovation programme under grant agreement No 731667 (MULTIDRONE).

[1] E.g., the NVIDIA Jetson series

ing can be performed by simply re-detecting the target at each video frame, a better approach is to periodically re-initialize the ROI using a 2D visual target detector and employ a separate visual tracker for the intermediate intervals. In general, correlation-based trackers are suitable for real-time operation in embedded computing environments [15].

In this paper, the current state-of-the-art on real-time object detection/tracking is overviewed. Additionally, a relevant, modular implementation suitable for on-drone execution (running on top of the popular Robot Operating System) is presented and empirically evaluated on a number of datasets.

2. REAL-TIME OBJECT DETECTION & TRACKING

In the following subsections we provide an overview of recently proposed solutions to the problems of visual object detection and tracking, focusing on the performance of these solutions and their applicability on embedded systems such as the ones studied in this work.

2.1. Object Detection

Deep Convolutional Neural Networks (CNNs) have been excelling continuously on various challenging visual analysis tasks and competitions, including the ILSVRC object recognition and detection challenges [16], and the PASCAL VOC challenges [17]. Deep models with parameter-heavy architectures have been successfully trained and deployed on such tasks, partly due to the availability of large collections of annotated datasets, such as the ImageNet or COCO datasets [18], and partly due to the continuous development of increasingly more powerful GPUs. However, the power consumption and sheer size of such models inhibit their use on mobile and embedded systems, as the GPUs available for deployment on such systems are inadequate in terms of computational power, thus severely slowing down the performance of large models and rendering real-time deployment almost impossible. Moreover, memory constraints prohibit the direct deployment of large models even when real-time requirements can be relaxed. On the other hand, applications related to visual analysis tasks have become progressively more popular, increasing the demand of deployment of large CNNs on mobile devices.

One approach to achieving real-time performance with restricted computational hardware is to use one-stage deep neural detectors, structured around the concept of "anchors". These detectors, such as Single-Shot Detector (SSD) [19] and You Only Look Once (YOLO) [20], are based on the notion of a convolutional Region Proposal Network (RPN). They simultaneously regress the pixel coordinates of visible object ROIs (in the form of spatial offsets from the pre-defined anchors) and assign them class labels. Although they are suitable for operating on-board a UAV, they typically lag in accuracy relative to slower two-stage, region-based detectors, such as Faster R-CNN [21]. Advances such as the Focal Loss [22] have been proposed to mitigate this issue, with limited success.

This has lead recent research towards the optimization of heavyweight CNN architectures for deployment on devices with limited resources. In [23], several object classification and detection algorithms are studied and their performance on various mobile devices, including a Jetson TX2 platform, is compared in terms of speed. The latency versus throughput trade-off is evaluated for different batch sizes and it is shown that using batch sizes larger than 1 is more computationally efficient. In real-time applications, however, images must be processed one-by-one sequentially as they are captured.

In [24], MobileNets are pitted against other popular feature extractors, including the Inception V2 model [25], in the context of feature extraction for object detection, and the effect of altering the input size on the detection precision is examined, among other factors. Larger input sizes lead to larger heatmaps and denser object detection, but impose heavy memory and computational constraints. In contrast, smaller input sizes are processed faster but lead to coarser, less accurate predictions.

2.2. Object Tracking

Most of the 2D visual object tracking algorithms employ the tracking-by-detection approach [26–28], where a discriminative model is trained by employing a ROI representation of the first video frame and then used to detect the target ROI in the next ones. It is typically updated within successive video frames. Correlation filter-based tracking algorithms tend to vary with regard to: a) the optimization procedure followed in order to train the model, and b) the target template representation (e.g., HOG feature descriptors [29], grayscale raw pixel values, etc.).

Correlation filter-based 2D visual target tracking algorithms are suitable for real-time applications, especially on embedded systems that tend to have limited computational resources. A correlation filter tracker regresses the representations of all possible object template translations to a Gaussian distribution. The original ROI object template is regressed to its peak. Due to the circulant structure of the template representations, the regression problem can be solved in the Fourier domain, thus accelerating the learning and testing processes of the tracker. Correlation filter-based approaches seem to be more robust to target rotations than methods based on classification [30].

The success of CNNs in various visual analysis tasks, has led to their adoption for visual tracking. SiamFC [31] is one such CNN-based tracker, trained as a fully convolutional siamese network, which performs cross correlation between the features extracted from the target and a candidate region to find the new position of the target. In contrast to GOTURN, data augmentation is unnecessary, due to the network's fully

convolutional nature. Its successor, CFNet [32], included a Discriminative Correlation Filter (DCF) module, presented as a fully learnable layer.

More recently, anchor-based ROI selection was incorporated into a siamese architecture for tracking, coined SiamRPN [33]. The main benefit of using anchors to match the bounding box of the target is that the tracker can handle aspect ratio changes, when traditional trackers typically only deal with size changes while maintaining a constant aspect ratio.

3. AN EMBEDDED UAV VIDEO ANALYSIS SYSTEM

Based on the overview in Section 2, a modular software system for real-time, embedded on-drone video analysis has been implemented using the popular Robot Operating System (ROS) middleware [34]. The latter provides the abstractions of *topics* (following a publisher/subscriber model) and *services*, to permit easy inter-process communication across devices. Standard ROS message libraries, as well as a set of custom messages and services have been employed for inter-module interactions.

The presented *2D Visual Information Analysis* system consists of a visual object detector and a visual object tracker. A Master Visual Analysis node (MVA) serves as a service client for the initialization of the detection and tracking tasks. It receives an uncompressed video frame from the UAV's camera in real-time and generates 2D positions of the tracked targets as regions-of-interest/bounding boxes in pixel coordinates. The system is initialized by a call to the *follow_target* service, which informs 2D Visual Information Analysis about the current target type and target ID.

The MVA subsequently calls the *detect* service, which the detector provides, and receives possible ROIs (e.g., bounding boxes of persons of interest). These, or a subset of them, are forwarded to the tracker via the the *track* service, which initializes the tracker. After initialization, the tracker produces ROIs for all tracked objects, given just the video input from the video streamer node. The tracker may be re-initialized by calling the *follow_target* service. The proposed system is illustrated in Figure 1. Note that the proposed system is very modular and can work with any detector and tracker combination, given that the detector can produce bounding boxes of possible candidates from an input image, and that the tracker can then be initialized with those candidates and start tracking the objects in subsequent frames.

3.1. Object Detection

For the purpose of object detection, we focus our study on single-stage detectors, namely SSD and YOLO. Although region-based detectors, such as Faster R-CNN, are more accurate, they tend to be slower than single-stage detectors as demonstrated in [24], motivating our choice of evaluated detectors.

SSD SSD [19] is a single stage multi-object detector, meaning that a single feed forward pass of an image suffices for the extraction of multiple bounding boxes with coordinate and class information and no region proposal occurs internally. In [24], SSD was used as a meta-architecture for single stage object detection and compared against region-based detectors. Among the findings of that work, was that SSD with MobileNets and Inception V2 for the feature extraction step provided the best time performance at the cost of lower detection precision, as evaluated on the challenging COCO dataset.

YOLO Although similar in nature to SSD, YOLO [20] is a widely used object detector, whose popularity may be attributed to its simplicity, stemming from its ability to detect multiple objects with a single forward pass of an image, in combination with its speed which surpasses that of SSD. A smaller version, named Tiny YOLO, is also available and performs object detection based on the same principles. Using half the convolutional layers, Tiny YOLO sacrifices precision for the sake of speed. The tiny version is also fully convolutional and subsamples the input image by a factor of 32.

3.2. Object Tracking

Multithreaded KCF For the 2D visual target tracking task of our system we developed a multithreaded version of the KCF algorithm[2] in order to achieve faster target tracking speeds. To this end, for every successive frame, the tracking process spawns three threads running in parallel, each one dedicated to a different scale factor of the ROI.

SiamFClite We furthermore develop a more lightweight version of SiamFC, by introducing a depth factor α, similar to the one used by MobileNets [24] to improve their speed. More specifically, the number of filters in each layer of the siamese architecture is multiplied by $\alpha \in (0, 1]$, thus producing a lighter network which requires fewer operations per layer. Let n_l denote the number of filters for layer $l = 1, \ldots, 5$ for the five layers of AlexNet, the base feature extractor of SiamFC. The developed SiamFClite tracker is parameterized by $\alpha \sum_{l=1}^{5} n_l$ filters, as opposed to the $\sum_{l=1}^{5} n_l$ of the original SiamFC.

4. EMPIRICAL EVALUATION

4.1. Real-time Object Detection

We evaluate the detectors on another single-class problem, that of detecting bicycles in a cycling race. For this purpose,

[2]Original KCF code in C++: https://github.com/joaofaro/KCFcpp

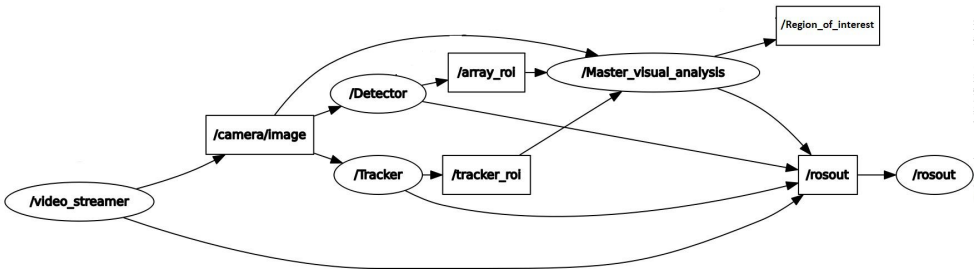

Fig. 1. The ROS computational graph of the developed detection and tracking framework.

we have gathered a dataset consisting of about 12k images from cycling events and annotated about 77k cyclists along with their bicycles. As most of the shots are aerial, the annotated objects are small relative to the image size. The dataset also contains many partially occluded objects as well as objects with motion blur. Finally, the bicycles to be detected are professional and easy to confuse with other vehicles, such as motorcycles, especially in distant shots.

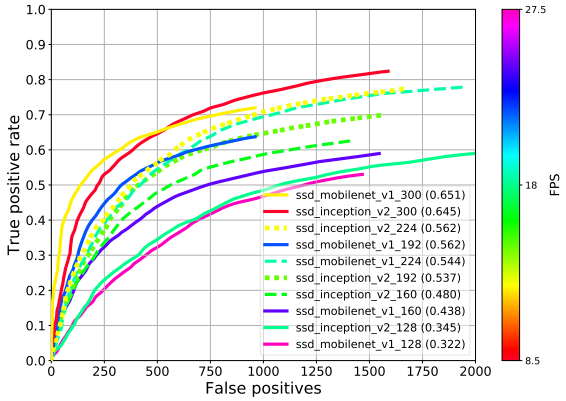

Fig. 2. True positive rate vs false positives for the SSD detectors with MobileNet and Inception base models on the Bicycle Detection benchmark.

Figure 2 illustrates the performance of the SSD with MobileNet v1 and Inception v2 backbones detectors. As expected, at a fixed 500 false positive detections, the performance rises dramatically as the resolution increases — from 32.2 to 65.1 for MobileNet and from 34.5 to 64.5 for Inception. By allowing for more false positives, the Inception models achieve higher recall rates. It is also noteworthy that the number of false positives is much larger than in the face detection scenario, testifying to the difficulty of this task. At around 22 FPS and 56.2 recall rate (at 500 false positives), we identify the MobileNet v1 model at 192×192 input resolution to offer a great trade-off between speed and accuracy. The results are also summarized in Table 1 for both backbones and all input sizes.

For the Tiny YOLO detector, the recall curves are illustrated in Figure 3. It is obvious that Tiny YOLO is very prone

Input Size	Extractor	FPS	Recall
300×300	Inception v2	8.5	64.5
	MobileNet v1	12.4	65.1
224×224	Inception v2	12.7	56.2
	MobileNet v1	18.4	54.4
192×192	Inception v2	14.7	53.7
	MobileNet v1	22.0	56.2
160×160	Inception v2	16.4	48.0
	MobileNet v1	24.4	43.8
128×128	Inception v2	18.0	34.5
	MobileNet v1	27.5	32.2

Table 1. Frames per second and recall scores for various input sizes for the SSD with Inception v2 and MobileNet v1 feature extractors on the Bicycles Benchmark.

to false positives, which depicts the performance of various Tiny YOLO configurations. At the smallest input resolution of 224×224, the detector runs at 39 FPS but achieves a disappointing recall rate of 27.3, when the SSD detectors at the same input size achieve over 30 and fewer false positives in general. However, at 288×288 input size, Tiny YOLO achieves a recall rate of 35.7 and 23 FPS, much faster than any of the SSD detectors and almost real time.

4.2. Real-time Object Tracking

We evaluate the performance of four trackers, which have been incorporated into the proposed 2D Visual Information Analysis package: baseline KCF, our multithreaded KCF and SiamFClite trackers, as well as the recently proposed SiamRPN. Their performance comparison is summarized in Table 2, in terms of speed (FPS) and overlap with the groundtruth target, as measured on the OTB100 dataset [35]. More specifically, we report the Area Under the Curve score (AUC), for the overlap success percentage versus overlap threshold curves, averaged over the 100 sequences of OTB100. The multithreaded KCF manages to outperform the standard version by more than 20 fps. SiamFClite also manages to achieve an impressive AUC score while running at 30fps, whereas SiamRPN in general is not real-time, but achieves the best tracking performance by a large margin.

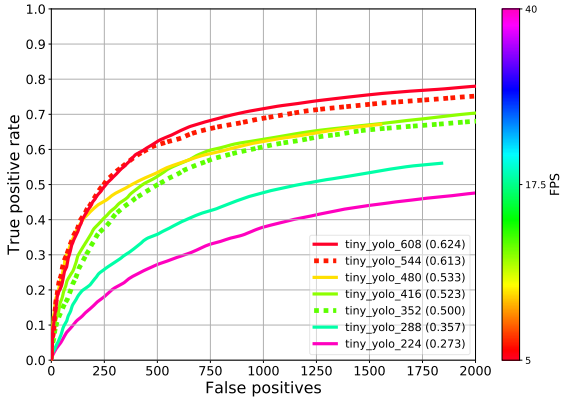

Fig. 3. True positive rate vs false positives for the Tiny YOLO detectors on the bicycle benchmark.

Tracker	FPS	AUC
KCF	30	47.7
KCF (multithreaded)	51	47.7
SiamFClite	30	54.4
SiamRPN	15	63.7

Table 2. Comparison of various trackers. All speed measurements made on a Jetson TX2 platform.

5. CONCLUSIONS

An overview of the state-of-the-art in real-time object detection/tracking from video footage has been presented, from the perspective of embedded UAV video analysis. The combination of one-stage deep neural detectors and correlation-based trackers seems to provide the best balance between accuracy and real-time performance, under the energy and computational constraints imposed by the UAV setting. A specific modular software system incorporating a range of detectors and trackers was implemented in a ROS environment and a evaluated on a number of relevant datasets. The results indicate that a sophisticated, neural network-based detection and tracking system can be deployed at real-time even on embedded devices.

REFERENCES

[1] I. Mademlis, V. Mygdalis, C. Raptopoulou, N. Nikolaidis, N. Heise, T. Koch, J. Grunfeld, T. Wagner, A. Messina, F. Negro, et al., "Overview of drone cinematography for sports filming," *European Conference on Visual Media Production (CVMP), short*, 2017.

[2] A. Torres-González, J. Capitán, R. Cunha, A. Ollero, and I. Mademlis, "A multidrone approach for autonomous cinematography planning," *Iberian Robotics Conference (ROBOT)*, 2017.

[3] I. Mademlis, V. Mygdalis, N. Nikolaidis, and I. Pitas, "Challenges in Autonomous UAV Cinematography: An Overview," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2018.

[4] I. Mademlis, N. Nikolaidis, A. Tefas, I. Pitas, T. Wagner, and A. Messina, "Autonomous unmanned aerial vehicles filming in dynamic unstructured outdoor environments," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 147–153, 2018.

[5] I. Karakostas, I. Mademlis, N. Nikolaidis, and I. Pitas, "UAV cinematography constraints imposed by visual target tracking," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2018.

[6] A. Messina, S. Metta, M. Montagnuolo, F. Negro, V. Mygdalis, I. Pitas, J. Capitán, A. Torres, S. Boyle, and D. Bull, "The future of media production through multi-drones' eyes," in *International Broadcasting Convention (IBC)*, 2018.

[7] I. Karakostas, I. Mademlis, N. Nikolaidis, and I. Pitas, "Shot type feasibility in autonomous UAV cinematography," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[8] I. Mademlis, V. Mygdalis, N. Nikolaidis, M. Montagnuolo, F. Negro, A. Messina, and I. Pitas, "High-level multiple-UAV cinematography tools for covering outdoor events," *IEEE Transactions on Broadcasting*, 2019.

[9] D. Triantafyllidou, P. Nousi, and A. Tefas, "Fast deep convolutional face detection in the wild exploiting hard sample mining," *Big Data Research*, vol. 11, pp. 65 – 76, 2018.

[10] P. Nousi, E. Patsiouras, A. Tefas, and I. Pitas, "Convolutional neural networks for visual information analysis with limited computing resources," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2018.

[11] P. Nousi and A. Tefas, "Discriminatively trained autoencoders for fast and accurate face recognition," in *Proceedings of the International Conference on Engineering Applications of Neural Networks (ICANN)*. Springer, 2017.

[12] N. Passalis and A. Tefas, "Concept detection and face pose estimation using lightweight convolutional neural networks for steering drone video shooting," in *Proceedings of the European Signal Processing Conference (EUSIPCO)*. IEEE, 2017.

[13] N. Passalis and A. Tefas, "Improving face pose estimation using long-term temporal averaging for stochastic optimization," in *Proceedings of the International Conference on Engineering Applications of Neural Networks (ICANN)*. Springer, 2017.

[14] V. Mygdalis, A. Iosifidis, A. Tefas, and I. Pitas, "Semi-

supervised subclass support vector data description for image and video classification," *Neurocomputing*, vol. 278, pp. 51–61, 2018.

[15] O. Zachariadis, V. Mygdalis, I. Mademlis, N. Nikolaidis, and I. Pitas, "2D visual tracking for sports UAV cinematography applications," *Proceedings of the IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2017.

[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[17] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," .

[18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C.L. Zitnick, "Microsoft COCO: Common objects in context," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 740–755.

[19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single-shot multibox detector," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 21–37.

[20] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," *arXiv preprint*, vol. 1612, 2016.

[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 2015, pp. 91–99.

[22] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[23] J. Hanhirova, T. Kämäräinen, S. Seppälä, M. Siekkinen, V. Hirvisalo, and A. Ylä-Jääski, "Latency and throughput characterization of convolutional neural networks for mobile computer vision," *arXiv preprint arXiv:1803.09492*, 2018.

[24] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, and S. Guadarrama, "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[25] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015, pp. 448–456.

[26] M. Danelljan, G. Hager, F.S. Khan, and M. Felsberg, "Discriminative scale space tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[27] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P.H.S. Torr, "Staple: Complementary learners for real-time tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1401–1409.

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

[29] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*, 2005.

[30] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S.L. Hicks, and P.H.S. Torr, "Struck: Structured output tracking with kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 10, pp. 2096–2109, 2016.

[31] L. Bertinetto, J. Valmadre, J.F. Henriques, A. Vedaldi, and P.H.S. Torr, "Fully-convolutional siamese networks for object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 850–865.

[32] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P.H.S. Torr, "End-to-end representation learning for correlation filter-based tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2805–2813.

[33] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High-performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8971–8980.

[34] M. Quigley, K. Conley, Gerkey. B.P., Faust. J., Foote. T., J. Leibs, R. Wheeler, and A.Y. Ng, "ROS: an open-source Robot Operating System," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) Workshop on Open Source Software*, 2009.

[35] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 2411–2418.