Efficiency and Precision Trade-offs in UAV Tracking with Filter Pruning and Dynamic Channel Weighting

Pengzhi ZHONG* [a], Dan ZENG* [b], Xucheng WANG [a] and Shuiwang LI [a,1]
[a] *Guangxi Key Laboratory of Embedded Technology and Intelligent System, Guilin University of Technology, China*
[b] *Research Institue of Trustworthy Autonomous Systems, Southern University of Science and Technology, China*

Abstract. Due to the limitations of computing resources, battery capacity, and maximum load of unmanned aerial vehicle (UAV), efficiency is a critical issue in UAV tracking. Deep learning (DL)-based trackers are known for their high precision but hardly achieve real-time tracking on a single CPU. The traditional framework of discriminative correlation filters (DCF) is famous for its high efficiency but its precision is barely satisfactory. Despite the inferior precision, DCF-based trackers rather than DL-based ones are widely adopted in UAV tracking to trade precision for efficiency. This paper aims to trade off efficiency and precision for UAV tracking using model compression techniques (i.e., filter pruning), which has not been well explored before. To combat the possible precision drop caused by pruning, we propose a dynamic channel weighting strategy to improve the rank-based pruning method which uses the average rank of each filter response as the pruning criterion. The dynamic channel weighting seeks to adjust the contribution to the loss of each channel through optimization. As a result, we achieved better precisions when compressing the original model SiamFC++ with a global pruning ratio instead of tedious layer-wise ones. Extensive experiments on four UAV benchmarks, i.e., UAV123@10fps, DTB70, UAVDT, and Vistrone2018, show that the proposed tracker demonstrates a remarkable balance between precision and efficiency.

Keywords. UAV tracking, filter pruning, real time, dynamic channel weighting

1. Introduction

UAV tracking is of great importance in various military and non-military applications, such as counter-terrorism, border security, product deliveries, agriculture, etc. [1,2]. However, compared to visual tracking in general scenes, UAV tacking faces more formidable challenges. As efficiency is currently a fundamental issue in UAV tracking, discriminative correlation filters (DCF)-based trackers instead of deep learning (DL)-

[1]Corresponding Author: Shuiwang Li, Guilin University of Technology, No. 319, Yanshan Street, Yanshan District, Guilin City, China; lishuiwang0721@163.com. * These authors contributed equally.
E-mail address: zhongpengzhi@glut.edu.cn (Pengzhi Zhong), zengd@sustech.edu.cn (Dan Zeng), xcwang@glut.edu.cn (Xucheng Wang).

based ones are frequently adopted to trade precision for efficiency. Unfortunately, although the tracking precisions of DCF-based trackers has been improved, they still cannot compare with most state-of-the-art DL-based trackers, whereas, state-of-the-art DL-based trackers can hardly run at real-time speed on a single CPU. Very recently, an efficient deep tracker for UAV tracking was proposed in [2], which uses a lightweight backbone to achieve a remarkable tracking precision. However, this tracker is not yet real-time on a single CPU. Importantly, it implies that an lightweight DL-based tracker could be a good substitute for DCF-based trackers to balance efficiency and precision, which inspires us to exploit model compression to trade precision for efficiency in seeking lightweight DL-based trackers. Model compression technique is typically used to deploy deep networks in resource-constrained devices without greatly compromising model accuracy [3]. Prevalent methods include pruning, quantization, low-rank approximation, and knowledge distillation [4]. However, the selection of DL tracker and compression method is essential to achieve real-time yet high precision tracking performance. SiamFC++ [5] 's precision and speed make it a good choice as a DL-based tracker to be compressed. The rank-based filter pruning method [6] is very straightforward and has high training efficiency, which is chosen as our model compression method. But the determination of its layer-wise pruning ratios is arduous and time-consuming. Fortunately this can be avoided by using a global pruning ratio. In order to use a global pruning ratio this paper proposes a dynamic channel weighting strategy to diminish the loss of precision probably caused by it. We call the proposed method PW-SiamFC++ since our tracker is based on filter pruning and dynamic weighting.

Our contributions can be summarized as follows: (1) We provide a novel approach to trade-off efficiency and precision for UAV tracking through filter pruning. Filter pruning is key to enhance the efficiency in our tracker and has not been well explored before. (2) To combat the accuracy drop caused by filter pruning with a global pruning ratio, we propose a dynamic channel weighting to adjust the contribution of each channel in every convolutional layer to the total loss, with which the proposed tracker, i.e., PW-SiamFC++, is able to prune the SiamFC++ with a rank-based criterion to about 60% of its original model size, meanwhile remarkably achieving an overall better precision. (3) We conduct experiments on four UAV tracking benchmarks, namely UAV123@10fps, DTB70, UAVDT, and VisDrone2018 which demonstrate that the PW-SiamFC++ tracker achieves SOTA performance in UAV tracking.

2. Related Works

2.1. Visual Tracking Methods

MOSSE [7] is the first DCF tracker that utilizes the minimum output sum of squared error filter for tracking and brings other variants [8]. DCF-based trackers can be calculated in the Fourier domain, which leading to competitive performance with high efficiency, which, therefore, stood out in the UAV tracking community. However, due to the limited representation ability of handcrafted features, it is difficult for DCF-based trackers to maintain robustness under challenging scenarios. As to DL-based trackers, SiamFC [9] regards visual tracking as a general similarity learning problem, and uses Siamese network to measure the similarity between the target and the search image. Since then, a

number of DL-based trackers based on the Siamese architecture have been proposed. Recently, deeper architectures have been developed to further improve tracking precision, including SiamRPN++ [10], SiamBAN [11] and etc. However, their tracking efficiency is largely sacrificed. In contrast, SiamFC++ [5] has a lightweight backbone and an effective quality assessment branch for classification, making it a simple but effective framework. Unfortunately, its CPU speed seems farfetched to meet strict real-time requirements. This paper intends to improve the efficiency of SiamFC++ while maintaining its precision as much as possible with filter pruning.

2.2. Filter Pruning

Filter pruning is generally divided into weight pruning and filter pruning [4]. Weight pruning usually removes neurons or weights, while filter pruning removes the entire filters or channels. A pruning pipeline is usually comprised of three steps: pretraining, pruning and finetuning, and it normally involves four classic topics, i.e., pruning structure, pruning ratio, pruning criterion, and pruning schedule [4]. The pruning ratio indicates how many weights to remove, and there are usually two ways to settle it. The first is a predefined full layer ratio or multiple layer-wise ratios. The second is to indirectly adjust the pruning ratio, such as using a regularization-based pruning method. The pruning criterion is used to determine which weights to prune. Last but not least, pruning schedule is to specify how the network changes from start state to the target digital, one-shot or progressively. The former is more efficient without involving complex training strategies. Recently, an effective filter pruning method was proposed in [6]. However, the determination of layer-wise pruning ratios in this method is laborious and time-consuming. To overcome this problem, we propose to use a global pruning ratio. Furthermore, to combat the precision drop, we propose a dynamic channel weighting strategy to dynamically adjust the weights. Hopefully, with adjusted weights, the contribution to the loss of each channel mimics or surpasses the optimal state it could reach when the model is pruned in a layer-wise manner.

3. Proposed Method

3.1. PW-SiamFC++ Overview

The proposed PW-SiamFC++ consists of a backbone, a neck, and a head network. The overview of PW-SiamFC++ is shown in Fig. 1. The input consists of two branches, namely the template branch and the search branch, taking the target patch Z and the search patch X as input. The two branches share the same backbone network, denoted by $\phi(\cdot)$. The output features of them interact with the cross-correlation operation $\star$ as follows,

$$f_l(X,Z) = \psi_l(\phi(X)) \star \psi_l(\phi(Z)), \psi_l \in \{\psi_{cls}, \psi_{reg}\}, \tag{1}$$

where $\psi_{cls}(\cdot)$ and $\psi_{reg}(\cdot)$ denote the task-specific layer for regression and classification, respectively. The classification branch is used to predict the category for each location, while the regression branch is to compute the target bounding box at this location, where w and h are the width and height of the outputs, respectively. A center-ness branch is used to assess classification qualities, which is finally used to reweight the classification

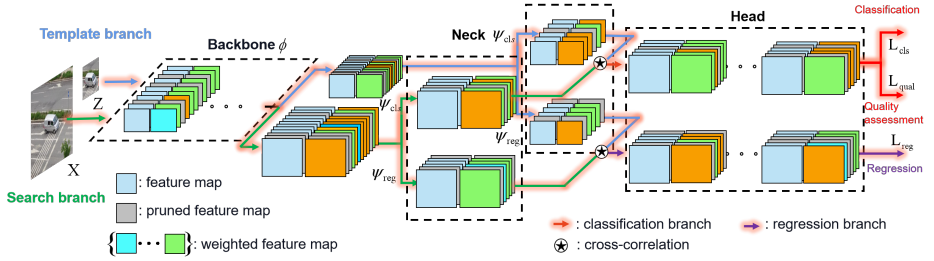

Figure 1. The proposed PW-SiamFC++ method. It inherits the network structure of SiamFC++. Note that ψ_{cls} and ψ_{reg} denote the task-specific convolutional layers for classification and regression, respectively.

scores. PW-SiamFC++ has the same pipeline as SiamFC++, with the differences being in the pruned and the weighted feature maps, which will be explained in detail below.

3.2. Filter Pruning and Dynamic Channel Weighting

We utilize the rank-based filter pruning method with the average rank of each filter response as the pruning criterion to prune the filters considered less important. We use a group of 3-D filters $W_{C^i} = \{w_1^i, w_2^i, ..., w_m^i\} \in \mathbb{R}^{n_i \times n_{i-1} \times k_i \times k_i}$ to denote the i-th ($1 \leqslant i \leqslant K$) convolutional layer C^i of SiamFC++, where k_i denotes the kernel size, n_i indicates the number of filters in C^i, and the j-th filter in C^i is parameterized by $w_j^i \in \mathbb{R}^{n_{i-1} \times k_i \times k_i}$. The output feature maps of C^i are denoted by $O_{C^i} = \{o_1^i, o_2^i, ..., o_m^i\} \in \mathbb{R}^{n_i \times g \times h_i \times w_i}$, where $o_j^i \in \mathbb{R}^{g \times h_i \times w_i}$ is associated with w_j^i, w_i and h_i, respectively, denote the width and height of the feature maps, g is the number of input images. The rank-based filter pruning solves:

$$\min_{\delta_{i,j}} \sum_{i=1}^{K} \sum_{j=1}^{n_i} \delta_{i,j} \mathbb{E}_{I \sim P(I)}[\Re(o_j^i(I))], \quad s.t \sum_{j=1}^{n_i} \delta_{i,j} = n_p^i, \tag{2}$$

where $I \sim P(I)$ represents an input image, n_p^i represents the number of filters pruned in C^i. $\delta_{i,j} \in \{0,1\}$ indicates whether or not pruning w_j^i, it is if $\delta_{i,j} = 1$, otherwise $\delta_{i,j} = 0$. $\Re(\cdot)$ is a measure of information richness, computing a feature map' s rank. It is empirically proved in [6] that expectation ranks generated by a single filter is robust to the inputs [6], for which Eq. (2) is approximated by

$$\min_{\delta_{i,j}} \sum_{i=1}^{K} \sum_{j=1}^{n_i} \delta_{i,j} \sum_{t=1}^{g} \Re(o_j^i(I_t)), \quad s.t \sum_{j=1}^{n_i} \delta_{i,j} = n_p^i, \tag{3}$$

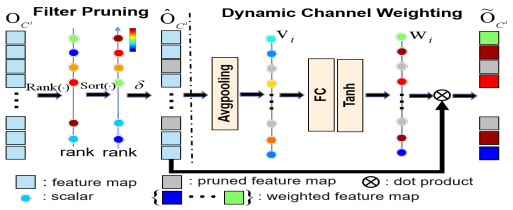

Figure 2. An illustration of filter pruning and dynamic channel weighting proposed in our tracker. Filter pruning intends to prune the filters according to the rank of the feature map. Dynamic channel weighting learns new weights of the remaining channels to counteract the adverse effects of pruning.

where t is the input images for indexed. Eq. (3) using the feature maps of least average ranks is easily to minimize pruning n_p^i filters The original rank-based filter pruning approach uses layer-wise pruning ratios to achieve more freedom in pruned structures and thus better performance. But the determination of layer-wise pruning ratios is arduous and time-consuming. To avoid this we use a global pruning ratio instead here. And we proposes a dynamic channel weighting strategy to diminish the loss of precision probably due to the global pruning ratio. After the filter pruning, the remaining channels will be dynamically reweighted according to filter responses. Considering that the relative importance of each channel has been changed, hopefully, with the proposed dynamic channel weighting, the adverse impacts of these changes will be canceled and even be led to the beneficial side. We illustrate filter pruning and dynamic channel weighting of a convolutional layer in Fig. 2. The dynamic channel weighting process is as follows. First, the remaining $\hat{n}_i$ (i.e., $n_i - n_p^i$) output feature maps of O_{C^i} in the i-th convolutional layer, i.e., $\hat{O}_{C^i}$, are mapped to a vector $\mathbf{v}_i \in \mathbb{R}^{\hat{n}_i}$ by a global average pooling. Then $\mathbf{v}_i$ is followed by a fully connected layer and the $tanh(\cdot)$ activation function successively to produce a weighting vector $\mathbf{w}_i$. Finally, the weighting vector $\mathbf{w}_i$ is used to reweight the original output feature maps $\hat{O}_{C^i}$ by dot product to get $\tilde{O}_{C^i}$, the reweighted one. The pruned SiamFC++ with the dynamic channel weighting layers incorporated makes the network architecture of our PW-SiamFC++. The losses for finetuning the PW-SiamFC++ is just the same as that for training SiamFC++ [5].

3.3. Pruning and Weighting Schedule

The following is the process of pruning and weighting. Firstly, calculate the average ranks associated with every filter to get the rank sets $\{R^i\}_{i=1}^K = \{\{r_1^i, r_2^i, ..., r_{n_i}^i\}\}_{i=1}^K$, where R^i is the rank set of the i-th convolutional layer. Second, each rank set R^i to decreasing order for sorted, and end up with $\bar{R}^i = \{r_{s_1^i}^i, r_{s_2^i}^i, ..., r_{s_{n_i}^i}^i\}$, where s_j^i is the index of the j-th top value in R^i. Third, conduct filter pruning according to a predefined global pruning ratio ρ, with which R^i turns to $\hat{R}^i = \{r_{s_1^i}^i, r_{s_2^i}^i, ..., r_{s_{\hat{n}_i}^i}^i\}$, and incorporate the dynamic channel weighting layers to obtain PW-SiamFC++. Finally, initialize the retained filters with the original weights in SiamFC++ and all fully connected layers for dynamic channel weighting with Kaiming initialization [12], and then finetune the PW-SiamFC++.

4. Experiments

Our experiments are conducted on four challenging UAV benchmarks, i.e., UAV123@10fps [13], DTB70 [14], UAVDT [15] and Vistrone2018 [16], with a PC equipped with i9-10850K processor (3.6GHz), an NVIDIA TitanX GPU, and 16GB RAM. And the global pruning ratio of our PW-SiamFC++ is set to 0.4. Others are the same as SiamFC++ [5].

4.1. Comparison with DCF-based Trackers

Ten DCF-based trackers are used for comparison, including KCF [17], fDSST [18], BACF [19], Staple-CA [20], ECO-HC [21], STRCF [22], MCCT-H [23], ARCF-HC [24], AutoTrack [1], and RACF [8].

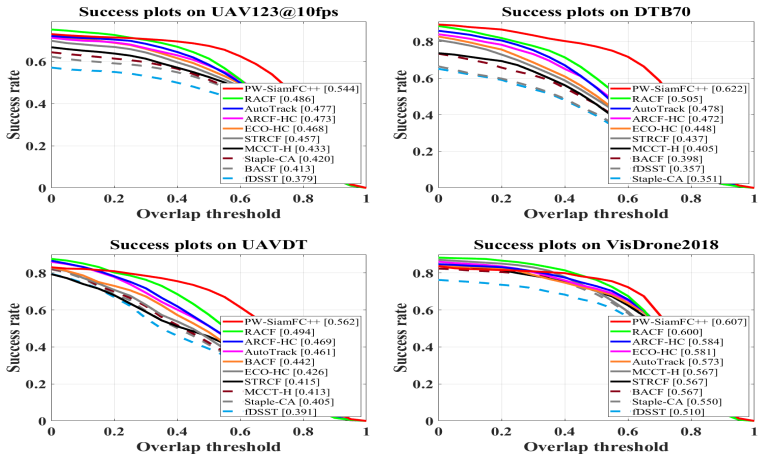

Figure 3. Overall performance of ten hand-crafted based trackers and our PW-SiamFC++ on the four benchmarks. The area under curve (AUC) are used for marked in the success plots respectively and ranking .

Table 1. Average FPS(speed) and precision comparision between hand-crafted based trackers and PW-SiamFC++ on DTB70, VisDrone2018, UAVDT and UAV123@10fps . On a single GPU we evaluated all the reported FPS. Note that PW-SiamFC++ is the best real-time tracker (with speed >30FPS) on CPU.green , blue and Red respectively:the third place, second and first.

	KCF [17]	fDSST [18]	BACF [19]	Staple-CA [20]	ECO-HC [21]	STRCF [22]	MCCT-H [23]	ARCF-HC [24]	AutoTrack [1]	RACF [8]	**PW-SiamFC++ Ours**
Precision	53.3	60.4	65.3	64.2	68.8	67.1	66.8	71.9	**72.3**	**75.7**	**78.1**
FPS (CPU)	**655.6**	**203.6**	57.0	67.7	**88.9**	29.9	66.7	36.0	61.8	37.5	64.8

Quantitative assessment: Fig. 3 shows the success plots of the trackers on the four benchmarks. As can be seen, PW-SiamFC++ surpassed all other trackers to a large margin on three benchmarks, i.e., UAVDT , DTB70 and UAV123@10fps. Specifically, PW-SiamFC++ dramatically outperforms RACF, the second best tracker, in terms of area under curve (AUC), with gains of 5.8%, 11.7% and 6.8%, respectively. We use average FPS on a single CPU over the four benchmarks as the metric of speed. Table 1 demonstrates the FPS and average precision of the competing trackers. We can see that PW-SiamFC++ in precision outperforms all the competing trackers and it is also real-time on a single CPU (speed >30FPS), specifically 64.8 FPS.

Qualitative assessment: In Fig. 4, we qualitatively compare our method with four top CPU-based trackers, i.e., RACF [8], AutoTrack [1], ARCF-HC, and ECO-HC [24], on 2 sequences from UAV123@10fps [13] and DTB70 [14], (i.e. person10 and BMX4), respectively. Different colors represent different methods. It can be seen in these challenging situations that all trackers except of ours finally fail to track the objects when objects are experiencing large deformations (i.e., BMX4) and severe occlusion (i.e., person10). However, our PW-SiamFC++ is more satisfying and performs better in these examples, which can be attributed to the powerful deep representation learning inherent in our model.

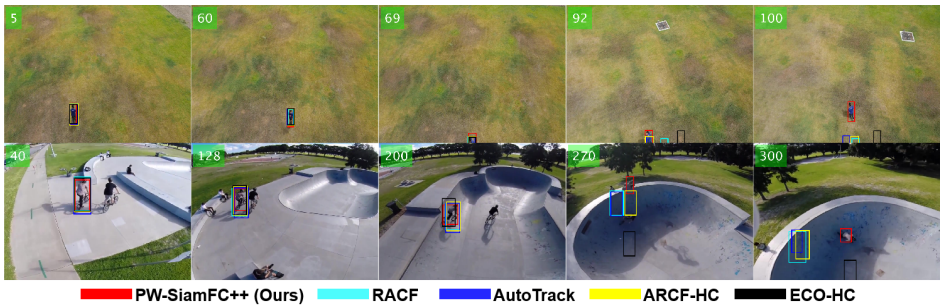

Figure 4. Two sequences for qualitative evaluation from DTB70 [14] and UAV123@10fps [13], (i.e. person10 and BMX4), respectively.

Table 2. FPS(speed) and precision comparison between deep-based trackers and PW-SiamFC++ on UAVDT [15]. On a single GPU we evaluated all the reported FPS. Note that ρ indicates the global pruning ratio.

	KYS [25]	SiamR-CNN [26]	PrDimp18 [27]	D3S [28]	SiamGAT [29]	HiFT [2]	LightTrank [30]	SOAT [31]	TransT [32]	AutoMatch [33]	**PW-SiamFC++(Ours)** $\rho = 0.3$	$\rho = 0.4$
Precision	79.8	66.5	73.2	72.2	76.4	65.2	80.4	**82.1**	**82.6**	**82.1**	**81.3**	79.2
FPS (GPU)	30.2	7.2	48.5	44.6	74.8	**135.3**	84.8	29.4	42.1	50.4	**259.0**	**266.9**

Table 3. How the precision of PW SiamFC++ varies with the global pruning ratio, which ranges from 0.8 to 0.1 in step of 0.1. The improved precisions by the dynamic channel weighting component are marked in bold.

ρ	UAV123@10fps		DTB70		UAVDT		VisDrone2018	
	w/o	w/	w/o	w/	w/o	w/	w/o	w/
0.1	70.8	**70.9**	79.6	**80.5**	**81.4**	79.6	**79.6**	77.6
0.2	71.6	**72.2**	80.0	**80.1**	76.9	**81.6**	**80.2**	74.5
0.3	71.3	**72.3**	**81.0**	80.0	**83.9**	81.3	75.6	**76.5**
0.4	**71.9**	71.5	79.5	**81.1**	78.8	**79.2**	79.3	**79.9**
0.5	**71.1**	70.2	77.6	**79.2**	77.5	**78.9**	72.7	**77.0**
0.6	68.7	**68.8**	78.6	**79.1**	76.6	**77.8**	**75.2**	75.1
0.7	**69.1**	66.9	77.9	**78.7**	77.8	**79.4**	76.7	**80.0**
0.8	**65.2**	66.4	**76.4**	74.8	74.6	**75.9**	71.8	**72.2**

4.2. Comparison with DL-based Trackers

The proposed PW-SiamFC++ is also compared with ten state-of-the-art DL-based trackers, namely, KYS [25], SiamR-CNN [26], PrDiMP18 [27], D3S [28], SiamGAT [29], HiFT [2], LightTrack [30], SOAT [31], TransT [32], and AutoMatch [33]. Table 2 shows the precisions and the FPSs on UAVDT. As is shown, although PW-SiamFC++ (pruning ratio $\rho = 0.4$) is not as accurate as TransT, SOAT, AutoMatch and LightTrank, the gap is less than 3.5% and, remarkably, PW-SiamFC++ is 6 times faster than the first tracker TransT on GPU. Moreover, when a smaller pruning ratio, i.e., $\rho = 0.3$, is chosen, PW-SiamFC++ achieves 81.3% in precision, a smaller gap of 1.3% to the first tracker TransT, and still gets 259.0 FPS in GPU speed, much faster than the first two Trackers. This proves, in terms of speed and accuracy, that PW-SiamFC++ can achieve a good balance.

4.3. Ablation Study

Impact of dynamic channel weighting: We finetuned PW-SiamFC++ with and without the dynamic channel weighting at different global pruning ratios to understand the impact of this proposed component. Specifically, each convolutional layer in the head, neck and backbone is pruned with the same global ratio, ranging from 0.8 to 0.1. Note that the

larger the ratio ρ is, the more filters will be pruned. The precisions of PW-SiamFC++ with and without the proposed dynamic channel weighting with respect to the global pruning ratio are shown in Table 3. As can be seen, the highest precisions are basically achieved at pruning ratios less than 0.4, which is in line with our expectation on filter pruning techniques, namely, pruning may compromise models' accuracy. Surprisingly, most precisions are improved with the proposed dynamic channel weighting incorporated, especially at higher pruning ratios. This enables us to maintain favorable precisions with higher pruning ratios and demonstrates the effectiveness of the proposed dynamic channel weighting in pursuing a higher global pruning ratio for boosting efficiency.

5. Conclusion

In this work, we use filter pruning and the proposed dynamic channel weighting to trade off efficiency and precision in UAV tracking and achieve state-of-the-art performance on four public benchmarks. Experimental results show that the proposed dynamic channel weighting strategy is beneficial to pursue a higher global pruning ratio when applying rank-based filter pruning to improve the efficiency of DL-based trackers for UAV tracking. Remarkably, the proposed PW-SiamFC++ not only significantly improves efficiency over the baseline SiamFC++ but also increases precision on DTB70, UAVDT, and VisDrone2018, well combating the adverse effects (i.e., a precision drop) of using filter pruning. It's worth noting that in this work, we only employ a ranked-based filter pruning to compress the baseline SiamFC++. Our future works include investigating other filter pruning methods, better pruning criterion, and other baseline trackers.

6. Acknowledgement

Thanks to the Guangxi Science and Technology Base and Talent Special Project (No. Guike AD22035127)

References

[1] Y. Li, C. Fu, F. Ding, and et al., "Autotrack: Towards high-performance visual tracking for uav with automatic spatio-temporal regularization," in *Proceedings of CVPR*, 2020, pp. 11 920–11 929.

[2] Z. Cao, C. Fu, J. Ye, and et al., "Hift: Hierarchical feature transformer for aerial tracking," in *Proceedings of ICCV*, 2021, pp. 15 457–15 466.

[3] T. Choudhary, V. K. Mishra, A. Goswami, and et al., "A comprehensive survey on model compression and acceleration," *Artificial Intelligence Review(AIR)*, pp. 1–43, 2020.

[4] H. Wang, C. Qin, Y. Zhang, and et al., "Emerging paradigms of neural network pruning," *arXiv preprint arXiv:2103.06460*, 2021.

[5] Y. Xu, Z. Wang, Z. Li, and et al., "Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines," in *Proceedings of AAAI*, vol. 34, no. 07, 2020, pp. 12 549–12 556.

[6] M. Lin, R. Ji, Y. Wang, and et al., "Hrank: Filter pruning using high-rank feature map," in *Proceedings of CVPR*, 2020, pp. 1526–1535.

[7] D. S. Bolme, J. R. Beveridge, B. A. Draper, and et al., "Visual object tracking using adaptive correlation filters," in *Proceedings of CVPR*, 2010, pp. 2544–2550.

[8] S. Li, Y. Liu, Q. Zhao, and et al., "Learning residue-aware correlation filters and refining scale estimates with the grabcut for real-time uav tracking," in *Proceedings of 3DV*, 2021, pp. 1238–1248.

[9] L. Bertinetto, J. Valmadre, J. F. Henriques, and et al., "Fully-convolutional siamese networks for object tracking," in *Proceedings of ECCV*. Springer, 2016, pp. 850–865.

[10] B. Li, W. Wu, Q. Wang, and et al., "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of CVPR*, 2019, pp. 4277–4286.
[11] Z. Chen, B. Zhong, G. Li, and et al., "Siamese box adaptive network for visual tracking," in *Proceedings of CVPR*, 2020, pp. 6668–6677.
[12] K. He, X. Zhang, S. Ren, and et al., "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of ICCV*, 2015, pp. 1026–1034.
[13] M. Mueller and et al., "A benchmark and simulator for uav tracking," *Proceedings of ECCV*, 2016.
[14] S. Li and D.-Y. Yeung, "Visual object tracking for unmanned aerial vehicles: A benchmark and new motion models," in *Proceedings of AAAI*, 2017.
[15] D. Du, Y. Qi, H. Yu, and et al., "The unmanned aerial vehicle benchmark: Object detection and tracking," in *Proceedings of ECCV*, 2018, pp. 370–386.
[16] L. Wen, P. Zhu, D. Du, and et al., "Visdrone-sot2018: The vision meets drone single-object tracking challenge results," in *Proceedings of ECCV Workshops*, 2018, pp. 0–0.
[17] J. F. Henriques, R. Caseiro, and et al., "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence(TPAMI)*, vol. 37, pp. 583–596, 2015.
[18] M. Danelljan, G. Häger, F. S. Khan, and et al., "Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking," in *Proceedings of CVPR*, 2016, pp. 1430–1438.
[19] H. K. Galoogahi, A. Fagg, and S. Lucey, "Learning background-aware correlation filters for visual tracking," in *Proceedings of ICCV*, 2017, pp. 1144–1152.
[20] M. Mueller, N. G. Smith, and B. Ghanem, "Context-aware correlation filter tracking," in *Proceedings of CVPR*, 2017, pp. 1387–1395.
[21] M. Danelljan, G. Bhat, F. S. Khan, and et al., "Eco: Efficient convolution operators for tracking," in *Proceedings of CVPR*, 2017, pp. 6931–6939.
[22] F. Li, C. Tian, W. Zuo, and et al., "Learning spatial-temporal regularized correlation filters for visual tracking," in *Proceedings of CVPR*, 2018, pp. 4904–4913.
[23] N. Wang, W. gang Zhou, Q. Tian, and et al., "Multi-cue correlation filters for robust visual tracking," in *Proceedings of CVPR*, 2018, pp. 4844–4853.
[24] Z. Huang, C. Fu, Y. Li, and et al., "Learning aberrance repressed correlation filters for real-time uav tracking," in *Proceedings of ICCV*, 2019, pp. 2891–2900.
[25] G. Bhat, M. Danelljan, L. V. Gool, and et al., "Know your surroundings: Exploiting scene information for object tracking," in *Proceedings of ECCV*. Springer, 2020, pp. 205–221.
[26] P. Voigtlaender, J. Luiten, P. H. S. Torr, and et al., "Siam r-cnn: Visual tracking by re-detection," in *Proceedings of CVPR*, 2020, pp. 6577–6587.
[27] M. Danelljan, L. V. Gool, and R. Timofte, "Probabilistic regression for visual tracking," in *Proceedings of CVPR*, 2020, pp. 7181–7190.
[28] A. Lukezic, J. Matas, and M. Kristan, "D3s – a discriminative single shot segmentation tracker," in *Proceedings of CVPR*, 2020, pp. 7133–7142.
[29] D. Guo, Y. Shao, and et al., "Graph attention tracking," in *Proceedings of CVPR*, 2021, pp. 9538–9547.
[30] B. Yan, H. Peng, K. Wu, and et al., "Lighttrack: Finding lightweight neural networks for object tracking via one-shot architecture search," in *Proceedings of CVPR*, 2021, pp. 15 175–15 184.
[31] Z. Zhou, W. Pei, X. Li, and et al., "Saliency-associated object tracking," in *Proceedings of ICCV*, vol. abs/2108.03637, 2021.
[32] X. Chen, B. Yan, and et al., "Transformer tracking," in *Proceedings of CVPR*, 2021, pp. 8122–8131.
[33] Z. Zhang, Y. Liu, X. Wang, and et al., "Learn to match: Automatic matching network design for visual tracking," in *Proceedings of ICCV*, vol. abs/2108.00803, 2021.