Sample Purification-Aware Correlation Filters for UAV Tracking with Cooperative Deep Features

Changhong Fu*, Fuling Lin, Fan Li, and Yujie He

https://vision4robotics.github.io

Introduction

Challenges & Motivations

- The representations of a single feature restrict the discriminant power of the correlation filter.
- → **Cooperating deep features with different encoding ability**
- UAV tracking scenarios exist many challenging issues, e.g., viewpoint variations, the object/camera fast motion, occlusion, and background clutters, resulting in the corrupted training samples and suboptimal encoding ability of the appearance model.
- → **Sample purification to construct the training set with high confidence**

Contribution 1: Cooperative features

Objective

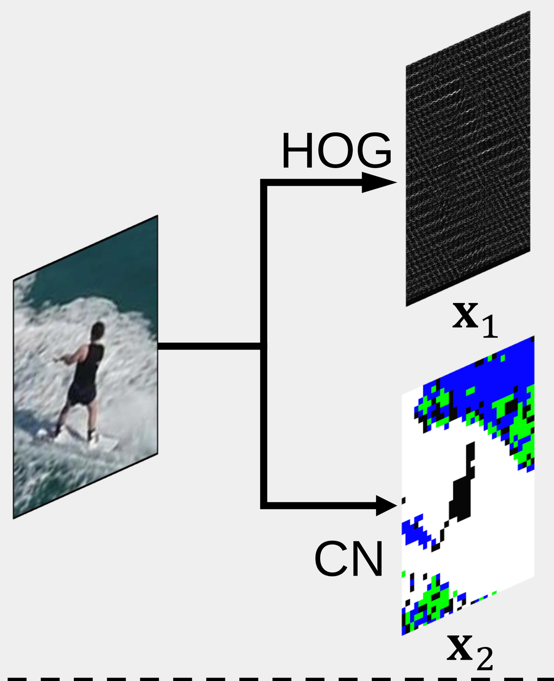
Learn multiple kernelized correlation filters for N different features

Loss function

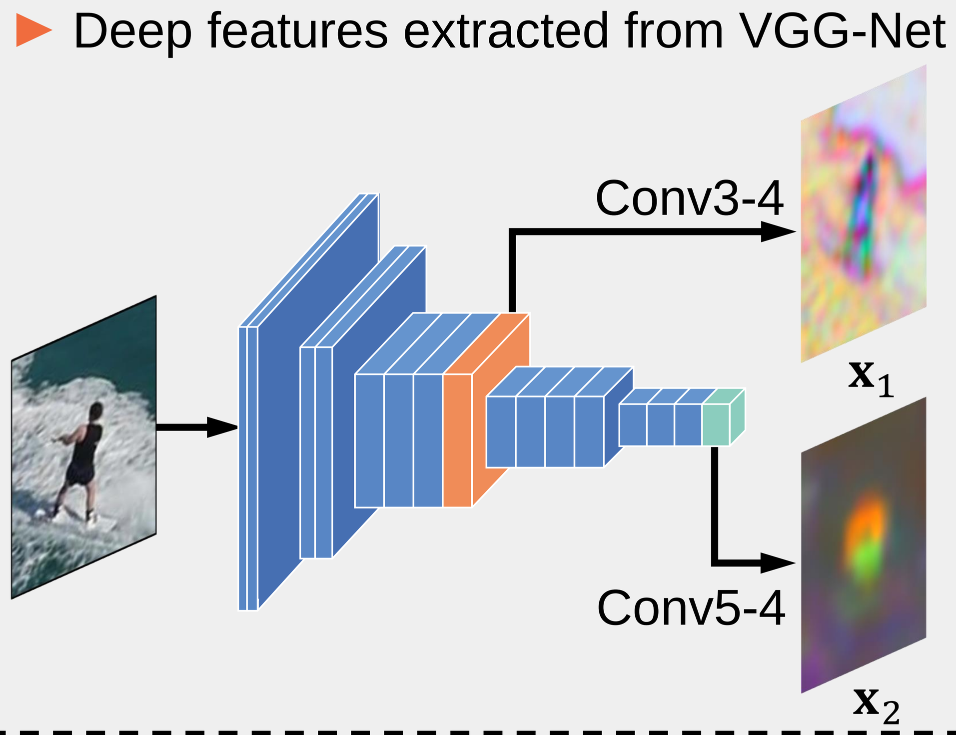
$$\hat{\mathcal{L}}(\mathbf{w}_n^*) = \sum_{n=1}^{N}\left(\|\hat{\mathcal{C}}_n(\mathbf{x}_n,\mathbf{x}_n) - \hat{\mathbf{y}}_n\|_2^2 + \lambda_n \hat{\mathbf{k}}^{\mathbf{x}_n\mathbf{x}_n}\|\hat{\mathbf{w}}_n^*\|_2^2\right) + \gamma\sum_{i,j=1}^{N}\|\hat{\mathcal{C}}_i(\mathbf{x}_i,\mathbf{x}_i) - \hat{\mathcal{C}}_j(\mathbf{x}_j,\mathbf{x}_j)\|_2^2$$

Optional features combination

- Hand-crafted features



- Deep features extracted from VGG-Net



Contribution 2: Sample purification

Objective

Integrate the sample weights $\{\alpha_n\}$ and temporal weights $\{t\}$ into the cooperative kernelized correlation filters

Loss function

$$\hat{\mathcal{J}}(\alpha_n^s,\mathbf{w}_n^*) = \sum_{n=1}^{N}\sum_{s=1}^{S}\alpha_n^s\left(\lambda_n\hat{\mathbf{k}}^{\mathbf{x}_n^s\mathbf{x}_n^s}\|\hat{\mathbf{w}}_n^*\|_2^2 + \|\hat{\mathcal{C}}_n^s(\mathbf{x}_n^s,\mathbf{x}_n^s) - \hat{\mathbf{y}}_n\|_2^2\right)$$
$$+\gamma\sum_{i,j=1}^{N}\sum_{s=1}^{S}\|\hat{\mathcal{C}}_i^s(\mathbf{x}_i^s,\mathbf{x}_i^s) - \hat{\mathcal{C}}_j^s(\mathbf{x}_j^s,\mathbf{x}_j^s)\|_2^2 + \sum_{n=1}^{N}\left(\mu_n\sum_{s=1}^{S}\frac{(\alpha_n^s)^2}{t^s}\right)$$

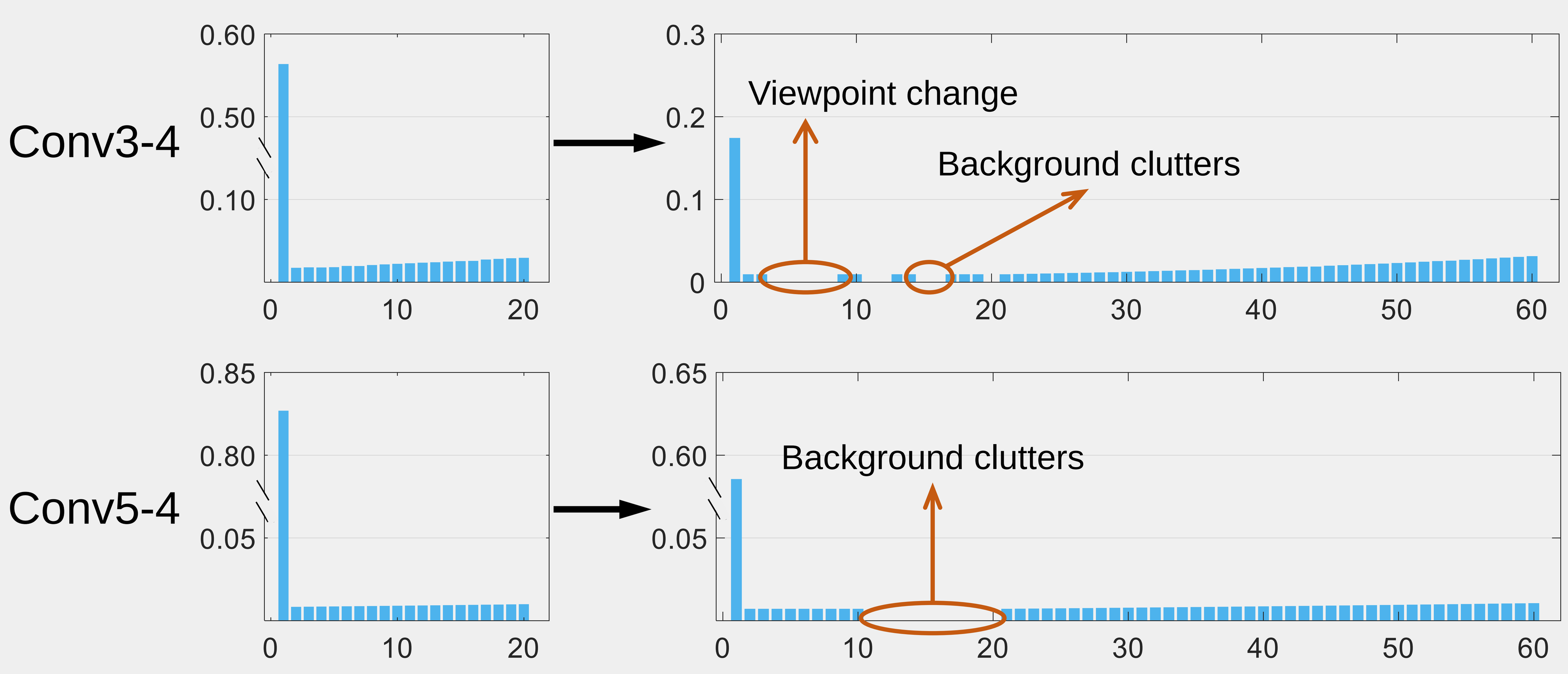
Remark

- The function is constituted by both $\mathbf{w}_n^*$ and the sample weights $\{\alpha_n\}$.
- S is the number of samples in the training-set. s_0 is the most recent samples.
- $a = (S - s_0 + \frac{(1-q)^{-s_0}-1}{q})^{-1}$ is determined by the condition $\sum_s t^s = 1$. Each t^s is calculated as:

$$t^s = \begin{cases} a, & s = 1,\dots,S-s_0 \\ a(1-q)^{S-s_0-s}, & s = S-s_0+1,\dots,S \end{cases}$$

- Samples weights can update adaptively.

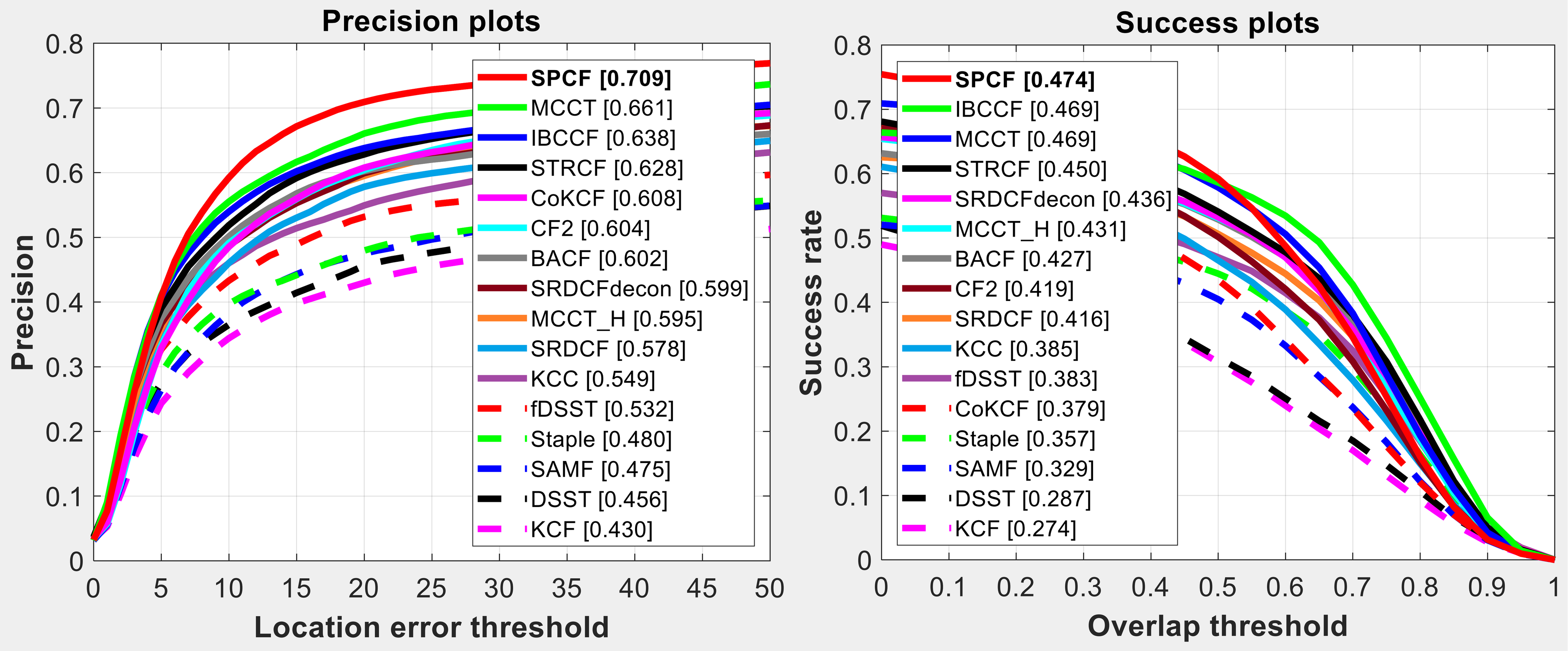
▷ $\{\alpha_n\}$ initialization ▷ $\{\alpha_n\}$ redistribution



Experiments

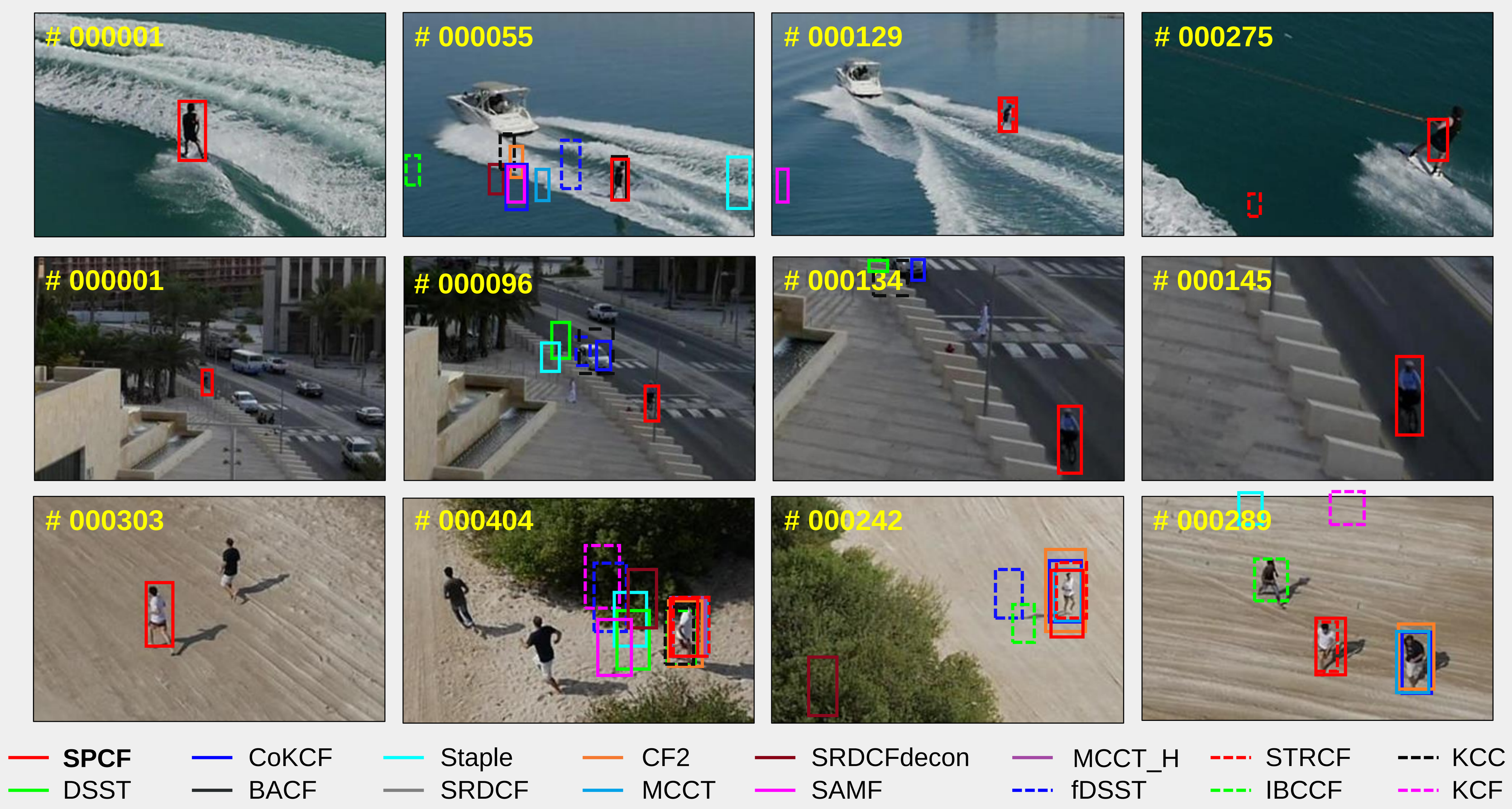
The UAV123 benchmark protocol [1] is used. Experimental results demonstrate that **SPCF** outperforms favorably against other 15 state-of-the-art trackers.

Overall performance



Examples of the UAV tracking results

The first, second, and third columns show the tracking results on the image sequences from *wakeboard3_1*, *bike3*, and *group2_2*.



Attribute-based performance

Attributes	Precision (CLE = 20 pixels)		Success (AUC score)	
	Best of others	**SPCF**	Best of others	**SPCF**
Camera motion	0.635	**0.708**	0.464	**0.480**
Fast motion	0.470	**0.552**	0.309	**0.318**
Partial occlusion	0.591	**0.628**	0.392	**0.399**
Viewpoint change	0.583	**0.662**	0.419	**0.439**
Aspect ratio change	0.573	**0.644**	0.387	**0.401**

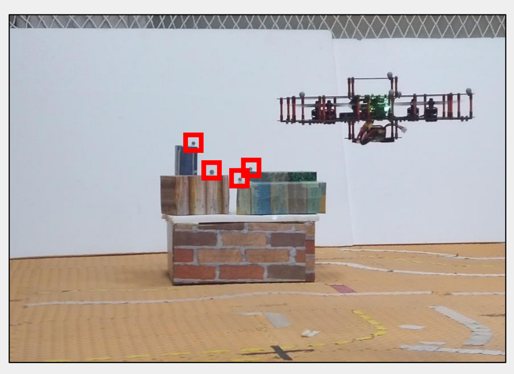
Conclusion

- A new integrated learning method for multiple features is devised.
- A novel developed joint training framework is proposed to learn both sample weights and filters (**SPCF**).
- Extensive experiments on challenging UAV image sequences demonstrate that **SPCF** performs favorably against other 15 state-of-the-art trackers in accuracy and robustness.

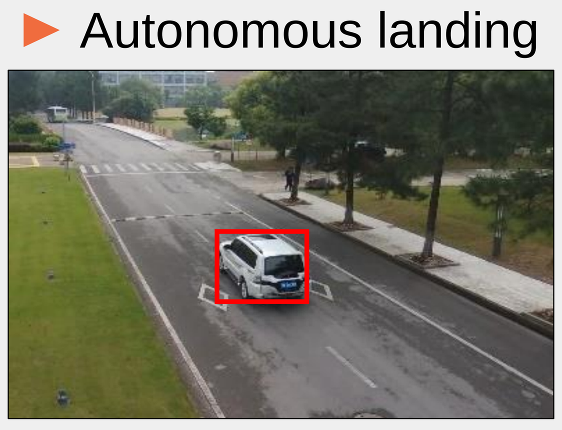
Application

The SPCF tracker is also applied for various challenging tasks currently.

- Pose estimation



- Autonomous landing



- RGB-T tracking

Reference

[1] M. Mueller, N. Smith, and B. Ghanem, “A benchmark and simulator for uav tracking,” in Proceedings of the European Conference on Computer Vision (ECCV), 2016, pp. 445–461.

Source code