Signal Processing 167 (2020) 107324

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Surrounding-aware correlation filter for UAV tracking with selective spatial regularization

Changhong Fu[a,*], Weijiang Xiong[a], Fuling Lin[a], Yufeng Yue[b,*]

[a] School of Mechanical Engineering, Tongji University, Shanghai 201804, China
[b] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore

ARTICLE INFO

Article history:
Received 28 April 2019
Revised 22 September 2019
Accepted 29 September 2019
Available online 30 September 2019

Keywords:
Unmanned aerial vehicle (UAV)
Visual object tracking
Discriminative correlation filter
Surrounding information
Selective spatial regularization

ABSTRACT

The great advance of visual object tracking has provided unmanned aerial vehicle (UAV) with intriguing capability for various practical applications. With promising performance and efficiency, discriminative correlation filter-based trackers have drawn great attention and undergone remarkable progress. However, background interference and boundary effect remain two thorny problems. In this paper, a surrounding-aware tracker with selective spatial regularization (SASR) is presented. SASR tracker extracts surrounding samples according to the size and shape of the object in order to utilize context and maintain the integrality of the object. Additionally, a selective spatial regularizer is introduced to address boundary effect. Central coefficients in the filter are evenly regularized to preserve valid information from the object. While the others are penalized according to their spatial location. Under the framework of SASR tracker, surrounding information and selective spatial regularization prove to be complementary to each other, which actually did not draw much attention before. They managed to improve not only the robustness against various distractions in the surrounding but also the flexibility to catch up with frequent appearance change of the object. Qualitative evaluation and quantitative experiments on challenging UAV tracking sequences have shown that SASR tracker has performed favorably against 23 state-of-the-art trackers.

1. Introduction

With high flexibility and maneuverability, unmanned aerial vehicle (UAV) has been contributing to various public services, such as visual surveillance [1], emergency response [2], humanitarian rescue [3] and local logistics [4], during which visual object tracking serves as a crucial function module. Visual object tracking is one of the most popular topics in intelligent UAV application. The task of UAV tracking can be defined in an operational manner: tracking is the analysis of video sequences for the purpose of establishing the location of the object over a sequence of frames starting from the bounding box given in the first frame [5]. Although diverse trackers have been designed for UAV tracking [6–10], it remains a challenging task because of background clutters, rotation, fast motion and deformation, which frequently occur during object tracking. Those scenarios become even more difficult as a result of camera motion and viewpoint change [11]. This paper investigates the problem of designing an accurate and robust tracker for UAV tracking tasks under capricious conditions and proposes a novel approach to address them.

Recently, the advent of discriminative correlation filter (DCF) based trackers has stimulated worldwide research interests through intriguing efficiency and accuracy [12,13]. On the assumption that a circular shift is applied to the initial sample, a set of shifted samples are actually generated, forming a circulant matrix. Knowing that the Fourier matrix can diagonalize any circulant matrix [14], a robust filter can be efficiently trained in Fourier domain through element-wise production. However, circular shift involves boundary effect [15] at the same time, and the negative samples used in training step actually do not exist in the “real-world” [16]. It is those invalid samples that impose strict restrictions on the region of the sample images. For concern about side-effect brought by corrupted samples, the filter is usually trained on a relatively small image patch, containing inadequate surrounding information. As a result, the tracker may be easily distracted by other objects in the background.

In order to mitigate boundary effect, Danelljan et al. [15] introduce spatial regularization and propose SRDCF tracker. With the coefficient spatially penalized according to their distance to the

* Corresponding authors.
E-mail addresses: changhongfu@tongji.edu.cn (C. Fu), yueyufeng@ntu.edu.sg (Y. Yue).

https://doi.org/10.1016/j.sigpro.2019.107324
0165-1684/

center, the tracker is expected to focus on information near the center. Thereby, the search region can be significantly expanded and more surrounding information can be included. Similarly, aiming to learn more background features, Mueller et al. [11] employ global context and come up with context-aware correlation filter (CACF) for tracking. Same as the object patch, circulant shift is also applied to all patches directly cropped from the image. Negative training samples are thus densely extracted around the object, providing abundant valid surrounding information. Both methods prove to be effective for enhancing the robustness of the tracker against disturbance from the background.

Despite the compatibility of the aforementioned methods, the promising cooperation between the two frameworks has long been overlooked. In this work, a novel tracker is presented, combining effective surrounding information and selective spatial regularization together. Though both theoretical derivation and practical experiments, the two flexible components have proved complementary to each other. Spatial regularization relieves boundary effect brought by extra surrounding patches. In return, large quantities of valid negative samples, which is the aim of expanding the search region in SRDCF [15], are directly provided by surrounding patches (Fig. 3). For the purpose of focusing on the whole object instead of the central part alone, an elliptic-paraboloid-shaped regularizer with a protective area is employed (as shown in Fig. 5). Rather than taking surrounding samples next to the object box, their positions are chosen slightly farther from the object (as shown in Fig. 2(c)) based on deeper analysis and experiments. A typical error function chosen for DCF trackers is ridge regression, for which the optimization problem is convex [17]. Therefore, the alternating direction method of multipliers (ADMM) [18] can be employed to solve the problem efficiently.

The main contributions of this work are summarized below:

- A novel discriminative tracker, i.e., SASR, is presented for UAV tracking, incorporating collaborative surrounding information and spatial regularization.
- A new regularizer is introduced to repress futile boundary information in shifted samples and emphasize the whole object throughout tracking tasks.
- A balanced strategy, which selects a set of complementary surrounding patches, is proposed. In addition, it offers reasonably enlarged visual field (shown in Fig. 3) for the tracker.
- Using the ADMM method, an efficient optimization algorithm is formulated to obtain a solution for a tracker with multiple patches and spatial regularization.

Qualitative evaluation and quantitative experiments on challenging UAV tracking sequences have shown that SASR tracker performs favorably against 23 state-of-the-art trackers. To the best of our knowledge, this is the first time that a unified approach combining surrounding information with selective spatial regularization, i.e., SASR tracker, is proposed and employed for UAV tracking.

The rest of this paper is organized as follows. Section 2 gives a brief review of the previous literature most relevant to this work. Details about the proposed SASR tracker are introduced in Section 3. Section 4 presents the detailed results of extensive experiments and discussion about future work. Conclusions are finally drawn in Section 5.

2. Related work

Visual tracking is a favorable application for UAV. Given the first frame and the location of the object, a tracker is required to estimate the trajectory of the object in oncoming sequences. In recent years, the tracking community has done overall and profound research. Many impressive achievements having been made, it is beyond the scope of this paper to give a comprehensive review of UAV tracking. The following five sections discuss some representative tracking approaches that are tightly related to SASR. Each of the sections is connected with an essential component of SASR.

2.1. Correlation filter-based tracking

On account of high computational efficiency, correlation filter (CF) has drawn increasing attention among the tracking community. Generally, CF-based trackers take effect by discriminating the object from the background. Bolme et al. [19] propose to learn a filter through minimum output sum of squared error, i.e., MOSSE tracker. Introduced by Henriques et al. [12], kernel method serves as an important component in CSK tracker that uses circulant structure with kernels. Ridge regression is used to replace the sum of squared error in case of overfitting. Henriques et al. [13] later incorporate a more powerful multi-channel feature, i.e., histogram of oriented gradients (HOG) [20], with CSK and extend it into KCF tracker. Danelljan et al. [21] investigate the effect of color information in the tracking process. The color names feature (CN) [22] used in [21] describes the color information and proves complementary to HOG that highlights the outline of the object. The fusion of HOG and CN, which are known as hand-crafted features, are widely employed by subsequent trackers that demonstrate state-of-the-art performance while maintaining high computational efficiency [23,24]. In order to estimate the scale variance of the object, Danelljan et al. [25] come up with discriminative scale space tracking, i.e., DSST tracker. By attaching an extra scale filter to DCF framework, DSST is capable of adapting to scale variance of the object. Despite CF-based trackers have made great progress, they are still troubled by boundary effect which is an inherent shortcoming brought by circulant shift.

2.2. Boundary effect-aware tracking

As is inferred by its name, boundary effect mainly takes place near the edge of an input image patch. Especially when the original base sample is shifted by half the length of the patch, the object will be split by the edge. Half of the object will lie along the upper edge, while the other half along the lower edge as illustrated in Fig. 4. For the purpose of relieving boundary effect, Danelljan et al. [15] propose to learn spatially regularized correlation filter, i.e., SRDCF tracker. With a more general ridge regression, the coefficients in the filter are regularized according to their Euclidean distance to the center point of the current bounding box. The longer the distance is, the heavier the penalization will be. In this way, more attention is paid to the central part and boundary effect is partly avoided since it takes place near the edge of the sample patches. Later, SRDCF is extended to SRDCFdecon [26] tracker, which adaptively updates according to the quality of the samples. More recently, motivated by online passive-aggressive algorithms [27], Li et al. [28] propose to learn a spatial-temporal regularized correlation filter, i.e., STRCF tracker. By incorporating a temporal regularizer with SRDCF, STRCF tracker learns a more robust object model in case of steep appearance variation. However, those regularized trackers still lack valid surrounding information as mentioned before.

2.3. Background information-aware tracking

In order to learn a more discriminative tracker, surrounding information is indispensable since the mechanism of CF-based tracker is to distinguish the object from the background. Kiani Galoogahi et al. [29] suggest learning a background-aware correlation filter, i.e., BACF tracker. By applying a circulant shift directly to the entire frame and cropping central elements from each shifted

sample, the tracker is then trained with real-world negative samples. Mueller et al. [11] propose a context-aware correlation filter, i.e., CACF tracker. The CACF tracker attaches four context patches tightly next to the object patch to cover four directions. The context patches posses the same circulant structure as the object patch, providing additional context information. Rich experiments on four trackers [13,19,23,24] without spatial regularization have shown that the extended tracker can be trained almost in the same efficient way as KCF [13]. With more surrounding information integrated into the training set, the filter is more robust to distractions in the environment. However, practical experiments have shown that those trackers still lack representative information because of simple hand-crafted features.

2.4. Convolutional feature-based tracking

Features occupy an essential position in visual tracking algorithms [30]. With extraordinary discriminative power, convolutional neural networks (CNNs) have been successfully applied to image classification [31]. Ma et al. [32] employ pure convolutional features extracted from CNN in CF2 tracker and achieve remarkable performance improvement. Li et al. [33] introduce a family of 1D boundary filters to localize the four boundaries of the object and develop a method to cope with aspect ratio variation. Wang et al. [34] propose multi-cue correlation filters, i.e., MCCT tracker. By constructing multiple experts from the same feature pool and combine the output of every expert, the tracker makes a more reasonable decision when encountered with complicated scenarios. Danelljan et al. [35] introduce continuous convolution operators for visual tracking, i.e., C-COT tracker. An implicit interpolation is applied to the original discrete data, and the learning problem is then mapped into the continuous spatial domain. Furthermore, Danelljan et al. [36] propose an efficient convolution operator, i.e., ECO tracker, based on C-COT [35]. With considerable redundancy removed from the model, great improvement is made in both efficiency and performance. However, those deep trackers have also overlooked the benefit of surrounding information.

2.5. End-to-end learning for tracking

In recent years, great progress in datasets, deep learning frameworks, and GPU devices has made it possible to directly train a neural network for tracking tasks. Yun et al. [37] propose a tracker controlled by sequentially pursuing actions learned by deep reinforcement learning. Valmadre et al. [38] interpret the correlation filter learner, which has a closed-form solution, as a differentiable layer in a deep neural network. Siamese network-based trackers have also received significant attentions [38–40]. They formulate visual tracking as a cross-correlation problem and are expected to better utilize the advantages of deep neural networks from end-to-end learning. Tao et al. [39] propose to learn a ubiquitous matching function to search an instance that is most similar to the object in the first frame. Bertinetto et al. [40] equip a basic tracking algorithm with a fully-convolutional Siamese network. Despite remarkable progress, those trackers pay little attention to online-learning of the object and rely heavily on training data.

3. Proposed tracking approach

As discussed in previous sections, circulant shift evokes corrupted samples, corrupted samples arouses boundary effect, boundary effect limits sample region and limited region leads to background ignorance. Although efforts have been made to alleviate the intricate problem either directly from background ignorance or boundary effect that causes it, little attention, however, has been paid to construct an integrated approach to cope with them all together.

Table 1
Notations.

Denotation	Symbol	Note
Desired response	$\mathbf{y}$	Gaussian-shaped label
The sample from k-th patch	$\mathbf{x}_k^d$	The d-th channel
Proposed filter	$\mathbf{w}$	$M \times N$ matrix
Auxiliary variable	$\mathbf{g}$	$\mathbf{g} = \mathbf{w}$
Selective spatial regularizer	$\boldsymbol{\psi}$	Constant $M \times N$ matrix
The weight of the k-th patch	α_k	Scalar

Section 3.1 shows the workflow of the SASR framework, which is proposed as a unified approach to address the problem above. Section 3.2 derives the training formula of SASR tracker, aiming to utilize surrounding information and relieving boundary effect at the same time. Meanwhile, a reasonable approximation is made to accelerate training step. Section 3.3 gives a strategy to choose more appropriate surrounding patches. Section 3.4 introduces the property and merits of selective spatial regularization. Section 3.5 describes the update method.

3.1. Tracking pipeline

To facilitate the understanding of SASR tracking framework, Table 1 presents the notations of some key components in SASR tracker.

Fig. 1 illustrates the workflow of the SASR tracker. Generally, the whole procedure consists of sampling, feature extraction, training, and detection. When receiving the first input frame at the start of tracking progress, what the tracker has is just the object position and size. With the strategy mentioned in Algorithm 1, the tracker first calculates the positions for K surrounding patches and then crop them off together with the object patch. All $K+1$ patches are weighted by cosine window and then sent to feature extractor which consists of a deep convolutional network [31] and a hand-crafted extractor. The output of the 4-th convolution layer is gathered to provide higher-level semantic information. Meanwhile, the hand-crafted extractor send out silhouette features (fHOG [41]) and color features (CN [21]). Then, all the features $\mathbf{x}$ together with a Gaussian label function $\mathbf{y}$ is utilized to train a selectively regularized filter $\mathbf{w}$ by minimizing Eq. (1). The trained filter is then used in the subsequent detection step.

When receiving the next frame, a set of features $\mathbf{X}$ is extracted under different scales for the purpose of adapting to scale variation. The detection step is performed by taking element-wise product between the filter $\mathbf{w}$ and each possible shifted sample of the feature set $\mathbf{X}$. The filter sends out a high response when detecting the object and low response when encountered with something else. That is equivalent to taking the spatial correlation between $\mathbf{X}$ and $\mathbf{w}$, which is, in practice, obtained by dot product in the Fourier domain. $\mathbf{X} \star \mathbf{w}$ is known as the response map, whose maximum value tells the position and size of the object in this frame. Based on the new location of the object, SASR updates the model, moves to the new location, extract the object feature set and retrain the filter to get a new one at this frame, just like the training step mentioned above.

The fHOG feature is an optimized version of HOG, which preserves all information and achieves significant speed-up. The dimensions of fHOG, CN and deep feature used in SASR are 31, 11, and 512 respectively. They are not illustrated in full because of limited space.

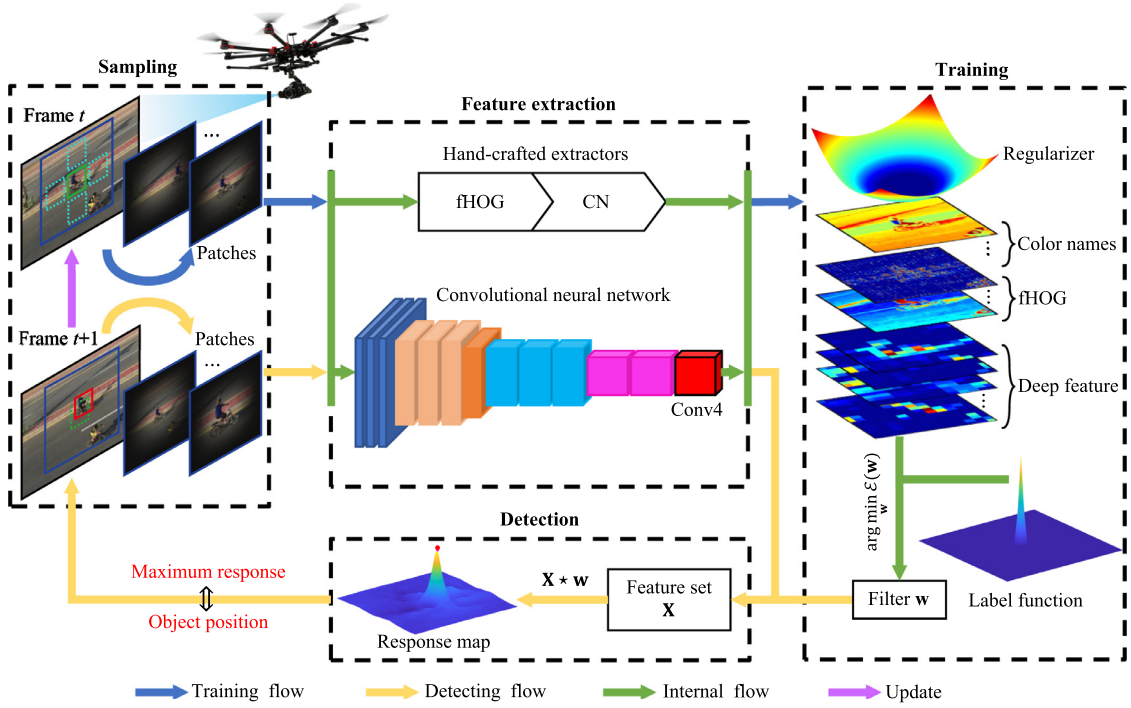


Fig. 1. Tracking workflow of the proposed SASR tracker. Starting with an input frame (upper left) and the object location, the SASR tracker follows the blue arrow and goes through patches sampling, feature extraction, and training. The sampling process follows the strategy in Section 3.3. The trained filter $\mathbf{w}$ is then used to detect the object. In the next frame (lower left), SASR first follows the gold arrow and goes through feature extraction to obtain the feature set $\mathbf{X}$. Then, the spatial correlation between $\mathbf{X}$ and $\mathbf{w}$ will give out the object location. After that, SASR updates its model and moves back to the training loop. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

3.2. The SASR framework

A typical filter $\mathbf{w} \in \mathbb{R}^{MN}$ of the SASR framework can be obtained by minimizing the loss function below:

$$\mathcal{E}(\mathbf{w}) = \frac{1}{2}\left\| \sum_{d=1}^{D} \mathbf{x}^d \star \mathbf{w}^d - \mathbf{y} \right\|^2 + \sum_{d=1}^{D} \left\| \psi \cdot \mathbf{w}^d \right\|^2 + \frac{1}{2}\sum_{k=1}^{K} \left\| \sum_{d=1}^{D} \alpha_k \mathbf{x}_k^d \star \mathbf{w}^d \right\|^2, \tag{1}$$

where $\mathbf{x}^d \in \mathbb{R}^{MN}$ represents the sample vector extracted from the object patch. The superscript d denotes the d-th channel of the D-channel feature space, The star $\star$ stands for spatial correlation between two vectors. $\mathbf{y} \in \mathbb{R}^{MN}$ is the desired response of the object patch, more specifically, a Gaussian-shaped label. A regularization term is introduced in the second place, in which the dot $\cdot$ represents Hadamard product. $\boldsymbol{\psi}$ is the selective spatial regularizer constructed in Section 3.4, which is a constant matrix. In the third term, $\mathbf{x}_k \in \mathbb{R}^{MN} (k = 1, 2 \ldots, K)$ is the sample matrix from the k-th patch among all K context patches, containing some objects in the background. α_k denotes the weight of the k-th context patch. The third term means that the desired response of the surrounding patch $\mathbf{x}_k$ should be zero, which can help repress the response of irrelevant adjacent contextual information. It also serves as a penalization when the response from any surrounding patch $\mathbf{x}_k$ is unusually high.

In order to preserve the integrality of the object, each surrounding patch is selected to have a proper gap to the object patch. Therefore, each $\mathbf{x}_k$, where $k = 1, 2 \ldots, K$, contains little information from the object. A balanced strategy to choose proper patches is presented in Section 3.3.

By introducing an auxiliary variable $\mathbf{g}$, requiring $\mathbf{g} = \mathbf{w}$ and using Augmented Lagrangian Method (ALM) [18], Eq. (1) is equivalent to:

$$\begin{aligned} \mathcal{L}(\mathbf{w}, \mathbf{g}, \zeta) = {} & \frac{1}{2}\left\| \sum_{d=1}^{D} \mathbf{x}^d \star \mathbf{w}^d - \mathbf{y} \right\|^2 \\ & + \sum_{d=1}^{D} \left[\mathcal{D}(\mathbf{w}^d) - \mathcal{D}(\mathbf{g}^d) \right]^{\mathrm{T}} \mathcal{D}(\zeta^d) \\ & + \frac{1}{2}\sum_{d=1}^{D} \left\| \psi \cdot \mathbf{g}^d \right\|^2 + \frac{\gamma}{2}\sum_{d=1}^{D} \left\| \mathbf{w}^d - \mathbf{g}^d \right\|^2 \\ & + \frac{1}{2}\sum_{k=1}^{K} \left\| \sum_{d=1}^{D} \alpha_k \mathbf{x}_k^d \star \mathbf{w}^d \right\|^2, \end{aligned} \tag{2}$$

where $\zeta^d \in \mathbb{R}^{MN}$ denotes the d-th channel of the Lagrangian vector. γ is the penalty factor. The note $\mathcal{D}(\mathbf{w}^d)$ represents the diagonal matrix with each element in vector $\mathbf{w}^d$ along its diagonal. The superscript T stands for the transpose of the corresponding matrix.

By introducing $\mathbf{h} = \frac{1}{\gamma}\zeta$, Eq. (2) can also be reformulated as [28]:

$$\begin{aligned} \mathcal{L}(\mathbf{w}, \mathbf{g}, \mathbf{h}) = {} & \frac{1}{2}\left\| \sum_{d=1}^{D} \mathbf{x}^d \star \mathbf{w}^d - \mathbf{y} \right\|^2 + \frac{1}{2}\sum_{d=1}^{D} \left\| \psi \cdot \mathbf{g}^d \right\|^2 \\ & + \frac{\gamma}{2}\sum_{d=1}^{D} \left\| \mathbf{w}^d - \mathbf{g}^d + \mathbf{h}^d \right\|^2 \\ & + \frac{1}{2}\sum_{k=1}^{K} \left\| \sum_{d=1}^{D} \alpha_k \mathbf{x}_k^d \star \mathbf{w}^d \right\|^2. \end{aligned} \tag{3}$$

Table 2
Expression of variables.

Variable	Expression	Size
$\mathbf{q}$	$\mathbf{q} = \alpha_0 \delta_j(\hat{\mathbf{x}}_0)\hat{y}_j + \gamma \delta_j(\hat{\mathbf{g}}) - \gamma \delta_j(\hat{\mathbf{h}})$	$D \times 1$
U	$U = [\alpha_0 \delta_j(\hat{\mathbf{x}}_0), \ldots, \alpha_K \delta_j(\hat{\mathbf{x}}_K)]$	$D \times (K+1)$
Y	$Y = [\hat{y}_{j0}, \ldots, \hat{y}_{jK}]^{\mathrm{H}}$	$(K+1) \times 1$
$S_{\mathbf{xx}}$	$S_{\mathbf{xx}} = \sum_{k=0}^{K} \alpha_k^2 \delta_j(\hat{\mathbf{x}}_k)^H \delta_j(\hat{\mathbf{x}}_k)$	Scalar
$S_{\mathbf{gx}}$	$S_{\mathbf{gx}} = U^{\mathrm{H}} \delta_j(\hat{\mathbf{g}})$	$(K+1) \times 1$
$S_{\mathbf{hx}}$	$S_{\mathbf{hx}} = U^{\mathrm{H}} \delta_j(\hat{\mathbf{h}})$	$(K+1) \times 1$
b	$b = \gamma + S_{\mathbf{xx}}$	Scalar

The ADMM algorithm allows $\mathcal{L}(\mathbf{w}, \mathbf{g}, \mathbf{h})$ to be solved in several iteration. Before that, decompose $\mathcal{L}(\mathbf{w}, \mathbf{g}, \mathbf{h})$ into two sub-functions:

$$p(\mathbf{w}) = \frac{1}{2}\left\| \sum_{d=1}^{D} \mathbf{x}^d \star \mathbf{w}^d - \mathbf{y} \right\|^2 + \frac{\gamma}{2} \sum_{d=1}^{D} \left\| \mathbf{w}^d - \mathbf{g}^d + \mathbf{h}^d \right\|^2 + \frac{1}{2} \sum_{k=1}^{K} \left\| \sum_{d=1}^{D} \alpha_k \mathbf{x}_k^d \star \mathbf{w}^d \right\|^2, \tag{4a}$$

$$p(\mathbf{g}) = \frac{1}{2} \sum_{d=1}^{D} \left\| \psi \cdot \mathbf{g}^d \right\|^2 + \frac{\gamma}{2} \sum_{d=1}^{D} \left\| \mathbf{w}^d - \mathbf{g}^d + \mathbf{h}^d \right\|^2. \tag{4b}$$

Considering each sub-problem is convex, the optimal solution can be found by setting partial derivative to zero. $p(\mathbf{g})$ can be solved in spatial domain, and the closed form solution is:

$$\mathbf{g} = \gamma(\mathbf{w} + \mathbf{h}) \div (\boldsymbol{\Psi}^2 + \gamma) \tag{5}$$

where $\div$ denotes element-wise division. $\boldsymbol{\Psi}^2$ denotes a $M \times N$ matrix whose elements are the square of the corresponding element in ψ.

Using Parsevals theorem, $p(\mathbf{w})$ can be solved efficiently in Fourier domain. The optimal solution for $\hat{\mathbf{w}}$ (the Fourier transform of $\mathbf{w}$) can be expressed in an element-wise manner:

$$\delta_j(\hat{\mathbf{w}}) = \frac{\mathbf{q}}{\gamma} - \frac{U}{\gamma b} \times (S_{\mathbf{xx}} Y + \gamma S_{\mathbf{gx}} - \gamma S_{\mathbf{hx}}) \tag{6}$$

where $\delta_j(\hat{\mathbf{w}}) \in \mathbb{R}^D$ denotes the vector consisting of the j-th elements of $\hat{\mathbf{w}}$ along all D channels. The other variables are listed in Table 2.

More details for the derivation are presented in Appendix A.

Then ADMM is carried out by alternatively solving $p(\mathbf{w})$ and $p(\mathbf{g})$ and updating $\mathbf{h}$:

$$\mathbf{w}^{(i+1)} = \arg\min_{\mathbf{w}} p(\mathbf{w}, \mathbf{g}^i, \mathbf{h}^i) \tag{7a}$$

$$\mathbf{g}^{(i+1)} = \arg\min_{\mathbf{g}} p(\mathbf{g}, \mathbf{w}^{(i+1)}, \mathbf{h}^i) \tag{7b}$$

$$\mathbf{h}^{(i+1)} = \mathbf{h}^{(i)} + \mathbf{w}^{(i+1)} - \mathbf{g}^{(i+1)}. \tag{7c}$$

The penalty parameter γ is also updated at the end of each iteration:

$$\gamma^{i+1} = min(\gamma^{max}, \beta\gamma^i) \tag{8}$$

where $\gamma^{\max}$ is the maximum value of γ, and i denotes the number of iteration, β is a scale factor.

In Eq. (7), updating $\mathbf{h}$ requires $\mathbf{w}$ and $\mathbf{g}$ in spatial domain. Therefore, the filter from Eq. (6) must be transformed back into spatial domain via inverse Fast Fourier Transform. If $\mathbf{h}$ converges after several iterations, the present $\mathbf{w}$, $\mathbf{g}$ and $\mathbf{h}$ will be the optimal solution to the original problem $\mathcal{L}(\mathbf{w}, \mathbf{g}, \mathbf{h})$.

3.3. Selection of surrounding patches

Cosine window and padding are frequently used in CF-based tracking [12,19]. Concretely, artifacts in shifted samples are mitigated after weighted by a cosine window [13] and padding provides some context information as shown in Fig. 2(a). While the object patch is labeled by a Gaussian function, all surrounding patches are labeled by zero, as negative samples ought to have zero response. Since the surrounding patches must have the same size as the object patch, the position of the center point is the only factor to consider. Intuitively the cyan boxes in Fig. 2(b) show the best positions for surrounding patches since they have the same size as the object and is right next to the object box. However, it turns out those in Fig. 2(c) are more reasonable.

As is illustrated in Fig. 2, a surrounding patch also contains the object. After windowing, the closer the surrounding patch is, the clearer the object will be. When a surrounding patch is labeled with zero, the object is also included, which means it is weakened. In order to alleviate the loss of the object, the surrounding patches cannot be too close to the center. However, the farther the patches locate, the more irrelevant background information they involve. Even if camera motion and viewpoint change occur frequently during UAV tracking, the background is still not as active as the object. On the assumption that objects far away are less likely to enter the searching area of the tracker, setting their responses to zero brings redundancy to the tracker.

To make a trade-off between the contradictions, the center point of each patch is selected like those in Fig. 2(f). The precise locations of the four surrounding patches are given by Algorithm 1:

Algorithm 1: Location selection.

Input: Object position (x, y) and size(a, b)
Base offset rate r
Output: Center locations for surrounding patches

$len = \sqrt{ab}$ % base offset length
$asp\_rate = max(a, b)/min(a, b)$ % aspect ratio of the bounding box
if $a < b$ **then**
  $x\_rate = r$ % offset ratio in x direction
  $y\_rate = r * \sqrt{asp\_rate}$ % offset ratio in y direction
else
  $y\_rate = r$
  $x\_rate = r * \sqrt{asp\_rate}$
end

$$positions = \begin{bmatrix} x + x\_rate * len & y \\ x - x\_rate * len & y \\ x & y + y\_rate * len \\ x & y - y\_rate * len \end{bmatrix}$$

When the aspect ratio of the bounding box, i.e., asp_rate, is relatively high, Algorithm 1 will not place two surrounding patches too far away from the object patch and the other two too close. Meanwhile, the base offset rate r creates a gap to protect the object, as shown in Fig. 3. In this way, a balanced trade-off can be achieved.

Fig. 3 reveals the visual field of the tracker, which has been significantly expanded by the surrounding patches. Without them, the tracker is limited to the small patch in Fig. 2(d). The positions of surrounding patches actually affect the shape and range of the visual field. A proper offset rate r will yield an appropriate visual field for the whole tracking process. Those inside the blue box can help repress background distraction, while those out of it is more likely to bring irrelevant information into the filter. The blurred

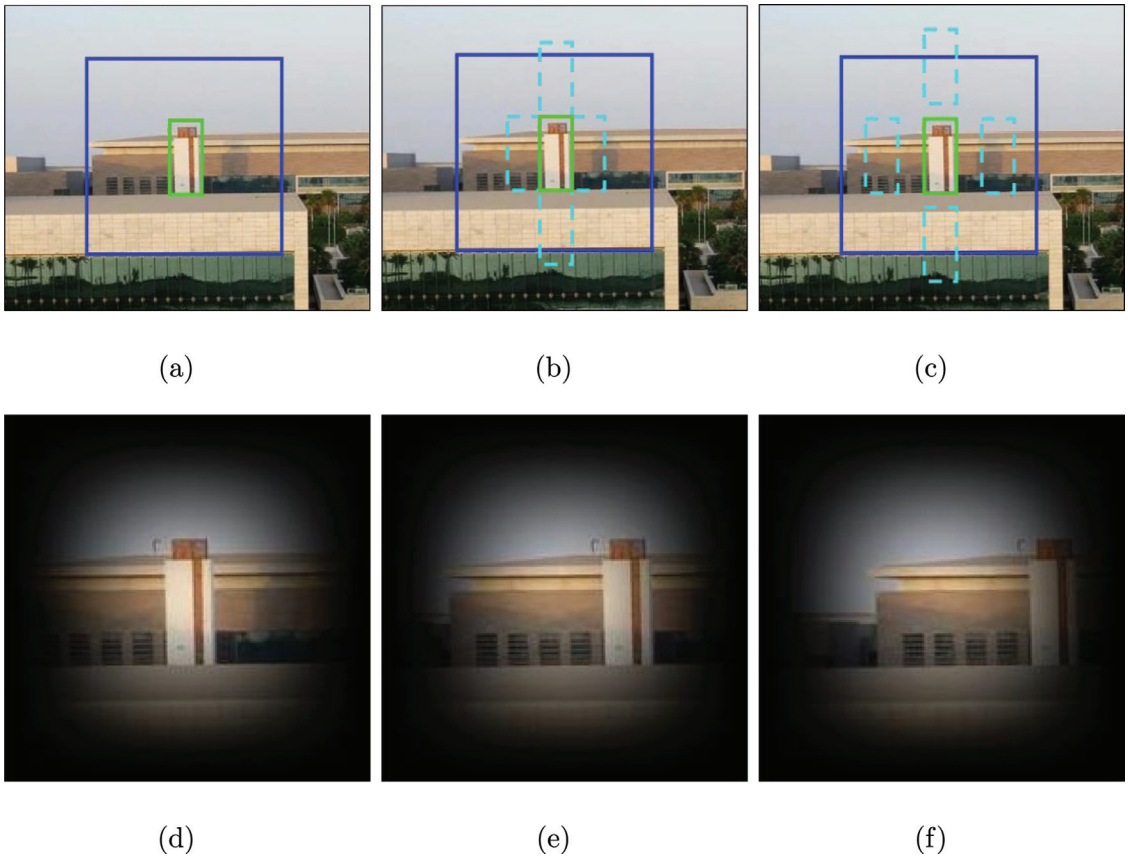


Fig. 2. Comparison of different strategies to select surrounding patches. The blue box denotes the image patch after padding, which is also the searching area of the tracker. The green box shows the groundtruth in this frame The cyan boxes stands for surrounding patches. Fig. 2(c) gives proper choices for positions of the surrounding patches. Fig. 2(d) is the object patch after weighted by a cosine window. Fig. 2(e) and (f) are the patches corresponding to the left cyan box in Fig. 2(b) and (c) respectively. Since a surrounding patch is labeled with zero, the object is set to zero as well, which means it is weakened. Image selected from *building1* in UAV123 [42]. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

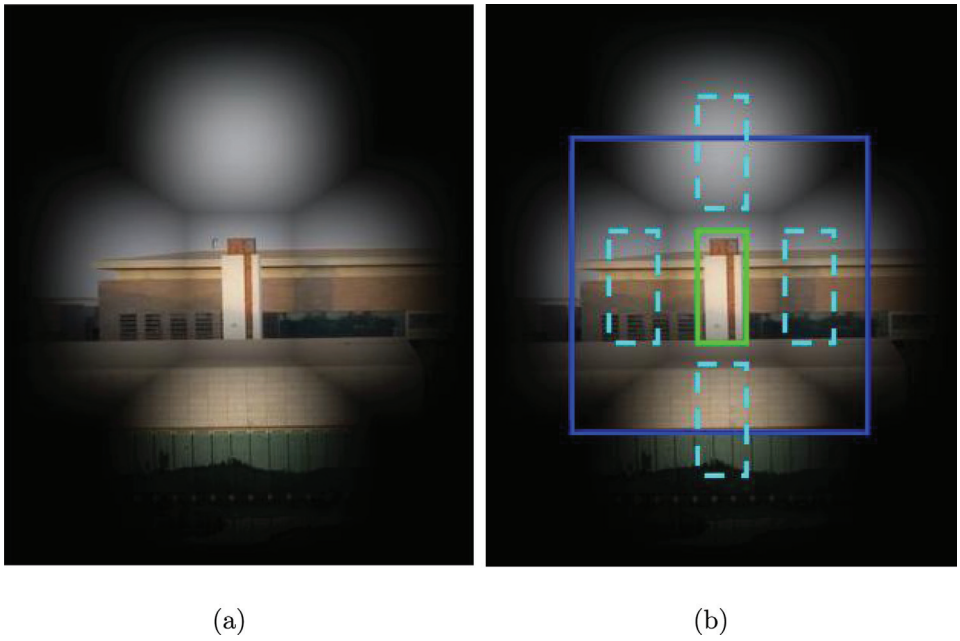


Fig. 3. Visual field of the tracker. By matching the object in each patch, the object patch along with all surrounding patches are synthesized into Fig. 3(a). Fig. 3(b) are labeled with the boxes in Fig. 2(c). At the intersection of every two patches, the larger pixel values are preserved. Exaggerated for better view.

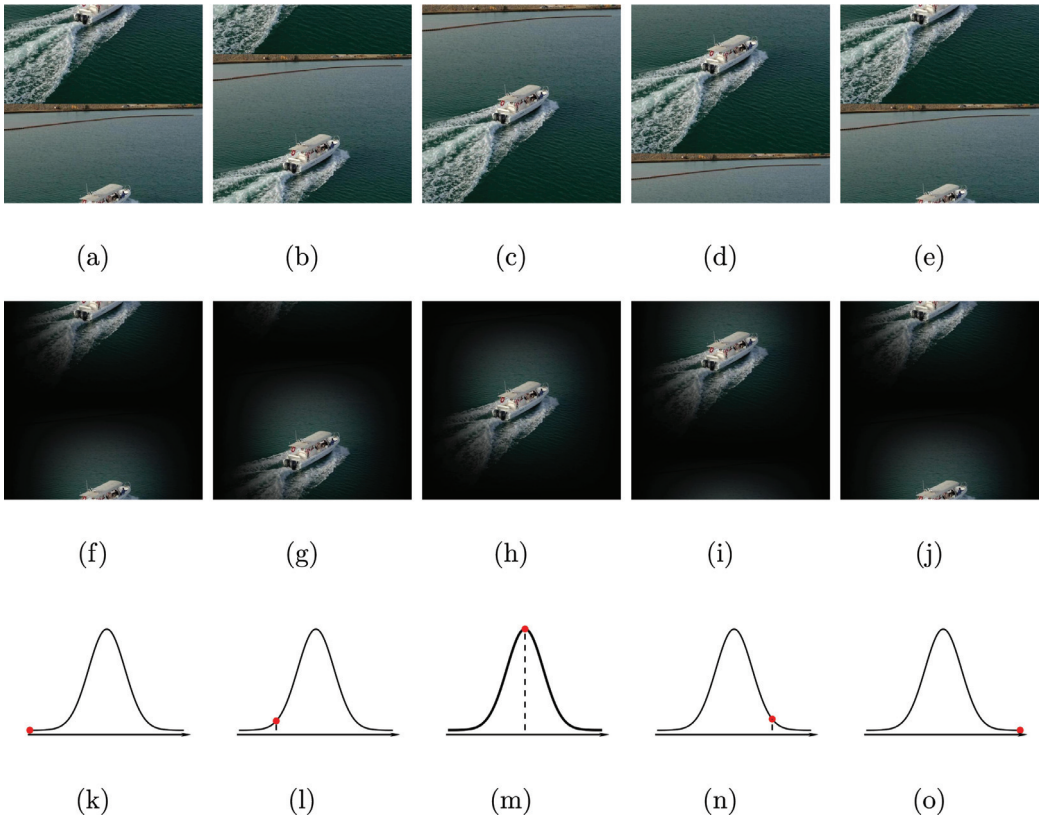


Fig. 4. Boundary effect brought by circulant shift. The first row shows a base sample (c) and four generated ones through circulant shift. Fig. (a) and (e) are shifted by half the patch size, while Fig. (b) and (d) a quarter the patch size. The second row shows the patches generated by a sample patch that is weighted by a cosine window. The third row shows their corresponding value in the label function. Only one-dimensional circulant shift is showed here for convenience. Patch selected from *boat5* in UAV123 [42].

edges around the object are the boundaries dividing it from the surrounding patches. For the reason that the surrounding samples are labeled zero, the closer the edges are, the more severely the object is weakened. Positions given by Algorithm 1 can make a balance between the conflicts.

3.4. Selective spatial regularizer setup

As is stated in Section 2.2, boundary effect takes place mainly near the edge of the image patches. Fig. 4(c) shows a base sample without windowing and the rest four in the first row are shifted ones generated with it.

Obviously, if not weighted by a cosine window, most shifted samples are corrupted. After windowing, the central part is preserved but the border of each patch fades out (Fig. 4(h)). Positive samples generated by the windowed patch are relatively acceptable (Fig. 4(g) and (i)) since they are merely slightly shifted. On the contrary, negative samples used in the training step are still severely corrupted (Fig. 4(f) and (j)). In those acceptable samples, the object lies near the center, while the rest background fades out as a result of the cosine window. In those corrupted samples, the object lies along the border. Under either condition, the boundary area of the patches contains little valuable information. Therefore, a reasonable way to reduce the redundancy in the filter is to prevent it from fitting the boundary and emphasize the central area.

With that aim, SASR has incorporated a selective spatial regularizer $\boldsymbol{\psi}$, whose formula is given as:

$$\psi(x,y) = \begin{cases} \psi(\frac{a}{2}, \frac{b}{2}), & if(x,y) \in \Omega \\ \psi_0 + \eta(\frac{x^2}{a^2} + \frac{y^2}{b^2}), & \text{others} \end{cases} \tag{9}$$

where a and b are the width and height of the object respectively. $\boldsymbol{\psi}_0$ is the minimum value of the regularizer. The shape of $\boldsymbol{\psi}$ and its correspondence relationship with the image patch is illustrated in Fig. 5. Specially, Ω is the area within the circum-ellipse of the bounding box. Notably, Ω acts as a protective area, preventing the object from being penalized. Therefore, the tracker will not lose valuable information inside the bounding box. Without it, the object would be penalized along with the background and the tracker would not be able to accurately estimate scale variation as shown in Section 4.5.1.

3.5. Model update

In order to improve the robustness to all those challenging scenarios, correlation filter-based trackers typically employ an online learning strategy. SASR updates its object feature model using a linear interpolation:

$$\mathbf{x}^t_{model} = (1-\mu)\mathbf{x}^{t-1}_{model} + \mu\mathbf{x}^t \tag{10}$$

where μ is the learning rate, $\mathbf{x}^t$ is the object feature extracted at frame t and $\mathbf{x}^t_{model}$ is the model feature. In practice, $\mathbf{x}^t_{model}$ is actually used in Eq. (A.14) to train the filter $\mathbf{w}$.

Considering $\mathbf{w}$ is already trained with an integrated feature model $\mathbf{x}^t_{model}$, it is not updated through linear interpolation. A filter $\mathbf{w}$ trained at a new frame directly will directly replace the old one. An extra linear interpolation will make the filter less adaptive to fast appearance variation.

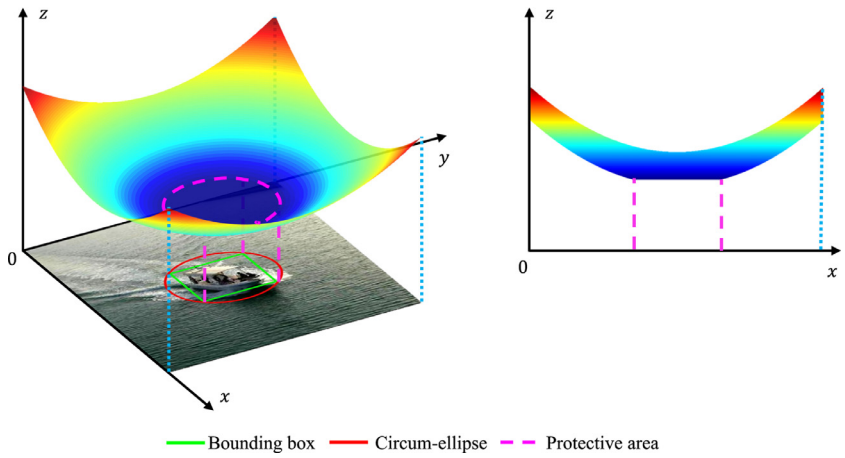


Fig. 5. Correspondence between the regularizer and an input image patch. The regularizer can be described as an elliptic-paraboloid with a flattened bottom, which protects the object from being penalized. The fatten bottom of the regularizer can be clearly viewed from the picture on the right. Pixels within the Red ellipse in the image patch corresponds to the protective area in the regularizer (circled by the magenta curve).

The tracking pipeline of SASR tracker is summarized in Algorithm 2:

Algorithm 2: SASR tracker.

Input: A video sequence consisting of T frames
Position ($\mathbf{p}$) and size ($\mathbf{s}$) of the object in the first frame
Output: Position and size of the object in all frames

Construct label function $\mathbf{y}$ and regularizer ψ
Read the first frame
Calculate *positions* for surrounding patches using Algorithm 1
Crop patches with $\mathbf{p}$, $\mathbf{s}$ and *positions*
Extract features of the patches
Train filter $\mathbf{w}$ by minimizing Eq. (1)
current frame $\leftarrow$ next frame
while *not end of the sequence* **do**
  Read current frame
  Crop a set of object patches according to $\mathbf{p}$ and $\mathbf{s}$
  Extract feature set $\mathbf{X}$
  Detecting by *response* = $\mathbf{X} \star \mathbf{w}$
  Find new object position $\mathbf{p}_{new}$ and $\mathbf{s}_{new}$
  $\mathbf{p} \leftarrow \mathbf{p}_{new}$, $\mathbf{s} \leftarrow \mathbf{s}_{new}$
  Calculate *positions* for surrounding patches using Algorithm 1
  Crop patches with $\mathbf{p}$, $\mathbf{s}$ and *positions*
  Extract features of the patches
  Update feature model using Eq. (10)
  Retrain filter $\mathbf{w}$ by minimizing Eq. (1)
  Record current position and scale
  current frame $\leftarrow$ next frame
end
return position and size of the object in all frames

4. Experiments

To evaluate the accuracy and robustness of SASR, this section presents comprehensive experiments on challenging sequences from UAV123 [42] and UAVDT [43].

The sequence set evaluates SASR along with 23 modern state-of-the-art trackers from four classes: (i) CF-based trackers with deep features, including ECO [36], C-COT [35], MCCT [34], DeepSTRCF [28], IBCCF [33] and CF2 [32]. (ii) CF-based trackers with surrounding information or spatial regularization, including CACF [1] [11], BACF [29], SRDCF [15], SRDCFdecon [26], CSR-DCF [44]. (iii) end-to-end deep learning methods, including SiamFC [40], UDT [45], UDT+ [45], ADNet [37] and CFNet [38]. (iv) other representative algorithms, including KCC [46], MEEM [47], SAMF [23], DSST [48], KCF [13] and PTAV [49].

Codes used in all the following experiments are publicly available. For a fair comparison, all default parameters are left unchanged in any of the tested algorithms.

4.1. Evaluation criteria

Following the regulation proposed by Wu et al. [50], the benchmark evaluates the performance of a tracker from distance precision rate (DPR) and overlap success rate (OSR).

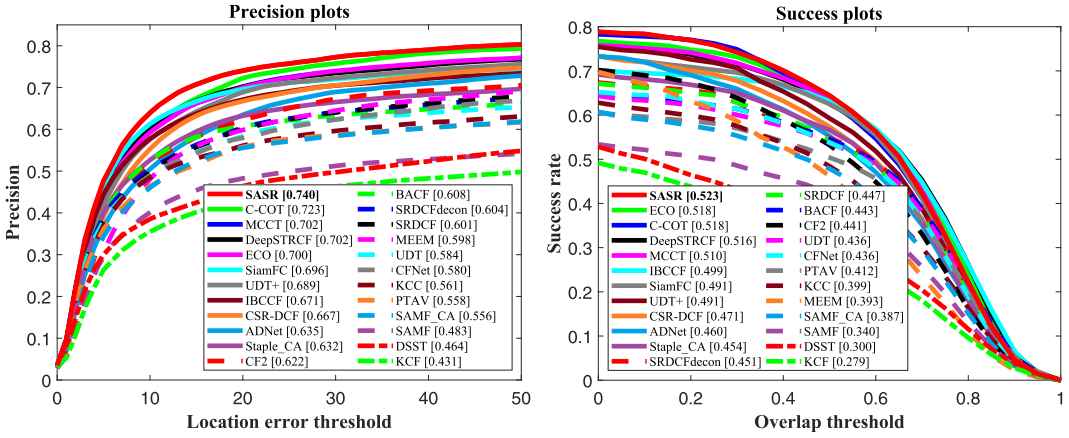
Center location error (CLE) denotes the Euclidian distance between the center points of bounding-boxes from tracking results and manually annotated groundtruth. DPR for a single sequence shows the proportion of frames whose CLE are within an acceptable threshold. Typically CLE at 20 pixels is used to rank the trackers in DPR diagram. Results are also assessed using a fraction "intersection over union (IoU)" whose numerator is the area of intersection between a result box and its corresponding groundtruth and whose denominator is the union of the two boxes. OSR for a single sequence demonstrates the ratio of frames whose IoU is greater than or equal to a certain value that varies from 0 to 1. Commonly, trackers are ranked according to mean OSR which is also known as area-under-the-curve (AUC).

The experiments keep the convention of one-pass evaluation.

4.2. Implementation details

The square sample patch centering at the object has a side length of $\sqrt{5ab}$ (a, b are the width and height of the object respectively). Base offset rate r in Algorithm 1 is set to 1.08. $\boldsymbol{\psi}_0$ and η in Eq. (9) are chosen as 10^{-3} and 10^7 respectively. Although α_k in Eq. (1) do not necessarily need to be the same, in general UAV tracking, however, we can not assume which one is more effective. Therefore, all α_k are set to 0.05. As shown in Fig. 1, fHOG, CN and deep features are employed. Following the settings in [28], the

[1] It is noted that CACF consists of four trackers, i.e., MOSSE_CA, DCF_CA, SAMF_CA and Staple_CA, in which SAMF_CA and Staple_CA are more competitive, as shown in the paper of CACF [11]. Therefore, we have incorporated SAMF_CA and Staple_CA in our experiments for comparisons.



(a) Distance Precision Rate (DPR) (b) Overlap Success Rate (OSR)

Fig. 6. Tracking results on challenging UAV tracking sequences from UAV123 [42]. Comparison with 23 state-of-the-art trackers has shown that SASR demonstrates first-class performance. SASR has a promotion at 1.7% in terms of DPR against the second best C-COT (ECCV2016), and 0.5% in OSR against the second best ECO (CVPR2017).

initial penalty parameter γ^0 in ADMM iteration, maximum value $\gamma^{\max}$ and scale factor β are set to 10, 100 and 1.2 respectively.

All codes are tested with Matlab R2017b and experiments are carried out on a standard desktop computer with Intel Core i7-8700K (3.7 GHz), 32GB RAM and NVIDIA Quadro P2000 GPU.

A Matlab implementation of SASR is available at https://github.com/vision4robotics/SASR-tracker and a video demonstration consisting of several representative sequences is available at https://youtu.be/SuJ7rcCU4kk.

4.3. Overall, attribute-based and qualitative evaluations on UAV123

A comprehensive evaluation is carried out on 100 challenging UAV tracking sequences from UAV123. They are labeled with 12 attributes, including aspect ratio change (ARC), background clutter (BC), camera motion (CM), fast motion (FM), full occlusion (FOC), illumination variation (IV), low resolution (LR), out-of-view (OV), partial occlusion (POC), scale variation (SV), similar object (SOB) and viewpoint change (VC).

4.3.1. Overall evaluation

DPR and OSR plots in Fig. 6 demonstrate the average performance over all sequences.

By ranking the first place in both DPR and OSR, SASR demonstrates a competing capability for tracking an arbitrary object. With both representative features and powerful machine learning algorithm, 5 trackers, including SASR, ECO [36], C-COT [35], MCCT [34] and DeepSTRCF [28], rank top in both diagrams. Deep learning methods, including UDT+ [45], SiamFC [40] and ADNet [37], have also achieved preferable results. With effective contextual data, SAMF_CA [11] has obtained significant improvement over its baseline SAMF [23]. BACF [29], which exploits background information, and SRDCF [15], which contains spatial regularization, have shown similar performance. With the support of surrounding information, Staple_CA [11], with only hand-crafted features, even outperformed the deep tracker CF2 [32]. With integrated powerful features, effective surrounding information and selective spatial regularization, SASR shows competitive performance among all trackers.

4.3.2. Attribute-based evaluation

The testing sequence set is labeled with 12 attributes, as mentioned in Section 4, and similar evaluation is also carried out for each attribute. Figs. 7 and 8 presents the performance for each attribute respectively.

For DPR, SASR achieves 8 best results out of 12 attributes including ARC (0.654), FOC (0.475), IV (0.674), LR (0.582), POC (0.667), SV (0.698), SOB (0.771) and VC (0.673). UDT+ [45] demonstrates the best ability to handle BC, outperforming SASR by 0.1%. In sequences with CM and FM, C-COT [35] performs best among all, probably owing to its powerful continuous convolution, but the performance of SASR is very close to C-COT [35] in terms of CM. When the object is out of view, DeepSTRCF [28] shows the best tracking capability, perhaps because of its temporal regularization. Meanwhile, surrounding information and selective spatial regularization help SASR perform almost equally to DeepSTRCF [28].

For OSR, SASR ranks top 3 in 10 out of 12 attributes including ARC (0.437), CM(0.515), FOC (0.242), IV (0.447), LR (0.328), OV (0.416), POC (0.441), SV (0.485), SOB (0.533) and VC (0.469). By mapping the discrete data into the continuous domain, C-COT [35] shows the best performance in FM and LR. With multiple experts, MCCT [34] is the best for BC and IV, and SASR shows almost equivalent ability to address IV. ECO [36] demonstrates the best capability to adapt to viewpoint change, and SASR performs closely to it.

4.3.3. Qualitative evaluation

To further evaluate the capability of SASR, rich experiments are performed in comparison with other trackers. Fig. 9 shows some of the representative sequences, each contains several challenging scenarios.

In sequence *person7_1*, the object moves fast and undergoes violent deformation. When the object moves quickly (e.g., frame #028 and #075), UDT [45], Staple_CA [11], PTAV [49] and CF2 [32] drift away because of fast motion. MCCT [34] is able to handle that, but failed when the object has gone through deformation (e.g., frame #123 and #230). SASR, ECO [36], C-COT [35], SiamFC [40] and DeepSTRCF [28] managed to track the object under those scenarios.

In *group3_3*, PTAV [49] and SRDCFdecon [26] lose the object when it deforms (e.g., frame #088). MCCT [34] is distracted by a similar object in subsequent frames (e.g., frame #494).

In *wakeboard4*, when the viewpoint changes, UDT+ [45] and CF2 [32] gradually become distracted by the background (e.g., frame #125). When the object moves fast and the background is cluttered, SiamFC [40] and MCCT [34] fail to catch up with it (e.g., frame #141 and #190).

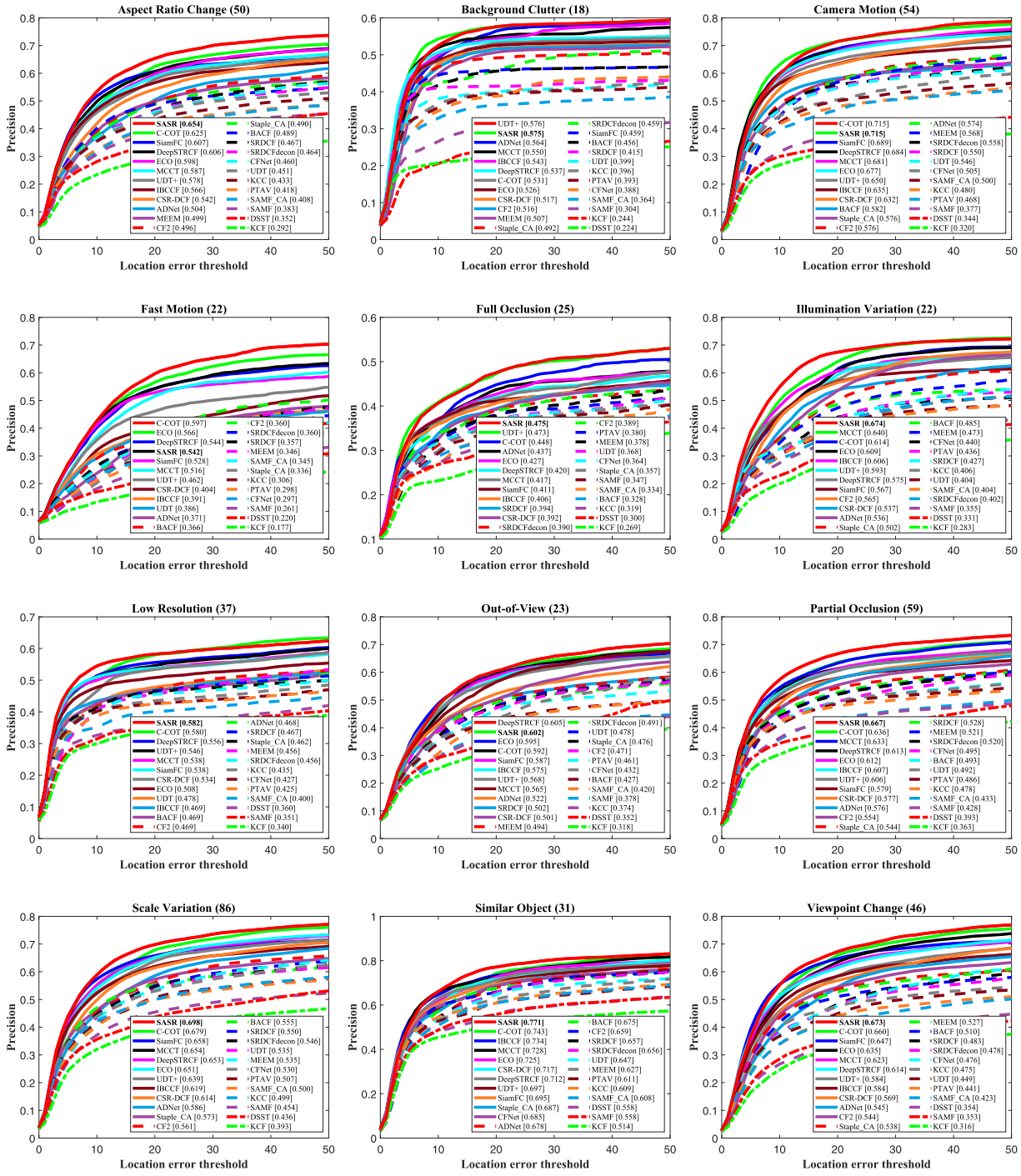


Fig. 7. Precision plots on different attributes.

Similarly in *wakeboard5*, trackers without powerful features, i.e., SRDCFdecon [26], DSST [48] and PTAV [49], drift away when the background is cluttered (e.g., frame #031). When the object undergoes both scale variation and deformation (e.g., frame #336), SRDCF [15] and DeepSTRCF [28], with out selective regularization, fail to accurately estimate the scale of the object, although they do not lose it completely. When the object suffers from occlusion(e.g., frame #552), UDT+ [45], MCCT [34] and C-COT [35] are misled by a similar object. SASR and ECO [36] complete the whole task successfully.

In *person16*, CFNet [38], BACF [29], DeepSTRCF [28], ECO [36] and C-COT [35] miss the object when it go through full occlusion(e.g., frame #082). With a powerful verifier, PTAV [49] is able to re-identify the object even after long occlusion. Owing to surrounding information, SASR can repress the littery response evoked by obstructions. Therefore, it can complete the tracking task like other state-of-the-art trackers.

Likewise in *car7*, because of severe occlusion (e.g., frame #094 and #161), CFNet [38], CF2 [32], DeepSTRCF [28], MCCT [34] and C-COT [35] are distracted by a similar object in vicinity. After the first occlusion, only 5 trackers manage to estimate the trajectory of the object, including SASR, ECO [36], UDT+ [45], SRDCFdecon [26], and SRDCF [15]. However, SRDCF [15] fail to catch up with the object after the second occlusion (e.g., frame #317).

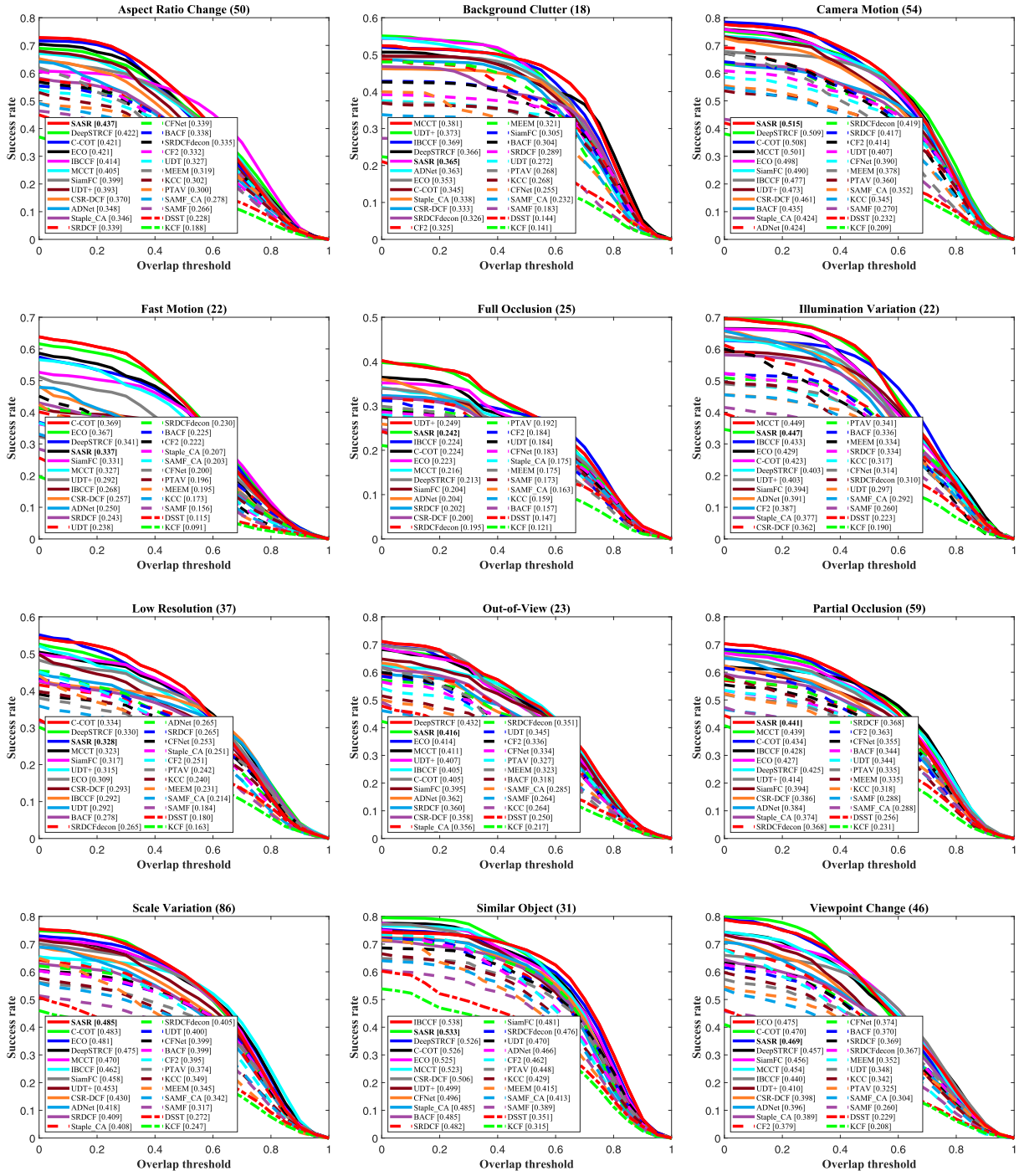


Fig. 8. Overlap success rate on different attributes.

4.4. Speed comparison on UAV123

Fig. 10 compares the speed, i.e., frames per second (FPS), of 5 top-ranking trackers in Section 4.3, i.e., SASR, ECO [36], C-COT [35], MCCT [34] and DeepSTRCF [28]. They have seized the top 5 places in both DPR and OSR graphs.

Among those algorithms, SASR generally runs faster and is more accurate than C-COT [35] and MCCT [34]. Meanwhile, the computational cost brought by the upgrade in accuracy is still acceptable.

4.5. Performance with different modules and offset rates on UAV123

This section presents more detailed experiments in order to illustrate the effect of the moduels in SASR framework. Section 4.5.1 investigates the tracking results with 3 different modules in the SASR framework, including surrounding-aware module (SA), spatial regularization (SR) and protective area (PA). Then, Section 4.5.2 further demonstrates the effect of the base offset rate r in SA module.

Fig. 9. The performance snapshot of SASR and 23 modern state-of-the-art trackers on 6 challenging sequences with various attributes. The sequences from top to bottom are *person7_1, group3_3, wakeboard4, wakeboard5, person16* and *car7* [42].

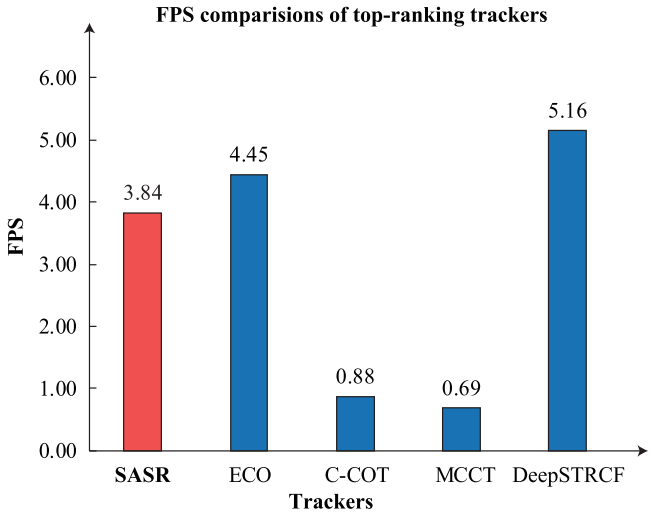


Fig. 10. FPS comparison of top-ranking trackers. SASR achieves favorable upgrade in accuracy with acceptable computational cost.

Table 3
Performance of SASR with different modules.

Tracker	SA	SR	PA	DPR	OSR
SASR_basic	×	×	×	0.618	0.449
SASR_SA	✓	×	×	0.624	0.453
SASR_SA+SR	✓	✓	×	0.686	0.451
SASR_full	✓	✓	✓	0.740	0.523

4.5.1. Performance with different modules

Table 3 presents the overall performance of SASR with different modules. The ✓ for SASR_SA in column SA denotes that module SA is activated and the two × in column SR and PA denotes that those two modules are deactivated. If SA is disabled, all α_k in Eq. (1) are set to zero. When SR is turned down, the regularizer $\boldsymbol{\psi}$ in Eq. (1) is set to a constant number $\boldsymbol{\psi}_0$. Deactivating module PA leads to a consequence that the ellipse in Fig. 5 decays into a point.

SASR_basic, where none of the three modules is activated, has just moderate performance. Based on SASR_basic, module SA improves the tracking ability both in DPR and OSR. Based on

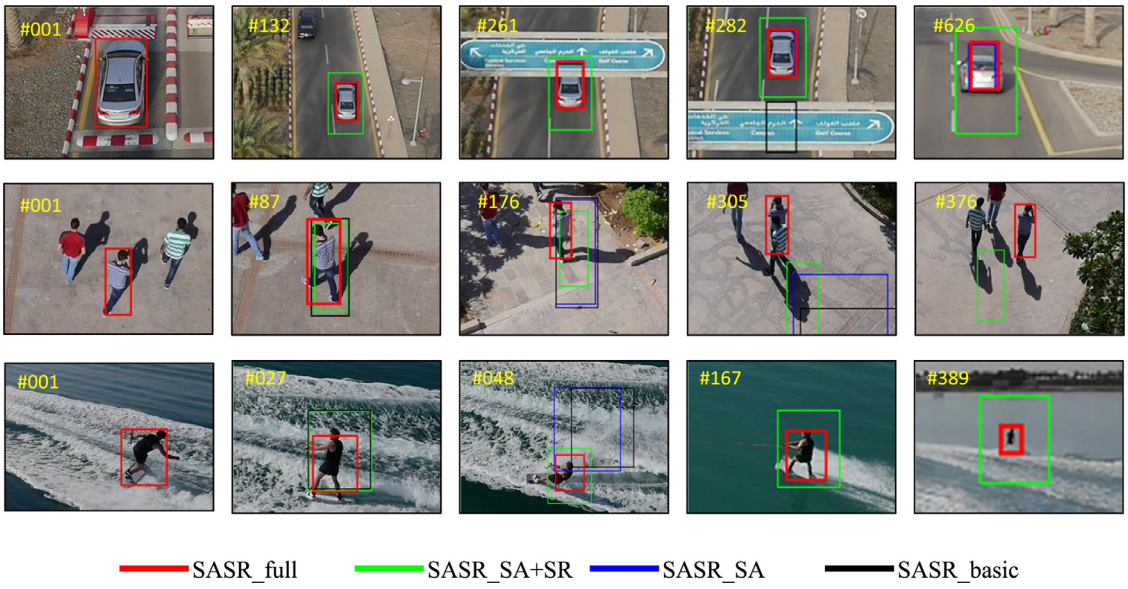


Fig. 11. The effect of different modules in sequence *car9*, *group1_1* and *wakeboard5* from UAV123 [42].

SASR_SA, module SR has brought significant improvement for SASR in terms of DPR accuracy. With the protective effect from module PA, the complete SASR tracker, noted as SASR_full, has further obtained a favorable boost in performance.

Fig. 11 shows three representative sequences and compares the performance of the trackers in Table 3.

Surrounding information helps repress those irrelevant objects in the background. In *car9*, when the object is blocked out, SASR_basic is distracted by the obstruction, while the other three can correctly follow the object with the help of module SA (e.g., frame #261 and #282).

In sequence *group1_1*, there are several two similar objects around the object to track. When the viewpoint changes, SASR_basic and SASR_SA gradually drift away (e.g., frame #87 and #176). Later, SASR_SA+SR is distracted by the shadow of another object because it does not preserve all information from the correct object. While SASR_full manage to recognize the distractor, repress its interference and keep on the object accurately throughout the whole process.

Selective spatial regularization helps alleviate boundary effect and preserve all valuable information from the object. SASR_basic and SASR_SA work totally without the regularizer ψ. In *wakeboard5*, they suffer from boundary effect and are gradually distracted by the background (e.g., frame #027 and #048). Without the protective area Ω, SASR_SA+SR focus only on a small region in the very central area while the other part of the object is also heavily penalized by the regularizer. Therefore, it can not adapt flexibly to scale variation.

4.5.2. Performance with different offset rates

The base offset rate r in Algorithm 1 controls the shape of the visual field. Proper r makes the surrounding patches cover possible distractions in order to repress their response. While larger r imparts redundant surrounding information to the filter, whose capacity is limited to its own size, smaller r does harm to the integrality of the object, as discussed in Section 3.3. Our method offers a location selection strategy of surrounding patches. Generally, this strategy is suitable for algorithms with extended patches including CACF trackers [11]. Based on the framework of SAMF_CA in CACF trackers, we have sequentially developed the surrounding-aware location selection strategy (SA), the spatial regularizer (SR) and the protective area (PA). Therefore, SAMF_CA is selected as a baseline framework to solely illustrate the effect of the SA module and avoid the influences from the other modules. Fig. 12 presents presents the tracking results of SAMF_CA baseline framework with different offset rate r.

With a proper offset rate r, our method constructs a better visual field for SAMF_CA, and more appropriate information can be imported into the training process of the filter. In turn, the performance of the tracker can be promoted, as shown in Fig. 12.

4.6. Additional experiments on UAVDT

This section further evaluates the top 5 trackers in previous section using the recently proposed UAVDT [43]. The dataset consists of 50 fully annotated UAV tracking sequences and focuses on complex scenarios with new level challenges. Its tracking sequences are labeled with 9 attributes, including background clutter (BC), camera motion (CM), illumination variations (IV), large occlusion (LO), long-term tracking (LT), object blur (OB), object motion (OM), scale variations (SV) and small object (SO).

Fig. 13 illustrates the tracking results of those competitive algorithms.

Generally, the experiments on the challenging dataset UAVDT [43] have further proved the power of those state-of-the-art trackers. By ranking first in DPR, SASR still demonstrates first-class tracking ability. Although SASR is second to ECO [36] in OSR, the difference is rather small, at only 0.3%. The performance of MCCT [34] and that of DeepSTRCF [28] are quite close. Meanwhile, C-COT [35] does not performed as satisfactorily as it has on UAV123 [42].

The module SR alleviates boundary effect, therefore SASR can use a larger searching area like SRDCF [15]. As a result, SASR demonstrates good ability to address object motion. Since the module SA can repress background noises and the module PA protects the object, SASR can flexibly adapt to scale variation. Table 4 compares the performance of those trackers for *object motion* and *scale variation* attributes, and the results have shown the merits of SASR.

4.7. Limitations and future work

Although SASR outperformed other 23 state-of-the-art trackers in general evaluation, the performance under some specific

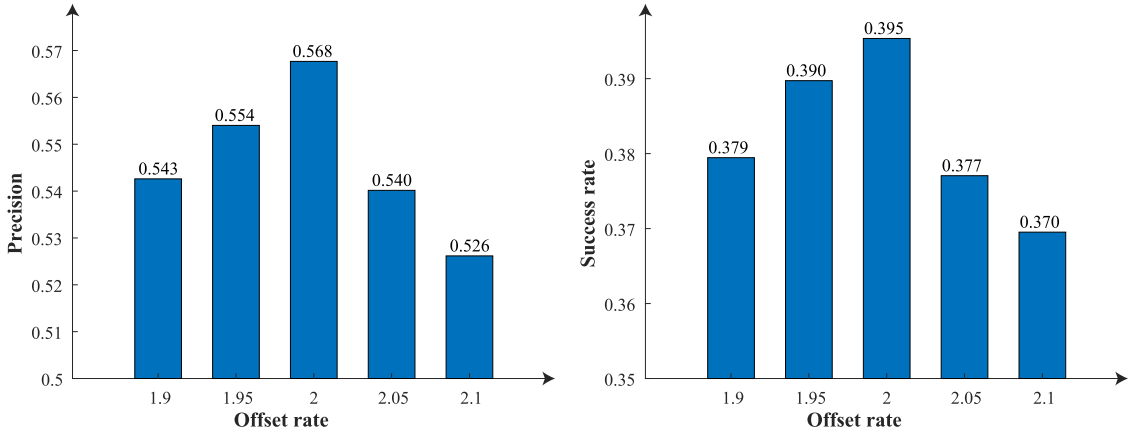
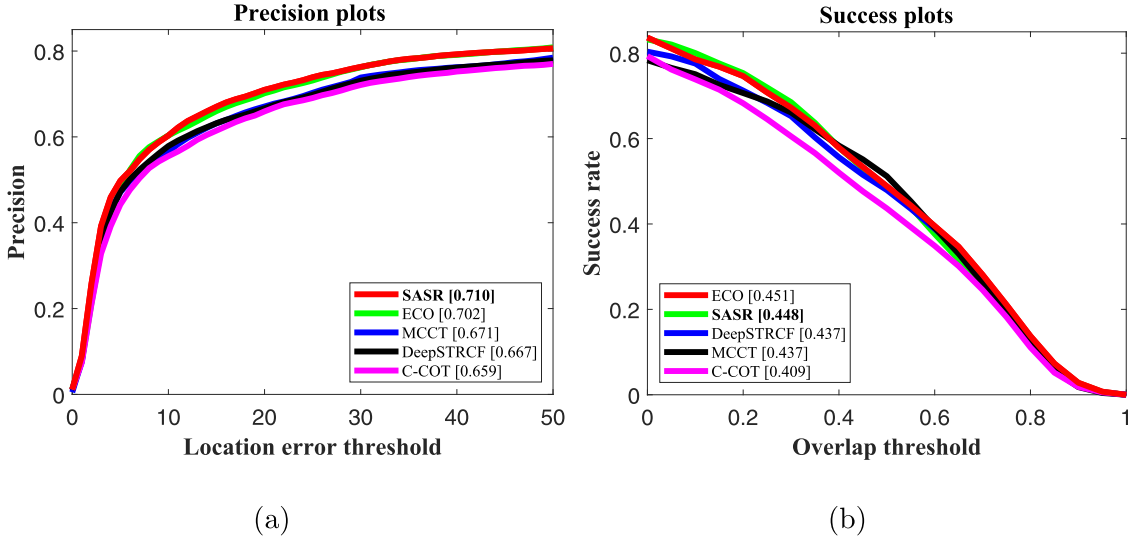


Fig. 12. Performance Performance with different offset rate *r*.



(a) (b)

Fig. 13. Tracking results on UAVDT.

Table 4
Tracking results for *object motion* and *scale variation* attributes.

Attribute	Type	SASR	ECO	C-COT	MCCT	DeepSTRCF
Object motion	DPR	**0.639**	0.627	0.561	0.579	0.571
	OSR	**0.395**	0.395	0.341	0.380	0.374
Scale variation	DPR	**0.655**	0.632	0.559	0.598	0.580
	OSR	**0.436**	0.431	0.379	0.418	0.414

scenario is still not satisfactory. As shown in Section 4.3.2, when the object moves extraordinarily fast, SASR is prone to be distracted. The object is likely to move partly out of the circum-ellipse in Fig. 5 if it moves at a high speed. As a result, that part will be penalized by the regularizer and some information of the object may be overlooked. Similarly, when the object is totally out of view, the tracker learns irrelevant information because of linear interpolation used in the model update step.

The framework of SASR still can be improved and its efficiency can get further promoted after code optimization. The quality of deep features is essential for the performance of a deep tracker. Recent advances in deep learning have made it possible to use a smaller network to provide equivalent performance [51]. With such a network, the efficiency of SASR can be greatly improved. In a fast-changing environment, the surroundings of an object may vary from time to time. As a consequence, the best positions of surrounding patches may not always stay static. Deeper insight into their effect may lead to a better adaptive strategy to choose the patches. We expect SASR to inspire more works on robust and accurate UAV tracking approaches.

5. Conclusions

In this work, a novel unified framework for UAV tracking, i.e., SASR, is presented. By incorporating both effective surrounding information and selective spatial regularization, SASR means to exploit valuable information in both foreground and background while repressing irrelevant interference. Surrounding information is sampled to enlarge the visual field and avoid weakening the object, while the selective spatial regularization preserves all valuable information from the object. Qualitative and quantitative experiments have shown that the novel tracker demonstrates high capability in UAV tracking. With both diverse features (fHOG, CN and deep features) and powerful machine learning algorithm, SASR performs favorably against 23 modern state-of-the-art trackers in terms of both accuracy and robustness. The unified approach, i.e., SASR, can help improve the performance of UAV tracking and extend its application.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The work was supported by the National Natural Science Fundation of China (no. 61806148) and the Fundamental Research Funds for the Central Universities (no.22120180009).

Appendix A

A1. Details for optimizing the two sub-functions

Since L2-norm is element-wise computation, on the assumption that those feature channels are independent, $p(\mathbf{g})$ can be decomposed into D separate problems. Hadamard product can be expressed equivalently by the product of two diagonal matrix. For each feature channel in sub-function $p(\mathbf{g})$:

$$\frac{\partial p(\mathbf{g}^d)}{\partial \mathbf{g}^d} = \mathcal{D}(\psi)^{\mathrm{T}}\mathcal{D}(\psi)\mathcal{D}(\mathbf{g}^d) - \gamma\left[\mathcal{D}(\mathbf{w}^d) - \mathcal{D}(\mathbf{g}^d) + \mathcal{D}(\mathbf{h}^d)\right]. \tag{A.1}$$

Since all channels share the same regularizer $\boldsymbol{\psi}$, information from all channels can be combined together.

$$\frac{\partial p(\mathbf{g})}{\partial \mathbf{g}} = \boldsymbol{\Psi}^2 \cdot \mathbf{g} - \gamma(\mathbf{w} - \mathbf{g} + \mathbf{h}) = 0, \tag{A.2}$$

where $\boldsymbol{\Psi}^2$ is a matrix of size $M \times N$, same as $\boldsymbol{\psi}$, reshaped back from $\mathcal{D}(\psi)^T\mathcal{D}(\psi)$. The dot $\cdot$ refers to Hadamard production. Therefore, the optimal solution for sub-function $p(\mathbf{g})$ is:

$$\mathbf{g} = \gamma(\mathbf{w} + \mathbf{h}) \div (\boldsymbol{\Psi}^2 + \gamma), \tag{A.3}$$

where $\div$ denotes element-wise division.

According to the convolution property of Fourier transform, the DFT of the spatial correlation in Eq. (4a) can be expressed equivalently by dot product between the Fourier transform of the corresponding vectors. By applying Parsevals theorem, sub-function $p(\mathbf{w})$ can be transformed into Fourier domain as:

$$p(\hat{\mathbf{w}}) = \frac{1}{2}\left\|\sum_{d=1}^{D}\hat{\mathbf{x}}^d \cdot \hat{\mathbf{w}}^d - \hat{\mathbf{y}}\right\|^2 + \frac{\gamma}{2}\|\hat{\mathbf{w}} - \hat{\mathbf{g}} + \hat{\mathbf{h}}\|^2 + \frac{1}{2}\sum_{k=1}^{K}\left\|\sum_{d=1}^{D}\alpha_k\hat{\mathbf{x}}_k^d \cdot \hat{\mathbf{w}}^d\right\|^2, \tag{A.4}$$

where the hat $\hat{}$ over a vector represents the Fourier transform of it. $p(\hat{\mathbf{w}})$ has been multiplied by MN, but the notation is kept simple. More detailed derivation is given in A.2.

Consider solving sub-problem $p(\hat{\mathbf{w}})$ element by element:

$$p(\delta_j(\hat{\mathbf{w}})) = \frac{1}{2}\left\|\delta_j(\hat{\mathbf{x}})^{\mathrm{H}}\delta_j(\hat{\mathbf{w}}) - \hat{y}_j\right\|^2 + \frac{\gamma}{2}\left\|\delta_j(\hat{\mathbf{w}}) - \delta_j(\hat{\mathbf{g}}) + \delta_j(\hat{\mathbf{h}})\right\|^2 + \frac{1}{2}\sum_{k=1}^{K}\left\|\alpha_k\delta_j(\hat{\mathbf{x}}_k)^{\mathrm{H}}\delta_j(\hat{\mathbf{w}})\right\|^2, \tag{A.5}$$

where $\delta_j(\mathbf{w}) \in \mathbb{R}^D$ denotes the vector consisting of the j-th elements of $\mathbf{w}$ along all D channels. The superscript H denotes Hermitian transpose of a complex vector or matrix. Since a surrounding patch behaves exactly like the object patch, treating them the same way will be a rational choice. Similarly by setting the gradient to zero, the optimal solution is given by:

$$\delta_j(\hat{\mathbf{w}}) = \left(\sum_{k=0}^{K}\alpha_k^2\delta_j(\hat{\mathbf{x}}_k)\delta_j(\hat{\mathbf{x}}_k)^{\mathrm{H}} + \gamma I\right)^{-1}\mathbf{q}, \tag{A.6}$$

where $\mathbf{x}_0$ refers to sample vector from the object patch, $\alpha_0 = 1$, the other α_k denotes the weight of the k-th surrounding patch and $\mathbf{q} = \alpha_0\delta_j(\hat{\mathbf{x}}_0)\hat{y}_j + \gamma\delta_j(\hat{\mathbf{g}}) - \gamma\delta_j(\hat{\mathbf{h}})$.

The $D \times D$ inverse in Eq. (A.6) is intractable for efficient training, especially for a tracker with deep feature. More details are given in A.2. Using Woodbury formula [52], it can be expanded to reduce computational complexity:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}. \tag{A.7}$$

In our case, $A = \gamma I, C = I$, $U = [\alpha_0\delta_j(\hat{\mathbf{x}}_0), \ldots, \alpha_K\delta_j(\hat{\mathbf{x}}_K)]$ and $V = U^H$. By substituting them back to Eq. (A.7), the inverse factor can be expanded as:

$$\left(\sum_{k=0}^{K}\alpha_k^2\delta_j(\hat{\mathbf{x}}_k)\delta_j(\hat{\mathbf{x}}_k)^{\mathrm{H}} + \gamma I\right)^{-1} = \frac{1}{\gamma} - \frac{1}{\gamma}U(\gamma + U^HU)^{-1}U^H. \tag{A.8}$$

At this step, the dimension of the matrix to inverse have been reduced from the dimension of features D to the number of patches $K + 1$. High dimensional deep features are widely adopted by the tracking community, but usually several patches will be enough. For the sake of efficiency, simplification must not stop until element-wise computation. With this aim, we need to inspect the property of matrix U^HU whose element is given by:

$$U^HU(a, b) = \alpha_a\alpha_b\delta_j(\hat{\mathbf{x}}_a)^{\mathrm{H}}\delta_j(\hat{\mathbf{x}}_b). \tag{A.9}$$

Kernel trick is widely used in visual tracking problems[12,13,46,53]. It maps the original feature into a higher dimensional space, in which the similarity between one candidate and the object can be measured easier. Dot product is also known as linear kernel, it tells, to an extend, the similarity between two vectors. When distinguishing features are employed, the difference among all patches can be revealed by inner product, and the following assumption holds:

$$\hat{\mathbf{x}}_i^H\hat{\mathbf{x}}_j \approx 0 \ \text{ if } i \neq j. \tag{A.10}$$

Then matrix U^HU becomes approximately diagonal, thus the information from it lies mainly along its diagonal. But that actually breaks the solution into K separate equations for each patch respectively, which is against the motivation of utilizing all context patches. A proper choice for the approximation of matrix U^HU can be its trace (the sum of its elements along the diagonal), which combines all principal information while maintains a neat form:

$$U^HU \approx tr(U^HU) = \sum_{k=0}^{K}\alpha_k^2\delta_j(\hat{\mathbf{x}}_k)^H\delta_j(\hat{\mathbf{x}}_k). \tag{A.11}$$

Therefore, the approximate solution to $p(\delta_j(\mathbf{w}))$ is given by:

$$\delta_j(\hat{\mathbf{w}}) = \frac{1}{\gamma}\left(I - \frac{\sum_{k=0}^{K}\alpha_k^2\delta_j(\hat{\mathbf{x}}_k)\delta_j(\hat{\mathbf{x}}_k)^{\mathrm{H}}}{\gamma + \sum_{k=0}^{K}\alpha_k^2\delta_j(\hat{\mathbf{x}}_k)^{\mathrm{H}}\delta_j(\hat{\mathbf{x}}_k)}\right)\mathbf{q}. \tag{A.12}$$

With more investigation into Eq. (A.12), the computation speed can get further promotion. $\sum_{k=0}^{K}\alpha_k^2\delta_j(\hat{\mathbf{x}}_t)\delta_j(\hat{\mathbf{x}}_t)^{\mathrm{H}} = UU^H$ has the size of $D \times D$. Suppose $\hat{y}_{j0} = \hat{y}_j$ and $\hat{y}_{j1}, \ldots, \hat{y}_{jK} = 0$, $\alpha_0\delta_j(\hat{\mathbf{x}}_0)\hat{y}_j$ is then equal to a matrix multiplication:

$$\alpha_0\delta_j(\hat{\mathbf{x}}_0)\hat{y}_j = \sum_{k=0}^{K}\alpha_k\delta_j(\hat{\mathbf{x}}_k)\hat{y}_{jk} = UY \tag{A.13}$$

where $Y = [\hat{y}_{j0}, \ldots, \hat{y}_{jK}]^{\mathrm{H}}$ is actually composed by the label function for each patch. $\delta_j(\hat{\mathbf{g}})$ and $\delta_j(\hat{\mathbf{h}})$ are both of size $D \times 1$, which is already conformable with matrix U^{H} of size $(K+1) \times D$, Eq. (A.12) can also be expressed as:

$$\delta_j(\hat{\mathbf{w}}) = \frac{\mathbf{q}}{\gamma} - \frac{U}{\gamma b} \times (S_{\mathbf{xx}} Y + \gamma S_{\mathbf{gx}} - \gamma S_{\mathbf{hx}}) \tag{A.14}$$

where $S_{\mathbf{xx}} = \sum_{k=0}^{K} \alpha_k^2 \delta_j(\hat{\mathbf{x}}_k)^H \delta_j(\hat{\mathbf{x}}_k)$ is the approximation made in Eq. (A.11) and $b = \gamma + S_{\mathbf{xx}}$ is also a scalar. $S_{\mathbf{gx}} = U^{\mathrm{H}} \delta_j(\hat{\mathbf{g}}), S_{\mathbf{hx}} = U^{\mathrm{H}} \delta_j(\hat{\mathbf{h}})$ are both of size $(K+1) \times 1$, which is relatively smaller.

A2. Mathematical convenience

Here are the details about Eq. (A.4), (A.5), (A.6). Parseval's theorem for discrete Fourier transform is known as:

$$\sum_{n=0}^{N-1} \|\mathbf{x}(n)\|^2 = \frac{1}{N} \sum_{n=0}^{N-1} \|\hat{\mathbf{x}}(n)\|^2 \tag{A.15}$$

where $\mathbf{x} \in \mathbb{R}^N$ and $\hat{\mathbf{x}}$ denotes the DFT of it. The variables in Eq. (4a) are all of size $M \times N$, therefore, when applying Parseval's theorem, each term will be divided by MN:

$$\begin{aligned} MN \times p(\mathbf{w}) = & \frac{1}{2} \left\| \sum_{d=1}^{D} \mathcal{F}(\mathbf{x}^d \star \mathbf{w}^d) - \mathcal{F}(\mathbf{y}) \right\|^2 \\ & + \frac{\gamma}{2} \sum_{d=1}^{D} \left\| \mathcal{F}(\mathbf{w}^d) - \mathcal{F}(\mathbf{g}^d) + \mathcal{F}(\mathbf{h}^d) \right\|^2 \\ & + \frac{1}{2} \sum_{k=1}^{K} \left\| \sum_{d=1}^{D} \alpha_k \mathcal{F}(\mathbf{x}_k^d \star \mathbf{w}^d) \right\|^2 \end{aligned} \tag{A.16}$$

where $\mathcal{F}$ stands for the discrete Fourier transform. The discrete correlation $\mathbf{x}^d \star \mathbf{w}^d$ can be reformulated using a circulant matrix:

$$\mathbf{x}^d \star \mathbf{w}^d = \mathcal{C}(\mathbf{x}^d)\mathbf{w}^d \tag{A.17}$$

where $\mathcal{C}(\mathbf{x}^d)$ is a circulant matrix generated by $\mathbf{x}^d$. During this derivation, those matrices are vectorized, but the notations are kept simple for better understanding. The DFT of a vector $\mathbf{x}$ can be expressed as its multiplication with the Fourier matrix F, which can diagonalize $\mathcal{C}(\mathbf{x}^d)$:

$$\mathcal{C}(\mathbf{x}^d) = F diag(\hat{\mathbf{x}}^d) F^{\mathrm{H}} \tag{A.18a}$$

$$\mathcal{C}(\mathbf{x}^d) = F^{\mathrm{H}} diag((\hat{\mathbf{x}}^d)^*) F \tag{A.18b}$$

where * denotes complex conjugate, the superscript H means Hermitian transpose and $diag(\hat{\mathbf{x}}^d)$ stands for a diagonal matrix with elements in $\hat{\mathbf{x}}^d$ along its diagonal. Eq. (A.18b) can be obtained simply by taking the conjugate of Eq. (A.18a). Therefore Eq. (A.16) is equal to:

$$\begin{aligned} MN \times p(\mathbf{w}) = & \frac{1}{2} \left\| \sum_{d=1}^{D} F(F^{\mathrm{H}} diag((\hat{\mathbf{x}}^d)^*) \hat{\mathbf{w}}^d) - \hat{\mathbf{y}} \right\|^2 \\ & + \frac{\gamma}{2} \sum_{d=1}^{D} \left\| \hat{\mathbf{w}}^d - \hat{\mathbf{g}}^d + \hat{\mathbf{h}}^d \right\|^2 \\ & + \frac{1}{2} \sum_{k=1}^{K} \left\| \sum_{d=1}^{D} \alpha_k F(F^{\mathrm{H}} diag((\hat{\mathbf{x}}^d)^*) \hat{\mathbf{w}}^d) \right\|^2 . \end{aligned} \tag{A.19}$$

Since F is an unitary matrix:

$$\begin{aligned} MN \times p(\mathbf{w}) = & \frac{1}{2} \left\| \sum_{d=1}^{D} (\hat{\mathbf{x}}^d)^* \cdot \hat{\mathbf{w}}^d - \hat{\mathbf{y}} \right\|^2 \\ & + \frac{\gamma}{2} \sum_{d=1}^{D} \left\| \hat{\mathbf{w}}^d - \hat{\mathbf{g}}^d + \hat{\mathbf{h}}^d \right\|^2 \\ & + \frac{1}{2} \sum_{k=1}^{K} \left\| \sum_{d=1}^{D} \alpha_k (\hat{\mathbf{x}}_k^d)^* \cdot \hat{\mathbf{w}}_k^d \right\|^2 . \end{aligned} \tag{A.20}$$

In the main text, $\hat{\mathbf{x}}^d$, instead of $(\hat{\mathbf{x}}_k^d)^*$, is used for mathematical convenience.

For Eq. (A.5), set its partial derivative with respect to $\mathbf{w}$ at zero:

$$\begin{aligned} \frac{\partial p(\delta_j(\hat{\mathbf{w}}))}{\partial \mathbf{w}} = & (\delta_j(\hat{\mathbf{w}})^{\mathrm{H}} \delta_j(\hat{\mathbf{x}}) - \hat{y}_j^{\mathrm{H}}) \delta_j(\hat{\mathbf{x}})^{\mathrm{H}} \\ & + \gamma [\delta_j(\hat{\mathbf{w}})^{\mathrm{H}} - \delta_j(\hat{\mathbf{g}})^{\mathrm{H}} + \delta_j(\hat{\mathbf{h}})^{\mathrm{H}}] \\ & + \delta_j(\hat{\mathbf{w}})^{\mathrm{H}} \sum_{k=1}^{K} \delta_j(\hat{\mathbf{x}}_k) \delta_j(\hat{\mathbf{x}}_k)^{\mathrm{H}} = 0. \end{aligned} \tag{A.21}$$

$$\begin{aligned} & \delta_j(\hat{\mathbf{w}})^{\mathrm{H}} \left(\sum_{k=0}^{K} \delta_j(\hat{\mathbf{x}}_k) \delta_j(\hat{\mathbf{x}}_k)^{\mathrm{H}} + \gamma \right) \\ & = \hat{y}_j^{\mathrm{H}} \delta_j(\hat{\mathbf{x}})^{\mathrm{H}} + \gamma \delta_j(\hat{\mathbf{g}})^{\mathrm{H}} - \gamma \delta_j(\hat{\mathbf{h}})^{\mathrm{H}}. \end{aligned} \tag{A.22}$$

Then take Hermitian transpose both sides:

$$\begin{aligned} & \left(\sum_{k=0}^{K} \delta_j(\hat{\mathbf{x}}_k) \delta_j(\hat{\mathbf{x}}_k)^{\mathrm{H}} + \gamma \right) \delta_j(\hat{\mathbf{w}}) \\ & = \delta_j(\hat{\mathbf{x}}) \hat{y}_j + \gamma \delta_j(\hat{\mathbf{g}}) - \gamma \delta_j(\hat{\mathbf{h}}) \end{aligned} \tag{A.23}$$

which is equivalent to Eq. (A.6).

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.sigpro.2019.107324.

References

[1] J. Prokaj, G. Medioni, Persistent tracking for wide area aerial surveillance, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1186–1193.

[2] P.T. Eendebak, A.W.M. van Eekeren, R.J.M. den Hollander, Landing spot selection for UAV emergency landing, in: Unmanned Systems Technology XV, 8741, International Society for Optics and Photonics, 2013, p. 874105.

[3] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I.L. Grixa, F. Ruess, M. Suppa, D. Burschka, Toward a fully autonomous UAV: research platform for indoor and outdoor urban search and rescue, IEEE Robot. Autom. Mag. 19 (3) (2012) 46–56.

[4] K. Kang, J.V.R. Prasad, E. Johnson, Active control of a uav helicopter with a slung load for precision airborne cargo delivery, Unmanned Syst. 4 (03) (2016) 213–226.

[5] A.W.M. Smeulders, D.M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, M. Shah, Visual tracking: an experimental survey, IEEE Trans. Pattern Anal. Mach. Intell. 36 (7) (2014) 1442–1468.

[6] Y. Yin, X. Wang, D. Xu, F. Liu, Y. Wang, W. Wu, Robust visual detection–learning–tracking framework for autonomous aerial refueling of UAVs, IEEE Trans. Instrum. Meas. 65 (3) (2016) 510–521.

[7] K.E. Wenzel, A. Masselli, A. Zell, Automatic take off, tracking and landing of a miniature UAV on a moving carrier vehicle, J. Intell. Robot. Syst. 61 (1–4) (2011) 221–238.

[8] C.-T. Dang, H.-T. Pham, T.-B. Pham, N.-V. Truong, Vision based ground object tracking using AR. drone quadrotor, in: 2013 International Conference on Control, Automation and Information Sciences (ICCAIS), IEEE, 2013, pp. 146–151.

[9] A.C. Woods, H.M. La, Dynamic target tracking and obstacle avoidance using a drone, in: International Symposium on Visual Computing, Springer, 2015, pp. 857–866.

[10] R. Bartak, A. Vykovsky, Any object tracking and following by a flying drone, in: Fourteenth Mexican International Conference on Artificial Intelligence (MICAI), IEEE, 2015, pp. 35–41.

[11] M. Mueller, N. Smith, B. Ghanem, Context-aware correlation filter tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1396–1404.

[12] J.F. Henriques, R. Caseiro, P. Martins, J. Batista, Exploiting the circulant structure of tracking-by-detection with kernels, in: European conference on computer vision, Springer, 2012, pp. 702–715.
[13] J.F. Henriques, R. Caseiro, P. Martins, J. Batista, High-speed tracking with kernelized correlation filters, IEEE Trans. Pattern Anal. Mach. Intell. 37 (3) (2015) 583–596.
[14] R.M. Gray, Others, Toeplitz and circulant matrices: a review, Foundations and Trends®in Communications and Information Theory 2 (3) (2006) 155–239.
[15] M. Danelljan, G. Hager, F. Shahbaz Khan, M. Felsberg, Learning spatially regularized correlation filters for visual tracking, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 4310–4318.
[16] H. Kiani Galoogahi, T. Sim, S. Lucey, Correlation filters with limited boundaries, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4630–4638.
[17] S. Bubeck, Others, Convex optimization: algorithms and complexity, Found. Trends®.Learn. 8 (3–4) (2015) 231–357.
[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Others, Distributed optimization and statistical learning via the alternating direction method of multipliers, Found. Trends® Mach.learn. 3 (1) (2011) 1–122.
[19] D.S. Bolme, J.R. Beveridge, B.A. Draper, Y.M. Lui, Visual object tracking using adaptive correlation filters, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 2544–2550.
[20] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: international Conference on computer vision & Pattern Recognition (CVPR'05), 1, IEEE Computer Society, 2005, pp. 886–893.
[21] M. Danelljan, F. Shahbaz Khan, M. Felsberg, J. de Weijer, Adaptive color attributes for real-time visual tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1090–1097.
[22] J. Van De Weijer, C. Schmid, J. Verbeek, D. Larlus, Learning color names for real-world applications, IEEE Trans. Image Process. 18 (7) (2009) 1512–1523.
[23] Y. Li, J. Zhu, A scale adaptive kernel correlation filter tracker with feature integration, in: European conference on computer vision, Springer, 2014, pp. 254–265.
[24] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, P.H.S. Torr, Staple: complementary learners for real-time tracking, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 1401–1409.
[25] M. Danelljan, G. Häger, F.S. Khan, M. Felsberg, Discriminative scale space tracking, IEEE Trans. Pattern Anal. Mach. Intell. 39 (8) (2017) 1561–1575.
[26] M. Danelljan, G. Hager, F. Shahbaz Khan, M. Felsberg, Adaptive decontamination of the training set: aunified formulation for discriminative visual tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1430–1438.
[27] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, Y. Singer, Online passive-aggressive algorithms, J. Mach. Learn. Res. 7 (Mar) (2006) 551–585.
[28] F. Li, C. Tian, W. Zuo, L. Zhang, M.-H. Yang, Learning spatial-temporal regularized correlation filters for visual tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4904–4913.
[29] H. Kiani Galoogahi, A. Fagg, S. Lucey, Learning background-aware correlation filters for visual tracking, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1135–1143.
[30] N. Wang, J. Shi, D.-Y. Yeung, J. Jia, Understanding and diagnosing visual tracking systems, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 3101–3109.
[31] K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the devil in the details: delving deep into convolutional nets, arXiv:1405.3531 (2014).
[32] C. Ma, J.-B. Huang, X. Yang, M.-H. Yang, Hierarchical convolutional features for visual tracking, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 3074–3082.
[33] F. Li, Y. Yao, P. Li, D. Zhang, W. Zuo, M.-H. Yang, Integrating boundary and center correlation filters for visual tracking with aspect ratio variation, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2001–2009.
[34] N. Wang, W. Zhou, Q. Tian, R. Hong, M. Wang, H. Li, Multi-cue correlation filters for robust visual tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4844–4853.
[35] M. Danelljan, A. Robinson, F.S. Khan, M. Felsberg, Beyond correlation filters: learning continuous convolution operators for visual tracking, in: European Conference on Computer Vision, Springer, 2016, pp. 472–488.
[36] M. Danelljan, G. Bhat, F. Shahbaz Khan, M. Felsberg, ECO: efficient convolution operators for tracking, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 6638–6646.
[37] S. Yun, J. Choi, Y. Yoo, K. Yun, J. Young Choi, Action-decision networks for visual tracking with deep reinforcement learning, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2711–2720.
[38] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, P.H. Torr, End-to-end representation learning for correlation filter based tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2805–2813.
[39] R. Tao, E. Gavves, A.W. Smeulders, Siamese instance search for tracking, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 1420–1429.
[40] L. Bertinetto, J. Valmadre, J.F. Henriques, A. Vedaldi, P.H. Torr, Fully-convolutional siamese networks for object tracking, in: European conference on computer vision, Springer, 2016, pp. 850–865.
[41] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, IEEE Trans. Pattern Anal. Mach. Intell. 32 (9) (2010) 1627–1645.
[42] M. Mueller, N. Smith, B. Ghanem, A benchmark and simulator for uav tracking, in: European conference on computer vision, Springer, 2016, pp. 445–461.
[43] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li, W. Zhang, Q. Huang, Q. Tian, The unmanned aerial vehicle benchmark: object detection and tracking, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 370–386.
[44] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, M. Kristan, Discriminative correlation filter with channel and spatial reliability, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6309–6318.
[45] N. Wang, Y. Song, C. Ma, W. Zhou, W. Liu, H. Li, Unsupervised deep tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 1308–1317.
[46] C. Wang, L. Zhang, L. Xie, J. Yuan, Kernel cross-correlator, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[47] J. Zhang, S. Ma, S. Sclaroff, MEEM: robust tracking via multiple experts using entropy minimization, in: European conference on computer vision, Springer, 2014, pp. 188–203.
[48] M. Danelljan, G. Häger, F. Khan, M. Felsberg, Accurate scale estimation for robust visual tracking, in: British Machine Vision Conference, Nottingham, September 1–5, 2014, BMVA Press, 2014.
[49] H. Fan, H. Ling, Parallel tracking and verifying: a framework for real-time and high accuracy visual tracking, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5486–5494.
[50] Y. Wu, J. Lim, M.H. Yang, Object tracking benchmark., IEEE Trans. Pattern Anal. Mach. Intell. 37 (9) (2015) 1834.
[51] J. Frankle, M. Carbin, The lottery ticket hypothesis: finding sparse, trainable neural networks, arXiv:1803.03635(2018).
[52] G. Strang, Introduction to Linear Algebra, fifth ed., Wellesley-Cambridge Press Wellesley, MA, 2016.
[53] M. Tang, J. Feng, Multi-kernel correlation filter for visual tracking, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 3038–3046.