2018 24th International Conference on Pattern Recognition (ICPR)
Beijing, China, August 20-24, 2018

SPCNet: Scale Position Correlation Network for End-to-End Visual Tracking

Qiang Wang*†, Jin Gao†, Mengdan Zhang*†, Junliang Xing†, and Weiming Hu†
*University of Chinese Academy of Sciences, Beijing, China.
†National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China.
{qiang.wang, jin.gao, mengdan.zhang, jlxing, wmhu}@nlpr.ia.ac.cn

Abstract—**We present a novel Scale Position Correlation Network (SPCNet) for learning to track objects robustly and efficiently. Different from most previous Correlation Filter (CF) based tracking models, SPCNet unifies the feature representation learning and CF based appearance modeling within one end-to-end learnable framework. In particular, SPCNet learns to track objects within a joint scale-position space, and is very effective in learning features for the accurate prediction of object scale and position. To learn our model from end to end, the SPCNet introduces a differentiable correlation filter layer into a Siamese architecture. Therefore, the localization error can be effectively back-propagated through the whole network, enabling fast adaptation of feature learning and appearance modeling for the objects to be tracked. Such task driven feature learning admits a very lightweight design that can be efficiently pre-trained. In addition, the dense appearance modeling in the joint scale-position space is also efficient. It benefits from the computation of gradients within the Fourier frequency domain. Such careful architecture design ensures that SPCNet is effective and efficient with a small model size. Extensive experimental analyses and evaluations on three largest benchmarks, OTB-2013, OTB-2015, and VOT2015, demonstrate its superiority over many state-of-the-art algorithms.**

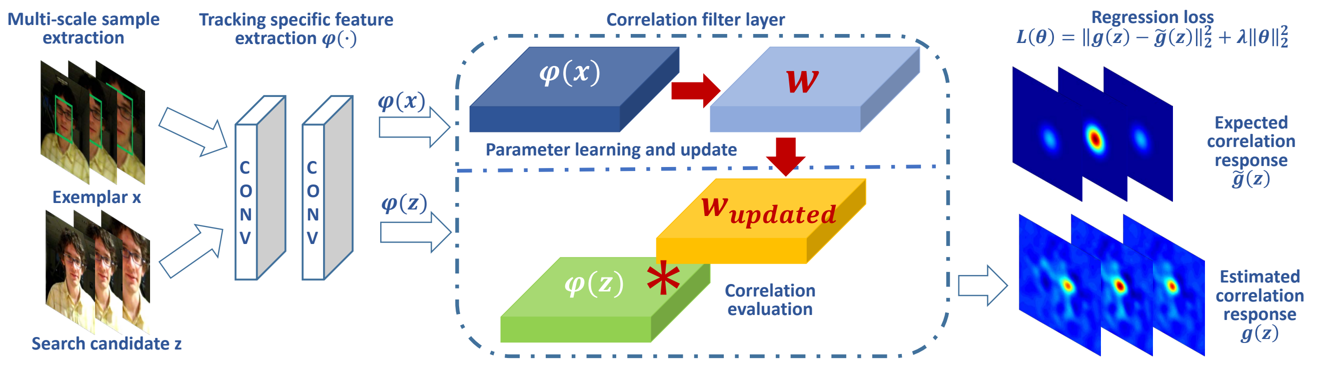
I. Introduction

Object tracking is a fundamental problem in computer vision with wide applications such as human computer interaction [1] and assistant driving systems [2]. One common setting for this problem is to specify the object of interest in the first frame with a bounding box. The objective is to estimate the trajectory of the object in subsequent frames [3], [4]. Despite significant progress made in last decade [5], [6], [7], [8], [9], it still remains a challenging problem due to object deformation, scale variation, partial occlusion, *etc*. Moreover, maintaining real-time processing is also vital for visual tracking. This condition rules out many state-of-the-art trackers.

Recently, correlation filter (CF) based trackers [10], [11], [7] have received great attention because of their remarkable tracking performance and efficiency. These trackers model the correlations between the object patch and the full set of approximate surrounding patches in the position space by solving a ridge regression problem efficiently in the Fourier frequency domain. To further boost the tracking performance, many improvements have been made [12], [9], [13], [14]. To enhance the discriminative power of a tracker, the object representation has evolved from simple hand-crafted features (*eg*, raw gray features [10], HOG [15], [7] and Color Names [16]) to pre-trained multi-layer deep features [13], [17], [18], [9]. The scale factor is considered in DSST [11] by extending the object search space from a single position space to the joint scale-position space. The correlation analysis is also transformed into a verification problem to leverage the effective Siamese network and external large video dataset [19], [20].

Despite significant progress, some obstacles for extending CF based trackers to more realistic application scenarios still remain prominent. First, the object scale factor is usually ignored or not well considered in the appearance modeling process. This makes it difficult for trackers to take advantage of the correlations between object position and scale. For instance, the seperately exploited position filter and one-dimensional scale filter in [11] can not handle the simultaneous variations of object scale and position well. Moreover, the visual features for these trackers are either hand-crafted or chosen from unchangeable pre-trained networks for the image classification task [21], [22] or object detection task [23]. Therefore, feature extraction and object tracking appearance modeling are isolated from each other and cannot co-adapt to achieve better tracking performance. By exploiting more layers of deep features and combining multiple features, the increasingly complex models, with massive trainable parameters, inevitably introduce the risk of severe over-fitting and expensive computing [9]. A good trade-off between tracking accuracy and speed is often absent, which hampers the application of real-time visual tracking. For Siamese network based trackers [24], [19], feature extraction and correlation analysis are performed simultaneously, and the tracking speed is quite fast. However, there is no on-line learning process. A fixed metric obtained from the correlation analysis is used. It is not possible to take advantage of video-specific cues, and tracking adaptability is lost.

To address the above issues, we present an end-to-end lightweight network architecture for visual tracking. The proposed SPCNet learns the convolutional feature representations and performs adaptive correlation tracking in the joint scale-position space simultaneously. Specifically, SPCNet adopts a Siamese architecture to automatically learn optimized features for better CF based object tracking. The correlation operation is treated as a special correlation filter layer. More importantly, the introduced correlation filter layer is *differentiable* and thus the overall network can be trained end-to-end through error back-propagation. SPCNet only requires a few convolutional layers to learn features which encode the prior tracking knowl-



Multi-scale sample extraction
Tracking specific feature extraction $\varphi(\cdot)$
Correlation filter layer
Regression loss
$L(\theta) = \|g(z) - \tilde{g}(z)\|_2^2 + \lambda\|\theta\|_2^2$
Exemplar x
Search candidate z
CONV
CONV
$\varphi(x)$
$\varphi(z)$
w
Parameter learning and update
$w_{updated}$
Correlation evaluation
Expected correlation response $\tilde{g}(z)$
Estimated correlation response $g(z)$

Fig. 1: The overall SPCNet architecture. SPCNet is quite lightweight with only two convolutional layers and one correlation filter layer. It is end-to-end trainable in the joint scale and position space of the object state, which permits task-driven feature extraction and adaptive appearance modeling in the CF based tracking framework.

edge in the pre-training process of the whole tracking network by leveraging large video object detection datasets and provides outstanding efficiency. The correlation filter layer works on convolutional features extracted from the object template and the search area in the joint scale-position space, and then a joint correlation analysis is performed to estimate the object position and size simultaneously. The weights of this layer are learnt and updated online in the same joint space to make the tracker adaptive to continuous object appearance changes. Since the derivation of the correlation filter layer is still done in the Fourier frequency domain, the efficiency property of CF is preserved. Despite the fact that our innovation of introducing a differentiable CF layer into a Siamese architecture bears some similarity to the very recent work of [20], we in contrast consider a CF layer which explicitly operates in the scale space in addition to the position space, and the differentiable CF layer can propagate the gradient from both the position and scale estimation errors. Our used regression loss in the end-to-end training is also more suitable for tracking than the element-wise logistic loss in [20].

Benefiting from the above novel designs, our proposed SPCNet provides high speed of over 60 FPS, while still achieving competitive tracking accuracy with state-of-the-art trackers with heavier weights.

II. The Proposed SPCNet

SPCNet offers high tracking accuracy and real-time tracking speed. Its pipeline is summarized as follows. The SPCNet consists of two convolutional layers for feature learning and a correlation filter layer for visual tracking, as shown in Fig. 1. The regression loss of joint scale-position correlation response is back-propagated through the whole network to adjust the parameters for both feature extraction and object appearance modeling. In the offline training stage, the SPCNet is trained from scratch and end-to-end based on the large ILSVRC15 video object detection dataset [25]. This greatly enhances the representation power of its feature extraction module, without requiring scarce online tracking data while obtaining features suited to tracking. The convolutional layers for feature extraction are extensively trained off-line, and for this reason are very robust and accurate. In the online tracking stage, these convolutional layers are frozen to save processing time, avoid overfitting, and reduce tracking drift. Meanwhile, the correlation filter layer is updated continuously to tackle object appearance variations. The object size and position are simultaneously estimated according to the maximum of the joint scale-position correlation response.

In the following subsections, we first introduce the joint scale-position estimation of the object in the SPCNet based tracking process. Then, the learning process of the tracking task driven features is elaborated. Finally, we explain the update process of the correlation filter layer in the spirit of Recurrent Neural Network (RNN).

A. *Joint Scale-Position Estimation in Visual Tracking*

In order to enhance the discriminative power of SPCNet in visual tracking, our correlation filter layer models the object and contextual appearance correlations in the joint scale-position space instead of the pure position space. Thus, the simultaneous variation of object size and position is learnt by our SPCNet and accurately estimated in the tracking process.

When a new video frame arrives, based on the object size and position in previous frame, we first extract the candidate samples in the joint scale-position space. This is done by cropping the object region from current frame and its scaled versions, and further resizing them to a fixed size with $M \times N$ pixels. Let S be the number of scales and $\mathbf{z}_s$ be the candidate sample in the s-th scale space, these candidate samples can thus be denoted as $\{\mathbf{z}_s\}_{s=1}^{S}$. The convolutional layers for feature extraction work on these samples and output $D-$dimensional feature representations $\{\varphi^l(\mathbf{z}_s)\}_{l=1}^{D}$. The full set of parameters of the correlation filter layer is represented as $\mathbf{w}$, containing one $M \times N$ correlation filter $\mathbf{w}^l$ per feature layer. Therefore, let $\hat{\mathbf{x}}$ denote the discrete Fourier transform of $\mathbf{x}$, *ie*, $\mathcal{F}(\mathbf{x})$, $\mathbf{x}^*$ represent the complex conjugate of $\mathbf{x}$, and $\odot$ denote the Hadamard product. Then, the correlation response of the sample $\mathbf{z}_s$ output by the correlation filter layer is obtained as:

$$g(\mathbf{z}_s) = \sum_{l=1}^{D} \mathbf{w}^l \star \varphi^l(\mathbf{z}_s) = \mathcal{F}^{-1}\left(\sum_{l=1}^{D} \hat{\mathbf{w}}^{l*} \odot \hat{\varphi}^l(\mathbf{z}_s)\right), \quad (1)$$

where $\star$ denotes a circular correlation operator. The object size and position are estimated in one go based on the maximum of the joint scale-position correlation response.

The main advantage of such a correlation filter layer is that its parameter set $\mathbf{w}$ can be learnt and updated online to adapt to the appearance variation of the current video object. This approach is superior to the fixed similarity metric in conventional Siamese networks. This parameter learning process in the joint scale-position space is described as follows.

The correlation filter layer is learnt using a set of training samples from T frames in the joint scale-position space, which can be represented as $\{(\varphi(\mathbf{x}_{s,t}), \mathbf{y}_s)\}_{s=1,t=1}^{S,T}$. Here $\mathbf{y}_s$ is the desired correlation output in the s-th scale space, and $\{\mathbf{y}_s\}_{s=1}^{S}$ is constructed as a 3-D Gaussian function with its peak at the target's centre position and scale (see the expected correlation response $\tilde{g}(\mathbf{z})$ in Fig. 1 for more details). These training samples are extracted based on the tracking results in previous frames and are resized to the same spatial size $M \times N$. The correlation response of these training samples estimated by our correlation filter layer is given as $\{g(\mathbf{x}_s) = \sum_{l=1}^{D} \mathbf{w}^l \star \varphi^l(\mathbf{x}_s)\}_{s=1}^{S}$. The correlation filter layer is optimized by minimizing a ridge regression loss:

$$\epsilon = \sum_{t=1}^{T} \beta_t \left(\sum_{s=1}^{S} \|g(\mathbf{x}_{s,t}) - \mathbf{y}_s\|_2^2 + \lambda \sum_{l=1}^{D} \left\|\mathbf{w}^l\right\|_2^2 \right), \quad (2)$$

where $\beta_t \geq 0$ is the impact of training samples from the t-th frame, and the constant $\lambda \geq 0$ controls the relative weight of the regularization term.

We transform the ridge regression problem of Eqn. (2) to the Fourier domain using Parseval's formula. The objective function is real valued, positive, and convex, so it is possible to obtain the global optima by setting the partial derivative equal to zero. Then, the solution is obtained as:

$$\hat{\mathbf{w}}^l = \frac{\sum_{t=1}^{T} \beta_t \left(\sum_{s=1}^{S} \hat{\mathbf{y}}_s^* \odot \hat{\varphi}^l(\mathbf{x}_{s,t}) \right)}{\sum_{t=1}^{T} \beta_t \left(\sum_{s=1}^{S} \sum_{k=1}^{D} \hat{\varphi}^k(\mathbf{x}_{s,t}) \odot (\hat{\varphi}^k(\mathbf{x}_{s,t}))^* + \lambda \right)}. \quad (3)$$

Since this correlation filter layer is learnt in the Fourier frequency domain using several discrete Fourier transforms and element-wise multiplications, the computing in this layer is quite efficient. In particular, the closed-form solution provided here can avoid an expensive iterative optimization process.

B. Task Driven Feature Learning for tracking

Existing CF based trackers usually separate feature extraction from correlation analysis, which tends to give sub-optimal results. The features exploited in these trackers are either handcrafted or chosen from pre-trained networks for the image classification task or the object detection task. These features may not be suitable for tracking. Therefore, we introduce a differentiable correlation filter layer into a Siamese network to enable the learnable parameters in the two components to co-adapt and cooperate to provide an expected correlation response. The regression loss of the correlation response in the joint scale-position space is back-propagated through the whole network to automatically learn tracking task driven features. Here, we give details of the backward propagation in SPCNet.

As shown in Fig. 1, SPCNet consists of two branches. A filter learning branch exploits object exemplars in the joint scale-position space to learn the parameters in the correlation filter layer, as aforementioned in Eqn. (3). The other tracking branch works on candidate search samples from the joint space and finally calculates their correlation response in the correlation filter layer. Then, the network is trained by minimizing the differences between the real response and the expected 3-D Gaussian-shaped response. Therefore, let each training pair be $(\{\mathbf{z}_s\}_{s=1}^{S}, \{\mathbf{x}_s\}_{s=1}^{S})$, the expected 3-D Gaussian-shaped response for candidate samples be $\{\tilde{g}(\mathbf{z}_s)\}_{s=1}^{S}$ (see Fig. 1), and convolutional parameters for feature extraction be $\boldsymbol{\theta}$. Our offline training problem is formulated as:

$$L(\boldsymbol{\theta}) = \sum_{s=1}^{S} \|g(\mathbf{z}_s) - \tilde{g}(\mathbf{z}_s)\|_2^2 + \gamma \|\boldsymbol{\theta}\|_2^2, \quad (4)$$

where

$$g(\mathbf{z}_s) = \mathcal{F}^{-1} \left(\sum_{l=1}^{D} \hat{\mathbf{w}}^{l*} \odot \hat{\varphi}^l(\mathbf{z}_s; \boldsymbol{\theta}) \right), \quad (5)$$

$$\hat{\mathbf{w}}^l = \frac{\sum_{s=1}^{S} \hat{\mathbf{y}}_s^* \odot \hat{\varphi}^l(\mathbf{x}_s; \boldsymbol{\theta})}{\sum_{s=1}^{S} \sum_{k=1}^{D} \hat{\varphi}^k(\mathbf{x}_s; \boldsymbol{\theta}) \odot (\hat{\varphi}^k(\mathbf{x}_s; \boldsymbol{\theta}))^* + \lambda}. \quad (6)$$

An explicit regularization should be incorporated for better convergence. We use the weight decay method in the conventional parameter optimization to make implicit this regularization. Besides, to restrict the magnitude of feature map values and increase the stability of the training process, we add a Local Response Normalization (LRN) layer [21] at the end of the convolutional layers.

Since the intermediate variables are complex-valued, we first give some preliminary facts. According to [26], the gradient of discrete Fourier transform and inverse discrete Fourier transform are formulated as:

$$\hat{g} = \mathcal{F}(g), \frac{\partial L}{\partial \hat{g}^*} = \mathcal{F}\left(\frac{\partial L}{\partial g}\right), \frac{\partial L}{\partial g} = \mathcal{F}^{-1}\left(\frac{\partial L}{\partial \hat{g}^*}\right). \quad (7)$$

Since the operations in the forward pass only contain Hadamard product and division, we can calculate the derivative per-element:

$$\frac{\partial L}{\partial \hat{g}_{uv}^*(\mathbf{z}_s)} = \left(\mathcal{F}\left(\frac{\partial L}{\partial g(\mathbf{z}_s)} \right) \right)_{uv}. \quad (8)$$

For the back-propagation of the tracking branch,

$$\begin{aligned} \frac{\partial L}{\partial (\hat{\varphi}_{uv}^l(\mathbf{z}_s))^*} &= \frac{\partial L}{\partial \hat{g}_{uv}^*(\mathbf{z}_s)} \frac{\partial \hat{g}_{uv}^*(\mathbf{z}_s)}{\partial (\hat{\varphi}_{uv}^l(\mathbf{z}_s))^*} \\ &= \frac{\partial L}{\partial \hat{g}_{uv}^*(\mathbf{z}_s)} (\hat{\mathbf{w}}_{uv}^l), \end{aligned} \quad (9)$$

$$\frac{\partial L}{\partial \varphi^l(\mathbf{z}_s)} = \mathcal{F}^{-1}\left(\frac{\partial L}{\partial (\hat{\varphi}^l(\mathbf{z}_s))^*} \right). \quad (10)$$

For the back-propagation of the filter learning branch, we treat $\hat{\varphi}^l_{uv}(\mathbf{x}_s)$ and $(\hat{\varphi}^l_{uv}(\mathbf{x}_s))^*$ as independent variables.

$$\frac{\partial \hat{g}^*_{uv}(\mathbf{z}_s)}{\partial \hat{\varphi}^l_{uv}(\mathbf{x}_s)} = \frac{(\hat{\varphi}^l_{uv}(\mathbf{z}_s))^* \hat{\mathbf{y}}^*_{s,uv} - \hat{g}^*_{uv}(\mathbf{z}_s)(\hat{\varphi}^l_{uv}(\mathbf{x}_s))^*}{\sum_{s=1}^{S}\sum_{k=1}^{D} \hat{\varphi}^k_{uv}(\mathbf{x}_s)(\hat{\varphi}^k_{uv}(\mathbf{x}_s))^* + \lambda}, \tag{11}$$

$$\frac{\partial \hat{g}^*_{uv}(\mathbf{z}_s)}{\partial (\hat{\varphi}^l_{uv}(\mathbf{x}_s))^*} = \frac{-\hat{g}^*_{uv}(\mathbf{z}_s)\hat{\varphi}^l_{uv}(\mathbf{x}_s)}{\sum_{s=1}^{S}\sum_{k=1}^{D} \hat{\varphi}^k_{uv}(\mathbf{x}_s)(\hat{\varphi}^k_{uv}(\mathbf{x}_s))^* + \lambda}, \tag{12}$$

$$\frac{\partial L}{\partial \varphi^l(\mathbf{x}_s)} = \mathcal{F}^{-1}\left(\frac{\partial L}{\partial (\hat{\varphi}^l(\mathbf{x}_s))^*} + \left(\frac{\partial L}{\partial \hat{\varphi}^l(\mathbf{x}_s)}\right)^*\right). \tag{13}$$

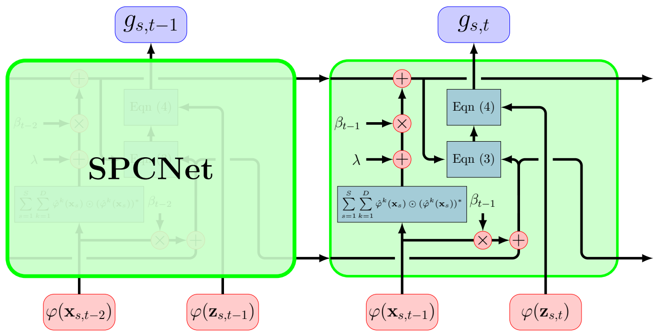
Once the error is propagated backwards to the real-valued feature maps, the rest of the back-propagation can be conducted as traditional CNN optimization. Since all operations of back-propagation in the correlation filter layer are still Hadamard operations in the Fourier frequency domain, we can retain the efficiency property of CF and apply the offline training on large-scale datasets. After the offline training has been completed, we get a tailored feature extractor for online CF tracking.

C. *Online Model Update: RNN-alike Implementation*

The online update of our correlation filter layer makes our tracker quite adaptive to continuous object appearance changes. Compared to general Siamese networks, our similarity embedding is able to adapt to the specific appearance of the tracking inistance. In particular, we find that the solution in Eqn. (3) of the optimization problem in Eqn. (2) can be posed as an incremental update process. The advantage is that we do not have to maintain a large exemplar set and only need small memory footprint. Therefore, we update our correlation filter layer similarly to RNN as shown in Fig. 2, while keeping it a much simpler architecture. Specifically, at the time instance t, we use the latest updated linear correlation filter $\mathbf{w}_{t-1}$ to test the samples $\{\varphi(\mathbf{z}_{s,t})\}_{s=1}^{S}$ and get the response output $\{g(\mathbf{z}_{s,t})\}_{s=1}^{S}$. The object size and position are simultaneously estimated by searching the joint scale-position space where the maximum value of the response output exists. Based on this tracking result, the training samples $\{\varphi(\mathbf{x}_{s,t})\}_{s=1}^{S}$ are extracted to incrementally update $\mathbf{w}_{t-1}$ to a new set of $\mathbf{w}_t$, such that the new filter will approximately output an expected Gaussian response when correlated with the training samples $\{\varphi(\mathbf{x}_{s,t})\}_{s=1}^{S}$.

III. Experiments

In this section, we first introduce our implementation details and the experimental settings. Then, ablation studies of the effectiveness and efficiency performance analysis of each component are carried out on the OTB-2013 [27] dataset. After that, the overall performance of our SPCNet tracker is evaluated on the whole OTB [27], [3] and VOT2015 [4] in comparison with a majority of the state-of-the-art trackers.



$g_{s,t-1}$
$g_{s,t}$
SPCNet
Eqn (4)
Eqn (3)
β_{t-1}
λ
$\sum_{s=1}^{S}\sum_{k=1}^{D}\hat{\varphi}^k(\mathbf{x}_s)\odot(\hat{\varphi}^k(\mathbf{x}_s))^*$
β_{t-1}
$\varphi(\mathbf{x}_{s,t-2})$
$\varphi(\mathbf{z}_{s,t-1})$
$\varphi(\mathbf{x}_{s,t-1})$
$\varphi(\mathbf{z}_{s,t})$

Fig. 2: The online tracking process of SPCNet. The Numerator (output of the horizontal pipeline at the bottom) and Denominator (output of the horizontal pipeline at the top) of $\hat{\mathbf{w}}_p$ are recurrently forward-propagated and updated as Eqn. (3).

A. *Implementation Details*

Training Data Preparation. To increase the generalization capability and discriminative power of our feature representation, and avoid over-fitting to the scarce tracking data, our SPCNet is pre-trained from scratch off-line on the training set of the ILSVRC15 video object detection dataset [25]. This training set consists of 7,911 objects and has little correlation with OTB and VOT. In each video snippet of an object, we collect each pair of frames within the nearest 10 frames, and feed the cropped pair of target patches of $2\times$ padding size to the network, resulting in 5,507,660 pairs in total. The cropped inputs are resized to a spatial resolution which is consistent between the off-line training and on-line tracking phases. A case study of the trade-off between the tracking accuracy and speed suggests a resolution of 125×125.

Network Setting. The feature extraction of our SPCNet consists of 2 convolutional layers with kernel size set to 3×3 and a Relu layer appended at the end of each convolutional layer. An LRN layer is also added to output the final feature representation. There is no pooling layer and the final output feature representation is forced to 32 channels. For the hyper-parameters in the correlation filter layer, the regularization coefficient λ is set to 10^{-4} and the Gaussian spatial bandwidth is set to 0.1 for both on-line tracking and off-line training. Similar to [28], we use a patch pyramid with the scale factors $\{a^s | a = 1.01, s = \lfloor -\frac{S-1}{2} \rfloor, \lfloor -\frac{S-3}{2} \rfloor, ..., \lfloor \frac{S-1}{2} \rfloor, S = 3\}$.

We apply stochastic gradient descent (SGD) with a momentum of 0.9 to train the network from scratch and set the weight decay γ to 0.0005. The learning rate exponentially decays from 10^{-2} to 10^{-5}. The model is trained for 50 epochs with a mini-batch size of 32. SPCNet is implemented in MATLAB with MatConvNet [29]. Experiments are conducted on a workstation with an Intel Xeon 2630 at 2.4GHz and a NVIDIA GeForce GTX 1080 GPU.

Benchmark Datasets. OTB is a standard benchmark for visual tracking which contains 100 fully annotated targets with 11 different attributes. We follow the protocol of OTB and report results based on success plots and precision plots for evaluation. The success plots show the percentage of frames in which the overlap score exceeds a threshold. The precision plots show the percentage of frames where the center location

TABLE I: Ablation study of tracking components on OTB using mean overlap precision (OP) at a threshold of 0.5, mean distance precision (DP) of 20 pixels and mean speed (FPS).

Trackers	OTB-2013		OTB-2015		FPS
	OP	DP	OP	DP	
DCF [7]	61.6	72.8	54.8	68.9	**292**
DCFNet1s	67.7	79.1	63.7	76.8	*187*
SAMF [30]	67.7	78.5	64.0	74.3	12
DSST [11]	67.1	74.7	60.9	68.9	46
DCF+VGG	62.1	66.1	61.7	66.9	88
DCF+SiamFC	66.8	74.2	64.0	68.0	77
DCFNet	*78.5*	*86.7*	*72.8*	*79.4*	109
SPCNet	**84.3**	**88.1**	**77.6**	**82.9**	67

error is within a threshold. The VOT challenge is one of the most influential and largest annual events in the tracking field. On this benchmark, the expected average overlap (EAO) is exploited to quantitatively analyze the tracking performance.

B. Ablation Studies

Evaluations on Individual Components. The tracking performance comparisons with some baselines and our tracker's variants, e.g., DCF (linear correlation filter version of [7]), SAMF [30], DSST [11], DCFNet-1s, DCF+VGG, DCF+SiamFC, DCFNet, and SPCNet are shown in TABLE I. DCFNet-1s is a variant of SPCNet without considering scale factors in both training and tracking process. DCFNet enhances DCFNet-1s with scale estimation at 3 adjacent scale levels only in the tracking process. Compared to the traditional CF based tracker DCF using hand-crafted features, our variant DCFNet-1s using the self-learnt features and linear correlation filters obtains a DP gain of 6.3$\sim$7.9% and an OP gain of 6.1$\sim$9.9%, while retaining real-time tracking speed. By extending the search space from a single position space to the joint scale-position space, DCFNet significantly boosts the tracking performance and outperform the traditional multi-scale trackers SAMF and DSST by a large margin. DCF+VGG and DCF+SiamFC use the original feature extraction in VGG [22] and SiamFC [19] to replace the feature extraction in DCFNet. Compared with them, our specifically learnt feature representation in DCFNet leads to superior tracking performance by a notable margin. Thus, the feature representation learnt from the end-to-end network pre-training is effective for tracking. Our complete version SPCNet carries out appearance modeling in the joint scale-position space both in the training and tracking process and ranks first using all the precision metrics.

Evaluations of Run Times. To highlight the trade-off which we can make between the tracking accuracy and speed, we additionally compared two kinds of different settings of the proposed SPCNet model: one kind is pre-trained using 4 different input spatial resolutions and the other is pre-trained using 5 different numbers of final output feature channels. The tracking AUC accuracy and speed analyses of these SPCNet family and some other real-time trackers are reported in Fig. 3. From our experiments, we observe that decreasing input spatial resolution can cause a large reduction in the AUC accuracy, although it can provide a significant speedup. The AUC performance of a small number of output feature channels only falls by 4% while the run time cost is reduced by a factor of 3. Compared with recent fast trackers [7], [31], [14], [32], [33], [19], [20], [34], we find that the proposed SPCNet family achieve better performance both in accuracy and speed. According to the different computational resources available, a member in the SPCNet family with tracking speeds ranging from 60 FPS to 190 FPS can be used.

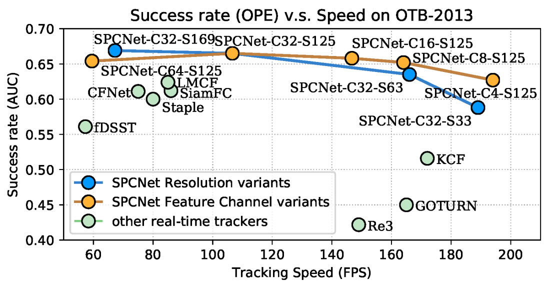
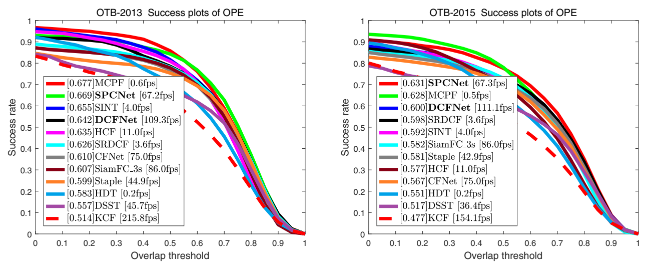


Fig. 3: The tracking speed and AUC performance on OTB-2013. The tracking speed ranges from 60 FPS to 190 FPS with different input spatial resolutions and different number of the output feature channels. SCPNet-CX-SY stands for a member in the SPCNet family consisting of X output feature channels and $Y \times Y$ input resolution.

C. State-of-the-Art Comparison

Comparison on OTB. Fig. 4 shows the results of SPCNet on OTB-2013 and OTB-2015. From Fig. 4(a) and Fig. 4(b) we find that, our automatic feature learning and appearance correlation modeling in the joint scale-position space are effective and lead to AUC gains of more than 11% in the success plots of OPE on OTB-2013 and OTB-2015 compared with KCF and DSST that exploit HOG features and do not consider the scale factor in the training process. Although our feature learning network only contains two convolutional layers and is much shallower than [13], [18], we can achieve superior performance with much faster speed. Furthermore, our algorithm is orders of magnitude faster (100$\times$) than the recent top ranked CF based tracker MCPF [35], while achieving a comparable performance. With a more adaptive online update strategy, the proposed method works better than the recent SINT [24] and SiamFC [19]. Compared to CFNet [20] which is end-to-end pre-trained in the position space, SPCNet achieves an AUC gain of more than 6% because it is learnt end-to-end in the joint scale-position space and a more appropriate regression loss is used.

Comparison on VOT. SPCNet is compared with the 62 participating trackers in the VOT2015 challenge as shown in Fig. 5. The horizontal dashed line is the VOT2015 state-of-the-art bound. SPCNet is ranked within the Top-10 trackers in the overall performance evaluation based on the expected average overlap (EAO) measure, while compared to the rest of the Top-10 trackers, it has the fastest tracking speed based on the speed evaluation unit called equivalent filter operations (EFO). Compared to CF based trackers such as DeepSRDCF [17], MUSTer [37] and KCF [7], we find that our SPCNet can achieve an excellent balance between performance and speed.



(a) OPE on OTB-2013 (b) OPE on OTB-2015

Fig. 4: Success plots on OTB-2013, OTB-2015 compared with CF based trackers: KCF [7], DSST [11], HCF [13], SRDCF [12], HDT [18], CFNet [20], MCPF [35] and others: TGPR [36], Staple [14], SINT [24], SiamFC [19].

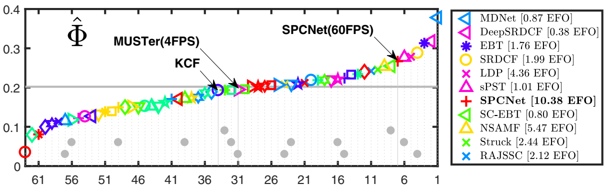


Fig. 5: An illustration of the expected average overlap plot on the VOT2015 challenge. The Top-11 trackers are listed in the legend and their tracking speeds are shown in EFO values.

IV. Conclusion

A SPCNet tracker is proposed to unify the feature representation learning and CF based appearance modeling within an end-to-end learnable framework. This framework is based on a joint scale-position space, which enables feature learning in order to obtain accurate predictions of object scale and position. SPCNet is quite efficient benefiting from the lightweight feature learning network and the Fourier frequency domain based fast correlation modeling in the correlation filter layer. Evaluations on several benchmarks demonstrate that our tracker is faster, stronger and much more compact than several state-of-the-art deep learning based trackers.

V. Acknowledgements

This work is supported by the Natural Science Foundation of China (Grant No. 61672519, 61751212, 61472421, 61602478), the NSFC-general technology collaborative Fund for basic research (Grant No. U1636218), the Key Research Program of Frontier Sciences, CAS, Grant No. QYZDJ-SSW-JSC040, and the CAS External cooperation key project.

References

[1] C. Manresa, J. Varona, R. Mas, and F. J. Perales, "Hand tracking and gesture recognition for human-computer interaction," *ELCVIA*, 2005.
[2] J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *TITS*, 2006.
[3] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *TPAMI*, 2015.
[4] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernández, T. Vojir, G. Hager, G. Nebehay, and R. Pflugfelder, "The visual object tracking vot2015 challenge results," in *ICCV Workshop*, 2015.
[5] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *TPAMI*, 2012.
[6] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. L. Hicks, and P. H. Torr, "Struck: Structured output tracking with kernels," *TPAMI*, 2016.
[7] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *TPAMI*, 2015.
[8] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *CVPR*, 2016.
[9] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, "Eco: Efficient convolution operators for tracking," in *CVPR*, 2017.
[10] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *CVPR*, 2010.
[11] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *BMVC*, 2014.
[12] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *ICCV*, 2015.
[13] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *ICCV*, 2015.
[14] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: Complementary learners for real-time tracking," in *CVPR*, 2016.
[15] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *ECCV*, 2012.
[16] M. Danelljan, F. Shahbaz Khan, M. Felsberg, and J. van de Weijer, "Adaptive color attributes for real-time visual tracking," in *CVPR*, 2014.
[17] M. Danelljan, G. Häger, F. Shahbaz Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *ICCV Workshop*, 2015.
[18] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang, "Hedged deep tracking," in *CVPR*, 2016.
[19] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *ECCV Workshop*, 2016.
[20] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in *CVPR*, 2017.
[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.
[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556*, 2014.
[23] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.
[24] R. Tao, E. Gavves, and A. W. Smeulders, "Siamese instance search for tracking," in *CVPR*, 2016.
[25] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *IJCV*, 2015.
[26] C. Boeddeker, P. Hanebrink, L. Drude, J. Heymann, and R. Haeb-Umbach, "On the computation of complex-valued gradients with application to statistically optimum beamforming," *arXiv:1701.00392*, 2017.
[27] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *CVPR*, 2013.
[28] M. Zhang, J. Xing, J. Gao, and W. Hu, "Robust visual tracking using joint scale-spatial correlation filters," in *ICIP*, 2015.
[29] A. Vedaldi and K. Lenc, "Matconvnet: Convolutional neural networks for matlab," in *ACMMM*, 2015.
[30] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration." in *ECCV Workshops*, 2014.
[31] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Discriminative scale space tracking," *TPAMI*, 2017.
[32] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *ECCV*, 2016.
[33] M. Wang, Y. Liu, and Z. Huang, "Large margin object tracking with circulant feature maps," *CVPR*, 2017.
[34] D. Gordon, A. Farhadi, and D. Fox, "Re3: Real-time recurrent regression networks for object tracking," *arXiv:1705.06368*, 2017.
[35] T. Zhang, C. Xu, and M.-H. Yang, "Multi-task correlation particle filter for robust object tracking," in *CVPR*, 2017.
[36] J. Gao, H. Ling, W. Hu, and J. Xing, "Transfer learning based visual tracking with gaussian processes regression," in *ECCV*, 2014.
[37] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao, "Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking," in *CVPR*, 2015.