Computational Visual Media
https://doi.org/10.1007/s41095-021-0212-1

Research Article

SiamCPN: Visual tracking with the Siamese center-prediction network

Dong Chen[1,2,4], **Fan Tang**[3], **Weiming Dong**[1,2,4] (✉), **Hanxing Yao**[4,5], **and Changsheng Xu**[1,2,4]

Abstract Object detection is widely used in object tracking; anchor-free object tracking provides an end-to-end single-object-tracking approach. In this study, we propose a new anchor-free network, the Siamese center-prediction network (SiamCPN). Given the presence of referenced object features in the initial frame, we directly predict the center point and size of the object in subsequent frames in a Siamese-structure network without the need for per-frame post-processing operations. Unlike other anchor-free tracking approaches that are based on semantic segmentation and achieve anchor-free tracking by pixel-level prediction, SiamCPN directly obtains all information required for tracking, greatly simplifying the model. A center-prediction sub-network is applied to multiple stages of the backbone to adaptively learn from the experience of different branches of the Siamese net. The model can accurately predict object location, implement appropriate corrections, and regress the size of the target bounding box. Compared to other leading Siamese networks, SiamCPN is simpler, faster, and more efficient as it uses fewer hyperparameters. Experiments demonstrate that our method outperforms other leading Siamese networks on GOT-10K and UAV123 benchmarks, and is comparable to other excellent trackers on LaSOT, VOT2016, and OTB-100 while improving inference speed 1.5 to 2 times.

Keywords Siamese network; single object tracking; anchor-free; center point detection

1 Introduction

Single-object tracking is a fundamental problem in visual media processing. It is widely used in applications requiring location and appearance characteristics (shape, color, etc.) of targets, such as interactive visual media editing, intelligent monitoring, human–computer interaction, augmented reality, etc. In general, single-object tracking aims to find a target, marked in the first frame, in subsequent frames of a video or image sequence. By modeling the appearance and movement of the target, the tracker can predict its motion to estimate the position of the target. In particular, such a tracker can track any object without specifying the object's category by learning essential information related to the target, such as its appearance and spatial extent. However, widespread interfering factors, such as strong illumination changes, severe deformation of non-rigid objects, similar backgrounds, and occlusion, bring considerable challenges to this task.

Despite these difficulties, many excellent visual object tracking algorithms [1–4] have emerged. Among them, tracking by Siamese networks has attracted much attention in recent years [5–8]. A Siamese network of shared parameters receives two inputs for feature extraction: one branch marks out the template target region, while the other branch is used for search. After performing feature extraction through the deep backbone network, the task of finding the target object becomes one of

1 School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100040, China. E-mail: D. Chen, chendong2018@ia.ac.cn; W. Dong, weiming.dong@ia.ac.cn (✉); C. Xu, changsheng.xu@ia.ac.cn.
2 NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China.
3 School of Artificial Intelligence, Jilin University, Changchun 130012, China. E-mail: tangfan@jlu.edu.cn.
4 CASIA-LLVISION Joint Lab, Beijing 100190, China.
5 LLVISION Technology Co., LTD., Beijing 100190, China. E-mail: yaohx@llvision.com.
Manuscript received: 2021-01-10; accepted: 2021-02-04

calculating the similarity of the two output feature maps. Usually, cross correlation is used to do so. The features after cross-correlation generate a fixed-size response map whose peak is regarded the position of the target object. SINT [8] and SiamFC [5] first used this approach to solve the single-object tracking problem. SiamRPN [7] improved the performance of SiamFC [5] by introducing a region proposal network [9]. Using the Siamese network structure, foreground–background classification and bounding box regression can also be performed on the proposed region, which can effectively improve the accuracy of the predicted bounding box, avoid the multiscale test in SiamFC [5], and achieve state-of-the-art performance on multiple benchmarks. In later research, SiamRPN++[6], DaSiamRPN [10], and SiamDW [11] improved tracking performance via the backbone network structure, residual block structure, sampling strategy, and in other ways. However, all of these approaches had relied on a predefined configuration of anchors. RPN-based models use multi-channel response maps to detect and regress region proposals, in which the number of channels in the output response map depends on the pre-configured anchors.

Furthermore, the existence of an anchor generates a large number of redundant prediction boxes and thus requires additional post-processing procedures such as non-maximum suppression to eliminate candidate boxes to obtain the final result, which also increases the calculation. On the basis of semantic segmentation theory, some researchers have recently improved these defects via pixel-level prediction, and perform object tracking in an anchor-free manner [12–14]. FCAF [14] suggested using an anchor-free proposal network (AFPN) to replace the region proposal network. The AFPN network consisted of a correlation section and a supervised section with two branches, one for classification and the other for regression. To suppress prediction of low-quality bounding boxes, a centricity branch was added, similar to that in SiamCAR [12]. However, as SiamCAR performs classification at the pixel level, mapping the predicted position back to the original image may cause deviations that can result in jitter during tracking. Therefore, after obtaining the prediction results of multiple adjacent pixels in the target area and upsampling the response map, the prediction results of multiple adjacent points are weighted and averaged to give the final target box. However, this post-processing procedure increases the computational burden during tracking. Moreover, although anchor-free approaches can simplify the region proposal module used in anchor-based trackers, post-processing is still needed because the outputs of the networks are based on semantic segmentation form.

As an alternative to the above methods, we propose a Siamese center prediction network (SiamCPN) based on keypoint detection by predicting the position and size of the target region in a "real" end-to-end manner. It uses a multi-channel heatmap in which one channel is used to predict the target position while the other two channels are used to adjust the center offset and regress the object size. In this manner, all of the information required for tracking can be directly obtained without any post-processing, thus greatly simplifying the model. A center-prediction sub-network (CPN) is applied to multiple stages of the backbone as a means of adaptively correlating the feature maps from the Siamese network. The outputs of SiamCPN are the directly predicted objects; no post-processing procedure is needed. Our main contributions are as follows:

- SiamCPN, a network for single-object tracking that can be implemented in a simple, true end-to-end manner. A few channels of the response maps are learned to directly predict the center and size of the target region, thus achieving anchor-free tracking.
- A CPN to adaptively correlate multistage outputs from the backbone.
- A demonstration that SiamCPN has superior performance on multiple datasets and is competitive in terms of inferencing speed to other methods selected in this work.

2 Related work

This section mainly focuses on tracking approaches based on Siamese networks.

Tao et al. [8] proposed SINT and pointed out that the object-tracking problem could be converted into a matching process between a template frame and other frames. By using a Siamese network that could accept

two inputs at the same time, SINT learnt a matching function between different regions in the two input frames. After obtaining target information from the first frame, all following frames could be fed into the network to calculate their similarity with the target in the first frame. However, this method required inputs to generate several region proposals in the image before passing data through the network, which was time-consuming. Bertinetto et al. [5] proposed SiamFC, further defining the tracking problem as a similarity learning problem, thereby obtaining a single-channel score map for object detection. SiamFC [5] quickly gained researchers' attention owing to its simple architecture, high accuracy, and high speed; it only requires offline training without online fine tuning.

Following these initial approaches, functional modules from related research have been applied to visual tracking by Siamese networks [6, 7, 10–12, 15–17]. Li et al. [7], who proposed SiamRPN, combined the RPN network from Faster R-CNN [9] with the Siamese network. SiamRPN replaced multiscale detection in SiamFC by means of bounding box regression, improving inference speed and accuracy. SiamRPN also adopted the idea of one-shot learning. During tracking, the template patch in the first frame could be fed into the template branch as the detection kernel and then used to perform a cross-correlation operation with the features of the search region in subsequent frames for tracking. Wang et al. [15] proposed the SiamMask network that could simultaneously perform object tracking and segmentation based on a Siamese network by adding a mask branch for heatmap prediction to achiev object segmentation. Zhu et al. [10] argued that methods based on a Siamese network could only distinguish the target and the background when no semantic relationship exists. When similar-looking backgrounds and objects occur, the setup usually does not work well. Furthermore, a tracker based on a Siamese network cannot update a model online during the tracking stage, which can lead to accuracy loss. In addition, certain trackers cannot deal with the challenges of occlusion and target drawing in scenes during long-term tracking. In response to the above three problems, Zhu et al. [10] introduced DASiamRPN with high-quality training data for training. Existing datasets were used for object detection to enrich positive samples and difficult negative samples to improve the generalization and discrimination abilities of the tracker, respectively. A perception module was also introduced to improve the choice criterion for the optimal boundary.

When researchers replaced AlexNet [18] with a deeper convolution network for feature extraction based on the Siamese network structure, they discovered the problem of location bias [6, 11], suggesting that the earlier works like SiamFC and SiamRPN could only use shallow networks for feature extraction. Zhang and Peng [11] analyzed the three factors of stride size, padding, and receptive field in convolutional networks. After several experiments, they found that the existence of padding in a deep network would cause tracking position deviation, and thus, that stride should be made as small as possible (8 is recommended). Furthermore, the size of the receptive field and the output stride should be considered at the same time. On the basis of such observations, Zhang and Peng proposed SiamDW and adopted a new residual module to reduce the impact of padding. Li et al. [6] also explored the abovementioned problems and argued that a Siamese network could not use a deep network structure because of its lack of strict translational invariance; moreover, padding could destroy translation invariance. The sampling strategy was improved by transforming the original fixed position sampling to uniform sampling near the center. They trained a Siamese network tracker using ResNet [19] as the backbone network. Compared with previous work, the performance of the tracker was notably improved. Apart from maintaining real-time performance (35 frames per second), SiamRPN++ [6] achieved excellent scores in terms of expected average overlap rate, robustness, and accuracy.

However, anchor-based methods not only require several experiments to determine suitable hyperparameters but also need tedious post-processing operations. Recently, some works based on semantic segmentation have achieved pixel-by-pixel object tracking in an anchor-free and proposal-free manner [12–14]. On the basis of keypoint detection theory [20–24], only the center point of the bounding box and other information are used to predict and correct the position and size of the target. This approach allows our SiamCPN

to operate faster and perform better while using the same feature extraction strategy as Refs. [6, 12]. Furthermore, our proposed method is more concise and effective in exploring an advanced and convenient object-tracking solution than other methods.

3 Method

3.1 Overview

The overall structure of our SiamCPN is shown in Fig. 1. Features are extracted by the Siamese fully convolutional backbone. Multiple CPNs are used to measure the similarity of the outputs from different stages of the Siamese feature extraction backbone. The final result is obtained by enhancing the average weighted outputs of these multi-CPN modules. In this section, we discuss the overall structure of the proposed SiamCPN (Section 3.2) and then explore the CPN (Section 3.3) and the loss functions for training the SiamCPN (Section 3.4).

3.2 Siamese center prediction network

In SiamCPN, a modified ResNet-50 is used as the backbone to build a fully convolutional network for feature extraction. The stride of the network is reduced and its receptive field is increased simultaneously via dilated convolution to ensure the spatial consistency of *conv*4 and *conv*5.

Tracking algorithms based on a Siamese network usually obtain input from two branches called the template branch and the search branch. As shown in Fig. 1, the network branch input by template Z is a template branch, and the network branch corresponding to another input X is a search branch. The template branch takes a specified template patch $\mathbf{Z}$ in the first frame as the input, whereas the search branch takes the search region $\mathbf{X}$ as the input. These two inputs are fed into a shared-parameter CNN to generate output feature maps $\varphi(Z)$ and $\varphi(X)$. Then, the similarity response of the two different output feature maps $\varphi(Z)$ and $\varphi(X)$ is calculated by cross-correlation. Finally, the output response map passes through the CPN head to generate multiple response maps, given by

$$H = \mathrm{CPN}(\mathcal{F}_b^*(X), \mathcal{F}_b^*(Z)) \tag{1}$$

where CPN denotes the center-prediction sub-network. The CPN calculates the cross correlation

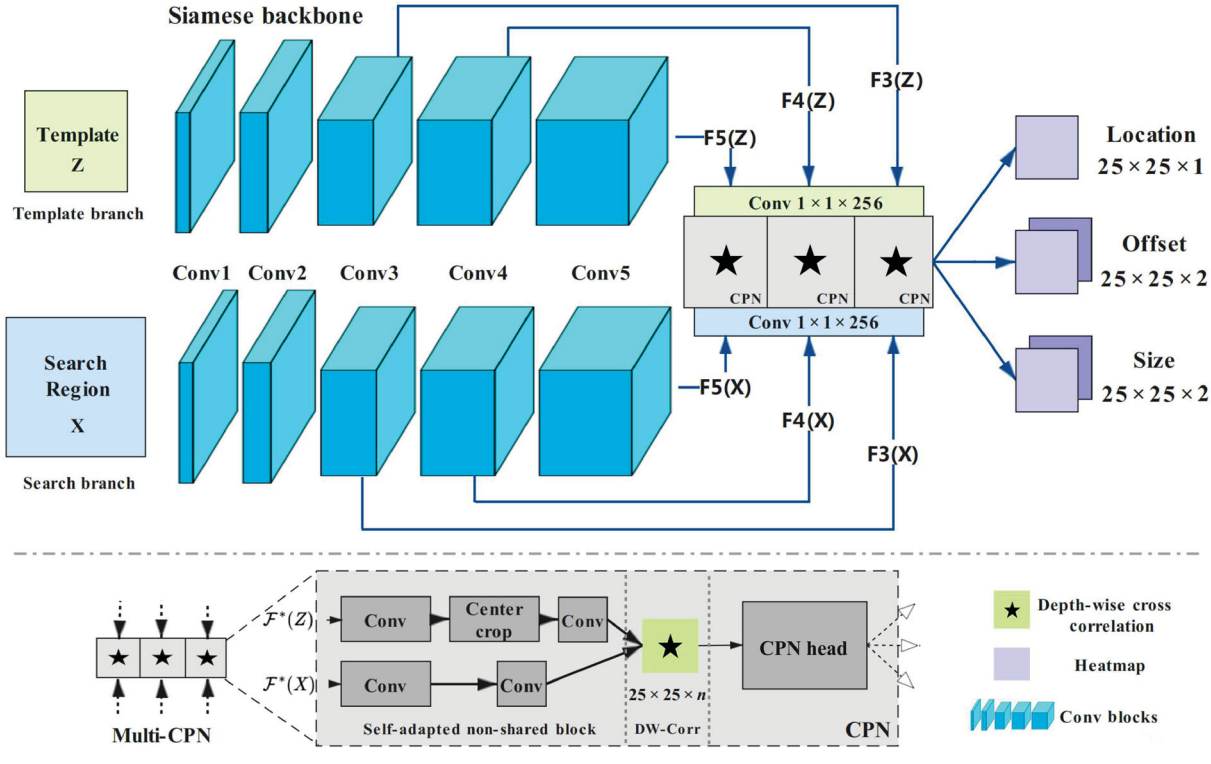


Fig. 1 SiamCPN. Above: the entire framework. Blue boxes: conv-blocks in ResNet-50. The output of Conv3, Conv4, and Conv5 are fed into CPN. Gray block: CPN module. ⋆ denotes depth-wise cross-correlation. Our network produces three 25 × 25 outputs. Below: architecture of the CPN module. Before input to the CPN, channels are reduced to the same size. The self-adapted block forms the first part of the CPN module. Only the structure of a single output branch is shown.

between the channel-aligned feature $F_b^*(X)$ and $F_b^*(Z)$, which come from block b of the backbone network. The CPN adaptively generates a single-channel or multichannel heatmap H.

Low-level features better representing visual attributes (such as edges, corners, colors, and shapes) are essential in predicting object positions, whereas high-level features can better represent semantic attributes that are essential for making a distinction. Therefore, we also consider the use of multistage features for tracking. Here, we use features extracted from the last three residual blocks of the backbone, denoted $\mathcal{F}_3(*), \mathcal{F}_4(*), \mathcal{F}_5(*)$, where $*$ represents the template patch $\mathbf{Z}$ or search region $\mathbf{X}$. Before cross-correlation, the channel sizes of $\mathcal{F}_3(*), \mathcal{F}_4(*), \mathcal{F}_5(*)$ should be unified (to 256 in our experiments). Thus, a convolutional layer, with kernel size 1×1 for adjustment, is appended to these three blocks. As shown in Fig. 1, the unified features $\mathcal{F}_3^*(*), \mathcal{F}_4(*), \mathcal{F}_5^*(*)$, generated by block3, block4, and block5, respectively, are adopted as the inputs to the multiple CPN module.

The main output of our approach is a heatmap $\hat{Y} \in [0,1]^{w/r \times h/r \times 1}$, where w and h are the width and height of $\mathbf{X}$, and r is the output stride. We set $w = h = 255$. When $\hat{Y}_{x,y} = 1$, the corresponding position (x, y) is regarded as the detected center point position; otherwise, it is the background. In addition, to correct positional deviation due to the span of the network during the learning process, we predict the center offset to more accurately regress the position.

3.3 Center prediction sub-network

Given the unified feature maps $\mathcal{F}_b^*(X)$ and $\mathcal{F}_b^*(Z)$ from the two branches, the CPN adaptively calculates the cross correlation and outputs a heatmap of the center, corresponding offset, and size of the object. A self-adapted block, depth-wise cross correlation, and a prediction head are used in the proposed CPN.

3.3.1 Self-adapted block

To effectively fuse features from two branches for the final prediction, we propose a self-adapted block whose parameters are not shared to solve the varying problem in each prediction branch. In particular, features from the template and search branches are first passed through different convolutional layers. Then, the center region is cropped from the feature of the template branch to reduce the computational burden on the cross-correlation operation. The cropped center size is set to 7 to preserve accurate information about the object. Then, the template branch is passed through a group convolutional layer, for computational reasons. Unlike the template branch, the search branch only needs to append another group convolutional layer. In general, the self-adapted block allows the modules in each branch to acquire enough meaningful knowledge during training to improve prediction. Figure 1(below) shows details of the sampled block. We only show part of a prediction branch (location, offset, or size) for a given CPN module. Three similar parts are used to obtain the different CPN outputs.

3.3.2 Depth-wise correlation

Cross-correlation is the core operation during tracking, and the goal is to determine the most similar patches from the search region in the semantic embedding space.

$$R = \mathcal{F}_b^*(X) \star \mathcal{F}_b^*(Z) \tag{2}$$

where * denotes depth-wise correlation, which is used to generate the multichannel response map R. To efficiently achieve information association, we use depth-wise cross correlation to embed the information from the two branches. Then, the calculation is performed in a channel-by-channel manner. Each channel of R represents different meaningful semantic information, which can then be used to predict the target-related attributes. The CPN head takes R into a convolutional layer with normalization and outputs three 25×25 heatmaps with one, two, and two channels, respectively.

3.4 Objective

As the desired output is in the form of a heatmap, the ground truth is built in the same format. First, we generate the corresponding center coordinates $p = ((x_1 + x_2)/2, (y_1 + y_2)/2)$ in the original image. Then, we obtain the corresponding center coordinates $\tilde{p} = \lfloor p/R \rfloor$ in the downsampled feature map. Finally, the keypoints in the feature map are distributed in the form of Gaussian kernels for the labeled bounding box:

$$Y_{xy} = \exp\left(-\frac{(x - \tilde{p}_x)^2 + (y - \tilde{p}_y)^2}{2\sigma_p^2}\right) \tag{3}$$

where σ_p is a standard deviation related to target size, a scheme similar to that in Ref. [24]. By overlaying the Gaussian distribution on the heatmap, Gaussian keypoints can be continuously added based on the

heatmap. The training objective is a penalty-reduced pixel-wise logistic regression with focal loss [25]:

$$L_c = \frac{-1}{N}\sum_{xy}\begin{cases}\left(1-\hat{Y}_{xy}\right)^{\alpha}\log\left(\hat{Y}_{xy}\right), & \text{if } Y_{xy}=1\\ \left(1-Y_{xy}\right)^{\beta}\left(\hat{Y}_{xy}\right)^{\alpha}\log\left(1-\hat{Y}_{xy}\right), & \text{otherwise}\end{cases} \tag{4}$$

where N is the number of keypoints in the search region. Following the scheme in Ref. [24], we set $\alpha = 2$ and $\beta = 4$ in the experiments.

To reduce the impact of position shift caused by downsampling, an offset branch is added to predict the deviation of the center point; we use L_1 loss for training:

$$L_{\text{off}} = \frac{1}{N}\sum_{p}\left|\hat{O}_{\tilde{p}} - \left(\frac{p}{r}-\tilde{p}\right)\right| \tag{5}$$

which only works at the location of the center point predicted by the heatmap, whereas all other places are ignored. The output $\hat{O}_{\tilde{p}} \in \mathcal{R}^{w/r\times h/r\times 2}$ contains two channels for offsets in the w and h directions, respectively.

A prediction of the relevant attributes of the target center is insufficient during tracking, and target size information also needs to be obtained. After estimating the location of the center using a heatmap, we directly regress the width and height of the object by using the L_1 loss at the center as follows:

$$L_{wh} = \frac{1}{N}\sum_{k=1}^{N}\left|\hat{S}_{p_k} - s_k\right| \tag{6}$$

where $\hat{S} \in \mathcal{R}^{w/r\times h/r\times 2}$ contains two channels for width and height of the object.

The overall training objective is expressed as

$$L = L_c + \lambda_{\text{off}}L_{\text{off}} + \lambda_{wh}L_{wh} \tag{7}$$

where the constants λ_{off} and λ_{wh} weight the offset loss and size loss, respectively. During training, we set $\lambda_{\text{off}} = 1$ and $\lambda_{wh} = 0.1$ for the experiments.

Finally, the average of the outputs of the three CPN modules is calculated from multiple stages as the overall prediction. Thus, SiamCPN decomposes the tracking problem into three subproblems: determining the location of the object center, predicting the center offset, and regression of object size. Combining these multilevel features enhances the capabilities of the CPN module and allows it to obtain good predictions.

4 Experiments

4.1 Implementation

SiamCPN was implemented in Python with PyTorch and trained on 4 TITAN X GPUs. To enable a fair comparison, the input sizes of the template patch and search regions were set in the same manner as in Refs. [6] and [7], i.e., 127×127 and 255×255 pixels. The backbone was pretrained on ImageNet [26].

4.2 Training

We conducted training using six large datasets: GOT-10K [27], LaSOT [28], COCO [29], DET [26], VID [26], and YouTube-BB [30]. During training process, we set the batch size to 32 and used stochastic gradient descent for optimization. In general, SiamCPN was trained for 20 epochs. The first 10 epochs are only used for preliminary training and excluded the last three blocks for the backbone network. The last three blocks of ResNet-50 were added for training in the remaining epochs. To ensure a fair comparison, training and evaluation on GOT-10K [27] and LaSOT [28] were conducted separately in accordance with SiamCAR [12], whereas training on the other four datasets was conducted for the evaluations on OTB [31, 32], VOT2016 [33], and UAV123 [34].

4.3 Testing

We implemented an offline tracking strategy for testing. The template branch is only computed once in the first frame and then fixed over the whole tracking period. As a result, the object in the first frame is adopted as the template patch for tracking, and the current frame is adopted as the search region that is fed into the backbone network. In general, the purpose of the inference process is to extract the required bounding box from the generated heatmap. Therefore, after the input of the two branches is passed through the SiamCPN, the position of the peak in the heatmap is considered to be the location of the object. Then, we adjust the position of the center point by using the predicted offset and determine the final box by referring to the center point and predicted object size. For evaluations on different datasets, a comparison with authors' own measurements was conducted. As availability of data for different methods varies, comparison with different state-of-the-art methods is conducted using different benchmarks.

4.4 Comparison with state-of-the-art

In our experiments, we found the proposed method to be faster than other methods and easier to train, test, and deploy. It can be adapted without introducing additional hyperparameters. After training the model, it was tested directly on different benchmarks. SiamCPN outperforms existing methods on relevant benchmarks while maintaining its speed advantage and only needs simple test conditions.

4.4.1 Assessment using GOT-10K

The GOT-10K [27] dataset contains more than 10,000 video segments of real-world moving objects and over 1.5 million manually labeled bounding boxes. The dataset has the WordNet [45] backbone and covers a majority of the 560+ classes of real-world moving objects and 80+ classes of motion patterns. The test set embodies 84 object classes and 32 motion classes with 180 video segments, allowing for efficient evaluation. For a fair comparison, the protocol for deep trackers was used so that all approaches could use the same training data provided by the dataset. The primary evaluation indicators for GOT-10K were the average overlap (AO) and success rate (SR). AO represents the average overlap of all estimated boxes and ground-truth boxes. SR includes $SR_{0.5}$ and $SR_{0.75}$, which represents the rate of successfully tracked frames whose overlap exceeds 0.5 and 0.75, respectively.

A comparison was conducted to the baselines provided by the GOT-10K website, including to Siamese-based approaches, such as SiamRPN++ [6] and SiamFC [5]. To show the effectiveness of the proposed CPN based on the anchor-free strategy, a comparison was also made to other three tracking methods [12, 13, 16], selected on the basis of their anchor-free tracking strategies. using released models and code. As Fig. 2 shows, SiamCPN outperforms the other trackers. Table 1 gives detailed results using different metrics. AP and SR for OCEAN [13] are much lower than those listed in Ref. [13], perhaps due to choice of unpublished hyper-parameters (e.g., window penalty) that need careful selection for each test set. By contrast, when testing our model with a specific test set, fine tuning of parameters is not required: our method uses fewer hyperparameters and is more convenient to use than the other methods investigated in this study.

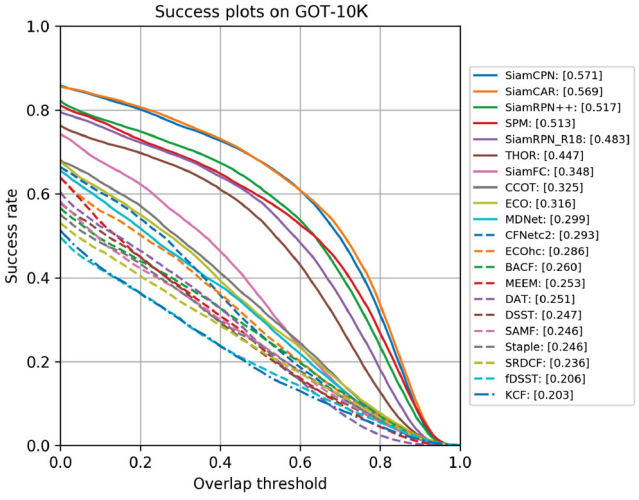


Fig. 2 Comparisons using GOT-10K [27]. SiamCPN outperforms Siamese-based and other baseline methods.

Table 1 Evaluation on GOT-10K. Top-2 results are shown in bold; * means the method adopts an anchor-free tracking strategy

Tracker	*AO*	$SR_{0.5}$	$SR_{0.75}$
KCF [35]	0.203	0.177	0.065
fDSST [36]	0.206	0.187	0.075
SRDCF [37]	0.236	0.227	0.094
Staple [38]	0.246	0.239	0.089
CFnet [39]	0.293	0.265	0.087
MDnet [40]	0.299	0.303	0.099
ECO [41]	0.316	0.309	0.111
CCOT [42]	0.325	0.328	0.107
SiamFC [5]	0.374	0.404	0.144
SPM [43]	0.513	0.593	0.359
SiamRPN++ [6]	0.517	0.616	0.325
*OCEAN [13]	0.520	0.614	0.329
*SiamFC++ [16]	0.529	0.617	0.381
ATOM [44]	0.556	0.634	0.402
*SiamCAR [12]	**0.569**	**0.670**	**0.415**
Ours	**0.571**	**0.678**	**0.414**

Inferencing speed is an important factor in assessing model performance. Table 2 shows tracking frame rates in fps, for different approaches: the following four approaches were tested under the same conditions (using a Titan X GPU): SiamFC [5], SiamRPN++ [6], SiamCAR [12], and our method. SiamRPN++ and SiamCAR were selected as they adopt the same feature extraction strategy as SiamCPN; SiamFC uses a shallow backbone network [18] for feature extraction and is commonly considered a fast single-object tracker. The inferencing speeds of SiamRPN++ and SiamCAR are 19.05 and 17.7 fps, respectively, whereas the proposed SiamCPN

Table 2 Inferencing speeds in fps of different Siamese-based trackers under the same test conditions. Using the same strategy for feature extraction, our SiamCPN is faster than SiamRPN++ and SiamCAR

Tracker	SiamFC [5]	SiamRPN++ [6]	SiamCAR [12]	Ours
fps	25.81	19.05	17.7	33.79

reaches 33.79 fps. Using the same strategy for feature extraction with the same hardware, our method is faster than SiamCAR and SiamRPN++. Furthermore, SiamCPN is faster than SiamFC. These findings indicate that the proposed CPN has excellent performance with regard to inferencing speed.

4.4.2 Assessment using LaSOT

The LaSOT [28] dataset contains more than 3.52 million frames of hand-labeled pictures and 1400 videos, and is by far the largest single-target tracking dataset with dense labeling. On average each LaSOT sequence has 2512 frames, all carefully checked and manually marked. Thus, approximately 3.52 million high-quality bounding box annotations can be generated. LaSOT contains 70 categories, each having 20 sequences. To assess existing trackers and provide a broad benchmark for future comparisons using LaSOT, 35 representative trackers were evaluated under different protocols, and their performances are analyzed using different metrics.

Figure 3 shows the success and precision plots using LaSOT. A comparison was conducted with the top 15 trackers, including SiamRPN++ [6], MDNet [40], DSiam [46], and others. The results obtained are comparable with those from SiamRPN++ but performs better than those for the other baseline methods. The ability of our model to outperform most selected methods using a large-scale dataset shows that our method is feasible and effective.

4.4.3 Assessment using VOT2016

The VOT2016 [33] dataset includes 60 video sequences with different challenging factors for evaluating tracking performance. It also includes two basic evaluation indicators (accuracy rate and robustness) and combines them into EAO (expected average overlap) as the overall performance evaluation metric. The accuracy rate corresponds to the AO rate under successful tracking, while robustness is measured on the basis of the total number of tracking failures. To test the effectiveness and stability of our proposed anchor-free strategy, we set up comparative experiments integrating different trackers, including FCAF [14] based on an anchor-free strategy and use of semantic segmentation for object tracking. As shown in Table 3, our model outperforms the other methods for all metrics selected in this study.

4.4.4 Assessment using OTB-100

The OTB-100 [32] dataset was developed from OTB-50 [31] dataset, which consists of 50 fully annotated video sequences, containing a total of 51 objects of different sizes, and more than 29,000 frames. Each target is affected by various interfering factors during

Table 3 Tracker comparison using VOT2016. Top-2 results are shown in bold. Our method outperforms the other trackers. EAO and accuracy outperform FCAF by 0.7% and 0.8% respectively. The robustness result indicates that our model has good stability during tracking

Tracker	EAO	Accuracy	Robustness
Ours	**0.363**	**0.589**	**0.56**
FCAF [14]	**0.356**	**0.581**	1.02
SiamRPN [7]	0.344	0.560	1.08
CCOT [42]	0.331	0.539	0.85
MLDF [33]	0.311	0.490	**0.83**
Staple [38]	0.295	0.544	1.35

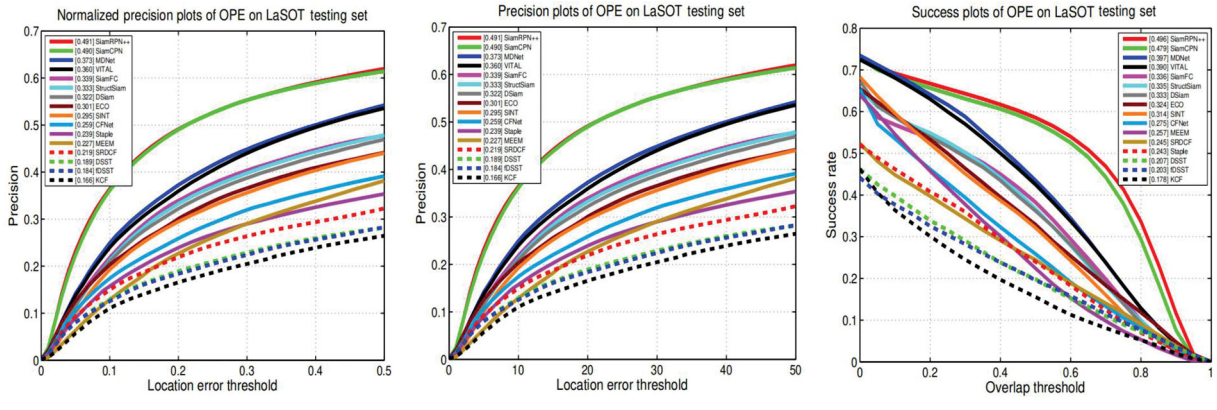


Fig. 3 Comparisons with the top-15 trackers on LaSOT [28]. Our model is comparable to SiamRPN++ [6] and outperforms the other baselines.

the tracking process. To fully evaluate the robustness of tracking algorithms with respect to various factors that may affect tracking, OTB50 was used to provide 11 common video attribute annotations: illumination changes, scale changes, occlusion, deformation, motion blur, fast motion, in-plane rotation, out-of-plane rotation, out of view, background interference, and low resolution. Each video frame contains at least two attribute annotations. In addition, the OTB-50 dataset integrates 29 popular tracking algorithms and unifies input and output formats to facilitate algorithm performance evaluation. In 2015, the OTB-50 dataset was further expanded to the OTB-100 dataset with 100 labeled video sequences. We compare our method with the top 9 baselines, including MUSTer [47], MEEM [48], STRUCK [49], and other methods whose tracking results are provided by the OTB website. As Figs. 4 and 5 show, our SiamCPN outperforms all other methods in terms of both metrics.

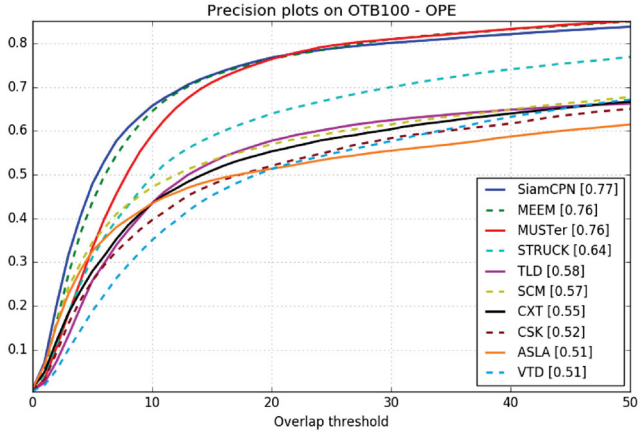


Fig. 4 Precision evaluation on OTB-100 [32]. Our approach is superior to the comparators.

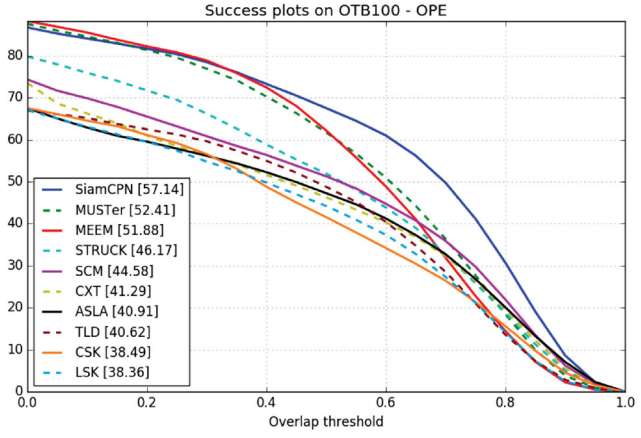


Fig. 5 Success evaluation on OTB-100 [32]. Our approach is superior to the comparators.

4.4.5 Assessment using UAV123

The UAV123 [34] dataset contains a total of 123 video sequences and more than 110k frames. All sequences are fully annotated with upright bounding boxes. We compared our method to 14 baselines provided by the UAV123 website, including MUSTer [47], SRDCF [37], MEEM [48], and other approaches. Success and precision of OPE were used to evaluate the overall performance in this study. As shown in Figs. 6 and 7, our SiamCPN outperforms all other trackers on both metrics. In addition, as shown in Table 4, SiamCPN provides the best results while using a much simpler network than state-of-the-art RPN-based trackers, and it does not require heuristic parameter tuning.

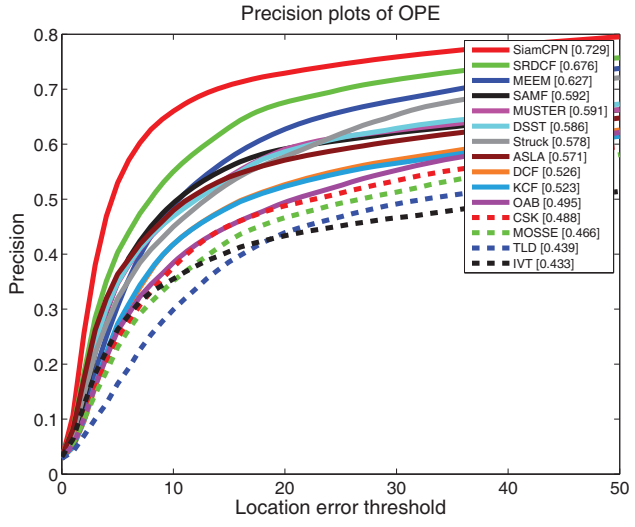


Fig. 6 Precision evaluation on UAV123 [34]. SiamCPN significantly outperforms the baseline and state-of-the-art approaches.

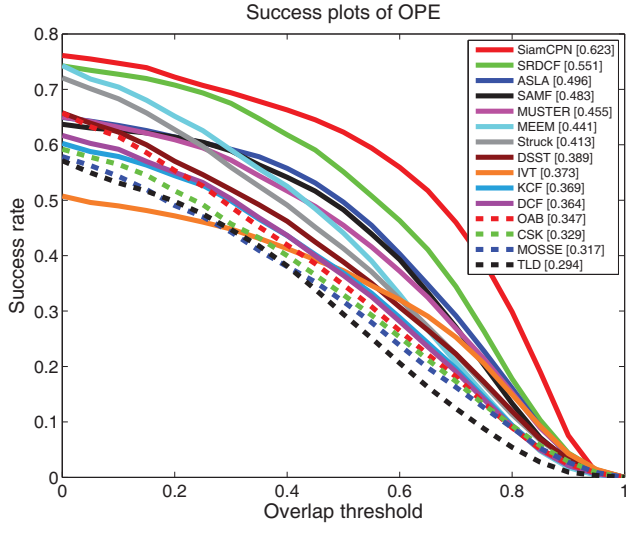


Fig. 7 Success evaluation on UAV123 [34]. Our method is more accurate than the baseline and state-of-the-art approaches.

Table 4 Comparison to Siamese-based trackers on UAV123. Top-2 results are shown in bold

Tracker	SiamFC [5]	SiamRPN [7]	DaSiamRPN [10]	SiamRPN++ [6]	SiamCAR [12]	Ours
Success	0.485	0.557	0.569	0.610	**0.614**	**0.623**

5 Conclusions

In this study, we decomposed the object-tracking problem into three sub-problems to predict center position, center point offset, and object-size. Our proposed SiamCPN can be treated as an encoding–decoding framework. By ensuring feature extraction and correlation calculation in CPN, the differences of the two input frames can be encoded into the response maps. The CPN head can also decode the response maps into heatmaps for visual tracking. The proposed method is simpler and faster than many other Siamese-based methods and achieves excellent performance on various large-scale datasets such as GOT-10K and LaSOT. Our research provides a new approach for Siamese networks when combined with the anchor-free detection method. In future, we will continue to explore the potential of Siamese networks in tracking. Specifically, we will focus on enriching the expressive ability of template branches by extracting more powerful features, and finding target-related information from high-level semantics. However, it is difficult to solve various challenges in real scenes by relying only on visual features. Incorporating temporal information into the model will make it more robust.

Acknowledgements

Code and experimental data are available at `https://github.com/KevinDongDong/SCPN`. Other data required for experiments (including training data, test data, etc.) are provided by websites given in references.

We thank the anonymous reviewers for their valuable comments. This work was supported by the National Key R&D Program of China (Grant No. 2018YFC0807500), and the National Natural Science Foundation of China (Grant Nos. U20B2070 and 61832016).

References

[1] Danelljan, M.; Häger, G.; Shahbaz Khan, F.; Felsberg, M. Accurate scale estimation for robust visual tracking. In: Proceedings of the British Machine Vision Conference, 2014.

[2] Henriques, J. F.; Caseiro, R.; Martins, P.; Batista, J. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 37, No. 3, 583–596, 2015.

[3] Kalal, Z.; Mikolajczyk, K.; Matas, J. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 34, No. 7, 1409–1422, 2012.

[4] Fan, R. C.; Zhang, F. L.; Zhang, M.; Martin, R. R. Robust tracking-by-detection using a selection and completion mechanism. *Computational Visual Media* Vol. 3, No. 3, 285–294, 2017.

[5] Bertinetto, L.; Valmadre, J.; Henriques, J. F.; Vedaldi, A.; Torr, P. H. S. Fully-convolutional Siamese networks for object tracking. In: *Computer Vision – ECCV 2016 Workshops. Lecture Notes in Computer Science, Vol. 9914.* Hua, G.; Jégou, H. Eds. Springer Cham, 850–865, 2016.

[6] Li, B.; Wu, W.; Wang, Q.; Zhang, F. Y.; Xing, J. L.; Yan, J. J. SiamRPN++: Evolution of Siamese visual tracking with very deep networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4277–4286, 2019.

[7] Li, B.; Yan, J. J.; Wu, W.; Zhu, Z.; Hu, X. L. High performance visual tracking with Siamese region proposal network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8971–8980, 2018.

[8] Tao, R.; Gavves, E.; Smeulders, A. W. M. Siamese instance search for tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1420–1429, 2016.

[9] Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 39, No. 6, 1137–1149, 2017.

[10] Zhu, Z.; Wang, Q.; Li, B.; Wu, W.; Yan, J. J.; Hu, W. M. Distractor-aware Siamese networks for visual object tracking. In: *Computer Vision – ECCV 2018. Lecture Notes in Computer Science, Vol. 11213.* Ferrari, V.; Hebert, M.; Sminchisescu, C.; Weiss, Y. Eds. Springer Cham, 103–119, 2018.

[11] Zhang, Z. P.; Peng, H. W. Deeper and wider Siamese networks for real-time visual tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4586–4595, 2019.

[12] Guo, D. Y.; Wang, J.; Cui, Y.; Wang, Z. H.; Chen, S. Y. SiamCAR: Siamese fully convolutional classification and regression for visual tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 6268–6276, 2020.

[13] Zhang, Z. P.; Peng, H. W.; Fu, J. L.; Li, B.; Hu, W. M. Ocean: Object-aware anchor-free tracking. In: *Computer Vision – ECCV 2020. Lecture Notes in Computer Science, Vol. 12366*. Vedaldi, A.; Bischof, H.; Brox, T.; Frahm, J. M. Eds. Springer Cham, 771–787, 2020.

[14] Han, G.; Du, H.; Liu, J. X.; Sun, N.; Li, X. F. Fully conventional anchor-free Siamese networks for object tracking. *IEEE Access* Vol. 7, 123934–123943, 2019.

[15] Wang, Q.; Zhang, L.; Bertinetto, L.; Hu, W. M.; Torr, P. H. S. Fast online object tracking and segmentation: A unifying approach. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 1328–1338, 2019.

[16] Xu, Y. D.; Wang, Z. Y.; Li, Z. X.; Yuan, Y.; Yu, G. SiamFC++: Towards robust and accurate visual tracking with target estimation guidelines. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, No. 7, 12549–12556, 2020.

[17] Peng, S. Y.; Yu, Y. X.; Wang, K.; He, L. Accurate anchor free tracking. *arXiv preprint* arXiv: 2006.07560, 2020.

[18] Krizhevsky, A.; Sutskever, I.; Hinton, G. E. ImageNet classification with deep convolutional neural networks. *Communications of the ACM* Vol. 60, No. 6, 84–90, 2017.

[19] He, K. M.; Zhang, X. Y.; Ren, S. Q.; Sun, J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770–778, 2016.

[20] Cao, Z.; Hidalgo, G.; Simon, T.; Wei, S.; Sheikh, Y. OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 43, No. 1, 172–186, 2018.

[21] Newell, A.; Yang, K. Y.; Deng, J. Stacked hourglass networks for human pose estimation. In: *Computer Vision – ECCV 2016. Lecture Notes in Computer Science, Vol. 9912*. Leibe, B.; Matas, J.; Sebe, N.; Welling, M. Eds. Springer Cham, 483–499, 2016.

[22] Papandreou, G.; Zhu, T.; Kanazawa, N.; Toshev, A.; Tompson, J.; Bregler, C.; Murphy, K. Towards accurate multi-person pose estimation in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3711–3719, 2017.

[23] Zhou, X. Y.; Wang, D. Q.; Krähenbühl, P. Objects as points. *arXiv preprint* arXiv: 1904.07850, 2019.

[24] Law, H.; Deng, J. CornerNet: Detecting objects as paired keypoints. In: *Computer Vision – ECCV 2018. Lecture Notes in Computer Science, Vol. 11218*. Ferrari, V.; Hebert, M.; Sminchisescu, C.; Weiss, Y. Eds. Springer Cham, 765–781, 2018.

[25] Lin, T. Y.; Goyal, P.; Girshick, R.; He, K. M.; Dollár, P. Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, 2999–3007, 2017.

[26] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision* Vol. 115, No. 3, 211–252, 2015.

[27] Huang, L. H.; Zhao, X.; Huang, K. Q. GOT-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, doi: 10.1109/TPAMI.2019.2957464, 2019.

[28] Fan, H.; Lin, L. T.; Yang, F.; Chu, P.; Deng, G.; Yu, S. J.; Bai, H.; Xu, Y.; Liao, C.; Ling, H. LaSOT: A high-quality benchmark for large-scale single object tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5369–5378, 2019.

[29] Lin, T. Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C. L. Microsoft COCO: Common objects in context. In: *Computer Vision – ECCV 2014. Lecture Notes in Computer Science, Vol. 8693*. Fleet, D.; Pajdla, T.; Schiele, B.; Tuytelaars, T. Eds. Springer Cham, 740–755, 2014.

[30] Real, E.; Shlens, J.; Mazzocchi, S.; Pan, X.; Vanhoucke, V. YouTube-BoundingBoxes: A large high-precision human-annotated data set for object detection in video. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 7464–7473, 2017.

[31] Wu, Y.; Lim, J.; Yang, M. H. Online object tracking: A benchmark. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2411–2418, 2013.

[32] Wu, Y.; Lim, J.; Yang, M. H. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 37, No. 9, 1834–1848, 2015.

[33] Kristan, M.; Leonardis, A.; Matas, J.; Felsberg, M.; Pflugfelder, R.; Čehovin, L.; Vojíř, T.; Häger, G.; Lukežič, A.; Fernández, G. et al. The visual object tracking VOT2016 challenge results. In: *Computer Vision – ECCV 2016 Workshops. Lecture Notes in Computer Science, Vol. 9914*. Hua, G.; Jégou, H. Eds. Springer Cham, 777–823, 2016.

[34] Mueller, M.; Smith, N.; Ghanem, B. A benchmark and simulator for UAV tracking. In: *Computer Vision – ECCV 2016. Lecture Notes in Computer Science, Vol. 9905*. Leibe, B.; Matas, J.; Sebe, N.; Welling, M. Eds. Springer Cham, 445–461, 2016.

[35] Henriques, J. F.; Caseiro, R.; Martins, P.; Batista, J. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 37, No. 3, 583–596, 2015.

[36] Danelljan, M.; Hager, G.; Khan, F. S.; Felsberg, M. Discriminative scale space tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 39, No. 8, 1561–1575, 2017.

[37] Danelljan, M.; Häger, G.; Khan, F. S.; Felsberg, M. Learning spatially regularized correlation filters for visual tracking. In: Proceedings of the IEEE International Conference on Computer Vision, 4310–4318, 2015.

[38] Bertinetto, L.; Valmadre, J.; Golodetz, S.; Miksik, O.; Torr, P. H. S. Staple: Complementary learners for real-time tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1401–1409, 2016.

[39] Valmadre, J.; Bertinetto, L.; Henriques, J.; Vedaldi, A.; Torr, P. H. S. End-to-end representation learning for correlation filter based tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 5000–5008, 2017.

[40] Nam, H.; Han, B. Learning multi-domain convolutional neural networks for visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4293–4302, 2016.

[41] Danelljan, M.; Bhat, G.; Khan, F. S.; Felsberg, M. ECO: Efficient convolution operators for tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 6931–6939, 2017.

[42] Danelljan, M.; Robinson, A.; Shahbaz Khan, F.; Felsberg, M. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In: *Computer Vision – ECCV 2016. Lecture Notes in Computer Science, Vol. 9909,* Leibe, B.; Matas, J.; Sebe, N.; Welling, M. Eds. Springer Cham, 472–488, 2016.

[43] Wang, G. T.; Luo, C.; Xiong, Z. W.; Zeng, W. J. SPM-tracker: Series-parallel matching for real-time visual object tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 3638–3647, 2019.

[44] Danelljan, M.; Bhat, G.; Khan, F. S.; Felsberg, M. ATOM: Accurate tracking by overlap maximization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4655–4664, 2019.

[45] Miller, G. A. WordNet. *Communications of the ACM* Vol. 38, No. 11, 39–41, 1995.

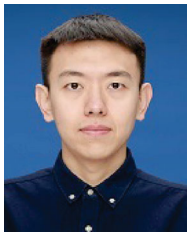
[46] Guo, Q.; Feng, W.; Zhou, C.; Huang, R.; Wan, L.; Wang, S. Learning dynamic Siamese network for visual object tracking. In: Proceedings of the IEEE International Conference on Computer Vision, 1781–1789, 2017.

[47] Hong, Z. B.; Zhe, C.; Wang, C. H.; Mei, X.; Prokhorov, D.; Tao, D. C. MUlti-Store Tracker (MUSTer): A cognitive psychology inspired approach to object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 749–758, 2015.

[48] Zhang, J. M.; Ma, S. G.; Sclaroff, S. MEEM: Robust tracking via multiple experts using entropy minimization. In: *Computer Vision – ECCV 2014. Lecture Notes in Computer Science, Vol. 8694*. Fleet, D.; Pajdla, T.; Schiele, B.; Tuytelaars, T. Eds. Springer Cham, 188–203, 2014.

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

Dong Chen is a student in the School of Artificial Intelligence, University of the Chinese Academy of Sciences. He received his B.E. degree in computer science and technology from Shihezi University in 2017. He is currently working towards an M.Eng. degree at the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences. His research interests include computer vision and machine learning.



Fan Tang is an assistant professor in the School of Artificial Intelligence, Jilin University. He received his B.Sc. degree in computer science from North China Electric Power University in 2013 and his Ph.D. degree from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, in 2019. His research interests include computer graphics, computer vision, and machine learning.

Weiming Dong is a professor in the Sino-European Lab in Computer Science, Automation and Applied Mathematics (LIAMA) and National Laboratory of Pattern Recognition (NLPR) at the Institute of Automation, Chinese Academy of Sciences. He received his B.Sc. and M.Sc. degrees in computer science in 2001 and 2004, both from Tsinghua University, China. He received his Ph.D. degree in computer science from the University of Lorraine, France, in 2007. His research interests include computational visual media and computational creativity.

Hanxing Yao received his B.Sc. degree in architectural engineering in 1999 and his M.Sc. degree in computer science in 2002, both from Chongqing University, China. He is the director of the AI Department of Beijing LLVISION Technology Co., Ltd. His research interests include computer vision and video retrieval.

Changsheng Xu is a professor in the National Lab of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences and Executive Director of the China–Singapore Institute of Digital Media. His research interests include multimedia content analysis, indexing and retrieval, pattern recognition, and computer vision. He holda 30 granted or pending patents and has published over 200 refereed research papers in these areas. He is an Associate Editor of *IEEE Trans. on Multimedia*, *ACM Trans. on Multimedia Computing, Communications and Applications*, and *ACM/Springer Multimedia Systems Journal*.