SiamCAN: Real-Time Visual Tracking Based on Siamese Center-Aware Network

Wenzhang Zhou, Longyin Wen, Libo Zhang, Dawei Du, Tiejian Luo, and Yanjun Wu

***Abstract*— In this article, we present a novel Siamese center-aware network (SiamCAN) for visual tracking, which consists of the Siamese feature extraction subnetwork, followed by the classification, regression, and localization branches in parallel. The classification branch is used to distinguish the target from background, and the regression branch is introduced to regress the bounding box of the target. To reduce the impact of manually designed anchor boxes to adapt to different target motion patterns, we design the localization branch to localize the target center directly to assist the regression branch generating accurate results. Meanwhile, we introduce the global context module into the localization branch to capture long-range dependencies for more robustness to large displacements of the target. A multi-scale learnable attention module is used to guide these three branches to exploit discriminative features for better performance. Extensive experiments on 9 challenging benchmarks, namely VOT2016, VOT2018, VOT2019, OTB100, LTB35, LaSOT, TC128, UAV123 and VisDrone-SOT2019 demonstrate that SiamCAN achieves leading accuracy with high efficiency. Our source code is available at https://isrc.iscas.ac.cn/gitlab/research/siamcan.**

***Index Terms*— Visual tracking, Siamese center-aware network, multi-scale learnable attention.**

I. Introduction

VISUAL object tracking is a hot research direction with a wide range of applications, such as surveillance, autonomous driving, and human-computer interaction. Although significant progress has been made in recent years, it is still a challenging task due to various factors, including occlusion, abrupt motion, and illumination variation.

Manuscript received August 12, 2020; revised January 7, 2021; accepted February 4, 2021. Date of publication March 3, 2021; date of current version March 17, 2021. This work was supported in part by the Key Research Program of Frontier Sciences, CAS, under Grant ZDBS-LY-JSC038 and in part by the National Natural Science Foundation of China under Grant 61807033. The work of Libo Zhang was supported in part by the Youth Innovation Promotion Association, CAS, under Grant 2020111, and in part by the Outstanding Youth Scientist Project of ISCAS. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Ming-Ming Cheng. *(Wenzhang Zhou and Longyin Wen contributed equally to this work.) (Corresponding author: Libo Zhang.)*

Wenzhang Zhou and Tiejian Luo are with the School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 101400, China.

Longyin Wen was with JD Finance America Corporation, Mountain View, CA 94043 USA. He is now with ByteDance, Inc., Mountain View, CA 94041 USA.

Libo Zhang and Yanjun Wu are with the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences (ISCAS), Beijing 100190, China (e-mail: libo@iscas.ac.cn).

Dawei Du is with the Computer Science Department, University at Albany, State University of New York, Albany, NY 12222 USA.

Digital Object Identifier 10.1109/TIP.2021.3060905

Modern visual tracking algorithms can be roughly divided into two categories: (1) the correlation filter based approaches [1]–[4], and (2) the deep convolution network based approaches [5]–[8]. The correlation filter (CF) based approach trains a regressor for tracking using circular correlation via Fast Fourier Transform (FFT). With the arrival of the deep convolution network, some researchers use offline learned deep features [1], [9], [10] to improve the accuracy. Considering efficiency, those trackers abandon model update in tracking process, which greatly harms the accuracy and generally performs worse than the CF based approaches.

Recently, the deep Siamese-RPN method [11] is presented, which formulates the tracking task as the one-shot detection task, *i.e.*, using the bounding box in the first frame as the only exemplar. By exploiting the domain specific information, Siamese-RPN surpasses the performance of the CF based methods. Some methods [6], [7], [12] attempt to improve the method [11] by using layer-wise and depth-wise feature aggregations, simultaneously predicting target bounding box and class-agnostic binary segmentation, and using ellipse fitting to estimate the bounding box rotation angle and size for better performance. However, the aforementioned methods rely on the pre-set anchor boxes to regress the bounding box of target, which can not adapt to various motion patterns and scales of targets, especially when the *fast motion* and *occlusion* challenges occur.

To this end, we propose the Siamese Center-Aware Network (SiamCAN) for visual tracking. Specifically, it consists of the Siamese feature extraction subnetwork, followed by three parallel branches, *i.e.*, anchor-based classification and regression branches, and anchor-free localization branch. Similar to [11], the classification branch is used to distinguish the target from background, while the regression branch is used to regress the bounding box of target. To reduce the impact of manually designed anchor boxes to adapt to different motion patterns and scales of targets, we design the localization branch to localize the target center directly. It helps the regression and classification branches generate more accurate results. Meanwhile, the global context module [13] is merged into the localization branch to capture long-range dependency to deal with large target displacement. Meanwhile, the multi-scale learnable attention module is introduced to guide these three branches to exploit more discriminative representation, resulting in better performance. The whole network is trained in an end-to-end fashion offline with the large-scale image pairs by the standard SGD algorithm with back-propagation [14] in the training sets of MS COCO [15], ImageNet DET/VID [16], and

YouTube-BoundingBoxes [17] datasets. For inference, visual tracking is formulated as the local one-shot detection task by using the bounding box of target in the first frame as the only exemplar. Comprehensive experiments on nine benchmarks including VOT2016 [18], VOT2018 [19], VOT2019 [20], OTB100 [21], LaSOT [22], TC128 [23], LTB35 [24], UAV123 [25], and VisDrone-SOT2019 [26] show the effectiveness of our SiamCAN method. For example, our method achieves EAO score of 0.513, *i.e.*, 8.9% relative improvement compared to the second best tracker on VOT2016. Moreover, the ablation study is conducted to verify the influence of important components in our method.

The main contributions of this work are summarized as follows. 1) We propose a Siamese center-aware network (SiamCAN) for visual tracking, where the classification, regression, and localization branches are jointly trained using the proposed multi-scale learnable attention module. 2) The localization branch is enhanced by global context to obtain more robustness for large target displacement. 3) Extensive experiments are conducted on 9 challenging benchmarks to demonstrate the effectiveness of the proposed SiamCAN method comprehensively.

II. Related Works

Extracting discriminative features is a critical step in visual tracking. MOSSE [27] and CSK [28] attempt to exploit discriminative correlation filters learned dynamically to adapt to appearance variations of targets. KCF [29] learns a dual correlation filter with HOG feature based on CSK [28]. SAMF [30] is a scale adaptive method with powerful features including color and HOG information for visual tracking. The methods in [1], [2], [9] compute the convolutions between the target template features and the search area to determine the target size and location in the video sequences. However, the aforementioned methods rely on hand-crafted features, which may be less effective than deep learnt features for visual tracking, especially in complex scenarios.

Compared to using hand-crafted features, several methods attempt to extract features from a pretrained convolutional neural network (CNN) to improve the tracking performance. FCNT [31] conducts an in-depth study on the properties of CNN features offline pre-trained on ImageNet and designs a visual tracking method using the fully convolutional neural network. MDNet [32] attempts to learn the generic target representations using shared layers, and integrates a binary classification layer updated online for object tracking. MemDTC [33] presents a dynamic memory network to learn feature representation of targets and uses LSTM as a memory controller to adapt the template. To address the two problems in deep classification network based methods, *i.e.*, capturing rich appearance variations and handling extreme class imbalance between positive and negative samples, VITAL [34] uses the adversarial learning to identify the mask that maintains the most robust features of targets over a long temporal span. ATOM [8] proposes a tracking architecture, formed by dedicated target estimation and classification components to extract high-level knowledge of objects for better performance.

Some other researchers attempt to use the Siamese network for visual tracking. SINT [5] and SiamFC [35] formulate the visual tracking problem as the pairwise similarity learning of the target in consecutive frames using the Siamese network. Dong *et al.*[36] use the triplet loss to train the Siamese network to exploit discriminative features instead of the pairwise loss, which can mine the potential relationship among exemplar, *i.e.*, positive and negative instances, and contains more elements for training. To take fully use of semantic information, He *et al.*[37] construct a twofold Siamese networks, which is composed of a semantic branch and an appearance branch, and each of them is a similarity-learning Siamese network. In [38], the semantic and objectness information is used to produce a class-agnostic ridge regression network for object tracking.

Inspired by Region Proposal Network (RPN) in object detection, SiamRPN [11] formulates visual tracking as a local one-shot detection task in inference, which uses the Siamese network for feature extraction and RPN for target classification and regression. Fan and Ling [39] construct a cascaded RPN from deep high-level to shallow low-level layers in a Siamese network. Zhu *et al.*[40] design a distractor-aware Siamese networks for accurate long-term tracking by using an effective sampling strategy to control the distribution of training data, and make the model focus on the semantic distractors. SiamRPN ++ [6] is improved from SiamRPN [11] by performing layer-wise and depth-wise aggregations, which not only improves the accuracy but also reduces the model size. Zhang and Peng [41] design a residual network for visual tracking with controlled receptive field size and network stride. Han *et al.*[42] introduce the anchor-free detection network into visual tracking directly. Moreover, SiamMask [7] combines the fully-convolutional Siamese tracker with a binary segmentation head for accurate tracking. To track the rotated target accurately, Chen *et al.*[12] improve SiamMask [7] by using ellipse fitting to estimate the bounding box rotation angle and size with the mask on the target. However, the aforementioned algorithms fail to consider the variations of target motion patterns, resulting in failures when the *fast motion*, *occlusion*, and *camera motion* challenges occur.

Inspired by the anchor-free object detection methods [43], [44], various object tracking methods [45]–[48] are developed to directly predict the locations and sizes of targets in video sequences. Chen *et al.* [45] directly localize the target objects and regress the bounding boxes in a fully convolutional network. Guo *et al.* [46] integrate a centerness branch in parallel with the classification and regression branches in the Siamese network to improve the accuracy. Li *et al.*[47] develop the Siamese keypoint prediction network to generate cascade heatmaps to cover both accuracy and robustness. Besides, Zhang *et al.*[48] directly predict the position and size of target in an anchor-free fashion rather than refining the reference anchor boxes. In contrast to the aforementioned methods, we introduce the localization branch with the global context module to help the regression and classification branches to generate accurate results.

On the other hand, increasing interest has been attracted in long-term tracking [49]–[53]. Zhang *et al.* [49] propose the new deep regression and verification network, where a

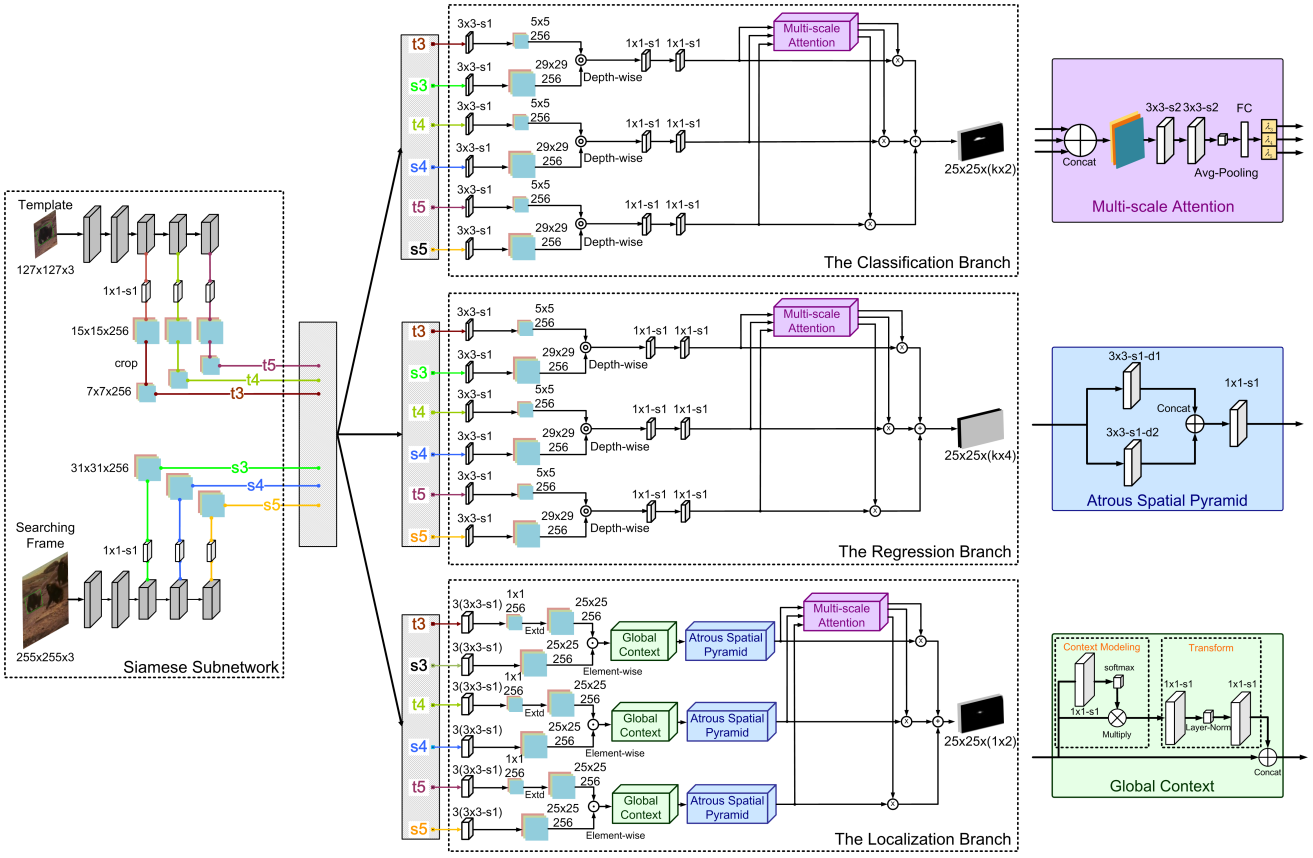


Fig. 1. The architecture of our SiamCAN method, which consists of the Siamese feature extraction subnetwork followed by the classification, regression, and localization branches in parallel. The pairs of feature maps from different layers of the Siamese feature extraction subnetwork are fed into the three branches. "3 × 3-s1-d2" denotes a convolution layer with 3 × 3 kernel, stride 1 and dilation rate 2.

dynamic switch scheme between local search and image-wide re-detection is designed based on the outputs from both regression and verification networks. Lee *et al.* [50] propose a memory model based on the Siamese network, where Memory stores in MMLT are divided into short and long-term stores in tracking and re-detection processes respectively. Yan *et al.* [51] perform a robust and real-time long-term tracking framework based on the skimming and perusal modules, where the perusal module is used to infer the optimal candidate proposal with its confidence score and the skimming module is used to choose the most possible regions from a large number of sliding windows efficiently. SiamRCNN [52] combines Siamese network with two-stage detector and tracklet-based dynamic programming algorithm, which re-detects the same object in the first-frame template to avoid target drifting. Different from offline-trained Siamese networks, Dai *et al.* [53] propose the meta-updater module to integrate geometric, discriminative, and appearance cues in a sequential manner by a cascaded LSTM module. Thus a binary output is learned to guide the update of different tracker embedded with the meta-updater.

III. Siamese Center-Aware Network

As shown in Fig. 1, our Siamese center-aware network is a feed-forward network, which is formed by a Siamese feature extraction subnetwork, followed by three parallel branches, *i.e.*, the classification branch, the regression branch, and the localization branch. The classification branch is designed to distinguish the foreground proposals from the background, and the regression branch is used to regress the bounding box of target based on the preset anchor boxes. Inspired by [44], we integrate a localization branch to localize the target center directly, which can help the regression branch estimate the size of the target more accurately. Let k be the number of pre-set anchors. Thus, we have $2k$ channels for classification, $4k$ channels for regression and 2 channels for localization, and denote the output feature maps of the three branches as $\mathcal{O}^{\text{cls}}_{w\times h\times 2k}$, $\mathcal{O}^{\text{reg}}_{w\times h\times 4k}$, and $\mathcal{O}^{\text{loc}}_{w\times h\times 2}$. Notably, each point in $\mathcal{O}^{\text{cls}}_{w\times h\times 2k}$, $\mathcal{O}^{\text{reg}}_{w\times h\times 4k}$, and $\mathcal{O}^{\text{loc}}_{w\times h\times 2}$ contains $2k$, $4k$, and 2 channel vectors, representing the positive and negative activations of each anchor at the corresponding locations of the original map for each branch. In the following sections, we will describe each module in our SiamCAN in detail.

Siamese feature extraction subnetwork. Inspired by [11], we use the fully convolutional network without padding in the Siamese feature extraction subnetwork. Specifically, there are two components in the Siamese feature extraction subnetwork, *i.e.*, the *template module* encoding the target patch in the historical frame, and the *detection module* encoding the image patch including the target in the current frame. The two components share parameters in CNN. Let α be the target patch fed into the template module, and β be the image

patch fed into the detection module. We denote the output feature maps of the Siamese feature extraction subnetwork at the i-th layer as $\phi_i(\alpha)$ and $\phi_i(\beta)$. Then, we split each of them into three branches, *i.e.*, $\phi_i^{\text{cls}}(\alpha)$ and $\phi_i^{\text{cls}}(\beta)$ for the classification branch, $\phi_i^{\text{reg}}(\alpha)$ and $\phi_i^{\text{reg}}(\beta)$ for the regression branch, and $\phi_i^{\text{loc}}(\alpha)$ and $\phi_i^{\text{loc}}(\beta)$ for the localization branch, by a convolution layer with kernel size 3×3 and stride 1, but keeping the number of channels unchanged. Similar to the previous work [6], we use the ResNet-50 network [54] as the backbone. To reduce the computational complexity, we extract the feature maps from the backbone with the channel 256 by one 1×1 convolutional layer, and then crop the center 7×7 regions from the 15×15 feature maps as the template feature [55]. Due to the paddings of all layers in the backbone, the 7×7 feature map can still represent the entire target region.

Classification branch. As shown in Fig. 1, the classification branch takes the multi-scale features produced by the *template* and *detection* modules of the Siamese feature extraction subnetwork, *e.g.*, t3, s3, t4, s4, t5, and s5, to compute the correlation feature maps between the input *template* ($\phi_i^{\text{cls}}(\alpha)$) and *detection* ($\phi_i^{\text{cls}}(\beta)$) feature maps, *i.e.*,

$$F_{w\times h\times 2k}^{\text{cls}}(m) = \phi_m^{\text{cls}}(\alpha) \star \phi_m^{\text{cls}}(\beta), \tag{1}$$

where $\star$ denotes depth-wise convolution operation. We use two convolution layers with the kernel size 1×1 and stride size 1, to produce the features with $2k$ channels, *i.e.*, $\mathcal{F}_{w\times h\times 2k}^{\text{cls}}(m)$, $m = 1, \cdots, L$, where L is the total number of feature maps for prediction. After that, we use the multi-scale attention module to guide the branch to exploit discriminative features for accurate results. Specifically, we first concatenate the feature maps at different layers, *i.e.*, $\mathcal{F}_{w\times h\times 2k}^{\text{cls}}(m)$, $m = 1, \cdots, L$, and use two convolutional layers with the kernel size 3×3 and stride size 2, followed by an average pooling and fully connected layers to produce the weights, *i.e.*, γ_i^{cls}. After that, the feature maps with different scales are summed with the weights γ_m^{cls} to generate the final predictions $\mathcal{O}_{w\times h\times 2k}^{\text{cls}}$, *i.e.*,

$$\mathcal{O}_{w\times h\times 2k}^{\text{cls}} = \sum_{m=1}^{L} \gamma_m^{\text{cls}} \cdot \mathcal{F}_{w\times h\times 2k}^{\text{cls}}(m). \tag{2}$$

Each point in the output $\mathcal{O}_{w\times h\times 2k}^{\text{cls}}$ is a $2k$ channel vector, indicating the positive and negative activations of each anchor at the corresponding locations of the original map. Notably, the weights γ_m^{cls}, $m = 1, \cdots, L$, are learned in the training phase, *i.e.*, the gradients of the whole network can be back-propagated to update γ_m^{cls}, $m = 1, \cdots, L$. Please see Fig. 1 for more details.

Regression branch. As described above, the regression branch is designed to generate the accurate bounding box of target in the current video frame. As shown in Fig. 1, we compute the correlation feature maps between the input *template* and *detection* feature maps. For example, for the feature map at the m-th layer, the correlation feature map $F_{w\times h\times 4k}^{\text{reg}}(m)$ is computed as

$$F_{w\times h\times 4k}^{\text{reg}}(m) = \phi_m^{\text{reg}}(\alpha) \star \phi_m^{\text{reg}}(\beta), \tag{3}$$

where $\phi_m^{\text{reg}}(\alpha)$ and $\phi_m^{\text{reg}}(\beta)$ are the m-th feature map from the *template* and *detection* modules. After that, two convolution layers with the kernel size 1×1 and stride size 1, are applied on $F_{w\times h\times 4k}^{\text{reg}}(m)$ to produce the corresponding feature map $\mathcal{F}_{w\times h\times 4k}^{\text{reg}}(m)$, $m = 1, \cdots, L$, keeping the channel size $4k$ unchanged, where L is the total number of feature maps used for prediction. Similar to the *classification branch*, we use the multi-scale attention module with the learnable weights γ_m^{reg}, $m = 1, \cdots, L$, to make the branch focus on exploiting discriminative features to generate accurate results, *i.e.*,

$$\mathcal{O}_{w\times h\times 4k}^{\text{reg}} = \sum_{m=1}^{L} \gamma_m^{\text{reg}} \cdot \mathcal{F}_{w\times h\times 4k}^{\text{reg}}(m), \tag{4}$$

where $\mathcal{O}_{w\times h\times 4k}^{\text{reg}}$ is the output of the regression branch. Each point on $\mathcal{O}_{w\times h\times 4k}^{\text{reg}}$ contains a $4k$ channel vector, indicating the normalized distance between the predicted anchor box and the ground-truth bounding box.

Localization branch. As discussed above, the anchor-based classification and regression branches are used to distinguish the target from background. However, different targets have different motion patterns, *i.e.*, some targets move fast, while some targets move slowly. In some challenging scenarios such as *fast motion* and *small object*, it is difficult for the regression branch to estimate the locations of targets accurately with the pre-set anchors.

To make our tracker adapt to various motion patterns of targets, we introduce a *localization branch* to directly localize the target center to help the *regression branch* and *classification branch* produce accurate results. Firstly, we obtain the multi-scale features $\Psi_i^{\text{loc}}(\alpha)$ and $\Psi_i^{\text{loc}}(\beta)$ from the Siamese feature extraction subnetwork. Then the global features $\phi_i^{\text{loc}}(\alpha)$ and $\phi_i^{\text{loc}}(\beta)$ of objects are extracted by three convolutional layers. Finally, we compute the correlation feature map with $\phi_i^{\text{loc}}(\alpha)$ and $\phi_i^{\text{loc}}(\beta)$ to predict the center map of the tracking object. The correlation feature map can be computed as

$$F_{w\times h\times 2}^{\text{loc}}(m) = \mathbb{S}[\phi_m^{\text{loc}}(\alpha)] \odot \phi_m^{\text{loc}}(\beta), \tag{5}$$

where $\mathbb{S}[\cdot]$ denotes the resize operation to make the two feature maps to be the same size, and $\odot$ denotes element-wise multiplication operation, see Fig. 1.

Global context. Since the traditional convolution layers only capture pixel relations in a local neighborhood, we insert the global context module [13] to integrate long-range dependency between target and background regions by explicitly using a query-independent attention map for all query positions. As shown in Fig. 1, the global context consists of the Context Modeling and Transform modules. The Context Modeling can capture the pairwise relations between each pixel and all other pixels via self-attention mechanism. The Transform can capture channel-wise dependencies using two 1×1 convolutional layers. Thus the model can obtain the global context features so that the tracker is more robust to large target displacement.

Inspired by [56], we design the atrous spatial pyramid module to capture the context information at multiple scales, which applies two parallel atrous convolution with different rates, followed by a convolution layer with 1×1 kernel size

and stride 1. In this way, we can generate the multi-scale discriminative features $\mathcal{F}^{\text{loc}}_{w\times h\times 2}(m)$, where $m = 1, \cdots, L$. Then, similar to the *classification* and *regression* branches, we use the multi-scale attention module with the learnable weights γ^{loc}_m, $m = 1, \cdots, L$ to generate the final predictions. That is,

$$\mathcal{O}^{\text{loc}}_{w\times h\times 2} = \sum_{m=1}^{L} \gamma^{\text{loc}}_m \cdot \mathcal{F}^{\text{loc}}_{w\times h\times 2}(m). \tag{6}$$

Notably, each point on the prediction $\mathcal{O}^{\text{loc}}_{w\times h\times 2}$ is a two channel vector, representing the offset of the corresponding center location in the original map.

Loss function. The loss function in our method is formed by three terms corresponding to three branches, *i.e.*, the classification loss L_{cls}, the regression loss L_{reg}, and the localization loss L_{loc}. The overall loss function is computed as:

$$\mathcal{L} = \lambda_{\text{cls}} L_{\text{cls}}(\text{u}, \text{u}^*) + \lambda_{\text{reg}} L_{\text{reg}}(\text{p}, \text{p}^*) + \lambda_{\text{loc}} L_{\text{loc}}(\text{c}, \text{c}^*), \tag{7}$$

where λ_{cls}, λ_{reg} and λ_{loc} are the parameters used to balance the three loss terms. u and u^* are the predicted and ground-truth labels of the target bounding boxes, p and p^* are the predicted and ground-truth bounding boxes, and c and c^* are the predicted and ground-truth labels of the target center. We use the cross-entropy loss to supervise the *classification* and *localization* branches, and L1 loss to supervise for the *regression* branch.

Specifically, the classification loss $L_{\text{cls}}(\text{u}, \text{u}^*)$ is computed as

$$L_{\text{cls}}(\text{u}, u^*) = -\frac{1}{2}\sum_i \sum_j \sum_k \big(u^*_{i,j,k} \log u_{i,j,k} + (1 - u^*_{i,j,k}) \log(1 - u_{i,j,k})\big), \tag{8}$$

where $u^*_{i,j,k}$ is the ground-truth label of the k-th anchor at (i, j) of the output $\mathcal{O}^{\text{cls}}_{w\times h\times 2k}$, and $u_{i,j,k}$ is the predicted label of the k-th anchor at (i, j) generated by the softmax operation from $\mathcal{O}^{\text{cls}}_{w\times h\times 2k}$ over 2 categories.

Meanwhile, we use the L1 loss function to compute the regression loss $L_{\text{reg}}(p, p^*)$, *i.e.*,

$$L_{\text{reg}}(\text{p}, \text{p}^*) = \frac{1}{N}\sum_i \sum_j \sum_k [u^*_{i,j,k} > 0] \|\delta(p_{i,j,k}, p^*_{i,j,k})\|_1, \tag{9}$$

where N is the number of positive anchors, and the Iverson bracket indicator function $[u^*_{i,j,k} > 0]$ outputs 1 when the condition is true, *i.e.*, the anchor is not negative $u^*_{i,j,k} > 0$, and 0 otherwise. $p_{i,j,k} = (x, y, w, h)$ and $p^*_{i,j,k} = (x^*, y^*, w^*, h^*)$ are the predicted and ground-truth bounding boxes, where (x, y) and (x^*, y^*) are the center coordinates and (w, h) and (w^*, h^*) are the sizes. We use the normalized distances δ to compute the regression loss, *i.e.*, $\delta(p, p^*) = ((x^* - x)/x, (y^* - y)/y, \ln(w^*/w), \ln(h^*/h))$.

Moreover, we also use the cross-entropy loss for the localization branch as follows.

$$L_{\text{loc}}(\text{c}, \text{c}^*) = -\frac{1}{2}\sum_i \sum_j \big(c^*_{i,j} \log c_{i,j} + (1 - c^*_{i,j}) \log(1 - c_{i,j})\big), \tag{10}$$

where $c^*_{i,j}$ is the ground-truth label of the center of the target at (i, j) of the output $\mathcal{O}^{\text{loc}}_{w\times h\times 2}$, and $c_{i,j}$ is the predicted label of the center of the target at (i, j) generated by the softmax operation from $\mathcal{O}^{\text{loc}}_{w\times h\times 2}$. Notably, we generate the ground-truth center location of the target c^* (where $c^*_{i,j} \in [0, 1]$) using the Gaussian kernel with the object size-adaptive standard deviation [57].

A. Training and Inference

Data augmentation. We use several data augmentation strategies such as blur, rescale, rotation, flipping and gray scaling to construct a robust model to adapt to variations of objects using the video sequences in MS COCO [15], ImageNet DET/VID [16], and YouTube-BoundingBoxes [17]. For the positive image pairs, we randomly select two frames from the same video sequences with the interval less than 100 frames, or different image patches including target object in the MS COCO and ImageNet DET datasets. Meanwhile, for the negative image pairs, we randomly select an image from the datasets and another one without including the same target. Notably, the ratio between the positive and negative pairs is set to 4 : 1.

Anchors design and matching. For each point, we pave 5 anchors with stride 8 on each pixel, where the anchor ratios are set to $[1/3, 1/2, 1/1, 2/1, 3/1]$ and the anchor scale is set to 8. Meanwhile, during the training phase, we determine the correspondence between the anchors and ground-truth boxes based on the jaccard overlap. Specifically, if the overlap between the anchor and ground-truth box is larger than 0.6, the anchor is determined as positive. Meanwhile, if the overlap between the anchor and all ground-truth boxes is less than 0.3, the anchor is determined as negative.

Optimization. The whole network is trained in an end-to-end manner using the SGD optimization algorithm with 0.9 momentum and 0.0001 weight decay on the training sets of MS COCO [15], ImageNet DET/VID [16], and YouTube-BoundingBoxes [17] datasets. Notably, we use three stages to train the proposed method empirically. For the first two stages in the training process, we disable the multi-scale attention modules in the three branches, *i.e.*, set equal weights to different scales of features. First, the backbone ResNet-50 network in the Siamese feature subnetwork is initialized by the pre-trained model on the ILSVRC CLS-LOC dataset [16]. We train the classification and regression branches in the first 10 epochs with other parameters fixed, and then train the Siamese feature subnetwork, and the classification and regression branches in the next 10 epochs. Second, we finetune the classification, regression and localization branches with other parameters fixed in the first 10 epochs, and then train the whole network in the next 10 epochs. Third, we enable the multi-attention module and learn the weights of different scales of features with other parameters fixed in the first 15 epochs. After that, the whole network is finetuned in the next 5 epochs. In each stage, we set the initial learning rate to 0.001, and gradually increase it to 0.005 in the first 5 epochs. We decrease it to 0.0005 in the next 15 epochs.

Inference. In the inference phase, our tracker takes the current video frame and the template target patch as input, and outputs the classification, regression, and localization results. Then, we perform softmax operation on both the outputs of the classification and localization results to obtain the positive activations, *i.e.*, u with the size $w \times h \times k$, and c with the size $w \times h \times 1$. After that, we expand the localization result c to make it to the same size of the classification result u. In this way, the final prediction is computed by the weight combination of four terms, *i.e.*, the localization result c, the classification result u, the cosine window ξ with the size $w \times h$ (expanding to $w \times h \times k$), and the scale change penalty ρ with the size $w \times h \times k$ [11],

$$\Theta_{w\times h\times k} = \omega_2 \cdot \rho \cdot \big(\omega_1 \cdot \mathrm{u} + (1-\omega_1)\cdot \mathrm{c}\big) + (1-\omega_2)\cdot \xi, \quad (11)$$

Notably, the cosine window ξ is used to suppress the boundary outliers [29], and the scale change penalty ρ to suppress large change in size and ratio [11]. The weights ω_1 and ω_2 are used to balance the above terms, which are set to 0.7 and 0.6, empirically. After that, we can obtain the optimal center location and scale of target based on the maximal score on $\Theta_{w\times h\times k}$. Notably, the target size is updated by the linear interpolation to guarantee the smoothness of size.

IV. Experiments

Our SiamCAN method is implemented using the Pytorch tracking platform PySOT.[1] We conduct the experiments on nine datasets including VOT2016 [18], VOT2018 [19], VOT2019 [20], LTB35 [24], LaSOT [22], TC128 [23], UAV123 [25], VisDrone-SOT2019 [26] and OTB100 [21]. All experiments are conducted on a workstation with an Intel i7-7800X CPU, 8G memory, and 2 NVIDIA RTX2080 GPUs.

Evaluation protocol. For the VOT2016 [18], VOT2018 [19] and VOT2019 [20] datasets, we use the evaluation protocol in the VOT Challenge [18], [19], *i.e.*, the *Expected Average Overlap* (EAO), *Accuracy* (A), and *Robustness* (R) are used to evaluate the performance of trackers. The *Accuracy* score indicates the average overlap of the successfully tracked frames, and the *Robustness* score indicates the failure rate of the tracking frames.[2] EAO takes both accuracy and robustness into account, which is used as the primary metric for ranking trackers.

Meanwhile, for the OTB100 [21], LaSOT [22], TC128 [23], UAV123 [25], and VisDrone-SOT2019 [26] datasets, we use the success and precision scores to evaluate the performance of trackers based on the evaluation methodology in [21]. The success score is defined as the area under the success plot, *i.e.*, the percentage of successfully tracked frames[3] *vs.* bounding box overlap threshold in the interval $[0, 1]$. The precision score is computed as the percentage of frames whose predicted location is within a given distance threshold from the center of ground-truth box based on the Euclidean distance on the image plane. We set the distance threshold to 20 pixels in our evaluation. In general, the success score is used as the primary metric for ranking trackers.

For the long-term tracking LTB35 dataset [24], we use three metrics including tracking precision (P), tracking recall (R) and tracking F-score in evaluation. The tracking methods are ranked by the maximum F-score over different confidence thresholds, *i.e.*, $\mathrm{F} = \frac{2\mathrm{P}\cdot\mathrm{R}}{\mathrm{P}+\mathrm{R}}$.

TABLE I

Comparison Results on VOT2016 [18]. $*$ Denotes That the Result is Obtained Using the PySOT Platform

Tracker	A	R	EAO
SiamFC [35]	0.530	0.460	0.235
C-COT [9]	0.539	0.238	0.331
CSRDCF [58]	0.510	0.240	0.338
SiamRPN [11]	0.560	1.080	0.344
FCAF [42]	0.581	1.020	0.356
C-RPN [39]	0.594	0.950	0.363
SiamRPN+ [41]	0.580	0.240	0.370
ECO [1]	0.550	0.200	0.375
ASRCF [59]	0.563	0.187	0.391
DaSiamRPN [40]	0.610	0.220	0.411
ROAM++ [60]	0.599	0.174	0.441
SiamMask* [7]	**0.643**	0.219	0.455
SiamRPN++* [6]	0.642	0.196	0.464
SiamMask_E* [12]	**0.677**	0.224	0.466
PTS [61]	0.642	**0.144**	**0.471**
SiamCAN*	0.636	**0.149**	**0.513**

A. State-of-the-Art Comparison

We compare the proposed method to the state-of-the-art trackers on nine challenging datasets. Note that the tracking results of other trackers are directly taken from the published articles or corresponding github repositories.

VOT2016. We conduct experiments on the VOT2016 dataset [18], which contains 60 sequences in total. As presented in Table I, we find that SiamCAN achieves the best EAO score of 0.513 and the second best robustness score of 0.149. The SiamMask_E [12] and SiamMask [7] methods obtain the top two accuracy scores. This is because they estimate the bounding box based on the mask generated by the segmentation head, resulting in a relatively more accurate bounding box, especially for the non-rigid targets. SiamMask_E [12] even uses ellipse fitting to estimate the bounding box rotation angle and size. Besides, by using a fully convolutional Siamese network to segment the target, PTS [61] acquires the best robustness score of 0.144 and the second best EAO score of 0.471. Compared with SiamRPN ++ [6], our SiamCAN method produces slightly inferior accuracy score but better robustness score, indicating that the localization branch can significantly decrease the tracking failure.

VOT2018. The VOT2018 dataset is also formed by 60 challenging video sequences with fully annotations. We evaluate the proposed SiamCAN method on VOT2018 [19], and report the results in Table II. Ocean [48] obtains the best robustness score of 0.117 and the EAO score of 0.489. This is attributed

[1] https://github.com/STVIR/pysot

[2] We define the failure of tracking if the overlap between the tracking result and ground-truth is reduced to 0.

[3] If the overlap between the predicted bounding box and ground-truth box in a frame is larger than a threshold, we regard the frame as a successfully tracked frame.

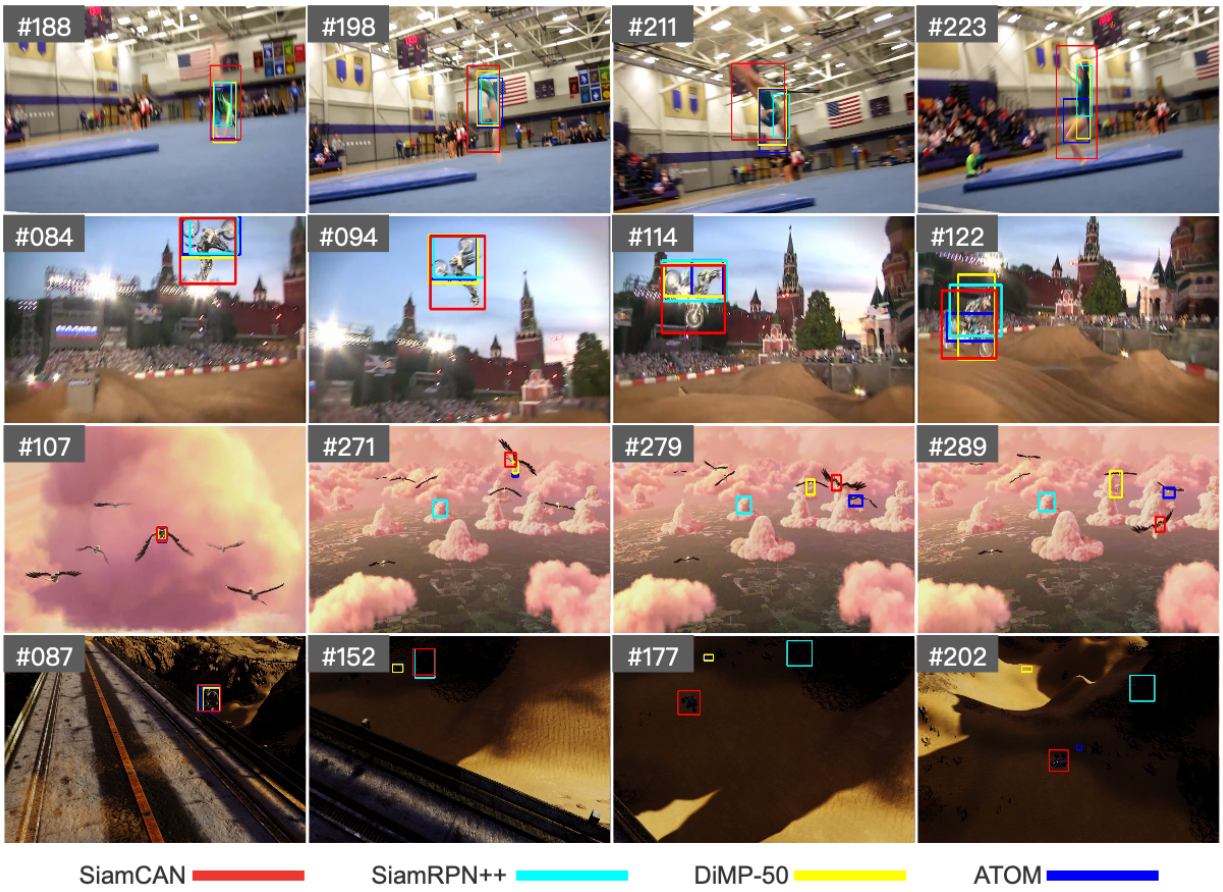


Fig. 2. Visual results of our approach compared with state-of-the-art trackers. The first and second rows denote the VOT2018 dataset [19]. The third row denotes the OTB100 dataset [21], and the fourth row denotes the UAV123 dataset [25].

TABLE II
COMPARISON RESULTS ON VOT2018 [19]. ∗ DENOTES THAT THE RESULT IS OBTAINED USING THE PYSOT PLATFORM

Tracker	A	R	EAO
SiamFC [35]	0.503	0.585	0.188
DSiam [62]	0.215	0.646	0.196
SiamRPN [11]	0.490	0.460	0.244
ECO [1]	0.484	0.276	0.280
SA_Siam_R [37]	0.566	0.258	0.337
DeepSTRCF [2]	0.523	0.215	0.345
DRT [3]	0.519	0.201	0.356
RCO [19]	0.507	0.155	0.376
UPDT [63]	0.536	0.184	0.378
DaSiamRPN [40]	0.586	0.276	0.383
MFT [19]	0.505	**0.140**	0.385
LADCF [4]	0.503	0.159	0.389
DomainSiam [38]	0.593	0.221	0.396
PTS [61]	0.612	0.220	0.397
ATOM [8]	0.590	0.204	0.401
SiamRCNN [52]	0.609	0.220	0.408
SiamRPN++* [6]	0.601	0.234	0.415
SiamMask* [7]	**0.615**	0.248	0.423
DiMP-50 [64]	0.597	0.153	0.440
SiamKPN [47]	0.606	0.192	0.440
SiamMask_E* [12]	**0.655**	0.253	0.446
SiamBAN [45]	0.597	0.178	0.452
Ocean [48]	0.592	**0.117**	**0.489**
SiamCAN*	0.605	0.183	**0.462**

to the powerful online updating mechanism, which can reduce tracking failures effectively. Following Ocean [48], our SiamCAN method achieves the second best EAO score of 0.462 without any online updating module. In terms of accuracy score, the similar trend can be observed that SiamMask_E [12] and SiamMask [7] achieve the best result. Except the trackers with segmentation module (*e.g.*, SiamMask_E [12], SiamMask [7] and PTS [61]), our SiamCAN method achieves comparable accuracy and robustness scores. In addition, as shown in Fig. 2, SiamCAN can obtain more precise target location and bounding box when the target moves fast.

VOT2019. Since the VOT2018 dataset is not saturated [19], VOT2019 [20] is refreshed by replacing 20% of the sequences from the update pool of $1,000$ sequences in the GOT-10k dataset [65]. As presented in Table III, DiMP-50 [64] and Ocean [48] achieve the top two EAO scores. Compared with other offline-trained Siamese network based methods (*i.e.*, SiamRPN ++ [6], SiamMask [12] and SiamBAN [45]), our method achieves better EAO score of 0.330.

LTB35. In addition, we evaluate our SiamCAN tracker on the LTB35 dataset [24], which is first presented in the VOT2018-LT challenge [19]. It includes 35 sequences with $14,687$ frames. Each sequence contains 12 long-term target disappearing cases in average. We compare SiamCAN to several state-of-the-art methods (such as SPLT [51], MBMD [49], MMLT[50], LTSINT [66] and SiamVGG [67]) in the VOT2018-LT challenge [19] and two recent methods in

TABLE III
COMPARISON RESULTS ON VOT2019 [19]

Tracker	A	R	EAO
MemDTC [33]	0.485	0.587	0.228
SiamMask [12]	0.594	0.461	0.287
SiamRPN++ [6]	0.599	0.482	0.285
ATOM [8]	**0.603**	0.411	0.292
DiMP-50 [64]	0.592	**0.278**	**0.379**
Ocean [48]	0.594	**0.316**	**0.350**
SiamBAN [45]	**0.602**	0.396	0.327
SiamCAN	0.597	0.371	0.330

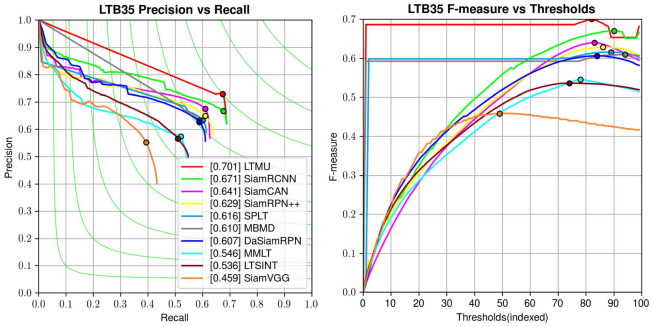


Fig. 3. Evaluation results on the LTB35 dataset [24], including recall and precision (left) and F-score (right).

long-term tracking task, LTMU [53] and SiamRCNN [52] in Fig. 3. It can be seen that two long-term trackers LTMU [53] and SiamRCNN [52] perform better than other methods. Without any re-detection module, our method obtain the third best F-score of 0.641, 1.2% higher than SiamRPN ++ [6]. Note that the results are even better than several long-term trackers including SPLT [51], MBMD [49], and MMLT [50]. It indicates the effectiveness of the proposed localization branch.

LaSOT. The LaSOT dataset [22] is composed of 280 videos with average 2, 500 frames per sequence. As shown in Fig. 4, our SiamCAN produces worse success score than the top 3 trackers including SiamRCNN [52], DiMP-50 [64] and LTMU [53]. This is attributed to the re-detection and model online updating mechanism in these trackers, which is effective to handle the long-term visual tracking. Note that the online updating mechanism is complementary to our method. We believe that it can be used in SiamCAN to further improve the performance. Compared to SiamRPN ++ [6], our method produces higher success (0.538 *vs.* 0.496) and precision scores (0.535 *vs.* 0.491).

TC128. The TC128 dataset [23] comprises 128 video sequences that are specifically collected to evaluate color-enhanced trackers. The videos are labeled with 11 attributes, similar to those in OTB [21]. We compare the proposed method to DiMP-50 [64], SiamRPN ++ [6], MEEM (LAB) [68], Struck (HSV) [69], KCF (LAB) [29], ASLA (LAB) [70] in Fig. 5. As shown in Fig. 5, we find that DiMP-50 [64] performs the best among all trackers, *i.e.*, 0.601 success score and 0.809 precision score. Without the online updating mechanism, our method produces the second

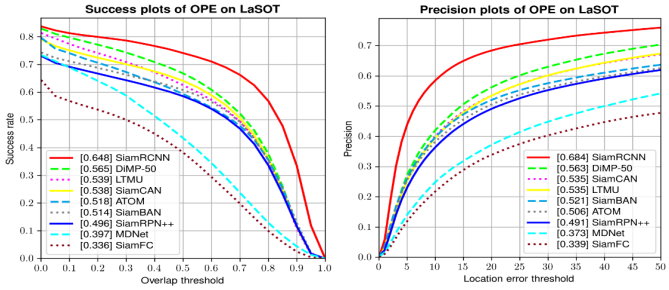


Fig. 4. Success and precision plots on the LaSOT dataset [22].

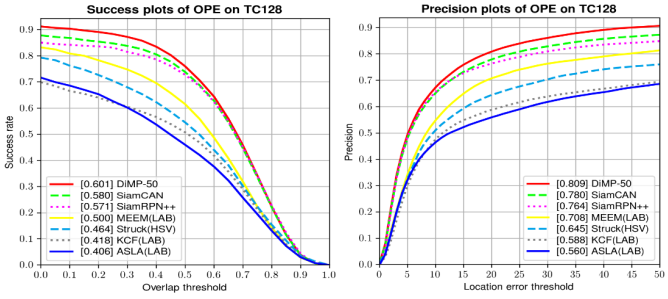


Fig. 5. Success and precision plots on the TC-128 dataset [23].

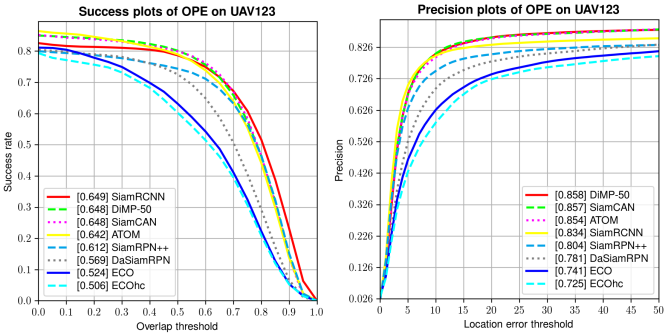


Fig. 6. Success and precision plots on the UAV123 dataset [25].

best results in terms of both success and precision scores, *i.e.*, 0.580 success and 0.780 precision scores. Thus, we can conclude that the online updating mechanism is also effective to handle color variations.

UAV123. We also evaluate our SiamCAN method on the UAV123 dataset [25], which is collected from an aerial viewpoint and includes 123 sequences in total with more than 110, 000 frames. As shown in Fig. 6, our method performs on par with the state-of-the-art trackers SiamRCNN [52] and DiMP-50 [64]. SiamRCNN [52] acquires the highest success score of 0.649 but worse precision socre of 0.834 than DiMP-50 [64] and our method. DiMP-50 [64] produces the same success score 0.648 as our method but a little bit worse precision score (0.857 *vs.* 0.858). Compared to SiamRPN ++ [6], our method achieves higher success (0.648 *vs.* 0.642) and precision scores (0.857 *vs.* 0.840). It is attributed to the localization branch and the multi-scale attention module introduced in our tracker.

VisDrone-SOT2019. VisDrone-SOT2019 [26] is a recent drone-captured dataset, formed by 167 video sequences with

TABLE IV

COMPARISONS WITH TOP 10 TRACKERS IN VISDRONE-SOT2019 CHALLENGE [26]

Tracker	Success score
ED-ATOM [8]	**63.5**
ATOMFR [8]	61.7
SMILE [6], [8]	59.4
Siam-OM [8], [40]	59.3
DR-V-LT [6]	57.9
SiamRPN++ [6]	56.8
TIOM [8]	55.3
PTF [8]	54.4
DATOM_AC [8]	54.1
ACNT [8]	53.2
SiamCAN*	**64.6**

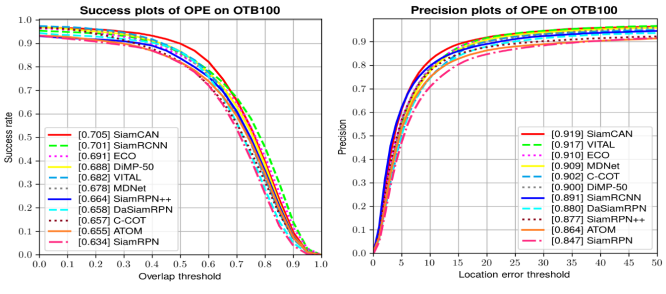


Fig. 7. Success and precision plots on the OTB100 dataset [21].

188, 998 frames in total. Specifically, the dataset includes 86 sequences with 69, 941 frames in total for training, 11 sequences with 7, 046 frames for validation and 60 sequences with 112, 011 frames for testing in the challenge. Compared to other application scenarios, drones bring new challenges to tracking methods, such as abrupt camera motion, small target, and view point changes. We compare our SiamCAN method to the top 10 methods[4] in the VisDrone-SOT 2019 challenge. As described in [26], ED-ATOM, ATOMFR, SMILE, Siam-OM, TIOM, PTF, DATOM_AC and ACNT are improved from ATOM [8], and the rest methods are the variants of SiamRPN ++ [6]. As presented in Table IV, our SiamCAN method achieves the best results with 64.6 success score. The second best performer ED-ATOM produces 1.1 lower success score than SiamCAN, and DR-V-LT (*i.e.*, the variant of SiamRPN ++) only produces 57.9 success score. The results indicate that our method performs well on drone-captured video sequences.

OTB100. OTB100 [21] consists of 100 video sequences with 11 visual attributes. We compare our SiamCAN method with existing trackers, shown in Fig. 7. Our method achieves the best performance in both success and precision scores. Although SiamRCNN [52] acquires the second best success rate of 0.701, its precision score is inferior than several compared methods. VITAL [34] achieves the second best precision score of 0.917 but much worse success score of 0.682. Compared to SiamRPN ++ [6], our method improves 0.009 in success score (*i.e.*, 0.705 *vs.* 0.696) and 0.004 in precision score (0.919 *vs.* 0.915).

[4]Please refer to [26] for the descriptions of the compared methods.

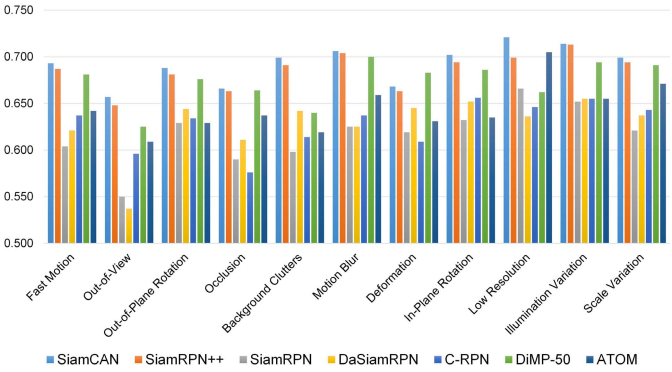


Fig. 8. Success score of the proposed method in each attribute on OTB100 [21].

In addition, we report the tracking performance of the proposed method based on the 11 attributes in OTB100 [21] in Fig. 8. Compared to the Siamese network based trackers, *i.e.*, SiamRPN ++ [6], SiamRPN [11], C-RPN [39] and DaSiamRPN [40], and other state-of-the-art methods, *i.e.*, DiMP-50 [64] and ATOM [8], our method performs the best in most of the attributes, especially in *fast motion*, *out-of-view*, *low resolution* and *background clutters*. Most of the previous Siamese network based trackers rely on the pre-set anchor boxes, making it difficult to adapt to different motion patterns of targets, resulting in inaccurate tracking results in challenging scenarios such as *fast motion* or *low resolution* (*i.e.*, indicating small scale target). The localization branch in the proposed method is effective to coarsely localize the target to help the regression branch to generate accurate results, making our tracker to be less sensitive to the variations of motion patterns and scales with the pre-set anchors.

B. Ablation Study

In Table V, we construct several variants to demonstrate the effectiveness of different components in SiamCAN. Notably, we use the same parameter settings and training data for a fair comparison. Meanwhile, we also present the qualitative results of the proposed SiamCAN method and its variants on 6 video sequences (*i.e.*, *Car1_2*, *Car1_3*, *Car4*, *Car18*, *Person7_1* and *Person19_1*) in Fig. 9, where the fast motion challenge occurs frequently in those sequences. The x-axis denotes the frame indexes of the video sequence, and y-axis denotes the IoU overlap score between the predicted bounding box and ground-truth box of the target. Besides, we analyze the effectiveness of multi-stage optimization and the fusion architecture design in our network.

Localization branch. To validate the effectiveness of the localization branch, we integrate the localization branch into the SiamRPN ++ method [6], named as "w/o context", in Table V. Compared with SiamRPN ++, the "w/o context" method improves the EAO scores from 0.464 to 0.488 on VOT2016 and 0.415 to 0.432 on VOT2018, respectively. As the qualitative evaluation results show in Fig. 9, we find that the "w/o context" method also improves the tracking accuracy. We speculate that the localization branch directly focuses on the localize the center of target, which is effective to help

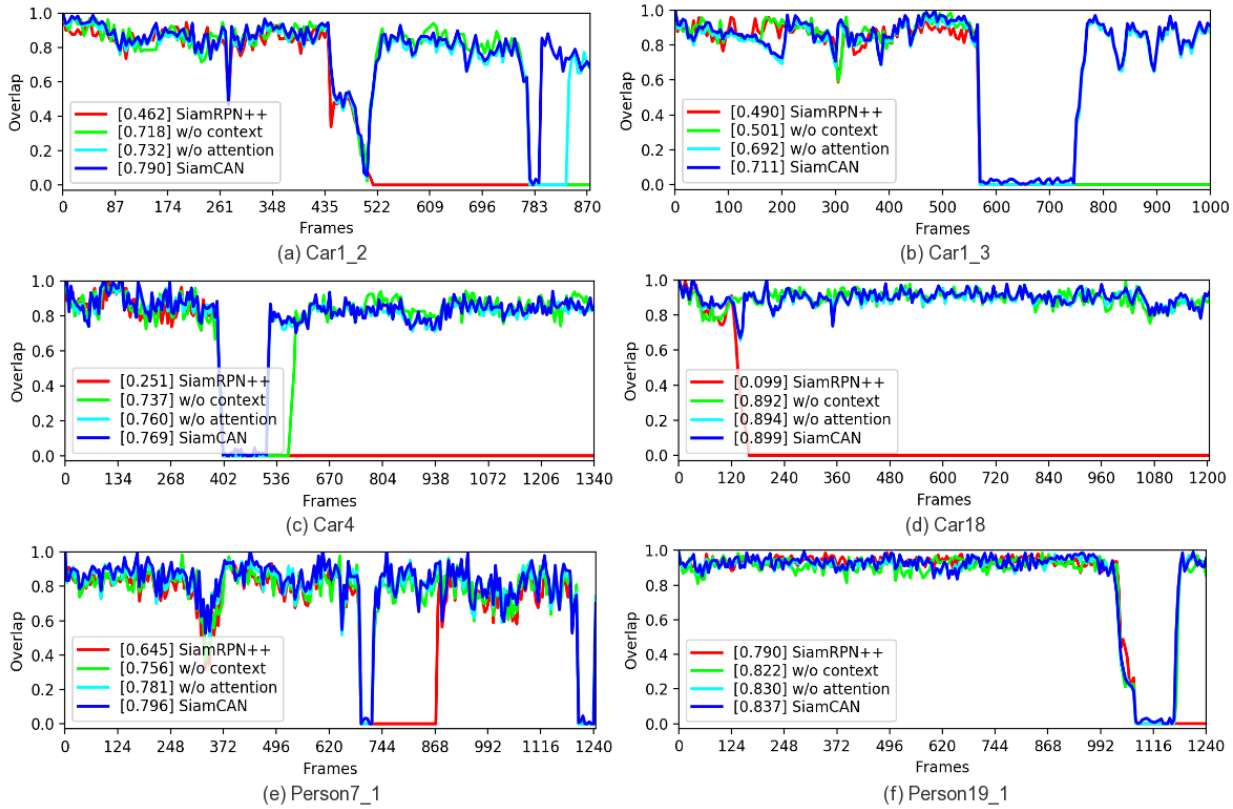


Fig. 9. Tracking results of six videos for different SiamCAN variants.

TABLE V

EFFECTIVENESS OF DIFFERENT COMPONENTS IN SIAMCAN BASED ON EAO

Component	SiamCAN					
localization branch?			✓	✓	✓	✓
global context?				✓		✓
multi-scale attention?		✓			✓	✓
VOT2016	0.464	0.472	0.488	0.494	0.504	**0.513**
VOT2018	0.415	0.421	0.432	0.446	0.447	**0.462**

TABLE VI

COMPARISON OF MULTI-STAGE OPTIMIZATION METHOD ON VOT2018 [19]

Optimization	A	R	EAO
one-stage	**0.615**	0.243	0.410
two-stage	0.604	**0.178**	**0.463**
three-stage	0.605	0.183	0.462

the classification and regression branches in SiamRPN ++ [6] to generate accurate results. The significant improvements of tracking accuracy demonstrate the effectiveness of the localization branch in the visual tracking task.

Global context module. To demonstrate the effectiveness of the global context module, we construct the "w/o attention" method by integrating the global context module into the "w/o context" method in Table V. As presented in the 4-th and 5-th columns in Table V, if we remove the global context module, the EAO scores of the "w/o attention" method drop 0.006 (0.494 *vs.* 0.488) on VOT2016 and 0.014 (0.446 *vs.* 0.432) on VOT2018, respectively. Meanwhile, as the qualitative evaluation results shown in Fig. 9, the tracking accuracy of the "w/o attention" method is improved slightly compared to the "w/o context" method. The results indicate that the global context module in the localization branch can capture long-range dependency to improve the tracking accuracy noticeably.

Multi-scale attention. We compare the "w/o attention" method to the proposed SiamCAN method to demonstrate the effectiveness of the multi-scale attention module. As shown in Table V, we find that the EAO score is significantly affected on the VOT2016 and VOT2018 datasets after removing the multi-scale attention module, *i.e.*, 0.494 *vs.* 0.513 EAO scores on VOT2016 and 0.446 *vs.* 0.462 EAO scores on VOT2018. Meanwhile, if we integrate the multi-scale attention module into the SiamRPN ++ method [6] (*i.e.*, the 3-rd column in Table V), the tracking accuracy is improved, *i.e.*, 0.472 *vs.* 0.464 EAO scores on VOT2016 and 0.421 *vs.* 0.415 EAO scores on VOT2018 respectively. As the qualitative evaluation in Fig. 9, our SiamCAN performs better that the "w/o attention" method. The learnable multi-scale attention module attempts to exploit the multi-scale discriminative features to guide the three branches, *i.e.*, the classification, regression, and localization branches, to produce accurate results. We can conclude that all of the three aforementioned modules are critical to the tracking performance in the proposed SiamCAN method.

Multi-stage optimization. To verify the effectiveness of multi-stage optimization strategy, we conduct several experiences on VOT2018 [19] in Table VI. Specifically, the training

TABLE VII
COMPARISON BETWEEN OUR SIAMCAN VARIANT AND RELATED SIAMESE TRACKERS

Trackers	Architecture	OTB100 [21]		UAV123 [25]	
		Success Score	Precision Score	Success Score	Precision Score
SiamBAN [45]	Anchor-free	0.696	0.910	0.631	0.833
SiamCAR [46]	Anchor-free	0.697	0.910	0.614	0.760
SiamRPN++ [6]	Anchor-based	0.664	0.877	0.612	0.804
SiamRCNN [52]	Anchor-based	**0.701**	0.891	**0.649**	0.834
SiamCAN	Fusion	**0.701**	**0.916**	0.640	**0.843**

TABLE VIII
COMPARISON OF THE COMPLEXITY OF TRACKERS

Trackers	Online Update	FPS	GFLOPs	# Params
SiamKPN [47]		28	76.07	90.75M
ATOM [8]	✓	28	3.97	17.29M
DiMP-50 [64]	✓	29	10.35	43.10M
Ocean [48]	✓	52	23.66	26.46M
SiamMask_E [12]		52	20.09	21.48M
SiamRPN++ [6]		53	59.56	53.95M
SiamBAN [45]		54	59.55	53.93M
SiamMask [12]		54	20.08	21.48M
DaSiamRPN [40]		96	21.85	90.44M
SiamCAN		45	64.62	61.14M

phase of our method is formed by three stages, called "three-stage" training strategy, *i.e.*, (1) training the classification and regression branches, (2) training all the three branches without the multi-attention module, and (3) training the whole network. The "one-stage" training indicates that we directly train the whole network, and the "two-stage" training indicates that we first train the classification and regression branches, and then fine-turn the whole network. As shown in Table VI, the "two-stage" and "three-stage" training strategy produce comparable results, and there is a sharp performance drop using the "one-stage" strategy. We speculate that the multi-stage strategy can obtain more optimal parameters of the network than the "one-stage" strategy in the training phase.

Fusion architecture design. As described in Section III, our network fuses the anchor-based classification and regression branches and anchor-free localization branch. To verify the effectiveness of our architecture design, we compare the results of our SiamCAN without using the multi-scale attention module to the other anchor-based and anchor-free trackers on the OTB100 [21] and UAV123 [25] datasets in Table VII. As shown in Table VII, we find that the variant of our SiamCAN produces better results than all of the anchor-based (*e.g.*, SiamRPN ++ [6] and SiamRCNN [52]) and anchor-free (*e.g.*, SiamBAN [45] and SiamCAR [46]) methods on the OTB100 [21] and UAV123 [25] datasets.

C. Complexity

We report the tracking speeds (FPS), computational complexity (GFLOPs) and model complexity (the number of parameters) of our SiamCAN and other algorithms (*e.g.*, DaSiamRPN [40], DiMP-50 [64], Ocean [48], and SiamRPN ++ [6]) on the VOT2018 dataset [19] in Table VIII. Note that all the trackers are run on the same machine for a fair comparison.

In terms of tracking speeds, DaSiamRPN [40] runs fastest with 96 FPS but produces inferior EAO score of 0.383. Ocean [48] acquires the real-time speed of 52 FPS and the best EAO score of 0.489. Comparing to SiamRPN ++ [6], our SiamCAN produces more accurate results (*i.e.*, 0.415 *vs.* 0.462 EAO scores) with a slightly lower running speed (*i.e.*, 53 *vs.* 45 FPS). Meanwhile, we notice that three previous trackers using the online updating scheme (*i.e.*, ATOM [8], DiMP-50 [64] and Ocean [48]) have lower GFLOPs than most of the Siamese network based methods. However, the online updating scheme requires additional run-time overhead, resulting in relative lower tracking speed. In summary, our SiamCAN achieves a good trade-off between the accuracy and running speed.

V. CONCLUSION

In this article, we propose a novel Siamese center-aware network for visual tracking, which integrates a new localization branch to deal with various motion patterns in complex scenarios. Specifically, it directly localizes the target center to help the regression branch generate accurate results and reduce tracking failures, especially when the fast motion, occlusion, and low resolution challenges occur. Our tracker sets the new state-of-the-art on three challenging tracking datasets, *i.e.*, VOT2016, OTB100, and VisDrone-SOT2019, and performs on par with the state-of-the-art on UAV123, with the real-time running speed 45 FPS.

REFERENCES

[1] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ECO: Efficient convolution operators for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6931–6939.
[2] F. Li, C. Tian, W. Zuo, L. Zhang, and M.-H. Yang, "Learning spatial-temporal regularized correlation filters for visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4904–4913.
[3] C. Sun, D. Wang, H. Lu, and M.-H. Yang, "Correlation tracking via joint discrimination and reliability learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 489–497.
[4] T. Xu, Z.-H. Feng, X.-J. Wu, and J. Kittler, "Learning adaptive discriminative correlation filters via temporal consistency preserving spatial feature selection for robust visual object tracking," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5596–5609, Nov. 2019.
[5] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese instance search for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1420–1429.
[6] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "SiamRPN++: Evolution of siamese visual tracking with very deep networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4282–4291.
[7] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr, "Fast online object tracking and segmentation: A unifying approach," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1328–1338.

[8] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ATOM: Accurate tracking by overlap maximization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4660–4669.

[9] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 472–488.

[10] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 FPS with deep regression networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 749–765.

[11] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8971–8980.

[12] B. X. Chen and J. K. Tsotsos, "Fast visual object tracking with rotated bounding boxes," *CoRR*, 2019.

[13] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "GCNet: Non-local networks meet squeeze-excitation networks and beyond," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 1971–1980.

[14] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989.

[15] T. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[16] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[17] E. Real, J. Shlens, S. Mazzocchi, X. Pan, and V. Vanhoucke, "YouTube-BoundingBoxes: A large high-precision human-annotated data set for object detection in video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7464–7473.

[18] M. Kristan *et al.*, "The visual object tracking VOT2016 challenge results," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2016, pp. 777–823.

[19] M. Kristan *et al.*, "The sixth visual object tracking VOT2018 challenge results," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2018, pp. 3–53.

[20] M. Kristan *et al.*, "The seventh visual object tracking VOT2019 challenge results," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops*, Oct. 2019, pp. 2206–2241.

[21] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1834–1848, Sep. 2015.

[22] H. Fan *et al.*, "LaSOT: A high-quality benchmark for large-scale single object tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5374–5383.

[23] P. Liang, E. Blasch, and H. Ling, "Encoding color information for visual tracking: Algorithms and benchmark," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5630–5644, Dec. 2015.

[24] A. Lukezic, L. C. Zajc, T. Vojír, J. Matas, and M. Kristan, "Now you see me: Evaluating performance in long-term visual tracking," *CoRR*, 2018.

[25] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for UAV tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 445–461.

[26] P. Zhu, L. Wen, D. Du, X. Bian, Q. Hu, and H. Ling, "Vision meets drones: Past, present and future," *CoRR*, 2020.

[27] D. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2544–2550.

[28] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 702–715.

[29] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[30] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2015, pp. 254–265.

[31] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3119–3127.

[32] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4293–4302.

[33] T. Yang and A. B. Chan, "Learning dynamic memory networks for object tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 152–167.

[34] Y. Song *et al.*, "VITAL: VIsual tracking via adversarial learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8990–8999.

[35] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional siamese networks for object tracking," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2016, pp. 850–865.

[36] X. Dong and J. Shen, "Triplet loss in siamese network for object tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 472–488.

[37] A. He, C. Luo, X. Tian, and W. Zeng, "A twofold siamese network for real-time object tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4834–4843.

[38] M. H. Abdelpakey and M. S. Shehata, "Domainsiam: Domain-aware siamese network for visual object tracking," in *Proc. Int. Symp. Vis. Comput.*, vol. 11844, 2019, pp. 45–58.

[39] H. Fan and H. Ling, "Siamese cascaded region proposal networks for real-time visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7952–7961.

[40] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 103–119.

[41] Z. Zhang and H. Peng, "Deeper and wider siamese networks for real-time visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4591–4600.

[42] G. Han, H. Du, J. Liu, N. Sun, and X. Li, "Fully conventional anchor-free siamese networks for object tracking," *IEEE Access*, vol. 7, pp. 123934–123943, 2019.

[43] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9626–9635.

[44] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *CoRR*, 2019.

[45] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6667–6676.

[46] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen, "SiamCAR: Siamese fully convolutional classification and regression for visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6268–6276.

[47] Q. Li, Z. Qin, W. Zhang, and W. Zheng, "Siamese keypoint prediction network for visual object tracking," *CoRR*, 2020.

[48] Z. Zhang, H. Peng, J. Fu, B. Li, and W. Hu, "Ocean: Object-aware anchor-free tracking," in *Proc. Eur. Conf. Comput. Vis.*, vol. 12366, 2020, pp. 771–787.

[49] Y. Zhang, D. Wang, L. Wang, J. Qi, and H. Lu, "Learning regression and verification networks for long-term visual tracking," *CoRR*, 2018.

[50] H. Lee, S. Choi, and C. Kim, "A memory model based on the siamese network for long-term tracking," in *Proc. Eur. Conf. Comput. Vis. Workshops*, vol. 11129, 2018, pp. 100–115.

[51] B. Yan, H. Zhao, D. Wang, H. Lu, and X. Yang, "'Skimming-perusal' tracking: A framework for real-time and robust long-term tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 2385–2393.

[52] P. Voigtlaender, J. Luiten, P. H. S. Torr, and B. Leibe, "Siam R-CNN: Visual tracking by re-detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6577–6587.

[53] K. Dai, Y. Zhang, D. Wang, J. Li, H. Lu, and X. Yang, "High-performance long-term tracking with meta-updater," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6297–6306.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[55] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5000–5008.

[56] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, Atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[57] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 765–781.

[58] A. Lukezic, T. Vojir, L. Zajc, J. Matas, and M. Kristan, "Discriminativecorrelation filter with channel and spatial reliability," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4847–4856.

[59] K. Dai, D. Wang, H. Lu, C. Sun, and J. Li, "Visual tracking via adaptive spatially-regularized correlation filters," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4670–4679.

[60] T. Yang, P. Xu, R. Hu, H. Chai, and A. B. Chan, "ROAM: Recurrently optimizing tracking model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6717–6726.

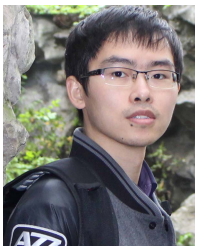
[61] J. Wang, Y. He, X. Wang, X. Yu, and X. Chen, "Prediction-tracking-segmentation," *CoRR*, 2019.

[62] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang, "Learning dynamic siamese network for visual object tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1781–1789.
[63] G. Bhat, J. Johnander, M. Danelljan, F. S. Khan, and M. Felsberg, "Unveiling the power of deep tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 493–509.
[64] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, "Learning discriminative model prediction for tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6182–6191.
[65] L. Huang, X. Zhao, and K. Huang, "GOT-10k: A large high-diversity benchmark for generic object tracking in the wild," *CoRR*, 2018.
[66] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese instance search for tracking," *CoRR*, 2016.
[67] Y. Li and X. Zhang, "SiamVGG: Visual tracking using deeper siamese networks," *CoRR*, 2019.
[68] J. Zhang, S. Ma, and S. Sclaroff, "MEEM: Robust tracking via multiple experts using entropy minimization," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 188–203.
[69] S. Hare *et al.*, "Struck: Structured output tracking with kernels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2096–2109, Oct. 2016.
[70] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1822–1829.

Wenzhang Zhou received the master's degree from the University of Electronic Science and Technology of China in 2018. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, University of Chinese Academy of Sciences, China. His research interests include visual tracking and object detection.

Longyin Wen received the B.Eng. degree in automation from the University of Electronic Science and Technology of China (UESTC) in 2010 and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences (CASIA) in 2015. He moved to the University at Albany, State University of New York, for postdoctoral research. He was a Principal Scientist in JD Finance America Corporation, Mountain View, CA, USA, and also a Computer Research Scientist in the AI and Image Analytics organization of GE Global Research, focusing on driving the application of cutting-edge deep learning and computer vision algorithms into several automated inspection projects, such as aviation engine service in GE Aviation and wind turbine inspection in GE Renewables. He is currently a Research Scientist with ByteDance, Inc., Mountain View. He published more than 50 articles in top conferences and journals with more than 2000 Google Scholar citations. His research interests include computer vision, machine learning, and deep learning.

Libo Zhang received the Ph.D. degree in computer software and theory from the University of Chinese Academy of Sciences, Beijing, China, in 2017. He is currently an Associate Research Professor with the Institute of Software, Chinese Academy of Sciences, Beijing. He is selected as a member of the Youth Innovation Promotion Association, the Chinese Academy of Sciences, and the Outstanding Youth Scientist of the Institute of Software, Chinese Academy of Sciences. His current research interests include image processing and pattern recognition.



Dawei Du received the B.Eng. and M.S. degrees from the University of Electronic Science and Technology of China, Chengdu, China, in 2010 and 2013, respectively, and the Ph.D. degree from the University of Chinese Academy of Sciences, Beijing, China, in 2018. He is currently a Postdoctoral Researcher with the University at Albany, State University of New York, Albany, NY, USA. His current research interests include visual tracking, object detection, and video segmentation.

Tiejian Luo received the Ph.D. degree in computer software and theory from the Graduate University of Chinese Academy of Sciences (UCAS), Beijing, China, in 2001. He is currently a Full Professor with the School of Computer Science and Technology, UCAS, and also he is also a Research Professor with the Institute of Software, Chinese Academy of Sciences. He is also the Director of the Information Dynamics and Engineering Application Laboratory, UCAS. His current research interests include web mining and deep learning.

Yanjun Wu received the B.Eng. degree in computer science from Tsinghua University in 2006 and the Ph.D. degree in computer science from the Institute of Software, Chinese Academy of Sciences (ISCAS), Beijing, China. He is currently a Research Professor with ISCAS, where he is also the Director of the Intelligent Software Research Center. His current research interests include computer vision, operating systems, and system security.