# SiamBAN: Target-aware Tracking with Siamese Box Adaptive Network

Zedu Chen, Bineng Zhong, Guorong Li, *Member, IEEE*, Shengping Zhang, Rongrong Ji, *Senior Member, IEEE*, Zhenjun Tang, *Member, IEEE*, Xianxian Li

**Abstract**—Variation of scales or aspect ratios has been one of the main challenges for tracking. To overcome this challenge, most existing methods adopt either multi-scale search or anchor-based schemes, which use a predefined search space in a handcrafted way and therefore limit their performance in complicated scenes. To address this problem, recent anchor-free based trackers have been proposed without using prior scale or anchor information. However, an inconsistency problem between classification and regression degrades the tracking performance. To address the above issues, we propose a simple yet effective tracker (named Siamese Box Adaptive Network, SiamBAN) to learn a target-aware scale handling schema in a data-driven manner. Our basic idea is to predict the target boxes in a per-pixel fashion through a fully convolutional network, which is anchor-free. Specifically, SiamBAN divides the tracking problem into classification and regression tasks, which directly predict objectiveness and regress bounding boxes, respectively. A no-prior box design is proposed to avoid tuning hyper-parameters related to candidate boxes, which makes SiamBAN more flexible. SiamBAN further uses a target-aware branch to address the inconsistency problem. Experiments on benchmarks including VOT2018, VOT2019, OTB100, UAV123, LaSOT and TrackingNet show that SiamBAN achieves promising performance and runs at 35 FPS.

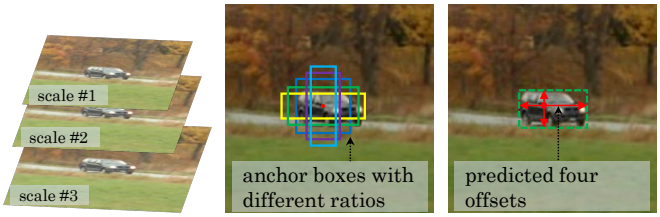
**Index Terms**—Visual tracking, fully convolutional network, anchor-free, no-prior box.

## 1 Introduction

Given the state of a target in the initial frame of a video, visual tracking, being one of the most challenging task in computer vision, aims to predict the states of the target in the subsequent frames. In the past decade, visual tracking has been attracting increasing attention due to its many potential applications, such as intelligent surveillance, human-machine interaction, robotics and autonomous driving.

In general, the state of the target is parameterized by a bounding box surrounding the target, i.e., coordinates of the top-left corner and the width and height. To obtain the accurate target states, a considerable amount of top-performing trackers with different characteristics have been proposed to address various challenges, such as illumination variations, background clutter, occlusions, fast motion, scale and appearance changes. Among these challenges, it is well-known that variation of scales or aspect ratios greatly challenges a tracker. However, most existing trackers adopt either comparatively simple multi-scale search or heuristic anchor-based schemes that may result in an inefficient search and degrade the tracking performance. Representative multi-scale search-based trackers (as shown in Figure 1 (a)) [1], [2], [3], [4], [5] are very time-consuming due to heavy image pyramid operations. In addition, they fail to obtain accurate scale estimation since a fixed aspect ratio is predicted. Recently, SiamRPN [6] introduces a region proposal network (RPN) into SiamFC [1]. By regressing the target region from the pre-defined anchor boxes (as shown in Figure 1 (b)), SiamRPN avoids the time-consuming multi-scale search and the predicted bounding box of the target is more accurate. Follow-up works such as DaSiamRPN [7], SiamRPN++ [8], SiamDW [9], SPM [10], C-RPN [11] improve SiamRPN through distractor-aware training, deeper network, multi-stage prediction, etc. However, to accurately estimate the bounding box of the target, these trackers rely on heuristic knowledge to carefully design hyper-parameters such as the numbers, sizes and aspect ratios of anchor boxes.

On the contrary, neuroscientists have shown that the bio-visual primary visual cortex can quickly and effectively ex-




(a) Multi-scale Search (b) Anchor-based (c) Anchor-free (ours)

Fig. 1. Different scale or aspect ratio handing methods: multi-scale search (such as SiamFC, ECO), anchor-based (such as SiamRPN, SiamRPN++), and anchor-free (such as ours) trackers.

- *Z. Chen, B. Zhong, Z. Tang, X. Li are with the Guangxi Key Lab of Multi-source Information Mining & Security, Guangxi Normal University, Guilin 541004, China. E-mail: zeduchen@gmail.com, bnzhong@gxnu.edu.cn, tangzj230@163.com, lixx@gxnu.edu.cn*
- *G. Li is currently an Associate Professor with the University of Chinese Academy of Sciences. E-mail: liguorong@ucas.ac.cn*
- *S. Zhang is currently a Professor with the School of Computer Science and Technology, Harbin Institute of Technology, Weihai, Shandong 264209, China. E-mail: s.zhang@hit.edu.cn*
- *R. Ji is currently a Professor with Media Analytics and Computing Lab, Department of Artificial Intelligence, School of Informatics, Xiamen University, 361005, China E-mail: rrji@xmu.edu.cn*

*(Corresponding authors: Bineng Zhong, Shengping Zhang and Xianxian Li.)*


Fig. 2. The tracking results predicted by the fusion scores (combined classification scores and target-aware scores) and classification scores. The red and blue bounding boxes correspond to the highest fusion and classification scores, respectively. Fus and Cls represent the fusion and classification scores, respectively.

tract the contours or boundaries of the observed objects from complex environments [12]. In other words, humans can recognize the locations and boundaries of objects without any pre-defined anchor boxes. So can we design an accurate and robust visual tracking framework without relying on any candidate boxes? Inspired by the anchor-free detectors [13], [14], [15], [16], [17], the answer is affirmative. To the best of our knowledge, we are among the pioneers [18], [19], [20] to utilize an anchor-free methodology to explore visual tracking without candidate boxes, which leads to competitive solutions [21], [22] to address the variations of scales or aspect ratios in tracking literature. The basic idea of the pioneering trackers [18], [19], [20] is to disassemble visual tracking into classification and regression sub-tasks. One of the limitations of these trackers lies in predicting the spatial scores to assist the classification without considering the quality of the predicted bounding boxes. In contrast, we observe that considering the quality of the predicted bounding boxes is important to associate classification and regression. Consequently, we propose a new anchor-free based tracker that is conceptually different and based on a new target-aware strategy. Specifically, by utilizing the expressive power of the fully convolutional network (FCN), we present a simple yet effective visual tracking framework named Siamese box adaptive network (SiamBAN) to tackle the challenge of accurately estimating the bounding boxes of the tracked target. The framework consists of a Siamese network backbone and multiple box adaptive heads, which can be optimized end-to-end during training. SiamBAN classifies a target and regresses its bounding box directly in a per-pixel prediction fashion (as shown in Figure 1), thereby transforming the tracking task into a classification-regression problem. SiamBAN directly predicts the target-background score and a 4D vector of each spatial position on the correlation feature maps. The 4D vector describes the relative offset from the centre point of the search region corresponding to the spatial position to the four sides of the bounding box.

In addition, we have observed that sometimes the classification scores are not proportional to the quality of the bounding boxes, i.e., the bounding box with the highest classification score may not be the optimal target state. As shown in Figure 2, if the optimal target states are chosen based only on classification scores, the green bounding boxes would be the winners. But apparently, the red bounding boxes are better options. Therefore, to have a better correspondence between the predicted scores and bounding boxes, we design a target-aware branch to estimate the quality of the predicted bounding boxes. The final tracking results are determined by a fusion score that combines both the classification and target-aware scores. In the cases of Figure 2, the red bounding boxes corresponding to the highest

fusion scores show that the classification and regression are more consistent. During inference, we search for the target in a window centered at the previous position. Through the bounding box corresponding to the highest fusion score, we can obtain the displacement and size change of the target between frames.

The main contributions of this work are three-fold.

- We design a simple yet powerful Siamese box adaptive network for addressing the variations of scales or aspect ratios in visual tracking. The no-prior box design in SiamBAN avoids hyper-parameters and complicated computation associated with the candidate boxes, making SiamBAN more flexible and fast.
- We propose a target-aware branch to effectively alleviate the inconsistency problem between classification and regression, and thus make SiamBAN more robust.
- SiamBAN not only achieves the state-of-the-art results, but also runs at 35 FPS on tracking benchmarks including VOT2018 [23], VOT2019 [24], OTB100 [25], UAV123 [26], LaSOT [27] and TrackingNet [28].

A preliminary version of this work has been presented in a conference [18]. Compared to the preliminary version, we make several extensions in this work. First, we add a target-aware branch to assist classification. It learns to evaluate the quality of the predicted bounding boxes by considering the IoU of the predicted ones and the ground truth. Second, we conduct experiments to validate whether the target-aware branch can effectively alleviate the inconsistency problem between classification and regression. The AUC on OTB100 [25], UAV123 [26], LaSOT [27] consistently increases from 0.696, 0.631, 0.514 to 0.702, 0.644, and 0.531, respectively. The same observation was made for VOT2018 [23], VOT2019 [24] and TrackingNet [28]. Third, we perform additional experiments and more analysis. Specifically, we show how much a backbone network, the post-processing operations, several weighting parameters and the classification branch affects the performance of our tracker. In addition, we systematically compare the new version of SiamBAN with the state-of-the-art trackers with more detailed results and comprehensive analysis, and add one more benchmark, i.e. TrackingNet [28].

## 2 Related Works

Visual tracking is one of the fundamental research topics in computer vision in recent decades. A comprehensive survey of the related trackers is beyond the scope of this paper, so we only briefly review two aspects that are most relevant to our work: Siamese network based visual trackers and anchor-free object detectors. Comprehensive surveys on visual tracking methods can be found in [29], [30].

### 2.1 Siamese Network Based Visual Trackers

Recently, Siamese network based trackers attract great attention from the visual tracking community due to their end-to-end training capabilities and high efficiency [1], [2], [3], [6], [7], [8], [9], [10], [11], [19], [20], [31], [32], [33], [34], [35], [36], [37], [38]. Typically, these trackers can be roughly divided into three categories, i.e., multi-scale search based trackers [1], [2], [3], [32], [34], [35], [36], anchor-based trackers [6], [7], [8], [9], [10], [11], [37], and anchor-free trackers [19], [20], [38].

**Multi-scale search based trackers:** SINT [31] treats the visual tracking problem as an image verification problem, and learns the similarity matching function through the Siamese network. SINT first crops the image patches in the adjacent area of the previous frame, and then finds the image patch closest to the target in the feature space for tracking. Although SINT uses the optical flow method and adaptive sampling strategy to reduce the number of candidate bounding boxes, the large number of candidate region sampling makes its tracking speed far from meeting of a real-time requirement. SiamFC [1] uses a Siamese network to extract features, and performs cross-correlation operations between the template and searched area to generate the response map. In the response map, the location with the maximum value is determined as the optimal location of the tracked target. Due to the fully convolutional structure and no model update, SiamFC can run at 86 FPS. However, SiamFC searches for the targets through a sliding window. Thus, it can only roughly predict the target scales by comparing the response values obtained from inputs of multiple scales.

Inspired by SiamFC, a series of trackers with or without online updating have been proposed. To enhance the online adaptive ability of SiamFC, some methods [32], [33] with online updating template features are proposed. CFNet [32] combines correlation filters and Siamese networks for end-to-end training. During online tracking, CFNet uses the updating skills of the correlation filter to fine-tune the network parameters, thereby improving the representation ability of the template features. Instead of directly replacing the template features with the features of the previous frames, DSiam [33] learns the transformation of the features from the first frame and previous frames. Thus, DSiam uses the features transformed from the first frame as the updated template. The transformed parameters can be solved quickly in the Fast Fourier Transform (FFT) domain to ensure the speed of DSiam.

On the other hand, to maintain the efficiency, some trackers [2], [3], [34], [35], [36] do not update their models during tracking. RASNet [2] believes that the online update of the network tends to overfit the targets and is computationally expensive. It uses spatial and channel attention mechanisms to enhance the representation power of the model. Based on the structure of SiamFC, SA-Siam [3] uses complementary semantic and appearance features to obtain two response maps. The final results are obtained through the weighted sum of the two response maps. The semantic features are directly used for pre-training weights on ImageNet. Meanwhile, the appearance features are trained with random initialization. In addition, SA-Siam also uses an attention mechanism on the channel to fuse the features from different layers of the convolutional networks. StructSiam [34] proposes a local structure learning method, which takes into account both the local pattern of a target and its structural relationship to effectively deal with the problem of non-rigid appearance change and partial occlusion of the target. SiamFC-tri [35] uses a triple loss to replace the logistic loss in SiamFC. Therefore, it can effectively learn the relationship

between samples and alleviate the imbalance of positive and negative samples during the training process. By combining pairwise positive and negative samples, the network is guided to learn more discriminative features. MemTrack [36] reads and writes historical target templates through a memory network controlled by LSTM (Long Short-Term Memory). The residual template obtained from the retrieved memory is firstly adapted to the appearance changes. Then, it is merged with the initial template from the first frame to get the final template of the Siamese network. By just performing forward operations during the inference process, MemTrack effectively avoids updating its model through time-consuming backpropagation. However, these trackers [1], [2], [3], [32], [33], [34], [35], [36] need a multi-scale search to cope with scale variation and cannot effectively handle aspect ratio changes caused by target appearance variations.

**Anchor-based trackers:** Recently, some authors rely on the anchor techniques to estimate a more accurate target bounding box. SiamRPN [6] introduces RPN [39] into SiamFC and uses the features of the two branches of a Siamese network to perform cross-correlation operations as the inputs of RPN. RPN is composed of a classification branch and a regression branch. The classification branch is used to classify whether the preset anchor boxes fit the tracking target. Meanwhile, the regression branch is used to adjust the position and size of the anchor boxes. On the basis of SiamRPN, DaSiamRPN [7] trains a model with a stronger discriminative ability by enriching training data and constructing negative samples with rich semantic information. To improve the performance of SiamRPN, SPM [10] designs a series-parallel matching structure, which uses a coarse-to-fine paradigm to localize a target. To obtain more accurate target localization, C-RPN [11] proposes a cascaded RPN multi-stage tracking framework, in which difficult negative samples are used to enhance the discriminative ability of the model. SiamRPN++ [8], SiamMask [37] and SiamDW [9] remove the influence factors such as padding in different ways. Compared to the trackers [6], [7] using AlexNet [40], the performance of these Siamese network based trackers is greatly improved by utilizing modern deep neural networks such as ResNet [41], ResNeXt [42] and MobileNet [43]. Although anchor-based trackers [6], [8], [9], [10], [11] can handle changes in scale and aspect ratio, one of their limitations is the need to carefully design and fix the parameters of the anchor boxes. This is because that designing parameters often requires heuristic adjustments and involves many tricks to achieve good performance.

**Anchor-free trackers:** In contrast to anchor-based trackers, a few of attempts have been made to utilize proposal-free and anchor-free methodology to improve the performance of a tracker. To the best of our knowledge, we are among the first (i.e., SiamBAN [18], SiamFC++ [19], SiamCAR [20]) to utilize the anchor-free methodology in tracking literature. SiamBAN, SiamFC++ and SiamCAR disassemble tracking into two sub-tasks, i.e., classification and state estimation. SiamFC++ and SiamCAR add a quality assessment branch similar to FCOS [17] to the classification branch to assist classification, while our tracker adds a target-aware branch to assist classification. The quality assessment branches of SiamFC++ and SiamCAR estimate the prior spatial scores, only considering the supervision information from ground-truth bounding boxes. Neglecting the predicted bounding boxes makes SiamFC+ and SiamCAR unable to connect classification and regression. Ocean [38] is another anchor-free based tracker that learns object-aware features to assist classification and uses an online learning method to update its model. However, its object-aware features and the features used for classification are sampled from different locations. Consequently, the different sampling locations may introduces the inconsistency and ambiguity between its object-aware and classification features, and thus degrade its performance. In this paper, we propose a new anchor-free based tracker that is conceptually different and based on a new target-aware strategy. Our tracker directly predicts the bounding boxes of a target in a per-pixel manner, which is efficient and flexible. To alleviate the inconsistency problem between classification and regression, our target-aware branch explicitly and effectively evaluates the quality of the predicted bounding boxes.

### 2.2 Anchor-free Object Detectors

Over the years, anchor-free methodology has attracted more and more attentions from object detection community. However, anchor-free detection is not a new concept. DenseBox [13] first introduces an FCN framework to jointly perform face detection and landmark localization. UnitBox [14] offers another option for performance improvement by carefully designing optimization losses. YOLOv1 [15] proposes to divide the input image into a grid and then predicts bounding boxes and class probabilities on each grid cell.

Recently, many new anchor-free detectors emerge. These detection methods can be roughly classified into keypoint based object detection [44], [45], [46] and dense detection [16], [17]. Specifically, CornerNet [44] proposes to detect an object bounding box as a pair of keypoints. ExtremeNet [45] presents to detect four extreme points and one center point of objects using a standard keypoint estimation network. RepPoints [46] introduces the representative points to represent a target. The representative points can capture fine-grained localization information and identify local areas significant for object classification. FSAF [16] proposes a feature selective anchor-free module to address the limitations imposed by heuristic feature selection for anchor-based single-shot detectors with feature pyramids. FCOS [17] proposes to directly predict the possibility of object existence and the bounding box coordinates without anchor reference.

Compared to object detection, there are two key challenges in the visual tracking task, i.e. unknown categories and discrimination between different objects. The anchor-free detectors usually assume the categories of the objects to be detected are pre-defined. However, the categories of the targets are unknown before tracking. Meanwhile, anchor-free detectors typically focus on detecting the objects from different categories, while in tracking, it is necessary to determine whether the two objects are the same one. Therefore, a template branch that can encode the appearance information is need in our framework to identify the target and background.

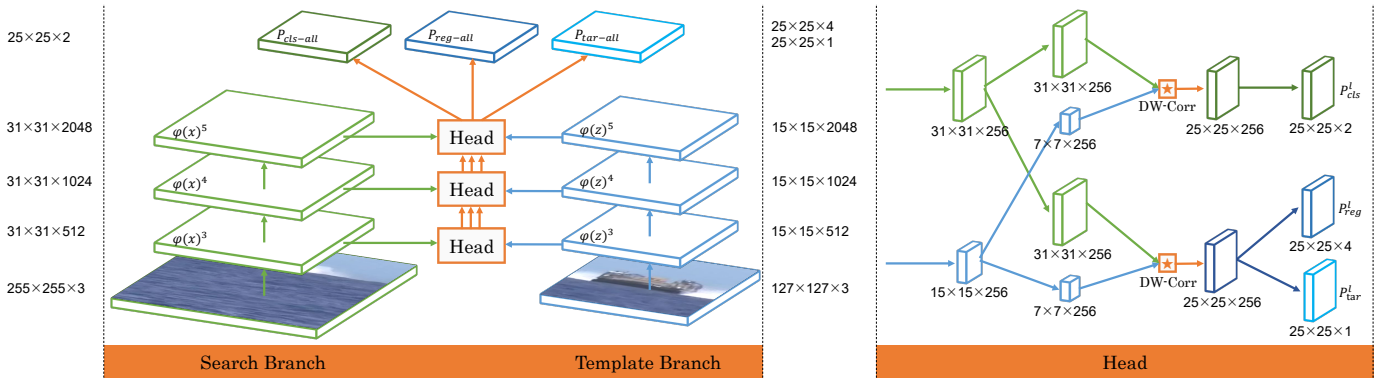



Fig. 3. The framework of the proposed Siamese box adaptive network. The left sub-figure shows its main structure, where $\varphi(x)^3$, $\varphi(x)^4$, $\varphi(x)^5$, $\varphi(z)^3$, $\varphi(z)^4$, and $\varphi(z)^5$ denote the feature maps of the backbone network. $P_{cls-all}$, $P_{reg-all}$, and $P_{tar-all}$ denote the classification, regression and target-aware map, respectively. The right sub-figure shows each SiamBAN head, where DW-Corr means depth-wise cross-correlation operation.

## 3 SiamBAN Framework

In this section, we describe the proposed SiamBAN framework. As shown in Figure 3, SiamBAN consists of a Siamese network backbone and multiple box adaptive heads. The Siamese network backbone is responsible for computing the convolutional feature maps of a template patch and a search patch, which uses an off-the-shelf convolutional network. The box adaptive head includes a classification module and a regression module. Specifically, the classification module performs target-background classification on each point of the correlation layers. Meanwhile the regression module performs the bounding box and target-aware score predictions on the corresponding positions, respectively.

### 3.1 Siamese Network Backbone

Modern deep neural networks [41], [42], [43] have been proven to be effective in Siamese network based trackers [8], [9], [37]. In our tracker, we adopt ResNet-50 [41] as the backbone network. Although ResNet-50 with continuous convolution striding can learn abstract feature representations, it reduces feature resolution. However, Siamese network based trackers need detailed spatial information to perform dense predictions. To deal with this problem, we remove the downsampling operations from the last two convolution blocks. To improve the receptive field, we use atrous convolution [47], which is proven to be effective for visual tracking [6], [37]. In addition, inspired by multi-grid methods [48], we adopt different atrous rates in our model. Specifically, we set the stride to 1 in the $conv4$ and $conv5$ blocks, the atrous rate to 2 in the $conv4$ block, and the atrous rate to 4 in the $conv5$ block, respectively.

The Siamese network backbone consists of two identical branches. One is called the template branch, which receives the template patch as input (denoted as $z$). The other one is called the search branch, which receives the search patch as input (denoted as $x$). The two branches share parameters in a convolutional neural network to ensure that the same transformation is applied to both patches. To reduce the computational burden, we add a $1 \times 1$ convolution to reduce the number of output feature channels to 256. Meanwhile, we only use the center $7 \times 7$ region features of the template branch [8], [32], which can still capture the entire target region. For convenience, the output features of the Siamese network are represented as $\varphi(z)$ and $\varphi(x)$, respectively. Table 1 shows the details of our backbone architecture.

TABLE 1
The backbone architecture of our SiamBAN. Details of each building block are shown in square brackets.

| Block | Backbone | Search Branch output size | Template Branch output size |
|---|---|---|---|
| conv1 | $7 \times 7$, 64, stride 2 | $125 \times 125$ | $61 \times 61$ |
| conv2_x | $3 \times 3$ max pool, stride 2 $\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$ | $63 \times 63$ | $31 \times 31$ |
| conv3_x | $\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$ | $31 \times 31$ | $15 \times 15$ |
| conv4_x | $\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$ | $31 \times 31$ | $15 \times 15$ |
| conv5_x | $\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$ | $31 \times 31$ | $15 \times 15$ |
| adjust | $1 \times 1$, 256 | $31 \times 31$ | $7 \times 7$ |
| xcorr | depth-wise | $25 \times 25$ | |

### 3.2 Box Adaptive Head

As shown in right panel of Figure 3, the box adaptive head consists of a classification module and a regression module. Both modules receive features from the template branch ($\varphi(z)$) and the search branch ($\varphi(x)$). According to our designing schema, each point of the correlation layers

of the classification module needs to output two channels for target-background classification. Meanwhile, each point of the correlation layers of the regression module needs to output four channels for predicting the target bounding boxes.

Specifically, $\varphi(z)$ and $\varphi(x)$ are respectively followed by a $1 \times 1$ convolution layer to obtain $[\varphi(z)]_{cls}$, $[\varphi(z)]_{reg}$ and $[\varphi(x)]_{cls}$, $[\varphi(x)]_{reg}$. Each module combines the features using a depth-wise cross-correlation layer [8]:

$$\begin{aligned} P_{cls-feat} &= [\varphi(x)]_{cls} \star [\varphi(z)]_{cls}, \\ P_{reg-feat} &= [\varphi(x)]_{reg} \star [\varphi(z)]_{reg}, \end{aligned} \tag{1}$$

where $\star$ denotes the convolution operation with $[\varphi(z)]_{cls}$ or $[\varphi(z)]_{reg}$ as the convolution kernel, $P_{cls-feat}$ denotes the classification features, $P_{reg-feat}$ indicates the regression features. $P_{cls-feat}$ is followed by a $1 \times 1$ convolution layer and four $3 \times 3$ convolution layers to output target-background classification map $P_{cls}$. $P_{reg-feat}$ is followed by a $1 \times 1$ convolution layer and four $3 \times 3$ convolution layers to output regression map $P_{reg}$.

For each location on the output maps, we can map it to the input search patch. For example, the location $(i,\ j)$ on the output map corresponding to the location on the search patch is $( \lfloor \frac{w_{im}}{2} \rfloor + (i - \lfloor \frac{w}{2} \rfloor) \times s,\ \lfloor \frac{h_{im}}{2} \rfloor + (j - \lfloor \frac{h}{2} \rfloor) \times s])$ denoted as $(p_i,\ p_j)$, where $w$ and $h$ represent the width and height of the output maps, respectively. $w_{im}$ and $h_{im}$ represent the width and height of the input search patches, respectively. $s$ represents the total stride of the network. For the regression, the anchor-based trackers [6], [7], [8] treat the location $(p_i,\ p_j)$ as the center of the anchor box and regress its center point, width and height. That is, for the position $(i,\ j)$, the regression can adjust all of its offset values. Consequently, the position $(i,\ j)$ may not be inside the predicted bounding box, which may result in inconsistencies between classification and regression. Therefore, we do not adjust the location $(p_i,\ p_j)$ and only calculate its offset to the bounding box. Make sure that the position $(p_i,\ p_j)$ is inside the predicted bounding box. In addition, since our regression targets are positive real values, we apply $exp(x)$ at the last level of the regression module to map any real value to $(0,\ +\infty)$.

### 3.3 Multi-level Prediction

After utilizing ResNet-50 with atrous convolution, we can use multi-level features for prediction. Although the spatial resolutions of the $conv3$, $conv4$ and $conv5$ blocks of our backbone network are the same, they have atrous convolutions with different expansion rates. Thus, the difference between their receptive fields is large, and the captured information is naturally complementary. Features from earlier layers can capture fine-grained information, which is useful for precise localization. Meanwhile, features from latter layers can encode abstract semantic information, which is robust to target appearance changes [49], [50]. To take full advantage of different characteristics of multi-level features, we use multiple box adaptive heads for prediction. The classification maps and the regression maps obtained by each head are adaptively fused:

$$\begin{aligned} P_{cls-all} &= \sum_{l=3}^{5} \alpha^l P_{cls}^l, \\ P_{reg-all} &= \sum_{l=3}^{5} \beta^l P_{reg}^l, \end{aligned} \tag{2}$$

where $\alpha^l$ and $\beta^l$ are the weights corresponding to each map, respectively. They are optimized together with the network. By combining the classification maps and the regression maps independently, they can effectively focus on their own sub-tasks.

### 3.4 Target-aware Tracking

After using multi-layer prediction, SiamBAN can achieve state-of-the-art performance. However, as shown in Figure 2, we find that in some cases, classification and regression are inconsistent, i.e., a higher classification score cannot choose a more accurate bounding box. We find that the reason is that during training, classification and regression are independent of each other. Therefore, the classification labels are not related to the predicted bounding boxes. In order to make the tracking scores consistently reflect the quality of the bounding boxes, we design a simple yet effective strategy to evaluate the quality of the predicted bounding boxes. Specifically, we add a target-aware branch parallel to the regression branch to estimate the IoU of the predicted bounding boxes and the ground truth. Therefore, $P_{reg-feat}$ is followed by a $1 \times 1$ convolution layer and four $3 \times 3$ convolution layers to output the regression map $P_{reg}$ and target-aware map $P_{tar}$. Except for the last convolution layer, other convolution layers are shared in our tracker. Similar to Equation (2), the target-aware maps obtained by each head are adaptively fused:

$$P_{tar-all} = \sum_{l=3}^{5} \gamma^l P_{tar}^l, \tag{3}$$

where $\gamma^l$ is the weight corresponding to each map. It is optimized together with the network.

### 3.5 Ground-truth and Loss

**Classification Labels, Regression Targets and Target-aware Labels.** As shown in Figure 4, the target on each search patch is marked with a ground-truth bounding box. The width, height, top-left corner, center point and bottom-right corner of the ground-truth bounding box are represented by $g_w$, $g_h$, $(g_{x_1},\ g_{y_1})$, $(g_{x_c},\ g_{y_c})$ and $(g_{x_2},\ g_{y_2})$, respectively. With $(g_{x_c},\ g_{y_c})$ as the center and $\frac{g_w}{2}$, $\frac{g_h}{2}$ as the axes length, we can get the ellipse $E_1$:

$$\frac{(p_i - g_{x_c})^2}{(\frac{g_w}{2})^2} + \frac{(p_j - g_{y_c})^2}{(\frac{g_h}{2})^2} = 1. \tag{4}$$

With $(g_{x_c},\ g_{y_c})$ as the center and $\frac{g_w}{4}$, $\frac{g_h}{4}$ as the axes length, we can get the ellipse $E_2$:

$$\frac{(p_i - g_{x_c})^2}{(\frac{g_w}{4})^2} + \frac{(p_j - g_{y_c})^2}{(\frac{g_h}{4})^2} = 1. \tag{5}$$

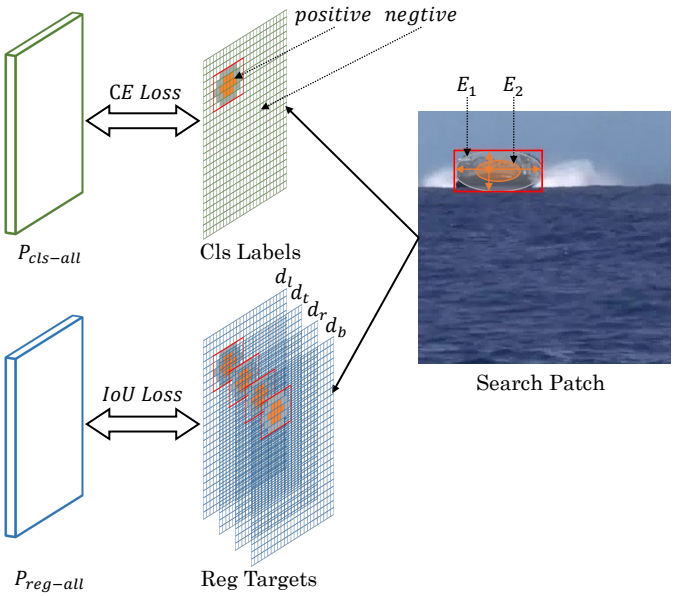



Fig. 4. Illustrations of classification labels and regression targets. Prediction values and supervision signals are as shown in this figure, where $E_1$ and $E_2$ represent the two ellipses, respectively. We use a cross entropy and an IoU loss for classification and box regression, respectively.

Given a location $(p_i,\ p_j)$, we determine its classification label according to three conditions. Firstly, its classification label is positive if the location $(p_i,\ p_j)$ falls within the ellipse $E_2$. Secondly, its classification label is negative if the location $(p_i,\ p_j)$ falls outside the ellipse $E_1$. Thirdly, its classification label is ignored if the location $(p_i,\ p_j)$ falls between the ellipses $E_2$ and $E_1$. Please note that the target-aware labels are not fixed. They are adaptively generated based on the IoU (Intersection over Union) of the predicted bounding boxes and the ground-truth bounding boxes. The locations $(p_i,\ p_j)$ with positive labels are used to regress the bounding box, and the regression targets can be formulated as:

$$\begin{aligned} d_l &= p_i - g_{x_1}, \\ d_t &= p_j - g_{y_1}, \\ d_r &= g_{x_2} - p_i, \\ d_b &= g_{y_2} - p_j, \end{aligned} \tag{6}$$

where $d_l$, $d_t$, $d_r$ and $d_b$ are the distances from the location to the four sides of the bounding box, as shown in Figure 4.

It should be noted that the target-aware labels are not fixed. They are adaptively generated based on the IoU of the predicted bounding boxes and the ground-truth bounding boxes.

**Classification Loss, Regression Loss and Target-aware Loss.** We define our multi-task loss function as follows:

$$L = \lambda_1 L_{cls} + \lambda_2 L_{reg} + \lambda_3 L_{tar}, \tag{7}$$

where $L_{cls}$ and $L_{tar}$ represent the cross entropy loss for the classification and target-aware branches, respectively. $L_{reg}$ represents the IoU Loss for the regression branch. We set $\lambda_1 = \lambda_2 = \lambda_3 = 1$, and the comparative experiments of different weights can be found in Section 4.3 on parameter sensitive analysis. Similar to GIoU [51], we define the IoU loss as:

$$L_{reg} = 1 - IoU, \tag{8}$$

where $IoU$ represents the area ratio of intersection to union of the predicted bounding box and the ground-truth bounding box. The location $(p_i,\ p_j)$ with a positive label is within the ellipse $E_2$ and the regression value is greater than 0. As a result, $0 < IoU \leq 1$, while $0 \leq L_{reg} < 1$. The IoU loss can make $d_l$, $d_t$, $d_r$ and $d_b$ jointly be regressed.

### 3.6 Training and Inference

**Training.** Our entire network can be trained end-to-end on large-scale datasets. We train SiamBAN with image pairs sampled on videos or still images. The training sets include ImageNet VID [52], YouTube-BoundingBoxes [53], COCO [54], ImageNet DET [52], GOT10k [55] and LaSOT [27]. The size of the template patch is $127 \times 127$ , while the size of the search patch is $255 \times 255$.

Compared to anchor-based trackers [6], [8], we use less negative samples. Please note that the number of negative samples is still much greater than that of positive samples in our tracker. In the experiments, we collect 64 samples from a pair of images. Among these samples, there are 16 positive samples, while the rest are negative samples.

**Inference.** During inference, we crop the template patch from the first frame and feed it to the feature extraction network. The extracted template features are saved. Thus, we do not have to calculate them in subsequent tracking process. For subsequent frames, we crop the search patches and extract features based on the target positions in the previous frames. Then, we perform prediction in the search regions to obtain the final classification map $P_{cls-all}$, the final regression map $P_{reg-all}$ and the final target-aware map $P_{tar-all}$, respectively. The final predicted scores are obtained by the weighted summation of the final classification scores and the final target-aware scores:

$$P_{score} = (1-\omega)P_{cls-all} + \omega P_{tar-all}, \tag{9}$$

where $P_{cls-all}$ denotes the predicted values of the final classification map and $P_{tar-all}$ the predicted values of the final target-aware map. In our experiments, the value of $\omega$ is set as 0.6, and the ablation study on the different values of $\omega$ can be found in Section 4.3.

In addition, we can get the predicted bounding boxes by the following equation:

$$\begin{aligned} p_{x_1} &= p_i - d_l^{reg}, \\ p_{y_1} &= p_j - d_t^{reg}, \\ p_{x_2} &= p_i + d_r^{reg}, \\ p_{y_2} &= p_j + d_b^{reg}, \end{aligned} \tag{10}$$

where $d_l^{reg}$, $d_t^{reg}$, $d_r^{reg}$ and $d_b^{reg}$ denote the prediction values of the regression map. $(p_{x_1},\ p_{y_1})$ and $(p_{x_2},\ p_{y_2})$ are the top-left corner and bottom-right corner of the predicted bounding boxes, respectively.

After the predicted bounding boxes are generated, we use the cosine window and scale change penalty to smooth target movements and changes [6]. Then, the predicted bounding box with the best score is selected and its size is updated by linear interpolation with the state in the previous frame. Algorithm 1 summarizes the tracking process of SiamBAN.

**Algorithm 1:** Tracking with SiamBAN

**Input:** frame sequences $\{X_{t=1}^{T}\}$ and ground-truth bounding box $b_1$ of $X_1$, trained model SiamBAN;

**Output:** Tracking results $\{b_{t=2}^{T}\}$;

Extract target template $z$ in $X_1$ using $b_1$;
Extract features $\{\varphi(z)\}_{l=3}^{5}$ for $z$ from SiamBAN;
Obtain features $\{[\varphi^l(z)]_{cls}\}_{l=3}^{5}$ and features $\{[\varphi^l(z)]_{reg}\}_{l=3}^{5}$ of $z$ from $\{\varphi^l(z)\}_{l=3}^{5}$;
**for** $t = 2$ *to* $T$ **do**
  Extract the search region $x$ in $X_t$ using $b_{t-1}$
  Extract features $\{\varphi^l(x)\}_{l=3}^{5}$ for $x$ from SiamBAN;
  Obtain features $\{[\varphi^l(x)]_{cls}\}_{l=3}^{5}$ and features $\{[\varphi^l(x)]_{reg}\}_{l=3}^{5}$ of $x$ from $\{\varphi^l(x)\}_{l=3}^{5}$;
  Obtain classification features $\{P_{cls-feat}\}_{l=3}^{5}$ and regression features $\{P_{reg-feat}\}_{l=3}^{5}$ using Equation (1);
  Obtain classification map $\{P_{cls}\}_{l=3}^{5} \leftarrow \{P_{cls-feat}\}_{l=3}^{5}$;
  Obtain regression map $\{P_{reg}\}_{l=3}^{5}$ and target-aware map $\{P_{tar}\}_{l=3}^{5} \leftarrow \{P_{reg-feat}\}_{l=3}^{5}$;
  Obtain final classification map $P_{cls-all}$ and final regression map $P_{reg-all}$ using Equation (2);
  Obtain final target-aware map $P_{tar-all}$ using Equation (3);
  Compute fusion score $P_{score}$ using Equation (9);
  Get predicted bounding boxes using Equation (10);
  Select the best bounding box as tracking result $b_t$;
**end**

## 4 Experiments

### 4.1 Implementation Details

We initialize our backbone networks with the weights pre-trained on ImageNet [52] and the parameters of the first two layers are frozen. Our network is trained with stochastic gradient descent (SGD) with a minibatch of 28 pairs. There are 20 epochs in total, using a warmup learning rate of 0.001 to 0.005 in the first 5 epochs and a learning rate exponentially decayed from 0.005 to 0.00005 in the last 15 epochs. In the first 10 epochs, we only train the box adaptive heads. In the last 10 epochs, we fine-tune the backbone network with one-tenth of the current learning rate. The weight decay and momentum are set to 0.0001 and 0.9, respectively. The initialized values of $\alpha^l$, $\beta^l$ and $\gamma^l$ are set to 1, 1 and 1, respectively. Our approach is implemented in Python using PyTorch on a PC with Intel Xeon(R) 4108 1.8GHz CPU, 64G RAM, Nvidia GTX 1080Ti. The code is available at https://github.com/hqucv/siamban.

### 4.2 Comparison with State-of-the-art Trackers

#### 4.2.1 Quantitative Evaluation

We compare our SiamBAN with the state-of-the-art trackers on six tracking benchmarks, i.e., VOT2018 [23], VOT2019 [24], OTB100 [25], UAV123 [26], LaSOT [27] and TrackingNet [28]. Our tracker achieves the state-of-the-art results and runs at 35 FPS.

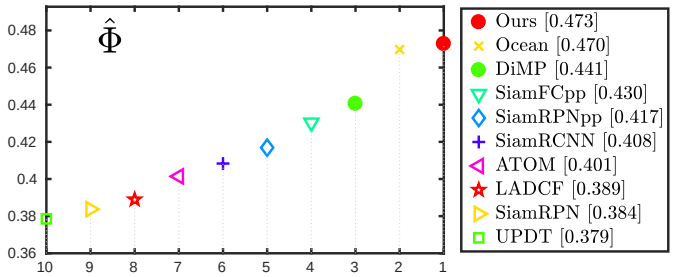



Fig. 5. Expected averaged overlap performance on VOT2018 [23]. SiamRPNpp is SiamRPN++, the same below.

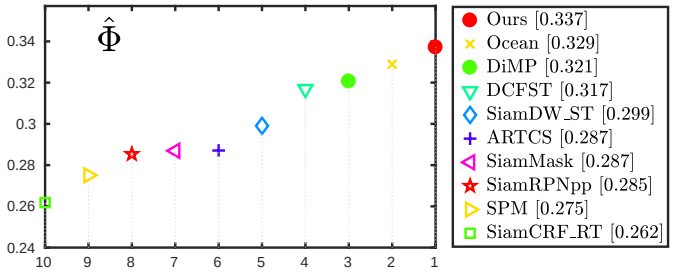



Fig. 6. Expected averaged overlap performance on VOT2019 [24].

**VOT2018 [23].** We evaluate our tracker on the Visual Object Tracking challenge 2018 (VOT2018), which consists of 60 video sequences with an average length of 356 frames. During the evaluation process, the tracker is initialized using the ground truth annotation in the first frame of a video sequence and re-initialized once it loses the targets. The overall performance of the tracker is evaluated using the EAO (Expected Average Overlap), which combines accuracy (average overlap during successful tracking periods) and robustness (failure rate). Figure 5 and Table 2 report the comparison results between our tracker and several top-performing trackers including UPDT [5], SiamRPN [6], LADCF [56], ATOM [57], Siam R-CNN [58], SiamRPN++ [8], SiamFC++ [19], DiMP [59], and Ocean [38]. Among these trackers, Siam R-CNN achieves the best accuracy, but its robustness is worse than ours. With online learning, DiMP obtains the best robustness. Our tracker, without any online learning, gains robustness close to DiMP. Compared with SiamRPN++, our tracker achieves similar accuracy. Meanwhile, it improves the robustness and EAO by 33.8% and 13.4% respectively. Our tracker has the highest EAO and ranks second in terms of robustness. As expected, our tracker not only accurately estimates the target location but also maintains good robustness. This is because our tracker can effectively handle the variations of scales or aspect ratios.

**VOT2019 [24].** We evaluate our tracker on Visual Object Tracking challenge 2019 (VOT2019). The VOT2019 sequences are replaced by 20% compared to the VOT2018. Figure 6 and Table 3 show the EAO, robustness and accuracy of our trackers, SiamCRF_RT [24], SPM [10], SiamRPN++ [8], SiamMask [37], ARTCS [24], SiamDW_ST [9], DCFST

TABLE 2
Detailed comparisons on VOT2018 [23]. The best two results are highlighted in red and blue fonts. DiMP is the ResNet-50 version (DiMP-50), Ocean is the offline Ocean, the same below.

| | UPDT [5] | SiamRPN [6] | LADCF [56] | ATOM [57] | Siam R-CNN [58] | SiamRPN++ [8] | SiamFC++ [19] | DiMP [59] | Ocean [38] | **Ours** |
|---|---|---|---|---|---|---|---|---|---|---|
| EAO(↑) | 0.379 | 0.384 | 0.389 | 0.401 | 0.408 | 0.417 | 0.430 | 0.441 | 0.470 | 0.473 |
| Accuracy(↑) | 0.536 | 0.588 | 0.503 | 0.590 | 0.617 | 0.604 | 0.590 | 0.597 | 0.603 | 0.598 |
| Robustness(↓) | 0.184 | 0.276 | 0.159 | 0.203 | 0.220 | 0.234 | 0.173 | 0.152 | 0.164 | 0.155 |

TABLE 3
Detailed comparisons on VOT2019 [24]. DiMP is realtime version, as reported in [24].

| | SiamCRF_RT [24] | SPM [10] | SiamRPN++ [8] | SiamMask [37] | ARTCS [24] | SiamDW_ST [9] | DCFST [24] | DiMP [59] | Ocean [38] | **Ours** |
|---|---|---|---|---|---|---|---|---|---|---|
| EAO(↑) | 0.252 | 0.262 | 0.275 | 0.285 | 0.287 | 0.287 | 0.299 | 0.317 | 0.321 | 0.337 |
| Accuracy(↑) | 0.549 | 0.577 | 0.599 | 0.594 | 0.602 | 0.600 | 0.585 | 0.582 | 0.595 | 0.604 |
| Robustness(↓) | 0.346 | 0.507 | 0.482 | 0.461 | 0.482 | 0.467 | 0.376 | 0.371 | 0.376 | 0.351 |

TABLE 4
Detailed comparisons on TrackingNet [28] in terms of AUC, precision (P) and normalized precision ($P_{norm}$).

| | ECO [4] | CFNet [32] | SiamFC [1] | UPDT [5] | MDNet [60] | DaSiamRPN [7] | UpdateNet [61] | ATOM [57] | DiMP [59] | **Ours** |
|---|---|---|---|---|---|---|---|---|---|---|
| AUC(↑) | 0.554 | 0.578 | 0.571 | 0.611 | 0.606 | 0.638 | 0.677 | 0.703 | 0.740 | 0.716 |
| P(↑) | 0.492 | 0.533 | 0.533 | 0.557 | 0.565 | 0.591 | 0.625 | 0.648 | 0.687 | 0.685 |
| $P_{norm}$(↑) | 0.618 | 0.654 | 0.663 | 0.702 | 0.705 | 0.733 | 0.752 | 0.771 | 0.801 | 0.794 |

[24], DiMP [59] and Ocean [38], respectively. Compared with Ocean, our tracker is better in terms of EAO, robustness and accuracy. Although SiamRPN++ achieves similar accuracy to our tracker, our tracker decreases robustness by 27.2% and increases EAO by 22.5%. Among these trackers, our tracker has the highest accuracy and EAO, which show that our tracker can accurately estimate the target bounding boxes. These results verify that the proposed target-aware branch can effectively alleviate the inconsistency problem between classification and regression, and thus makes our tracker more robust.

**Comparison of attributes on VOT2019.** All sequences of VOT2019 are per-frame annotated by the following visual attributes: camera motion, illumination change, occlusion, size change, and motion change. Frames that do not correspond to any of the five attributes are represented as unassigned. Compared with VOT2018, VOT2019 is more challenging. Thus, we further conduct attribute comparison experiments on VOT2019. We compare the EAO of the visual attributes of the top-performing trackers. As shown in Figure 7, our tracker achieves relatively more balanced results. Our tracker ranks first on attributes of camera motion. Our tracker ranks second on attributes of illumination change, occlusion, size change. This shows that our tracker is robust to camera motion while having the ability to cope with illumination change, occlusion, size change. In terms of motion change, except for DiMP, the performance of our tracker and other compared trackers are not promising. The reason is that the motion change typically leads to significant appearance changes or blur on VOT2019. However, for simplicity and computational efficiency reasons, we do not use any online learning algorithms to update our tracker.

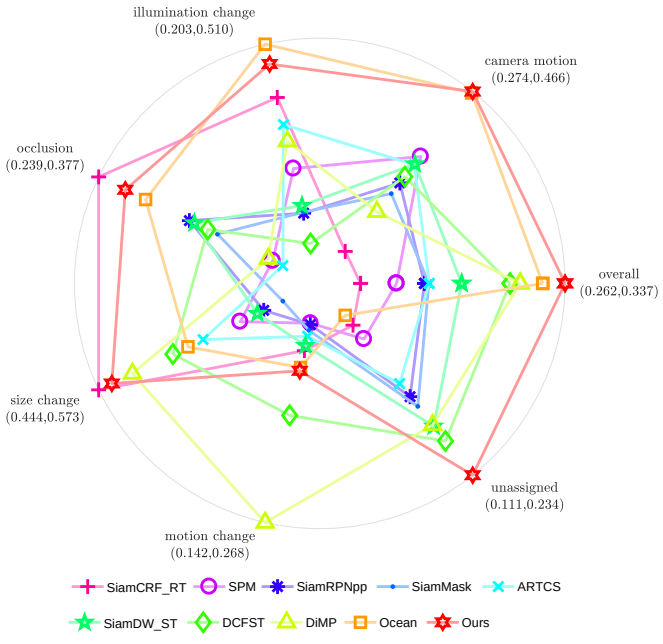
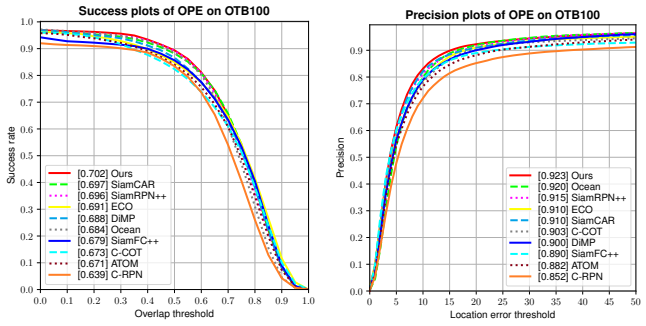


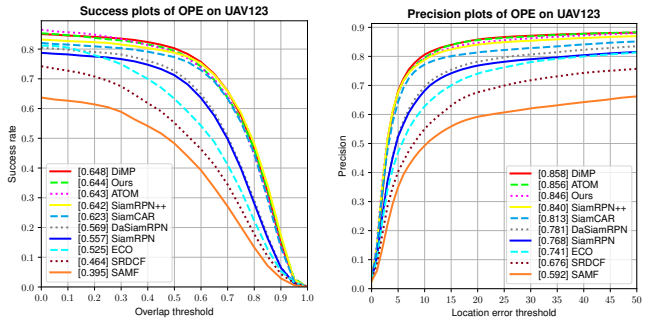

Fig. 7. Comparison of EAO on VOT2019 [24] for the following visual attributes: camera motion, illumination change, occlusion, size change and motion change. Frames that do not correspond to any of the five attributes are marked as unassigned. The values in parentheses indicate the EAO range of each attribute and overall of the trackers.

**OTB100 [25].** OTB100 is a widely used public tracking benchmark consisting of 100 sequences. Our SiamBAN




(a) Success Plot (b) Precision Plot

Fig. 8. Success and precision plots on OTB100 [25].

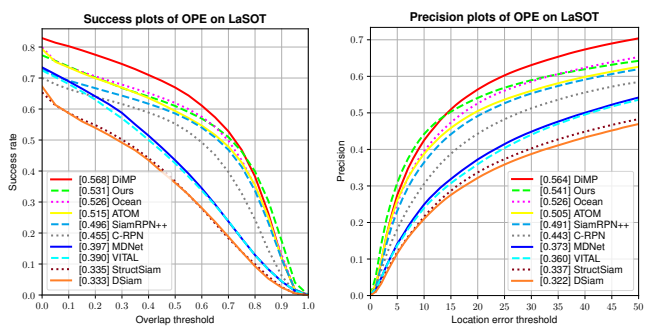



(a) Success Plot (b) Precision Plot

Fig. 9. Success and precision plots on UAV123 [26].

tracker is compared with numerous state-of-the-art trackers including SiamCAR [20], SiamRPN++ [8], ECO [4], DiMP [59], Ocean [38], SiamFC++ [19], C-COT [62], ATOM [57], C-RPN [11]. Figure 8 illustrates the success and precision plots of the compared trackers. Our tracker achieves better results than that of SiamRPN++. The benefit of our tracker comes from the non-prior box design in the proposed simple yet effective Siamese box adaptive network. Compared with other anchor-free trackers [19], [20], [38], our tracker performs better. The proposed SiamBAN ranks first in terms of AUC and precision, demonstrating the effectiveness of our tracker. These results indicate that our target-aware branch can provide more reliable bounding boxes predictions to alleviate the inconsistency problem between classification and regression in typical anchor-free style trackers.

**UAV123 [26].** UAV123 is an aerial video benchmark, which contains 123 sequences captured from a low-altitude aerial perspective. The benchmarks can be used to assess whether a tracker is suitable for deploying to a UAV in real-time scenarios. We compare our tracker with other 9 state-of-art real-time trackers, including DiMP [59], ATOM [57], SiamRPN++ [8], SiamCAR [20], DaSiamRPN [7], SiamRPN [6], ECO [4], ECO-HC [4], and SRDCF [63]. Figure 9 shows the success and precision plots. Due to having a better correspondence between the predicted scores and bounding boxes, our tracker outperforms a typical anchor-free tracker (i.e., SiamCAR) by 3.37% and 4.1% in AUC and precision, respectively. Overall, our tracker achieves competitive performance.




(a) Success Plot (b) Precision Plot

Fig. 10. Success and precision plots on LaSOT [27].

**LaSOT [27].** LaSOT is a high-quality, large-scale dataset with a total of 1,400 sequences. The length of sequences in LaSOT are longer than that of sequences in the previous datasets. An average sequence length in LaSOT is more than 2,500 frames. Each sequence has various challenges from the wild where a target may disappear and reappear in the view. This property will tests the ability of a tracker to re-track the target. We evaluate our tracker on the testing set consisting of 280 videos and compare it with trackers including DiMP [59], Ocean [38], ATOM [57], SiamRPN++ [8], C-RPN [11], MDNet [60], VITAL [64], StructSiam [34], DSiam [33]. The results including success plots and precision plots are illustrated in Figure 10. Compared with SiamRPN++, the proposed tracker improves AUC and precision by 7.1% and 10.2%, respectively. Our tracker ranks second in terms of AUC and precision, indicating that our tracker can achieve superior performance in longer and more challenging sequences. The robustness of our tracker lies in the systematically fusion of three key components, i.e., a classification module, a regression module, and a target-aware branch.

**TrackingNet [28].** TrackingNet is a large-scale benchmark for visual tracking in the wild for training and testing, including more than 30K videos with more than 14 million dense bounding box annotations. We evaluate our tracker on the testing set, which contains more than 500 videos without publicly available ground-truth. Table 4 shows the comparison results of our tracker with other trackers (i.e., ECO [4], CFNet [32], SiamFC [1], UPDT [5], MDNet [60], GFS-DCF [65], DaSiamRPN [7], UpdateNet [61] and ATOM [57]) in terms of AUC, precision and normalized precision, respectively. These experimental results are obtained through an online evaluation server. Among the compared trackers, DiMP achieves the best tracking results. Our tracker ranks second in terms of AUC, precision and normalized precision. Notably, this dataset has various categories of wild scenes, and thus the experimental results show that our tracker can achieve promising performance in the tested wild scenes.

### 4.2.2 *Qualitative Evaluation*

Figure 11 shows the qualitative comparison results on six challenging sequences (i.e., $Basketball$, $CarScale$, $Diving$, $Gym$, $Jump$, $Skating1$) from OTB100 [25], which contain


Fig. 11. Qualitative comparison on six challenging sequences from OTB100 [25], i.e., $Basketball$, $CarScale$, $Diving$, $Gym$, $Jump$, $Skating1$.

similar objects, scale variations, deformation, motion blur, and illumination variations, etc. For clarity of visualization, we only show the qualitative comparison results between our tracker and other five top-performing trackers, i.e., Ocean [38], DiMP [59], ECO [4], SiamRPN++ [8], and SiamCAR [20]. Overall, our tracker can achieve better performance in complex scenes. In the $Basketball$ sequence, due to the interference of similar athletes, except Ocean and our tracker, all the other trackers lose the target in frame #700. In the $CarScale$, the scale of a car significantly change. As a result, the ECO with a multi-scale searching schema cannot accurately estimate the scale of the car. In the $Diving$ and $Gym$ sequence, due to the different poses of the athletes, some trackers (i.e., Ocean, ECO, and DiMP) cannot generate accurate bounding boxes in frames such as frame #280 of sequence $Gym$. In the $Jump$ sequence, due to motion blur, neither SiamRPN++ nor ECO can accurately estimate the bounding boxes of the target. Meanwhile, our tracker can obtain accurate results. Due to similar dancers and illumination variations in $Skating1$ sequence, some trackers (i.e., DiMP and SiamRPN++) lost the targets. In frame #390, only SiamCAR and our tracker can accurately estimate the bounding boxes of the target.

#### 4.2.3 Limitations and Discussion

Despite our tracker has obtained competing performance compared with the state-of-the-art trackers, we observe that it does not perform well in case of fast motion and heavily occlusion, which are still opening challenges for tracking. Our failure cases from two typical sequences (i.e., $Ball3$ and $Agility$) are shown in Figure 12. For clearly illustration, we also show the results of other three top-performing trackers, i.e., Ocean [38], DiMP [59], SiamRPN++ [8]. For the case of fast motion, our tracker as well as the other three ones fails to localize the ball in the $Ball3$ sequence due to its fast motion. Similar to most existing trackers [8], [38], [59], our tracker chooses a fixed searching window size for simplicity and computational efficiency reasons. Thus, our tracker fails when the ball is outside our local searching area. An interesting extension would be to use a coarse-to-fine searching scheme or collect history frames to predict target motion. Firstly, for the coarse-to-fine searching scheme, we can online train a global detector to provide the coarse localization of the potential candidates at the coarse level. Then, at the fine level, we can use the SiamBAN to verify a local region centered on the predicted locations from the coarse stage. For the case of heavily occlusion, our tracker fails to track the dog in the $Agility$ sequence due

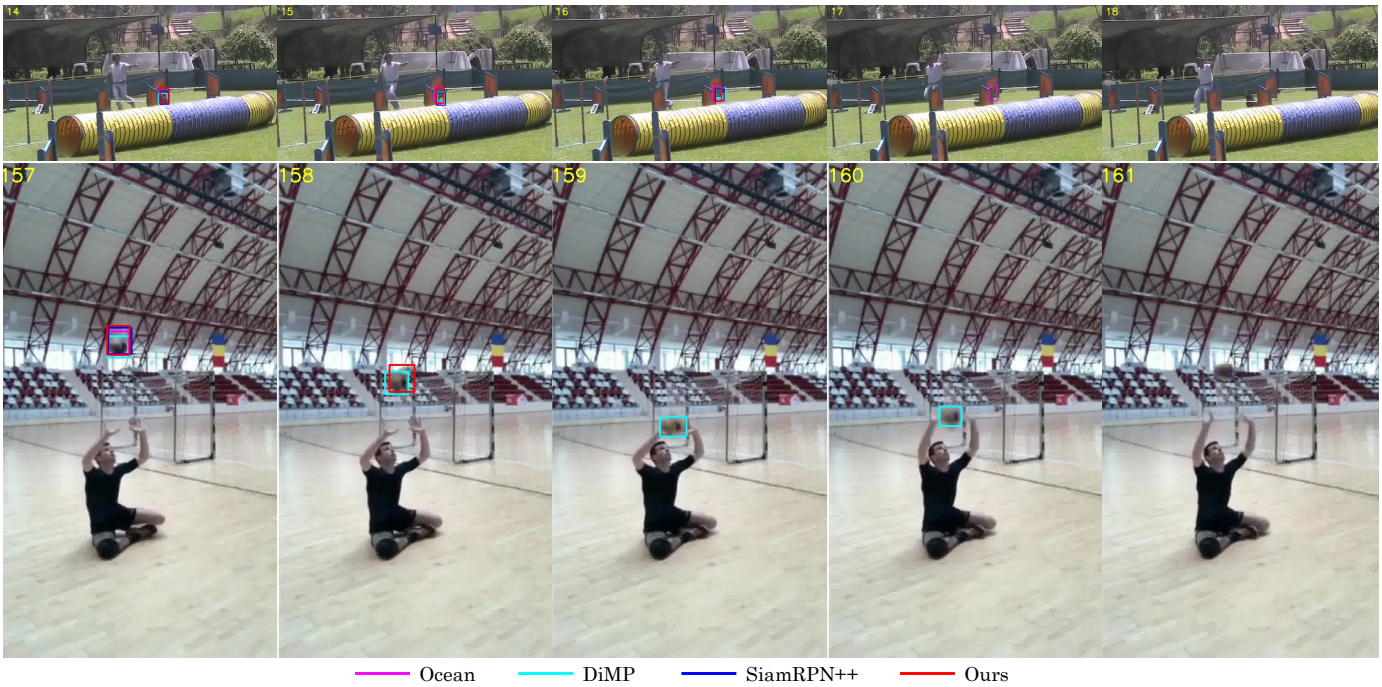



Fig. 12. Failure cases on two challenging sequences (i.e., $Agility$, and $Ball3$) from VOT2019 [24].

to the serious occlusion caused by the interfering objects. Like most Siamese network based trackers [6], [8], [19], we didn't introduce special design to deal with occlusion. One possible solution is to design special occlusion classifier [66] and re-detection module [67]. In this way, our tracker can judge whether the occlusion exists. Once the occlusion happens, our tracker can combine re-detection and motion prediction to re-localize the targets.

TABLE 5
The performance of our SiamBAN with different backbones as the feature extractors on OTB100 [25].

| Backbones | AlexNet | MobileNet-v2 | ResNet-18 | ResNet-34 | ResNet-50 |
|---|---|---|---|---|---|
| AUC (↑) | 0.669 | 0.673 | 0.672 | 0.684 | 0.702 |

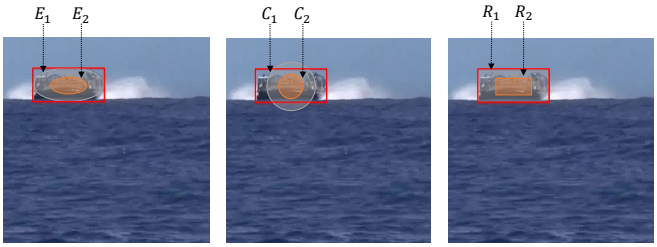
### 4.3 Ablation Study

In this subsection, we provide a diagnostic analysis of our SiamBAN. Due to space limitations, we only show the ablation study on OTB100 [25]. However, similar results can be obtained on other five datasets. Firstly, we show how much a backbone network affects the performance of our SiamBAN. Then, we investigate how much the multi-level prediction, sample label assignment, the target-aware branch and post-processing operations contribute to the overall tracking performance, respectively. Thirdly, we perform a sensitivity analysis on several weighting parameters in our multi-task loss function (i.e., Equation (7)) and a target-aware weighting parameter in our final predicted scores (i.e., Equation (9)). Fourthly, we investigate the impact of the classification branch on our SiamBAN. Finally, we compare the conference version [18] and the current version of SiamBAN.

**Backbone Architecture.** Since our SiamBAN is an open framework in which any backbone networks can be integrated and its success depends on the availability of good features, a following question arises: How sensitive is our SiamBAN to the representation power of a backbone network? To answer this question, we have tested difference backbones including AlexNet [40], MobileNet-v2 [43], ResNet-18 [41], ResNet-34 [41], ResNet-50 [41] on OTB100 [25]. The comparison results are shown in Table 5. When the depth of the network is gradually increased from AlexNet to ResNet-50, the AUC of the corresponding SiamBAN steadily improves from 0.669 to 0.702. Obviously, for different backbone networks, a good performance can be obtained across a wide range of backbone networks. The experiments have shown that the performance of our SiamBAN is impacted less by the different backbone networks. This property significantly eases the selection of a backbone network and other (potentially more efficient and robust) backbone networks may also be considered in the literature.

**Discussion on Multi-level Prediction.** To explore the role of different level features and the effect of aggregation of multi-level features, we perform an ablation study on multi-layer prediction. It can be found from Table 6 that when only single-layer feature is used, $conv4$ performs best. Compared with the single-layer features, when using the aggregation of the two-layer features, the performance has been improved. Among the different combination of the two-layer features, the performance of our tracker is best when aggregating both $conv4$ and $conv5$. Finally, after aggregating three layers of features, our tracker achieves the best results.

**Discussion on Sample Label Assignment.** The sample label assignment plays a key role in the performance of a




(a) Ellipse Labels (b) Circle Labels (c) Rectangle Labels

Fig. 13. Three sample label assignment methods: ellipse labels, circle labels, rectangular labels. $E_1$, $E_2$, $C_1$, $C_2$, $R_1$, $R_2$ represent ellipse $E_1$, ellipse $E_2$, circle $C_1$, circle $C_2$, rectangle $R_1$, rectangle $R_2$, respectively.

tracker. However, many Siamese network based trackers [1], [32], [33] do not pay enough attention to it. For example, SiamFC considers the elements of a score map within the radius $R$ of the center to be positive samples. Its label assignment method only considers the center position of the target, ignoring the size of the targets. Intuitively, the sample label assignment methods should be adaptively customized for targets with different sizes and shapes. Therefore, we design a label assignment method that takes into account the target scale and aspect ratio. Please note that we also set a buffer to ignore the ambiguous samples. Please see Section 3.5 for more details.

To illustrate the advantages of our label assignment method, we evaluate two variants of our tracker with the other two label assignment methods. As shown in Figure 13, for convenience, we refer to these three types of labels as ellipse, circle, and rectangle labels, respectively. For fair comparison, we define circles (i.e., $C_1$ and $C_2$ ) and rectangles ($R_1$ and $R_2$) in a similar way to define ellipses ($E_2$ and $E_2$). Specifically, with $(g_{x_c},\ g_{y_c})$ as the center and $\frac{\sqrt{g_w \times g_h}}{2}$, $\frac{\sqrt{g_w \times g_h}}{4}$ as the radius, we can get the circles $C_1$ and $C_2$. The position and size of the rectangle $R_1$ are the same as those of the ground-truth bounding box. The center of the rectangle $R_2$ is set to $(g_{x_c},\ g_{y_c})$. Meanwhile, its width and height are set to $\frac{g_w}{2}$ and $\frac{g_h}{2}$, respectively.

As shown in Table 6, with the same number of iterations and training datasets, our tracker outperforms its two variants with circle and rectangle labels. We believe that the reason is that ellipse labels can provide more accurately positive and negative samples than that of circular and rectangular labels. Benefit from the accurate samples, our tracker can effectively distinguish the targets from the backgrounds.

**Discussion on Our Target-aware Branch.** To test the contributions of the target-aware branch, we evaluate a degraded tracker of our trackers by removing the target-aware branch and verify its performance on OTB100. Specifically, we remove the target-aware branch in both the training and testing phases, and the degraded tracker is consistent with our conference version [18]. As shown in Table 6, the AUC score on OTB100 decreases from 0.702 to 0.696 when we remove the target-aware branch. This clearly validates the advantage of the proposed target-aware branch.

**Discussion on Post-processing Operations.** In our tracker, the post-processing step includes the cosine window penalty W, scale and aspect ratio penalty P, and bounding box smoothing S. To test how much they affect the tracking performance, we have added an ablation experiment on OTB100 as shown in Table 7. After completely turning off the post-processing step, our tracker has an AUC of 0.653. When our tracker adopts one post-processing operation (i.e., the cosine window penalty), its AUC significantly increases from 0.653 to 0.697. When our tracker uses two post-processing operations (i.e., the combination of the cosine window penalty and the bounding box smoothing), it obtains an AUC with 0.701. Our tracker achieves the best AUC with 0.702 when all three post-processing operations are used. This ablation analysis verifies the advantage of the post-processing operations for our tracker.

TABLE 6
Quantitative comparison results of our tracker and its variants with different detection heads and different label assignment methods on OTB100 [25]. L3, L4, L5 represent $conv3$, $conv4$, $conv5$, respectively. Circle, Rectangle, Ellipse, Target-aware represent circle labels, rectangle labels, ellipse labels, target-aware branch, respectively.

| L3 | L4 | L5 | Circle | Rectangle | Ellipse | Target-aware | AUC (↑) |
|---|---|---|---|---|---|---|---|
| ✓ | | | | | ✓ | | 0.675 |
| | ✓ | | | | ✓ | | 0.683 |
| | | ✓ | | | ✓ | | 0.662 |
| ✓ | ✓ | | | | ✓ | | 0.687 |
| ✓ | | ✓ | | | ✓ | | 0.681 |
| | ✓ | ✓ | | | ✓ | | 0.689 |
| ✓ | ✓ | ✓ | ✓ | | | | 0.686 |
| ✓ | ✓ | ✓ | | ✓ | | | 0.688 |
| ✓ | ✓ | ✓ | | | ✓ | | 0.696 |
| ✓ | ✓ | ✓ | | | ✓ | ✓ | 0.702 |

TABLE 7
AUC under different combinations of the post-processing operations on OTB100 [25]. W, S, P denote the cosine window penalty, scale and aspect ratio penalty, and bounding box smoothing, respectively.

| W | P | S | AUC (↑) |
|---|---|---|---|
| | | | 0.653 |
| ✓ | | | 0.697 |
| | ✓ | | 0.657 |
| | | ✓ | 0.659 |
| ✓ | ✓ | | 0.696 |
| ✓ | | ✓ | 0.701 |
| | ✓ | ✓ | 0.657 |
| ✓ | ✓ | ✓ | 0.702 |

**Parameter Sensitive Analysis.** Our SiamBAN contains some key weighting parameters in training and prediction stages, i.e., several weighting parameters in our multi-task loss function (i.e., Equation (7)) and a target-aware weighting parameter in our final predicted scores (i.e., Equation (9)). Thus, one may puts forward the following questions: (1) how the performance of our SiamBAN will be affected by the small changes of the parameter values; and (2) how to select the parameters. To answer the questions, we check the AUC for different parameter settings. We only change one parameter at a time due to too much combinations.

Specifically, to explore the effect of different weights in the multi-task loss function, we have tested the different weights on OTB100 as shown in Table 8, from which we

TABLE 8
AUC under different weighting parameter values of our multi-task loss function on OTB100 [25].

| $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | AUC (↑) |
|---|---|---|---|
| 0.5 | 1.0 | 1.0 | 0.688 |
| 1.0 | 0.5 | 1.0 | 0.693 |
| 1.0 | 1.0 | 0.5 | 0.694 |
| 1.0 | 0.5 | 0.5 | 0.696 |
| 0.5 | 1.0 | 0.5 | 0.688 |
| 0.5 | 0.5 | 1.0 | 0.689 |
| 1.0 | 1.0 | 1.0 | 0.702 |

TABLE 9
AUC under different values of target-aware weight $\omega$ on OTB100 [25].

| $\omega$ | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 |
|---|---|---|---|---|---|---|
| AUC (↑) | 0.681 | 0.688 | 0.692 | 0.702 | 0.693 | 0.682 |

TABLE 10
AUC on OTB100 [25] for different variants of our SiamBAN by modifying the classification branch.

| Trackers | AUC (↑) |
|---|---|
| SiamBAN | 0.702 |
| SiamBAN-V1 | 0.674 |
| SiamBAN-V2 | 0.683 |

TABLE 11
Comparison of conference version (SiamBAN$_{conf}$) and current version (SiamBAN).

| Trackers | VOT2018 EAO | VOT2019 EAO | OTB100 AUC | UAV123 AUC | LaSOT AUC | TrackingNet $P_{norm}$ |
|---|---|---|---|---|---|---|
| SiamBAN$_{conf}$ | 0.452 | 0.327 | 0.696 | 0.631 | 0.514 | 0.791 |
| SiamBAN | 0.473 | 0.337 | 0.702 | 0.644 | 0.531 | 0.794 |

can see that the best AUC score (0.702) can be obtained even simply setting them to 1, 1, and 1, respectively. In addition, the experiments have also illustrated that our multi-task loss function is not very sensitive to the small changes of the weight values. It is obvious that, for the weighting parameters, a good value can be chosen across a wide range of values. This property also show that a good set of weighting parameter values is easily obtained.

Moreover, to explore the impact of different target-aware weights on our SiamBAN, we have tested its performance with different values of $\omega$ (in Equation (9)) on OTB100 [25]. As shown in Table 9, the AUC of our SiamBAN gradually increases as the value of $\omega$ changes from 0 to 0.6. When the value of $\omega$ is set to 0.6, the AUC of our SiamBAN achieves the highest value of 0.702. Then, its performance decreases when $\omega$ continues to increase. When the value of $\omega$ is set to 0 or 1 (i.e., only $P_{cls-all}$ or $P_{tar-all}$ is used as the final predicted scores), our SiamBAN does not perform well, which follows our observations that both $P_{cls-all}$ and $P_{tar-all}$ make the contributions to our SiamBAN. This clearly validates that our simple yet effective fusing method in the prediction scores can improve the tracking performance.

**The Contributions from the Classification Branch.** To explore the impact of the classification branch of our SiamBAN, we have evaluated its two variants by modifying the classification branch and tested their performance on OTB100. The first variant of SiamBAN is denoted as SiamBAN-V1, in which the classification branch is removed from our SiamBAN in both the training and inference stages. The implemention of the SiamBAN-V1 is achieved by removing the upper branch on the right panel of Figure 3. Thus, SiamBAN-V1 only contains the regression and target-aware branches. The second variant of SiamBAN is denoted as SiamBAN-V2, in which the classification "arm" from the depth-wise cross-correlation onward (last two blocks in the upper green branch of Figure 3) is moved to the lower branch, i.e., attaching it to the output of the depth-wise cross-correlation in the lower "arm" during the training and inference stages. As shown in Table 10, SiamBAN-V1 without the classification branch is worse than the trackers with the classification branch, e.g., SiamBAN and SiamBAN-V2. The comparison results clearly verify that the classification branch makes the significant contributions to our SiamBAN. Moreover, our SiamBAN is better than SiamBAN-V2, which verifies that our SiamBAN can effectively combine the classification, regression and target-aware branches in a box adaptive head to make them focus on their own sub-tasks.

**Comparison of Different Versions of SiamBAN.** To facilitate the comparison between the conference version [18] and the current version of SiamBAN, we show their performance on the six challenging tracking benchmarks in Table 11. As shown in Table 11, we can see that the current version of SiamBAN consistently outperforms its conference version on each used benchmark in EAO, AUC or $P_{norm}$. Furthermore, compared to the conference version, the robustness (failure rate) of the current version of SiamBAN on VOT2018 and VOT2019 drops from 0.178 to 0.155 and from 0.396 to 0.351, respectively. These experimental results validate that our SiamBAN can improve its tracking performance by incorporating the target-aware branch to effectively alleviate the inconsistency problem between classification and regression.

## 5 Conclusions

In this paper, we exploit the expressive power of the fully convolutional network and propose a simple yet effective visual tracking framework named SiamBAN, which does not require a multi-scale searching schema and pre-defined candidate boxes. SiamBAN directly classifies objects and regresses bounding boxes in a unified network. Therefore, the visual tracking problem becomes a classification-regression problem. In addition, to have better correspondence between the classification scores and the regressed bounding boxes, SiamBAN further uses a target-aware branch to assist the classification branch to locate the target. Extensive experiments on six visual tracking benchmarks demonstrate that SiamBAN achieves the state-of-the-art performance and runs at 35 FPS, confirming its effectiveness and efficiency.

## ACKNOWLEDGMENTS

This work was supported by the Project of Guangxi Science and Technology (No. 2022GXNSFDA035079 and GuiKeAD21075030), the National Natural Science Foundation of China (No. 61972167, 61772494, 61872112), the Guangxi "Bagui Scholar" Teams for Innovation and Research Project, the Guangxi Collaborative Innovation Center of Multi-source Information Integration and Intelligent Processing, and the Guangxi Talent Highland Project of Big Data Intelligence and Application.

## REFERENCES

[1] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European Conference on Computer Vision*. Springer, 2016, pp. 850–865.

[2] Q. Wang, Z. Teng, J. Xing, J. Gao, W. Hu, and S. Maybank, "Learning attentions: residual attentional siamese network for high performance online visual tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4854–4863.

[3] A. He, C. Luo, X. Tian, and W. Zeng, "A twofold siamese network for real-time object tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4834–4843.

[4] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, "ECO: Efficient convolution operators for tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6638–6646.

[5] G. Bhat, J. Johnander, M. Danelljan, F. Shahbaz Khan, and M. Felsberg, "Unveiling the power of deep tracking," in *European Conference on Computer Vision*, 2018, pp. 483–498.

[6] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8971–8980.

[7] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *European Conference on Computer Vision*, 2018, pp. 101–117.

[8] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "SiamRPN++: Evolution of siamese visual tracking with very deep networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4282–4291.

[9] Z. Zhang and H. Peng, "Deeper and wider siamese networks for real-time visual tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4591–4600.

[10] G. Wang, C. Luo, Z. Xiong, and W. Zeng, "SPM-Tracker: Series-parallel matching for real-time visual object tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3643–3652.

[11] H. Fan and H. Ling, "Siamese cascaded region proposal networks for real-time visual tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7952–7961.

[12] D. Purves, G. J. Augustine, D. Fitzpatrick, W. C. Hall, A.-S. LaMantia, J. O. McNamara, and L. E. White, *Neuroscience. 4th*. Oxford University Press, 2008, vol. 857.

[13] L. Huang, Y. Yang, Y. Deng, and Y. Yu, "DenseBox: Unifying landmark localization with end to end object detection," *arXiv preprint arXiv:1509.04874*, 2015.

[14] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "UnitBox: An advanced object detection network," in *24th ACM International Conference on Multimedia*. ACM, 2016, pp. 516–520.

[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[16] C. Zhu, Y. He, and M. Savvides, "Feature selective anchor-free module for single-shot object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 840–849.

[17] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *IEEE International Conference on Computer Vision*, 2019, pp. 9627–9636.

[18] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6668–6677.

[19] Y. Xu, Z. Wang, Z. Li, Y. Yuan, and G. Yu, "Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines." in *AAAI*, 2020, pp. 12 549–12 556.

[20] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen, "Siamcar: Siamese fully convolutional classification and regression for visual tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6269–6277.

[21] S. Cheng, B. Zhong, G. Li, X. Liu, Z. Tang, X. Li, and J. Wang, "Learning to filter: Siamese relation network for robust tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4421–4431.

[22] W. Han, X. Dong, F. S. Khan, L. Shao, and J. Shen, "Learning to fuse asymmetric feature maps in siamese trackers," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 570–16 580.

[23] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey *et al.*, "The sixth visual object tracking VOT2018 challenge results," in *European Conference on Computer Vision*, 2018, pp. 0–0.

[24] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, R. Pflugfelder, J.-K. Kamarainen, L. Cehovin Zajc, O. Drbohlav, A. Lukezic, A. Berg *et al.*, "The seventh visual object tracking VOT2019 challenge results," in *IEEE International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[25] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, 2015.

[26] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *European Conference on Computer Vision*. Springer, 2016, pp. 445–461.

[27] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling, "LaSOT: A high-quality benchmark for large-scale single object tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5374–5383.

[28] M. Muller, A. Bibi, S. Giancola, S. Alsubaihi, and B. Ghanem, "TrackingNet: A large-scale dataset and benchmark for object tracking in the wild," in *European Conference on Computer Vision*, 2018, pp. 300–317.

[29] P. Li, D. Wang, L. Wang, and H. Lu, "Deep visual tracking: Review and experimental comparison," *Pattern Recognition*, vol. 76, pp. 323–338, 2018.

[30] S. M. Marvasti-Zadeh, L. Cheng, H. Ghanei-Yakhdan, and S. Kasaei, "Deep learning for visual tracking: A comprehensive survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[31] R. Tao, E. Gavves, and A. W. Smeulders, "Siamese instance search for tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1420–1429.

[32] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr, "End-to-end representation learning for correlation filter based tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2805–2813.

[33] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang, "Learning dynamic siamese network for visual object tracking," in *IEEE International Conference on Computer Vision*, 2017, pp. 1763–1771.

[34] Y. Zhang, L. Wang, J. Qi, D. Wang, M. Feng, and H. Lu, "Structured siamese network for real-time visual tracking," in *European Conference on Computer Vision*, 2018, pp. 351–366.

[35] X. Dong and J. Shen, "Triplet loss in siamese network for object tracking," in *European Conference on Computer Vision*, 2018, pp. 459–474.

[36] T. Yang and A. B. Chan, "Learning dynamic memory networks for object tracking," in *European Conference on Computer Vision*, 2018, pp. 152–167.

[37] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr, "Fast online object tracking and segmentation: A unifying approach," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1328–1338.

[38] Z. Zhang, H. Peng, J. Fu, B. Li, and W. Hu, "Ocean: Object-aware anchor-free tracking," in *European Conference on Computer Vision*, 2020.

[39] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[42] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1492–1500.

[43] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[44] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," in *European Conference on Computer Vision*, 2018, pp. 734–750.

[45] X. Zhou, J. Zhuo, and P. Krahenbuhl, "Bottom-up object detection by grouping extreme and center points," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 850–859.

[46] Z. Yang, S. Liu, H. Hu, L. Wang, and S. Lin, "RepPoints: Point set representation for object detection," in *IEEE International Conference on Computer Vision*, 2019, pp. 9657–9666.

[47] 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 Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[48] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, "Understanding convolution for semantic segmentation," in *2018 IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2018, pp. 1451–1460.

[49] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Robust visual tracking via hierarchical convolutional features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 11, pp. 2709–2723, 2019.

[50] Y. Qi, S. Zhang, L. Qin, Q. Huang, H. Yao, J. Lim, and M.-H. Yang, "Hedging deep features for visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 5, pp. 1116–1130, 2019.

[51] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 658–666.

[52] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[53] 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 *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5296–5305.

[54] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision*. Springer, 2014, pp. 740–755.

[55] L. Huang, X. Zhao, and K. Huang, "GOT-10k: A large high-diversity benchmark for ggeneric object tracking in the wild," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[56] T. Xu, Z.-H. Feng, X.-J. Wu, and J. Kittler, "Learning adaptive discriminative correlation ffilters via temporal consistency preserving spatial feature selection for robust visual object tracking," *IEEE Transactions on Image Processing*, 2019.

[57] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ATOM: Accurate tracking by overlap maximization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4660–4669.

[58] P. Voigtlaender, J. Luiten, P. H. Torr, and B. Leibe, "Siam r-cnn: Visual tracking by re-detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6578–6588.

[59] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, "Learning discriminative model prediction for tracking," in *IEEE International Conference on Computer Vision*, 2019.

[60] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4293–4302.

[61] L. Zhang, A. Gonzalez-Garcia, J. v. d. Weijer, M. Danelljan, and F. S. Khan, "Learning the model update for siamese trackers," in *IEEE International Conference on Computer Vision*, 2019, pp. 4010–4019.

[62] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *European Conference on Computer Vision*. Springer, 2016, pp. 472–488.

[63] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *IEEE International Conference on Computer Vision*, 2015, pp. 4310–4318.

[64] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, R. W. Lau, and M.-H. Yang, "VITAL: Visual tracking via adversarial learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8990–8999.

[65] T. Xu, Z.-H. Feng, X.-J. Wu, and J. Kittler, "Joint group feature selection and discriminative filter learning for robust visual object tracking," in *IEEE International Conference on Computer Vision*, 2019, pp. 7950–7960.

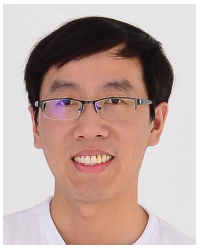
[66] M. Mathias, R. Benenson, R. Timofte, and L. Van Gool, "Handling occlusions with franken-classifiers," in *IEEE International Conference on Computer Vision*, 2013, pp. 1505–1512.

[67] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2011.


**Zedu Chen** received the M.S. degree in computer science from the Huaqiao University, Xiamen, China in 2021. He is currently a visiting scholar in Guangxi Normal University, Guilin, China. His research interests include computer vision and machine learning.

**Bineng Zhong** received the B.S., M.S., and Ph.D. degrees in computer science from the Harbin Institute of Technology, Harbin, China, in 2004, 2006, and 2010, respectively. From 2007 to 2008, he was a Research Fellow with the Institute of Automation and Institute of Computing Technology, Chinese Academy of Science. From September 2017 to September 2018, he is a visiting scholar in Northeastern University, Boston, MA, USA. From November 2010 to October 2020, he is a professor with the School of Computer Science and Technology, Huaqiao University, Xiamen, China. Currently, he is a professor with the Department Computer Science, Guangxi Normal University, Guilin, China. His current research interests include pattern recognition, machine learning, and computer vision.

**Guorong Li** received the B.S. degree in technology of computer application from the Renmin University of China, in 2006, and the Ph.D. degree in technology of computer application from the Graduate University of Chinese Academy of Sciences in 2012.



**Shengping Zhang** received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2013. He is currently a Professor with the School of Computer Science and Technology, Harbin Institute of Technology in Weihai. He had been a post-doctoral research associate with Brown University and with Hong Kong Baptist University, and a visiting student researcher with University of California at Berkeley. He has authored or co-authored over 60 research publications in refereed journals and conferences. His research interests include deep learning and its applications in computer vision.

**Rongrong Ji** is currently a Professor and the Director of the Intelligent Multimedia Technology Laboratory, and the Dean Assistant with the School of Information Science and Engineering, Xiamen University, Xiamen, China. His work mainly focuses on innovative technologies for multimedia signal processing, computer vision, and pattern recognition, with over 100 papers published in international journals and conferences. He serves as an Associate/Guest Editor for international journals and magazines such as Neurocomputing, Signal Processing, Multimedia Tools and Applications, the IEEE MultiMedia Magazine, and the Multimedia Systems. He also serves as program committee member for several Tier-1 international conferences.

**Zhenjun Tang** received the B.S. and M.Eng. degrees from Guangxi Normal University, Guilin, P.R. China, in 2003 and 2006, respectively, and the Ph.D. degree from Shanghai University, Shanghai, P.R. China, in 2010. He is now a professor with the Department of Computer Science, Guangxi Normal University. His research interests include image processing, video processing, and multimedia security. He has contributed more than 70 international journal papers. He is a reviewer of more than 30 SCI journals, such as IEEE journals, Elsevier journals and Springer journals.

**Xianxian Li** received the PHD degree in computer science and technology from Beihang University, Beijing, China. He is currently a professor with the School of Computer Science and Engineering, Guangxi Normal University. His research interests include machine learning, data security, blockchain and distributed system.