Instance-based Feature Pyramid for Visual Object Tracking

Zhixiong Pi, Yuanjie Shao, Changxin Gao, Nong Sang*
Key Laboratory of Image Processing and Intelligent Control, School of Artificial Intelligence and Automation, HuaZhong University of Science and Technology, Wuhan, China
pzxiong@hust.edu.cn, shaoyuanjie@hust.edu.cn, cgao@hust.edu.cn, nsang@hust.edu.cn

***Abstract*—The deep learning based methods have improved the visual tracking precision significantly. However, the background distraction and the high precise localization remain challenging problems. Despite that some methods have fused the deep and shallow layer features to solve these problems, the existing fusion methods are still too naive to take the advantage of both the deep and shallow layer features fully. In this paper, we propose a new adaptive feature fusion method, called the instance-based feature pyramid (IBFP) to obtain the discriminative high-resolution feature, which not only inherits the discriminative information from the deep layer feature but also keeps the high precision localization information of the shallow layer feature. For utilizing the deep and shallow features effectively, we design an instance-based upsampling (IBU) module to fuse them, and a compressed space channel selection (CSCS) module to re-weight the feature channels adaptively. We insert the IBU and CSCS modules in the Siamese tracker for end-to-end training and testing. By using the proposed IBU and CSCS modules, we fuse the deep and shallow features in a series manner. Experiments on large-scale benchmark datasets demonstrate that the proposed modules boost the capabilities of distinguishing the targets and the similar distractors and perform favorably against the state-of-the-art.**

***Keywords*—Siamese tracker, Feature fusion, Visual tracking**

I. Introduction

The recent years have witnessed increasing interest in developing visual tracking to advance various vision applications. Benefiting from the development of the deep convolutional neural network (CNN), the visual tracking precision has been improved significantly via the powerful CNN features. However, the background distraction and the high precise localization are still challenging problems for visual tracking. Different from the recognition tasks where target labels are fixed for offline training, visual tracking requires target object identification by only using the online initial frame. The target object may become background distractors in other sequences. It is very important to extract the discriminative instance-specific features for distinguishing the target and the distractors. At the same time, the high-resolution features are crucial for high precise localization. Generally, the deep layer features are more discriminative but with a lower resolution rate, while the shallow layer features have a higher resolution. Combining the deep features and the shallow features is an obvious method to improve the tracking precision. However, existing combination methods are very naive to simply concatenate the shallow and the deep layer features or use these features separately. These methods cannot take the advantages of both the discriminative deep feature and the high-resolution shallow feature sufficiently. By simply concatenating the deep and shallow features or using these features separately, the siamese tracker locates the target via the correlation response at the end of the pipeline. The deep feature cannot offer guidelines for the high-resolution shallow feature. Different from these methods, we construct a target-related feature pyramid to fuse the deep and shallow features for searching the current target in a coarse-to-fine manner. The discriminative deep feature can provide useful guidelines to help the tracker locates the target at a high-resolution level. The simple illustration of our feature fusion method is in Fig. 1.

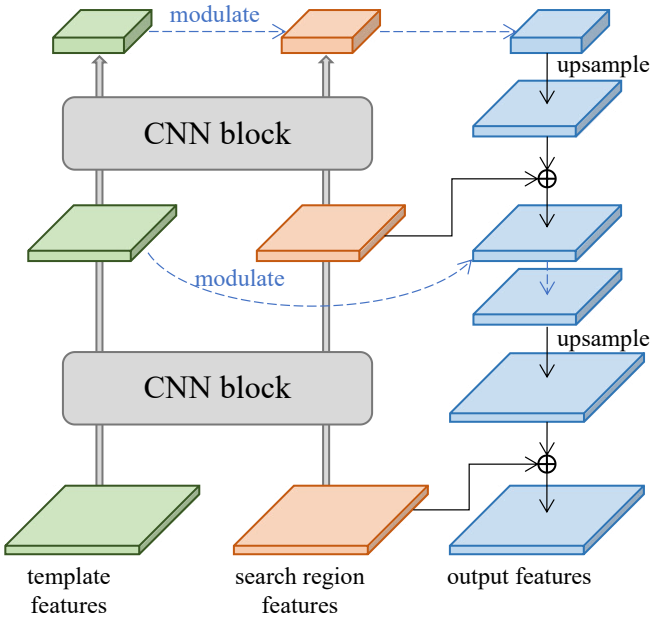

Fig. 1. The simple illustration of the proposed instance-based feature pyramid. The search region features are modulated by the template features and upsampled with a series manner. $\oplus$ is the feature fusion operator, which is concatenation in our experiments.

In this paper, we propose a new instance-based upsampling (IBU) method to fuse the shallow and the deep layer features, which maintain the high-resolution features, as well as the discriminative information. The IBU module is integrated into the Siamese network for end-to-end training and inference. We calculate the depth-wise correlation between the template

and the search region features at a network layer. Then, we concatenate the response tensor and the original search region feature and use a transpose convolutional layer to upsample the concatenated feature. After that, the upsampled feature is fused with the high-resolution feature to get the output of the IBU module. In our tracker, we insert the IBU modules between the features from different stages of the backbone. The final feature is obtained in a series manner.

For adapting to the different targets, we further propose a compressed space channel selection (CSCS) module. The CSCS module compresses the feature space first and calculates the channel weights in the compressed space via the cross attention between the template and the search region features. According to the depth-wise correlation response, the CSCS module can select the useful feature channels and suppress the noise channels.

We summarise the main contributions of this work as follows:

- We propose an efficient module of fusing the deep and shallow features adaptively according to the current instance, called the instance-based upsampling module, which can maintain the high-resolution features, as well as the discriminative information from the low-resolution features.
- We propose a compressed space channel selection module to re-weight the channels of the extracted features according to the depth-wise correlation response map, which can suppress the noise feature channels and emphasize the reliable channels.
- The experiments conducted on the benchmark datasets indicate the effectiveness of the instance-based upsampling module and the compressed space channel selection module in improving tracking accuracy. The proposed method performs favorably against the state-of-the-art.

II. Related work

Generic visual object tracking has a significant improvement in the last several years. Recently, Siamese network based trackers [2], [21], [43], [44], [36], [22], [10], [25], [42], [1] have raised much attention due to the good balance of accuracy and efficiency. In this section, we mainly discuss the Siamese trackers.

SiamFC [2] propose to use the fully-convolutional Siamese network to learn a matching metric between the target and the proposals offline, which achieves high efficiency and good accuracy. Due to the success of SiamFC [2], many extensions are proposed. Inspired by the region proposal network, SiamRPN [21] extracts region proposals and utilizes the bounding box regression to refine the tracking results for high accuracy localization. Compared with SiamFC [2], via combining the classifier and the regression module, SiamRPN [21] can speed up the inference stage significantly by avoiding the multi-scale testing process. Despite the improvement brought by the region proposal network, SiamRPN [21] tracker is influenced severely by the distractors. To discriminate the distractors and the target more effectively, some methods employ the hard negative mining during the training process like DaSiamRPN [46], and others use the deeper and wider backbones to extract the more discriminative features, such as SiamRPN++ [20] and DWSiam [43]. SiamRPN++ [20] replaces the AlexNet [19] backbone with the ResNet [14]. Besides, random shifting augmentation is used to suppress the center bias caused by the padding operation. Since then, very deep networks are widely used in the visual tracking task. SiamRPN++ [20] can achieve the state-of-the-art performance, but pre-setting anchors are needed, which increases the hyper-parameters. Many experiments prove the tracking results are severely related to these hyper-parameters of anchors. Inspired by FCOS [33] and CenterNet [45] in the detection, to reduce the anchor-related hyper-parameters and improve the tracking robustness, SiamCAR [12] employ the anchor-free regression branch to predict the distances from the points to the 4 sides of the bounding box. The anchor-free tracker is more flexible to the shape and scale changes. Danelljan et al. [4] employ the online conjugate gradient method to update the template filter and the offline trained IOU predictor to get the high accuracy bounding box. Bhat et al. [3] design and train the online optimizer based on the meta-learning strategy for the fast adaptation on the unseen test frames.

However, These works do not use both the shallow and deep features sufficiently. ATOM [4] only use the layer4 feature of ResNet to locate the target center. SiamCAR [12] and SiamRPN++ [20] modifies the ResNet backbone to keep the high resolution on deep layers, and simply concatenates features from different layers or uses these features separately. To maintain the high resolution on deep layers, the computation cost will increase significantly. Based on the observation above, we propose a new instance-based upsampling module to combine the shallow and deep features. The proposed module can use both the shallow and deep features effectively to locate the target in a coarse-to-fine manner. We further propose a compressed space channel selection method to adjust the channel weights of the features, which can suppress some useless channels and make the feature fusion more flexible.

III. Proposed Approach

In this section, we review the siameseFC framework first and present how the proposed module is integrated into the siamese tracking network. Then, we illustrate the details of the IBU and the CSCS modules, respectively. At last, we introduce the prediction head and the training loss.

A. Review of SiameseFC

Generally, siameseFC tracking network is separated into two parts, the feature extractor, and the correlation manipulation. The feature extractor $f(.)$ is a CNN model used to extract the features of the target template in the initial frame and the search region in the subsequent frames. Assuming the movement of the target is smooth between two adjacent frames, siameseFC tracker crops the search region $\mathbf{X}$ in the current frame centered at the target position of the last frame, which is larger than the size of the target patch. The target

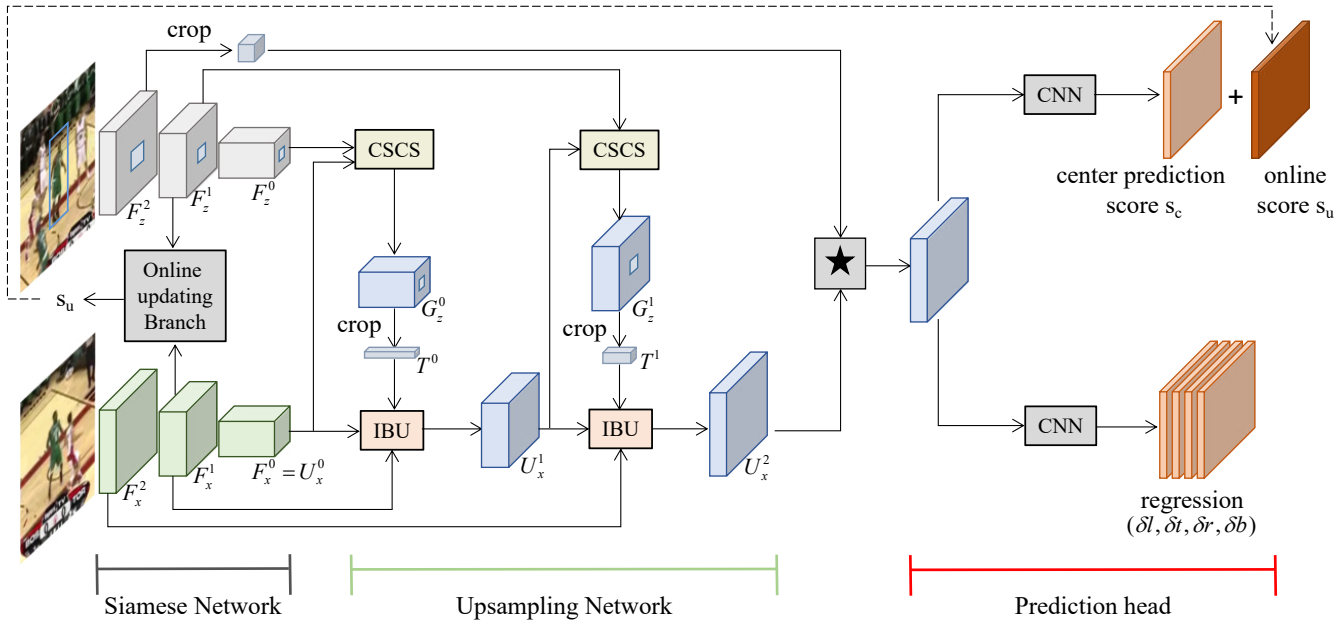

Fig. 2. The pipeline of our algorithm. We use the Siamese network as the backbone to extract the features of the template and the search region. The upsampling network is consists of several CSCS modules and IBU modules. After obtaining the upsampled search region feature, we calculate the depth-wise correlation response between the cropped template feature and the upsampled feature to get the response tensor. Based on this tensor, the prediction head with the center prediction branch and the regression branch outputs the center scores and the bounding boxes for every point.

patch with little background in the initial frame is cropped as the template $\mathbf{Z}$. First, the template feature and the search region feature are extracted via the same feature extractor as shown in Eq. 1.

$$\mathbf{F}_z = f(\mathbf{Z}), \mathbf{F}_x = f(\mathbf{X}). \tag{1}$$

Then, the similarity scores between the template and every position in the search region are calculated via the correlation manipulation:

$$\mathbf{S} = \mathbf{F}_z * \mathbf{F}_x. \tag{2}$$

The position with the maximum score is the predicted target center.

Considering the visual tracking task needs to distinguish different targets at the instance level, where both the shallow features and the deep features are useful, we design the IBU module to fuse the features from the different layers and locate the target in a series manner. We further insert the CSCS module before the IBU module to suppress some noisy channels. The whole pipeline of our model is shown in Fig. 2. Next, we will illustrate the IBU module and the CSCS module in detail.

B. Instance-based Upsampling

For the convenience of explanation, we only illustrate the first IBU module in Fig. 3. The other IBU module has similar architecture.

The IBU module takes the cropped deep template feature $\mathbf{T}^0 \in \mathcal{R}^{\mathrm{h}\times\mathrm{w}\times\mathrm{c}_0}$, the deep search region feature $\mathbf{U}_x^0 \in \mathcal{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{c}_0}$, and the shallow search region feature $\mathbf{F}_x^1 \in \mathcal{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{c}_1}$ as the inputs, and outputs the upsampled search region feature $\mathbf{U}_x^1 \in \mathcal{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{c}_1}$. It should be noticed that the input $\mathbf{U}_x^0$ of the first IBU module is $\mathbf{F}_x^0$, while the upsampled feature from the previous IBU module is used as the input for the second IBU module. First, the IBU module calculates the depth-wise correlation between the cropped deep template feature and the deep search region feature to get a response tensor $\mathbf{R}$:

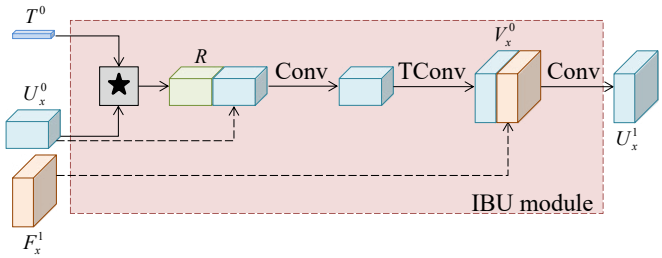

Fig. 3. The architecture of the proposed instance-based upsampling module (IBU). We calculate the depth-wise correlation response between the cropped template feature and the upsampled feature from the previous IBU module first. Then, we concatenate the response tensor and the input feature U_x^0. A convolutional layer is used to reduce the channel number by a half. $TConv$ represents the transpose convolutional layer, which is used to upsample the search region feature. In the end, we fuse the upsampled feature and the shallow layer feature by the concatenation operation and a convolutional layer.

$$\mathbf{R} = \mathbf{T}^0 *_{dw} \mathbf{U}_x^0, \tag{3}$$

where $*_{dw}$ represents depth-wise correlation. $\mathbf{R} \in \mathcal{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{c}_0}$ reflects the significance of the target at each channel. Then, the response tensor $\mathbf{R}$ is concatenated with the deep search

region feature $\mathbf{U}_x^0$ and sent to a convolutional layer to reduce the channel number. A transpose convolutional layer is used to obtain the upsampled feature $\mathbf{V}_x^0$:

$$\mathbf{V}_x^0 = TConv(Concat(\mathbf{R}, \mathbf{U}_x^0)), \tag{4}$$

where the $\mathbf{V}_x^0 \in \mathcal{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{c}_1}$ has the same dimension with the shallow search region feature $\mathbf{U}_x^0$. In the end, the IBU module fuses the upsampled feature and the shallow search region feature to get the output:

$$\mathbf{U}_x^1 = Conv(Concat(\mathbf{V}_x^0, \mathbf{F}_x^1)). \tag{5}$$

The IBU module fuses the deep layer feature and the shallow layer feature. The channel-wise response map from the deep layer is used to provide the instance information and light the meaningful parts of the search region. In our model, these IBU modules are stacked from the deep to the shallow layers, which filter out the background gradually and find the target in a coarse-to-fine manner.

C. Compressed Space Channel Selected

We take the second CSCS module as an example to introduce the operation of the CSCS module. The details of the module are shown in Fig. 4.

We design the CSCS module to re-weight the channels of the template feature according to the current search region. First, a global average pooling layer and a global max pooling layer are used to extract the average vector and the maximum vector of the search region. Then, the channel numbers of the average vector and the maximum vector are compressed by the fully connected layers. The compressed average vector and the compressed maximum vector are added to get the added compressed vector. Then, the channel dimension of the input template feature is also compressed by a convolutional layer. The added compressed vector is used as the channel weights to re-weight the channels of the compressed template feature. After re-weighting the compressed template feature, the CSCS module recovers its channel numbers via an extra convolutional layer. We mark the input template feature with some background as $\mathbf{F}_z^1 \in \mathcal{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{c}_1}$. $\mathbf{U}_x^1 \in \mathcal{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{c}_1}$ is the upsampled search region feature from the previous IBU module. $\mathbf{A}_z^1 \in \mathcal{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{d}_1}$ represents the adjusted feature in the compressed space. The output is $\mathbf{G}_z^1 \in \mathcal{R}^{\mathrm{H}\times\mathrm{W}\times\mathrm{c}_1}$. We can formulize the CSCS module as:

$$\mathbf{W} = conv(gap(\mathbf{U}_x^1)) + conv(gmp(\mathbf{U}_x^1)), \tag{6}$$

$$\mathbf{A}_z^1 = \mathbf{W} *_{dw} conv(\mathbf{F}_z^1), \tag{7}$$

$$\mathbf{G}_z^1 = conv(\mathbf{A}_z^1) + \mathbf{F}_z^1, \tag{8}$$

where $gap(.)$ and $gmp(.)$ represent the global average pooling operation and the global max pooling operation, respectively. $*_{dw}$ represents depth-wise correlation.

The CSCS module uses the global information of the current search region feature to re-weight the channels of the template feature, which generates a cross channel attention to suppress the noisy channels.

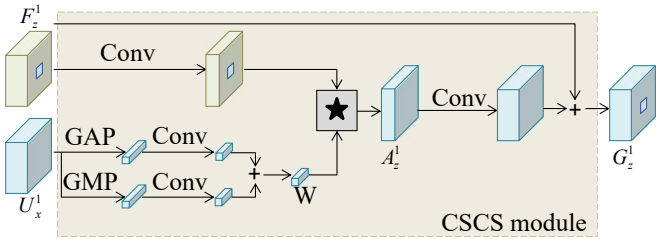

Fig. 4. The architecture of the proposed compressed space channel selection module (CSCS). We use the global average pooling (GAP) and the global max pooling (GMP) to squeeze the space dimension of the search region feature, which obtains the average vector and the maximum vector. Then, three convolutional layers are used to project the template feature, the average vector, and the maximum vector to a compressed space. in this compressed space, we add the average vector and the maximum vector first. Then, we calculate the depth-wise correlation between the template feature and the added vector to get the response tensor A_z^1. The response tensor is projected by another convolutional layer and added with the input template feature F_z^1.

D. Prediction Head

The prediction head has two branches. one branch outputs the center prediction score which represents how close the point nears the target center. A higher center score means the model predicts the corresponding point closer to the real target center. Another branch provides the bounding boxes prediction at each anchor point. The output of this branch is an $\mathrm{H} \times \mathrm{W} \times 4$ tensor whose four channels represent the distances from the corresponding anchor point to the four sides of the predicted bounding box.

During the training phase, we take the gaussian function as the center prediction label. Different from the SiamCAR [12] which uses the classification score and the centerness score to predict the target center, we locate the target center by only one response map. To suppress the influence of the simple negative samples, we use a focal hinge MSE loss to learn the center prediction. We define the focal hinge MSE loss,

$$\ell_c = \begin{cases} y_c^\gamma * |s_c - y_c| & , y_c \geq \tau \\ \alpha * \max(0, s_c) & , y_c < \tau, \end{cases} \tag{9}$$

where α and γ are used to adjust the importance of the negative samples and the positive samples, respectively. τ is the hinge loss threshold, which is used to suppress the influence of the simple negative samples. s_c and y_c are the center prediction score map and the center score label. By using the focal hinge MSE loss, we can balance the positive and negative samples and reduce the side effect from the large quantities of simple negative samples.

The bounding box regression labels are the distances from the anchor points to the four sides of the groundtruth box. We use the L1 loss to learn the regression,

$$\ell_b = \frac{1}{N}\sum_{i=1}^{N}\sum_{j}|p_{ij} - g_j|, j \in \mathrm{l}, \mathrm{t}, \mathrm{r}, \mathrm{b} \tag{10}$$

where p_i and g_i are the coordinates of the predicted bounding box and the groundtruth, respectively. $j \in \mathrm{l}, \mathrm{t}, \mathrm{r}, \mathrm{b}$ represents the left, top, right, and bottom coordinates. N is the total

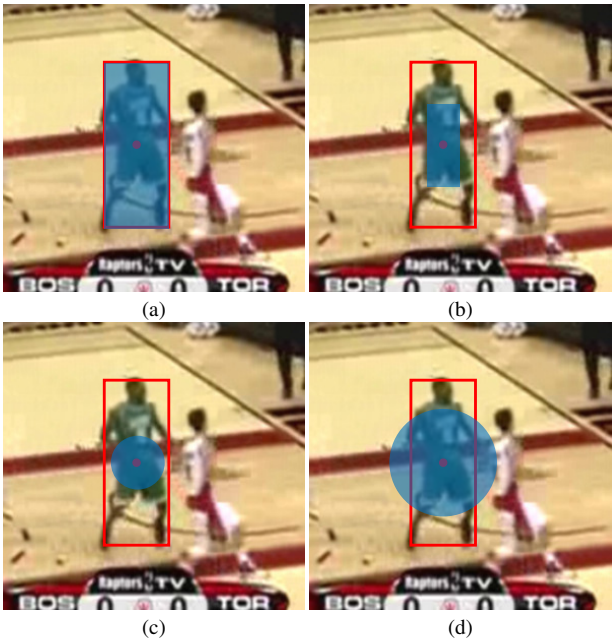

Fig. 5. Anchor points used to learn the bounding box regression. The red box and the red point are the groundtruth box and the target center. Anchor points inside the blue area are selected to train the bounding box regression. (a) shows the point selection method of SiamCAR [12]. (b) is our method to select the samples. (c), (d) are two kinds of compared methods.

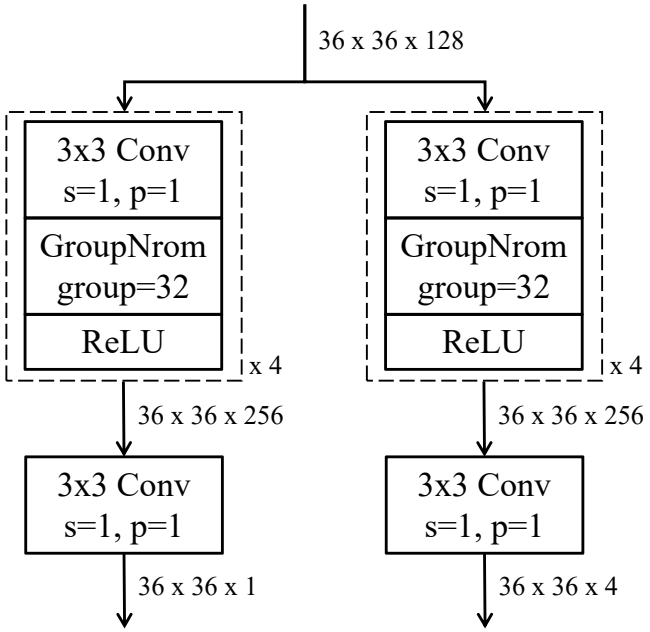

Fig. 6. The architecture of the prediction head. 's' and 'p' represent stride and padding parameters.

number of the samples for learning the bounding box regression. We define the regression samples according to the size of the groundtruth box. Instead of using all the anchor points inside the groundtruth box like SiamCAR, we argue that some points inside the groundtruth box are the background, especially when the points near the four sides of the box, so we only use the points near the target center to learn the bounding box regression. Considering the aspect ratio of the bounding box, we set two different distance thresholds for the horizontal and vertical directions, as shown in Fig. 5(b).

E. Online Updating

For adapting to the change of the target during the tracking process, the template feature should be updated. However, The updating can bring the accumulative error and cause tracking shift, so we need to keep the initial template to suppress the shifting. Inspired by the updating method of ATOM [4] tracker, we use an online branch to calculate the online response map. We update the online template every fixed frames or when hard negatives occur. The hinge MSE loss is used to supervise the template updating. We refer the readers to ATOM [4] for more details about the online updating. Then, the online response map and the center prediction map are mixed by a fixed weight:

$$s = w * s_c + (1 - w) * s_u, \tag{11}$$

where s_c and s_u are the center prediction map and the online response map, respectively. w is the mix weight.

IV. Experiments

In this section, we illustrate our implementation details and evaluate the proposed tracker IBFP. The evaluation process consists of ablation studies and state-of-the-art comparisons. The benchmark datasets we use are OTB-2013 [37], OTB-2015 [38], VOT-2019 [18], LaSOT [8], UAV123 [29], and GOT-10K [26] where there are 50, 100, 60, 280, 123, and 180 video sequences, respectively. We follow standard evaluation protocols for comparison. On the OTB-2013 [37], OTB-2015 [38], UAV123 [29], and LaSOT [8] datasets, we use distance precision (Prec.) rates at a threshold of 20 pixels and overlap success (AUC) rates. On the VOT-2018 [18] dataset, we use expected average overlap (EAO), accuracy values (Av) and robustness values (Rv). On the GOT-10K [26] dataset, we evaluate the performance via average overlap (AO), and success rates (SR) at overlap threshold of 0.5 and 0.75. Our implementation will be made available to the public.

A. Implementation Details

a) Network Architecture: We use the pre-trained ResNet18 [14] structure as the feature extraction backbone. The size of the template image is the same as the search region image, whose area is 5 times that of the target box. The input images are reshaped as $288 \times 288 \times 3$. Upsampling network fuses the features from layer5, layer4, and layer3 and outputs the $36 \times 36 \times 128$ upsampled feature. The proposed IBU module uses a transpose convolutional layer to upsample the low-resolution search region feature, which doubles the width and the height dimensions and reduces the channel number by half. The CSCS module compresses the channel dimension with a fixed compression ratio η, which is set as

0.5 in our experiments. The prediction head has two branches. Each branch is composed of 5 convolutional layers. The channel number of the hidden layers is 256. The outputs of the prediction head are the $36 \times 36 \times 1$ center prediction map and the $36 \times 36 \times 4$ regression map. The architecture of the prediction head is shown in Fig. 6.

b) Training Stratagy: The parameters of the proposed IBU module and CSCS module are set randomly in the initialization process. Then we train the whole model end-to-end. The training datasets we use are GOT-10K [26], TrackingNet [30], COCO [27], and LaSOT [8]. As the two branches of the siamese network share parameters, we only use one branch during training. We set the parameters α, γ, and τ of the focal hinge MSE loss as 0.9, 0.1, and 0.05 in our experiments. The siamese network is trained on the training video sequences by 60 epochs with an ADAM solver with a learning rate decay of 0.2 every 15 epochs. The beginning 5 epochs are the warm-up process. An epoch includes 1000 iterations. The training batch size is set to 128. It takes 20 hours to train the model with four Nvidia GTX TITAN X GPUs.

c) Online Tracking: The proposed algorithm fine-tunes the template feature according to the initial frame. The offline branch always uses the initial template to match the target. The online branch that is consists of four convolutional layers updates the online template every 10 frames or when the hard negative samples occur. The channel numbers of the hidden layers are 256. The features from layer4 are used by the online branch. We extract the search region features centering at the predicted target box center of the previous frame. The final response map is the weighted addition of the offline center prediction map and the online response map whose weights are 0.3 and 0.7, respectively. For adding the offline and online maps, we upsample the online response map to the same size as the offline center prediction map first. The average speed for online tracking is about 50 FPS with one Nvidia GTX TITAN X GPU.

B. Ablation Studies

We do ablation studies on the OTB-2015 [38] dataset. Four configurations are compared in Table. I. The first model does not use the IBU module and the CSCS module, and only extracts the layer4 features for the center prediction and the regression. Compared with the first configuration, the second model upsamples and concatenates the features from layer3, layer4, and layer5 as the input of the prediction head. The third model integrates the IBU modules, but without the CSCS module in it. The fourth model is our final method, which integrates both the IBU modules and the CSCS modules.

When the tracker only uses the layer4 features to predict the target center and the bounding box, the AUC score and the precision is 65.2% and 83.3%, respectively. The simple feature fusion method which upsamples the features from the different layers and concatenates them straightforward only improves the tracking performance marginally. Compared with the simple fusion method, the proposed feature fusion module

TABLE I
ABLATION STUDIES ON OTB-2015 DATASET. DISTANCE PRECISION (PREC.) RATES AT A THRESHOLD OF 20 PIXELS AND OVERLAP SUCCESS (AUC) RATES ARE USED TO EVALUATE THE PERFORMANCE.

Multi-layer	IBU	CSCS	AUC↑	Prec.↑
			0.652	0.833
✓			0.659	0.844
✓	✓		0.674	0.860
✓	✓	✓	**0.681**	**0.866**

IBU improves the performance significantly by 2.2% in AUC score and 2.7% in precision. Based on the model with the IBU modules, the tracking performance can be improved further, from 67.4% to 68.1% AUC score, by integrating the proposed CSCS modules.

TABLE II
INFLUENCES OF DIFFERENT SAMPLE SELECTION METHODS FOR TRAINING BOUNDING BOX REGRESSION. DISTANCE PRECISION (PREC.) RATES AT A THRESHOLD OF 20 PIXELS AND OVERLAP SUCCESS (AUC) RATES ON OTB-2015 ARE USED TO EVALUATE THE PERFORMANCE.

	distance	ratio	AUC↑	Prec.↑
(a)	-	0.5	0.673	0.862
(b)	-	0.25	**0.681**	**0.866**
(c)	24	-	0.671	0.863
(d)	36	-	0.668	0.859

In Table II, we test the influences of the different methods for selecting the samples that are used to train the bounding box regression. The 'distance' in this table represents using an absolute distance threshold. 'ratio' means using a ratio to calculate the thresholds, related to the width and height of the target box, for the horizontal and vertical directions. According to Fig. 5, (a) means using all the points inside the groundtruth. (b) represents using different distance thresholds for the horizontal and the vertical directions with respect to the size of the box. (c) defines the distance threshold as 24 pixels. (d) defines the distance threshold as 36 pixels. Method (b) is our final method. The distance thresholds of the horizontal and vertical directions in this method are 0.25 times the width and height of the target box, respectively.

Table. II compares four kinds of regression sample selection methods shown in Fig. 5. The best result is obtained when we set the ratio related to the box height and width as 0.25 to calculate the distance thresholds. It is because this definition takes the bounding box size and aspect ratio into account, which can make the trained regression features more likely locates at the target area and suppress the bad effect from the background features during the training process.

C. Results on OTB Dataset

We compares our tracker IBFP with 9 trackers, including CCOT [6], MDNet [31], ATOM [4], DiMP-18 [3], DaSiamRPN [46], TADT [24], GCT [11], GradNet [23], and SiamFC-tri [7] on both the OTB-2015 [38] and OTB-2013 [37] datasets. On the OTB-2015 [38] dataset, we draw the precision

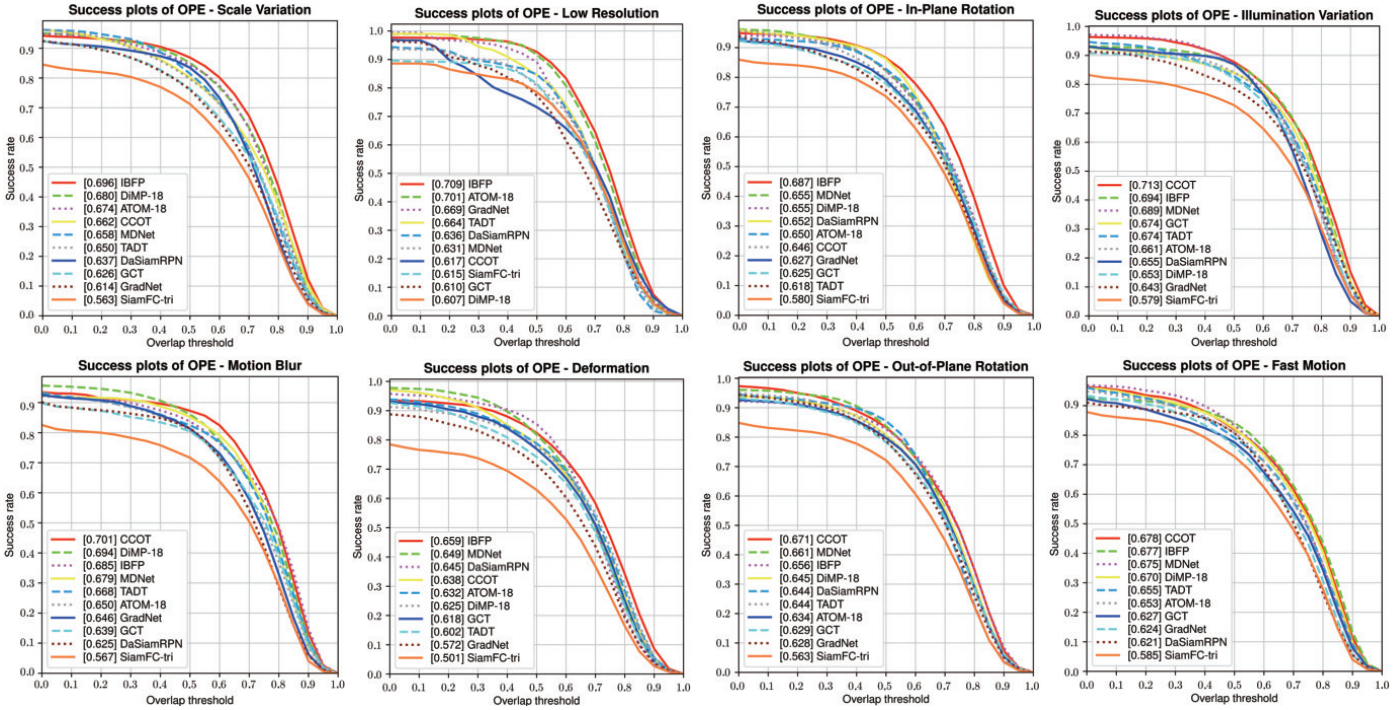

Fig. 7. Precision plots and overlap success plots under different situations on OTB-2013 dataset.

plots and the success plots to compare the total tracking performance under all the situations, as Fig. 8 shows. On the OTB-2013 [37] dataset, we show the tracking performance under the different situations and challenges in Fig. 7.

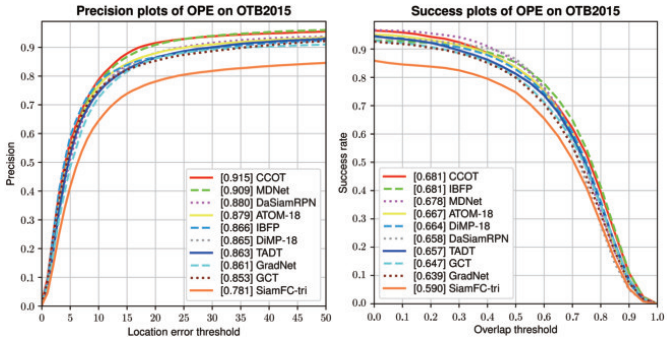

Fig. 8. Precision plots and overlap success plots on OTB-2015 dataset.

We use ResNet18 [14] as the backbone. Compared with ATOM [4] and DiMP-18 [3] who also utilize ResNet18 [14] to extract features, our tracker achieves 68.1% AUC score, higher than them by 1.4% and 1.7%, respectively. According to Fig. 7, our tracker can handle different situations well. Under the in-plane rotation and the significant scale variation situations, our tracker achieves a remarkable performance, which results from that the IBU modules select and fuse the high-level feature with more rotation invariance and the high-resolution low-level feature adaptively.

D. Results on LaSOT Dataset

LaSOT [8] dataset has 280 test videos with 70 different categories. We use the precision plots, normalized precision plots, and success plots to evaluate the tracking performance. We compare the performance of our tracker and some state-of-the-art trackers on the huge dataset LaSOT [8] to demonstrate the effectiveness. Fig. 9 illustrates the tracking results of DiMP-18 [3], GlobalTrack [15], DaSiamRPN [46], ATOM [4], SiamRPN++ [20], D3S [28], C-RPN [9], VITAL [32], GFS-DCF [39], ECO [5], BACF [17], and our tracker IBFP in terms of distance precision (Prec.) rates at a threshold of 20 pixels, normalized precision, and overlap success (AUC) rates.

On the large dataset LaSOT [8], our tracker obtains 54.3% AUC score and 53% Precision score, which is better than ATOM [4] by 2.9% AUC score. Benefited from the effective feature fusion method, the proposed tracker IBFP also outperforms GlobalTrack [15] which takes the whole frame as input and SiamRPN++ [20] which integrates ResNet50 [14] as the backbone, by 2.6% and 4.8% AUC scores, respectively. Compared with other RPN-based trackers like DaSiamRPN [46] and C-RPN [9], Our tracker also achieves better performance. It is worth noting that the success rate of our tracker IBFP is higher than that of other trackers significantly on the overlap threshold period [0.5, 1.0]. It means our tracker can predict the target bounding box more precisely, which results from fusing the discriminative deep layer features and high-resolution shallow layer features effectively.

We also compare the tracking performance of our tracker IBFP with some state-of-the-art trackers under the different situations and challenges on the LaSOT dataset. The results in terms of AUC score are listed in Table. III. The situations and challenges include illumination variation (IV), partial occlusion (PO), deformation (DF), motion blur (MB), camera motion (CM), rotation (RT), background clutter (BC), view-

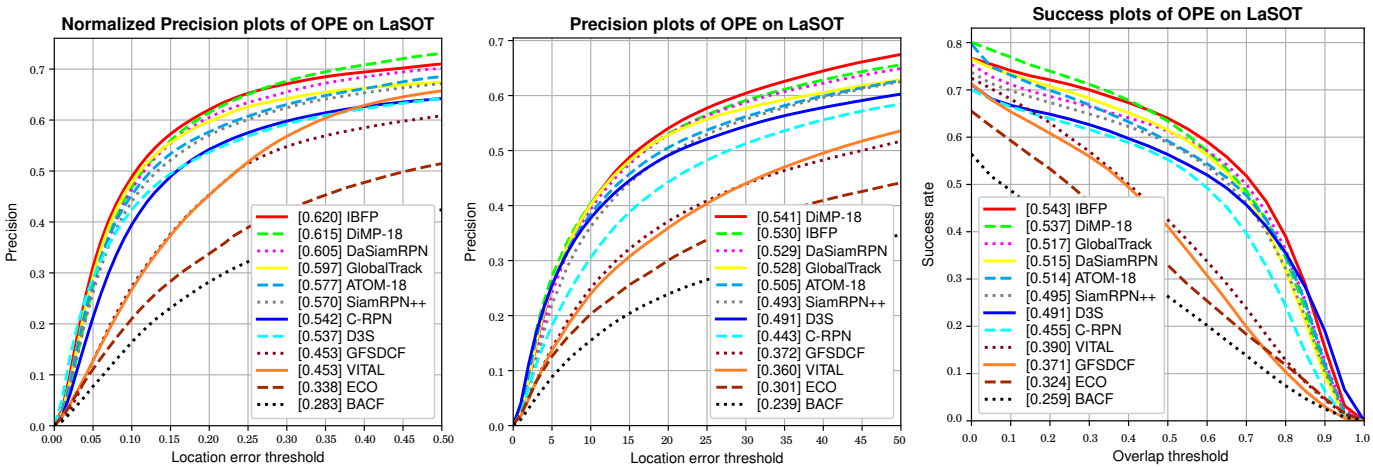

Fig. 9. Normalized precision plots, precision plots, and overlap success plots on LaSOT dataset.

TABLE III
TRACKING PERFORMANCE UNDER DIFFERENT SITUATIONS ON LASOT DATASET IN TERMS OF OVERLAP SUCCESS (AUC) RATES. THE COLOR **RED** AND **BLUE** NOTATE THE BEST AND THE SECOND BEST RESULTS, RESPECTIVELY

	BACF	ECO	GFSDCF	VITAL	C-RPN	D3S	SiamRPN++	ATOM-18	DaSiamRPN	GlobalTrack	DiMP-18	IBFP
IV	0.312	0.373	0.436	0.403	0.487	0.527	0.530	0.549	0.565	0.562	**0.564**	**0.568**
PO	0.222	0.290	0.334	0.361	0.432	0.445	0.466	0.474	0.483	0.491	**0.500**	**0.517**
DF	0.238	0.279	0.357	0.384	0.479	0.525	0.528	0.512	0.538	0.531	**0.549**	**0.561**
MB	0.246	0.305	0.366	0.363	0.413	0.463	0.442	0.494	0.491	0.497	**0.529**	**0.523**
CM	0.286	0.358	0.397	0.397	0.482	0.505	0.513	0.553	0.565	0.543	**0.582**	**0.571**
RT	0.224	0.285	0.345	0.371	0.438	0.472	0.485	0.483	0.493	0.510	**0.522**	**0.523**
BC	0.265	0.319	0.342	0.365	0.409	0.432	0.449	0.454	0.469	0.434	**0.468**	**0.468**
VC	0.254	0.317	0.334	0.339	0.405	0.425	0.444	0.455	0.477	**0.511**	0.506	**0.512**
SV	0.255	0.318	0.367	0.385	0.452	0.486	0.494	0.512	0.514	0.516	**0.533**	**0.540**
FO	0.188	0.254	0.274	0.301	0.348	0.355	0.366	0.427	0.421	0.438	**0.454**	**0.442**
FM	0.176	0.233	0.246	0.265	0.290	0.341	0.316	0.414	0.375	0.396	**0.408**	**0.398**
OV	0.181	0.239	0.295	0.304	0.365	0.404	0.416	0.446	0.469	**0.499**	0.487	**0.490**
LR	0.195	0.267	0.299	0.309	0.355	0.377	0.385	0.450	0.436	0.452	**0.470**	**0.468**
AR	0.228	0.288	0.345	0.358	0.435	0.471	0.472	0.497	0.494	0.507	**0.522**	**0.531**

point change (VC), scale variation (SV), full occlusion (FO), fast motion (FM), out-of-view (OV), low resolution (LR), and aspect ratio change (AR). Because the IBU modules utilize the deep and shallow features in a more effective way, our tracker IBFP achieves the best performance, 53.1% and 54% AUC, when the aspect ratio change (AR) and the scale variation (SV) are significant. Besides, the full use of the features from different layers and the channel selection mechanism of the CSCS modules also can help our tracker to handle the challenges of the partial occlusion (PO), deformation (DF), target rotation (RT), background clutter (BC), and viewpoint change (VC).

E. Results on VOT Dataset

We evaluate our method on the VOT-2019 [18] dataset. There are 60 sequences for evaluating the tracking performance. We compare our tracker with some state-of-the-arts, including TADT [24], SASiamRPN [13], MemDTC [40], SPM [35], ROAM++ [41], and ATOM [4] in terms of expected average overlap (EAO), accuracy values (Av) and robustness values (Rv) in Table. VI. Our tracker IBFP achieves 0.287 EAO score on the VOT-2019 dataset. The tracking accuracy value and robustness value of our tracker is 0.609 and 0.493, respectively.

F. Results on UAV Dataset

UAV123 [29] dataset contains 123 video sequences that has more than 110k frames from an aerial viewpoint. We list the performance in terms of distance precision (Prec.) rates at a threshold of 20 pixels and overlap success (AUC) rates on the UAV123 [29] dataset of our tracker and some state-of-the-art trackers in Table. IV, including ARCF [16], ECO [5], SiamRPN [21], UPDT, DaSiamRPN [46], SiamRPN++ [20], ATOM [4], and DiMP [3]. Our method achieves an AUC score

TABLE IV
STATE-OF-THE-ART COMPARISON ON UAV123 DATASET IN TERMS OF DISTANCE PRECISION (PREC.) RATES AT A THRESHOLD OF 20 PIXELS AND OVERLAP SUCCESS (AUC) RATES. THE COLOR **RED** AND **BLUE** NOTATE THE BEST AND THE SECOND BEST RESULTS, RESPECTIVELY

	ARCF	ECO	SiamRPN	UPDT	DaSiamRPN	SiamRPN++	ATOM	DiMP-18	DiMP-50	IBFP
AUC	0.47	0.525	0.527	0.545	0.586	0.613	0.642	0.643	**0.653**	**0.644**
Prec.	0.67	0.741	0.748	-	0.796	0.807	-	-	-	**0.835**

TABLE V
STATE-OF-THE-ART COMPARISON ON GOT-10K TEST SET IN TERMS OF AVERAGE OVERLAP (AO), AND SUCCESS RATES (SR) AT OVERLAP THRESHOLD OF 0.5 AND 0.75. THE COLOR **RED** AND **BLUE** NOTATE THE BEST AND THE SECOND BEST RESULTS, RESPECTIVELY

	MDNet	ECO	CCOT	SiamFC	SiamFCv2	DaSiamRPN	ATOM	DiMP-18	DiMP-50	IBFP
AO	0.299	0.316	0.325	0.348	0.374	0.444	0.556	0.579	**0.611**	**0.614**
$SR_{0.50}$	0.303	0.309	0.328	0.353	0.404	0.536	0.634	0.672	**0.717**	**0.716**
$SR_{0.75}$	0.099	0.111	0.107	0.098	0.144	0.220	0.402	0.446	**0.492**	**0.499**

TABLE VI
STATE-OF-THE-ART COMPARISON ON VOT2019 DATASET IN TERMS OF EXPECTED AVERAGE OVERLAP (EAO), ACCURACY VALUES (AV) AND ROBUSTNESS VALUES (RV). THE COLOR **RED** AND **BLUE** NOTATE THE BEST AND THE SECOND BEST RESULTS, RESPECTIVELY

Tracker	EAO	Av	Rv
TADT	0.207	0.516	0.677
MemDTC	0.228	0.485	0.587
SASiamRPN	0.253	0.559	0.492
SPM	0.275	0.577	0.507
ROAM++	0.281	0.561	**0.431**
ATOM	**0.292**	**0.603**	**0.411**
IBFP	**0.287**	**0.609**	0.493

of 64.4%, which performs favorably against the state-of-the-arts with about 50 FPS speed.

G. Results on GOT-10K Dataset

GOT-10K [26] dataset contains over 10,000 sequences, 180 of which are selected as the test sequences. The object classes of the test sequences are all different from that in the training set. For fair comparison, we only use the training set offered by GOT-10K [26] dataset to train the model. The evaluation results in terms of average overlap (AO), and success rates (SR) at overlap threshold of 0.5 and 0.75 are shown in Table. V. We compare the results of MDNet [31], ECO [5], CCOT [6], SiamFC [2], SiamFCv2 [34], DaSiamRPN [46], ATOM [4], DiMP [3], and our tracker IBFP. Our tracker get the highest average overlap and success rates at threshold of 0.75. Compared with the trackers with ResNet18 [14] backbone, our tracker achieves 61.4% AO score, with 5.8% and 3.5% higher than ATOM [4] and DiMP-18 [3]. On the GOT-10K [26] test set, our tracker also outperforms DiMP-50 [3] which uses ResNet50 [14] as the backbone in terms of AO score and $SR_{0.75}$ score.

H. Visualization

We show some visualization results of several state-of-the-art trackers and our tracker in Fig. 10. The visualization frames are from the sequences 'swing-10', 'motorcycle-3', 'deer-14', and 'hat-18'. We also visualize the intermediate and final response maps of our tracker in Fig. 11. The intermediate response maps are the average map through channels of the response tensors from the two IBU modules. We take the sequences 'Crowds' and 'Human4' as examples. From the response maps of the first column, we can observe that the response maps from the deep layer light a big area near the targets. After the first step fusion via the IBU module, the response maps in the second column become more focused, while there are some clutter. Via using the second IBU module to fuse and refine the features, the real target can be located certainly and precisely, as the response maps in the third column show.

V. CONCLUSION

Fusing the deep and shallow features is a reasonable way to improve the tracking performance. The proposed instance-based upsampling (IBU) module is a carefully designed feature fusion method for the general visual tracking task, which is used to mix the deep and shallow features in a series manner. The IBU module can guide the tracker to search the target from the low-resolution level to the high-resolution level. Taking both the discriminative information from the deep features and the precise location information from the shallow features, the IBU module can suppress some distractors and locate the target more precisely. By integrating the IBU modules, the proposed tracker can fully use the guidelines from the deep layer feature to locate the target at a high-resolution level. The proposed compressed space channel selection (CSCS) module works like cross attention, which re-weights the template feature channels according to the current search region. Via the CSCS module, the tracker can adjust the importance of each feature channel and select some useful features to locate the target on the basis of the search region. The impressive tracking performance on several public tracking datasets proves the effectiveness of the IBU module and CSCS module. By integrating the IBU and CSCS modules on the ResNet18 [14]

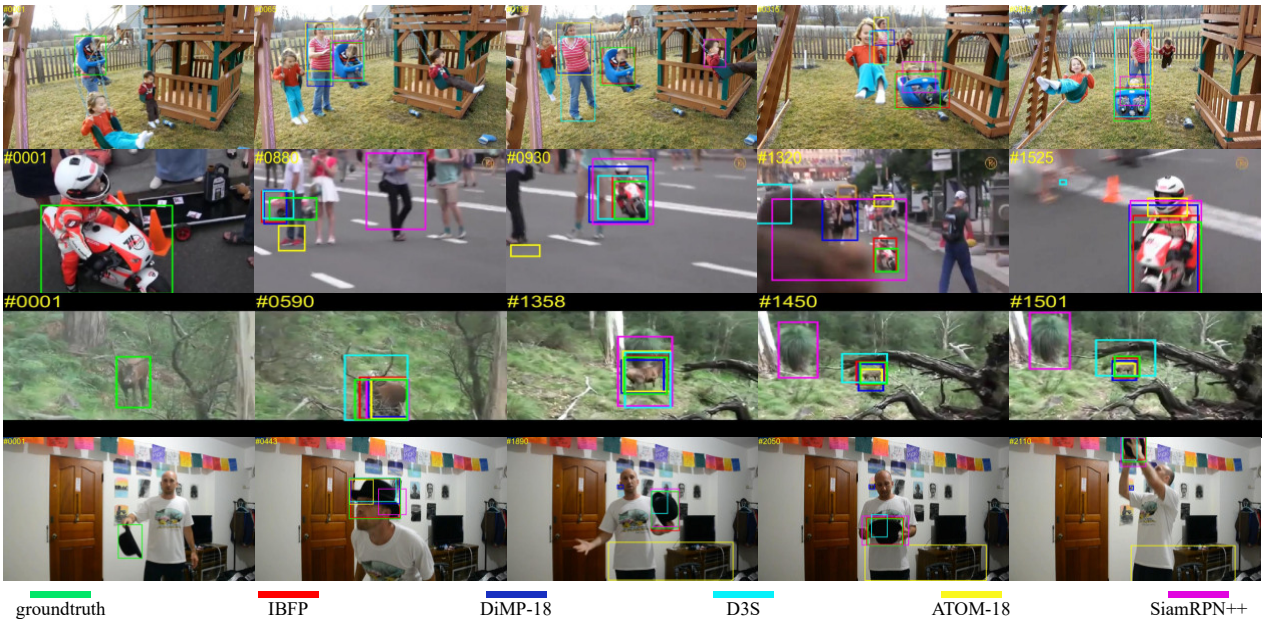

Fig. 10. Visualization of the tracking bounding boxes of some state-of-the-art trackers.

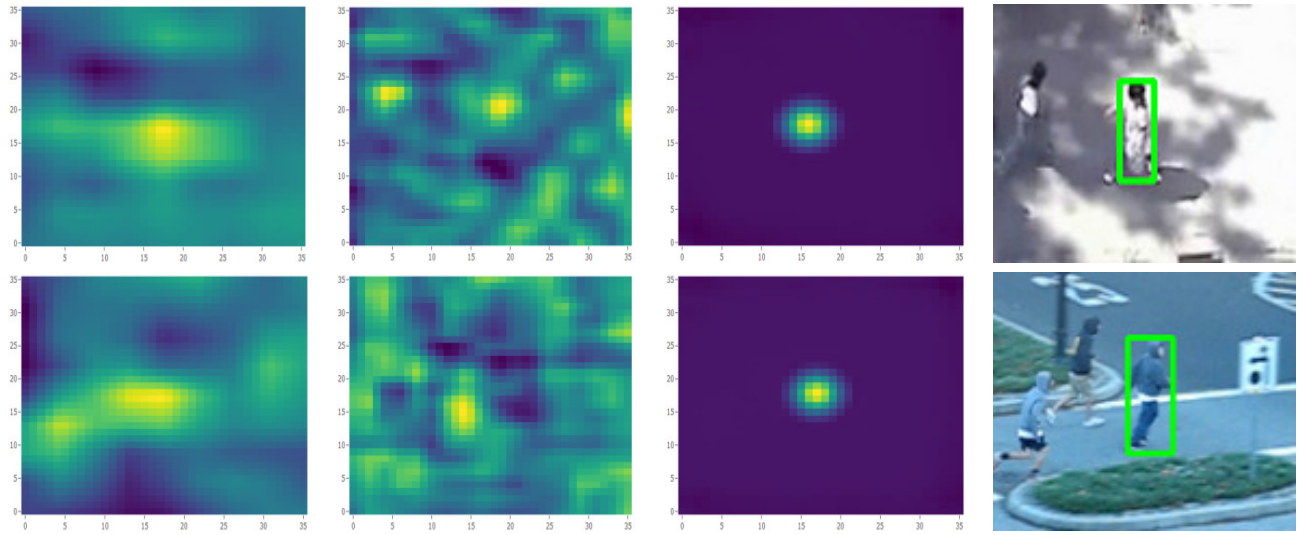

Fig. 11. The intermediate and the final response maps of our tracker. The first column shows the average response map of the response tensor from the first IBU module. The second column shows that from the second IBU module. The third column is the final response map. The last column shows the search area image.

backbone, our tracker can achieve good performance with about 50 FPS running speed.

ACKNOWLEDGMENT

This work was partially supported by the National Natural Science Foundation of China (No.61433007 and No.61271328).

REFERENCES

[1] M. H. Abdelpakey and M. S. Shehata. Dp-siam: Dynamic policy siamese network for robust object tracking. *IEEE Transactions on Image Processing (TIP)*, 29:1479–1492, 2020.

[2] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2016.

[3] G. Bhat, M. Danelljan, L. V. Gool, and R. Timofte. Learning discriminative model prediction for tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[4] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Atom: Accurate tracking by overlap maximization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4660–4669, 2019.

[5] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6638–6646, 2017.

[6] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual

tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 472–488. Springer, 2016.

[7] X. Dong and J. Shen. Triplet loss in siamese network for object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 459–474, 2018.

[8] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[9] H. Fan and H. Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[10] J. Fan, H. Song, K. Zhang, K. Yang, and Q. Liu. Feature alignment and aggregation siamese networks for fast visual tracking. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, 31(4):1296–1307, 2021.

[11] J. Gao, T. Zhang, and C. Xu. Graph convolutional tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[12] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen. Siamcar: Siamese fully convolutional classification and regression for visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[13] A. He, C. Luo, X. Tian, and W. Zeng. A twofold siamese network for real-time object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[14] K. He, X. Zhang, S. Ren, and S. Jian. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[15] L. Huang, X. Zhao, and K. Huang. Globaltrack: A simple and strong baseline for long-term tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pages 11037–11044, 2020.

[16] Z. Huang, C. Fu, Y. Li, F. Lin, and P. Lu. Learning aberrance repressed correlation filters for real-time uav tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2891–2900, 2019.

[17] H. Kiani Galoogahi, A. Fagg, and S. Lucey. Learning background-aware correlation filters for visual tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2017.

[18] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, R. Pflugfelder, J.-K. Kamarainen, L. Cehovin Zajc, O. Drbohlav, A. Lukezic, A. Berg, A. Eldesokey, J. Kapyla, G. Fernandez, A. Gonzalez-Garcia, A. Memarmoghadam, A. Lu, A. He, A. Varfolomieiev, A. Chan, A. Shekhar Tripathi, A. Smeulders, B. Suraj Pedasingu, B. Xin Chen, B. Zhang, B. Wu, B. Li, B. He, B. Yan, B. Bai, B. Li, B. Li, B. Hak Kim, and B. Hak Ki. The seventh visual object tracking vot2019 challenge results. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 25(2), 2012.

[20] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[21] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[22] D. Li, F. Porikli, G. Wen, and Y. Kuai. When correlation filters meet siamese networks for real-time complementary tracking. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, 30(2):509–519, 2020.

[23] P. Li, B. Chen, W. Ouyang, D. Wang, X. Yang, and H. Lu. Gradnet: Gradient-guided network for visual object tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6162–6171, 2019.

[24] X. Li, C. Ma, B. Wu, Z. He, and M.-H. Yang. Target-aware deep tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[25] Z. Liang and J. Shen. Local semantic siamese networks for fast tracking. *IEEE Transactions on Image Processing (TIP)*, 29:3351–3364, 2020.

[26] Lianghua, Huang, Xin, Zhao, and Kaiqi. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 2019.

[27] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014.

[28] A. Lukezic, J. Matas, and M. Kristan. D3s - a discriminative single shot segmentation tracker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[29] M. Mueller, N. Smith, and B. Ghanem. A benchmark and simulator for uav tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2016.

[30] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[31] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4293–4302, 2016.

[32] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, L. Rynson, and M.-H. Yang. Vital: Visual tracking via adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[33] Z. Tian, C. Shen, H. Chen, and T. He. Fcos: Fully convolutional one-stage object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2020.

[34] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr. End-to-end representation learning for correlation filter based tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[35] G. Wang, C. Luo, Z. Xiong, and W. Zeng. Spm-tracker: Series-parallel matching for real-time visual object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[36] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr. Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[37] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[38] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *IEEE Transactions of Pattern Analysis and Machine Intelligence (TPAMI)*, 2015.

[39] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7950–7960, 2019.

[40] T. Yang and A. B. Chan. Visual Tracking via Dynamic Memory Networks. *IEEE Transactions of Pattern Analysis and Machine Intelligence (TPAMI)*, 2019.

[41] T. Yang, P. Xu, R. Hu, H. Chai, and A. B. Chan. Roam: Recurrently optimizing tracking model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[42] T. Zhang, X. Liu, Q. Zhanga, and J. Han. Siamcda: Complementarity- and distractor-aware rgb-t tracking based on siamese network. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, pages 1–1, 2021.

[43] Z. Zhang and H. Peng. Deeper and wider siamese networks for real-time visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[44] W. Zhou, L. Wen, L. Zhang, D. Du, T. Luo, and Y. Wu. Siamcan: Real-time visual tracking based on siamese center-aware network. *IEEE Transactions on Image Processing (TIP)*, 30:3597–3609, 2021.

[45] X. Zhou, D. Wang, and P. Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[46] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu. Distractor-aware siamese networks for visual object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.