Bilinear Siamese Networks with Background Suppression for Visual Object Tracking

Hankyeol Lee
hankyeol@kaist.ac.kr
Seokeon Choi
seokeon@kaist.ac.kr
Youngeun Kim
youngeunkim@kaist.ac.kr
Changick Kim
changick@kaist.ac.kr

Korea Advanced Institute of Science and Technology (KAIST)
Daejeon, Korea

Abstract

In recent years, siamese networks have shown to be useful for visual tracking with high accuracy and real-time speed. However, since the networks only use the output of the last convolution layer, low-level feature maps which provide important spatial details for visual tracking are ignored. In this paper, we propose bilinear siamese networks for visual object tracking to take into account both high- and low-level feature maps. To effectively incorporate feature maps extracted from multiple layers, we adopt factorized bilinear pooling into our network. Also, we introduce a novel background suppression module to reduce the background interference. This module collects negative feature maps for the background in the first frame and suppresses the background information during tracking. Therefore, the module makes the tracker more robust to the background interference. Experimental results on the OTB-50 and OTB-100 benchmarks demonstrate that the proposed tracker has comparable performance with that of the state-of-the-art trackers while running in real-time.

1 Introduction

Visual object tracking is one of the most popular problems in the area of computer vision with many applications such as traffic control, surveillance, and automatic driving. The goal of visual object tracking is to track the target object in the following frames when the target object is specified only in the first frame. Despite significant progress in the past decades, designing a robust tracker to deal with large appearance changes, fast motion, and background clutter is still a problem.

In recent years, as Convolutional Neural Networks (CNNs) have demonstrated state-of-the-art results in various computer vision tasks [11, 12, 34], many CNN-based trackers have been proposed [3, 9, 18, 21]. SiamFC [3], which is one of the successful CNN-based trackers, has achieved high performance with real-time speed. In SiamFC, the feature maps of

© 2019. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

the target template and searching region are extracted by the same network, and then a response map is computed by cross-correlation between these two feature maps. The network is learned offline and the network weights are fixed during online tracking.

However, this approach has two issues. First, SiamFC only takes the output of the last convolution layer without regard to the low- and mid-level feature maps extracted from intermediate convolution layers. Only using the output of the last layer is not optimal representation for visual tracking as it does not fully capture the spatial details of the target. These details obtained by the intermediate layers are important to visual tracking because they are helpful for precise localization of the target object [9, 18]. Second, SiamFC does not take full advantage of the background information. Although negative templates can be easily collected from the background, tracking is performed only using a positive template given in the first frame, which might fail to track the target if background clutter or similar objects exist.

In this paper, we propose bilinear siamese networks to deal with the above issues. To exploit the output of multiple layers, we adopt a bilinear pooling approach, which is widely used for fine-grained classification [17, 29] and visual question answering [15, 30]. It creates feature maps considering the inter-layer feature relations effectively. In addition, we propose a background suppression module which makes the tracker more robust to the background interference. First, the module creates negative feature maps using the target template and the background information in the first frame. Then, it suppresses the influence of the background by utilizing these feature maps. The main contributions of our work can be summarized as follows:

1. We propose bilinear siamese networks for visual tracking, which effectively incorporate the output of multiple layers.

2. A novel background suppression module is developed to reduce the background information that interferes with tracking.

3. The experimental results on the OTB-50 and OTB-100 benchmarks [28] demonstrate that our method achieves comparable results compared with those of the state-of-the-art trackers while processing real-time speed.

The paper is organized as follows. Section 2 presents the related work. The details of the proposed approach are described in Section 3. Section 4 shows the experimental results on on the OTB-50 and OTB-100 benchmarks. At the end of this paper, we conclude with our results in Section 5.

2 Related Work

Before CNN becomes popular, many trackers using hand-craft features [4, 27] have been proposed. DSST [5] deals with scale using hog feature variations, SRDCF [7] reduces boundary effects. Bertinetto *et al*. [2] introduce a tracker considered color information as well. As CNNs have achieved superior performances in various computer vision tasks [11, 12, 34], many researchers in the field of tracking have started employing CNN features for designing the tracker. In CF2 [18] and HDT [22], multi-level CNN features are used instead of handcrafted features and final responses are obtained by hierarchical response and hedging, respectively. MCPF tracker [32] uses CNN features to combine the DCF framework and a particle filter and DeepSRDCF [6] employs CNN features in a SRDCF framework.

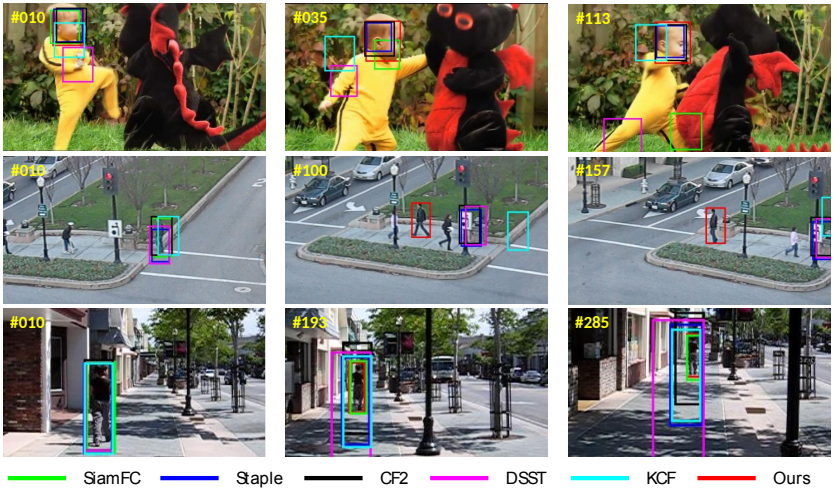


Figure 1: Qualitative results on three challenge sequences, *DragonBaby*, *Human3*, and *Human9*, respectively. Our tracker (red) effectively handles challenging situations compared to the other trackers

Danelljan *et al*. [8] propose the continuous convolution operator tracker (C-COT) to produce a continuous-domain response map and enable fusion of multi-resolution CNN features in a join learning formulation. ECO tracker [9] is proposed to improve the C-COT tracker in terms of performance and speed. The trackers mentioned above use a network that is trained for other tasks (e.g., VGGNet [25] for image classification task). Since the trackers are not trained for tracking, the results are not optimal.

To fully utilize the CNN features, many trackers have been proposed to train the network for tracking. MDNet [20] treats the tracking as a binary classification problem that distinguishes between positive and negative patches. The network is offline trained for tracking and then online updates the domain-specific layers. GOTURN [13] predicts the motion between consecutive frames using a deep regression network. CREST [26] reformulates DCF framework as a convolution layer so that can be trained end-to-end trainable using backpropagation.

One of the most notable trackers is fully convolutional Siamese networks (SiamFC) [3]. SiamFC tracks the target from searching region through a matching network offline trained on the video object detection dataset of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC15) [24]. The matching network is implemented by two-branch CNNs with shared parameters and predicts the similarity between the target template and searching region. Although SiamFC can run in real-time and have comparable performance, its tracking performance is inferior to state-of-the-art trackers. DSiamM [10] integrates correlation filters into the network to perform the online update. Yunhua Zhang *et al*. [33] propose a local structure-based siamese network using message passing.

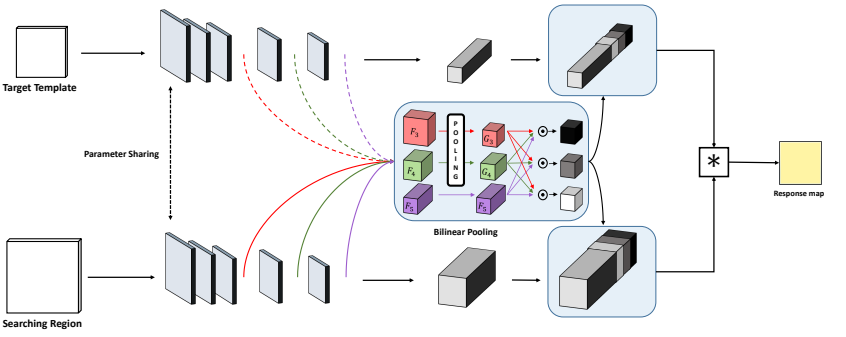


Figure 2: The overall network architecture of our approach.

3 Our Approach

In this section, we first introduce the general formulation of factorized bilinear pooling in Section 3.1. After that, based on this formulation, we present bilinear siamese networks to integrate the outputs from multiple layers in Section 3.2. Finally, the background suppression module is proposed to make the tracker robust to the background interference in Section 3.3

3.1 Factorized Bilinear Pooling

Factorized bilinear pooling has been widely used in fine-grained classification and visual question answering tasks [15, 17, 29, 30]. Kim *et al.* [15] proposed the multi-modal factorized bilinear pooling model for visual question answering tasks using the Hadamard product of two feature vectors. Given two feature vectors $\mathbf{x} \in \mathbb{R}^m$ and $\mathbf{y} \in \mathbb{R}^n$, the full bilinear model is defined as follows:

$$z_i = \mathbf{x}^T W_i \mathbf{y}, \tag{1}$$

where $W_i \in \mathbb{R}^{m \times n}$ is the i-th projection matrix and $z_i \in \mathbb{R}$ is the i-th output of the bilinear model. To obtain an output $\mathbf{z}$ with the dimension c^b, we need to learn $\mathbf{W} = [W_1, W_2, ..., W_{c^b}] \in \mathbb{R}^{m \times n \times c^b}$. According to the matrix factorization trick in [23], the projection matrix W_i can be factorized into two rank-one matrices.

$$z_i = \mathbf{x}^T W_i \mathbf{y} = \mathbf{x}^T U_i V_i^T \mathbf{y} = U_i^T \mathbf{x} \circ V_i^T \mathbf{y}, \tag{2}$$

where $U_i \in \mathbb{R}^m$ and $V_i \in \mathbb{R}^n$, $\circ$ is the Hadamard product. Then, we can get the output feature $\mathbf{z} \in \mathbb{R}^{c^b}$ as follows:

$$\mathbf{z} = P^T \left(U^T \mathbf{x} \circ V^T \mathbf{y} \right), \tag{3}$$

where $U \in \mathbb{R}^{m \times d}$ and $V \in \mathbb{R}^{n \times d}$ are factorized projection matrices, $P \in \mathbb{R}^{d \times c^b}$ is the embedding matrix, and d is a latent dimensionality of the joint embeddings.

3.2 Bilinear Siamese Networks

The network in SiamFC [3] consists of two max pooling layers and five convolution layers. Let $F_l \in \mathbb{R}^{m_l \times n_l \times c_l}$ denote the output of the l-th ($l \in \{1, ..., 5\}$) convolution layer, where m_l, n_l, and c_l are width, height, and the number of channels of each layer, respectively.

Whereas, in the SiamFC framework, only the output of the last layer F_5 is used for tracking, our network uses feature maps of multiple layers F_3, F_4, and F_5. The basic idea is that feature vectors with same spatial location obtained from multiple feature maps are aggregated through factorized bilinear pooling to capture pairwise feature relations. However, since SiamFC is fully-convolutional, which means that convolution operations are performed without padding, the spatial size of these feature maps are different. The spatial relationships between the feature maps are as follows:

$$\begin{aligned} m_3 &= m_5 + 4, \quad n_3 = n_5 + 4, \\ m_4 &= m_5 + 2, \quad n_4 = n_5 + 2. \end{aligned} \tag{4}$$

Based on the spatial relationship, we perform mixed pooling operations to match the size of F_3 and F_4 to the size of F_5 as follows:

$$\begin{aligned} G_3 &= \text{AvgPooling}(F_3, 5) + \text{MaxPooling}(F_3, 5), \\ G_4 &= \text{AvgPooling}(F_4, 3) + \text{MaxPooling}(F_4, 3), \end{aligned} \tag{5}$$

where $\text{AvgPooling}(F,k)$ and $\text{MaxPooling}(F,k)$ represent average pooling and max pooling with a $k \times k$ size window over F. In the end, G_3 and G_4 have the same spatial size as that of F_5 (i.e., $m_5 \times n_5$).

We apply factorized bilinear pooling over multiple feature maps to get the final output feature map $Z \in \mathbb{R}^{m_5 \times m_5 \times c}$. We denote the c_l-dimensional feature vector at a spatial location s of F_l and G_l as $\mathbf{f}_l^s = \left[f_1, f_2, ..., f_{c_l}\right]^T$ and $\mathbf{g}_l^s = \left[g_1, g_2, ..., g_{c_l}\right]^T$. Then, (2) can be rewritten as follows:

$$\begin{aligned} \mathbf{z}_1^s &= P_1^T \left(U^T \mathbf{g}_3^s \circ S^T \mathbf{f}_5^s\right), \\ \mathbf{z}_2^s &= P_2^T \left(V^T \mathbf{g}_4^s \circ S^T \mathbf{f}_5^s\right), \\ \mathbf{z}_3^s &= P_3^T \left(U^T \mathbf{g}_3^s \circ V^T \mathbf{g}_4^s\right), \end{aligned} \tag{6}$$

where $U \in \mathbb{R}^{c_3 \times d}$, $V \in \mathbb{R}^{c_4 \times d}$, and $S \in \mathbb{R}^{c_5 \times d}$ denote projection matrices, respectively. $P_1 \in \mathbb{R}^{d \times c_1^b}$, $P_2 \in \mathbb{R}^{d \times c_2^b}$, and $P_3 \in \mathbb{R}^{d \times c_3^b}$ are embedding matrices. Thus, these feature vectors can be integrated as follows:

$$\mathbf{z}^s = Concat\left(\mathbf{f}_5^s, \mathbf{z}_1^s, \mathbf{z}_2^s, \mathbf{z}_3^s\right), \tag{7}$$

where $\mathbf{z}^s \in \mathbb{R}^c$ ($c = c_5 + c_1^b + c_2^b + c_3^b$) is the final output feature vector at location s of Z. The overall architecture of the bilinear siamese networks is illustrated in Figure 2.

3.3 Background Suppression Module

Siamese networks compare the target template $O_1 \in \mathbb{R}^{127 \times 127 \times 3}$ to the searching region $X_t \in \mathbb{R}^{255 \times 255 \times 3}$ in the t-th frame to obtain a response map R_t as follows:

$$R_t = \varphi\left(O_1\right) * \varphi\left(X_t\right) + b \cdot \mathbb{1}, \tag{8}$$

where φ and $*$ represent the output of the network and the cross-correlation, respectively. $b \cdot \mathbb{1}$ denotes bias. This approach in SiamFC is vulnerable to the background interference because only the positive target template is employed.

In order to make the tracker robust to the background interference, we propose a background suppression module that considers both the background and target information. First,

a background-suppressed patch $X_1^b \in \mathbb{R}^{255\times255\times3}$ is generated by multiplication of X_1 and a Gaussian function when the target with a size of $w_1 \times h_1 \times 3$ is given in the first frame as:

$$X_1^b(x,y) = K(x,y;\sigma_x,\sigma_y) \circ X_1(x,y), \\ \sigma_x = \sigma \cdot \sqrt{\frac{h_1}{w_1}}, \quad \sigma_y = \sigma \cdot \sqrt{\frac{w_1}{h_1}}, \tag{9}$$

where σ denotes a standard deviation and $K(x,y;\sigma_x,\sigma_y)$ is the 2D Gaussian function with an amplitude of 1 and variances for each axis to consider the aspect ratio of the target.

Then, we obtain the background feature map by subtracting the background-suppressed feature map from the feature map of the image containing the background as follows:

$$S = \varphi(X_1) - \varphi\left(X_1^b\right), \tag{10}$$

where $S \in \mathbb{R}^{22\times22\times c}$ is the background feature map. Then, we can sample negative feature maps $\{N_i\}_{i=1}^K$ by sliding window with a size of 6×6 on the background feature map S, where $N_i \in \mathbb{R}^{6\times6\times c}$ and K is the number of the negative feature maps.

Finally, since we have not only the feature map of the target but also the negative feature maps that contain the background information, we can generate a response map considering both the background and target information as follows:

$$R_t = \varphi(O_1) * \varphi(X_t) - \beta \frac{1}{K} \sum_{i=1}^{K} N_i * \varphi(X_t) + b \cdot \mathbb{1}, \tag{11}$$

where β is a hyperparameter to control the influence of the background information. Note that the computational complexity increases K times if (11) is directly used. Since the cross-correlation operation is a linear operator, (11) can be reformulated to speed up as follows:

$$R_t = \left(\varphi(O_1) - \beta \frac{1}{K} \sum_{i=1}^{K} N_i\right) * \varphi(X_t) + b \cdot \mathbb{1}. \tag{12}$$

In contrast to (11), which uses K negative feature maps in every frame, (12) reduces the computational complexity because it uses negative feature maps only in the first frame. The framework of the background suppression module is depicted in Figure 3.

4 Experimental Results

4.1 Implementation Details

We set the output dimensions of factorized bilinear pooling c_1^b, c_2^b, and c_3^b to 64, 128, and 32, respectively. The latent dimensionality of the joint embeddings d is set to 512. The variance value for a 2D Gaussian function is $\sigma = 3.2 \times 10^3$ and the influence parameter of the background information is $\beta = 0.55$. The whole network is trained offline on the ILSVRC-2015 dataset [24]. We use stochastic gradient descent (SGD) with the momentum of 0.9 to train the network, and the initial learning rate is 10^{-3} with an exponential decay. The network is trained for 50 epochs with mini-batches of 32. In the offline training the tracker, we adopt the logistic loss

$$L(y,v) = \frac{1}{|\mathcal{D}|} \sum_{u\in\mathcal{D}} \log(1+\exp(-y[u]\cdot v[u])) \tag{13}$$

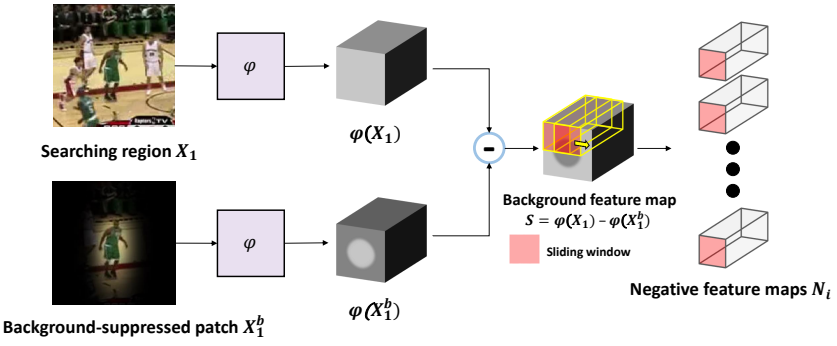


Figure 3: Background suppression module. Negative feature maps are sampled only in the first frame.

where v is the score map between a exemplar and candidate, y is the ground-truth label with a value of 1 or -1, $\mathcal{D}$ is the set of the positions of the score map and u describes a position of the score map. Our approach is executed in python with the Tensorflow library [1]. It runs 69 frames per second (fps) on a system with Intel(R) core(TM) i5-3450 3.4 GHz processor and a single NVIDIA GTX 1070 Ti with 8GB RAM.

4.2 Benchmarks and Evaluation Metrics

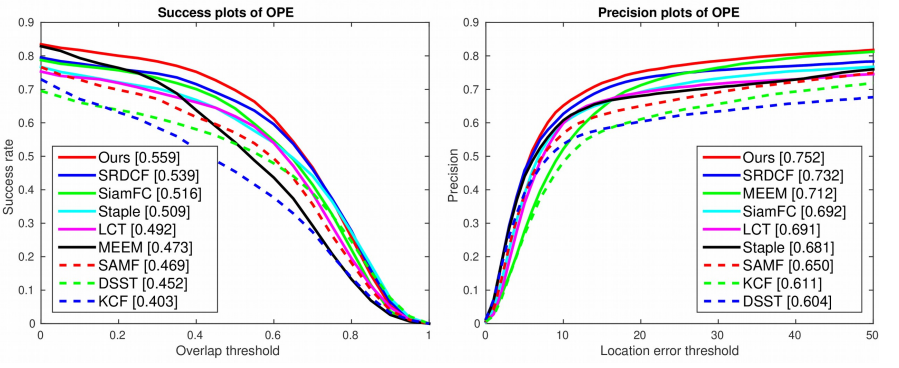
We evaluated our approach on the OTB-50 and OTB-100 datasets [28]. OTB-50 and OTB-100 consist of 50 and 100 sequences with 11 interference attributes: illumination variation (IV), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR), out-of-view (OV), background clutters (BC), and low resolution (LR).

Following the protocol in [28], we applied two evaluation metrics in one-pass evaluation (OPE): precision and success plots. The precision metric calculates the ratio of frames where the center position of the estimated target is within a certain threshold distance from the center position of the ground truth. The score using the threshold of 20 pixels is the representative precision score for each tracker in precision plots. The success metric computes the overlap percentage between the estimated and ground truth bounding boxes. The Area Under Curve (AUC) of success plots indicates the success score to rank the trackers.

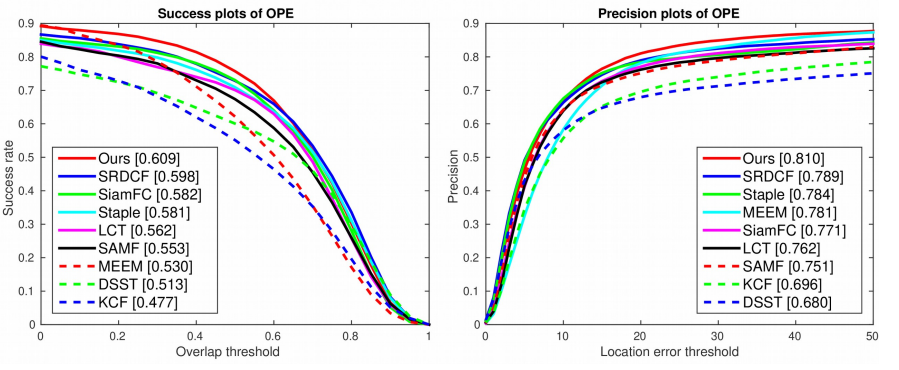
4.3 Evaluation on OTB-50 and OTB-100

We compared our tracker with other 8 state-of-the-art trackers (SRDCF [7], SiamFC [3], Staple [2], LCT [19], MEEM [31], SAMF [16], DSST [5], KCF [14]) on OTB-50 and OTB-100. As shown in Figure 4, our tracker achieves the best results both in success and precision plots on OTB-50 and OTB-100. Notably, our approach outperforms SiamFC, which is a baseline of our approach.

We also performed an attribute-based comparison on OTB-100. The AUC of the success plots under 11 challenging attributes was used to compare the performance. As shown in Table 1, the results demonstrate that our approach achieves the best performance on 9 out of 11 attributes. Notably, our approach outperforms SiamFC on all attributes, which means that the proposed approach makes the tracker more robust under challenging scenarios.



(a)



(b)

Figure 4: Success and precision plots of OPE on (a) OTB-50 and (b) OTB-100. The numbers in legend denote the area-under-curve (AUC) score for success plots and the representative precision score, respectively.

Figure 1 gives the qualitative results on three challenging sequences from our tracker and existing trackers. SiamFC and a few correlation filter based trackers fail to track the target in fast motion and out-of-view (*DragonBaby*). Also, all of the correlation filter based trackers miss the target under scale variation and fast motion (*Human9*). On the other hand, our tracker successfully tracks the target, whereas other trackers fail to find the target after occlusion (*Human3*)

4.4 Ablation Study

We conducted an ablation study on OTB-50 to investigate the effect of our factorized bilinear pooling and background subtraction methods. Figure 5 shows the success and precision plots of 6 variants. We used SiamFC as a baseline and experimented with changing the component. *2 layers* and *3 layers* denote factorized bilinear pooling using feature maps of two layers (i.e., F_4 and F_5) and feature maps of three layers (i.e., F_3, F_4, and F_5). Also, *Max.*, *Avg.*, and *Mix.* indicate max, average, and mixed (i.e., max + average) pooling methods, respectively. *BS* denotes the background suppression module.

According to Figure 5, the results of *2 layers + Mix.* and *3 layers + Mix.* show that *3*

	Ours	SRDCF	SiamFC	Staple	LCT	SAMF	MEEM	DSST	KCF
IV	0.590	**0.613**	0.568	0.598	0.566	0.534	0.517	0.558	0.479
SV	**0.608**	0.565	0.557	0.529	0.492	0.500	0.474	0.475	0.399
OCC	**0.578**	0.559	0.543	0.548	0.507	0.540	0.504	0.453	0.443
DEF	**0.557**	0.544	0.506	0.554	0.499	0.509	0.489	0.420	0.436
MB	**0.613**	0.610	0.568	0.558	0.532	0.534	0.545	0.488	0.456
FM	**0.610**	0.599	0.573	0.541	0.527	0.509	0.529	0.461	0.454
IPR	**0.589**	0.544	0.557	0.552	0.557	0.519	0.529	0.502	0.469
OPR	**0.589**	0.553	0.560	0.538	0.541	0.540	0.528	0.475	0.457
OV	**0.546**	0.460	0.506	0.481	0.452	0.480	0.488	0.386	0.393
BC	0.552	**0.580**	0.524	0.574	0.553	0.527	0.523	0.524	0.499
LR	**0.597**	0.480	0.573	0.411	0.330	0.459	0.355	0.379	0.307

Table 1: Ranked AUC scores under 11 attributes on OTB-100. The **bold** numbers indicate the best performance in each row.

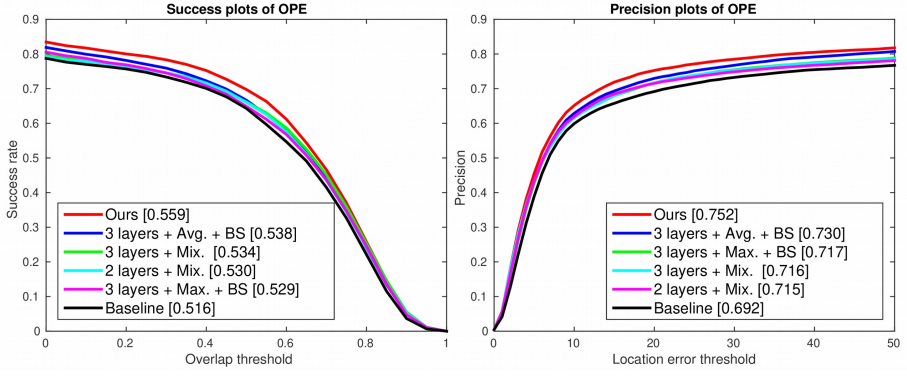


Figure 5: Ablation studies on OTB-50. By changing the number of layers for factorized bilinear pooling, pooling method, and the background suppression module, we compared 6 variants: (*i*) the baseline tracker (SiamFC); (*ii*) *2 layers + Mix.*; (*iii*) *3 layers + Mix.*; (*iv*) *3 layers + Max. + BS*; (*v*) *3 layers + Avg. + BS*; (*vi*) *3 layers + Mix. + BS* (ours).

layers have higher performance compared to *2 layers* both on success and precision scores. *3 layers + Mix. + BS* (ours) achieves about 2.9% and 3.6% improvement in success and precision scores compared to *3 layers + Mix.*, respectively. These results demonstrate that the proposed background suppression module leads to an improvement in tracking performance. Moreover, we compare the performance of 3 different pooling methods: *Mix.*, *Avg.*, and *Max.*. Mix pooling method achieves 3% and 2.1% higher in success scores and 3.5% and 2.2% in precision scores than Max and Avg pooling methods, respectively. Finally, compared with the baseline, our method achieves 4.3% and 6% performance improvement in the success and precision scores on OTB-50, respectively.

5 Conclusion

In this paper, we have proposed bilinear siamese networks for real-time visual tracking. The output of multiple layers is effectively integrated by factorized bilinear pooling. We further introduce a novel background suppression module to reduce the background interference, which makes the tracker more robust. The experimental results on OTB-50 and OTB-100 show that the proposed approach achieves very promising in the performance among existing state-of-the-art trackers while running in real-time.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. 2015.

[2] Luca Bertinetto, Jack Valmadre, Stuart Golodetz, Ondrej Miksik, and Philip HS Torr. Staple: Complementary learners for real-time tracking. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 1401–1409, 2016.

[3] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *Proc. Eur. Conf. Comput. Vis.*, pages 850–865, 2016.

[4] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[5] Martin Danelljan, Gustav Häger, Fahad Khan, and Michael Felsberg. Accurate scale estimation for robust visual tracking. In *Proc. Brit. Mach. Vis. Conf.*, 2014.

[6] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Convolutional features for correlation filter based visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 58–66, 2015.

[7] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Learning spatially regularized correlation filters for visual tracking. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 4310–4318, 2015.

[8] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European Conference on Computer Vision*, pages 472–488. Springer, 2016.

[9] Martin Danelljan, Goutam Bhat, F Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 21–26, 2017.

[10] Qing Guo, Wei Feng, Ce Zhou, Rui Huang, Liang Wan, and Song Wang. Learning dynamic siamese network for visual object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1763–1771, 2017.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 770–778, 2016.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 2980–2988, 2017.

[13] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision*, pages 749–765. Springer, 2016.

[14] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(3):583–596, 2015.

[15] Jin-Hwa Kim, Kyoung Woon On, Woosang Lim, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. Hadamard Product for Low-rank Bilinear Pooling. In *Int. Conf. Learn. Represent.*, 2017.

[16] Yang Li and Jianke Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *Proc. Eur. Conf. Comput. Vis.*, pages 254–265, 2014.

[17] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 1449–1457, 2015.

[18] Chao Ma, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang. Hierarchical convolutional features for visual tracking. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 3074–3082, 2015.

[19] Chao Ma, Xiaokang Yang, Chongyang Zhang, and Ming-Hsuan Yang. Long-term correlation tracking. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 5388–5396, 2015.

[20] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 4293–4302. IEEE, 2016.

[21] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 4293–4302, 2016.

[22] Yuankai Qi, Shengping Zhang, Lei Qin, Hongxun Yao, Qingming Huang, Jongwoo Lim, and Ming-Hsuan Yang. Hedged deep tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4303–4311, 2016.

[23] Steffen Rendle. Factorization machines. In *IEEE int. Conf. Data Mining*, pages 995–1000, 2010.

[24] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015.

[25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[26] Yibing Song, Chao Ma, Lijun Gong, Jiawei Zhang, Rynson WH Lau, and Ming-Hsuan Yang. Crest: Convolutional residual learning for visual tracking. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2574–2583. IEEE, 2017.

[27] Joost Van De Weijer, Cordelia Schmid, Jakob Verbeek, and Diane Larlus. Learning color names for real-world applications. *IEEE Transactions on Image Processing*, 18 (7):1512–1523, 2009.

[28] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(9):1834–1848, 2015.

[29] Chaojian Yu, Xinyi Zhao, Qi Zheng, Peng Zhang, and Xinge You. Hierarchical bilinear pooling for fine-grained visual recognition. In *Proc. Eur. Conf. Comput. Vis.*, pages 595–610, 2018.

[30] Zhou Yu, Jun Yu, Jianping Fan, and Dacheng Tao. Multi-modal factorized bilinear pooling with co-attention learning for visual question answering. *Proc. IEEE Int. Conf. Comput. Vis.*, pages 1839–1848, 2017.

[31] Jianming Zhang, Shugao Ma, and Stan Sclaroff. Meem: robust tracking via multiple experts using entropy minimization. In *Proc. Eur. Conf. Comput. Vis.*, pages 188–203, 2014.

[32] Tianzhu Zhang, Changsheng Xu, and Ming-Hsuan Yang. Multi-task correlation particle filter for robust object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 3, 2017.

[33] Yunhua Zhang, Lijun Wang, Jinqing Qi, Dong Wang, Mengyang Feng, and Huchuan Lu. Structured siamese network for real-time visual tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 351–366, 2018.

[34] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 2881–2890, 2017.