Saliency-Associated Object Tracking

Zikun Zhou[1], Wenjie Pei[1,*], Xin Li[2], Hongpeng Wang[1,2], Feng Zheng[3], and Zhenyu He[1,*]
[1]Harbin Institute of Technology, Shenzhen [2]Peng Cheng Laboratory
[3]Southern University of Science and Technology
zhouzikunhit@gmail.com wenjiecoder@outlook.com xinlihitsz@gmail.com
wanghp@hit.edu.cn zfeng02@gmail.com zhenyuhe@hit.edu.cn

Abstract

Most existing trackers based on deep learning perform tracking in a holistic strategy, which aims to learn deep representations of the whole target for localizing the target. It is arduous for such methods to track targets with various appearance variations. To address this limitation, another type of methods adopts a part-based tracking strategy which divides the target into equal patches and tracks all these patches in parallel. The target state is inferred by summarizing the tracking results of these patches. A potential limitation of such trackers is that not all patches are equally informative for tracking. Some patches that are not discriminative may have adverse effects. In this paper, we propose to track the salient local parts of the target that are discriminative for tracking. In particular, we propose a fine-grained saliency mining module to capture the local saliencies. Further, we design a saliency-association modeling module to associate the captured saliencies together to learn effective correlation representations between the exemplar and the search image for state estimation. Extensive experiments on five diverse datasets demonstrate that the proposed method performs favorably against state-of-the-art trackers.

1. Introduction

Visual object tracking aims to predict the target states in a tracking sequence given the initial state of the target object in the first sequence frame. It is a fundamental research topic in Computer Vision and has a wide range of applications including video surveillance, robotics, and motion analysis. Although deep trackers [6, 36, 38, 44], which benefit from excellent feature learning for images by deep neural networks, have achieved great progress in recent years, tracking targets with various real-time appearance variations, such as deformation, occlusion, and viewpoint changes, *etc.*, remains an extremely challenging task.

A classical type of deep tracking approaches [2, 7, 25, 49] performs tracking in a holistic strategy, which seeks to learn a precise deep feature representation for the whole target object and then localize the target in the search image. A prominent example is the Siamese-based trackers [1, 25, 26, 29, 39], which learn deep representations for both the target exemplar and the search image in the same feature space by a Siamese neural network, and then perform target tracking by feature matching between them. Such methods perform well in ordinary scenarios in which the target keeps stable appearances close to the exemplar, but struggle in the challenging scenarios where the target varies substantially. This is because the global appearance gap between the target exemplar and the target state in the search image results in an inevitable tracking error. The online learning trackers [8, 18, 38], which are another typical type of methods, are designed to adapt to the appearance

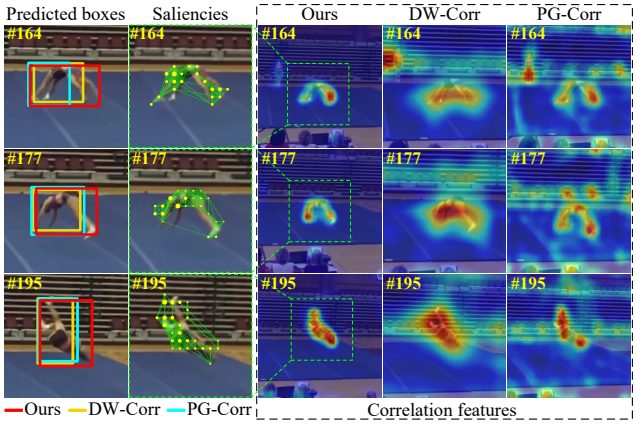

Figure 1. Given a search image in a tracking sequence, our *SAOT* first captures local saliencies (yellow dots) of the target that are discriminative for tracking, and then associates the captured saliencies together to learn precise correlations between the target exemplar and the search image for reflecting target states. Thus, our model can generate more precise correlation features than DW-Corr [25] (in the holistic tracking strategy) and PG-Corr [32] (in the part-based strategy), and accordingly predict more precise bounding boxes. The correlation features are visualized by averaging all channels (red color indicates higher correlation). Larger-size salient dots indicate higher saliency values.

*Corresponding authors.

variations of the target by learning an online filter. However, these methods still perform tracking in the holistic strategy and thus can hardly deal with drastic appearance variations.

In contrast to the holistic tracking strategy, another type of existing tracking methods [32, 34, 46, 48] adopts the part-based strategy, which first tracks local parts of the target object and then infers the target state by summarizing the tracking results of these parts. A common way of these part-based methods is to partition the target into regular patches equally and then perform tracking on all these patches in parallel. Whilst such a part-based tracking strategy mitigates the difficulties of tracking appearance-varying targets, a potential limitation is that not all the partitioned patches are equally informative for tracking. Some parts which are not discriminative are difficult to be tracked and may have adverse effects on inferring the global target state.

In this paper, we follow the part-based tracking strategy and propose the Saliency-Associated Object Tracker (*SAOT*). The key difference between our *SAOT* and other part-based tracking methods is that *SAOT* focuses on capturing and tracking the local saliencies of the target that are discriminative for tracking instead of simply tracking all partitioned patches in parallel. Specifically, we design a fine-grained saliency mining mechanism to capture local saliencies in the target that are discriminative and easily localized in the search image. Subsequently, these captured saliencies are associated together by modeling the interactions between them to learn global correlations between the target exemplar and the search image, which can reflect the target state in the search image precisely.

The rationales behind such design of our *SAOT* are: 1) the salient local regions in the target, which are tracked more precisely and easily than other regions, can potentially keep consistent distinctiveness in various appearance variations; 2) different associations between the saliencies correspond to different appearances of the same target, so that we model the associations between the captured saliencies to adapt to real-time appearance variations. Consequently, our *SAOT* is able to cope with various appearance variations of the target during tracking, such as deformation and occlusion. Figure 1 presents an example of tracking a gymnast, in which the appearance of the gymnast varies substantially during display. Owing to the captured saliencies robust to appearance variations, the bounding boxes predicted by our model are much more precise than those predicted based on DW-Corr [25] and PG-Corr [32], which are in the holistic strategy and the part-based strategy, respectively.

The tracking strategy of the proposed *SAOT*, which first deals with local saliencies with high confidence and then associates them together to achieve the global solution, is akin to the divide-and-conquer algorithm. To conclude, we make the following contributions: 1) A fine-grained saliency mining module is designed to capture local saliencies in the target which are discriminative for tracking. 2) We propose a saliency-association modeling module to associate the captured saliencies together to learn effective global correlations between the exemplar and the search image. 3) We achieve favorable performance against state-of-the-art methods in both quantitative and qualitative evaluations on five benchmarks (OTB2015, NFS30, LaSOT, VOT2018, and GOT10k), demonstrating the effectiveness of our *SAOT*.

2. Related Work

This section mainly discusses the related trackers from the perspectives of the holistic and part-based strategies.

Holistic-strategy trackers. Numerous Siamese-based trackers [1, 25, 26, 39] perform tracking in the holistic strategy. Such trackers measure the similarity between the exemplar and the search image by feature matching to localize the target, in which the feature maps of the exemplar are treated as a holistic kernel to perform cross-correlation on the search image. Most of them [1, 25, 26, 49] use the target from the first frame as a fixed exemplar to track the target in all subsequent frames, resulting in limited robustness to appearance variations of the target during tracking. Several adaptive Siamese-based methods [15, 28, 45, 51], which use the historical target states to update the representation of the exemplar, are proposed to address this limitation.

Many online learning trackers [18, 38] also perform tracking in the holistic strategy. These trackers learn a correlation filter [6, 8, 10, 18] or a convolutional filter [2, 7, 38] using online collected samples, and use the filter as a holistic appearance model to distinguish the target from backgrounds. Although the adaptive Siamese-based trackers [28, 45, 51] and online learning trackers [18, 38] model the target information from historical frames, they are less effective in handling drastic real-time appearance variations of the target due to the holistic tracking strategy.

Part-based trackers. Many traditional trackers [31, 34, 46, 47, 48] resort to the part-based strategy to handle the challenges of deformation and occlusion. Most of them [34, 46, 47, 48] directly track all the equally-partitioned patches of the target in parallel, instead of selecting the patches easy to be tracked according to their discriminability. As a result, the less discriminative patches may adversely affect the adaptability of these approaches. RPT [31] estimates the reliability for the randomly sampled patches of the target in a Monte Carlo framework and tracks reliable patches with multiple traditional correlation filters. However, the predicted positions of the patches are combined using a voting scheme in RPT, which can only estimate a coarse target state. In addition, the above part-based trackers are designed based on the less representative hand-crafted features, which limits their tracking performance.

PG-Net [32] is a recently proposed part-based deep tracker; it decomposes the feature maps of the exemplar

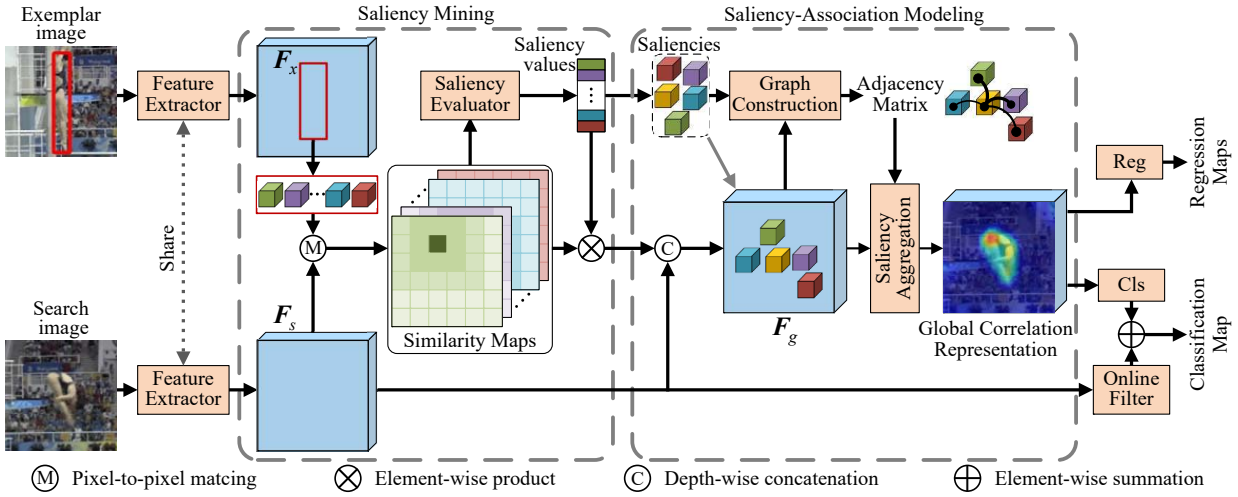

Figure 2. **Architecture of the proposed Saliency-Associated Object Tracker (*SAOT*).** It contains two core modules: 1) Saliency Mining module, which captures the saliencies of the target; 2) Saliency-Association Modeling module, which associates the captured saliencies to learn an effective correlation representation for state estimation. Reg and Cls denote the regression and classification heads, respectively.

into spatial and channel kernels to perform pixel-to-global matching with the search image. Similar to most part-based trackers, this method also equally treats all spatial kernels that represent a local part of the exemplar without considering their discriminability. Unlike PG-Net, our *SAOT* adopts a saliency mining mechanism to focus on the discriminative parts of the exemplar. Besides, we explicitly model the interactions between the captured saliencies to effectively associate them, instead of directly combining the matching results of the parts by global matching as PG-Net does.

3. Saliency-Associated Object Tracker

Given an exemplar image for the initial target and a search image in a tracking sequence, the goal of our Saliency-Associated Object Tracker (*SAOT*) is to learn robust correlation representations between them, which is able to effectively cope with various appearance variations of the target object during tracking, such as deformation and occlusion. To this end, our *SAOT* first captures the local saliencies in the target object that are discriminative for tracking by the proposed Saliency Mining module, then models the associations between these saliencies to learn effective global correlation features between the target exemplar and the search image for precise tracking.

3.1. Overall Framework

Figure 2 illustrates the overall framework of the proposed *SAOT*, consisting of two core modules: Saliency Mining module and Saliency-Association Modeling module.

Taken as input an exemplar image and a search image in a tracking sequence, our *SAOT* first employs a Siamese feature extractor to learn deep representations $\boldsymbol{F}_x \in \mathbb{R}^{h_x \times w_x \times c}$ and $\boldsymbol{F}_s \in \mathbb{R}^{h_s \times w_s \times c}$ in the same feature space for the target exemplar (cropped from the exemplar image according to the bounding box) and the search image, respectively. Herein we adopt widely used ResNet [17] pre-trained on Imagenet [12] as the feature extractor due to its excellent performance of image feature learning.

The Saliency Mining module is designed to capture the local saliencies of the target exemplar which are discriminative for tracking. It calculates similarity maps to measure the pixel-to-pixel correspondences between $\boldsymbol{F}_x$ and $\boldsymbol{F}_s$, and selects local sharp maximum points as saliencies. These captured saliencies correspond to the most discriminative regions of the exemplar, which can be easily localized with high confidence and accuracy.

The captured saliencies are then associated together by the Saliency-Association Modeling module of *SAOT* to learn effective global correlation representations between the exemplar and the search image. The obtained correlation representations are expected to reflect the target state in the search image precisely by aggregating the distributions of all saliencies in the search image with the learned interactions between them. Finally, the target state is estimated by a classification head for confidence estimation and a regression head for predicting the bounding box of the target.

3.2. Saliency Mining

Typically, not all local regions of the target exemplar are easy to be tracked. Thus we design the Saliency Mining module to capture the saliencies corresponding to discriminative local regions of the target exemplar that can be easily localized in the search image.

The proposed Saliency Mining module performs saliency mining in two steps: 1) constructs similarity maps for each pixel in the feature maps of the target exemplar $\boldsymbol{F}_x$ to achieve the distribution of matching score in the search image; 2) measures the saliency value of each pixel in $\boldsymbol{F}_x$ based on the obtained similarity maps to select saliencies.

Construction of similarity maps. As shown in Figure 3,

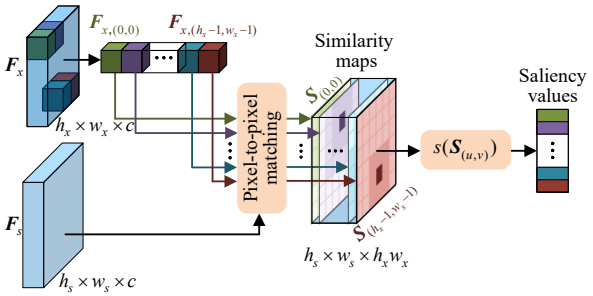

Figure 3. **Structure of the Saliency Mining module.** It first constructs the similarity maps by performing pixel-to-pixel matching between $\boldsymbol{F}_x$ and $\boldsymbol{F}_s$, and then computes the saliency value for every pixel in $\boldsymbol{F}_x$ based on the corresponding similarity map.

the similarity map is constructed for each pixel in $\boldsymbol{F}_x$ by measuring the pixel-to-pixel matching degree between this pixel and the each pixel in $\boldsymbol{F}_s$. To be specific, the matching degree between the pixel located at (u, v) in $\boldsymbol{F}_x$ and the pixel located at (p, q) in $\boldsymbol{F}_s$ is formulated as:

$$\boldsymbol{S}_{((u,v),(p,q))} = f(\boldsymbol{F}_{x,(u,v)}, \boldsymbol{F}_{s,(q,p)}), \tag{1}$$

where $\boldsymbol{F}_{x,(u,v)}$ denotes the vectorial representations at location (u, v) in $\boldsymbol{F}_x$ along the channel dimension, and similar denotation applies to $\boldsymbol{F}_{s,(p,q)}$. Herein f refers to a kernel function for measuring similarity between two vectors. In our implementation, the cosine similarity operator is adopted for f, which is an efficient and effective distance metric. Hence, the similarity in Eq. 1 is calculated by:

$$\boldsymbol{S}_{((u,v),(p,q))} = \frac{\boldsymbol{F}_{x,(u,v)} \cdot \boldsymbol{F}_{s,(q,p)}}{\|\boldsymbol{F}_{x,(u,v)}\| \|\boldsymbol{F}_{s,(p,q)}\|}, \tag{2}$$

where $\cdot$ denotes the inner product operator. The achieved similarities between the pixel at location (u, v) in $\boldsymbol{F}_x$ and all pixels in $\boldsymbol{F}_s$ form a single-channel similarity map denoted as $\boldsymbol{S}_{(u,v)} \in \mathbb{R}^{h_s \times w_s}$.

Saliency evaluation. For each pixel in the exemplar features $\boldsymbol{F}_x$, the maximum point in its similarity map is considered to be the matched position (with the largest confidence) for this pixel in the search image. We evaluate the saliency for this pixel based on the measurements of the peak distribution around the maximum point in the similarity map. Specifically, we consider two measurements: the intensity and the concentration of the peak distribution.

The intensity of a peak distribution is used to measure the relative strength of the maximum value compared to other values in the whole similarity map. A straightforward way to measure the intensity of a peak distribution is Peak-to-Sidelobe Ratio (PSR) [4] which is defined as:

$$\mathrm{PSR}(\boldsymbol{S}_{(u,v)}; \Phi) = \frac{\max(\boldsymbol{S}_{(u,v)}) - \mu_\Phi(\boldsymbol{S}_{(u,v)})}{\sigma_\Phi(\boldsymbol{S}_{(u,v)})}. \tag{3}$$

Herein Φ denotes the sidelobe w.r.t. to a peak distribution in the similarity map $\boldsymbol{S}_{(u,v)}$, which is defined as the region of $\boldsymbol{S}_{(u,v)}$ excluding the neighboring region around the maximum point (referred to as main lobe Ψ). Here main lobe and sidelobe are defined to roughly indicate the relevant and irrelevant regions to the peak distribution around the maximum point, respectively. μ_Φ and σ_Φ are the mean value and standard deviation of $\boldsymbol{S}_{(u,v)}$ of the sidelobe, respectively. In the initial definition [4], the size of main lobe Ψ for arbitrary similarity maps is pre-defined as a fixed value. We argue that such a definition is unreasonable since the distribution characteristics of similarity maps are not taken into account. Figure 4 shows two examples of similarity maps with different peak distributions around the maximum points, which apparently correspond to different sizes of the main lobes. Instead of fixing the size of main lobe, we define the boundary of main lobe Ψ as the closest contour around the peak, whose height value is equal to the mean value of the similarity map. Consequently, the intensity γ of a peak distribution in the similarity map $\boldsymbol{S}_{(u,v)}$ is defined as:

$$\begin{aligned} \Psi &\triangleq \mathrm{region}(\boldsymbol{S}_{(u,v)})\,|_{\mathrm{contour}(\mathrm{avg}(\boldsymbol{S}_{(u,v)}))}, \\ \gamma(\boldsymbol{S}_{(u,v)}) &= \mathrm{PSR}(\boldsymbol{S}_{(u,v)}; \mathrm{region}(\boldsymbol{S}_{(u,v)}) - \Psi), \end{aligned} \tag{4}$$

where $\mathrm{avg}(\boldsymbol{S}_{(u,v)})$ is the mean value of the similarity map.

Another measurement we use for saliency evaluation is the concentration of the peak distribution, which is inversely proportional to the coverage area of the peak distribution around the maximum point. Thus, we measure the concentration c of a peak distribution in the similarity map $\boldsymbol{S}_{(u,v)}$ by the reciprocal of the area of main lobe $A_\Psi(\boldsymbol{S}_{(u,v)})$:

$$c(\boldsymbol{S}_{(u,v)}) = A_\Psi^{-1}(\boldsymbol{S}_{(u,v)}). \tag{5}$$

Combining the defined intensity and the concentration, we evaluate the quality of a saliency $s(\boldsymbol{S}_{(u,v)})$ for the similarity map $\boldsymbol{S}_{(u,v)}$ by:

$$s(\boldsymbol{S}_{(u,v)}) = \gamma(\boldsymbol{S}_{(u,v)})[c(\boldsymbol{S}_{(u,v)})]^\alpha, \tag{6}$$

where α is a hyper-parameter to balance between the effects of intensity and concentration. The rationale behind this design is that the defined intensity and the concentration jointly reveal the sharpness of the peak distribution around the maximum point. A larger saliency value s implies that the corresponding pixel in the feature maps of the exemplar $\boldsymbol{F}_x$ is more discriminative for tracking and easier to be localized in the feature maps of the search image $\boldsymbol{F}_s$.

Considering that the tracker should be encouraged to focus on tracking the central area of the target exemplar, a regularized term, which is a Gaussian mask, is added into the saliency evaluation metric s:

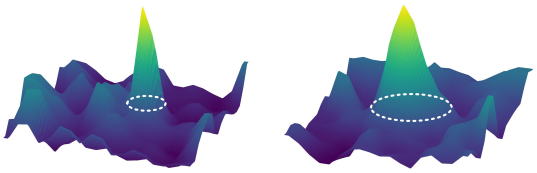

Figure 4. **Two similarity maps with different peak distributions.** The surface plots show the similarity values. Note that the main lobes, denoted by white dash circles, are of different sizes.

$$s(\boldsymbol{S}_{(u,v)}) = \gamma(\boldsymbol{S}_{(u,v)})[c(\boldsymbol{S}_{(u,v)})]^{\alpha} + \lambda g_{\mu_g,\sigma_g}(u,v). \quad (7)$$

Herein $g_{\mu_g,\sigma_g}(u,v)$ is a Gaussian function aligned to the center of the exemplar feature $\boldsymbol{F}_x$, and λ is a balance weight. During end-to-end training, the gradients can be back-propagated through the saliency evaluation metric, and we detail its back-propagation in the supplementary materials.

Based on the defined saliency evaluation metric in Eq. 7, we compute the saliency for each pixel in the feature maps of the exemplar $\boldsymbol{F}_x$ and select K most salient pixels as the set of captured saliencies $P_x = \{\boldsymbol{p}_x^k\}_{k=1}^K$. The matched positions of these saliencies in the feature maps of the search image $\boldsymbol{F}_s$ composes the counterpart of the saliency set in the search image $P_s = \{\boldsymbol{p}_s^k\}_{k=1}^K$.

3.3. Saliency-Association Modeling

The captured saliencies, which are discriminative local parts of the target for tracking, are further associated together by the Saliency-Association Modeling module of *SAOT* to learn effective global correlation representations between the exemplar and the search image. The obtained correlation representations are finally used for estimating the target state in the search image for tracking.

An intuitive way to associate the captured saliencies is to make connections between these local saliencies to form a global graph that is able to characterize the whole target. Following this way, the Saliency-Association Modeling module of our *SAOT* performs saliency association in two steps: 1) constructs an effective graph among the captured saliencies to model the interactions between these saliencies; 2) aggregates the saliencies based on the constructed graph to learn global correlation representations between the exemplar and the search image.

Construction of the saliency graph. We consider two types of information for node features when constructing the saliency graph: 1) the similarity maps $\boldsymbol{S}$ which contain the precise correspondence information from each local parts of the exemplar to the search image; 2) the feature maps of the search image $\boldsymbol{F}_s$. Two types of information, which have the equal size of feature maps ($h_s \times w_s$), are concatenated together in depth. Consequently, the resulting stacked feature maps (denoted as $\boldsymbol{F}_g$) can be considered as a graph which has total $h_s w_s$ regular nodes, while each node is represented by a vectorial feature whose dimension is $h_x w_x + c$. Note that the similarity maps are normalized by the corresponding saliency values before concatenation to emphasize more on the captured saliencies. Besides, the positions of the captured K saliencies in the graph are indicated in P_s obtained in the Saliency Mining module.

A key step of constructing the saliency graph is to model the interactions between nodes by connecting edges. Since we aim to associate the captured saliencies to achieve the effective global representation of the tracking target, we make pairwise edge-connections between K saliencies specified in P_s. Besides, we also connect each node in $\boldsymbol{F}_g$ to its eight neighbors to perform neighboring information interactions for feature fusing between adjacent nodes. The resulting connection set including these two types of edges is denoted as $\mathcal{C}$, which is illustrated in Figure 5.

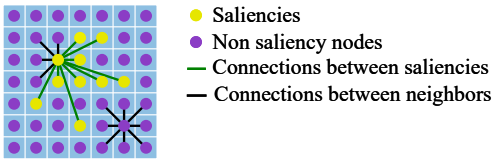

Figure 5. **Two kinds of connections considered for constructing the saliency graph.** The connections between saliencies are used to model the interactions between them, and those between neighbors are used for feature fusing between adjacent nodes.

To precisely model the interactions between the above specified connections, the edge weights are learned by the proposed Saliency-Association Modeling module rather than being fixed as binary values. Particularly, we customize a two-layer perception network to learn the edge weight for each connection specified before. Thus, the weighted adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}, N = h_s w_s$ for the saliency graph is modeled by:

$$\boldsymbol{A}_{ij} = \begin{cases} \sigma(\phi_2(\mathrm{ReLU}(\phi_1(|\boldsymbol{v}_i - \boldsymbol{v}_j|)))), & if \text{ edge } < i,j > \in \mathcal{C}; \\ 0, & else. \end{cases} \quad (8)$$

where $\boldsymbol{v}_i \in \mathbb{R}^{h_x w_x + c}$ and $\boldsymbol{v}_j \in \mathbb{R}^{h_x w_x + c}$ are features of two nodes in a connection. ϕ_1 and ϕ_2 denote the parameters of two fully connected layers, while σ refers to Sigmoid function which transforms the edge weights to lie in $(0, 1)$.

Aggregation of the captured saliencies. The second step for saliency association modeling is to aggregate the saliency information according to the constructed saliency graph. There are multiple ways to perform graph aggregation. We opt for Graph Convolutional Networks (GCN) [23] for its effectiveness and convenience to be integrated into the whole model for end-to-end training.

Specifically, we construct the two-layer GCN to perform saliency aggregation. Inspired by Li et al.[27], we adopt the high-order polynomial of $\boldsymbol{A}$ to model multi-scale interactions between nodes. Formally, the l-th layer graph convolution is formulated as:

$$\boldsymbol{X}^{(l+1)} = \sigma_l\Big(\sum_{m=1}^{M} w_m \hat{\boldsymbol{A}}^m \boldsymbol{X}^{(l)} \boldsymbol{\Theta}_{(m)}^{(l)}\Big), \quad (9)$$

where m and M are the polynomial order and total number of orders, respectively. Here, w_m is a trainable weight for the order m. $\hat{\boldsymbol{A}} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}} \tilde{\boldsymbol{A}} \tilde{\boldsymbol{D}}^{\frac{1}{2}}$ is the normalized adjacency matrix [23], where $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$ and $\tilde{\boldsymbol{D}}$ is the diagonal degree matrix of $\tilde{\boldsymbol{A}}$. $\boldsymbol{X}^{(l)} \in \mathbb{R}^{N\times d_l}$ and $\boldsymbol{X}^{(l+1)} \in \mathbb{R}^{N\times d_{l+1}}$ are the input and output features of all nodes at layer l respectively, where d_l and d_{l+1} are the corresponding feature dimensions, and d_0 is equal to node feature dimension ($h_x w_x + c$). $\boldsymbol{\Theta}_{(m)}^{(l)} \in \mathbb{R}^{d_l \times d_{l+1}}$ denotes the learnable pa-

rameter matrix at layer l for m-th order. σ_l is the activation function at layer l.

Through constructing the saliency graph and further performing saliency aggregation, the Saliency-Association Modeling module of *SAOT* is able to learn global correlation representations between the target exemplar and the search image, which is further used for predicting the target state in the search image.

3.4. Tracking Framework

Our model can be readily integrated into various typical tracking frameworks. As shown in Figure 2, we integrate our algorithm with a typical online learning tracker, namely online discriminative filter [2]. The output global correlation representations by our algorithm is fed into a classification head for predicting the classification map and a regression head for predicting the bounding box of the target. In particular, the output $\boldsymbol{p}_o$ of the classification head is used to regularize the response map $\boldsymbol{p}_r$ produced by online discriminative filter via weighted element-wise summation to generate the final classification map $\boldsymbol{p}_{cls}$. The predicted bounding box by the regression head, which is corresponding to the maximum classification score in $\boldsymbol{p}_{cls}$, is used as the final tracking result. Both the classification and regression heads are designed following FCOS [40].

End-to-end parameter learning. The whole model *SAOT* is trained in an end-to-end manner. Specifically, we employ IoU loss [37] and binary cross-entropy (BCE) loss [11] to train the regression and classification heads respectively in an offline manner. The online discriminative filter is trained following DiMP [2], whose offline training is performed jointly with the training of our *SAOT*.

4. Experiments

4.1. Experimental Setup

Implementation details. We use the fused feature of *conv-3* and *conv-4* of ResNet [17] as the Siamese representation for our *SAOT*, where the fusion weights are computed according to SKNet [30]. The target exemplar is cropped from the feature maps of the exemplar image according to its bounding box and pooled by a PrPool [20] layer to obtain its precise representation, whose size is set to 8×8. The search image is with an area 5^2 times that of the target and resized to 288×288. λ and σ_g in Eq. 7 are set to 1 and 2, respectively. K is set to 48. $\boldsymbol{p}_r$ is set to 0.8. We use the training splits of COCO [33], GOT10k [19], TrackingNet [35], and LaSOT [13] to train our model. During training, the parameters in ResNet are frozen, while the other parameters are optimized using ADAM [22] with a learning rate decayed from 1×10^{-3} to 8×10^{-6} and a weight decay of 1×10^{-4} except for those of the online discriminative filter, whose training settings are following DiMP [2]. Codes and raw results are available at https://github.com/ZikunZhou/SAOT.git.

Table 1. **AUC and precision (Pre.) for six variants of our *SAOT* on OTB2015 and NFS30.** The best scores are marked in **bold**.

Variants	OTB2015		NFS30	
	AUC	Pre.	AUC	Pre.
Base model	0.651	0.860	0.597	0.703
PPFM	0.687	0.881	0.625	0.736
PAM	0.701	0.909	0.641	0.761
SAOT (Ours)	**0.714**	**0.926**	**0.656**	**0.778**
DW-Corr	0.691	0.884	0.617	0.712
PG-Corr	0.693	0.896	0.619	0.711

Datasets and metrics. We evaluate our algorithm on the OTB2015 [43], NFS30 [21], LaSOT [13], VOT2018 [24], and GOT10k [19] datasets. Specifically, both OTB2015 and NFS30 consist of 100 sequences. They use precision and success to measure tracking performance, and the area under the curve (AUC) of the success plot is used for ranking. LaSOT is a large-scale dataset containing 1,400 sequences in total and 280 sequences in the testing set. It uses precision, normalized precision, and success as performance metrics. VOT2018 contains 60 sequences and uses expected average overlap (EAO) to measure the overall performance of trackers. GOT10k contains 10,000 and 180 sequences in the training and testing splits, respectively; it uses average overlap (AO) and success rate (SR) as performance metrics.

4.2. Ablation Study

To investigate the effectiveness of each proposed component, we perform ablation studies with six variants of *SAOT*:
1) **Base model**, which only contains the feature extractor, the classification and regression heads, and the online filter of *SAOT*. Herein, the classification and regression heads are constructed on the feature maps of the search image $\boldsymbol{F}_s$.
2) **PPFM**, which computes the similarity maps $\boldsymbol{S}$ between the exemplar and the search image by Pixel-to-Pixel Feature Matching to improve the base model. It uses a two-layer CNN to adjust the stacked feature maps of $\boldsymbol{S}$ and $\boldsymbol{F}_s$ to generate the correlation representation, on which the classification and regression heads are constructed.
3) **PAM**, which associates all local parts to generate the correlation representation by viewing all the local parts equally as saliencies, i.e., no saliencies are captured. We denote such a model as Part-Association Model.
4) ***SAOT***, our intact model which associates saliencies instead of all local parts as PAM does.
5) **DW-Corr**, which employs the Depth-Wise cross Correlation [25] in our framework to replace the saliency mining and saliency-association modeling modules.
6) **PG-Corr**, which employs the Pixel to Global cross Correlation [32] in our framework.

Table 1 presents the experimental results of these variants on the OTB2015 [43] and NFS30 [21] benchmarks.

Effect of the constructed similarity maps. The performance gaps between base model and PPFM clearly demonstrate the benefits of constructing similarity maps in the fea-

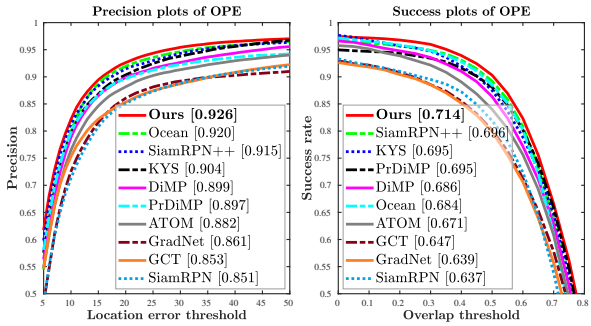

Figure 6. **Precision and success plots of different tracking methods on OTB2015.**

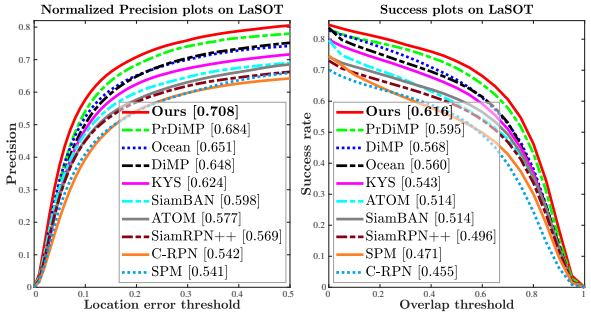

Figure 7. **Normalized precision and success plots of different tracking methods on the test set of LaSOT.**

ture space to model the fine-grained similarity between the exemplar and the search image.

Effect of association modeling. Compared with PPFM, PAM achieves performance gains of 1.4% and 1.6% in AUC on OTB2015 and NFS30, respectively. These results validate the benefits of associating the matched local parts by modeling the pairwise interactions between them, which generates a more robust correlation representation.

Effect of the saliency mining mechanism. The comparison between PAM and our *SAOT* manifests the effectiveness of the proposed saliency mining mechanism, which further improves tracking performance by 1.3% and 1.5% in AUC on OTB2015 and NFS30, respectively. This mechanism successfully enables the tracker to focus on local saliencies of the target that are discriminative for tracking.

Comparison between different correlation calculating methods. The performance of DW-Corr and PG-Corr decreases by 2.3%/2.1% and 3.9%/3.7% in AUC on OTB2015 and NFS30, respectively, compared with our *SAOT*. It demonstrates the superiority of the correlation representation learned by mining saliencies and associating them.

4.3. Comparison with State-of-the-art Trackers

Herein we compare our *SAOT* with 17 representative state-of-the-art methods on five benchmarks, including OTB2015, NFS30, LaSOT, VOT2018, and GOT10k. The methods involved in the comparison include 16 holistic-strategy trackers (KYS [3], Ocean [49], SiamBAN [5], SiamAttn [44], PrDiMP [9], Retina-MAML [41], DiMP [2], GradNet [28], ATOM [7], SiamRPN++ [25], C-RPN [14], GCT [15], SPM [42], DaSiamRPN [50], SiamRPN [26], and DSiam [16]) and one part-based tracker (PG-Net [32]). We discuss the experimetal results per dataset below.

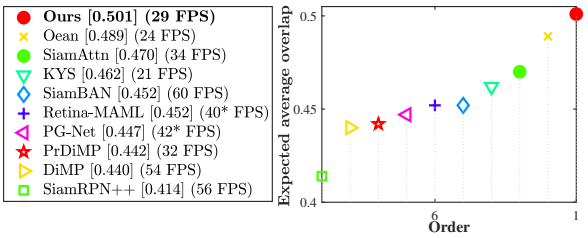

Figure 8. **Expected average overlap and average running speed of different trackers on VOT2018.** The notation * denotes the speed is reported by the authors as the code is not available.

Table 2. **AUC of different tracking methods on NFS30.**

	DaSiam RPN [50]	SiamRPN ++ [25]	ATOM [7]	SiamBAN [5]	DiMP [2]	KYS [3]	PrDiMP [9]	**Ours**
AUC	0.395	0.503	0.584	0.594	0.619	0.634	0.635	**0.656**

Table 3. **AO and SR of different trackers on GOT10k.**

	DSiam [16]	SiamRPN ++ [25]	ATOM [7]	Ocean [49]	DiMP [2]	PrDiMP [9]	KYS [3]	**Ours**
AO	0.417	0.518	0.556	0.611	0.611	0.634	0.636	**0.640**
$SR_{0.5}$	0.461	0.618	0.634	0.721	0.717	0.738	–	**0.749**

OTB2015. Figure 6 illustrates the precision and success plots on OTB2015. Our algorithm achieves the best AUC score of 0.714 and the best precision score of 0.926. Note that DiMP [2], Ocean [49], and our *SAOT* are all built based on the same online discriminative filter. The difference is that DiMP and Ocean perform tracking in the holistic tracking strategy while our method adopts the part-based strategy. Our method outperforms these two methods by a large margin (2.8% and 3.0% in AUC, respectively), which demonstrates the effectiveness of the proposed method.

NFS30. Table 2 reports the AUC scores on NFS30. While PrDiMP [9] and KYS [3] perform well on this dataset with AUC scores of 0.635 and 0.634, respectively, the proposed *SAOT*, achieving the best AUC score of 0.656, further improves tracking performance by 2.1% and 2.2% over these two trackers, respectively.

LaSOT. We follow protocol II [13] to evaluate the proposed *SAOT* on the test set of LaSOT. Figure 7 shows the normalized precision and success plots. Our *SAOT* achieves the best performance in both AUC and normalized precision. Compared to Ocean [49] and DiMP [2], our method achieves remarkable performance gains of 5.6%/4.8% and 5.7%/6.0% in AUC and normalized precision, respectively.

VOT2018. Figure 8 presents the EAO scores of different trackers on VOT2018. Although Ocean [49] obtains an impressive EAO score of 0.489, our method further improves the EAO score by 1.2%. Besides, compared with the state-of-the-art online discriminative filter-based meth-

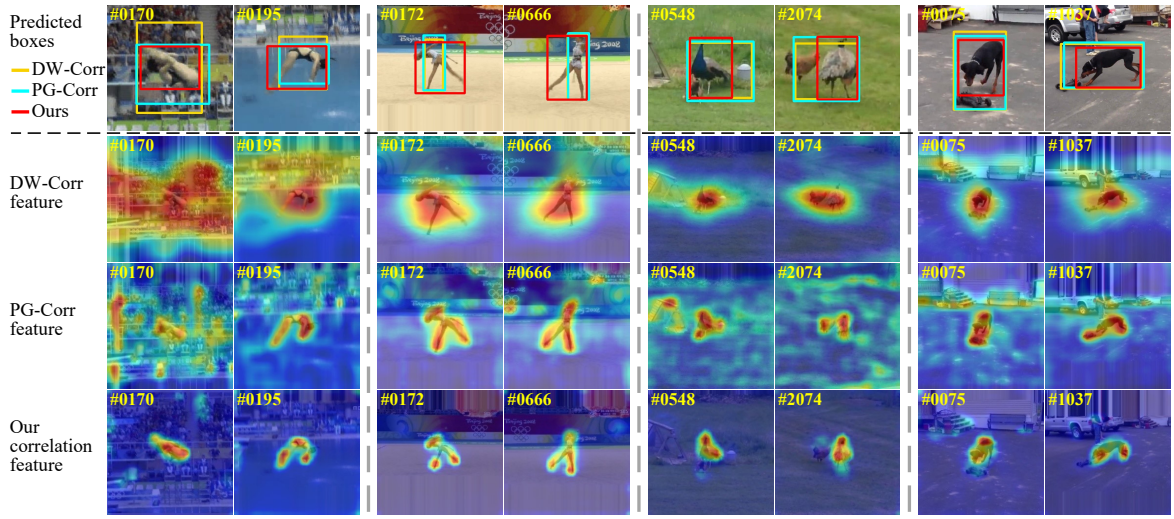

Figure 9. **Qualitative comparison between our *SAOT*, DW-Corr, and PG-Corr on four challenging tracking sequences (left two with deformation and the other two with distractors).** Our *SAOT* is able to learn more precise correlation features than those generated by DW-Corr and PG-Corr. Consequently, our *SAOT* predicts more precise bounding boxes than the other two methods.

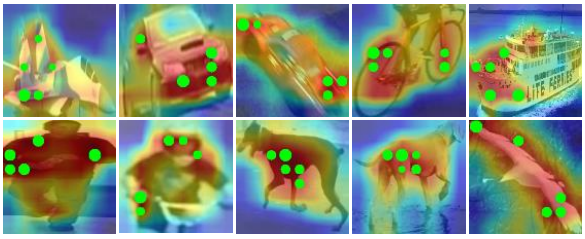

Figure 10. **Saliency maps of 10 target exemplars.** The saliency maps are obtained by visualizing the saliency scores calculated by Eq. 7 for the whole feature maps of each exemplar. Top-5 most salient local regions are indicated in green dots for each exemplar.

ods KYS [3] and PrDiMP [9], our *SAOT* achieves substantial performance gains of 3.9% and 5.9% in EAO, respectively. We also report the average running speeds of different trackers in Figure 8, which are tested using the same PC with an RTX2080 GPU on VOT2018 without reset. Our *SAOT* runs at 29 FPS, achieving real-time performance.

GOT10k. We follow the defined protocol [19] to train our *SAOT* for evaluating it on GOT10k. Table 3 reports the AO and SR scores on the test set of GOT10k. Compared with Ocean [49] and DiMP [2], the proposed method achieves performance gains of 2.9%/2.9% in AO and 2.8%/3.2% in $SR_{0.5}$, respectively. In addition, our algorithm performs favorably against PrDiMP [9] and KYS [3]. These experimental results on GOT10k, whose training and testing sets do not share the object class, validates the generalization ability of our approach across different object classes.

4.4. Qualitative Study

To obtain more insights into our method, we visualize the correlation representations and the saliency values.

Visualization of correlation representations. We visually compare our *SAOT* and the two other variants DW-Corr and PG-Corr. Figure 9 shows the correlation features and the bounding boxes on two challenging sequences with deformation (left) and two with distractors (right). Our model predicts more precise correlation features and bounding boxes than the other two methods, which implies its better capability to handle deformation and distractors as the captured saliencies are robust to deformation and distractors.

Visualization of saliency maps. Figure 10 illustrates the saliency maps of ten target exemplars. We observe that the proposed saliency evaluation metric assigns high saliency values to local regions that are discriminative for tracking.

5. Conclusion

In this work, we have presented the Saliency-Associated Object Tracker (*SAOT*), which first deals with the discriminative local saliencies and then associates them to achieve the global solution. Specifically, our *SAOT* employs the proposed Saliency Mining module to capture the saliencies of the target object, which are robust to target deformation and distractors. Further, we propose a Saliency-Association Modeling module to associate the captured saliencies by modeling the interactions between them, learning a precision correlation representation for reflecting the target state. The proposed method achieves favorable performance against state-of-the-art trackers on five datasets.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (U2013210, 62006060, and 62002241), the Shenzhen Research Council (JCYJ20210324120202006), the Special Research project on COVID-19 Prevention and Control of Guangdong Province (2020KZDZDX1227), and the Shenzhen Stable Support Plan Fund for Universities (GXWD20201230155427003-20200824125730001).

References

[1] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *ECCVW*, 2016.

[2] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *ICCV*, 2019.

[3] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Know your surroundings: Exploiting scene information for object tracking. In *ECCV*, 2020.

[4] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010.

[5] Zedu Chen, Bineng Zhong, Guorong Li, Shengping Zhang, and Rongrong Ji. Siamese box adaptive network for visual tracking. In *CVPR*, 2020.

[6] Kenan Dai, Dong Wang, Huchuan Lu, Chong Sun, and Jianhua Li. Visual tracking via adaptive spatially-regularized correlation filters. In *CVPR*, 2019.

[7] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*, 2019.

[8] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, 2017.

[9] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *CVPR*, 2020.

[10] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, 2016.

[11] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[13] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *CVPR*, 2019.

[14] Heng Fan and Haibin Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *CVPR*, 2019.

[15] Junyu Gao, Tianzhu Zhang, and Changsheng Xu. Graph convolutional tracking. In *CVPR*, 2019.

[16] Qing Guo, Wei Feng, Ce Zhou, Rui Huang, Liang Wan, and Song Wang. Learning dynamic siamese network for visual object tracking. In *ICCV*, 2017.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[18] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2014.

[19] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[20] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *ECCV*, 2018.

[21] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *ICCV*, 2017.

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[23] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017.

[24] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka ˇCehovin Zajc, Tomas Vojir, Goutam Bhat, Alan Lukezic, Abdelrahman Eldesokey, et al. The sixth visual object tracking vot2018 challenge results. In *ECCVW*, 2018.

[25] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 2019.

[26] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, 2018.

[27] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *CVPR*, 2019.

[28] Peixia Li, Boyu Chen, Wanli Ouyang, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Gradnet: Gradient-guided network for visual object tracking. In *ICCV*, 2019.

[29] Xin Li, Chao Ma, Baoyuan Wu, Zhenyu He, and Ming-Hsuan Yang. Target-aware deep tracking. In *CVPR*, 2019.

[30] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *CVPR*, 2019.

[31] Yang Li, Jianke Zhu, and Steven CH Hoi. Reliable patch trackers: Robust visual tracking by exploiting reliable patches. In *CVPR*, 2015.

[32] Bingyan Liao, Chenye Wang, Yayun Wang, Yaonong Wang, and Jun Yin. Pg-net: Pixel to global matching network for visual tracking. In *ECCV*, 2020.

[33] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[34] Si Liu, Tianzhu Zhang, Xiaochun Cao, and Changsheng Xu. Structural correlation filter for robust visual tracking. In *CVPR*, 2016.

[35] Matthias Muller, Adel Bibi, Silvio Giancola, Salman Alsubaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, 2018.

[36] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016.

[37] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, 2019.

[38] Yibing Song, Chao Ma, Lijun Gong, Jiawei Zhang, Rynson WH Lau, and Ming-Hsuan Yang. Crest: Convolutional residual learning for visual tracking. In *ICCV*, 2017.

[39] Ran Tao, Efstratios Gavves, and Arnold WM Smeulders. Siamese instance search for tracking. In *CVPR*, 2016.

[40] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019.

[41] Guangting Wang, Chong Luo, Xiaoyan Sun, Zhiwei Xiong, and Wenjun Zeng. Tracking by instance detection: A meta-learning approach. In *CVPR*, 2020.

[42] Guangting Wang, Chong Luo, Zhiwei Xiong, and Wenjun Zeng. Spm-tracker: Series-parallel matching for real-time visual object tracking. In *CVPR*, 2019.

[43] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015.

[44] Yuechen Yu, Yilei Xiong, Weilin Huang, and Matthew R Scott. Deformable siamese attention networks for visual object tracking. In *CVPR*, 2020.

[45] Lichao Zhang, Abel Gonzalez-Garcia, Joost van de Weijer, Martin Danelljan, and Fahad Shahbaz Khan. Learning the model update for siamese trackers. In *ICCV*, 2019.

[46] Tianzhu Zhang, Kui Jia, Changsheng Xu, Yi Ma, and Narendra Ahuja. Partial occlusion handling for visual tracking via robust part matching. In *CVPR*, 2014.

[47] Tianzhu Zhang, Si Liu, Changsheng Xu, Shuicheng Yan, Bernard Ghanem, Narendra Ahuja, and Ming-Hsuan Yang. Structural sparse tracking. In *CVPR*, 2015.

[48] Tianzhu Zhang, Changsheng Xu, and Ming-Hsuan Yang. Robust structural sparse tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):473–486, 2018.

[49] Zhipeng Zhang, Houwen Peng, Jianlong Fu, Bing Li, and Weiming Hu. Ocean: Object-aware anchor-free tracking. In *ECCV*, 2020.

[50] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *ECCV*, 2018.

[51] Zheng Zhu, Wei Wu, Wei Zou, and Junjie Yan. End-to-end flow correlation tracking with spatial-temporal attention. In *CVPR*, 2018.