Deformable Object Tracking with Gated Fusion

Wenxi Liu, Yibing Song, Dengsheng Chen, Shengfeng He,
Yuanlong Yu, Tao Yan, Gerhard P. Hancke, and Rynson W.H. Lau

***Abstract*—The tracking-by-detection framework receives growing attentions through the integration with the Convolutional Neural Networks (CNNs). Existing tracking-by-detection based methods, however, fail to track objects with severe appearance variations. This is because the traditional convolutional operation is performed on fixed grids, and thus may not be able to find the correct response while the object is changing pose or under varying environmental conditions. In this paper, we propose a deformable convolution layer to enrich the target appearance representations in the tracking-by-detection framework. We aim to capture the target appearance variations via deformable convolution, which adaptively enhances its original features. In addition, we also propose a gated fusion scheme to control how the variations captured by the deformable convolution affect the original appearance. The enriched feature representation through deformable convolution facilitates the discrimination of the CNN classifier on the target object and background. Extensive experiments on the standard benchmarks show that the proposed tracker performs favorably against state-of-the-art methods.**

***Index Terms*—visual tracking, deformable convolution, gating.**

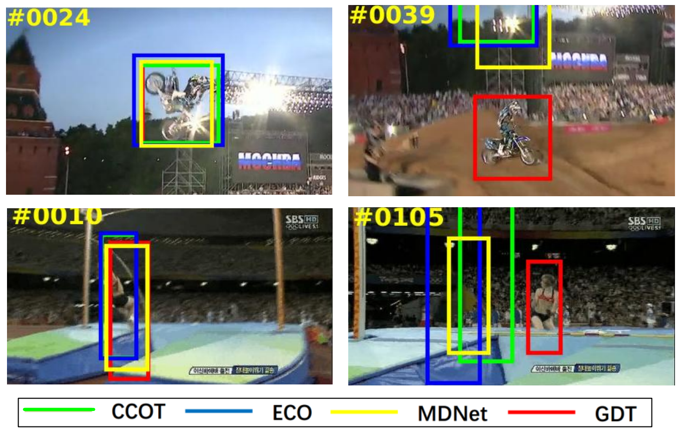


Fig. 1. Illustration of significant appearance variations (i.e., rotation and deformation) in videos *MotorRolling* and *Jumping*. Compared with the state-of-the-art methods, our gated-fusion deformable tracker (denoted as *GDT*) can extract features from rapid appearance variations and correctly localize the targets.

I. Introduction

Visual object tracking is one of the fundamental problems in computer vision and has many applications, e.g., security surveillance, autonomous driving, and human-computer interactions. In recent years, with the advancement of deep convolutional neural networks (CNNs), which can extract features that are more discriminative than the empirical ones, visual tracking has achieved favorable performance on multiple standard benchmarks.

Despite the demonstrated success, existing state-of-the-art tracking methods suffer from large object appearance variations. The setting of visual tracking uses only the first frame as input, which contains limited representations of the target appearance. Tracking accuracy deteriorates when the target object undergoes severe appearance variations (e.g., pose variation, deformation, and rotation as shown in Fig. 1). Existing methods are designed without sufficient modeling of such severe appearance variations, degrading the classifier's ability to discriminate the target object from the background.

As we have observed, the deformable objects do not always reside in the regular grids of the image space and the relative locations of the object parts often vary in video frames. However, the existing CNN-based tracking methods lack internal mechanisms to handle deformations, since the standard CNNs perform the convolution operation over a fixed geometric structure. In prior vision tasks, a common solution to this problem is to collect an extensive amount of training samples. Training data, however, is difficult to collect during online tracking. Hence, given only an input sample in the first frame, the normal convolutional features often fail to model the object with significant pose variation, deformation, or occlusion in tracking.

To tackle this problem, we present a deformable convolutional layer in the CNN-based tracking-by-detection framework in order to model the appearance variations. The deformable convolutional layer enables free-form deformation of the sampling grid. Thus, it can extract features adaptively according to the changing object appearances. In particular, when the target object undergoes severe appearance variations, the deformable convolution layer aims to generate a normal response similar to those in the ordinary scenarios by estimating the free-form deformation. In addition, although the training samples collected online are limited and similar, the deformable convolution is capable of adapting to unseen deformations via online learning.

On the other hand, relying solely on the deformable convolution may have some limitations, e.g., the degradation of scale estimation and localization. This is because it may treat scaling and shifting as some forms of deformations, and thus try to recover those negative samples and categorize them as positive

Wenxi Liu, Dengsheng Chen, and Yuanlong Yu are with the College of Mathematics and Computer Science, Fuzhou University, China.
Yibing Song is with the Tencent AI Lab, Shenzhen, China.
Tao Yan is with the Jiangsu Key Laboratory of Media Design and Software Technology, Jiangnan University, China.
Shengfeng He is with the School of Computer Science and Engineering, South China University of Technology, China.
Gerhard P. Hancke and Rynson W.H. Lau are with City University of Hong Kong, Hong Kong.
Shengfeng He and Yuanlong Yu are the corresponding authors. This project is led by Rynson W.H. Lau.

samples. We note that when the target object is in an ordinary scenario with minor appearance variations, the well-trained normal convolution features are effective. When the target object has significant appearance variations, the deformable convolution will be more effective. Hence, we introduce a soft gate mechanism to balance between the normal convolutional features and the deformable convolutional features. The soft gate approves the fusion between the deformable features and the original ones when the appearance variations of the target object are accurately modeled. Specifically, the soft gate adaptively blends these two types of features. As a result, the gated fusion of the normal convolutional features and the deformable ones will recover the rapid appearance variations of the target object into ordinary conditions. The gated fusion produces accurate deformable feature maps for the target object, which in turn enrich target appearances and facilitate the classifier prediction. In addition, the deformable convolutional layer with gated fusion is integrated into the tracking-by-detection framework for end-to-end training and prediction. Extensive experiments on the standard benchmarks indicate that the proposed tracker performs favorably against the state-of-the-art methods.

We summarize the contributions of this work as follows:

- We present a deformable convolutional layer for the CNN-based tracking-by-detection framework to model target appearance variations.
- We propose a gated fusion mechanism to control the effect of the deformable convolutional layer on the output feature maps. It facilities the classifier's discrimination on the target object and background.
- Our tracker outperforms state-of-the-art approaches on the standard tracking benchmarks, especially on those challenging scenarios.

For the rest of the paper, we first survey related literatures in Section II. We then present our deformable object tracker with gated fusion in Section III. Finally, we evaluate the performance of our proposed tracker in several public benchmarks in Section IV.

II. Related Works

In visual tracking, state-of-the-art trackers can be roughly categorized as: tracking-by-detection based methods, Siamese-network based methods, reinforcement-learning based methods, and deformable object trackers.

A. Tracking-by-detection Based Approaches

The tracking-by-detection framework considers the tracking task as a target/background classification problem. Numerous learning schemes have been proposed including P-N learning [1], online boosting [2], multiple instance learning [3], structured SVMs [4], CNN-SVMs [5], random forests [6], domain adaptation [7], LSTM-based [8], adversarial learning [9], reciprocative learning [10] and ensemble learning [11]. Our proposed method is based on a CNN-based tracking-by-detection framework. Here, we focus on handling the challenging tracking task: tracking deformable objects while maintaining high-quality tracking performance in ordinary situations. To accomplish this, we introduce a gated fusion module to adaptively approve the fusion of the normal convolutional features and the deformable convolutional features.

In recent years, the discriminative correlation filter (DCF) based trackers [12], [13] are widely studied. They gain much attention due to their real-time performance. In essence, they are related to tracking-by-detection methods, since they efficiently learn a discriminative regressor from foreground and background samples. In particular, they regress all the circular-shifted samples into soft labels and transfer the correlation as an element-wise product in the Fourier domain. However, most of the DCF based methods suffer from the boundary effect and the model overfitting issues, which are caused by the circular-shifted samples and the dense training samples collection. To handle these issues, there are many extensions, including: kernelized correlation filters [13], scale estimation [14], re-detection [15], spatial regularization [16], [17], [18], ADMM optimization [19], deep feature integrations [20], [21], [22], [23], [24], and end-to-end CNN predictions [25], [26], [27].s

B. Siamese-Network Based Approaches

Siamese-network based approaches have become popular in the visual tracking community, due to their balanced accuracy and speed [28], [29], [30], [31], [32], [33]. Unlike the tracking-by-detection methods, these methods formulate the object tracking as a similarity learning problem. By comparing the ground-truth patch of the target object with the candidate patches within the search window at the current frame, the most similar patch is considered as the target. The similarity learning is accomplished by a fully convolutional Siamese network framework, which receives a pair of inputs and outputs the similarity. Most of these methods require little or no online training. Hence, they can reach real-time performance and are less affected by drifting caused by the online updating. However, they require a large amount of training data in the offline stage to achieve the state-of-the-art performance.

C. Reinforcement-Learning Based Approaches

Recently, researchers introduce deep reinforcement learning (DRL) into visual tracking. Deep reinforcement learning utilizes the deep neural network to model the active-value function to play games (e.g., Atari [34]), which reaches human-level performance. As a potential research direction in tracking, prior works begin to use DRL to learn actions for robust tracking [35], [36], [37], e.g., shifting and scaling the tracking window. The tracking related policy is learned from the training data in an offline manner.

D. Deformable Object Trackers

Tracking deformable objects is an important problem in visual tracking [38], [39], [40], [41], [42], [43], [44]. There are prior works focusing on tracking non-rigid deformable objects while segmenting their contours, including dynamic graph tracker [45], temporally coherent part-based tracker [46], local and global tracker [47], superpixel tracker [48], adaptive structural local sparse appearance model [49], latent structural

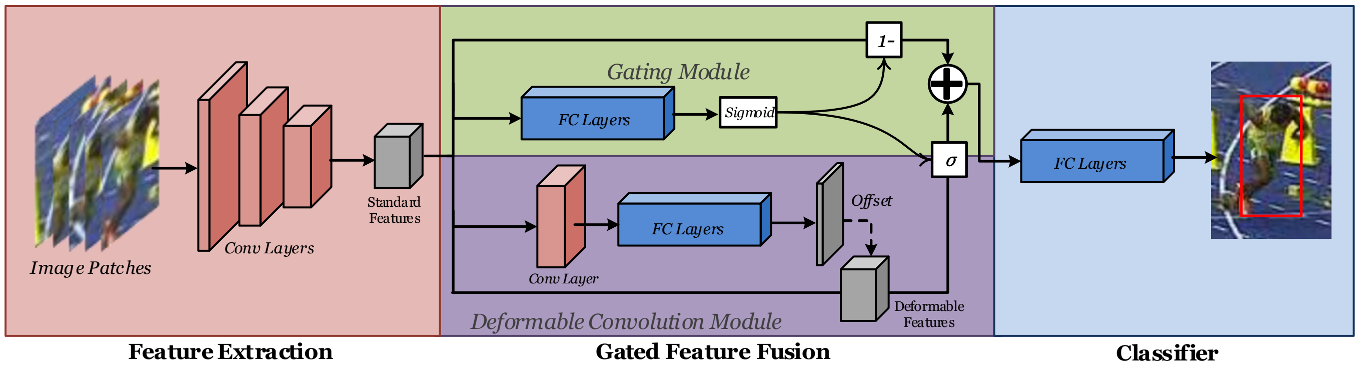


Fig. 2. Our proposed framework is composed of three stages: (1) feature extraction consisting of pretrained convolutional layers, (2) gated feature fusion, and (3) classifier consisting of fully connected layers. Our proposed method focuses on the gated feature fusion, which includes a deformable convolution module, and a gating module that controls the fusion of the deformable features and the standard features.

learning tracker [50], sparsity-based collaborative model [51]. The general approach is to divide the object into parts. Local connectivity is typically applied among different parts of the object and a trade-off of the visual and geometric agreement is then optimized. In [40], a shape-preserved kernelized correlation filter based tracker is proposed to use a set of deformable patches dynamically to collaborate on tracking of non-rigid objects. In order to track deformable objects, Du et al. [39] capture the high-order temporal correlations among target parts by a structure-aware hyper-graph. However, the overall tracking performance of these methods is not comparable to the holistic methods. In this paper, we introduce deformable convolution that can both model the appearance of the deformable object as well as be incorporated into the CNN-based framework and trained in an end-to-end manner. Therefore, it can achieve state-of-the-art performance on public benchmarks.

III. Deformable Tracker with Gated Fusion

Our approach is designed based on the tracking-by-detection framework, which includes the deformable convolution module and the gating module, as shown in Fig. 2). In particular, a set of candidate patches are randomly sampled from the current video frame and are fed to the network as input and its feature maps are obtained via several pretrained convolution layers (i.e., Conv layers) in the first stage. These standard feature maps are then sent to separate branches of the second stage. From Fig. 2, for the bottom two branches, the deformable convolution module resamples the standard feature maps to produce the deformable convolutional features. On the other hand, the top branch of the gated feature fusion stage keeps the standard convolutional features, while the second branch is for the gating module to infer the weights that balance the effect of the standard convolutional features and the deformable features. In the final stage, the fused features are sent to the fully connected layers (i.e., FC layers), which serve as a foreground/background classifier to detect the object in an online manner. In the following subsections, we introduce these modules and our tracking framework.

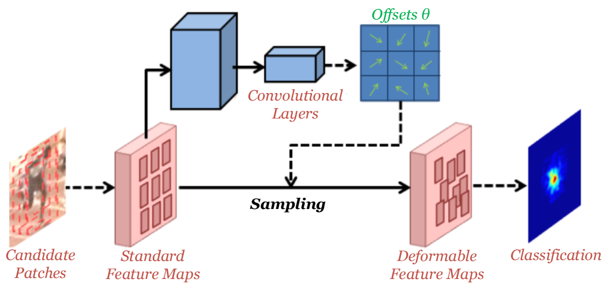


Fig. 3. Illustration of the *Deformable Convolution Module*. We demonstrate how the deformable features are computed. The standard feature maps are first passed through the convolutional layers to regress the deformation offsets Θ. The offsets tell how the feature maps should be deformed and thus how they should be sampled.

A. Deformable Convolution

In order to model object variations in tracking, we introduce a deformable convolution module. As shown in Fig. 2, the deformable convolution module is inserted in one of the branches of the CNN-based framework for extracting the deformable features. Fig. 3 illustrates how the deformable convolution module works.

To learn how to deform the feature maps, we are inspired by [52]. As shown in Fig. 3, the feature maps (the pink rectangle) are passed through an extra branch containing multiple convolutional layers (the blue rectangles) to regress the 2D offsets of the regular grid sampling locations. Then, the offsets are applied back to the feature maps and produce the new feature maps by resampling. Formally, the feature maps $\mathbf{X}$ ($\mathbf{X} \in \mathbb{R}^{H\times W\times C}$) are passed to the convolutional layers, which can be denoted as a non-linear function $\mathcal{F}_{deform}$. Thus, its output is reshaped and regresses the deformation offsets of the sampling locations Θ ($\Theta \in \mathbb{R}^{H\times W\times 2}$), as follows:

$$\Theta = \mathcal{F}_{deform}(\mathbf{X}), \tag{1}$$

where the convolutional layers consist of a convolutional layer with 3×3 kernel size followed by a fully connected layer whose output has the same size as Θ, i.e., $H * W * 2$. In particular, $\theta_{i,j}$ ($\theta_{i,j} \in \Theta$) refers to the 2D offset vector that directs how the element $\mathbf{X}_{i,j}$ ($\mathbf{X}_{i,j} \in \mathbf{X}$) in the standard feature maps deforms. Hence, the deformable feature maps are calculated as follows:

$$\begin{aligned} \mathbf{X}'_{i,j} &= \mathcal{F}_{sample}(\mathbf{X}, [i,j] + \theta_{i,j}), \\ \text{s.t. } &\theta_{i,j} \in \mathbb{R}^{1\times1\times2}, \theta_{i,j} \in \Theta, \\ &1 \le i \le H, 1 \le j \le W, \end{aligned} \tag{2}$$

where $\mathbf{X}'_{i,j}$ are the deformable features at location $[i,j]$, and are sampled from location $[i,j]+\theta_{i,j}$, instead of location $[i,j]$, with the shape of the offset vector $\theta_{i,j}$ transformed to $\mathbb{R}^{1\times2}$ in advance. Note that $\theta_{i,j}$ is often fractional. So, a sampling kernel is applied to the feature maps as shown in Fig. 3. Here, $\mathcal{F}_{sample}(\mathbf{X}, [d_x, d_y])$ is a bilinear interpolation kernel which samples $\mathbf{X}$ at location $[d_x, d_y]$, as follows:

$$\mathcal{F}_{sample}(\mathbf{X}, d) = \sum_{x=1}^{W}\sum_{y=1}^{H} G([x,y],[d_x,d_y])\mathbf{X}_{x,y}, \tag{3}$$

$$G([x,y],[d_x,d_y]) = \max(0, 1-|x-d_x|)\max(0,1-|y-d_y|), \tag{4}$$

where $\mathcal{F}_{sample}(\mathbf{X}, [d_x, d_y])$ samples the feature maps by calculating the weighted sum of the features at neighboring locations. $G(\mathbf{a}, \mathbf{b})$ computes the weights of features at location $\mathbf{a}$ for sampling the features at location $\mathbf{b}$.

As we know, the standard CNNs perform the convolution operation over a fixed geometric structure, which is not reasonable for handling the object with significant appearance variations. For example, when the athlete in Fig. 7 runs, his body parts rapidly change appearances and locations. The standard convolutional features cannot adapt to such variation well. On the contrary, the deformation convolution roughly estimates how the visual parts of the object will move by $\mathcal{F}_{deform}$ and then resamples the feature maps by bilinear interpolation $\mathcal{F}_{sample}$. It estimates the feature maps after object deformation and thus the approximated feature maps serve as the deformable features, which recovers the deformed object features into the standard ones. Since the deformation is simply modeled by the offsets of the sampling grids, it supports free-form deformation (e.g., rigid or non-rigid deformation, pose variation of human body and rotation) instead of affine transformation in STN [53]. Therefore, it is useful for tracking objects with not only non-rigid deformation but also in-plane rotation. Besides, the performance of the standard CNNs is degraded because the training samples collected online are limited and similar. Due to the online estimated offsets, the deformable convolution can adaptively model the unseen deformation in an online manner.

In Fig. 4, we illustrate an example of the effect of the deformable convolutional features during online tracking. As shown, we select some exemplar frames in which the target rotates and changes its pose from frames #2 to #79. We observe that the deformable convolution module adapts to the pose variation by extracting features according to the learned deformation offsets. Note that in frame #86, when the target recovers back to the original pose, the deformable convolution features also recover to the standard convolutional features. In summary, the deformable convolutional features are adaptive when capturing the appearance variations of the deformable object online.

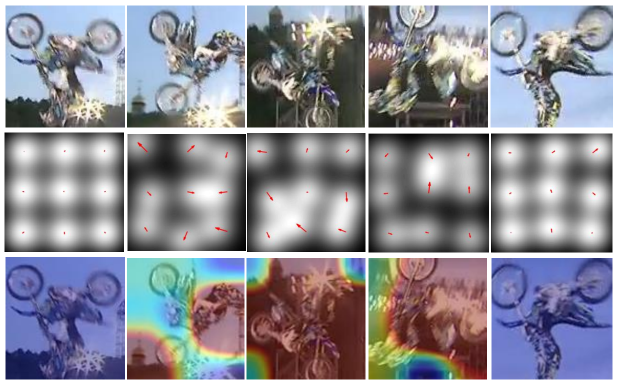


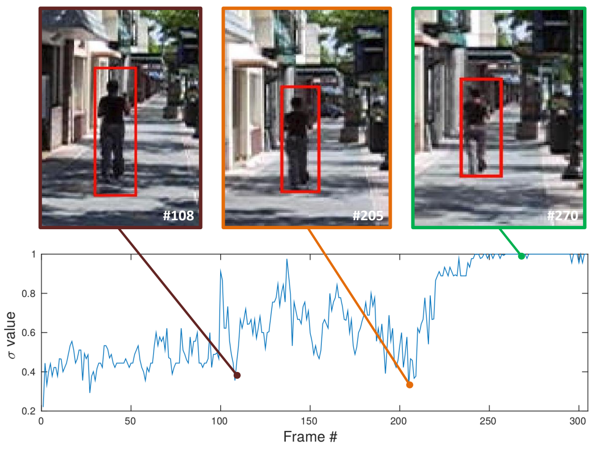
Fig. 4. In the first row, the tracked image patches from frames #2, #21, #31, #79, and #86 of the *MotorRolling* sequence are shown. In the second row, the deformation offsets are visualized. The bright regions and the red arrows indicate how the deformation offsets affect the feature extraction. In the third row, the differences between the standard convolutional feature maps and the deformed convolutional feature maps are highlighted. We select the feature maps from the 30^{th} channel to compute the difference, which is mapped back to the corresponding image patch for visualization. Warmer colors indicate larger differences.

B. Gated Fusion

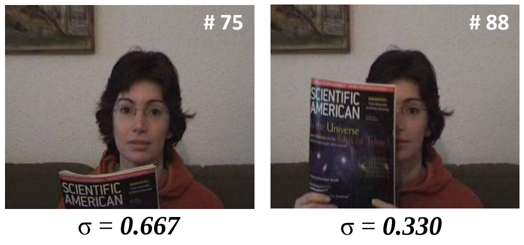
In practice, the deformable convolution layer may not perform well and its output feature maps will be erroneous. So, it needs to be compensated for by the standard convolutional features. Here, we introduce a gating module to control the fusion of both features, as shown in Fig. 2. The gating mechanism is first proposed in Long-Short Term Memory (LSTM) cells [54] for regulating the information flow through the network.

In our framework, we introduce a soft-gate to adaptively fuse the deformable convolutional features and standard convolutional features. The motivation of applying the gating module is to enhance the features fusion between the deformable convolutional features and the standard convolutional features. As mentioned in Sec. III-A, when tracking an object with severe appearance variations, the standard convolutional features may fail due to the unseen appearance of the object, while the deformable convolutional features can to some extent adapt to the deformation. However, the deformable convolution also has limitations. As we observe, in ordinary scenarios, the standard convolutional features demonstrate more robust performance in tracking objects with few appearance variations. In addition, in some extreme situations, such as significant illumination changes or severe occlusion, the deformation convolution cannot accurately estimate the deformation offsets, and it may produce in higher errors than the standard convolutional features. Thus, we do not only switch between these features, but also combine them adaptively as well.

The output of the gating module is learned by another branch of layers, $\mathcal{F}_{gate}$, followed by a sigmoid activation function, $\mathcal{F}_{\sigma}$. The sigmoid layer constrains the gating output to be within $[0,1]$, which serves as a weight that balances which type of features should be dominant. Thus, the output of the



(a)



(b)

Fig. 5. We demonstrate how the online learned gating output (i.e., the σ value) works in tracking. (a) shows how the σ value changes when tracking video *Human9*. We highlight three keyframes and their corresponding σ values. (b) shows the σ values of the two frames from video *FaceOcc1*.

gating module is:

$$\sigma = \mathcal{F}_{\sigma}(\mathcal{F}_{gate}(\mathbf{X})). \tag{5}$$

To better control the fusion, we set the dimension of the output as $\sigma \in \mathbb{R}^{H \times W}$, which is the same as the spatial dimension of the standard convolutional features and the deformable convolutional features. Given the deformable convolutional features $\mathbf{X}'$ and the standard convolutional features $\mathbf{X}$, the fusion process is computed as:

$$\mathbf{Y} = \mathbf{X}' \odot \sigma + \mathbf{X} \odot (1 - \sigma), \tag{6}$$

where $\odot$ indicates the element-wise multiplication, and $\mathbf{Y}$ refers to the fused features. When an object shows appearance variations, the gating module should compute a large σ. Thus, the deformable convolutional features will be dominant in the fused features. When the deformable convolution cannot achieve good performance, the gating module will output a small σ value.

In Fig. 5, we illustrate two examples of how the gating module works in online tracking. The gating module is offline trained with diversified samples and then performs online updating. To simplify the analysis, we compute the average value of σ. If the σ value is equal to 0, it means that there is no flow passing through the gate in the branch of the deformable convolution module and the fused features will be completely from the standard features. If the σ value is equal to 1, the full flow of the deformable convolution will be enabled and the standard features will not be used.

Fig. 5(a) illustrates how the gating output (i.e., the σ value) is online learned in the $Human9$ sequence, in which a person walking in the street with appearance deformation, scaling and illumination variations. We highlight three keyframes and their corresponding σ values. As we observe, when the person walks into the shadow in frames #108 and #209, σ drops to below 0.4 as the illumination sudden changes, making it difficult for the deformable features to model the target appearance accurately. Thus, the standard convolutional features take charge. In the end, σ increases to 1 as the person walks out of the shadow and its appearance is well observed again. Thus, the deformable convolutional features are used. Note that in the beginning of tracking, the gating output is relatively low because the gating module is under initialization by collecting training samples. Fig. 5(b) reflects that the deformable convolutional features may fail in occlusions. Thus, the standard convolutional features will take the major role in the fusion. As observed, when the face is occluded, the σ value significantly drops from 0.667 to 0.330.

In summary, our proposed framework mainly consists of two modules: the deformable convolution module and the gating module. The deformable convolution module generates the deformable features to handle the object appearance variations, while the gating module adaptively fuses the deformable convolutional features and standard ones in an online manner.

C. Tracking

As depicted in Fig. 2, our approach leverages the pretrained VGG-M [55] as the front end to extract standard features in the first stage. In the second stage, the extracted standard feature passes through some parallel branches for gated feature fusion. In the final stage, the fused features are transferred to fully connected layers, which serve as a non-linear classifier.

Model training. We train our model using positive and negative samples from the training data offline. We prepare the training data following [7]. For offline training, 50 positive and 200 negative samples are collected from each frame, where positive and negative examples have ≥ 0.7 and ≤ 0.5 IoU overlap ratios with ground-truth bounding boxes, respectively. To train the deformable convolution layers and the gating module, we follow three steps: 1) training the network without deformable convolution and gating; 2) training the network with the deformable convolution only; 3) training the network with both modules. Hence, in steps 1 and 2, we actually fine-tune the standard convolutional feature and the deformable convolutional feature, respectively. In step 3, we train the gating module to make sure the tracker can adaptively switch between both modules. All of them are trained in an end-to-end manner.

Model initialization. With the trained model, we fine-tune the network using the samples as the candidate proposals from the first frame of the input sequence. The front end of the network is frozen and only the gating feature fusion and the fully connected layers are updated. To estimate the scale of the object, following the practice of [7], we train a bounding box regression using a simple linear regression model to predict the precise target location using $conv3$ features of the samples near the target location at the first frame.

Model update. We incrementally update the tracker online. Around the estimated position, we generate multiple samples and assign them with binary labels according to the intersection-over-union ratios with the estimated bounding box. The optimal target state is the drawn sample with the highest positive score. In addition, we adopt hard negative mining in mini-batch sampling. The hard negative samples are selected from the classification confidence of samples, which are evaluated in our network. In particular, we select the negative samples with top confidence scores as the hard negative samples, which are used for updating the model.

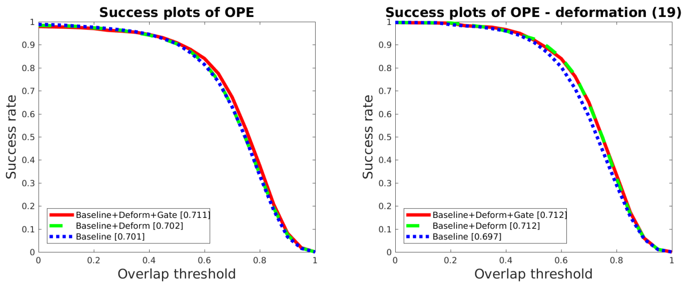


Fig. 6. Success plots on all sequences and the sequences with the deformation attribute in OTB-2013 dataset using the one-pass evaluation. The numbers in the legend indicate the area-under-the-curve success scores.

IV. Experiments

In this section, we introduce the implementation details of our proposed model and analyze the effects of the modules in the network. We refer to our tracker as **GDT** (**G**ated-fusion **D**eformable object **T**racker) and we compare GDT with state-of-the-art trackers in the benchmark datasets: Deform-SOT [39], OTB-2013 [56], OTB-2015 [57], VOT-2016 [58], and VOT-2017 [59] for performance evaluation.

A. Implementation Details and Experimental Setup

Network architecture. First, our feature extraction network is based on the first three convolutional layers from the VGG-M model [55]. Second, for the deformable convolution module, it consists of a convolutional layer and a fully connected layer, which generates the deformation offsets of $3 \times 3 \times 2$. Then, the feature maps are reconstructed according to the offsets by a bilinear sampler. For the gating module, it is produced by two consecutive fully connected layers followed by a sigmoid activation and outputs a 3×3 gating values. Finally, the classifier consists of three consecutive fully connected layers, with a softmax loss as the objective function.

Implementation details. For offline training the network, we apply the Sochastic Gradient Descent (SGD) solver to run for $200K$ iterations. The learning rate of the feature extraction convolutional layers is set as 10^{-4}, and that of the fully connected layers is set as 10^{-3}. For the evaluation in OTB, the training data is collected from labeled sequences of VOT challenges excluding the sequences from OTB-100. For the evaluation in VOT-2016 and VOT-2017, we use OTB data for training excluding the sequences in the test sets. For the test in Deform-SOT, our model is trained using the data from OTB and VOT excluding the test sequences used in Deform-SOT. At the initialization stage, we train the feature fusion and the fully connected layers for 30 iterations with a learning rate 0.0003 except for the last layer, which is 0.001. For online update, the feature extraction layers are frozen and the rest of the network is fine-tuned based on online collected samples. We assign a learning rate of 0.0005 for the feature fusion module. The momentum and weight decay are always set to 0.9 and 0.0005, respectively. Each mini-batch consists of 32 positives and 96 hard negatives selected out of 1024 negative examples. Our proposed tracker GDT runs on a PC with an i7 3.6GHz CPU and a NVIDIA Geforce 1080Ti GPU with the MatConvNet toolbox [60]. In tracking, the initial training time is around 30 seconds on average and the running time of tracking is 1.33 FPS. The tracker is updated every 10 frames online and the updating time is around 3 seconds.

Evaluation metrics. We follow the standard evaluation approaches. In the Deform-SOT, OTB-2013 and OTB-2015 datasets, we use the one-pass evaluation (OPE) with precision and success plots metrics. The precision metric measures the frame locations rate within a certain threshold distance from the groundtruth locations. The threshold distance is set to 20 pixels. The success plot metric is set to measure the overlap ratio between the predicted bounding boxes and the groundtruth. For the VOT-2016 [58] and VOT-2017 [59] datasets, we conduct comprehensive experiments including reset-based experiments and no-reset (unsupervised) experiments. The reset-based experiments are measured in terms of Expected Average Overlap (EAO), Accuracy and Robustness. Specifically, robustness is computed as $\exp(-S \cdot F)$, where F refers to the tracking failure rate and S is set as 100, according to [61]. The unsupervised experiments are evaluated in terms of Average Overlap (AO). Note that in the tables included in the following subsections, we mark the best, second best and third best results in *red/blue/green* colors, respectively.

B. Ablation Studies

To validate the effectiveness of each module, we separately train the following three models: network without the two modules (Baseline), network with the deformable convolution module only (Baseline+Deform), and the complete network (Baseline+Deform+Gate). We compare the performance of these models on the OTB-2013 dataset. In Fig. 6(left), we evaluate the overall tracking performance by the success rate metric on the entire dataset. In Fig. 6(right), we measure the tracking performance on the videos containing the *deformation* attribute.

As shown in Fig. 6, the deformable convolutional features help slightly boost the overall success rate from 0.701 to 0.702, but they help improve the capability of tracking deformable objects more significantly (success rate from 0.697 to 0.712). This demonstrates that the deformable convolutional features indeed benefit tracking deformable objects. However, the overall tracking capability is not obviously improved. The reason is that, as we have mentioned in Sec. III-B, the deformable convolutional features are not that robust as the standard convolutional features. After incorporating the gating module, the overall performance improves more significantly (from 0.702 to 0.711), while not degrading the capability of tracking deformable objects (remaining at 0.712). This result indicates that the gating module supplements the deformable convolutional features and helps improve the general tracking performance. More importantly, the capability of tracking deformable objects is not affected, which means that the gating module can adaptively switch the fusion to the deformable convolutional features when tracking deformable objects. As illustrated in Fig. 6(c-f), the deformable convolutional features do not significantly improve or even degrade the tracking performance. However, as we can observe, the inclusion of the gating module plays an important role to boost the performance.

Deformable convolution. To demonstrate the effects of the deformable convolution module, we choose the video sequences *MotorRolling*, *Bolt*, *Bolt2*, and *Diving*, which contain in-plane rotation and deformation, from the OTB-2015 dataset for testing. First, as shown in Fig. 4, with the pose variation, the deformable convolution module adaptively focuses on the rotating body of the target. Second, in Fig. 7, we densely draw image patches from the frames #15, #21, #26, #35 in the *Bolt2* video and pass them to the network to compute their classification confidence scores. This figure shows the locations of the drawn samples and their corresponding confidence scores. As observed, without deformable features, the classifier is uncertain around the target region when the athlete runs with appearance variations. In frames #15 and #21, the high confidence samples disperse, which indicates that the target pose affects the performance of the tracker. In frame #35, the confident samples are gathered in two clusters, which means that the tracker is confused by the target object and the neighboring object. In contrast, the high confidence area of the classifier with the deformable convolutional features is more concentrated on the target object, even though the target object exhibits some unseen poses. Third, in Fig. 8, the differences between the deformable features and the standard features are visualized. As observed, the major differences correspond to the body parts with significant deformation. This toy experiment reflects the adaptive effects of the deformable convolution module during online tracking.

C. Quantitative Evaluation

Deform-SOT Dataset. This dataset contains 50 challenging video sequences [39] that focus on tracking totally deformable targets in unconstrained environments. The annotated attributes include scale change, severe occlusion, abnormal movement, illumination variation and background clutter. Following the experiment setup and the results provided in [39], we compare with 8 part-based trackers: Structure-aware hypergraph based tracker (SAT [39]), Dynamic graph tracker (DGT [45]), Temporally coherent part-based tracker (TCP [46]), Local and global tracker (LGT [47]), Superpixel tracker (SPT [48]), Adaptive structural local sparse appearance model (ASLA [49]), Latent structural learning tracker (LSL [50]), Sparsity-based collaborative model (SCM [51]). We evaluate our GDT along with these trackers on 50 video sequences using the one-pass evaluation based on two metrics: distance precision and overlap success.

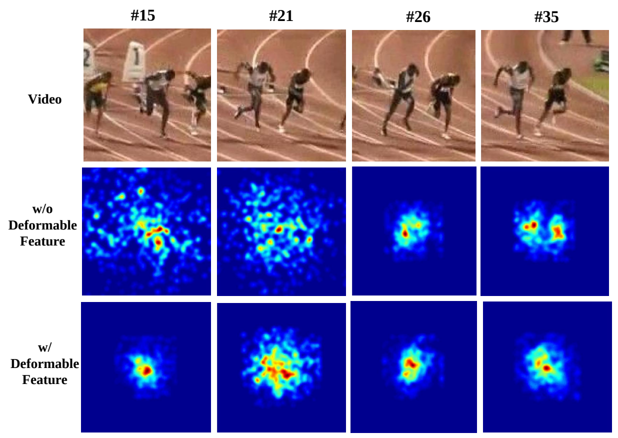


Fig. 7. A comparison of the predicted probabilities (i.e., confidence scores) from the classifier with and without the deformable convolutional features. To visualize this, we densely draw the samples from the region of interest and measure their confidence scores. Their classification confidence scores with regards to their locations are illustrated by heatmaps. The exemplar images are selected from the *Bolt2* video in the OTB-2015 benchmark.

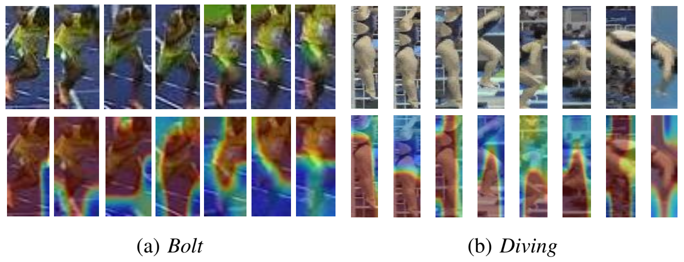


Fig. 8. In the first row of (a) and (b), the exemplar tracked image patches from *Bolt* and *Diving*, respectively, are shown. In the second row, the differences between the standard convolutional feature maps and the deformed convolutional feature maps are highlighted. In (a), patches from frames #3, #12, #29, #41, #54, #58, and #66 are selected for visualization, while in (b), patches are selected from frames #8, #16, #31, #44, #47, #97, #119, and #199.

In Fig. 9, we show the results of the comparison with the state-of-the-art deformable object trackers. The top-left first two diagrams show the overall performances of the overlap threshold and the location error, in which our approach significantly outperforms other methods. In the remaining diagrams, we show the results on videos annotated as scale change, severe occlusions, background clutter, abnormal movement, and illumination variation. Again, our approach performs better than all the other part-based trackers.

OTB-2013 Dataset. We compare GDT with 29 trackers from the OTB-2013 benchmark [56] and other 27 state-of-the-art trackers including TCNN [62], EBT [63], DSST [14], KCF [13], MEEM [64], LCT [15], MUSTer [65], HCFT [20], FCNT [25], SRDCF [18], CNN-SVM [5], DeepSRDCF [66], Staple [67], SRDCFdecon [68], CCOT [17], SiamFC [28], MDNet [7], ADNet [69], ECO [16], MCPF [22], and CREST [26]. We evaluate all the trackers on 50 video sequences using the one-pass evaluation with distance precision and overlap success metrics.

Fig. 10 shows the quantitative results from the compared trackers. For presentation clarity, we only show the top 10 trackers. The numbers listed in the legend indicate the AUC overlap success and 20 pixel distance precision scores. Overall,

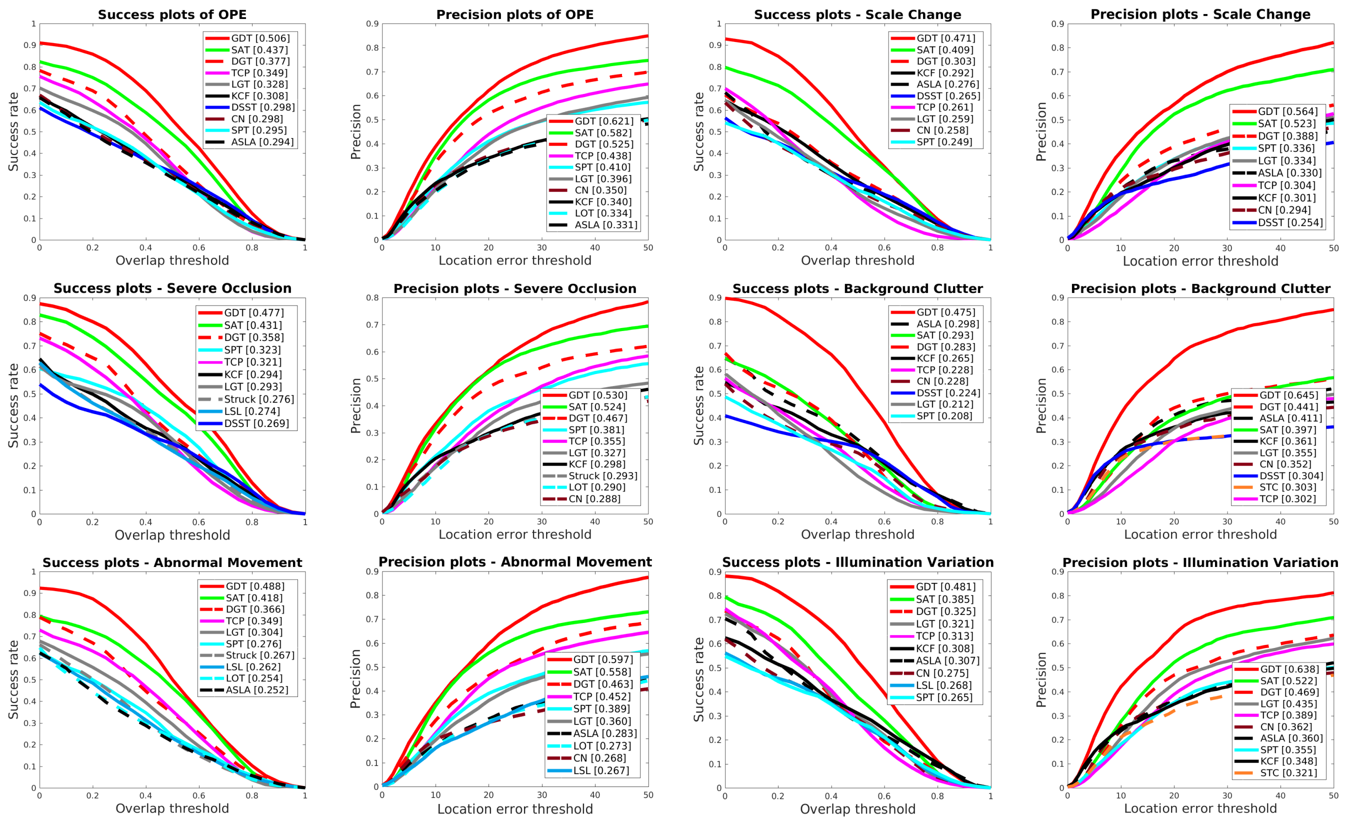


Fig. 9. The top-left first two diagrams show the overall performances on the complete Deform-SOT dataset. The other diagrams show overlap success plots and precision plots over sequences annotated with *scale change, severe occlusion, background clutters, abnormal movements*, and *illumination variations* in the dataset.

our GDT performs favorably against state-of-art trackers in both distance precision and overlap success. GDT achieves the top success rate (0.711) compared to other methods including ECO and MDNet. For the distance precision, GDT performs better than other trackers except for MDNet, which is similar to our baseline. It means that the deformable convolution slightly decreases the robustness of the model.

In addition, Fig. 12 compares the performance of selected six video attributes including *deformation, in-plane rotation, out-of-plane rotation*, and *illumination variation*, using one-pass evaluation. As can be seen from the results, our tracker GDT generally outperforms most of the state-of-the-art trackers. In particular, our tracker handles large appearance variations caused by deformation, in-plane and out-of-plane rotations better than all of the other methods. It indicates that the deformable convolution module enhances the capability of modeling a variety of appearance variations. In addition, the gating module blends the standard convolutional features and the deformable convolutional features, which to some extent can model the unseen appearance variations by approximately recovering them to the normal conditions.

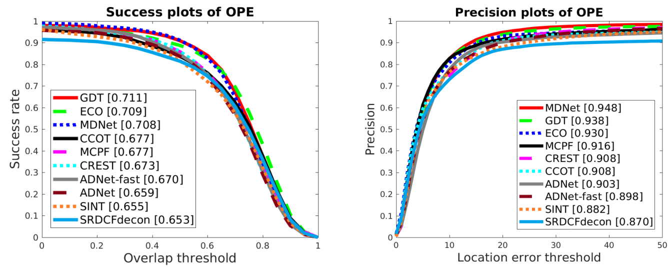


Fig. 10. Precision and success plots on OTB-2013 using one-pass evaluation.

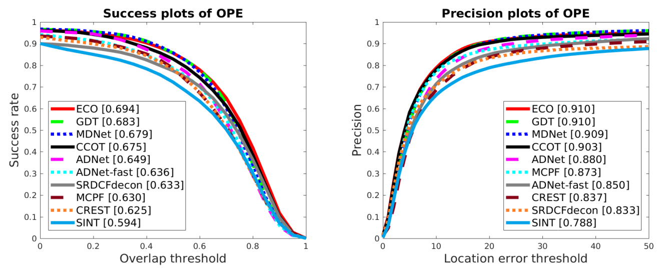


Fig. 11. Precision and success plots on OTB-2015 using one-pass evaluation.

OTB-2015 Dataset. We compare GDT with the state-of-the-art trackers on the OTB-2015 benchmark [57]. Fig. 11 shows that our proposed tracker GDT performs generally well. Although the ECO tracker achieves the best result in overlap success, GDT performs the same as ECO in distance precision. As far as we know, ECO is the state-of-the-art tracker in several public benchmarks, which improves continuous convolutional operators on a multi-level feature maps. Since the OTB-2015 dataset contains more challenging videos with fast motion and low resolution, our tracker fails to match up with ECO on success rate. Besides, our method outperforms

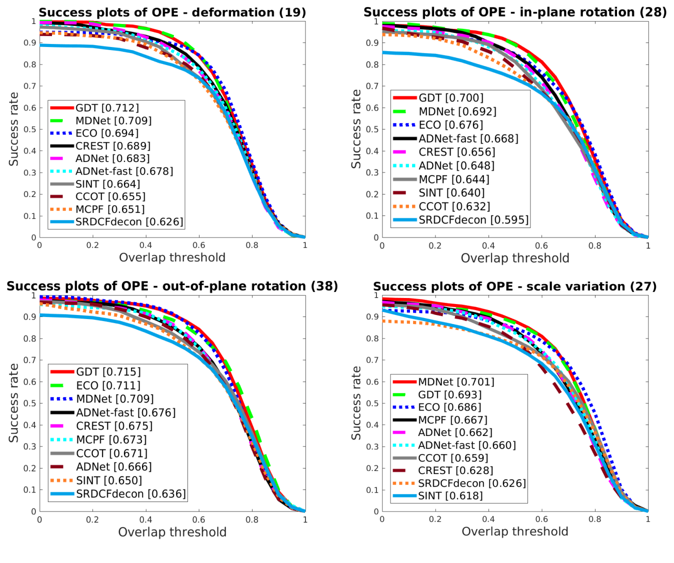


Fig. 12. Overlap success plots over tracking challenges with deformation, in-plane rotation, out-of-plane rotation, and scale variations on OTB-2013.

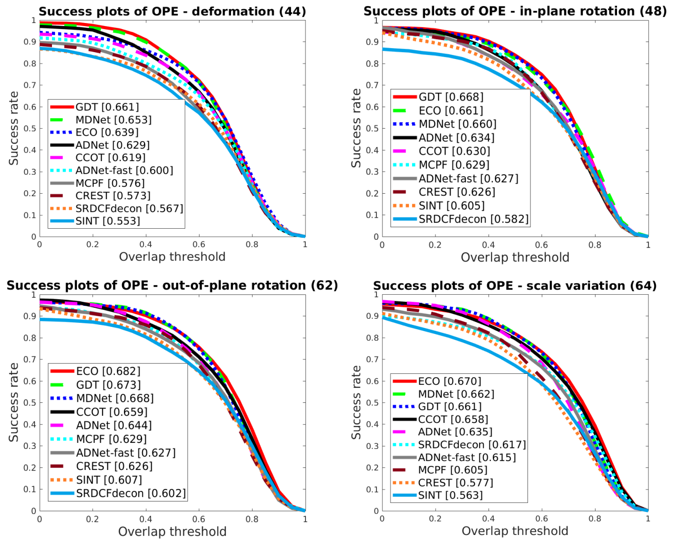


Fig. 13. Overlap success plots over tracking challenges with deformation, in-plane rotation, out-of-plane rotation, and scale variations on OTB-2015.

all of other methods on both success rate and precision. In addition, we also illustrate the tracking performance for videos annotated as *deformation, in-plane rotation, out-of-plane rotation*, and *illumination variation* in Fig. 13. Our model shows advantages in tracking deformable objects and objects with in-plane rotation as well.

VOT-2016 Dataset. We compare GDT with state-of-the-art trackers on the VOT-2016 benchmark, including ECO [16], Staple [67], MDNet [7], and CCOT [14]. Table I shows that GDT performs comparably to the state-of-the-art trackers on the EAO metric. GDT only performs worse than ECO on EAO and ranks at the top in accuracy. Considering the slight loss of robustness induced by the deformable convolution, our GDT's robustness is behind ECO and CCOT. In Fig. 14, we analyze EAO by comparing GDT with 41 other trackers, including ECO, CCOT, TCNN, Staple, and EBT. GDT has the second highest expected average overlap among them. Since VOT-2016 consists of many short-term challenging tracking scenarios with severe appearance deformation (e.g., *motorcross1*, *gymnastics1*), our approach outperforms most of the other methods in these scenarios.

In Tab. II, we analyze the performance of these trackers on videos with different attributes based on the accuracy and robustness metrics. The annotated attributes include camera change (Cam.), illumination change (Illu.), motion change (Mot.), occlusion (Occ.), size change (Size), and not assigned (N/A). For accuracy, our approach has obvious advantages for most attributes. Although GDT is less robust than CCOT and ECO, we notice that with regard to the size change attribute, our method outperforms the others. Tab. V shows the results of the unsupervised (non-reset) experiment. Our approach ranks the top on the overall performance and three attributes. Among them, occlusion and size change are closely related to deformable object tracking, in which our approach gains obvious advantages.

VOT-2017 Dataset. We compare GDT with state-of-the-art trackers on the VOT-2017 benchmark, including ECO [16], Staple [67], SiamFC [28], and CCOT [14]. Overall, our

TABLE I
COMPARISON WITH THE STATE-OF-THE-ART TRACKERS ON THE VOT-2016 DATASET. THE RESULTS ARE PRESENTED IN TERMS OF EXPECTED AVERAGE OVERLAP (EAO), ACCURACY, AND ROBUSTNESS. NOTE THAT THE BEST, SECOND BEST AND THIRD BEST RESULTS ARE MARKED IN *red/blue/green* COLORS, RESPECTIVELY. (THE SAME COLOR CODING SCHEME IS USED IN ALL REMAINING TABLES.)

	ECO	CCOT	Staple	MDNet	**GDT**
EAO	0.374	0.331	0.295	0.257	0.353
Accuracy	0.555	0.541	0.547	0.542	0.585
Robustness	0.818	0.788	0.686	0.714	0.774

TABLE II
TRACKING ACCURACY (A) AND ROBUSTNESS (R) ON THE VOT-2016 DATASET. THE ATTRIBUTES TESTED INCLUDE CAMERA CHANGE (CAM.), ILLUMINATION CHANGE (ILLU.), MOTION CHANGE (MOT.), OCCLUSION (OCC.), SIZE CHANGE (SIZE), AND NOT ASSIGNED (N/A).

		Cam.	Illu.	Mot.	Occ.	Size	N/A	All
A	ECO	0.576	0.646	0.500	0.408	0.519	0.595	0.555
	CCOT	0.558	0.656	0.475	0.437	0.507	0.578	0.541
	Staple	0.555	0.715	0.511	0.437	0.516	0.583	0.547
	MDNet	0.547	0.639	0.508	0.491	0.511	0.563	0.542
	GDT	0.566	0.667	0.558	0.536	0.557	0.610	0.585
R	ECO	0.834	0.904	0.708	0.490	0.817	0.932	0.818
	CCOT	0.748	0.818	0.666	0.555	0.769	0.857	0.788
	Staple	0.662	0.495	0.491	0.365	0.739	0.833	0.686
	MDNet	0.671	0.683	0.647	0.582	0.784	0.771	0.714
	GDT	0.707	0.857	0.688	0.521	0.850	0.872	0.774

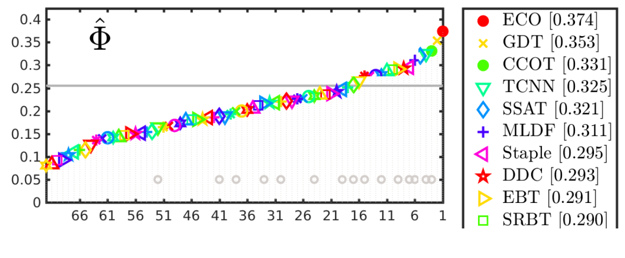


Fig. 14. Comparison with more state-of-the-art trackers on the VOT-2016 dataset. The results are presented specifically in terms of expected average overlap (EAO).

TABLE III
COMPARISON WITH THE STATE-OF-THE-ART TRACKERS ON THE VOT-2017 DATASET. THE RESULTS ARE PRESENTED IN TERMS OF EXPECTED AVERAGE OVERLAP (EAO), ACCURACY, AND ROBUSTNESS.

	ECO	CCOT	Staple	SiamFC	**GDT**
EAO	0.282	0.269	0.175	0.187	0.258
Accuracy	0.490	0.501	0.529	0.504	0.558
Robustness	0.753	0.717	0.511	0.547	0.645

TABLE IV
TRACKING ACCURACY (A) AND ROBUSTNESS (R) ON THE VOT-2017 DATASET. THE ATTRIBUTES TESTED INCLUDE CAMERA CHANGE (CAM.), ILLUMINATION CHANGE (ILLU.), MOTION CHANGE (MOT.), OCCLUSION (OCC.), SIZE CHANGE (SIZE), AND NOT ASSIGNED (N/A).

		Cam.	Illu.	Mot.	Occ.	Size	N/A	All
	ECO	0.513	0.519	0.481	0.339	0.446	0.497	0.490
	CCOT	0.516	0.478	0.488	0.391	0.451	0.513	0.501
A	Staple	0.554	0.483	0.508	0.431	0.491	0.534	0.529
	MDNet	0.513	0.510	0.498	0.379	0.475	0.523	0.504
	GDT	0.556	0.492	0.548	0.477	0.546	0.561	0.558
	ECO	0.726	0.390	0.621	0.364	0.767	0.840	0.753
	CCOT	0.717	0.244	0.587	0.347	0.663	0.789	0.717
R	Staple	0.452	0.308	0.469	0.273	0.607	0.585	0.511
	MDNet	0.599	0.308	0.308	0.215	0.479	0.697	0.547
	GDT	0.567	0.697	0.609	0.398	0.745	0.720	0.645

method does not perform better than ECO and CCOT on EAO and robustness, but ranks top on the accuracy, since the deformable convolution increases tracking accuracy but slightly reduces the robustness.

In Tab. IV, we analyze the performance of these trackers on different videos based on the accuracy and robustness metrics. Similar to the VOT-2016 results, our approach also performs better for most attributes. Due to the gating module, our approach performs well the in illumination change, occlusion, and size change attributes.

Tab. VI shows the results of the unsupervised experiment. The annotated attributes are the same as in VOT-2016. Our approach obtains the highest overall AO and outperforms the others in most attributes. As observed, our approach exhibits obvious advantages for the size change attribute, which is most related to deformable object tracking. In particular, GDT has a 30.2% margin over ECO on the Average Overlap metric.

TABLE V
THE AVERAGE OVERLAP (AO) FOR UNSUPERVISED EXPERIMENTS ON VOT-2016 DATASET. ATTRIBUTES INCLUDE CAMERA CHANGE (CAM.), ILLUMINATION CHANGE (ILLU.), MOTION CHANGE (MOT.), OCCLUSION (OCC.), SIZE CHANGE (SIZE), AND NOT ASSIGNED (N/A).

	Cam.	Illu.	Mot.	Occ.	Size	N/A	All
ECO	0.471	0.629	0.382	0.332	0.420	0.458	0.441
CCOT	0.484	0.594	0.397	0.328	0.461	0.510	0.470
Staple	0.419	0.559	0.355	0.235	0.388	0.408	0.390
MDNet	0.449	0.577	0.433	0.341	0.473	0.503	0.458
GDT	0.475	0.621	0.479	0.391	0.513	0.484	0.484

TABLE VI
THE AVERAGE OVERLAP (AO) FOR UNSUPERVISED EXPERIMENTS ON VOT-2017 DATASET. ATTRIBUTES INCLUDE CAMERA CHANGE (CAM.), ILLUMINATION CHANGE (ILLU.), MOTION CHANGE (MOT.), OCCLUSION (OCC.), SIZE CHANGE (SIZE), AND NOT ASSIGNED (N/A).

	Cam.	Illu.	Mot.	Occ.	Size	N/A	All
ECO	0.4190	0.420	0.362	0.261	0.370	0.419	0.403
CCOT	0.400	0.367	0.332	0.263	0.354	0.417	0.392
Staple	0.398	0.269	0.356	0.180	0.332	0.305	0.337
SiamFC	0.361	0.317	0.312	0.191	0.334	0.359	0.343
GDT	0.438	0.341	0.432	0.300	0.482	0.375	0.410

D. Qualitative Evaluation

Fig. 15 visually compare the results of the top-performing trackers: CCOT [14], MDNet [7], ECO [16] and our GDT on 12 challenging sequences. In the scenarios of *Diving*, *Jump*, *Skater2*, and *Trans*, which are close-up sequences from sports and movies, the objects exhibit severe pose variations. These results demonstrate the superiority of GDT qualitatively compared to the state-of-the-art trackers. For example, in *Diving*, CCOT loses track in the early stage and ECO fails to cover the body of the object. MDNet has a slight deviation at frame #91, while our tracker performs well in the entire sequence. Similarly in *Jump*, MDNet, CCOT, and ECO lose track and they cannot estimate the object size well. Since this sequence is very challenging, our tracker cannot estimate the object scale accurately either, but it still manages to track the person. In the sequence *Skater2* where two skaters frequently interact, the trackers tend to be distracted by the partner of the tracking target, while GDT always clings to the target. Both CCOT and ECO extract CNN features and learn correlation filters separately, and are not based on an end-to-end architecture. Hence, their features are not robust against deformation. MDNet also cannot track an object with severe deformation due to the lack of a mechanism to handle deformation. In the scenario *ClifBar* and *MotorRolling* where the object rotates, ECO and MDNet perform poorly. CCOT also drifts away in *MotorRolling*. There are some other examples (*Basketball*, *Girl2*, *Football*, *Freeman1*) with partial occlusions, in which MDNet does not handle well. With the aid of the deformable convolution, our tracker is better in adapting to the various object scales and tracking objects with rotations and partial occlusions.

V. CONCLUSION

We have proposed a deformable convolution layer to model target appearance variations in the CNN-based tracking-by-detection framework. We aim to capture target appearance variations via deformable convolution and supplement its normal convolution features through the online learned gating module. The gating module controls how the deformable convolutional features and the normal features are fused. Experimental results show that the proposed tracker performs favorably against the state-of-the-art methods.

There are still limitations in our proposed model. Our deformable convolution slightly degrades the robustness of the model, since its deformation estimation may fail in some extreme situations, including long-term occlusion, fast and large deformation, or significant illumination variation, as shown in Fig. 16. For the future work, we would like to improve the robustness of the deformable convolution by enhancing the features extraction stage of our framework. This can be accomplished by collecting more data or adopting a

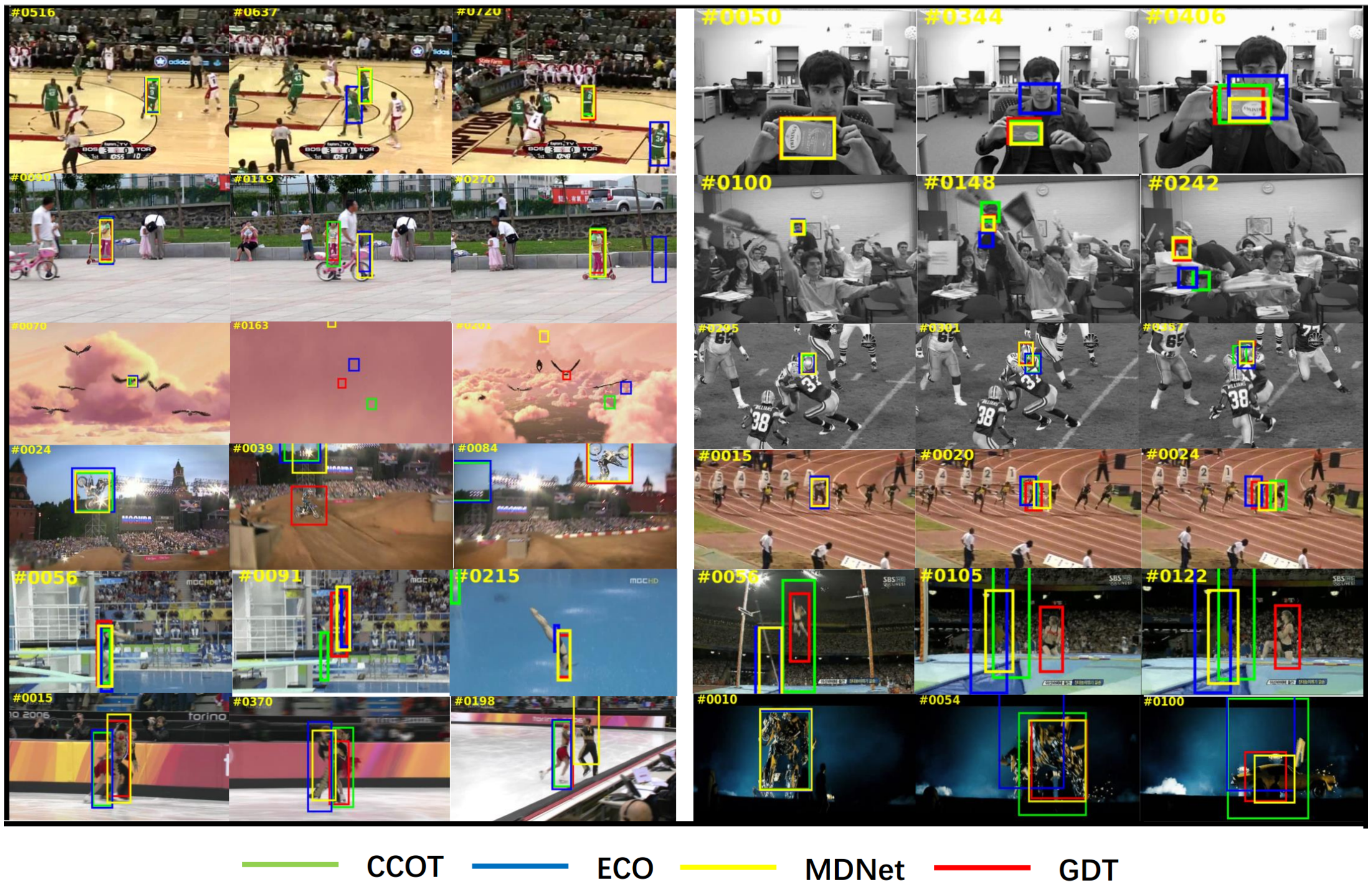
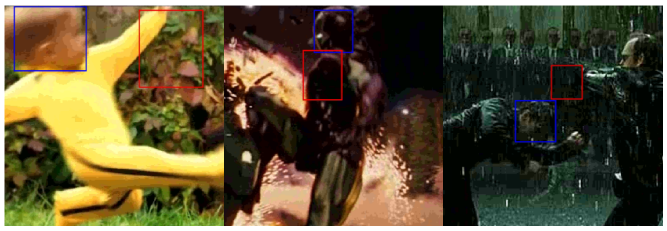


Fig. 15. Qualitative evaluation of GDT, CCOT [14], MDNet [41] and ECO [9] on 12 challenging sequences (from left to right and top to down: *Basketball*, *ClifBar*, *Girl2*, *Freeman1*, *Bird1*, *Football*, *MotorRolling*, *Bolt2*, *Diving*, *Jump*, *Skater2* and *Trans*) from OTB-2015. Our proposed tracker outperforms the state-of-the-art methods.



(a) *DragonBaby* (b) *Ironman* (c) *Matrix*

Fig. 16. Examples of failure cases. The ground-truth and the tracking bounding boxes are denoted in blue and red, respectively.

data augmentation technique (e.g., image warping) to independently train the deformable convolution module. In addition, the online learned gating module may not be adequately adaptive to difficult videos. To alleviate this problem, we aim to improve the gating module in the offline training stage. Moreover, in the future, we will consider to generalize our approach to different sources of image data, e.g., RGB-D data and medical images.

Acknowledgements

Our project is supported by the National Natural Science Foundation of China (Grant No.: 61702104, 61873067, and 61702194), the National Natural Science Foundation of Fujian Province (Grant No.: 2016J05155), and the Guangzhou Key Industrial Technology Research fund (No. 201802010036).

References

[1] Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking learning detection,” *TPAMI*.
[2] H. Grabner, M. Grabner, and H. Bischof, “Real-time tracking via on-line boosting.” in *BMVC*, 2006.
[3] B. Babenko, M.-H. Yang, and S. Belongie, “Visual tracking with online multiple instance learning,” in *CVPR*, 2009.
[4] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. L. Hicks, and P. H. Torr, “Struck: Structured output tracking with kernels,” *TPAMI*.
[5] S. Hong, T. You, S. Kwak, and B. Han, “Online tracking by learning discriminative saliency map with convolutional neural network,” in *ICML*, 2015.
[6] L. Zhang, J. Varadarajan, P. N. Suganthan, N. Ahuja, and P. Moulin, “Robust visual tracking using oblique random forests,” in *CVPR*, 2017.
[7] H. Nam and B. Han, “Learning multi-domain convolutional neural networks for visual tracking,” in *CVPR*, 2016.
[8] G. Ning, Z. Zhang, C. Huang, X. Ren, H. Wang, C. Cai, and Z. He, “Spatially supervised recurrent convolutional neural networks for visual object tracking,” in *IEEE International Symposium on Circuits and Systems*, 2017, pp. 1–4.
[9] Y. Song, C. Ma, X. Wu, B. L. Gong, Lijun, W. Zuo, C. Shen, R. W. H. Lau, and M.-H. Yang, “Vital: Visual tracking via adversarial learning,” in *CVPR*, 2018.
[10] S. Pu, Y. Song, C. Ma, H. Zhang, and M.-H. Yang, “Deep attentive tracking via reciprocative learning,” in *NIPS*, 2018.
[11] B. Han, J. Sim, and H. Adam, “Branchout: Regularization for online ensemble tracking with convolutional neural networks,” in *ICCV*, 2017.

[12] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *CVPR*, 2010.
[13] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *TPAMI*, 2015.
[14] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *BMVC*, 2014.
[15] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *CVPR*, 2015.
[16] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "Eco: efficient convolution operators for tracking," in *CVPR*, 2017.
[17] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *ECCV*, 2016.
[18] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *ICCV*, 2015.
[19] H. K. Galoogahi, T. Sim, and S. Lucey, "Correlation filters with limited boundaries," in *CVPR*, 2015.
[20] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Robust visual tracking via hierarchical convolutional features," in *TPAMI*, 2018.
[21] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang, "Hedged deep tracking," in *CVPR*, 2016.
[22] T. Zhang, C. Xu, and M.-H. Yang, "Multi-task correlation particle filter for robust object tracking," in *CVPR*, 2017.
[23] H. Kiani Galoogahi, A. Fagg, and S. Lucey, "Learning background-aware correlation filters for visual tracking," in *CVPR*, 2017.
[24] C. Sun, D. Wang, H. Lu, and M.-H. Yang, "Learning spatial-aware regressions for visual tracking," in *CVPR*, 2018.
[25] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *ICCV*, 2015.
[26] Y. Song, C. Ma, L. Gong, J. Zhang, R. W. Lau, and M.-H. Yang, "Crest: Convolutional residual learning for visual tracking," in *ICCV*, 2017.
[27] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr, "End-to-end representation learning for correlation filter based tracking," in *CVPR*, 2017.
[28] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *ECCV*. Springer, 2016, pp. 850–865.
[29] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *ECCV*, 2016.
[30] J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in *CVPR*, 2017.
[31] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang, "Learning dynamic siamese network for visual object tracking," in *ICCV*, 2017.
[32] B. Li, W. Wu, Z. Zhu, and J. Yan, "High performance visual tracking with siamese region proposal network," in *CVPR*, 2018.
[33] Q. Wang, Z. Teng, J. Xing, J. Gao, W. Hu, and S. Maybank, "Learning attentions: Residual attentional siamese network for high performance online visual tracking," in *CVPR*, 2018.
[34] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, and G. Ostrovski, "Human-level control through deep reinforcement learning," *Nature*, 2015.
[35] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Y. Choi, "Action-decision networks for visual tracking with deep reinforcement learning," in *CVPR*, 2017.
[36] X. Dong, J. Shen, W. Wang, Y. Liu, L. Shao, and F. Porikli, "Hyperparameter optimization for tracking with continuous deep q-learning," in *CVPR*, 2018.
[37] C. Huang, S. Lucey, and D. Ramanan, "Learning policies for adaptive tracking with deep feature cascades," in *ICCV*, 2017.
[38] C. Zhao, J. Wang, G. Zhu, Y. Wu, and H. Lu, "Learning weighted part models for object tracking," *Computer Vision and Image Understanding*, 2016.
[39] D. Du, H. Qi, W. Li, L. Wen, Q. Huang, and S. Lyu, "Online deformable object tracking based on structure-aware hyper-graph," *TIP*, 2016.
[40] X. Sun, N.-M. Cheung, H. Yao, and Y. Guo, "Non-rigid object tracking via deformable patches using shape-preserved kcf and level sets," in *CVPR*, 2017.
[41] A. Lukežič, L. Č. Zajc, and M. Kristan, "Deformable parts correlation filters for robust visual tracking," *IEEE Transactions on cybernetics*, 2017.
[42] Y. Zhong, A. K. Jain, and M.-P. Dubuisson-Jolly, "Object tracking using deformable templates," *TPAMI*, 2000.
[43] F. Leymarie and M. D. Levine, "Tracking deformable objects in the plane using an active contour model," *TPAMI*, 1993.
[44] J. Gao, T. Zhang, X. Yang, and C. Xu, "P2t: Part-to-target tracking via deep regression learning," *TIP*, 2018.
[45] C. Zhaowei, W. Longyin, L. Zhen, V. Nuno, and S. Z. Li, "Robust deformable and occluded object tracking with dynamic graph," *TIP*, 2014.
[46] W. Li, L. Wen, M. C. Chuah, Z. Yi, L. Zhen, and S. Z. Li, "Online visual tracking using temporally coherent part clusters," in *Applications of Computer Vision*, 2015.
[47] C. Luka, K. Matej, and L. Ales, "Robust visual tracking using an adaptive coupled-layer visual model," *TPAMI*.
[48] W. Shu, H. Lu, Y. Fan, and M. H. Yang, "Superpixel tracking," in *ICCV*, 2011.
[49] J. Xu, H. Lu, and M. H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *CVPR*, 2012.
[50] R. Yao, Q. Shi, C. Shen, Y. Zhang, and A. V. D. Hengel, "Part-based visual tracking with online latent structural learning," in *CVPR*, 2013.
[51] Z. Wei, H. Huchuan, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model," in *CVPR*, 2012.
[52] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," *ICCV*, 2017.
[53] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *NIPS*, 2015.
[54] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, 1997.
[55] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *BMVC*, 2014.
[56] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *CVPR*, 2013.
[57] ——, "Object tracking benchmark," *TPAMI*, 2015.
[58] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, F. Porikli, L. Cehovin, G. Nebehay, G. Fernandez, T. Vojir, A. Gatt *et al.*, "The visual object tracking vot2013 challenge results," in *ICCVW*, 2013.
[59] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin, "A novel performance evaluation methodology for single-target trackers," *TPAMI*, 2016.
[60] A. Vedaldi and K. Lenc, "MatConvNet: Convolutional neural networks for matlab," in *ACM International Conference on Multimedia*, 2015.
[61] L. Cehovin, A. Leonardis, and M. Kristan, "Visual object tracking performance measures revisited," *TIP*, 2016.
[62] H. Nam, M. Baek, and B. Han, "Modeling and propagating cnns in a tree structure for visual tracking," *arxiv*, 2016.
[63] G. Zhu, F. Porikli, and H. Li, "Beyond local search: Tracking objects everywhere with instance-specific proposals," in *CVPR*, 2016.
[64] J. Zhang, S. Ma, and S. Sclaroff, "Meem: robust tracking via multiple experts using entropy minimization," in *ECCV*, 2014.
[65] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao, "Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking," in *CVPR*, 2015.
[66] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *ICCVW*, 2015.
[67] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. Torr, "Staple: Complementary learners for real-time tracking," in *CVPR*, 2016.
[68] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking," in *CVPR*, 2016.
[69] S. Yoo, J. Yun, Y. Choi, K. Yun, and J. Y. Choi, "Action-decision networks for visual tracking with deep reinforcement learning," in *CVPR*, 2017.