Robust Tracking against Adversarial Attacks

Shuai Jia[1], Chao Ma[1]*, Yibing Song[2], and Xiaokang Yang[1]

[1] MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
{jiashuai,chaoma,xkyang}@sjtu.edu.cn
[2] Tencent AI Lab
yibingsong.cv@gmail.com

Abstract. While deep convolutional neural networks (CNNs) are vulnerable to adversarial attacks, considerably few efforts have been paid to construct robust deep tracking algorithms against adversarial attacks. Current studies on adversarial attack and defense mainly reside in a single image. In this work, we first attempt to generate adversarial examples on top of video sequences to improve the tracking robustness against adversarial attacks. To this end, we take temporal motion into consideration when generating lightweight perturbations over the estimated tracking results frame-by-frame. On one hand, we add the temporal perturbations into the original video sequences as adversarial examples to greatly degrade the tracking performance. On the other hand, we sequentially estimate the perturbations from input sequences and learn to eliminate their effect for performance restoration. We apply the proposed adversarial attack and defense approaches to state-of-the-art deep tracking algorithms. Extensive evaluations on the benchmark datasets demonstrate that our defense method not only eliminates the large performance drops caused by adversarial attacks, but also achieves additional performance gains when deep trackers are not under adversarial attacks. The source code is available at https://github.com/joshuajss/RTAA.

Keywords: Visual Tracking, Adversarial Attack

1 Introduction

Recent years have witnessed the success of CNNs for numerous computer vision tasks. Along with the success, the problem of attacking CNN models using adversarial examples emerges recently. That is, small perturbations on input images can lead the pretrained CNN models to complete failures. A number of adversarial attack methods inject perturbations into input images to degrade the performance of CNNs on a wide range of vision tasks, such as image classification [35], object detection [43], semantic segmentation [42], and face recognition [6]. In view of the vulnerability of CNNs, the defense approaches [33,44] aim to improve the robustness of CNNs against adversarial attacks. Despite the significant progress, current studies on adversarial attack and defense mainly

* Corresponding author.

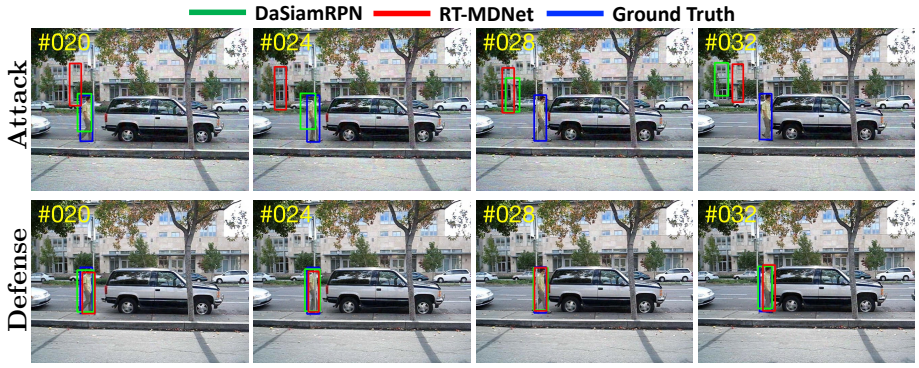


Fig. 1. Adversarial attack and defense for visual tracking. On top of the two state-of-the-art deep trackers DaSiamRPN [47] and RT-MDNet [13], we learn to generate adversarial examples to attack and defend them on the *David3* sequence [41].

rest in static images. Considerably less attention has been paid to generating adversarial examples on top of video sequences for robust deep tracking, where motion consistency between frames involves more challenges.

In this work, we start by investigating the vulnerability of the state-of-the-art deep trackers [27,18], which pose object tracking as a sequential detection problem to distinguish between the target and background. The CNN classifiers are often updated online with positive and negative examples, which are collected according to the previously estimated tracking results. In our investigation, we do not modify existing deep trackers and keep their sampling schemes unchanged. For adversarial attack, we learn perturbations and inject them into input frames, yielding indistinguishable binary samples (i.e., a portion of the samples are incorrect). We use these binary adversarial examples to retrain CNN classifiers to degrade their performance. Specifically, we minimize the classification loss difference between the correct and incorrect binary samples. When taking the temporal consistency between frames into consideration, we use the learned perturbations in the current frame to initialize the perturbation learning in the next frame. Applying the temporally generated perturbations for every frame further degrades the performance of deep trackers. In addition to the CNN classifiers, existing deep trackers widely use a regression network to refine bounding boxes. We first attempt to randomly shift and rescale ground truth boxes to attack the regression network. Note that attacking the bounding box locations significantly differs from existing adversarial attack approaches on object detection [43], where perturbations are generated via considering mis-classifications. Fig. 1 shows such an example that the state-of-the-art deep trackers under adversarial attacks drift rapidly (see the first row).

We step further to improve the robustness of deep trackers against adversarial attacks. Note that the adversarial perturbations are assumed to be unknown at this moment. Our motivation is to estimate the unknown perturbations in

the input videos and learn to eliminate their effects during tracking. The estimation process is similar to the attack but the involved samples are different. As an example shown in Fig. 1, we perform the proposed adversarial attack and defense approaches on two state-of-the-art deep tracking methods [47,13]. Extensive evaluations on large-scale benchmark datasets indicate the proposed defense approach is effective in improving the tracking robustness against adversarial attacks. When the trackers are not under adversarial attacks, the proposed defense scheme helps to achieve additional performance gains. This is because our defense is able to estimate the naturally existing adversarial perturbations during the image formation process.

The main contributions of this work are summarized as follows:

- We propose to generate adversarial examples to investigate the robustness of existing deep tracking algorithms. We inject dense adversarial perturbations into input video sequences in the spatiotemporal domain to degrade the tracking accuracy.
- We propose to defend deep trackers against adversarial attacks. We estimate the adversarial perturbations and eliminate their effect to alleviate performance drops caused by the adversarial attack.
- We perform adversarial attack and defense on state-of-the-art deep trackers. The extensive evaluations on the benchmark datasets demonstrate the effectiveness of both attack and defense. Our defense can further advance the state-of-the-art deep trackers not under adversarial attacks.

2 Related Works

2.1 Deep Visual Tracking

Existing object tracking approaches can be generally categorized as one-stage regression based methods and two-stage detection based methods. Deep learning advances both categories of tracking methods significantly. The regression based methods typically learn correlation filters over CNN features to locate target objects as in [23]. Since that, numerous methods are proposed to improve tracking performance in different aspects, including feature hedging [29], continuous convolution [5], particle filter integration [45], efficient convolution [4], spatiotemporal regularization [19], and roi pooling [34]. Meanwhile, there are end-to-end learnable regression networks aiming to directly predict response maps [37,11,36,31,21,38,39] for localizing the target object.

On the other hand, two-stage tracking-by-detection approaches first generate multiple proposals and then classify each as either the target or the background. The representative deep tracking-by-detection methods include multi-domain learning [27,13], ensemble learning [10], adversarial learning [32], reciprocative learning [28] and overlap maximization [3]. Recently, Siamese trackers [18,47,17,46,2,1,22] are prevalent due to their efficiency in online inference. The main difference between the Siamese trackers and other tracking-by-detection methods is that Siamese trackers typically do not online update CNN models

while other methods do. In this work, we deploy the proposed adversarial attack and defense schemes on two representative state-of-the-art trackers including one Siamese tracker [47] without online update and one detector based tracker [27] with online update. Our goal is to illustrate the general effectiveness of adversarial attack and defense on deep trackers with or without online update.

2.2 Adversarial Attacks and Defense

Recent studies [8,35] have shown that CNNs are vulnerable to adversarial examples. Despite the favorable performance on natural input images, the pretrained CNNs perform poorly given intentionally generated adversarial examples. Existing adversarial attack methods mainly fall into two categories: white-box and black-box attacks. The CNN models are assumed to be known in white-box attacks [8,25,24], whereas they are unknown in black-box attack [12,6]. In addition to algorithmic attacks, physical attack methods generate real world objects to lead CNNs models to misclassifications. These are typically useful to examine the robustness of automotive driving in the road sign scenarios [16,7,40].

Defending CNNs against adversarial attacks can be regarded as robustly learning CNNs with adversarial examples. Attempts have been made to formulate defense as a denoising problem. From this perspective, the adversarial examples produce noise on CNN features to distract the network inference process. In [20,44], denoising algorithms are proposed to eliminate the effect of noise. In addition, images are transformed to be non-differentiable in [9] to resist adversarial attacks. Different from existing attack and defense methods, we attack both the classification and regression modules of deep trackers to decrease accuracy. Then, we gradually estimate adversarial perturbations and eliminate their effect on the input images without modifying existing deep trackers.

3 Proposed Algorithm

This section illustrates how to perform adversarial attack and defense for visual tracking. Given an input video sequence and a labeled bounding box in the initial frame, we generate adversarial examples spatiotemporally to decrease tracking accuracy. On the other hand, our defense learns to estimate unknown adversarial perturbations and eliminate their effect from input sequences. We deploy the proposed attack and defense algorithms on the tracking-by-detection framework. The details are presented in the following.

3.1 Adversarial Example Generation

We generate adversarial perturbations based on the input frame and the output response of deep trackers, i.e., classification scores or regression maps. These perturbations are then added to the input frame for adversarial example generation. In the tracking-by-detection framework, deep trackers usually employ a CNN architecture containing two branches. The sampled proposals are classified

as either the target or background in the first branch, while the proposal axis is regressed in another branch for precise localization. A detailed illustration is referred to [18]. We denote an input frame by I, the proposal number by N, the binary classification loss by L_c, the bounding box regression loss by L_r, the correct classification label and regression label by p_c and p_r, respectively. Both labels p_c and p_r are predicted by the tracking results S^{t-1} from the last frame, while S^1 is the ground-truth annotation in the initial frame. The original loss function of the tracking-by-detection network can be written as:

$$\mathcal{L}(I, N, \theta) = \sum_{n=1}^{N} [L_c(I_n, p_c, \theta) + \lambda \cdot L_r(I_n, p_r, \theta)] \tag{1}$$

where I_n is one proposal in the image, λ is a fixed weight parameter, and θ denotes the CNN parameters to be optimized during the training process.

When generating adversarial perturbations, we expect CNNs to make inaccurate inference. We create a pseudo classification label $p_c^\star$ and a pseudo regression label $p_r^\star$. The adversarial loss is set to make L_c and L_r the same when we use correct and pseudo labels. The adversarial loss can be written as follows:

$$\begin{aligned} \mathcal{L}_{adv}(I, N, \theta) = \sum_{n=1}^{N} \{ & [L_c(I_n, p_c, \theta) - L_c(I_n, p_c^\star, \theta)] \\ & + \lambda \cdot [L_r(I_n, p_r, \theta) - L_r(I_n, p_r^\star, \theta)] \} \end{aligned} \tag{2}$$

where θ is fixed because the CNN is in the inference stage. The adversarial loss $\mathcal{L}_{adv}$ reflects the loss similarity between using correct and pseudo labels. When minimizing $\mathcal{L}_{adv}$, the CNN predictions will be close to pseudo labels and the performance will degrade rapidly.

We set pseudo labels specifically for each branch. In $p_c^\star$, there are two elements (i.e., 0 and 1) which indicate the probabilities of the input belonging to the target and background. We set $p_c^\star$ by reversing the elements of p_c to confuse the classification branch. On the other hand, p_r consists of four elements (x_r, y_r, w_r, h_r) representing the target location. We set $p_r^\star$ by adding a random distance offset and a random scale variation to p_r. Each element of $p_r^\star$ can be written as:

$$\begin{aligned} x_r^\star &= x_r + \delta_{\text{offset}} \\ y_r^\star &= y_r + \delta_{\text{offset}} \\ w_r^\star &= w_r * \delta_{\text{scale}} \\ h_r^\star &= h_r * \delta_{\text{scale}} \end{aligned} \tag{3}$$

where δ_{offset} and δ_{scale} indicate the random distance offset and random scale variation, respectively.

After computing the adversarial loss in Eq. 2, we take partial derivatives of the adversarial loss with respect to the input I. Formally, the partial derivative r is computed as:

$$r = \frac{\partial \mathcal{L}_{adv}}{\partial I}. \tag{4}$$

Algorithm 1: Adversarial Example Generation

Input: input video V with T frames; target location S^1;
Output: adversarial examples of T frames;

```
for t = 2 to T do
    Get current frame I_1^t;
    if t ≠ 2 then
        Update I_1^t via Eq. 6;
    end
    for m = 1 to M do
        Create p_c and p_r via IoU ratios between proposals and target
         location S^{t-1};
        Create p_c^⋆ by reversing elements of p_c;
        Create p_r^⋆ via Eq. 3;
        Generate adversarial loss via Eq. 2;
        Update I_m^t via Eq. 5;
    end
    return I_M^t;
end
```

We pass r into a sign function to reduce outlier effects. Given an input frame I, we take M iterations to produce the final adversarial perturbations. The output of the last iteration is added into the input frame, which can be written as follows:

$$I_{m+1} = I_m + \alpha \cdot sign(r_m) \tag{5}$$

where $\alpha = \frac{\epsilon}{M}$ is a constant weight, ϵ is the maximum value of the perturbations, m indicates the iteration index, I_m is the input frame for the m-th iteration, $\alpha \cdot sign(r_m)$ is the perturbations generated during the m-th iteration. The final adversarial example is I_M.

As video frames are temporally coherent, we consider the adversarial attacks in the spatiotemporal domain. When there are T frames in an input video sequence, we use the learned perturbations in the last frame as initialization for the current frame. Specifically, for the t-th frame, we use perturbations from the last frame to initialize I^t, which can be written as:

$$I_1^t = I_1^t + (I_M^{t-1} - I_1^{t-1}) \tag{6}$$

where $I_M^{t-1} - I_1^{t-1}$ is the perturbation from the last frame. Then we gradually update I^t by using Eq. 5 to generate the final perturbations for the t-th frame. The pseudo code is shown in Algorithm 1. Note that we use the IoU metric [30] to assign proposal labels.

3.2 Adversarial Defense

We propose an adversarial defense method to recover tracking accuracy that is deteriorated via adversarial attacks. Our motivation comes from the adversarial

Algorithm 2: Adversarial Example Defense

Input: input video V with T adversarial examples; target location S^1;
Output: adversarial examples of T frames;

```
for t = 2 to T do
    Get current frame I_1^t;
    if t ≠ 2 then
        Update I_1^t via Eq. 8;
    end
    for m = 1 to M do
        Create p_c and p_r via IoU ratios between proposals and target
         location S^{t-1};
        Create p_c^⋆ by reversing elements of p_c;
        Create p_r^⋆ via Eq. 3;
        Generate adversarial loss via Eq. 2;
        Update I_m^t via Eq. 7;
    end
    return I_M^t;
end
```

perturbations which are accumulated during the iterations in each frame. From Eq. 4, we observe that perturbations originate from partial derivatives. Instead of adding perturbations to the input frame to decrease tracking accuracy, we estimate the perturbations and subtract them from the input frame gradually. As a result, the effect of perturbations will be eliminated to help CNNs to make the correct inference. We defend adversarial examples without updating CNNs.

Given an input frame I with unknown adversarial perturbations, we first generate correct and pseudo labels according to the predicted location S^{t-1} from the previous frame. The label generation process is similar to that in Sec. 3.1 except that the utilized proposals during defense are resampled. Then, we estimate the adversarial loss by using Eq. 2 and compute the partial derivatives via Eq. 4. We apply partial derivatives r on the input frame I via the following operation:

$$I_{m+1} = I_m - \underset{\beta \cdot r \in [-\hat{\alpha}, \hat{\alpha}]}{\text{Trunc}}(\beta \cdot r) \tag{7}$$

where β is a constant weight, $\text{Trunc}(\cdot)$ is a truncation function to constrain the values of $\beta \cdot r$ within the range between $-\hat{\alpha}$ and $\hat{\alpha}$. The parameter $\hat{\alpha}$ resembles the parameter α in Eq. 5. Since the perturbation is unknown during defense, we empirically set different values for these two parameters. When the input videos contain T frames, we still transfer the perturbations from the last frame to the current frame as initialization. For the t-th frame, we update it initially as:

$$I_1^t = I_1^t - \gamma \cdot (I_1^{t-1} - I_M^{t-1}) \tag{8}$$

where γ is a constant weight. The pseudo code of adversarial defense is shown in Algorithm 2.

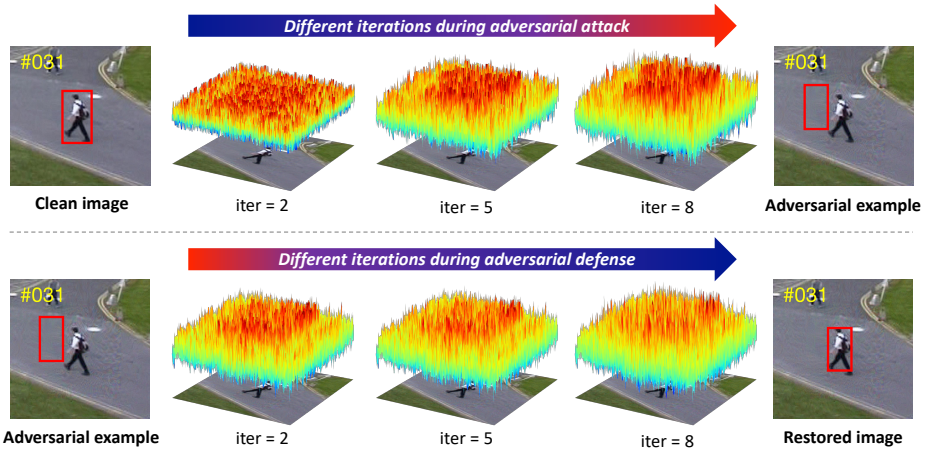


Fig. 2. Variations of adversarial perturbations during attack and defense. The 3D response map above the image represents the difference between the clean image and the adversarial example at the current iteration. In adversarial attack, the perturbations increase along with training iterations. In adversarial defense, the perturbations decrease when training iteration increases.

3.3 Deployment of Deep Trackers

From the perspective of online update, existing tracking-by-detection methods involves two main stages. In the first stage, trackers do not update online while utilizing an offline pretrained CNN model. In the second stage, trackers collect samples online from previous frames to update the model. Note that trackers with online update tend to improve model adaption by collecting samples incrementally, which may help defend adversarial perturbations. We deploy the proposed adversarial attack and defense on two state-of-the-art trackers, DaSiamRPN [47] and RT-MDNet [13]. Details are presented in the following:

DaSiamRPN. There are two output branches in DaSiamRPN to classify and regress proposals. During tracking, DaSiamRPN does not perform online update. When processing each frame, we follow Algorithm 1 and Algorithm 2 to generate and defend adversarial examples, respectively. As the inputs contain a template and search patch, we take partial derivatives with respect to the search patch when computing Eq. 4 for both attack and defense.

RT-MDNet. The CNN model of RT-MDNet only performs classification. During tracking, RT-MDNet online updates its model by collecting samples from previous frames. When processing each frame, we first generate adversarial examples and then perform prediction and model update. This configuration aims to analyze whether online update is effective to defend adversarial examples. We generate and defend adversarial examples using Algorithm 1 and Algorithm 2, except that we remove the regression terms when computing Eq. 2.

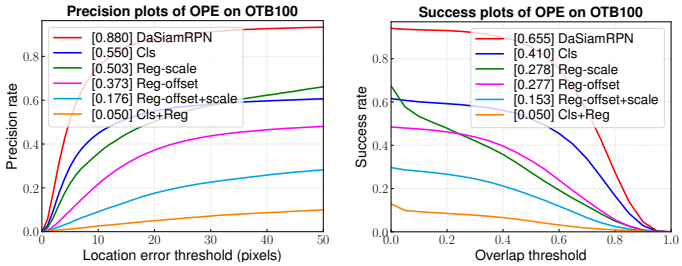


Fig. 3. Ablation studies of DaSiamRPN on the OTB100 dataset [41]. We denote Cls as the attack on the classification branch, Reg as the attack on the regression branch where there are offset and scale attacks.

Visualizations. We show how adversarial perturbations vary during different training iterations in Fig. 2. DaSiamRPN is our baseline tracker to be attacked and defended. Given an input frame, we visualize the perturbations during adversarial attack. Along with the training iterations, the variation of perturbations increases as well. The adversarial examples lead DaSiamRPN to drift rapidly. When defending this adversarial example, we observe that the variation of the perturbations decreases when training iteration increases. This indicates that our defense method is effective to estimate and exclude the adversarial perturbations, which helps to alleviate performance drops caused by adversarial attack.

4 Experiments

In this section, we evaluate the proposed attack and defense methods on benchmark datasets. We develop our methods on top of DaSiamRPN and RT-MDNet for all the experiments. The maximum variation value of each pixel in the perturbations is set to 10 (i.e., $\epsilon = 10$) for attack and set to 5 for (i.e., $\epsilon = 5$) defense, respectively. When computing IoU ratios between proposals and the target location S_{t-1}, we follow the threshold setting of DaSiamRPN and RT-MDNet to draw training samples. The parameters of deep trackers are kept fixed during both adversarial attack and defense.

4.1 Ablation Studies

We analyze the effectiveness of each component of the proposed method. We take the DaSiamRPN tracker as a baseline as it contains both the classification and regression branches. We first evaluate the baseline performance on the OTB100 dataset. Then, we separately apply our attack method on the classification and regression branches. When we attack the regression branch, we analyze the offset and scale variation effects. Meanwhile, we combine both classification and regression attacks. Fig. 3 shows the evaluation results where the attack on the

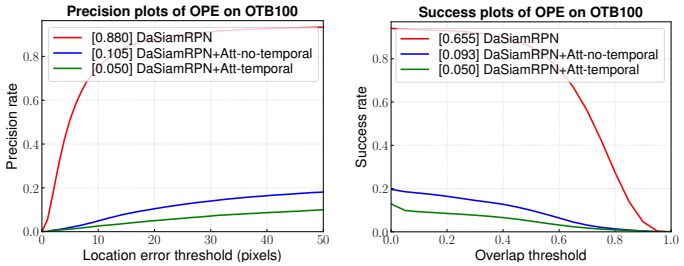


Fig. 4. Temporal consistency validation of DaSiamRPN on the OTB100 dataset [41].

regression branch degrades the performance more than the attack on the classification branch. Combining attacks on both branches leads to more degraded tracking performance.

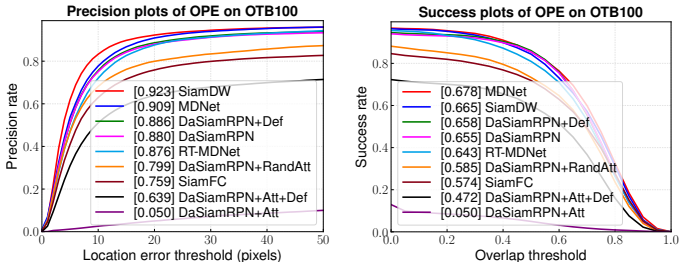
In addition to analyzing the attack in single frames, we show performance degradation via temporally consistent attack. We transfer the perturbation of the last frame to the current frame as initialization. Fig. 4 shows that our temporally consistent attack decreases tracking accuracies more than the static image attack.

4.2 Benchmark Performance

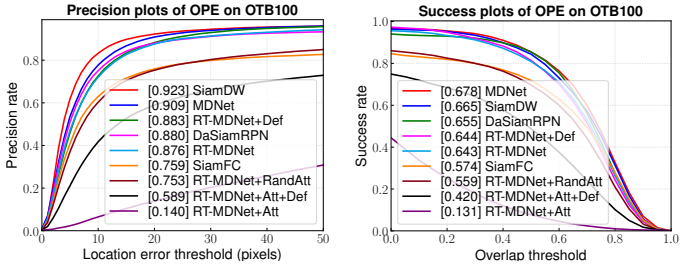
We deploy the proposed attack and defense methods on DaSiamRPN and RT-MDNet on four standard benchmark datasets (OTB100 [41], VOT-2016 [15], VOT-2018 [14] and UAV123 [26]). The results are presented as follows:

OTB100. There are 100 video sequences in this dataset with substantial target variations. We apply the one-pass evaluation (OPE) with success and precision plots. The precision metric evaluates the ratio of frames whose center location error is within a certain threshold among all frames. The success plot measures the IoU scores between the tracking results and ground truth at different thresholds. The area under the curve (AUC) success scores are also reported.

In order to evaluate our adversarial attack and defense methods, we first show the results of the baseline trackers. Second, we attack the baseline trackers by adding adversarial perturbations to input video sequences. In addition, we show the attack performance by adding random perturbations containing the same variations to those of adversarial perturbations. Fig. 5 shows that our attack method reduces the AUC scores of DaSiamRPN from 0.655 to 0.050, and the AUC scores of RT-MDnet from 0.643 to 0.131. The baseline trackers under adversarial attacks perform much worse than that under random perturbations. This indicates the effectiveness of our adversarial attack method on baseline trackers. On the other hand, our defense method is able to suppress the maximal value of adversarial perturbations to restore the tracking performance. It recovers the AUC score of DaSiamRPN to 0.721 and RT-MDNet to 0.653. This



(a) DaSiamRPN



(b) RT-MDNet

Fig. 5. Tracking performance of the adversarial attack and defense methods on the OTB100 dataset. We denote Att and Def as the adversarial attack and defense, respectively, and denote Rand as random perturbations.

demonstrates that our defense method can effectively remove the perturbations to restore tracking performance. In addition, we apply our defense method on original tracking sequences and improves AUC score from 0.880 to 0.886 for DaSiamRPN and from 0.876 to 0.883 for RT-MDNet. This indicates that perturbations (e.g., noise) exist in real world scenarios during image formation process (e.g., camera sensor noise, transformation from optical perception to digital storage). Our defense method is effective to estimate these naturally existing perturbations and eliminate their effects.

VOT-2018. There are 60 sequences on the VOT-2018 dataset. The VOT toolkit will reinitialize the tracker if it loses the target object during 5 consecutive frames. The evaluation metrics of VOT are expected average overlap (EAO), accuracy (Acc) and robustness (Rob). The accuracy represents the average overlap ratio and the robustness is measured by the number of reinitialization. EAO measures the overall performance of trackers.

Table 1 and Table 2 show the experimental results of DaSiamRPN and RT-MDnet. The failure number increases rapidly after attacking the original sequences. The EAO drops dramatically from 0.380 to 0.097 (i.e., a 74.5% decrease) for DaSiamRPN and from 0.176 to 0.076 (i.e., a 56.8% decrease) for RT-MDNet. By integrating our adversarial defense method, the EAO scores of DaSiamRPN and RT-MDNet are restored to 0.195 and 0.110, respectively. The performance

Table 1. Attack and defense on DaSiamRPN on VOT-2018 and VOT-2016 datasets.

	VOT-2018			VOT-2016		
	Acc ↑	Rob ↓	EAO ↑	Acc ↑	Rob ↓	EAO ↑
DaSiamRPN	0.585	0.272	0.380	0.625	0.224	0.439
DaSiamRPN+RandAtt	0.571	0.529	0.223	0.606	0.303	0.336
DaSiamRPN+Att	0.536	1.447	0.097	0.521	1.613	0.078
DaSiamRPN+Att+Def	0.579	0.674	0.195	0.581	0.722	0.211
DaSiamRPN+Def	0.584	**0.253**	**0.384**	0.622	**0.214**	0.418

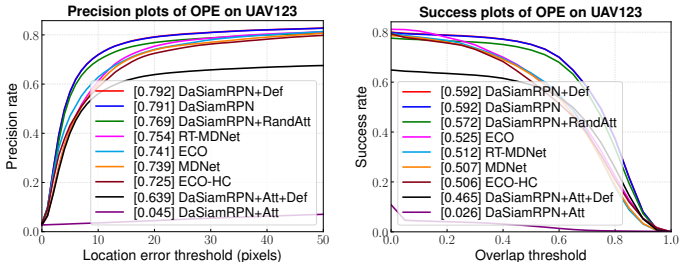
Table 2. Attack and defense on RT-MDNet on VOT-2018 and VOT-2016 datasets.

	VOT-2018			VOT-2016		
	Acc ↑	Rob ↓	EAO ↑	Acc ↑	Rob ↓	EAO ↑
RT-MDNet	0.533	0.567	0.176	0.567	0.196	0.370
RT-MDNet+RandAtt	0.503	0.871	0.137	0.550	0.452	0.235
RT-MDNet+Att	0.475	1.611	0.076	0.469	0.928	0.128
RT-MDNet+Att+Def	0.515	1.021	0.110	0.531	0.494	0.225
RT-MDNet+Def	0.529	**0.538**	**0.179**	0.540	**0.168**	**0.374**

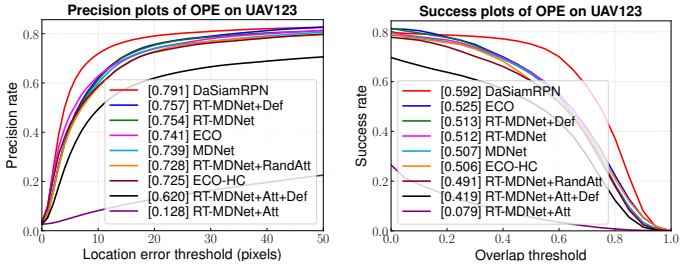
decrease and restoration indicate the effectiveness of our adversarial attack and defense method. In addition, our defense method also improves the performance slightly when the baseline trackers are not under adversarial attacks.

VOT-2016. The VOT-2016 dataset consists of 60 sequences. The evaluation metrics are the same as those used in the VOT-2018 dataset. Table 1 and Table 2 illustrate the performance of our attack and defense methods on DaSiamRPN and RT-MDNet. Our adversarial attack algorithm reduces the EAO scores by 82.2% for DaSiamRPN and 65.4% for RT-MDNet. With the use of the defense method on the adversarial examples, the accuracy of EAO is restored by 48.1% for DaSiamRPN and 60.8% for RT-MDNet. It indicates the effectiveness of both adversarial attack and defense methods. In addition, when applying our defense method to the attack-free deep trackers, the robustness gets largely improved. Due to the reinitialization scheme in VOT, we observe that EAO and robustness values decrease dramatically during attacks but accuracy value does not vary much.

UAV123. The UAV123 dataset contains 123 sequences with more than 110K frames, which are captured from low-altitude unmanned aerial vehicles. We adopt the success and precision plots to evaluate the performance. Fig. 6 illustrates the precision and success plots of DaSiamRPN and RT-MDNet. Under attacks, the AUC scores drop by 95.6% for DaSiamRPN and 84.6% for



(a) DaSiamRPN



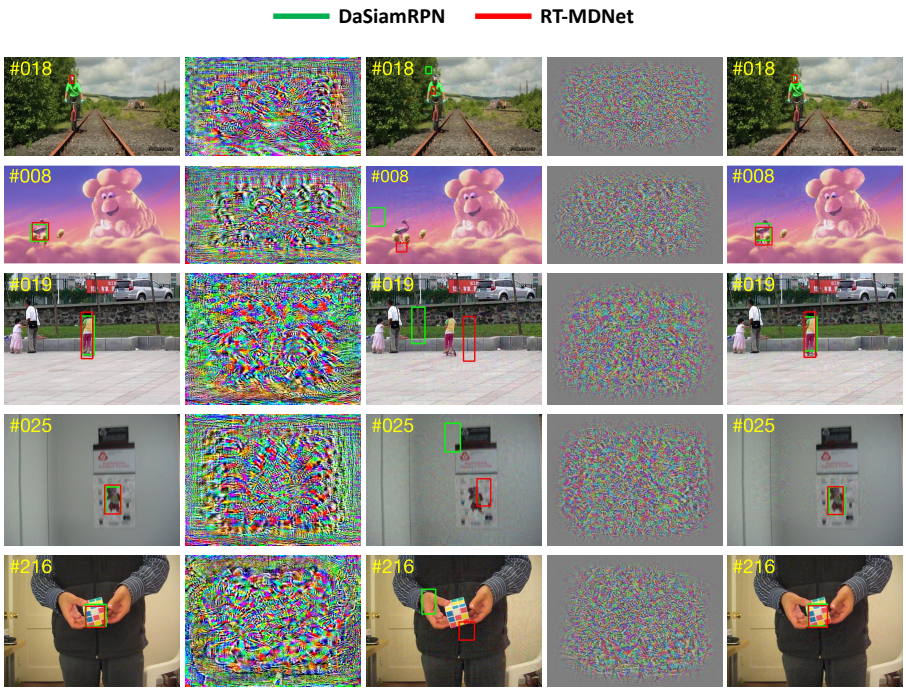
(b) RT-MDNet

Fig. 6. Results of the adversarial attack and defense methods on UAV123 dataset.

RT-MDNet. The precision rate at 20 pixels is reduced by 94.3% and 83.0% for DaSiamRPN and RT-MDNet. The AUC scores are restored by 80.8% and 82.2% respectively after defending the adversarial examples since the target objects on the UAV123 dataset mostly undergo large shape changes.

4.3 Qualitative Evaluations

Fig. 7 qualitatively shows the tracking results of our adversarial attack and defense methods for DaSiamPRN [47] and RT-MDNet [13] on 5 challenging sequences. We visualize the attack and defense perturbations in (b) and (d). In the original sequences shown in (a), both DaSiamRPN and RT-MDNet locate the target objects and estimate the scale changes accurately. After injecting adversarial perturbations in (b), DaSiamRPN fails to track the target and RT-MDNet estimates the target scale inaccurately as shown in (c). The tracking accuracy of RT-MDNet does not degrade severely compared to DaSiamRPN because RT-MDNet only contains the classification branch while DaSiamRPN contains both classification and regression branches. When we perform defense, the adversarial perturbations are estimated and shown in (d). By subtracting the estimated perturbations from the adversarial examples, DaSiamRPN and RT-MDNet are able to locate target objects correctly in the video sequences shown in (e).



(a) Input frames (b) Adversarial perturbations (c) Attack results (d) Estimated perturbations (e) Defense results

Fig. 7. Qualitative evaluation of adversarial attack and defense on video sequences.

5 Concluding Remarks

In this paper, we propose the adversarial attack and defense methods by generating lightweight perturbations within the deep tracking framework. When generating adversarial examples, we integrate the temporal perturbations into frames by perplexing trackers with indistinguishable correct and incorrect inferences. When defending adversarial examples, we suppress the maximum of adversarial perturbation to restore the tracking accuracy. Extensive experiments on four standard benchmarks demonstrate that the proposed methods perform favorably both in adversarial attack and defense. In addition, our defense method is capable of reducing interference from perturbations in the real world scenarios to robustify deep trackers.

Acknowledgements. This work was supported by National Key Research and Development Program of China (2016YFB1001003), NSFC (U19B2035, 61527804, 60906119), STCSM (18DZ1112300). C. Ma was sponsored by Shanghai Pujiang Program.

References

1. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. In: ECCV Workshop (2016)
2. Bhat, G., Danelljan, M., Van Gool, L., Timofte, R.: Learning discriminative model prediction for tracking. In: ICCV (2019)
3. Danelljan, M., Bhat, G., Khan, F.S., Felsberg, M.: Atom: Accurate tracking by overlap maximization. In: CVPR (2019)
4. Danelljan, M., Bhat, G., Shahbaz Khan, F., Felsberg, M.: ECO: Efficient convolution operators for tracking. In: CVPR (2017)
5. Danelljan, M., Robinson, A., Khan, F.S., Felsberg, M.: Beyond correlation filters: Learning continuous convolution operators for visual tracking. In: ECCV (2016)
6. Dong, Y., Su, H., Wu, B., Li, Z., Liu, W., Zhang, T., Zhu, J.: Efficient decision-based black-box adversarial attacks on face recognition. In: CVPR (2019)
7. Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., Song, D.: Robust physical-world attacks on deep learning models. In: CVPR (2018)
8. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: ICLR (2015)
9. Guo, C., Rana, M., Cisse, M., Van Der Maaten, L.: Countering adversarial images using input transformations. In: ICLR (2018)
10. Han, B., Sim, J., Adam, H.: Branchout: Regularization for online ensemble tracking with convolutional neural networks. In: CVPR (2017)
11. Held, D., Thrun, S., Savarese, S.: Learning to track at 100 fps with deep regression networks. In: ECCV (2016)
12. Ilyas, A., Engstrom, L., Athalye, A., Lin, J.: Black-box adversarial attacks with limited queries and information. arXiv preprint: 1804.08598 (2018)
13. Jung, I., Son, J., Baek, M., Han, B.: Real-time mdnet. In: ECCV (2018)
14. Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Cehovin Zajc, L., Vojir, T., Bhat, G., Lukezic, A., Eldesokey, A., et al.: The sixth visual object tracking vot2018 challenge results. In: ECCV Workshop (2018)
15. Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Cehovin Zajc, L., Vojir, T., Hager, G., Lukezic, A., Eldesokey, A., et al.: The visual object tracking vot2016 challenge results. In: ECCV Workshop (2016)
16. Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial examples in the physical world. In: ICLR (2017)
17. Li, B., Wu, W., Wang, Q., Zhang, F., Xing, J., Yan, J.: Siamrpn++: Evolution of siamese visual tracking with very deep networks. In: CVPR (2019)
18. Li, B., Yan, J., Wu, W., Zhu, Z., Hu, X.: High performance visual tracking with siamese region proposal network. In: CVPR (2018)
19. Li, F., Tian, C., Zuo, W., Zhang, L., Yang, M.H.: Learning spatial-temporal regularized correlation filters for visual tracking. In: CVPR (2018)
20. Liao, F., Liang, M., Dong, Y., Pang, T., Hu, X., Zhu, J.: Defense against adversarial attacks using high-level representation guided denoiser. In: CVPR (2018)
21. Lu, X., Ma, C., Ni, B., Yang, X., Reid, I., Yang, M.H.: Deep regression tracking with shrinkage loss. In: ECCV (2018)
22. Lu, X., Wang, W., Ma, C., Shen, J., Shao, L., Porikli, F.: See more, know more: Unsupervised video object segmentation with co-attention siamese networks. In: CVPR (2019)

23. Ma, C., Huang, J.B., Yang, X., Yang, M.H.: Hierarchical convolutional features for visual tracking. In: ICCV (2015)
24. Moosavi-Dezfooli, S.M., Fawzi, A., Fawzi, O., Frossard, P.: Universal adversarial perturbations. In: CVPR (2017)
25. Moosavi-Dezfooli, S.M., Fawzi, A., Frossard, P.: Deepfool: a simple and accurate method to fool deep neural networks. In: CVPR (2016)
26. Mueller, M., Smith, N., Ghanem, B.: A benchmark and simulator for uav tracking. In: ECCV (2016)
27. Nam, H., Han, B.: Learning multi-domain convolutional neural networks for visual tracking. In: CVPR (2016)
28. Pu, S., Song, Y., Ma, C., Zhang, H., Yang, M.H.: Deep attentive tracking via reciprocative learning. In: NeurIPS (2018)
29. Qi, Y., Zhang, S., Qin, L., Yao, H., Huang, Q., Lim, J., Yang, M.H.: Hedged deep tracking. In: CVPR (2016)
30. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. TPAMI (2016)
31. Song, Y., Ma, C., Gong, L., Zhang, J., Lau, R.W., Yang, M.H.: CREST: Convolutional residual learning for visual tracking. In: ICCV (2017)
32. Song, Y., Ma, C., Wu, X., Gong, L., Bao, L., Zuo, W., Shen, C., Lau, R.W., Yang, M.H.: VITAL: Visual tracking via adversarial learning. In: CVPR (2018)
33. Sun, B., Tsai, N.h., Liu, F., Yu, R., Su, H.: Adversarial defense by stratified convolutional sparse coding. In: CVPR (2019)
34. Sun, Y., Sun, C., Wang, D., He, Y., Lu, H.: Roi pooled correlation filters for visual tracking. In: CVPR (2019)
35. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. In: ICLR (2014)
36. Valmadre, J., Bertinetto, L., Henriques, J., Vedaldi, A., Torr, P.H.: End-to-end representation learning for correlation filter based tracking. In: CVPR (2017)
37. Wang, L., Ouyang, W., Wang, X., Lu, H.: Visual tracking with fully convolutional networks. In: ICCV (2015)
38. Wang, N., Song, Y., Ma, C., Zhou, W., Liu, W., Li, H.: Unsupervised deep tracking. In: CVPR (2019)
39. Wang, N., Zhou, W., Song, Y., Ma, C., Liu, W., Li, H.: Unsupervised deep representation learning for real-time tracking. IJCV (2020)
40. Wiyatno, R.R., Xu, A.: Physical adversarial textures that fool visual object tracking. In: ICCV (2019)
41. Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. TPAMI (2015)
42. Xiao, C., Deng, R., Li, B., Yu, F., Liu, M., Song, D.: Characterizing adversarial examples based on spatial consistency information for semantic segmentation. In: ECCV (2018)
43. Xie, C., Wang, J., Zhang, Z., Zhou, Y., Xie, L., Yuille, A.: Adversarial examples for semantic segmentation and object detection. In: ICCV (2017)
44. Xie, C., Wu, Y., Maaten, L.v.d., Yuille, A.L., He, K.: Feature denoising for improving adversarial robustness. In: CVPR (2019)
45. Zhang, T., Xu, C., Yang, M.H.: Multi-task correlation particle filter for robust object tracking. In: CVPR (2017)
46. Zhang, Z., Peng, H.: Deeper and wider siamese networks for real-time visual tracking. In: CVPR (2019)
47. Zhu, Z., Wang, Q., Li, B., Wu, W., Yan, J., Hu, W.: Distractor-aware siamese networks for visual object tracking. In: ECCV (2018)