Efficient Adversarial Attacks for Visual Object Tracking

Siyuan Liang[1,2], Xingxing Wei[4*], Siyuan Yao[1,2], and Xiaochun Cao[1,2,3*]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{liangsiyuan, yaosiyuan, caoxiaochun}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen 518055, China
[4] Beijing Key Laboratory of Digital Media, School of Computer Science and Engineering, Beihang University, Beijing 100191, China
{xxwei@buaa.edu.cn}

Abstract. Visual object tracking is an important task that requires the tracker to find the objects quickly and accurately. The existing state-of-the-art object trackers, i.e., Siamese based trackers, use DNNs to attain high accuracy. However, the robustness of visual tracking models is seldom explored. In this paper, we analyze the weakness of object trackers based on the Siamese network and then extend adversarial examples to visual object tracking. We present an end-to-end network FAN (Fast Attack Network) that uses a novel drift loss combined with the embedded feature loss to attack the Siamese network based trackers. Under a single GPU, FAN is efficient in the training speed and has a strong attack performance. The FAN can generate an adversarial example at 10ms, achieve effective targeted attack (at least 40% drop rate on OTB) and untargeted attack (at least 70% drop rate on OTB).

Keywords: Adversarial Attack, Visual Object Tracking, Deep Learning

1 Introduction

Some studies have shown that DNN-based models are very sensitive to adversarial examples [11]. In general, most recent methods for generating adversarial examples rely on the network structure and their parameters, and they utilize the gradient to generate adversarial examples by iterative optimization [28] [3]. Adversarial examples have successfully attacked deep learning tasks such as image classification [23], object detection [29], and semantic segmentation [10]. Researching adversarial examples can not only help people understand the principles of DNNs [30] but also improve the robustness of networks in visual tasks [14].

Visual Object Tracking (VOT) [18] aims to predict the object's locations in

* Corresponding Author

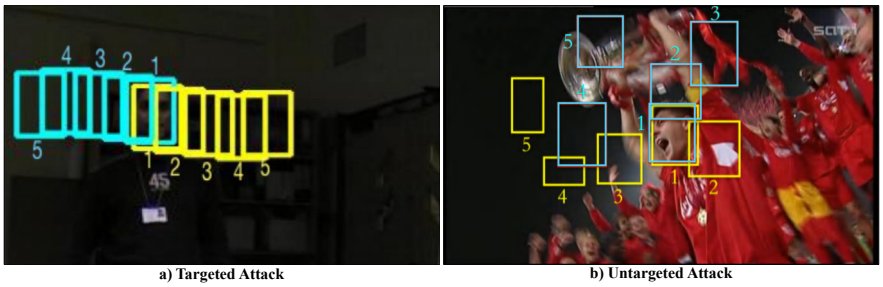

Fig. 1. Two examples of adversarial attacks for VOT. To better show the attacking results, we plot the bounding boxes in the initial frame. The numbers represent the results in the corresponding video frames. The blue box represents the predicted bounding box, and the yellow box represents the ground truth.

the subsequent frames in a video when given an objects location in the initial frame. In recent years, deep learning [4] based trackers have achieved excellent performance in many benchmarks. Among them, the SiamFC tracker [1] explores the similarity between video frames by using powerful deep features and has achieved great results in accuracy and robustness for the tracking task. Similar to the Faster R-CNN architecture [25] in object detection, the latest visual object tracking methods are based on the Siamese network, and many variants have been derived, such as SiamVGG [34], SiamRPN [20], SiamRPN++ [19] and so on. Therefore, the significance of investigating the robustness of trackers based on deep learning becomes quite crucial.

In different visual tasks, the attacking targets are different. In image classification task, the target is the classification problem. In the object detection task, the target is the regression (for SSD and YOLO) or classification problem (for Faster-RCNN). In object tracking, VOT searches the most similar regions in each frame with the reference patch. Therefore the target is essentially the similarity metric problem. Thus, attacking the tracking task is totally different from the other image recognition tasks, and the existing attacking methods cannot work well (the results in Section 4.4 verify this point).

Regarding the above motivation, in this paper, we study the adversarial attacks on Visual Object Tracking (VOT). **Firstly**, because the adversarial attack on VOT is seldom explored, we give a definition of the targeted attack and untargeted attack in the visual object tracking task. **Then**, we propose an end-to-end fast attack network (FAN) that combines the drift loss and embedded feature loss to jointly perform the targeted and untargeted attacks on VOT. Under the hardware condition of a single GPU, we only need 3 hours off-line training on the ILSVRC15 dataset. In the inference phase, the generator can generate adversarial perturbations in milliseconds speed for the OTB dataset [32] and the VOT dataset [16]. Figure 1 gives two examples. Targeted attack causes the tracker to track object along any specified trajectory. Untargeted attack makes the tracker

unable to keep track of the object. Overall, **our contributions** can be summarized as follows: (1) To the best of our knowledge, we are the first one to perform the targeted attack and untargeted attack against the Visual Object Tracking (VOT) task. We analyze the weakness of the trackers based on the Siamese network, and then give a definition of the targeted attack and untargeted attack in this task. (2) We propose a unified and end-to-end attacking method: FAN (fast attack network). We design a novel drift loss to achieve the untargeted attack effectively and apply the embedded feature loss to accomplish the targeted attack. Finally, we combine these two loss functions to jointly attack the VOT task. (3) After three hours of training, FAN can successfully attack VOT and OTB datasets without fine-tuning network parameters. In inference, FAN can quickly produce adversarial examples within 10ms, which is much faster than iterative optimization algorithms.

2 Related Work

2.1 Deep Learning in Object Tracking

Modern tracking systems based on the deep network can be divided into two categories. The first branch is based on a tracking-by-detection framework [27]. The second branch is mainly based on SiamFC [1] and SiamRPN [20]. For SiamFC, these methods focus on discriminative feature learning [35] [12] [34], exemplar-candidate pairs modeling [6], and dynamical hyperparameter optimization [8] [7]. For SiamRPN, some researchers introduce a more powerful network cascaded model [9] or deeper architecture [19] for region proposal searching. DaSiam [37] proposes a distractor-aware training strategy to generate semantic pairs and suppress semantic distractor. In summary, the Siamese trackers show their superior performance due to the high localization accuracy and efficiency, but most of these trackers are sensitive to the adversarial pertubations of the input data. Therefore, investigating the robustness of these trackers under adversarial attracks becomes crucial.

2.2 Iterative and Generative Adversary

The existing adversarial attacks are based primarily on the optimization algorithm and generation algorithm. The optimization-based adversarial attack discovers the noise's direction by calculating the DNNs' gradient within a certain limit [3]. I-FGSM [17] decomposes one-step optimization into multiple small steps, and iteratively generates adversarial examples for image classification. DAG [33] regards the candidate proposal for RPN [25] as a sample, and iteratively change the proposal's label to attack object detection and segmentation. Another type of adversarial attack is based on the generator, which can quickly generate adversarial perturbations [2]. GAP [24] uses the ResNet generator architecture to misclassify images of ImageNet [5]. UEA [29] generates transferable adversarial examples by combining multi-layer feature loss and classification

loss, aiming to achieve an untargeted attack in image and video detection. Due to speed limitations, adversarial attacks based on iterative optimization cannot achieve real-time attacks in the visual object tracking task.

3 Generating Adversarial Examples

3.1 Problem Definition

Let $V = \{I_1, ..., I_i, ..., I_n\}$ be a video that contains n video frames. For simplicity, we take one tracking object as the example, thus $\mathcal{B}^{gt} = \{b_1, ..., b_i, ..., b_n\}$ is used to represent the object's ground-truth position in each frame. The visual object tracking will predict the position $\mathcal{B}^{pred}$ of this object in the subsequent frames when given its initial state. For different datasets, the predicted output is different. In general, four points $b_i \in \mathcal{R}^4$ are used to represent the box.

In SiamFC [1], the tracker $f_\theta(\cdot)$ with parameters θ first transforms the reference frame I_R and annotation b^{init} to get an exemplar region $z = \tau(I_R, b^{init})$, and searches a large area b^{search} in the candidate frame I_C to get a candidate region $x = \tau(I_C, b^{search})$. After feature extraction $\varphi(\cdot)$, a fully-convolutional network is used to calculate the similarity between z and x to get the response score map $\mathcal{S} = f_\theta(z, x) = \varphi(z) * \varphi(x)$. A Cosine Window Penalty (CWP) [1] is then added to generate the final bounding box $b_i = CWP(\mathcal{S})$. CWP can penalize the large offset, making the predicted box not far from the previous box.

$\hat{V} = \{\hat{I}_1, ..., \hat{I}_i, ..., \hat{I}_n\}$ represents the adversarial video. The generator mainly attacks the candidate area $\hat{x}_i = \tau(\hat{I}_i, b_i^{search})$ in the adversarial frame $\hat{I}_i$. The definitions of targeted and untargeted attacks in VOT are given below:

(1) Targeted Attack. The adversarial video $\hat{V}$ guides the tracker to track the object along the specified trajectory $\mathcal{C}^{spec}$, i.e., $\forall i, ||\hat{c}_i - c_i^{spec}||_2 \leq \varepsilon$,s.t. $\hat{c}_i = center(CWP(f(z, \hat{x}_i)))$. $center(\cdot)$ gets the prediction center through the prediction box. The Euclidean distance between the prediction center $\hat{c}_i$ and the target center c_i^{spec} should be small. Here we set ε to 20 pixels.

(2) Untargeted Attack. The adversarial video $\hat{V}$ causes the adversarial trajectory $\mathcal{B}^{attack} = \{CWP(f(z, \hat{x}_i))\}_{i=1}^n$ to deviate from the original trajectory $\mathcal{B}^{gt}$ of an object. When the IOU of the prediction box and the ground-truth box is zero, i.e., $IOU(\mathcal{B}^{attack}, \mathcal{B}^{gt}) = 0$, we think that the untargeted attack is successful.

3.2 Drift Loss Attack

Trackers based on the Siamese network are highly dependent on the response map generated by the fully-convolutional network to predict the object's location. Because the SiamFC uses a search area x when predicting the object's location, we can attack this search area to achieve untargeted attack. Over time, the tracker will accumulate the predicted slight offset until the tracker completely loses the object.

In Figure 2 a), the darker the color in response map $\mathcal{S}$, the greater the response score. The red area and green area represent the response regions of the

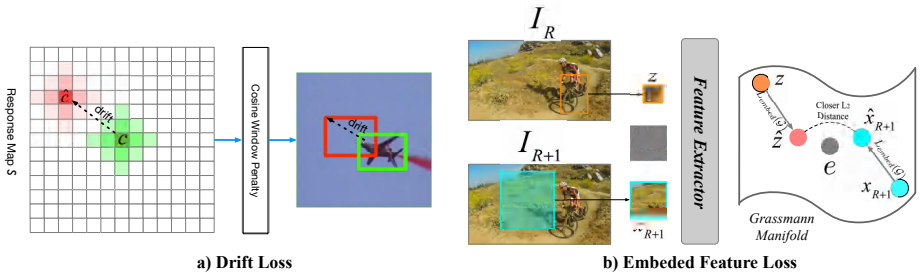

Fig. 2. Overview of the proposed drift loss and embedded feature loss. They are designed for untargeted attack and targeted attack. For details, see sections 3.2 and 3.3.

adversarial image and clean image. c represents the maximum score in the response map. For a well-trained tracker, the response map of clean examples are generally concentrated on the central area (green area). Thus, we propose a drift loss, which generates adversarial perturbations that drift the activation center of $\mathcal{S}$:

$$l(y, s) = \log(1 + exp(-ys)), \tag{1}$$

where s represents the response score and $y \in (-1, 1)$ represents the label of grid in response map $\mathcal{S}$. The central part of the response map $\mathcal{S}$ (green area) is labeled 1, and the rest is -1. In order to generate adversarial examples, the maximum response value of the non-intermediate response map is greater than the maximum response value of the ground-truth, so the score loss of the response map can be written as:

$$\mathcal{L}_{score}(\mathcal{G}) = \min_{p \in \mathcal{S}^{+1}} (l(y[p], s[p])) - \max_{p \in \mathcal{S}^{-1}} (l(y[p], s[p])), \tag{2}$$

where $p \in \mathcal{S}$ represents each position in the response map. The offset of the prediction box depends on the offset of the activation center in the response map. We want the activation center to be as far away from the center as possible, so the distance loss can be expressed as:

$$\mathcal{L}_{dist}(\mathcal{G}) = \frac{\beta_1}{\delta + \|p_{max}^{+1} - p_{max}^{-1}\|_2} - \xi, \tag{3}$$

where $p_{max}^{i} = \arg\max_{p \in \mathcal{S}^i}(s[p]), i = +1, -1$ represents position of max activation scores in positive areas or negative areas of response map. δ is a small real number, and β_1 controls weight in distance loss. ξ controls the offset degree of the activation center. Usually, the activation center leaves the central area. The drift loss consisting of score loss and distance loss can be written as:

$$\mathcal{L}_{drift} = \mathcal{L}_{dist} + \beta_2 \mathcal{L}_{score}. \tag{4}$$

3.3 Embedded Feature Loss Attack

Since the targeted attack requires the tracker to track along the specified trajectory, it is different from the untargeted attack. The drift loss in Section 3.2 is easy to achieve the untargeted attack, but its attack direction is random, and it cannot achieve targeted attack. The input to the targeted attack are a video V and the specified trajectory's centers $\mathcal{C}^{spec}$. Due to the great difference between the object and background, the response value of the candidate image x_{R+1} and the exemplar image z along the specified trajectory will gradually drop to be lower in the background area. Thus, the targeted attack will soon fail.

For effective targeted attack, we need increase the response value. As shown in Figure 2 b), we want to minimize the L_2 distance between the features of the adversarial exemplar and the specific trajectory area. Thus, we propose embedded feature loss that generates adversarial images $\hat{z}$ and $\hat{x}_{R+1}$. The features of the generated adversarial examples are close to the features of the embedded image e.

$$\mathcal{L}_{embed}(\mathcal{G}) = \|\varphi(q + \mathcal{G}(q)) - \varphi(e)\|_2, \tag{5}$$

In Eq 5, e represents the specified trajectory area, $q \in \{z, x_{R+1}\}$ represents input video area. z and x_{R+1} represent the exemplar frame and the $R+1$ frame to track. φ represents the feature function, and $\mathcal{G}(q)$ represents adversarial perturbation. After feature extraction, the features of the adversarial image and the embedded image should be as close as possible to achieve targeted attack.

In the training phase, the choice of embedded images is very important. For example, the feature distance between a shepherd dog and a sled dog is smaller than that of a shepherd dog and an Egyptian cat. In the actual attack, we find that attacking a video frame to an object will produce significant perturbations. We use Gaussian noise to replace the object feature in e to optimize Eq 5, but the specified trajectory remains unchanged.

3.4 Unified and Real-time Adversary

As shown in Figure 3, we train a GAN to generate adversarial examples. Necessarily, generating adversarial perturbations can be seen as an image translation task [24]. We generate adversarial perturbations for candidate images in the candidate frames, which are more difficult to perceive in space. We refer to cycle GAN [36] as a generator to learn the mapping from natural images to adversarial perturbations. We adopt the generator proposed in paper [15] and use nine blocks to generate adversarial perturbations. For the discriminator, we use PatchGAN [13], which uses the overlapping image patch to determine whether the image is true or false.

The loss of the discriminator can be expressed as:

$$\begin{aligned}\mathcal{L}_{\mathcal{D}}(\mathcal{G}, \mathcal{D}, \mathcal{X}) = \mathbb{E}_{x \sim p_{data}(x)}[(\mathcal{D}(\mathcal{G}(x) + x))^2] \\ + \mathbb{E}_{x \sim p_{data}(x)}[(\mathcal{D}(x) - 1)^2].\end{aligned} \tag{6}$$

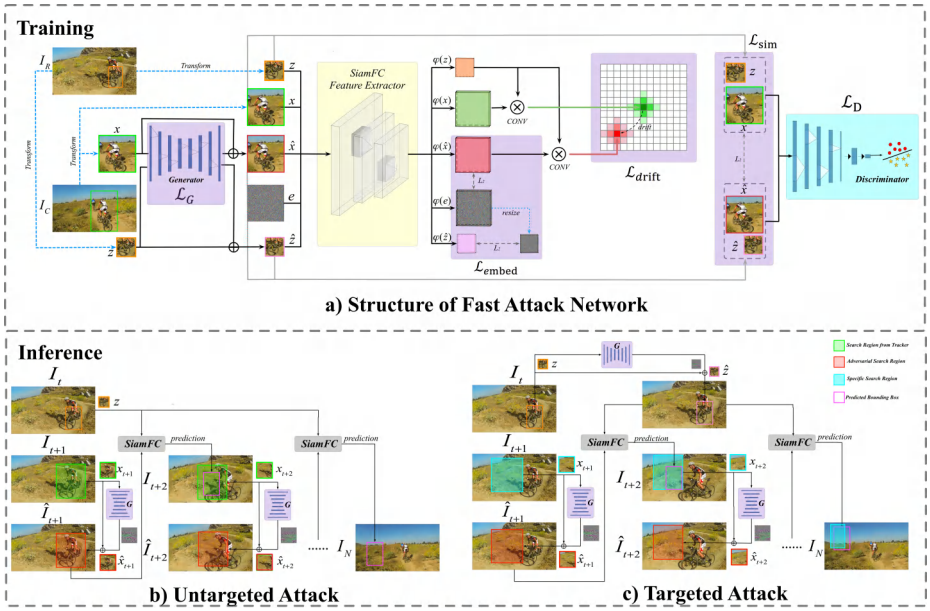

Fig. 3. The training and inference framework of FAN. In a), we train the generator and discriminator using the well-trained SiamFC architecture (yellow area + convolution parameters). The losses of the generator and the discriminator are highlighted by the purple parts and blue parts, respectively. We can achieve both a targeted attack and an untargeted attack by adjusting the loss weight of the generator. For untargeted attack b), we only generate adversarial perturbations for search area x in the candidate image I_C. For targeted attack c), we attack both the exemplar image z and the specific search area (the blue part in c), which is determined by the specific trajectory.

In the training phase, we train the discriminator by minimizing Eq 6. In order to make the image generated by the generator more realistic, the loss of the generator can be expressed as:

$$\mathcal{L}_{\mathcal{G}}(\mathcal{G}, \mathcal{D}, \mathcal{X}) = \mathbb{E}_{x \sim p_{data}(x)}[(\mathcal{D}(\mathcal{G}(x) + x) - 1)^2]. \tag{7}$$

In addition, we use the L_2 distance as a measure to minimize the loss of similarity so that the adversarial image is closer to the clean image in visual space. The loss of similarity can be expressed as:

$$\mathcal{L}_{sim}(\mathcal{G}) = \mathbb{E}[\|\mathcal{X} - \hat{\mathcal{X}}\|_2]. \tag{8}$$

Finally, the full objective for the generator can be expressed as:

$$\mathcal{L} = \mathcal{L}_{\mathcal{G}} + \alpha_1 \mathcal{L}_{sim} + \alpha_2 \mathcal{L}_{embed} + \alpha_3 \mathcal{L}_{drift}, \tag{9}$$

We propose a **unified network** architecture, which can achieve a targeted attack and untargeted attack by adjusting the hyperparameters. β_1, β_2 make L_{dist}

and L_{score} roughly equal. Thus, there is no need for special adjustment. ξ controls the offset degree of the activation center. α_1 and α_3 control the untargeted attack. We fix α_3 and adjust α_1 from the visual quality. α_2 controls embedding image features. We test value from 0.05-0.1 and the precision score improves ten percentage. For the targeted attack, we do not need drift loss, so set α_3 to 0, $\alpha_1 = 0.0024$ and $\alpha_2 = 0.1$. For the untargeted attack, we set α_2 to 0, $\alpha_1 = 0.0016$, and $\alpha_3 = 10$. In Eq 3, we set $\beta_1 = 1$, $\delta = 1 * 10^{-10}$, $\xi = 0.7$. In Eq 4, β_2 is set to 10. We use Adam algorithm [22] to optimize generator $\mathcal{G}$ and discriminator $\mathcal{D}$ alternatively. Using a GPU Titan XP, we can get the best weight by iterating about 10 epochs(about 3 hours) on the ILSVRC 2015 dataset.

Since the prediction box of the tracker in the current frame is strongly dependent on the results of the previous frame, we can make the prediction box produce a small error offset and eventually stay away from the ground-truth trajectory. We only add perturbations to the candidate image x for the untargeted attack. For targeted attack, we embed features of embedding images in exemplar states z and candidate images x by adding adversarial perturbations. Although the adversarial attack deals with a large number of videos, the generator can generate adversarial examples in milliseconds. This enables us to complete the **real-time adversarial attack** for visual object tracking.

4 Experiments

4.1 Datasets and Threat models

We train the generator and discriminator on the training set of the ILSVRC 2015 dataset [26]. We refer to the training strategy in SiamFC [1]. After training is completed, the generator is tested on four challenging visual object tracking datasets without parameter adjustment: OTB2013 [31], OTB2015, VOT2014, and VOT2018 [16]. Specifically, the VOT datasets will be re-initialized after the tracker fails to track. Therefore, it is more difficult to attack VOT datasets than OTB datasets. We use SiamFC based on Alexnet as a white-box attack model. SiamRPN [20], SiamRPN+CIR [35] and SiamRPN++ [19] as black-box attack models.

4.2 Evaluation Metrics

Since the targeted attack and untargeted attack are different, we define their own evaluation criteria, respectively.

Untargeted Attack Evaluation: In the OTB dataset, we use success score, precision score, and success rate as the evaluation criteria. The **success score** calculates the average IOU of the prediction box and the ground-truth. The **precision score** indicates the percentages of the video frames whose euclidean distance between the estimated centers and ground-truth centers is less than the given threshold. The percentage of successful attacked frames to all the frames is the **success rate**.

In the VOT dataset, we measure the accuracy in the videos using the **success score**. Considering the restart mechanism in the VOT dataset, robustness is a more important evaluation metric. **Mean-Failures** refer to calculating the average number of failures for the object tracking algorithm in all datasets.
Targeted Attack Evaluation: The target attack requires the tracker to move according to a specific trajectory, so we use the **precision score** as the evaluation criteria. The higher the precision score, the more effective the targeted attack.
Image Quality Assessment: We use **Mean-SSIM** to evaluate the quality of adversarial videos. Mean-SSIM calculates the average SSIM of frames in videos. The generated adversarial perturbations are difficult to be found when Mean-SSIM is close to 1.

4.3 Untargeted Attack Results

Datasets		Clean Videos	Adversarial Videos	Drop Rate
OTB2013	Success Score	0.53	0.14	74%
	Precision Score	0.71	0.17	76%
	Success Rate	0.66	0.12	**81%**
	Mean-SSIM	1	0.93	7%
OTB2015	Success Score	0.53	0.15	72%
	Precision Score	0.72	0.18	75%
	Success Rate	0.66	0.12	**81%**
	Mean-SSIM	1	0.93	7%
VOT2014	Success Score	0.54	0.42	22%
	Mean-Failures	28	112	300%
	Mean-SSIM	1	0.94	6%
VOT2018	Success Score	0.49	0.42	14%
	Mean-Failures	48	246	**413%**
	Mean-SSIM	1	0.97	**3%**

Table 1. Untargeted attacks on VOT and OTB datasets. We use drop rate to measure the attack performance. Large Mean-Failures means the tracker frequently lost objects.

In Table 1, we report the results of the untargeted attack on four tracking datasets. The second and the third columns represent the object tracking results of SiamFC on the clean video and the adversarial video. The drop rate of tracking evaluation metrics for OTB datasets has fallen by at least 72%, indicating that our attack method is effective. For the quality assessment, the highest drop rate is only 7%, which is sufficient to show that adversarial perturbations generated by our attack method are visually imperceptible.

We find that the success rate is the most vulnerable evaluation metrics on the OTB dataset, with a drop rate of 81%. This indicates that our attack method can effectively reduce the IOU between the prediction box and the ground-truth box.

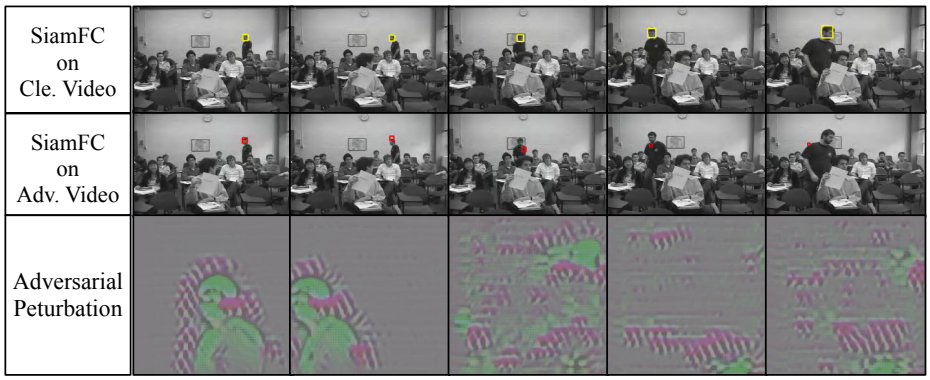

Fig. 4. We visualize the tracking results under the untargeted attack. Yellow represents a ground-truth bounding box and, red represents predicted bounding box by trackers.

Our attack method increases the number of tracking failures as high as 413% on the VOT2018 dataset. Therefore, our attack method can still effectively fool the tracker and cause it to lose objects under the reinitialization mechanism. However, compared with the OTB datasets, there is no significant decrease in the success score on VOT datasets. The reason may be that the success score is still high because the tracker keeps reinitializing the object. According to the definition of untargeted attack, Mean-Failures is more reasonable for evaluating adversarial attacks. Finally, the VOT2018's Mean-SSIM dropped only 3%. Our generator sparsely attacks video frames over time, resulting in that perturbations less difficult to be perceived.

We show an adversarial video in Figure 4, which is sampled equidistantly in time from left to right. We added slight perturbations in search images to successfully fool the SiamFC tracker. This kind of attack method does not produce too much deviation in a short-time and is difficult to be detected by trackers. The third line represents adversarial perturbations, and FAN can adaptively attack the critical feature areas without prior shapes.

The left-to-right in Figure 5 are the results of the uniform sampling of a video over time. By comparing the second row and the fourth row, we can see that the responding area of the clean image is concentrated, and the scores are not much different (the green part). However, the adversarial examples generated by FAN start to cause a large range of high scores in the response map and are relatively scattered. These scattered high-scoring areas will fool the SiamFC tracker to make it impossible to distinguish the object. Due to incorrect activation of the response map, the search areas in adversarial examples will gradually shrink over time. The subsequent adversarial perturbations will also increase the degree of narrowing of the search areas (the extent of the fourth line is reduced differently in equal time). The perturbations gradually decrease in space over time due to the FAN attack on the search areas.

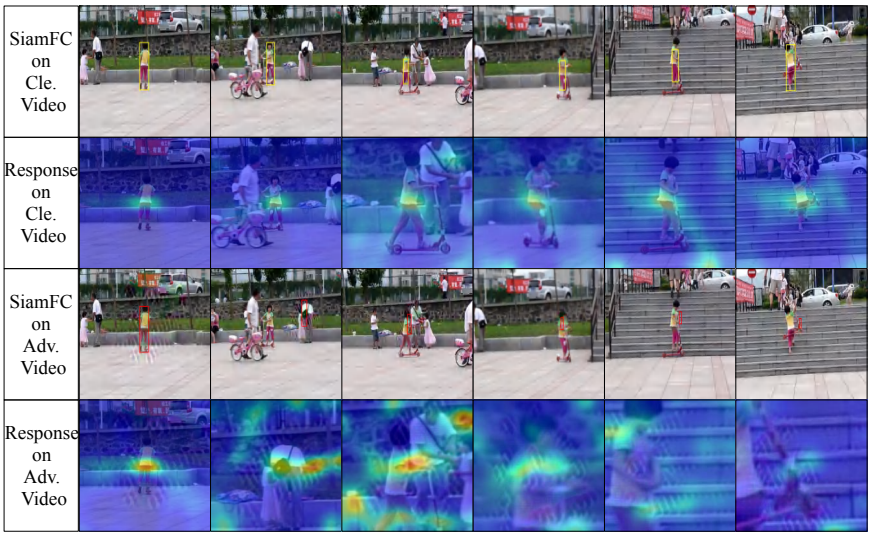

Fig. 5. The visualization of response maps between adversarial examples and clean videos, respectively. Blue indicates low response, and red indicates high response.

4.4 Comparisons with the Baselines

To better show the performance, we compare our FAN method with the widely used FGSM [11] and PGD [21]. The results are shown in Table 2.

Methods	Success Score	Precision Score	Success Rate	Mean-SSIM	Time(s)
FGSM	3%	2%	3%	0.95	0.03
PGD	3%	2%	3%	0.97	3.53
FAN	**74%**	**76%**	**81%**	**0.94**	**0.01**

Table 2. The untargeted attacks on OTB2013. Compared with the FAN method, the modified FGSM and PGD methods cannot achieve effective attacks. The percentage represents the drop rate compared to clean video.

Because FGSM and PGD are used to attack the image classification task, and cannot directly attack the visual object tracking task. Therefore we make some modifications. In object tracking, tracker searches the most similar regions in each frame with the reference patch. The most similar regions in the response map are labeled 1; the others in the response map are -1. Therefore, for FGSM and PGD, the attack target is to change the correct label in the response map (invert label 1 to -1). We perform experiments on the modified FGSM and PGD methods at OTB2013 and compared them with the FAN method.

The percentages in Table 2 represent the drop rate versus different metrics. We can see that these two methods are not effective for attacking VOT tasks. Besides, the average time for PGD to process a sample is 3.5s, which is not suitable for attacking a large number of frames. Under the same hardware conditions, our method process a sample only need 0.01s, and it can effectively attack clean videos.

4.5 Targeted Attack Results

We need to set specific trajectories for the video frames in the dataset to achieve targeted attack. Since the VOT datasets will be reinitialized when the tracker is far away from the ground-truth, there is no point in implementing a targeted attack on the VOT datasets. Our targeted attack method still works because it can cause the tracker to restart multiple times on the VOT dataset. For clean videos in the VOT2014, SiamFC will restart tracking per 108.8 frames. After attacked by our method, SiamFC will restart tracking per 14 frames, which shows our method significantly increases numbers of restart for tracker in the VOT dataset in the targeted attacks.

Datasets		Clean Videos	Adversarial Videos	Drop Rate
OTB2013	Precision Score	0.69	0.41	40.6%
	Mean-SSIM	1	0.92	8%
OTB2015	Precision Score	0.71	0.42	40.8%
	Mean-SSIM	1	0.92	8%

Table 3. An overview of the targeted attack results. We use precision scores to evaluate targeted attacks. A high precision score means that the tracker's prediction is close to the specified trajectory.

We conduct experiments on OTB2013 and OTB2015 datasets. Manually labeling specific trajectories on these datasets will be time-consuming. Therefore, we generate specific trajectories based on the original annotations. Here we consider the most difficult case of a targeted attack. That is, the generated specific trajectory is completely opposite to the original trajectory. We use the following rules to calculate the bounding box for specific trajectory:

$$b_t^{spec} = \begin{cases} b_0^{gt} & t = 1 \\ 2 * b_{t-1}^{gt} - b_t^{gt} & t \geq 2, \end{cases} \tag{10}$$

where b^{spec} represents the bounding box of specified trajectory, and b^{gt} represents ground-truth in datasets.

In Table 3, the first and second columns represent precision scores of tracker's predicted trajectory on clean videos and adversarial videos. Experiment results show that the tracking system after the targeted attack cannot reach the same

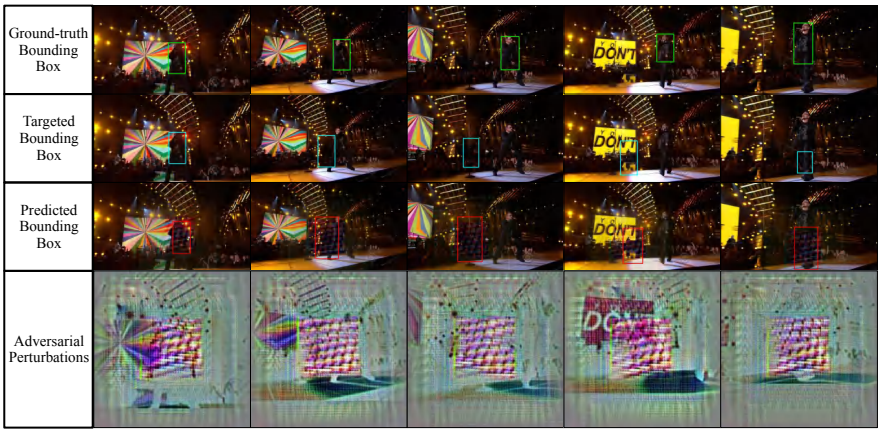

Fig. 6. The results under targeted attacks. Green represents a ground-truth bounding box, cyan represents the specific bounding box, and red represents the predicted bounding box by trackers. The cyan and red boxes are basically the same in time series, which indicates that targeted attack is successful.

precision scores on the clean video. The reason for this result may be that the automatically generated specific trajectory is not the best path that the targeted attack can choose. Even if the targeted attack of visual tracking is more difficult than an untargeted attack, FAN can still successfully attack most videos under the most difficult specific trajectories.

We visualize the results of the targeted attack in Figure 6. The first and third lines represent bounding boxes on the clean video and the adversarial video. The second line represents the specific trajectories we automatically generated according to Eq 10. It can be seen that the predicted bounding box by tracker is basically the same as the specific bounding box. The fourth line shows adversarial perturbations from the search region, which is significantly stronger than adversarial perturbations in the untargeted attack. Therefore, the targeted attack is more difficult than the untargeted attack under limited disturbance.

4.6 Transferability to SiamRPN

We use SiamRPN [20], SiamRPN+CIR [35], SiamRPN++ [19] as black-box attack models to verify the transferability of adversarial examples generated by FAN. SiamRPN uses an RPN network to perform location regression and classification on the response map. SiamRPN+CIR uses the ResNeXt22 network to replace SiamRPN's Alexnet. SiamRPN++ performs layer-wise and depth-wise aggregations to improve accuracy.

The experimental results are shown in Table 4. The first column refers to the drop rate of a white-box attack method. The other columns refer to the drop rate of black-box attack methods. We find that black-box attack methods have a

Methods		SiamFC	SiamRPN	SiamRPN+CIR	SiamRPN++
OTB2013	Success Score	74%	55%	46%	33%
	Precision Score	76%	47%	58%	35%
	Success Rate	81%	56%	47%	35%
OTB2015	Success Score	72%	44%	45%	32%
	Precision Score	75%	51%	58%	37%
	Success Rate	81%	55%	43%	39%

Table 4. Transferability of adversarial examples on two datasets.

lower drop rate than the white-box attack method. It is obvious that black-box attack methods are more difficult than a white-box attack method. Except for the precision score, the performance of the black-box attack in SiamRPN is better than SiamRPN+CIR. This may be due to SiamRPN and SiamFC using the same feature extraction network AlexNet. The black-box attack in SiamRPN++ performs the worst. This is because the architecture of SiamRPN++ can correct some spatial offsets. Even in this case, the drop rate of the black-box attacks can still reach 32%. The results show that our method can still show good transferability for different tracking methods.

5 Conclusion

In this paper, we accomplished the adversarial attacks for the Visual Object Tracking (VOT) task. We analyzed the weaknesses of DNNs based VOT models: the feature networks and the loss function, and then designed different attacking strategies. We firstly presented a drift loss to make the high-score area obtained by adversarial examples be offset with the original area. Then a pre-defined trajectory was embedded into the feature space of the original images to perform the targeted attack. Finally, we proposed an end-to-end framework to integrate these two modules. Experiments conducted on two public datasets verified the effectiveness of the proposed method. In addition, our method not only achieved excellent performance on the white-box attack, but also on the black-box attack, which expanded its application area. Furthermore, the image quality assessment showed that the generated adversarial examples had good imperceptibility, which guaranteed the security of the adversarial examples.

Acknowledgement

Supported by the National Key R&D Program of China (Grant No. 2018AAA0100600), National Natural Science Foundation of China (No. U1636214, 61861166002, No. 61806109), Beijing Natural Science Foundation (No. L182057), Zhejiang Lab (NO.2019NB0AB01), Peng Cheng Laboratory Project of Guangdong Province PCL2018KP004.

References

1. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. In: European Conference on Computer Vision (ECCV). pp. 850–865. Springer (2016)
2. Bose, A.J., Aarabi, P.: Adversarial attacks on face detectors using neural net based constrained optimization. In: 2018 IEEE 20th International Workshop on Multimedia Signal Processing (MMSP). pp. 1–6. IEEE (2018)
3. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 ieee symposium on security and privacy (sp). pp. 39–57. IEEE (2017)
4. Danelljan, M., Robinson, A., Khan, F.S., Felsberg, M.: Beyond correlation filters: Learning continuous convolution operators for visual tracking. In: European Conference on Computer Vision (ECCV). pp. 472–488. Springer (2016)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 248–255. Ieee (2009)
6. Dong, X., Shen, J.: Triplet loss in siamese network for object tracking. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 459–474 (2018)
7. Dong, X., Shen, J., Wang, W., Liu, Y., Shao, L., Porikli, F.: Hyperparameter optimization for tracking with continuous deep q-learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 518–527 (2018)
8. Dong, X., Shen, J., Wang, W., Shao, L., Ling, H., Porikli, F.: Dynamical hyperparameter optimization via deep reinforcement learning in tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence (2019)
9. Fan, H., Ling, H.: Siamese cascaded region proposal networks for real-time visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7952–7961 (2019)
10. Fischer, V., Kumar, M.C., Metzen, J.H., Brox, T.: Adversarial examples for semantic image segmentation. arXiv preprint arXiv:1703.01101 (2017)
11. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
12. He, A., Luo, C., Tian, X., Zeng, W.: A twofold siamese network for real-time object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4834–4843 (2018)
13. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1125–1134 (2017)
14. Jia, X., Wei, X., Cao, X., Foroosh, H.: Comdefend: An efficient image compression model to defend adversarial examples. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6084–6092 (2019)
15. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: European Conference on Computer Vision (ECCV). pp. 694–711. Springer (2016)
16. 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 vot2017 challenge results. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 1949–1972 (2017)

17. Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial examples in the physical world. arXiv preprint arXiv:1607.02533 (2016)
18. Lee, K.H., Hwang, J.N.: On-road pedestrian tracking across multiple driving recorders. IEEE Transactions on Multimedia **17**(9), 1429–1438 (2015)
19. Li, B., Wu, W., Wang, Q., Zhang, F., Xing, J., Yan, J.: Siamrpn++: Evolution of siamese visual tracking with very deep networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4282–4291 (2019)
20. Li, B., Yan, J., Wu, W., Zhu, Z., Hu, X.: High performance visual tracking with siamese region proposal network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8971–8980 (2018)
21. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083 (2017)
22. Mathieu, M., Couprie, C., LeCun, Y.: Deep multi-scale video prediction beyond mean square error. arXiv preprint arXiv:1511.05440 (2015)
23. Moosavi-Dezfooli, S.M., Fawzi, A., Fawzi, O., Frossard, P.: Universal adversarial perturbations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1765–1773 (2017)
24. Poursaeed, O., Katsman, I., Gao, B., Belongie, S.: Generative adversarial perturbations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4422–4431 (2018)
25. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015)
26. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015)
27. 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: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8990–8999 (2018)
28. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
29. Wei, X., Liang, S., Chen, N., Cao, X.: Transferable adversarial attacks for image and video object detection. arXiv preprint arXiv:1811.12641 (2018)
30. Wong, E., Kolter, J.Z.: Provable defenses against adversarial examples via the convex outer adversarial polytope. arXiv preprint arXiv:1711.00851 (2017)
31. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2411–2418 (2013)
32. Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(9), 1834–1848 (2015)
33. Xie, C., Wang, J., Zhang, Z., Zhou, Y., Xie, L., Yuille, A.: Adversarial examples for semantic segmentation and object detection. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 1369–1378 (2017)
34. Yin, Z., Wen, C., Huang, Z., Yang, F., Yang, Z.: Siamvgg-llc: Visual tracking using llc and deeper siamese networks. In: 2019 IEEE 19th International Conference on Communication Technology (ICCT). pp. 1683–1687. IEEE (2019)

35. Zhang, Z., Peng, H.: Deeper and wider siamese networks for real-time visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4591–4600 (2019)
36. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 2223–2232 (2017)
37. Zhu, Z., Wang, Q., Li, B., Wu, W., Yan, J., Hu, W.: Distractor-aware siamese networks for visual object tracking. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 101–117 (2018)