A Unified Multi-Scenario Attacking Network for Visual Object Tracking

Xuesong Chen[1], Canmiao Fu[2], Feng Zheng[*4], Yong Zhao[3],
Hongsheng Li[1], Ping Luo[5], Guo-Jun Qi[6]
[1]The Chinese University of Hong Kong [2]WeChat AI, Tencent [3]Peking University
[4]Depatment of Computer Science and Engineering, Southern University of Science and Technology
[5]The University of Hong Kong [6]Laboratory for MAPLE, Futurewei Technologies
{cedarchen, fcm, yongzhao}@pku.edu.cn, zhengf@sustech.edu.cn

Abstract

Existing methods of adversarial attacks successfully generate adversarial examples to confuse Deep Neural Networks (DNNs) of image classification and object detection, resulting in wrong predictions. However, these methods are difficult to attack models of video object tracking, because the tracking algorithms could handle sequential information across video frames and the categories of targets tracked are normally unknown in advance. In this paper, we propose a Unified and Effective Network, named UEN, to attack visual object tracking models. There are several appealing characteristics of UEN: (1) UEN could produce various invisible adversarial perturbations according to different attack settings by using only one simple end-to-end network with three ingenious loss function; (2) UEN could generate general visible adversarial patch patterns to attack the advanced trackers in the real-world; (3) Extensive experiments show that UEN is able to attack many state-of-the-art trackers effectively (*e.g.* SiamRPN-based networks and DiMP) on popular tracking datasets including OTB100, UAV123, and GOT10K, making online real-time attacks possible. The attack results outperform the introduced baseline in terms of attacking ability and attacking efficiency.

Introduction

Deep Neural Networks (DNNs) has made important breakthroughs in many applications over the past decade. However, recent researches on adversarial attacks (Carlini and Wagner 2017; Goodfellow, Shlens, and Szegedy 2014; Kurakin, Goodfellow, and Bengio 2016) have revealed that deep learning models, e.g., classification, detection, and semantic segmentation networks, are vulnerable to adversarial perturbations, which are in the form of some slight disturbances on images causing models to output wrong results. For example, (Szegedy et al. 2013) first shows that adversarial examples, generated by adding visually imperceptible perturbations to the original images, could make classification models predict a wrong label with high confidence. Further, (Thys, Van Ranst, and Goedemé 2019) successfully generates adversarial patches that can hide a person from a person detector while (Jia et al. 2019) studies adversarial attacks against the visual perception pipeline in autonomous driving. However, compared with classification, detection, one of the security-critical problems in computer vision—Single Object Tracking (SOT), has attracted limited research efforts for different attack scenarios. Therefore, we study the efficient multi-scenario adversarial attack algorithm against SOT in this paper.

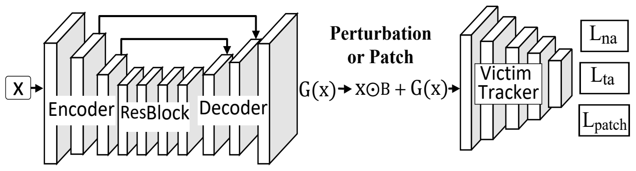


Figure 1: The framework of our UEN, which consists of a generator G and a attacked victim model F. Three different losses are developed for the multi-scenario attacks.

Using typical attack methods (Jia et al. 2019; Szegedy et al. 2013; Thys, Van Ranst, and Goedemé 2019) to generate adversarial perturbations against SOT is difficult and we analyze and summarize several reasons for it. Firstly, SOT algorithms could handle information across frames in real-time to locate the trajectory of the target in videos. Consequently, it is unreasonable to attack only one video frame to fail the tracker, although it works for attacks of classification or detection. Secondly, the requirements of attacks against SOT are different from image classification and detection. For example, *targeted* and *non-targeted* attack of classification tasks only need to make the network output a specific or arbitrary wrong category, while *non-targeted* attack on tracking should make the output box of a tracker deviate from the target area continuously. For the *targeted* attack, the attacker should hijack the output frame of the network to the specified distracter, which put demands on attacking ability of the confidence score, position, and shape of candidate boxes. Thirdly, for the real-world attacks, trackers' preprocessing of input images like downsampling, will cause information loss and the change of camera angle may affect the appearance of adversarial examples, which puts an obstacle to the robustness of the real-world adversarial attacks.

Recently, a few existing works relevant to SOT attacks, such as PAT (Wiyatno and Xu 2019) and SPARK (Guo et al. 2020), are both optimization-based methods and therefore have many limitations. By adapting existing attack methods for classification attacking, they generate adversarial exam-

[*]Corresponding author

Copyright © 2021, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

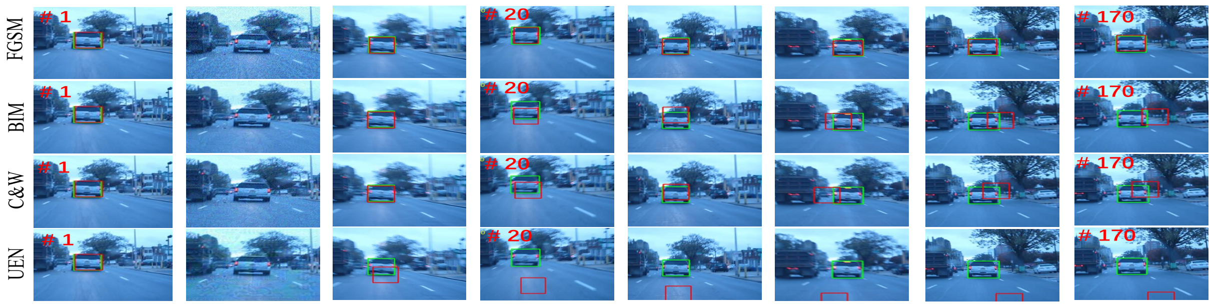


Figure 2: Comparative generated adversarial examples (the second column) and attacking results of UEN and other adapted baselines for non-target attacks against SiamRPN++. Green boxes mean the ground-truth, while red boxes denote the tracker's output and numbers represent the frame index. UEN makes the tracker fail within 20 frames.

ples online through forward-backward iterations, which requires access to the network weight to compute the gradient and is a really time-consuming and computationally intensive process. In addition, One-Shot Attack (Chen et al. 2020) only perturb the initial frame of a tracker, which means that it cannot attack a running tracker. At the same time, CSA (Yan et al. 2020a) employs an end-to-end network for *non-targeted* attack only, which enjoys higher efficiency than iterative methods and does not require model weight during attacking. All these methods focus on limited scenes, that is, assuming that the data in the tracking process is available so it is feasible to add disturbance to digital input images. However, for the real-world attack that has wider applications and can cause greater potential risks than online attacks, we cannot take the information for granted. Specifically, we cannot online utilize much information (like model weights and outputs) used in forward-backward iteration attacks and need to produce a physical adversarial patch in advance rather than online optimization.

In this paper, we present a unified and effective end-to-end network, named UEN, for multi-scenario adversarial attacking against SOT, as shown in Figure 1. Optimization methods generate examples slowly by accessing the parameters of the victim tracker in a white-box manner. Whereas we develop a semi white-box attack manner, which means that our UEN can generate adversarial perturbations for input images in real-time without interaction with the victim tracker once the generator is trained. Furthermore, to deal with different attack scenarios, including *target*, *non-target* and real-world attack, we propose three new losses corresponding to the property of the tracking task, namely GIoU-guided *non-targeted* loss, attribute-disentangled *targeted* loss and general adversarial patch loss. Specifically, for the *non-targeted* attack, the GIoU loss is used to guide the generator to generate adversarial examples, which induces the victim tracker to lose its tracking target as fast as possible (see Figure 2). For *targeted* attack, the attribute-disentangled *targeted* attack loss is explored to guide the generator to produce perturbations that can precisely control the output of the tracker, including the shape and locations of the bounding box, to mislead the victim tracker to track specific wrong distracter objects in real-time. Finally, compared with PAT that optimizes the adversarial patches for a specific instance by online iteration, the proposed UEN can generate robust general attack patches with transferability to different categories due to the prior knowledge brought by large amounts of off-line training. For example, the object stuck with a pre-generated adversarial patch can easily blind the tracking system. In summary, our main contributions are as follows:

1. We propose a unified and effective encoder-decoder adversarial attacker UEN to generate multi-scenario adversarial examples against SOT, which can be deployed to different attacking scene and enjoys high efficiency and strong attacking ability.
2. We employ the novel GIoU-guided non-targeted loss to perform *non-targeted* attack and use the attribute-disentangled targeted loss for *targeted* attack. And UEN achieves excellent attacking performance against state-of-the-art trackers for both tasks .
3. UEN is able to successfully perform the general visible adversarial patch attack in real-world in a one-shot manner. The object with the generated adversarial patch can easily blind the system like SiamRPN trackers.

Related Work

Adversarial Examples The adversarial examples can be roughly divided into two categories. The first is based on imperceptible perturbations, which can mislead the neural network but are not easily detectable by human vision. The second is adversarial attack patches that can be employed in the real-world to mislead DNNs in various scenarios. Early iterative-based attack strategies, such as C&W (Carlini and Wagner 2017), BIM (Kurakin, Goodfellow, and Bengio 2016), FGSM (Goodfellow, Shlens, and Szegedy 2014) and MI-FGSM (Dong et al. 2018) produce adversarial examples in the white-box setting, which assumes the victim model's structure and parameters are available, and achieve a high success rate on the classification task. Then, investigations of adversarial attacks are promoted to higher-level tasks like object detection and semantic segmentation. However, the optimization process suffers from problems of low efficiency and cannot satisfy the real-time request in video tasks. Recently, end-to-end methods like GANs (Liu et al. 2019; Xiao et al. 2018) are adopted to approximate adversarial perturbation distribution and achieve state-of-the-art attacking performance while enjoying high efficiency.

Attack Of Single Object Tracking SOT is a fundamental problem in computer vision. Before the rise of deep learning, correlation filtering played an important role in SOT (Danelljan et al. 2017; Lukezic et al. 2017). But in recent years, methods based on convolutional neural networks (CNNs) have achieved the state-of-the-art performance. These trackers can be roughly summarized into two categories: one is online update trackers, including DiMP (Bhat et al. 2019) and others (Jung et al. 2018; Nam and Han 2016; Danelljan et al. 2019), while another is off-line trained trackers, such as SiamRPN (Li et al. 2018) with AlexNet (Krizhevsky, Sutskever, and Hinton 2012) backbone and SiamRPN++ (Li et al. 2019) with MobileNetv2 (Sandler et al. 2018) and ResNet50 (He et al. 2016) backbones. However, there are limited papers that care about SOT related attacks, especially for real-world attack. Specifically, PAT, One-Shot Attack, Spark and Hijacking Attack (Yan et al. 2020b) generate digital adversarial examples by iteration to make trackers lose the object. CSA employs an end-to-end network to produce perturbations for the *non-targeted* attack only. But all these methods cannot generate adversarial patches for real-world attack.

Unified and Effective Network

Our Unified and Effective Network (UEN) is conceptually simple and perspicuous: an encoder-decoder generator is built upon a tracking model to generate adversarial perturbations or adversarial patches for multi-scenario attacking. Next, we will introduce our method in detail, including problem definition, model architecture and loss function.

Problem Definition

SOT correlates the information across frames, so it is almost impossible to make a tracker failure by only attacking one single video frame. We argue that a practical strategy is misleading the model to break away from the tracked target over time. Therefore, we consider an input video sequence $\mathbf{X} \in \mathbb{R}^{T \times W \times H \times C}$ with the ground truth bounding box $\mathbf{b}_t^{gt} \in \mathbb{R}^{4 \times 1}$ of object's location in t_{th} frame, where T is the number of video frames, W, H and C are the width, height, and channel of the video frames, respectively. Assuming a tracker F initialized by a cropped object template $\mathbf{z}$, for the frame $\mathbf{x}_t$ in a video, the tracker F will sample N bounding boxes as candidates set $\{\mathbf{b}_t^1, \mathbf{b}_t^2, ..., \mathbf{b}_t^N\}$ from the search region that is usually determined by the target's position in the last frame. After calculating the confidence scores set $\{(y_t^i, \mathbf{b}_t^i)\} = F(\mathbf{z}, \mathbf{x}_t)$, where $1 \leq i \leq N$, the bounding box $\mathbf{b}_t^p$, where $p = argmax_{1 \leq i \leq N}(y_t^i)$, with the highest score y_t^p will be adopted as the tracker's output.

The goal of an adversarial attack against tracking in this work is causing the tracker to lost its target by adversarial examples. Specifically, there are two main types of attacks: (1) Adversarial perturbation attack. Employing this attack, we can induce different behaviors of victim trackers over time, such as simply breaking away from their target or focusing on other specific distracters of the scene. For the former case, we name it Non-targeted Attack (NA), aiming to make $\mathbf{b}_t^p$ far away from $\mathbf{b}_t^{gt}$. For the latter, we name it Targeted Attack (TA), which aims to hijack tracker's output from $\mathbf{b}_t^p$ to $\mathbf{b}_t^{at}$, where $\mathbf{b}_t^{at}$ is the bounding box of other distracters. (2) Adversarial patch attacks. We aim to generate well-designed general attack patches to suppress the target's classification confidence by sticking the patch on the target, which can be employed to perform the real-world attack.

Framework

The overall architecture of our UEN is shown in Figure 1. Our UEN contains two components: a generator G and the target victim tracker F.

For G, we add several cascaded residual blocks and skip connections in the generator's encoder-decoder architecture to break through the information bottleneck, which is analogous to (Chen et al. 2019). Specifically, the structure can be represented by $\{c64s1, \star c128s2, \star c256s2 \times 4, rc256s1 \times 4, \star c256u2 \times 3, \star c128u2, c64u2, c3s1\}$, where c denotes channel, $\star$ denotes skip connection, s/u denotes stride/upsampling rate and r means residual block. And the kernel sizes for convolution and transpose convolution are 3 and 4, respectively. Given an input image, the generator G takes the cropped search region $\mathbf{x}$ as its input and then generate the output $G(\mathbf{x})$. The final output we feed into the tracker is $\mathbf{x} \odot B + G(\mathbf{x})$, where B is a binary mask.

Note that our method can perform various attacks: (1) when generating adversarial perturbation examples, $G(\mathbf{x})$ is clipped in the range of [-16,16], and the value of B is all 1. In particular, instead of using one branch for NA, UEN uses different convolution branches in parallel at the last layer of the decoder to efficiently generate adversarial perturbations with different attack attributes for TA. And each branch is supervised by its own unique *targeted* attack loss so that it can simultaneously generate attribute-disentangled adversarial examples; (2) when generating an adversarial patch, only value in the location of the patch is 0 for B, then the resized $G(\mathbf{x})$ (without clip) can be added to $\mathbf{x}$.

Loss Function

GIoU-Guided Non-targeted Loss IoU is an important indicator for measuring the performance of a tracker. If the IoU between the output and ground truth is less than 0.5, it will be considered as a failure instance. To this end, for attacks, we aim to make the predicted box as far away from the target as possible. Note that, when two boxes are at different distances (without overlap), the IoU indicator cannot provide a distinguishing measure for them. Therefore, we introduce a generalized concept of IoU — GIoU (Rezatofighi et al. 2019). This is served as a new metric or loss for bounding box regression and the value of IoU is extended from (0, 1) to (-1, 1), where -1 means that the two boxes are infinitely far apart. Specifically, the box in the corner always has the smallest GIoU value due to the largest position drift. Inspired by One-Shot Attack, we choose the candidate anchor sets with the largest drift position (at corner) and target's position (at center) as the optimization object:

$$\mathcal{L}_{\text{na}} = y^p + \frac{1}{M} \sum (\{y^{gt}\}_1^M - \{y^k\}_1^M), \tag{1}$$

where $p = argmax_{1 \leq i \leq N}(y^i)$ for original input $\mathbf{x}$, $k = \arg\min_{1 \leq i \leq N} GIoU(\mathbf{b}^p, \mathbf{b}^i)$ for adversarial input $\mathbf{x} \odot B + G(\mathbf{x})$. $\{y^{gt}\}_1^M$ and $\{y^k\}_1^M$ denote anchor score sets, with size of M, at positions of $\mathbf{b}^{gt}$ (at center) and $\mathbf{b}^k$ (at corner).

Vitim Models	Attacker	Non-targeted Attack						
		OTB100			UAV123		GOT10K	
		Suc(%)	Pre(%)	Time(ms)	Suc(%)	Pre(%)	Suc(%)	Pre(%)
SiamRPN(Alex)	Original	66.6	87.6	*	58.6	76.9	60.8	45.6
	FGSM	38.5	50.6	14.5	45.7	56.2	48.7	25.2
	C&W	41.4	4.6	83.6	44.8	47.9	45.7	24.9
	MI-FGSM	6.2	6.4	109.3	9.1	11.6	7.6	3.3
	BIM	4.6	4.6	99.3	4.7	7.2	6.4	2.8
	UEN(ours)	**3.6**	**2.9**	**2.25**	**1.9**	**4.4**	**4.5**	**1.6**
SiamRPN++(Mob)	Original	65.8	86.4	*	61.0	80.1	64.1	50.2
	FGSM	47.1	61.0	31.6	42.3	61.7	47.2	28.5
	C&W	38.1	43.9	208.9	28.5	41.6	40.7	25.7
	MI-FGSM	28.6	31.9	240.9	18.2	28.0	32.8	18.1
	BIM	11.9	13.7	225.5	8.4	13.5	14.2	7.0
	UEN(ours)	**0.3**	**0.3**	**2.25**	**1.2**	**2.8**	**2.8**	**1.0**
SiamRPN++(Res50)	Original	69.6	91.4	*	61.2	79.4	65.1	52.7
	FGSM	58.1	79.4	212.6	51.4	71.8	54.1	37.0
	C&W	48.3	63.3	1224.7	43.6	59.3	49.7	33.6
	MI-FGSM	36.6	49.4	1264.7	32.4	46.6	42.7	26.8
	BIM	33.7	44.8	1249.4	27.5	40.0	37.2	20.5
	UEN(ours)	**8.9**	**10.5**	**2.25**	**7.0**	**10.1**	**17.7**	**4.5**
		Capture	Confidence	Time(ms)	Capture	Confidence	Capture	Confidence
DiMP(Res18)	Original	97.5	89.2	*	89.3	89.2	96.4	85.1
	FGSM	70.1	52.7	31.5	72.2	62.3	76.6	49.4
	C&W	20.7	18.9	143.5	22.9	19.6	30.8	25.2
	MI-FGSM	0.9	10.3	180.2	0.3	5.8	14.0	17.1
	BIM	3.8	12.9	140.7	1.4	8.3	23.5	20.7
	UEN(ours)	**0.0**	**0.1**	**2.31**	**0.0**	**0.0**	**0.0**	**0.0**

Table 1: The *non-targeted* attack results of the proposed UEN on OTB100, UAV123 and GOT10K datasets. The cost time (on NVIDIA P40 GPU) and attack results of 4 adapted iteration-based attack methods are also reported for a more comprehensive comparison. UEN achieves the best performance on 4 state-of-the-art victim trackers.

Attribute-disentangled Targeted Loss This loss is introduced to supervise *targeted* attacks, which aims to guide the generator to produce specific adversarial examples that mislead different attributes of the victim model's output, such as location and shape. Suppose we want to fool a tracker to track a distracter in the scene, we need to mislead the tracker's output frame by frame through generating a specific sequence of adversarial examples. Specifically, considering $\mathbf{b}^p = \{c_x, c_y, w, h\}$ for original input $\mathbf{x}$ and $\mathbf{b}^{at} = \{c_x^{'} c_y^{'}, w^{'}, h^{'}\}$ for adversarial input $\mathbf{x} \odot B + G(\mathbf{x})$, we employ 8 attribute losses $\mathcal{L}_{attribute}$ as:

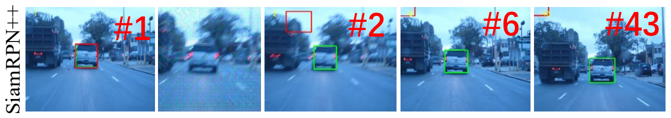
$$\begin{aligned}
\mathcal{L}_{c_x^-}/L_{c_x^+} &= \min(c_x^{'} - c_x \pm 16, 0) + \|c_y^{'} - c_y\|_1 + \\
&\quad \|w^{'} - w\|_1 + \|h^{'} - h\|_1; \\
\mathcal{L}_{c_y^-}/L_{c_y^+} &= \min(c_y^{'} - c_y \pm 16, 0) + \|c_x^{'} - c_x\|_1 + \\
&\quad \|w^{'} - w\|_1 + \|h^{'} - h\|_1; \\
\mathcal{L}_{w^-}/L_{w^+} &= \min(w_x^{'} - w_x \pm 16, 0) + \|c_x^{'} - c_x\|_1 + \\
&\quad \|c_y^{'} - c_y\|_1 + \|h^{'} - h\|_1; \\
\mathcal{L}_{h^-}/L_{h^+} &= \min(h_x^{'} - h_x \pm 16, 0) + \|c_x^{'} - c_x\|_1 + \\
&\quad \|c_y^{'} - c_y\|_1 + \|w^{'} - w\|_1.
\end{aligned} \tag{2}$$

Here, we set the upper offset value of a single frame to 16. And each of the eight losses supervises a corresponding decoder branch. Through the combination of a series of attribute-specific adversarial examples, our method can perform trajectory *targeted* attacks with supervision: $\mathcal{L}_{ta} = \alpha \cdot \frac{1}{M} \sum(\{y^{gt}\}_1^M - \{y^{at}\}_1^M) + \mathcal{L}_{attribute}$, where α is a hyper-parameter to balance different losses. $\{y^{gt}\}$ and $\{y^{at}\}$ denote score sets of anchors locate at $\mathbf{b}^{gt}$ (at center) and $\mathbf{b}^{at}$ (with offset to center), respectively. For shape *targeted* attack branches, $y^{gt} = y^{at}$ (without location offset) so the loss comes to $\mathcal{L}_{ta} = \mathcal{L}_{attribute}$.

General Adversarial Patch Loss One of the challenges of real-world attack is to disturb the tracker's recognition of target with a limited attacking patch area. To meet this challenge, we restrict the patch's area to 0.25 of the target box's area through the binary mask B. Trackers will pre-process images captured by the camera such as resizing under different scenes so we resize $G(\mathbf{x})$ to the restricted area to improve patches' robustness against shape change and information loss caused by downsampling. Another challenge of the real-world attack is that the generation of the adversarial patch should be performed in advance. This means that given only one target input, the generator is expected to produce the general patch that has attacking ability in different real-world scenes in a one-shot manner. Therefore, what we can do in training is using a large amount of offline data to learn how to reduce the tracker's confidence of targets with limited disturbance area. So for adversarial input $\mathbf{x} \odot B + G(\mathbf{x})$, the loss can be written as:

$$\mathcal{L}_{patch} = \frac{1}{M} \sum \{y^{con}\}_1^M, \tag{3}$$

where $\{y^{con}\}_1^M$ is the score set that includes the top M rank indexes of all candidate boxes' scores for original input $\mathbf{x}$.



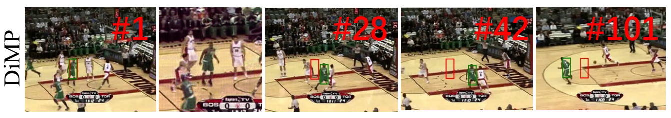


Figure 3: Some intuitive NA results for UEN and the second image is an instance of generated adversarial examples. Green boxes mean the ground-truth, while red boxes denote attacked outputs and the numbers denote the frame indexes. For the SiamRPN++, the perturbations produced by UEN cause the tracker to drift and output wrong boxes. Meanwhile, DiMP exhibits completely different behavior, dying and not updating the output.

Perturbations from	OTB100 Alex	OTB100 Mob	OTB100 Res50
SiamRPN(Alex)	94.6/96.7	24.5/22.6	38.9/35.3
SiamRPN++(Mob)	59.9/58.5	99.5/99.7	62.0/60.0
SiamRPN++(Res50)	38.5/37.0	27.0/25.5	85.6/88.5

Table 2: Transferability of adversarial examples between victim trackers on OTB100 datasets. Values in the table are success rate drop and precision rate drop, respectively.

Experimental Results

In this section, we describe our experimental settings and perform a thorough analysis of the attack results of our UEN about the *targeted* attack (TA), *non-targeted* attack (NA) and general adversarial patch attack.

Datasets We choose 3 popular benchmarks as our experimental datasets, including OTB100 (Wu, Lim, and Yang 2013), UAV123 (Mueller, Smith, and Ghanem 2016) and GOT10K (Huang, Zhao, and Huang 2019). Besides, for SiamRPN-based trackers, we employ COCO (Lin et al. 2014) as the training dataset for NA and TA, while the Re-ID dataset—MSMT17 (Wei et al. 2018) is transformed for the training of adversarial patch attacks in real-world. For DiMP, we use GOT10K as our training datasets.

Experimental Setting We train the generator for 20 epochs, employing an Adam optimizer and cosine decay scheduler with the initial value of 5e-3. For hyper-parameter, we set $\alpha = 10$, $M = 5$. Considering the adversarial input $\mathbf{x} \odot B + G(\mathbf{x})$, $G(\mathbf{x})$ is perturbation with shape of 256×256 and value of [-16,16] for TA and NA, while for real-world attack, $G(\mathbf{x})$ is resized into a patch whose area does not exceed 1/4 of the target box. In training and testing, we both put the resized patch on center of the target's box.

Victim Trackers We attack the state-of-the-art tracking algorithms in the current, including those with/without online updates. For the off-line trained algorithms, SiamRPN-based trackers are adopted to evaluate the attacking ability of UEN. For the online update tracking algorithms, we choose DiMP as the victim model which trains an IoU predictor and a confidence classifier to perform tracking. In testing, DiMP first uses a classifier to obtain the rough position of the target and then online optimizes the IoU confidence of candidate boxes to obtain the precise position.

Non-targeted Attack Metric For SiamRPN-based trackers, we employ the standard evaluation metrics to measure our *non-targeted* attack performance. Specifically, we applied the one-pass evaluation (OPE) with the precision (Pre), with the threshold of 20 pixels, and success plot (Suc) metrics.

For DiMP, however, we use confidence and capture rate to evaluate the impact of the *non-targeted* attack on the tracker. More specifically, the confidence reflects the determination degree of DiMP to the current tracking target. 0 means that the tracker believes the target is completely lost and 1 denotes that the tracker completely determines the target currently being tracked. For online updating trackers, they usually set a confidence threshold to measure the statement of the current frame. When the confidence of the current frame is lower than the threshold, the tracker will set the confidence flag to 'not found' and not update the output box, which means the tacker is dead and fail to track objects. However, the un-updated box may still have a high IoU value with the target because the motion trajectory of most targets in the evaluation dataset is around the center of images. At the same time, the capture rate reflects the ratio of the number of frames in which DiMP has not dead to the total number of frames in the video.

Targeted Attack Metric In practice, we decompose the *targeted* attack into two sub-tasks, namely trajectory attack and shape attack. For trajectory attack, we set a targeted trajectory $\{\mathbf{b}_t^{tr}\}$, which is generated by adding random offset value $\triangle t$ to the targeted position of the previous frame, $\mathbf{b}_t^{tr} = \mathbf{b}_{t-1}^{tr} + \triangle t$. For shape attack, we add random offset value $\triangle s$ to the width or height of the model's predicted box $\mathbf{b}_t^p$. Following SPARK, given the predicted bounding box $\mathbf{b}_t^p$, if $\|\mathbf{ce}(\mathbf{b}_t^p) - \mathbf{ce}(\mathbf{b}_t^{tr})\| < \eta$, where $\mathbf{ce}$ means the center location of box and η is the threshold, we define the trajectory attack as success at frame t. Similarly, for the shape attack, we define the attacker succeeded at frame t when $\|\mathbf{sh}(\mathbf{b}_t^p) - \mathbf{sh}(\mathbf{b}_t^{gt})\| > \varepsilon$, where $\mathbf{sh}$ means the width or height of the box and ε is the threshold. According to the above rules, we define trajectory attack success rate (TR) and shape attack success rate (SR) as *targeted* attack metrics. TR is equal to the ratio of the number of frames where trajectory attack is successful to the total number of attacked frames in the video and SR is similar to TR.

Results of Adversarial Perturbation Attack

Non-targeted Attack To demonstrate the advantages of UEN on efficiency and attack strength, we adapt several optimization-based attack algorithms commonly used in classification tasks as the baseline of NA. Specifically, we change the original loss function of these methods. This means that the generated perturbation aims to make the box that has the greatest confidence among all candidate boxes and with GIoU <= 0 as the output. Table 1 shows a thorough comparison between UEN and other optimization-based baselines. We report the *non-targeted* attacking results of the four state-of-the-art trackers on three datasets. We observe that: For attackers, compared to iterative-based meth-

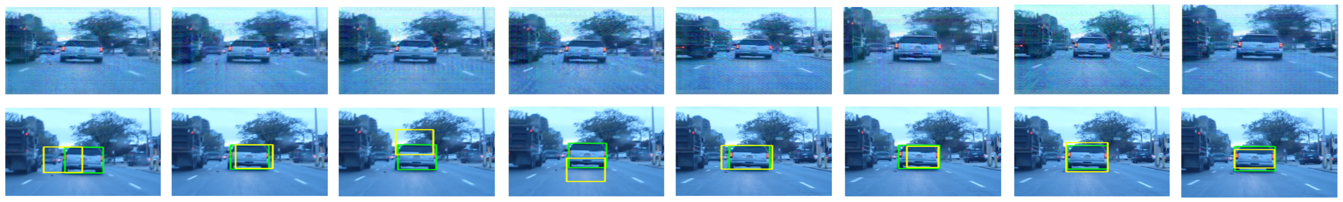


Figure 4: One frame to show the attribute-disentangled adversarial perturbations (the first row) against SiamRPN and attacked results (the second row). Green boxes mean the ground-truth, while yellow boxes denote attacking outputs. Employing different convolution layers in the last layer of the decoder, UEN can generate 8 attribute-specific attack examples simultaneously.

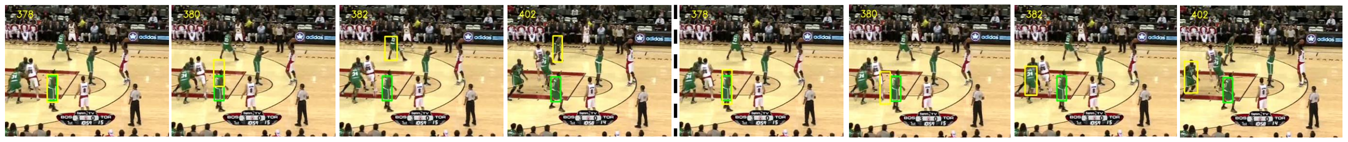


Figure 5: We adopt the video example Basketball from the OTB100 dataset to illustrate the *targeted* attack against SiamRPN. Green boxes mean the ground-truth, while yellow boxes denote attacked outputs. As shown, employing different hijacking trajectories (the left half and the right half) can mislead the tracker to focus on different distracters.

Vitim	OTB100			UAV123			GOT10K		
Model	TR(10)(%)	TR(15)	SR(%)	TR(10)(%)	TR(15)	SR(%)	TR(10)(%)	TR(15)	SR(%)
SiamRPN(Alex)	35.6	86.9	55.9	37.2	85.8	52.2	15.0	33.6	43.7
SiamRPN++(Mob)	74.2	83.1	45.3	59.5	65.4	29.2	14.1	21.1	34.1
SiamRPN++(Res50)	37.6	55.7	58.8	20.3	66.5	51.5	14.7	30.6	38.9

Table 3: The TR and SR results of *targeted* attacks of UEN on OTB100, UAV123 and GOT10K datasets. Different thresholds are reported for comprehensive evaluation.

ods, UEN achieves the highest attack performance among all victim trackers, whether the off-line trained methods like SiamRPN or the updating method DiMP. The trackers attacked by UEN all fail, which means they are almost impossible to track targets. This destructive attack power is more obvious for attacks against the online updating tracker. DiMP fails completely on the three datasets in the experiments, with confidence and capture rates decaying to 0.

For trackers, we can draw two conclusions. Firstly, from different accuracy drop of SiamRPN-based methods, we can see that more parameters result in better resistance to attack. Secondly, the off-line trained siamese-based trackers, without online updating, have greater perturbation tolerance. Because once these trackers are initialized, their template targets' information is fixed. Although this strategy limits the siamese networks' ability to online adapt to target appearance changes, it helps to prevent the siamese-based networks' target information from being disturbed when the networks face adversarial examples, resulting in a robust performance. In contrast, DiMP can update the classifier online during the tracking phase, so it can adapt to the target's appearance change thus gain higher performance. But when maliciously adding well-designed disturbances on the target, DiMP is more likely to be misled.

Table 1 also shows the average cost time of 5 different attackers on OTB100 dataset. UEN has the highest efficiency and absolutely meets the real-time requirements for the SOT task. Whereas iterative baselines calculate the gradients by back-propagation to update perturbations, making it difficult to run in real-time. In addition, the consumption of these methods increases in line with the network's capacity. The deeper the network, the slower the speed. Conversely, UEN generates adversarial examples in an end-to-end manner, and its speed only relies on the input image size. Once the generator has been trained and deployed for attacks, UEN does not needs to access the trackers while iterative-based methods always require access to them.

Figure 3 presents UEN's NA results against different trackers. For SiamRPN++, the location result of the current frame is an interpolation from the previous result, which makes the tracker's outputs more smooth. But when the tracker encounters attacks that make it deviates from the target, the error will be accumulated, causing the tracker to drift eventually. Meanwhile, DiMP does not update the tracking box when the confidence is low so it will lose its tracking ability when the target leaves the search area.

Transferability & Non-targeted Attack We also discuss the transferability across tracker models, that is, employing perturbations generated from one model to attack another different model. We define the success rate drop and precision rate drop to evaluate the transferability. Table 2 shows the transferability performance of UEN on OTB100 dataset. And it should be noted that the diagonal results indicate the white-box attack settings and the others indicate the black-box attack. Our UEN has excellent transferability compared to iterative methods like SPARK.

Targeted Attack Compared with *non-targeted* attacks, *targeted* attacks are more challenging. In practice, for trajectory attack, we set a targeted trajectory $\{\mathbf{b}_t^{tr}\}$ with the random offset value $\triangle t = \pm 15$ to the target's position of the previous frame. Similarly, for shape attack, we add random offset value $\triangle s = \pm 15$ to the width or height of the ground truth bounding box. In particular, when the scale of the tracked target is small, we only add a positive offset to the box to enlarge its scale.

Table 3 reports the *targeted* attack results on three

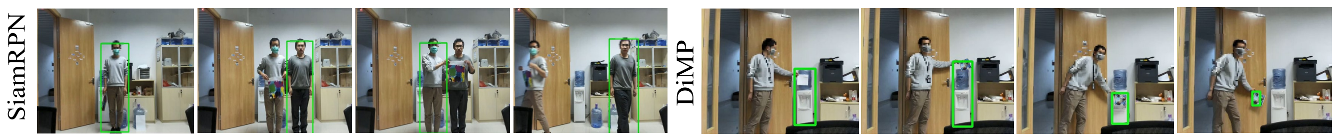


Figure 6: An example of adversarial patch attacks against SiamRPN and DiMP in the real-world. For SiamRPN, when faces an object with the adversarial patch, it will track other similar distracters. For DiMP, due to the online update strategy, it will gradually adapt to the appearance of the adversarial patch, leading the network to focus on the patch rather than the target itself.

Models	Patch Attack	
SiamRPN(Alex)	Suc/Pre(%)	(*)Suc/Pre(%)
	66.6/87.6	19.4/25.6
	Suc/Pre(P)(%)	(*)Suc/Pre(P)(%)
	66.1/80.4	24.0/28.3
DiMP(Res18)	Suc/Pre(%)	(*)Suc/Pre(%)
	67.6/87.2	44.1/78.3

Table 4: The adversarial patch attack results of the proposed UEN on OTB100 dataset. $*$ means attacked results and P denotes a person-subset of OTB100 dataset.

datasets. As mentioned above, online updating trackers will 'dead', which means that they will not update the output box when adversarial examples making the confidence below the threshold. Therefore, we only perform *targeted* attacks on SiamRPN-based methods that regress the output box frame-by-frame. We can see that UEN has good performance on both trajectory attacking ($\eta = 10$ or 15) and shape attacking ($\varepsilon = 15$) against siamese networks using different backbones.

Eight qualitative attribute-disentangled adversarial instances for one video frame are shown in Figure 4. We can observe that the adversarial examples generated by UEN successfully disentangle the attributes of the tracker's output box. Furthermore, after obtaining this ability, we can perform *targeted* attacks through attacking sequences. As shown in Figure 5, this kind of attack is more concealed than non-target attacks that directly destroy tracking systems, and the system owner cannot even realize that the system has been misled by the attacker.

Results of Adversarial Patch Attack

Table 4 reports the results of our adversarial patch attack experiments on the OTB100 dataset. Our experiments show very interesting phenomena: the response of the two tracking algorithms to perturbation attacks and patch attacks is completely different.

Firstly, for SiamRPN, we train an attacking patch generator for pedestrians with the help of the dataset generation method used in SiamRPN and also evaluate its transferability to other categories. As shown in Table 4, for the pedestrian sub-dataset of OTB100, the patch generated on one frame of each tracking target reduces the precision of SiamRPN by 52.1%. Furthermore, models trained on the pedestrian dataset have strong transferability to other categories, reducing the precision of SiamRPN by 62.0% on the whole dataset. These experimental results show that the training and tracking strategy of siamese networks makes their learned features too 'general' to discriminate different instances. To be specific, due to the heavy off-line learning, siamese networks can integrate general prior knowledge which makes it possible to have a certain generalization ability when dealing with untrained categories. However, without online updating, this off-line tracking strategy makes the features of siamese networks lack discrimination. When faced with adversarial patch attacks, the target's confidence is suppressed so SiamRPN will track other distracters.

Secondly, DiMP shows better resistance to adversarial patch attacks than SiamRPN, especially for precision metric, although it is more sensitive to non-target attacks. We infer the reasons are as follows: (1) Compared with the global perturbation, the adversarial patch has a limited area, so it will not have a destructive global impact on the feature and the tracker will not fail immediately. (2) The online update strategy increases the robustness of the model to a certain extent, but also makes the tracker faces the problem of target information pollution or even loss. To demonstrate our conclusions, we deploy UEN in real-world scenes. As shown in Figure 6, when facing real-world attacks, SiamRPN will immediately track other similar targets. But when the attack patch disappears, SiamRPN may find the target again because its target information is not polluted. By contrast, DiMP will gradually update to adapt to the target's appearance change, causing the tracker to focus more on the well-designed attack patch rather than the target itself.

Finally, we believe that after enjoying the prior knowledge brought by big datasets and the bonus of CNN's powerful feature extraction capabilities, how to design a better update strategy for tracking is a question worthy of attention by researchers. In addition, based on our experimental results, we claim that a long short-term update strategy may be a suitable choice.

Conclusions

In this work, we propose a unified, effective yet simple framework, named UEN, for adversarial attacks against SOT. Through an encoder-decoder generator, UEN can generate adversarial perturbations of arbitrary tracking objects for both *targeted* attack and *non-targeted* attack. Furthermore, our UEN can efficiently generate adversarial patches to deal with real-world attacking. Extensive experiments conducted on popular large datasets show the excellent performance of our model in terms of attacking ability and efficiency, and it is also instructive to develop well-designed new tracking algorithms with high-performance.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant 61972188.

References

Bhat, G.; Danelljan, M.; Gool, L. V.; and Timofte, R. 2019. Learning discriminative model prediction for tracking. In *ICCV*, 6182–6191.

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, 39–57.

Chen, X.; Lu, H.; Cheng, K.; Ma, Y.; Zhou, Q.; and Zhao, Y. 2019. Sequentially refined spatial and channel-wise feature aggregation in encoder-decoder network for single image dehazing. In *ICIP*, 2776–2780.

Chen, X.; Yan, X.; Zheng, F.; Jiang, Y.; Xia, S.-T.; Zhao, Y.; and Ji, R. 2020. One-Shot Adversarial Attacks on Visual Tracking With Dual Attention. In *CVPR*, 10176–10185.

Danelljan, M.; Bhat, G.; Khan, F. S.; and Felsberg, M. 2019. Atom: Accurate tracking by overlap maximization. In *CVPR*, 4660–4669.

Danelljan, M.; Bhat, G.; Shahbaz Khan, F.; and Felsberg, M. 2017. Eco: Efficient convolution operators for tracking. In *CVPR*, 6638–6646.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In *CVPR*, 9185–9193.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* .

Guo, Q.; Xie, X.; Juefei-Xu, F.; Ma, L.; Li, Z.; Xue, W.; Feng, W.; and Liu, Y. 2020. SPARK: Spatial-aware online incremental attack against visual tracking. In *ECCV*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

Huang, L.; Zhao, X.; and Huang, K. 2019. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *TPAMI* .

Jia, Y.; Lu, Y.; Shen, J.; Chen, Q. A.; Zhong, Z.; and Wei, T. 2019. Fooling Detection Alone is Not Enough: First Adversarial Attack against Multiple Object Tracking. *arXiv preprint arXiv:1905.11026* .

Jung, I.; Son, J.; Baek, M.; and Han, B. 2018. Real-Time MDNet. In *ECCV*, 83–98.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 1097–1105.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533* .

Li, B.; Wu, W.; Wang, Q.; Zhang, F.; Xing, J.; and Yan, J. 2019. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 4282–4291.

Li, B.; Yan, J.; Wu, W.; Zhu, Z.; and Hu, X. 2018. High performance visual tracking with siamese region proposal network. In *CVPR*, 8971–8980.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *ECCV*, 740–755.

Liu, A.; Liu, X.; Fan, J.; Ma, Y.; Zhang, A.; Xie, H.; and Tao, D. 2019. Perceptual-sensitive gan for generating adversarial patches. In *AAAI*, 1028–1035.

Lukezic, A.; Vojir, T.; Cehovin Zajc, L.; Matas, J.; and Kristan, M. 2017. Discriminative correlation filter with channel and spatial reliability. In *CVPR*, 6309–6318.

Mueller, M.; Smith, N.; and Ghanem, B. 2016. A benchmark and simulator for uav tracking. In *ECCV*, 445–461.

Nam, H.; and Han, B. 2016. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 4293–4302.

Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; and Savarese, S. 2019. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, 658–666.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 4510–4520.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* .

Thys, S.; Van Ranst, W.; and Goedemé, T. 2019. Fooling automated surveillance cameras: adversarial patches to attack person detection. In *CVPRW*, 0–0.

Wei, L.; Zhang, S.; Gao, W.; and Tian, Q. 2018. Person transfer gan to bridge domain gap for person reidentification. In *CVPR*, 79–88.

Wiyatno, R. R.; and Xu, A. 2019. Physical adversarial textures that fool visual object tracking. In *ICCV*, 4822–4831.

Wu, Y.; Lim, J.; and Yang, M.-H. 2013. Online object tracking: A benchmark. In *CVPR*, 2411–2418.

Xiao, C.; Li, B.; Zhu, J.-Y.; He, W.; Liu, M.; and Song, D. 2018. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610* .

Yan, B.; Wang, D.; Lu, H.; and Yang, X. 2020a. Cooling-Shrinking Attack: Blinding the tracker with imperceptible noises. In *CVPR*, 990–999.

Yan, X.; Chen, X.; Jiang, Y.; Xia, S.-T.; Zhao, Y.; and Zheng, F. 2020b. Hijacking Tracker: A Powerful Adversarial Attack on Visual Tracking. In *ICASSP*, 2897–2901.