Universal, Transferable Adversarial Perturbations for Visual Object Trackers

Krishna Kanth Nakka[1] and Mathieu Salzmann[1,2]
[1]CVLab, EPFL, Lausanne, Switzerland
[2]ClearSpace, Switzerland

Abstract—In recent years, Siamese networks have led to great progress in visual object tracking. While these methods were shown to be vulnerable to adversarial attacks, the existing attack strategies do not truly pose great practical threats. They either are too expensive to be performed online, require computing image-dependent perturbations, lead to unrealistic trajectories, or suffer from weak transferability to other black-box trackers. In this paper, we address the above limitations by showing the existence of a universal perturbation that is image agnostic and fools black-box trackers at virtually no cost of perturbation. Furthermore, we show that our framework can be extended to the challenging targeted attack setting that forces the tracker to follow any given trajectory by using diverse directional universal perturbations. At the core of our framework, we propose to learn to generate a single perturbation from the *object template* only, that can be added to every search image and still successfully fool the tracker for the entire video. As a consequence, the resulting generator outputs perturbations that are quasi-independent of the template, thereby making them universal perturbations. Our extensive experiments on four benchmarks datasets, i.e., OTB100, VOT2019, UAV123, and LaSOT, demonstrate that our universal transferable perturbations (computed on SiamRPN++) are highly effective when transferred to other state-of-the-art trackers, such as SiamBAN, SiamCAR, DiMP, and Ocean online.

Index Terms—Adversarial Attacks, Visual Object Tracking, Universal Attacks

1 Introduction

Visual Object Tracking (VOT) [1] is a key component of many vision-based systems, such as surveillance and autonomous driving ones. Studying the robustness of object trackers is therefore critical from a safety point of view. When using deep learning, as most modern tackers do, one particular security criterion is the robustness of the deep network to adversarial attacks, that is, small perturbations aiming to fool the prediction of the model. In recent years, the study of such adversarial attacks has become an increasingly popular topic, extending from image classification [2], [3] to more challenging tasks, such as object detection [4] and segmentation [5], [6].

VOT is no exception to this rule, and several works [7], [8], [9], [10], [11] have designed attacks to fool the popular Siamese-based trackers [12], [13], [14], [15]. Among these, while the attacks in [7], [8], [9] are either too time-consuming or designed to work on entire videos, thus not applicable to fool a tracker in real-time and in an online fashion, the strategies of [10], [11] leverage generative methods [16], [17] to synthesize perturbations in real-time, and can thus effectively attack in an efficient manner. Despite promising results, we observed these generative strategies to suffer from three main drawbacks: (i) They require computing a search-image-dependent perturbation for each frame, which reduces the running speed of the real-time trackers by up to 40 fps, making them ineffective for practical applications such as surveillance and autonomous driving; (ii) They assume the availability of white-box trackers and yield attacks that generalize poorly when transferred to unseen, black-box trackers; (iii) They largely focus on untargeted attacks, whose goal is to make the tracker output any, unspecified, incorrect object location, which can easily be detected because the resulting tracks will typically not be consistent with the environment.

In this paper, we argue that *learning to generate online attacks with high transferability is essential for posing practical threats to trackers and accessing their robustness.* Therefore, we propose to learn a transferable universal perturbation, i.e., a single pre-computed perturbation that can be employed to attack any given video sequence on-the-fly and generalizes to unseen black-box trackers. To achieve this, we introduce a simple yet effective framework that learns to generate a single, one-shot perturbation that is transferable across all the frames of the input video sequence. Unlike existing works [10], [11] that compute search-image-dependent perturbations for *every* search image in the video, we instead synthesize a *single* perturbation from the *template* only and add this perturbation to every subsequent search image. As a consequence of adding the same perturbation to each search image, thus remaining invariant to the search environment, the resulting framework inherently learns to generate powerful transferable perturbations capable of fooling not only every search image in the given video but also other videos and other black-box trackers. In other words, our frameworks learns to generate universal perturbations that are quasi-independent of the input template and of the tracker used to train the generator.

Moreover, in contrast to previous techniques, our approach naturally extends to performing *targeted* attacks so as to steer the tracker to follow any specified trajectory in a controlled fashion. To this end, we condition our generator on the targeted direction and train the resulting conditional generator to produce perturbations that correspond to arbitrary, diverse input directions. Therefore, at test time, we can then pre-compute directional universal perturbations for a small number of diverse directions, e.g., 12 in our experiments, and apply them in turn so as to

krishna.nakka@epfl.ch

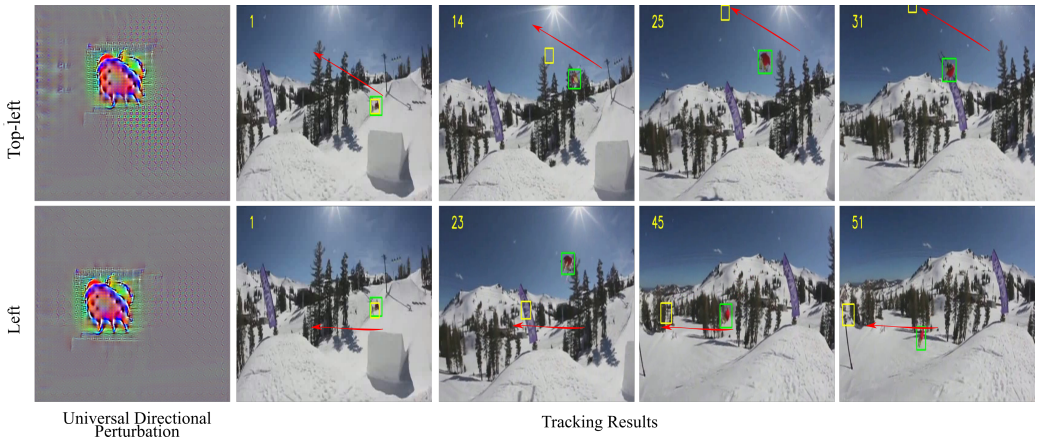


Fig. 1: Universal directional pertubations. Our approach learns an effective universal directional perturbation to attack a black-box tracker throughout the *entire* sequence by forcing it to follow a predefined motion, such as a fixed direction as illustrated above, or a more complicated trajectory, as shown in our experiments. The green box denotes the ground truth, the yellow box the output bounding box under attack, the red arrow the desired target direction. We refer the reader to the demo video in the supplementary material.

generate the desired complex trajectory. We illustrate this in Fig. 1, where a single precomputed universal directional perturbation can steer the black-box tracker to move along a given direction for the entire video sequence and will show more complex arbitrary trajectories in our experiments. We will make our code publicly available upon acceptance.

Overall, our contributions can be summarized as follows:

- We introduce a transferable attack strategy to fool unseen Siamese-based trackers by generating a single universal perturbation. This is the first work that shows the existence of universal perturbations in VOT.
- Our attacking approach does not compromise the operating speed of the tracker and adds no additional computational burden.
- Our framework naturally extends to performing controllable targeted attacks, allowing us to steer the tracker to follow complex, erroneous trajectories. In practice, this would let one generate plausible incorrect tracks, making it harder to detect the attack.

We demonstrate the benefits of our approach on 4 public benchmark datasets, i.e., OTB100, VOT2018, UAV123 and LaSOT, and its transferability using several state-of-the-art trackers, such as SiamBAN, SiamCAR, DiMP, and Ocean online.

2 Related Work

Visual Object Tracking. VOT aims to estimate the position of a template cropped from the first frame of a video in each of the subsequent frames. Unlike most other visual recognition tasks, e.g., image classification or object detection, that rely on predefined categories, VOT seeks to generalize to any target object at inference time. As such, early works mainly focused on measuring the correlation between the template and the search image [18], extended to exploiting multi-channel information [19] and spatial constraints [20], [21].

Nowadays, VOT is commonly addressed by end-to-end learning strategies. In particular, Siamese network-based trackers [12], [13], [14], [15], [22] have grown in popularity because of their good speed-accuracy tradeoff and generalization ability. The progress in this field includes the design of a cross-correlation layer to compare template and search image features [13], the use of a region proposal network (RPN) [23] to reduce the number of correlation operations [13], the introduction of an effective sampling strategy to account for the training data imbalance [22], the use of multi-level feature aggregation and of a spatially-aware sample strategy to better exploit deeper ResNet backbones [12], and the incorporation of a segmentation training objective to improve the tracking accuracy [15]. In our experiments, we will focus on SiamRPN++ [12] as a white box model and study the transferability of our generated adversarial attacks to other modern representative trackers, namely SiamBAN [24], SiamCAR [25], DiMP [26] and Ocean-online [27].

Adversarial Attacks. Inspired by the progress of advesarial attacks in image classification [2], [3], [28], [29], [30], [31], iterative adversarial attacks have been first studied in the context of VOT. In particular, SPARK [7] computes incremental perturbations by using information from the past frames; [8] exploits the full video sequence to attack the template by solving an optimization problem relying on a dual attention loss. Recently, [32] proposed a decision-based black-box attack based on IoU overlap between the original and perturbed frames. While effective, most of the above-mentioned attacks are time-consuming, because of their use of heavy gradient computations or iterative schemes. As such, they are ill-suited to attack an online visual tracking system in real time. [33] also relies on a gradient-based scheme to generate a physical poster that will fool a tracker. While the attack is real-time, it requires to physically alter the environment.

As an efficient alternative to iterative attacks, AdvGAN [17] proposed to train a generator that synthesizes perturbations in a single forward pass. Such generative perturbations were extended to VOT in [10], [11]. For these perturbations to be effective, however, both [10] and [11] proposed to attack every individual search image, by passing it through the generator. To be precise, while [10] studied the problem of attacking the template only the success of the resulting attacks was shown to be significantly lower than that of perturbing each search image. Doing so, however, degrades the tracker running speed by up to 40 fps and generalizes poorly to unseen object environments. Here, instead, we show the existence of universal transferable perturbations, which are trained

using a temporally-transferable attack strategy, yet effectively fool black-box VOT in every search image; the core success of our approach lies in the fact that the perturbation is generated from the template, agnostic to the search images but shared across all frames. This forces the generator to learn powerful transferable perturbation patterns by using minimal but key template information. Furthermore, our approach can be extended to producing targeted attacks by conditioning the generator on desired directions. In contrast to [11], which only briefly studied targeted attacks in the restricted scenario of one specific pre-defined trajectory, our approach allows us to create arbitrary, complex trajectories at test time, by parametrizing them in terms of successive universal targeted perturbations.

3 METHODOLOGY

Problem Definition. Let $\mathbf{X} = \{\mathbf{X}_i\}_1^T$ denote the frames of a video sequence of length T, and $\mathbf{z}$ be the template cropped from the first frame of the video, and $\mathcal{F}(\cdot)$ be the black-box tracker that aims to locate the template $\mathbf{z}$ in search regions extracted from the subsequent video frames. In this work, we aim to find a universal perturbation $\boldsymbol{\delta}$ that, when added to any search region $\mathbf{S}_i$ to obtain an adversarial image $\tilde{\mathbf{S}}_i = \mathbf{S}_i + \boldsymbol{\delta}$, leads to an incorrect target localization in frame i . Note that, unlike universal attacks on image classification [34] that aim to fool a fixed number of predefined object categories, in VOT the objects at training and testing times are non-overlapping.

3.1 Overall Pipeline

Figure 2 illustrates the overall architecture of our *training* framework, which consists of two main modules: a generator $\mathcal{G}$ and a siamese-based white-box tracker $\mathcal{F}_w$. To produce highly transferable perturbations, we propose a simple yet effective learning framework. We first train our perturbation generator to synthesize a *single* perturbation $\boldsymbol{\delta}$ from the template, and add this perturbation to every subsequent search image. As a result of adding the same perturbation, our model learns a temporally-transferable δ that can successfully attack every search image. This makes the learned δ independent of the search image, which further helps generalization to unseen object environments. In other words, by removing the dependence on the search region and relying only on the object template, our generator learns a universal adversarial function that disrupts object-specific features and outputs a perturbation pattern that is quasi-agnostic to the template. Thus, during the attack stage, we precompute a universal perturbation δ_u from any arbitrary input template and perturb the search region of any video sequence, resulting in an incorrect predicted location. Overall, our attack strategy is highly efficient and flexible, and enjoys superior transferability. Below, we introduce our loss functions in detail and then extend our framework to learning universal targeted perturbations.

3.2 Training the Generator

To train the generator, we extract a template $\mathbf{z}$ from the first frame of a given video sequence and feed it to the generator to obtain a perturbation $\boldsymbol{\delta} = \mathcal{G}(\mathbf{z})$. We then crop N search regions from the subsequent video frames using ground-truth information, and add $\boldsymbol{\delta}$ to each such regions to obtain adversarial search regions $\tilde{\mathbf{S}} = \{\tilde{\mathbf{S}}_i\}_1^N$. Finally, we feed the clean template $\mathbf{z}$ and each adversarial search region $\tilde{\mathbf{S}}_i$ to the tracker to produces an adversarial classification map $\tilde{\mathbf{H}}_i \in \mathbb{R}^{H\times W\times K}$ and regression map $\tilde{\mathbf{R}}_i \in \mathbb{R}^{H\times W\times 4K}$.

Standard Loss. Our goal is to obtain the adversarial classification $\tilde{\mathbf{H}}_i$ and regression maps $\tilde{\mathbf{R}}_i$ so as to fool the tracker, i.e., result in erroneously locating the target. To this end, we compute the classification map $\mathbf{H}_i \in \mathbb{R}^{H\times W\times K}$ for the unperturbed search image $\mathbf{S}_i$, and seek to decrease the score in $\tilde{\mathbf{H}}_i$ of any proposal j such that $\mathbf{H}_i(j) > \tau$, where $\mathbf{H}_i(j)$ indicates the probability for anchor j to correspond to the target and τ is a threshold. Following [10], we achieve this by training the perturbation generator $\mathcal{G}$ with the adversarial loss term

$$\begin{aligned}\mathcal{L}_{fool}(\mathcal{F},\mathbf{z},\tilde{\mathbf{S}}_i) = \lambda_1 \sum_{j|\mathbf{H}_i(j)>\tau} \max\left(\tilde{\mathbf{H}}_i(j) - (1-\tilde{\mathbf{H}}_i(j)), \mu_c\right) \\ +\lambda_2 \sum_{j|\mathbf{H}_i(j)>\tau} \left(\max\left(\tilde{\mathbf{R}}_i^w(j), \mu_w\right) + \max\left(\tilde{\mathbf{R}}_i^h(j), \mu_h\right)\right) ,\end{aligned} \tag{1}$$

where $\tilde{\mathbf{R}}_i^w(j)$ and $\tilde{\mathbf{R}}_i^h(j)$ represent the width and height regression values for anchor j. The first term in this objective aims to simultaneously decrease the target probability and increase the background probability for anchor j where the unattacked classification map contained a high target score. The margin μ_c then improves the numerical stability of this dual goal. The second term encourages the target bounding box to shrink, down to the limits μ_w and μ_h, to facilitate deviating the tracker.

Shift Loss. The loss $\mathcal{L}_{fool}$ discussed above only aims to decrease the probability of the anchors obtained from the unattacked search region. Here, we propose to complement this loss with an additional objective seeking to explicitly activate a different anchor box t, which we will show in our experiments to improve the attack effectiveness. Specifically, we aim for this additional loss to activate an anchor away from the search region center, so as to push the target outside the true search region, which ultimately will make the tracker be entirely lost. To achieve this, we seek to activate an anchor t lying at a distance d from the search region center. We then write the loss

$$\mathcal{L}_{shift}(\mathcal{F},\mathbf{z},\tilde{\mathbf{S}}_i) = \lambda_3 L_{cls}(\tilde{\mathbf{H}}_i(t)) + \lambda_4 L_{reg}(\tilde{\mathbf{R}}_i(\mathbf{t}), \mathbf{r}^*) , \tag{2}$$

where L_{cls} is a classification loss encoding the negative log-likelihood of predicting the target at location t, and L_{reg} computes the L_1 loss between the regression values at location t and pre-defined regression values $\mathbf{r}^* \in \mathbb{R}^4$, a vector of 4 parametrizing regression values associated with a ground truth propasal at location t.

Extension to Targeted Attacks. The untargeted shift loss discussed above aims to deviate the tracker from its original trajectory. However, it does not allow the attacker to force the tracker to follow a pre-defined trajectory. To achieve this, we modify our perturbation generator to be conditioned on the desired direction we would like the tracker to predict. In practice, we input this information to the generator as an additional channel, concatenated to the template. Specifically, we compute a binary mask $\mathbf{M}_i \in \{0,1\}^{(W\times H)}$, and set $\mathbf{M}_i(j) = 1$ at all spatial locations under the bounding box which we aim the tracker to output. Let $\mathbf{B}_i^t$ be such a targeted bounding box, and $\mathbf{r}_i^t$ the corresponding desired offset from the nearest anchor box. We can

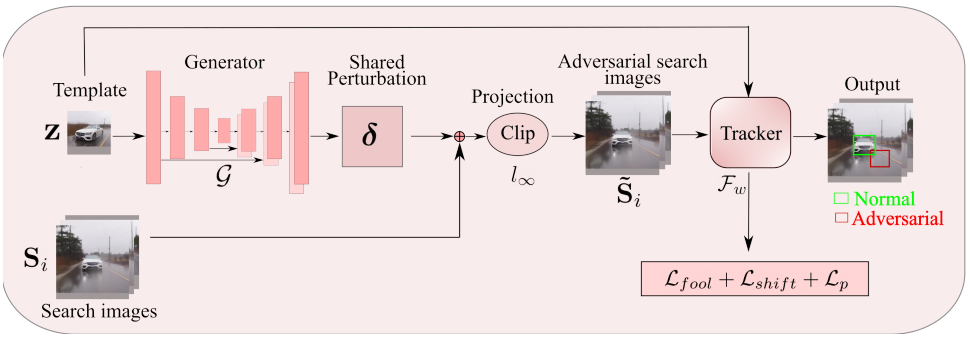


Fig. 2: Our temporally-transferable attack framework. Given the template, we generate a single temporally-transferable perturbation and add it to the search region of any subsequent frame to deviate the tracker.

then express a shift loss similar to the one in Eq. 2 but for the targeted scenario as

$$\mathcal{L}_{shift}(\mathcal{F}, \mathbf{z}, \tilde{\mathbf{S}}_i, \mathbf{M}_i) = \lambda_3 L_{cls}(\tilde{\mathbf{H}}_i(t)) + \lambda_4 L_{reg}(\tilde{\mathbf{R}}_i(t), \mathbf{r}_i^t) \ , \tag{3}$$

where, with a slight abuse of notation, t now encodes the targeted anchor.

3.2.1 Overall Loss Function.

In addition to the loss functions discussed above, we use a perceptibility loss $\mathcal{L}_p$ aiming to make the generated perturbations invisible to the naked eye. We express this loss as

$$\mathcal{L}_p = \lambda_5 \|\mathbf{S}_i - \text{Clip}_{\{\mathbf{S}_i, \epsilon\}}\{\mathbf{S}_i + \boldsymbol{\delta}\}\|_2^2 \ , \tag{4}$$

where the Clip function enforces an L_∞ bound ϵ on the perturbation. We then write the complete objective to train the generator as

$$\mathcal{L}(\mathcal{F}, \mathbf{z}, \mathbf{S}_i) = \mathcal{L}_{fool} + \mathcal{L}_{shift} + \mathcal{L}_p \ , \tag{5}$$

where $\mathcal{L}_{shift}$ corresponds to Eq. 2 in the untargeted case, and to Eq. 3 in the targeted one.

3.3 Universal Perturbations: Inference time

Once the generator is trained using the loss in Eq. 5, we can use it to generate a temporally-transferrable perturbation from the template $\mathbf{z}_t$ of any new test sequence, and use the resulting perturbation in an online-tracking phase at inference time. This by itself produces a common transferable perturbation for all frames of a given video, thereby drastically reducing the computational cost of perturbation compared to the image-dependendent perturbations in [10], [11]. Importantly, we observed that the trained generator learns to output a fixed perturbation pattern irrespective of the input template. This is attributed to the fact that our framework by design relies on exploiting key template information only, while being agnostic to the object's environment, thus forcing the generator to learn a universal adversarial function that disrupts the object-specific features in siamese-networks. Therefore, at inference time, we precompute a universal perturbation δ_u for an arbitrary input and apply it to any given test sequence to deviate the tracker from the trajectory predicted from unatttacked images. Furthermore, to force the tracker to follow complex target trajectories, such as following the ground-truth trajectory with an offset, we use precomputed universal directional perturbations for a small number, K, of predefined, diverse directions, with $K = 12$ in our experiments, and define the target trajectory as a sequence of these directions.

Relation to Prior Generative Attacks. Our proposed framework bears similarities with CSA in that both train a perturbation generator to fool a siamese tracker. However, our work differs from CSA in three fundamental ways. 1) In CSA, the perturbation is computed for every search image by passing it to the generator. By contrast, in our method, the perturbation depends only on the template and is shared across all search images. 2) In CSA, the attacks are limited to the untargeted setting, whereas our method extends to the targeted case and allows us to steer the tracker along any arbitrary trajectory. 3) By learning a perturbation shared across all search images, while being agnostic to them, our framework makes the perturbation more transferable than those of CSA, to the point of producing universal perturbations, as shown in our experiments.

4 Experiments

Datasets and Trackers. Following [10], we train our perturbation generator on GOT-10K [35] and evaluate its effectiveness on 3 short-term tracking datasets, OTB100 [36], VOT2018 [37] and UAV123 [38], and on one long-scale benchmark LaSOT [39]. We primarily use white-box SiamRPN++ (R) [12] tracker with ResNet-50 [40] backbone, and train our U-Net [41] generator. We study the transferability of attacks to 4 state-of-the-art trackers with different frameworks, namely, SiamBAN, SiamCAR, DiMP, and Ocean-online. We also transfer attacks to SiameseRPN++ (M) with MobileNet backbone, differing from the ResNet backbone of the white-box model. In contrast to SiamRPN++, which refines anchor boxes to obtain the target bounding boxes, SiamBAN and SiamCAR directly predict target bounding boxes in an anchor-free manner, avoiding careful tuning of anchor box size and aspect ratio; DiMP uses background information to learn discriminative target filters in an online fashion; Ocean-online uses a similar framework to DiMP to learn target filters in an object-aware anchor-free manner. We report the performance of our adversarial attacks using the metrics employed by each dataset to evaluate the effectiveness of unattacked trackers. In particular, we report the precision (P) and success score (S) for OTB100, UAV123, and LaSOT. For VOT2018, we report the tracker restarts (Re) and the Expected Average Overlap (EAO), a measure that considers both the accuracy (A) and robustness (R) of a tracker.

Evaluation Metrics. We report the performance of our adversarial attacks using the metrics employed by each dataset to evaluate the effectiveness of unattacked trackers. Specifically, for OTB100 and UAV123, we report the precision (P) and success score (S). The

precision encodes the proportion of frames for which the center of the tracking window is within 20 pixels of the ground-truth center. The success corresponds to the proportion of frames for which the overlap between the predicted and ground-truth tracking window is greater than a given threshold. For VOT2018, we report the Expected Average Overlap (EAO), a measure that considers both the accuracy (A) and robustness (R) of a tracker. Specifically, the accuracy denotes the average overlap, and the robustness is computed from the number of tracking failures. Furthermore, we also report the number of restarts because the standard VOT evaluation protocol reinitializes the tracker once it is too far away from the ground truth. To evaluate targeted attacks, we further report the proportion of frames in which the predicted and the target trajectory center are at a distance of at most 20 pixels.

Implementation Details. We implement our approach in PyTorch [42] and perform our experiments on an NVIDIA Telsa V100 GPU with 32GB RAM. We train the generator using pre-cropped search images uniformly sampled every 10 frames from the video sequences of GOT-10K. We use the Adam [43] optimizer with a learning rate of 2×10^{-4}. We set the margin thresholds m_c, m_w, m_h to -5 as in [10], and the l_∞ bound ϵ to $\{8, 16\}$. To fool the tracker, we use $\lambda_1 = 0.1$, $\lambda_2 = 1$, $\lambda_5 = 500$ as in csa, and activate an anchor at distance $d = 4$ directly below the true center and of size 64×64 for all untargeted experiments. Furthermore, we set the shift loss weights λ_3 and λ_4 to 0.1 and 1, respectively. For targeted attacks, we define $\mathbf{r}_i^t$ in Eq. 3 as a randomly-selected anchor at distance $d = 4$ from the true center and set its size to 64×64 for all datasets. We resize the search images to 255×255 and the template to 127×127 before passing them to the tracker.

Baselines. We compare our approach with the state-of-the-art, generator-based CSA [10] attack strategy, the only other online method performing untargeted attacks on RPN-based trackers.[1] Specifically, CSA can be employed in 3 settings: **CSA (T)**, which on attacks the template, **CSA (S)**, which attacks all search images, and **CSA (TS)**, which attacks both the template and all search regions. As will be shown below, CSA (T), the only version that, as us, generates a perturbation from only the template, is significantly less effective than CSA (S) and CSA (TS). These two versions, however, compute a perturbation for *each* search region, whereas our approach generates a single transferable perturbation from the template, and uses it at virtually no additional cost for the rest of the sequence, or even to attack other video sequences.

Computing a Universal Perturbation. During inference, we can use the object template cropped form any arbitrary sequence as input to the generator. For our experiments, without any loss of generality, we use the template cropped from the first video sequence of OTB100 to obtain a universal adversarial perturbation. Furthermore, for targeted attacks, we precompute $K = 12$ diverse universal directional perturbations with same template as input, and use them to force the tracker to follow any target trajectory for any input video sequence.

1. The attack strategy of [11] was tailored for fully convolutional and non-regression based trackers, such as SiamFC [13]. While the code for [11] is not public and authors did not respond to our communication, our own reimplementation showed that distance loss in [11] was only effective for pixel-level correlation networks, such as SiamFC, and not for regression-based trackers.

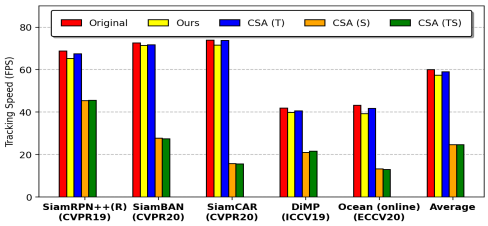


Fig. 3: Change in tracker speed. We compare the speed (FPS) of state-of-the-art trackers before and after attack.

5 Results

1. How efficient is the proposed approach at attacking modern trackers? One of the primary motivation for proposing universal transferable perturbations is to not compromise the running speed of the tracker. We therefore compare the operating speed of the tracker before and after the attack in Figure 3. Across the board, CSA (S) and CSA (TS) decrease the tracker speed significantly on average by 35 fps. For instance, SiamCAR under CSA (TS) makes the tracker operate below real-time (30 FPS) at 15.6 fps from original 71.6 fps, thus limiting its practical applicability for surveillance. While CSA (T) operates at a speed similar to our proposed approach, with a minimal speed degradation of about 1-5 fps, we significantly outperform it in terms of attack effectiveness as shown in the following sections.

2. How effective is the proposed approach at attacking modern trackers? Below, we evaluate the effectiveness of our proposed attack strategy. We denote the perturbation obtained with our complete loss as **Ours**, and refer to a variant of our method without the $\mathcal{L}_{shift}$ term as **Ours**$_f$. Furthermore, we denote the variant of our method that uses the template from the input video to compute a temporally transferable perturbation as **"TD"**, which has the same perturbation cost as CSA (T).

Results on Untargeted Attacks. From Tables 1, 2, 3, and 4, we can conclude that: (**1**) Our proposed approach consistently drops the performance of 5 black-box trackers in all settings. This highlights the generality of our approach in attacking black-box trackers with different frameworks. (**2**) Ours (TD), which uses the template from the video to compute a transferable perturbation for all search images, performs at a similar level to that of Ours, which uses a single universal transferable perturbation (see rows 6 vs 8). This validates that our trained generator is quasi-agnostic to the input template and enjoys the power of universality. (**3**) DiMP and Ocean, with online updates of discriminative filters to capture the appearance changes, are more robust to attacks on short-term datasets than other trackers. Interestingly, however, for a large-scale dataset such as LaSOT, the precision of Ocean-online and DiMP drops to 0.143 and 0.412 from the original 0.587 and 0.513, respectively. This implies that, once the tracker drifts to an incorrect position, the online updates corrupt the filters, which is especially noticeable in long video sequences. (**4**) In Table 4 on VOT2018, although CSA computes the perturbation from the new template when the tracker restarts after a failure, our universal perturbations significantly outperform CSA (TS), on average by $\sim$340 restarts. Moreover, our approach significantly decreases the EAO, which is the primary metric to rank trackers (row 4 vs 8).

Results on Targeted Attacks. Since manually creating intelligent

Methods	SiamRPN++ (M)		SiamBAN		SiamCAR		DiMP		Ocean online	
	S (↑)	P (↑)	S (↑)	P (↑)	S (↑)	P (↑)	S (↑)	P (↑)	S (↑)	P (↑)
Normal	0.657	0.862	0.692	0.910	0.696	0.908	0.650	0.847	0.669	0.884
CSA (T)	0.613	0.833	0.590	0.793	0.657	0.852	0.649	0.849	0.614	0.843
CSA (S)	0.281	0.440	0.371	0.531	0.373	0.536	0.641	0.840	0.390	0.645
CSA (TS)	0.348	0.431	0.347	0.510	0.391	0.559	0.642	0.844	0.423	0.705
Ours$_f$(TD)	0.347	0.528	0.478	0.720	0.444	0.599	0.643	0.839	0.492	0.768
Ours (TD)	**0.217**	**0.281**	**0.198**	**0.254**	**0.292**	**0.377**	**0.631**	**0.821**	**0.345**	**0.452**
Ours$_f$	0.408	0.616	0.478	0.721	0.567	0.770	0.646	0.843	0.592	0.829
Ours	**0.212**	**0.272**	**0.198**	**0.253**	**0.292**	**0.374**	0.638	0.837	**0.338**	**0.440**

TABLE 1: Untargeted attack results on OTB100 with $\epsilon = 8$.

Methods	SiamRPN++ (M)		SiamBAN		SiamCAR		DiMP		Ocean online	
	S (↑)	P (↑)	S (↑)	P (↑)	S (↑)	P (↑)	S (↑)	P (↑)	S (↑)	P (↑)
Normal	0.450	0.537	0.513	0.594	0.452	0.536	0.569	0.642	0.487	0.587
CSA (T)	0.465	0.553	0.462	0.444	0.352	0.419	0.501	0.593	0.446	0.516
CSA (S)	0.119	0.157	0.186	0.239	0.094	0.116	0.449	0.529	0.181	0.210
CSA (TS)	0.125	0.169	0.151	0.186	0.096	0.120	0.435	0.516	0.161	0.180
Ours$_f$(TD)	0.166	0.214	0.198	0.239	0.129	0.152	**0.418**	0.499	0.248	0.308
Ours (TD)	**0.114**	**0.146**	**0.095**	**0.108**	**0.079**	**0.092**	0.419	**0.487**	**0.112**	**0.128**
Ours$_f$	0.152	0.203	0.199	0.241	0.120	0.145	**0.390**	**0.461**	0.246	0.298
Ours	**0.111**	**0.146**	**0.095**	**0.109**	**0.075**	**0.089**	0.412	0.475	**0.126**	**0.143**

TABLE 2: Untargeted attack results on LaSOT with $\epsilon = 8$.

target trajectories is difficult and beyond the scope of this work, we consider two simple but practical scenarios to quantitatively analyze the effectiveness of our attacks.

1) The attacker forces the tracker to follow a fixed direction. We illustrate this with 4 different directions ($+45°$, $-45°$, $+135°$, $-135°$), and aiming to shift the box by (± 3, ± 3) pixels in each consecutive frame.
2) The attacker seeks for the tracker to follow a more complicated trajectory. To illustrate this, we force the tracker to follow the *ground-truth* trajectory with a fixed offset (± 80, ± 80).

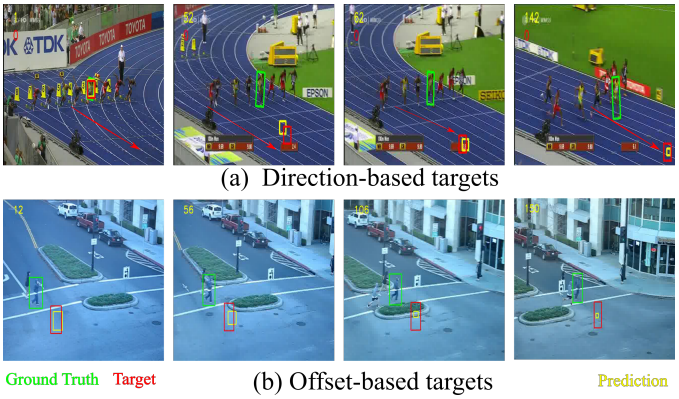
Note that our attacks are capable of steering tracker along any general trajectory, not limited to the two cases above.

In both cases, we pre-compute universal directions perturbations corresponding $K = 12$ diverse directions with template cropped from first video of OTB100, and use them to force the tracker to follow the target trajectory. To this end, we sample $K = 12$ points at a distance $d = 5$ from the object center in feature map of size 25×25 and, for each, synthesize a conditional mask $\mathbf{M}_i \in \{0, 1\}^{(W \times H)}$ whose active region is centered at the sampled point. We then feed each such mask with the template to obtain directional perturbations, which we will then transfer to the search images. During the attack for each frame, we compute the direction the tracker should move in and use the precomputed perturbation that is closest to this direction.

We report the precision score at a 20 pixel threshold for our two attack scenarios, averaged over 4 cases, in Table 5 for $\epsilon = 16$. For direction-based targets, our universal directional perturbations allow us to follow the target trajectory with promising performance. Our approach yields a precision of 0.627, 0.507, 0.536, and 0.335 on average on SiamRPN++(M), SiamBAN, SiamCAR and Ocean-online, respectively. For offset-based targets, which are more challenging than direction-based ones, our approach yields precision scores of 0.487, 0.350, 0.331 and 0.301 on average on the same 4 black-box trackers, respectively. Note that targeted attacks is quite challenging due to distractors and similar objects present in the search region. Nevertheless, our universal directional perturbations set a benchmark for image-agnostic targeted attacks on unseen

Methods	SiamRPN++ (M)		SiamBAN		SiamCAR		DiMP		Ocean online	
	S (↑)	P (↑)	S (↑)	P (↑)	S (↑)	P (↑)	S (↑)	P (↑)	S (↑)	P (↑)
Normal	0.602	0.801	0.603	0.788	0.619	0.777	0.633	0.834	0.584	0.788
CSA (T)	0.541	0.746	0.478	0.670	0.580	0.760	0.614	0.816	0.524	0.723
CSA (S)	0.288	0.466	0.299	0.485	0.270	0.440	0.593	0.798	0.264	0.489
CSA (TS)	0.270	0.452	0.278	0.487	0.271	0.428	0.598	0.811	0.278	0.510
Ours$_f$(TD)	0.369	0.561	0.372	0.569	0.337	0.503	**0.562**	**0.757**	0.404	0.648
Ours (TD)	**0.270**	**0.368**	**0.248**	**0.349**	**0.239**	**0.349**	0.573	0.770	**0.272**	**0.399**
Ours$_f$	0.356	0.549	0.372	0.569	0.316	0.469	**0.578**	**0.775**	0.392	0.634
Ours	**0.273**	**0.371**	**0.250**	**0.352**	**0.255**	**0.371**	0.579	0.777	**0.274**	**0.401**

TABLE 3: Untargeted attack results on UAV123 with $\epsilon = 8$.



(a) Direction-based targets

Ground Truth Target (b) Offset-based targets Prediction

Fig. 4: Visualizations for targeted attacks. (a) The tracker is forced to move in a constant direction, indicated by the red arrow. **(b)** The tracker is forced to follow the ground truth with a fixed offset of (80, 80) pixels.

black-box trackers. Figure 4 shows the results of targeted attacks on various datasets with SiamRPN++(M). The results at the bottom, where the tracker follows the ground-truth trajectory with an offset, illustrate the real-world applicability of our attacks, where one could force the tracker to follow a realistic, yet erroneous path. Such realistic trajectories can deceive the system without raising any suspicion.

3. What perturbation patterns does the proposed approach learn? To give insights to this question, we display the learned universal perturbations along with adversarial search regions in Figure 5. In the top row, we can see that, for untargeted attacks with the shift loss, our generator learns to place a universal object-like patch at the shift position. By contrast, the perturbation in CSA (S) is concentrated on the center region to decrease the confidence of the proposal. In the second row, we observe that, for targeted attacks, the perturbations are focused around the regions of the desired target box. This evidences that our conditioning scheme is able to capture the important information about the desired bounding box. Furthermore, as shown in the bottom row, our results remain imperceptible thanks to our similarity loss.

Comparison to Iterative Black-box attacks. We compare our approach with the state-of-the-art black-box IoU-based attack [32] in Tables 8 and 9. This IoU attack requires access to the tracker predictions. As such, it spends a significant query budget for each frame, thereby decreasing the tracker speed to less than 5 FPS. By contrast, our method learns a highly transferable universal perturbation on a substitute SiameseRPN++(R) tracker, and thus significantly outperforms the IoU attack with virtually no drop in tracker speed. Note that we could not run the IoU attack on large-scale datasets such as UAV123 and LaSOT because of impractical

Method	SiamRPN++(M)				SiamBAN				SiamCAR				DiMP				Ocean online			
	A (↑)	R (↓)	EAO (↑)	Re (↓)	A (↑)	R (↓)	EAO (↑)	Re (↓)	A (↑)	R (↓)	EAO(↑)	Re (↓)	A (↑)	R (↓)	EAO(↑)	Re (↓)	A (↑)	R (↓)	EAO (↑)	Re (↓)
Original	0.58	0.24	0.400	51	0.60	0.320	0.340	69	0.58	0.280	0.36	60	0.607	0.3000	0.323	64	0.56	0.220	0.374	47
CSA (T)	0.56	0.440	0.265	95	0.56	0.590	0.190	126	0.56	0.426	0.280	91	0.60	0.220	0.362	47	0.45	0.580	0.189	124
CSA (S)	0.42	2.205	0.067	471	0.43	1.807	0.076	386	0.48	1.597	0.101	341	0.59	0.239	0.367	51	0.20	1.462	0.083	202
CSA (TS)	0.40	2.196	0.067	469	0.38	1.789	0.075	382	0.45	1.475	0.107	315	0.58	0.286	0.322	61	0.22	1.221	0.082	261
Ours$_f$(TD)	0.48	1.625	0.089	347	0.48	1.508	0.079	322	0.52	1.569	0.096	335	0.60	0.26	0.337	56	0.47	0.445	0.232	95
Ours (TD)	0.51	**5.095**	**0.029**	**1088**	0.44	**5.071**	**0.024**	**1083**	0.60	**3.341**	**0.053**	**712**	0.59	**0.512**	**0.219**	**109**	0.38	**1.621**	**0.074**	**346**
Ours$_f$	0.45	2.098	0.070	448	0.46	1.915	0.070	409	0.51	1.842	0.091	393	0.59	0.267	0.350	57	0.42	0.515	0.209	110
Ours	0.52	**4.856**	**0.029**	**1037**	0.44	**5.034**	**0.024**	**1075**	0.59	**3.184**	**0.056**	**680**	0.56	**0.445**	**0.235**	**95**	0.39	**1.482**	**0.081**	**316**

TABLE 4: Untargeted attack results on VOT2018 with $\epsilon = 8$.

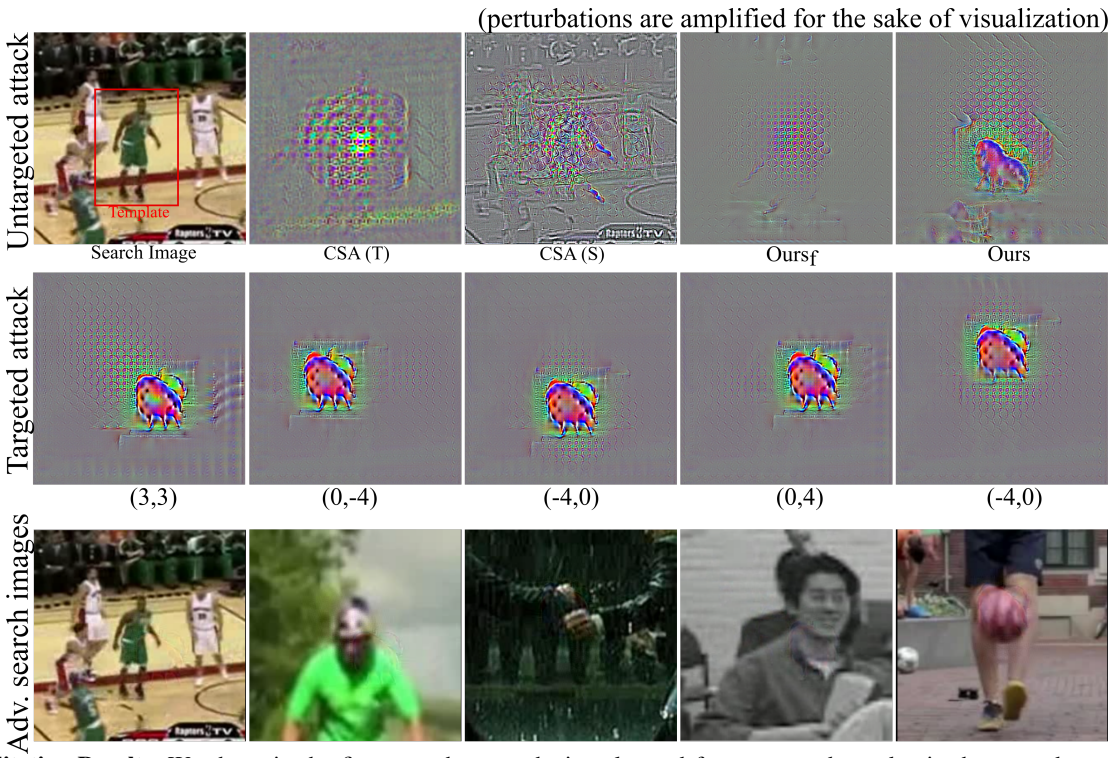


Fig. 5: Qualitative Results. We show, in the first row, the perturbations learned for untargeted attacks; in the second row, the universal directional perturbations for the targeted attack; in the last row, the adversarial search regions obtained with our targeted attack framework for $\epsilon = 16$.

Dataset	SiamRPN++ (M)		SiamBAN		SiamCAR		Ocean online	
	Direction	Offset	Direction	Offset	Direction	Offset	Direction	Offset
OTB100	0.521	0.544	0.345	0.257	0.340	0.295	0.128	0.113
VOT2018	0.745	0.515	0.672	0.455	0.661	0.501	0.412	0.372
UAV123	0.476	0.401	0.325	0.295	0.397	0.297	0.260	0.267
LaSOT	0.768	0.489	0.689	0.395	0.747	0.232	0.543	0.521
Average	0.627	0.487	0.507	0.350	0.536	0.331	0.335	0.301

TABLE 5: Targeted attack results. We report average precision scores for $\epsilon = 16$ for direction and offset-based target trajectories.

$\mathcal{L}_{fool}^{cls}$	$\mathcal{L}_{fool}^{reg}$	$\mathcal{L}_{shift}^{cls}$	$\mathcal{L}_{shift}^{reg}$	SiamRPN++ (M)	SiamBAN	SiamCAR	DiMP	Ocean online
-	-	-	-	0.862	0.910	0.908	0.847	0.884
✓	-	-	-	0.566	0.604	0.654	0.851	0.801
-	✓	-	-	0.617	0.726	0.790	0.841	0.827
-	-	✓	-	0.790	0.800	0.851	0.858	0.805
-	-	-	✓	0.858	0.890	0.884	0.858	0.879
✓	✓	-	-	0.616	0.721	0.770	0.843	0.829
-	-	✓	✓	0.695	0.735	0.734	0.828	0.750
✓	-	✓	-	0.328	0.316	0.531	0.827	0.592
-	✓	-	✓	0.682	0.769	0.826	0.848	0.852
✓	✓	✓	✓	0.272	0.252	0.374	0.837	0.440

TABLE 6: Component-wise analysis. Contribution of each loss for untargeted attacks on OTB100 using our approach. We report precision score and set $\epsilon = 8$.

runtimes.

6 Ablation Studies

In this section, we analyze the impact of each loss term of our framework. In Table 6, we report the precision scores on OTB100 with different combination of loss terms, where $\mathcal{L}_{fool}^{cls}$ and $\mathcal{L}_{fool}^{reg}$ represent the classification and regression components of the fooling loss of Eq. 1, and $\mathcal{L}_{shift}^{cls}$ and $\mathcal{L}_{shift}^{reg}$ represent the same terms for the shift loss of Eq. 2. To summarize, while all loss terms are beneficial, the classification-based terms are more effective than regression-based ones. For example, using either $\mathcal{L}_{fool}^{cls}$ or $\mathcal{L}_{shift}^{cls}$ has more impact than $\mathcal{L}_{fool}^{reg}$ or $\mathcal{L}_{shift}^{reg}$. In Table 7, we study the impact of the shift distance d in Eq. 2 on the performance of untargeted attacks. For a feature map of size 25×25 for SiamRPN++, the performance of our approach is stable for a drift in the range 4 to 8. However, for $d = 2$, our attacks have less effect on the tracker, and for $d = 10$, the influence of the attack decreases because of the Gaussian prior used by the tracker.

Effect of the number K of Directional Perturbations. In the

Shift d	SiamRPN++ (M)	SiamBAN	SiamCAR	DiMP	Ocean online
0	0.616	0.721	0.770	0.843	0.829
2	0.332	0.254	0.551	**0.814**	0.493
4	**0.272**	**0.252**	**0.374**	0.837	**0.440**
6	0.330	0.409	0.481	0.844	0.618
8	0.323	0.489	0.480	0.834	0.684
10	0.427	0.510	0.612	0.851	0.763

TABLE 7: Ablation study. Effect of d in $\mathcal{L}_{shift}$ for untargeted attacks on OTB100. We report precision score and set $\epsilon = 8$.

Methods	SiamBAN			SiamCAR			SiamRPN++(M)		
	FPS(↑)	S (↑)	P (↑)	FPS(↑)	S (↑)	P (↑)	FPS(↑)	S (↑)	P (↑)
Normal	72.3	0.692	0.910	71.2	0.696	0.908	94	0.657	0.862
IoU	5.6	0.513	0.682	4.3	0.595	0.778	4.5	0.505	0.670
Ours	**71.7**	**0.198**	**0.253**	**70.2**	**0.292**	**0.374**	**90**	**0.212**	**0.272**

TABLE 8: Comparison with black-box IoU attack on OTB100. We report untargeted attack results with $\epsilon = 8$.

targeted attack experiments, we typically precompute $K = 12$ diverse directional perturbations. For each one, we activate a target bounding boxes at a radius 4 from the center of 25×25 feature map of SiameseRPN++ [12]. In Table 10, we vary this parameter K and report the precision score for offset-based targeted attacks with $\epsilon = 16$. We observe that with minimum value of $K = 4$ encoding 4 directions yields lower precision scores than $K = 12$ for both datasets. However, as K increases from 4 to 12, the precision scores increase and, beyond that, either saturate or marginally decrease.

6.1 Effect of Hyperparameters

Untargeted Attacks. For all experiments in the main paper, we set $\lambda_1 = 0.1$ and $\lambda_2 = 1$ as in [10]. Furthermore, we set the weights of the additional terms λ_3 and λ_4 to the values of λ_1 and λ_2, respectively. This is motivated by the fact that our additional loss components perform similar task to those weighted by λ_1 and λ_2. We name the configuration with the above-mentioned values as the default configuration. We then independently vary each parameter while fixing the remaining ones to the default values on SiamesRPN++ [12] with $\epsilon = 8$.

As shown in Table 11, varying λ_1 and λ_3, encoding the classification loss terms, evidences that the default value of 0.1 performs the best for both the OTB100 [36]. Furthermore, Table 12 shows that varying λ_2 and λ_4, associated with the regression terms, highlights that the results are stable for values in the range of 0.001 to 1. Nevertheless, the default value of 1 for λ_4 indeed yields the best results. Besides, $\lambda_2 = 10$ performs marginally better than the default setting of $\lambda_2 = 1$.

Targeted Attacks. We perform a similar ablation study for targeted attacks on Siamese RPN++(M) [12] black-box tracker with $\epsilon = 16$. We report the results in Tables 13, 15, 14, and 16, corresponding to varying λ_1, λ_2, λ_3 and λ_4, respectively, on OTB100. In Table 13, we observe that setting λ_1 in the range of 0.1 to 1 yields the best results. In addition, we observe from Table 14 that $\lambda_3 = 1$ performs better than the default value of 0.1. Moreover, from Tables 15, 16, we find that $\lambda_2 = \lambda_4 = 10$ yields better precision scores than the default value of 0.1.

7 Additional Qualitative Results

We provide additional qualitative results for different attack settings. In Figure 6, we show the tracking outputs with offset-based targets on SiamRPN++(M) [12]. We observe that the universal directional perturbations are concentrated along the targeted di-

Methods	SiamBAN			SiamCAR			SiamRPN++(M)		
	FPS(↑)	EAO (↑)	Re (↓)	FPS(↑)	EAO (↑)	Re (↓)	FPS(↑)	EAO (↑)	Re (↓)
Normal	72.3	0.340	69	71.2	0.36	60	94.3	0.40	51
IoU	1.62	0.114	269	2.45	0.189	165	3.53	0.117	289
Ours	**71.7**	**0.024**	**1075**	**70.2**	**0.056**	**686**	**90.4**	**0.029**	**1037**

TABLE 9: Comparison with black-box IoU attack on VOT2018. We report untargeted attack results with $\epsilon = 8$.

λ_4	SiamRPN++ (M)	SiamBAN	SiamCAR
23	0.393	0.256	0.266
12	**0.544**	**0.257**	**0.295**
8	0.308	0.181	0.186
4	0.273	0.155	0.173

TABLE 10: Impact of the number of directional perturbations K for 3 black-box trackers with offset-based targeted attacks on OTB100. We report precision scores averaged over four cases for $\epsilon = 16$ and $(\Delta_x, \Delta_y) = (80, 80)$.

λ_1	SiamRPN++(M)		SiamBAN		SiamCAR		Ocean-online	
	S (↑)	P(↑)	S (↑)	P(↑)	S (↑)	P(↑)	S (↑)	P(↑)
0.001	0.536	0.711	0.562	0.762	0.618	0.817	0.592	0.783
0.01	0.352	0.472	0.322	0.421	0.491	0.641	0.503	0.672
0.1	**0.212**	**0.272**	**0.198**	**0.253**	**0.292**	**0.374**	**0.338**	**0.440**
1	0.321	0.451	0.272	0.352	0.342	0.44	0.401	0.531
10	0.352	0.501	0.363	0.501	0.413	0.572	0.521	0.721
100	0.321	0.482	0.395	0.556	0.392	0.547	0.538	0.774

(a) Varying λ_1

λ_3	SiamRPN++(M)		SiamBAN		SiamCAR		Ocean-online	
	S (↑)	P(↑)	S (↑)	P(↑)	S (↑)	P(↑)	S (↑)	P(↑)
0.001	0.307	0.433	0.325	0.444	0.423	0.571	0.452	0.662
0.01	0.031	0.412	0.261	0.332	0.351	0.462	0.401	0.541
0.1	**0.212**	**0.272**	**0.198**	**0.253**	**0.292**	**0.374**	**0.338**	**0.440**
1	0.471	0.642	0.38	0.491	0.562	0.742	0.502	0.641
10	0.442	0.601	0.38	0.501	0.482	0.623	0.501	0.602
100	0.471	0.632	0.443	0.579	0.543	0.718	0.488	0.638

(b) Varying λ_3

TABLE 11: Impact of varying λ_1 and λ_3 (corresponding to the classification losses $\mathcal{L}^{cls}_{fool}$ and $\mathcal{L}^{cls}_{shift}$) independently for untargeted attacks on OTB100 [36]. Setting λ_1, λ_3 to 0.1 performs consistently well on all black-box trackers.

λ_2	SiamRPN++(M)		SiamBAN		SiamCAR		Ocean-online	
	S (↑)	P(↑)	S (↑)	P(↑)	S (↑)	P(↑)	S (↑)	P(↑)
0.001	0.317	0.411	0.218	0.268	0.440	0.568	0.460	0.606
0.01	0.378	0.497	0.255	0.322	0.427	0.552	0.464	0.604
0.1	0.371	0.482	0.254	0.324	0.431	0.554	0.451	0.589
1	**0.212**	**0.272**	**0.198**	**0.253**	**0.292**	**0.374**	**0.338**	**0.440**
10	**0.162**	**0.207**	**0.158**	**0.195**	**0.304**	**0.398**	**0.325**	**0.420**
100	0.257	0.409	0.313	0.465	0.300	0.427	0.375	0.641

(a) Varying λ_2

λ_4	SiamRPN++(M)		SiamBAN		SiamCAR		Ocean-online	
	S (↑)	P(↑)	S (↑)	P(↑)	S (↑)	P(↑)	S (↑)	P(↑)
0.001	0.233	0.304	0.206	0.268	0.341	0.441	0.411	0.558
0.01	0.261	0.341	0.211	0.262	0.362	0.461	0.423	0.561
0.1	0.261	0.350	0.232	0.291	0.351	0.462	0.43	0.572
1	**0.212**	**0.272**	**0.198**	**0.253**	**0.292**	**0.374**	**0.338**	**0.440**
10	0.421	0.571	0.32	0.412	0.532	0.691	0.501	0.661
100	0.452	0.626	0.363	0.472	0.514	0.678	0.482	0.658

(b) Varying λ_4

TABLE 12: Impact of varying λ_2 and λ_4 (corresponding to the classification losses $\mathcal{L}^{cls}_{fool}$ and $\mathcal{L}^{cls}_{shift}$) independently for untargeted attacks on OTB100 [36]. Setting λ_2, λ_4 to 1 performs consistently well on all black-box trackers.

λ_1	SiamRPN++ (M)	SiamBAN	SiamCAR
0.001	0.233	0.141	0.096
0.01	0.282	0.185	0.122
0.1	**0.544**	0.257	0.295
1	0.478	**0.365**	**0.346**
10	0.129	0.185	0.326
100	0.007	0.007	0.017

TABLE 13: Impact of varying λ_1 for offset-based targeted attacks on 3 black-box trackers on OTB100. We report precision scores for $\epsilon = 16$ and $(\Delta_x, \Delta_y) = (80, 80)$.

λ_3	SiamRPN++ (M)	SiamBAN	SiamCAR
0.001	0.161	0.141	0.126
0.01	0.327	0.243	0.251
0.1	**0.544**	0.257	**0.295**
1	0.446	**0.334**	0.254
10	0.313	0.288	0.282
100	0.326	0.258	0.113

TABLE 14: Impact of varying λ_3 for offset-based targeted attacks on 3 black-box trackers on OTB100. We report precision scores for $\epsilon = 16$ and $(\Delta_x, \Delta_y) = (80, 80)$.

λ_2	SiamRPN++ (M)	SiamBAN	SiamCAR
0.001	0.398	0.294	0.240
0.01	0.380	0.283	0.208
0.1	0.394	0.311	0.270
1	**0.544**	0.257	0.295
10	0.535	**0.383**	**.485**
100	0.009	0.008	0.005

TABLE 15: Impact of varying λ_2 for offset-based targeted attacks on 3 black-box trackers on OTB100. We report precision scores for $\epsilon = 16$ and $(\Delta_x, \Delta_y) = (80, 80)$.

λ_4	SiamRPN++ (M)	SiamBAN	SiamCAR
0.001	0.479	0.339	0.359
0.01	0.457	0.357	0.321
0.1	0.428	0.301	0.267
1	**0.544**	0.257	0.295
10	0.551	**0.343**	**0.404**
100	0.136	0.179	0.106

TABLE 16: Impact of varying λ_4 for offset-based targeted attacks on 3 black-box trackers on OTB100. We report precision scores for $\epsilon = 16$ and $(\Delta_x, \Delta_y) = (80, 80)$.

rection and typically align with the target bounding box region. Furthermore, in Figures 7 and 8, we visualize the outputs for untargeted attacks without and with shift loss. Our results indicate that the generated perturbation concentrate around the center region and fool the tracker within the first few frames.

8 Conclusion

We have shown the existence of transferable universal perturbations to efficiently attack black-box VOT trackers on the fly. To do so, we have introduced a framework that relies on generating a one-shot temporally-transferable perturbation by exploiting only the template as input, thus being invariant to the search environment. Our trained generator produces perturbations that are quasi-agnostic to the input template, and are thus highly transferable to unknown objects. Furthermore, we have demonstrated that our universal directional perturbations allow us to steer the tracker to follow any specified trajectory. We believe that our work highlights the vulnerability of object trackers and will motivate researchers to design robust defense mechanisms.

Acknowledgments. This work was funded in part by the Swiss National Science Foundation.

References

[1] K.-H. Lee and J.-N. Hwang, "On-road pedestrian tracking across multiple driving recorders," *IEEE Transactions on Multimedia*, vol. 17, no. 9, pp. 1429–1438, 2015.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

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

[4] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1369–1378.

[5] A. Arnab, O. Miksik, and P. H. Torr, "On the robustness of semantic segmentation models to adversarial attacks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 888–897.

[6] V. Fischer, M. C. Kumar, J. H. Metzen, and T. Brox, "Adversarial examples for semantic image segmentation," *arXiv preprint arXiv:1703.01101*, 2017.

[7] Q. Guo, X. Xie, F. Juefei-Xu, L. Ma, Z. Li, W. Xue, W. Feng, and Y. Liu, "Spark: Spatial-aware online incremental attack against visual tracking," *arXiv preprint arXiv:1910.08681*, 2019.

[8] X. Chen, X. Yan, F. Zheng, Y. Jiang, S.-T. Xia, Y. Zhao, and R. Ji, "One-shot adversarial attacks on visual tracking with dual attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 176–10 185.

[9] Y. Jia, Y. Lu, J. Shen, Q. A. Chen, Z. Zhong, and T. Wei, "Fooling detection alone is not enough: First adversarial attack against multiple object tracking," *arXiv preprint arXiv:1905.11026*, 2019.

[10] B. Yan, D. Wang, H. Lu, and X. Yang, "Cooling-shrinking attack: Blinding the tracker with imperceptible noises," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 990–999.

[11] S. Liang, X. Wei, S. Yao, and X. Cao, "Efficient adversarial attacks for visual object tracking," *arXiv preprint arXiv:2008.00217*, 2020.

[12] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4282–4291.

[13] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European conference on computer vision*. Springer, 2016, pp. 850–865.

[14] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8971–8980.

[15] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr, "Fast online object tracking and segmentation: A unifying approach," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 1328–1338.

[16] O. Poursaeed, I. Katsman, B. Gao, and S. Belongie, "Generative adversarial perturbations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4422–4431.

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

[18] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 2544–2550.

[19] H. Kiani Galoogahi, T. Sim, and S. Lucey, "Multi-channel correlation filters," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 3072–3079.

[20] ——, "Correlation filters with limited boundaries," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4630–4638.

[21] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4310–4318.
[22] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 101–117.
[23] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.
[24] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6668–6677.
[25] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen, "Siamcar: Siamese fully convolutional classification and regression for visual tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6269–6277.
[26] G. Bhat, M. Danelljan, L. V. Gool, and R. Timofte, "Learning discriminative model prediction for tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6182–6191.
[27] Z. Zhang, H. Peng, J. Fu, B. Li, and W. Hu, "Ocean: Object-aware anchor-free tracking," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16*. Springer, 2020, pp. 771–787.
[28] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582.
[29] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9185–9193.
[30] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*. IEEE, 2017, pp. 39–57.
[31] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.
[32] S. Jia, Y. Song, C. Ma, and X. Yang, "Iou attack: Towards temporally coherent black-box adversarial attack for visual object tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6709–6718.
[33] R. R. Wiyatno and A. Xu, "Physical adversarial textures that fool visual object tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4822–4831.
[34] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1765–1773.
[35] L. Huang, X. Zhao, and K. Huang, "Got-10k: A large high-diversity benchmark for generic object tracking in the wild," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.
[36] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 2411–2418.
[37] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey *et al.*, "The sixth visual object tracking vot2018 challenge results," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 0–0.
[38] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *European conference on computer vision*. Springer, 2016, pp. 445–461.
[39] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling, "Lasot: A high-quality benchmark for large-scale single object tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5374–5383.
[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[41] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
[42] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.
[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Krishna Kanth Nakka is a Ph.D. student at EPFL. His research interests are understanding the vulnerabilities of deep neural networks and endeavors to develop robust and interpretable models. He received his Bachelor's and Master's degree in Electrical Engineering from the Indian Institute of Technology, Kharagpur in 2015.

Mathieu Salzmann is a Senior Researcher at EPFL and an Artificial Intelligence Engineer at ClearSpace. Previously, he was a Senior Researcher and Research Leader in NICTA's computer vision research group, a Research Assistant Professor at TTI-Chicago, and a postdoctoral fellow at ICSI and EECS at UC Berkeley. He obtained his PhD in 2009 from EPFL. His research interests lie at the intersection of machine learning and computer vision.

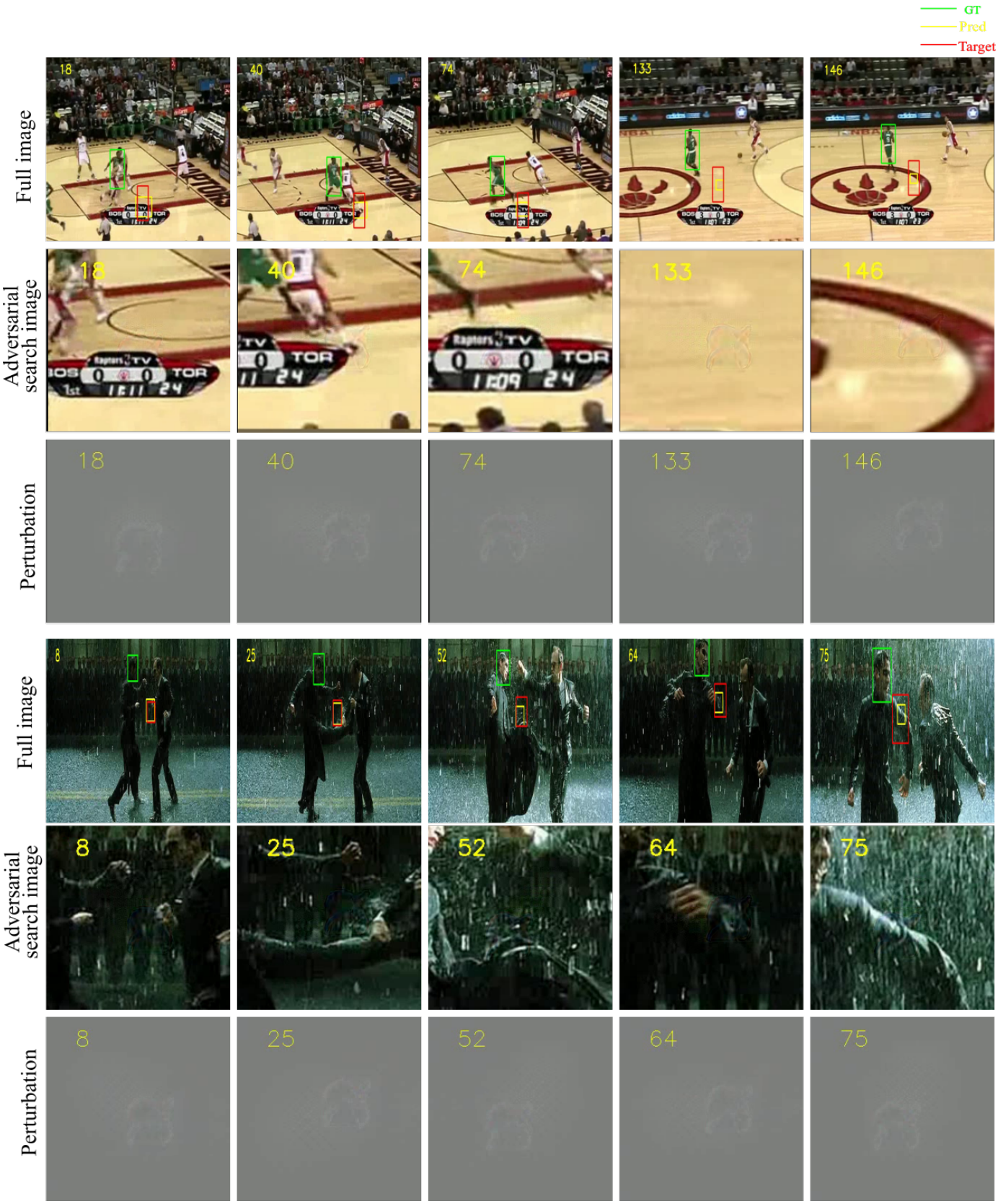


Fig. 6: Qualitative results for offset-based targeted attacks on the SiameseRPN++(M) [12] tracker to follow the ground-truth with a fixed offset of 80 pixels with $\epsilon = 16$. We visualize the tracking outputs along with the adversarial search images and the directional perturbations. Green represents the ground-truth bounding box, red represents the target bounding box, and yellow represents the predicted bounding box. We observe that the perturbation is concentrated along the direction of the target bounding box.

Fig. 7: Qualitative results for untargeted attacks with our approach (Ours) on the SiameseRPN++(M) [12] tracker with $\epsilon = 16$. We visualize the tracking outputs, the adversarial search regions and the single temporally transferable directional perturbation computed from the object template. Green represents the ground-truth bounding box, and yellow represents the predicted bounding box. We observe that the perturbation is concentrated around the center region and effective to fool the tracker for the entire video sequence.

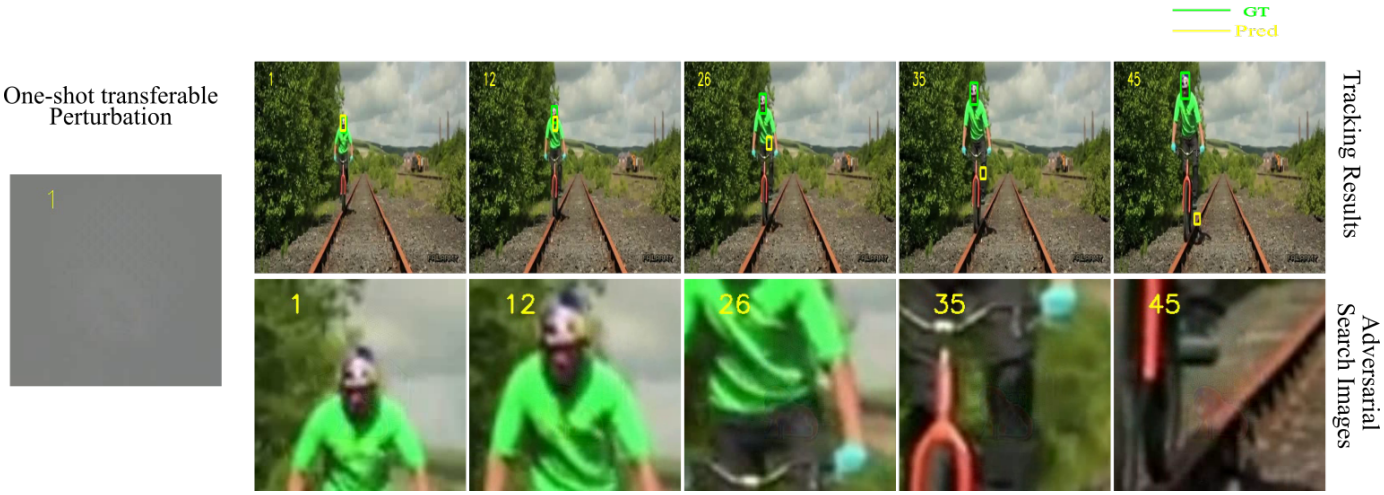


Fig. 8: Qualitative results for untargeted attacks with our approach(Ours$_{shift}$) on the SiameseRPN++ [12](M) tracker with $\epsilon = 8$. We visualize the tracking outputs, the adversarial search regions and the single temporally transferable directional perturbation computed from the object template. Green represents the ground-truth bounding box, and yellow represents the predicted bounding box. We observe that the perturbation is concentrated around the center region and effective to fool the tracker within a few frames.