Temporal Unlearnable Examples: Preventing Personal Video Data from Unauthorized Exploitation by Object Tracking

Qiangqiang Wu[1]* Yi Yu[2]* Chenqi Kong[2]† Ziquan Liu[3] Jia Wan[4]
Haoliang Li[1] Alex C. Kot[2] Antoni B. Chan[1]
[1]Department of Computer Science, City University of Hong Kong
[2]ROSE Lab, Nanyang Technological University
[3]Queen Mary University of London [4]Harbin Institute of Technology, Shenzhen
qiangqwu2-c@my.cityu.edu.hk, {yuyi0010, chenqi.kong, eackot}@ntu.edu.sg,
ziquan.liu@qmul.ac.uk, jiawan1998@gmail.com, {haoliang.li, abchan}@cityu.edu.hk

Abstract

With the rise of social media, vast amounts of user-uploaded videos (e.g., YouTube) are utilized as training data for Visual Object Tracking (VOT). However, the VOT community has largely overlooked video data-privacy issues, as many private videos have been collected and used for training commercial models without authorization. To alleviate these issues, this paper presents the first investigation on preventing personal video data from unauthorized exploitation by deep trackers. Existing methods for preventing unauthorized data use primarily focus on image-based tasks (e.g., image classification), directly applying them to videos reveals several limitations, including inefficiency, limited effectiveness, and poor generalizability. To address these issues, we propose a novel generative framework for generating Temporal Unlearnable Examples (TUEs), and whose efficient computation makes it scalable for usage on large-scale video datasets. The trackers trained w/ TUEs heavily rely on unlearnable noises for temporal matching, ignoring the original data structure and thus ensuring training video data-privacy. To enhance the effectiveness of TUEs, we introduce a temporal contrastive loss, which further corrupts the learning of existing trackers when using our TUEs for training. Extensive experiments demonstrate that our approach achieves state-of-the-art performance in video data-privacy protection, with strong transferability across VOT models, datasets, and temporal matching tasks.

1 Introduction

Visual Object Tracking (VOT) estimates target bounding boxes in each video frame based on the initial target state, playing a key role in applications like intelligent surveillance and autonomous driving. Recent advances in deep trackers, driven by large-scale training videos from the internet, have significantly improved VOT performance. However, concerns over unauthorized data exploitation by Deep Neural Networks (DNNs) are growing. For instance, personal videos uploaded to social media (*e.g.,* YouTube) may be used for VOT training without consent, raising privacy and copyright issues. Large-scale VOT datasets like TrackingNet [51], LaSOT [19, 20], and GOT-10k [32] primarily consist of such user-uploaded videos. Protecting sensitive trajectories—such as those of individuals, vehicles, and military assets—requires preventing unauthorized use of tracking data. Hence, safeguarding video data from exploitation in VOT training is essential.

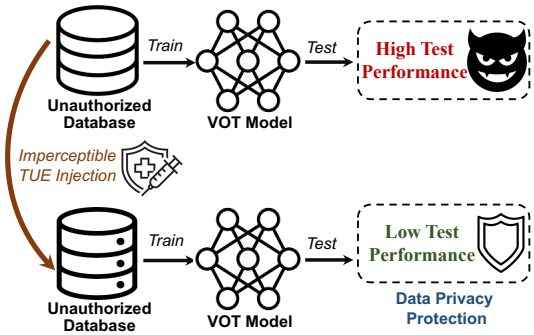

Figure 1. Illustration of our TUEs for preventing video data from unauthorized exploitation by deep VOT models. Adding imperceptible TUEs to training videos limits deep VOT models to only learning shortcuts information, resulting in poor generalization and degraded test performance.

In the context of 2D images, Unlearnable Examples (UEs) [23, 31, 86, 89] is a typical solution to protect private data from unauthorized exploitation by DNNs. UEs

*Equal contribution. †Corresponding author.

methods add imperceptible perturbations (*i.e.,* bounded noises) to the training images to hinder models from extracting useful information from the poisoned training data, thereby resulting in poor testing performance. Extending image-task UEs to videos remains underexplored. While uniform perturbations across frames may effectively adapt image-task UEs for video classification tasks like action recognition, they struggle with temporal matching tasks such as VOT and Video Object Segmentation (VOS). Our paper focuses on learning UEs for temporal matching, with VOT as a foundational task. Protecting VOT is crucial for safeguarding sensitive trajectories (e.g., missiles, vehicles, and individuals). Optimizing UEs for the basic VOT task could also improve their transferability to advanced temporal matching tasks such as VOS and long-term point tracking.

Extending image-task UEs to temporal matching tasks presents several challenges: (1) Video data have higher resolution and more frames, making UEs generation computationally intensive; (2) Unlike image classification, VOT relies on temporal matching across frames, with target objects changing in scale, complicating the design of UEs that support scale-invariant matching; (3) Existing UEs struggle to transfer across different tracking models, datasets, and matching tasks, limiting their effectiveness for data privacy in VOT.

To address the above challenges, we introduce Temporal Unlearnable Examples (TUEs) for video data (see Fig. 1). By injecting imperceptible TUEs, deep VOT models learn only limited information from the training set, resulting in poor generalization and degraded test performance. We propose a novel generative framework in which a generator is trained to produce TUEs that disrupt temporal matching in VOT models. These perturbations remain imperceptible to the human eye and do not compromise the data utility for human consumption. Compared to the traditional iterative EM approach [31] used for image UEs, our framework significantly improves efficiency—achieving 4× faster training speed and 28× greater parameter efficiency on the GOT-10k [32] dataset—making it highly scalable for large video datasets. Additionally, we introduce a temporal contrastive loss (TCL) to encourage trackers to rely more on the generated TUEs for temporal matching, thereby further enhancing the privacy protection.

We assess the transferability of our TUEs through experiments across various VOT models and datasets. TUEs trained on the simple SiamFC [4] tracker can effectively degrade the performance of state-of-the-art deep trackers with complex architectures, such as ViT [18, 92] and ResNet [27]. Moreover, a generator trained on a source dataset like GOT-10k [32] can be used for zero-shot TUE generation on unseen video datasets, without retraining. Finally, we show that our TUEs are task-transferable and perform well in other temporal matching tasks, such as video object segmentation.

In summary, the main contributions of our work are:

- To the best of our knowledge, we are the first to investigate preventing unauthorized video exploitation for VOT. Since none exists, we apply off-the-shelf image-task UEs to videos as baselines for VOT with specific designs, which reveal several limitations, *e.g.,* inefficiency, limited effectiveness, and poor generalizability.
- We propose a new generative framework to generate Temporal Unlearnable Examples (TUEs), which can effectively corrupt the temporal matching learning of VOT models. Our method achieves state-of-the-art performance in video data privacy protection and shows strong transferability across various trackers, datasets, and matching tasks.
- We introduce a Temporal Contrastive Loss (TCL) that encourages trackers to rely more on the generated TUEs for temporal matching learning, which further degrades the tracking performance while preserving the data privacy of training data.

2 Related Work

Visual Object Tracking (VOT) predicts target bounding boxes in each frame based on the initial target state. Early correlation-filter (CF) trackers [3, 14, 17, 24, 28, 48] were successful due to their performance and speed. With the rise of deep learning [27, 35], CF trackers [13, 15, 16, 42, 46, 63, 64, 69, 70] began incorporating deep features for VOT. SiamFC [4] and SINT [62] introduced deep Siamese networks for end-to-end VOT, leading to improvements in transformer tracking [12, 77, 83, 84], online memory design [82, 91], architecture design [8, 25, 29, 36, 37, 72], and new learning paradigms [2, 5, 9, 67, 68, 71, 76]. These trackers rely on large-scale datasets like GOT-10k [32] and LaSOT [19, 20], often sourced from user-uploaded social media videos. However, privacy concerns in VOT remain largely overlooked and demand urgent attention. Recently, several adversarial attacks have targeted deep trackers [7, 26, 33, 41, 61, 74, 81], such as generating temporally transferable perturbations [52] and introducing adversarial loss to reduce heatmap hot regions [79]. Additionally, VOT models' vulnerability to backdoor attacks has been highlighted [30, 40]. However, these methods often require access to both training and testing data or model parameters, which can be impractical. In contrast, our approach focuses on protecting training video data from unauthorized use, requiring access only to the training data, making it more feasible.

Unlearnable Examples (UEs). Data privacy in 2D images has been widely studied, with traditional methods focusing on preventing models from leaking sensitive information [1, 43, 45, 54, 57–59]. UEs [21, 31, 38,

39, 44, 50, 87] have recently emerged, where bounded perturbations (*e.g.,* $\|\boldsymbol{\delta}\|_\infty \leq \frac{8}{255}$) are added to training data, preserving labels while degrading model performance. This perturbative poisoning method [47] shows promise for data protection, causing models trained on such data to perform near-randomly on clean test data. Techniques like EM [31], NTGA [90], TAP [22], REM [23], LSP [85], and OPS [73] offer various strategies for generating protective perturbations. While UEs have been explored in tasks like image classification, segmentation, and point cloud classification [60, 66], their application in tasks like VOT matching remains unexplored. We aim to bridge this gap by exploring effective UEs for template matching across video frames.

3 Methodology

We first review the preliminaries in §3.1, and introduce UE baselines in §3.2. To overcome UE's limitations, we propose a generative framework for Temporal Unlearnable Examples (TUEs) in §3.3. We further present Temporal Contrastive Learning to enhance TUEs in §3.4. Finally, we summarize the overall pipeline in §3.5.

3.1 Preliminaries

We briefly introduce traditional UEs algorithms and the temporal matching pipeline in VOT.

Error-Minimizing (EM) Learning. Given a clean image dataset $\mathcal{D}_c = \{(\mathbf{I}_i, y_i)\}_{i=1}^m$ with m clean images, EM [31] aims to learn sample-wise UEs on $\mathcal{D}_c$ that will degrade a classifier F_θ on the testing set:

$$\min_\theta \mathbb{E}_{(\mathbf{I},y)\sim\mathcal{D}_c}\big[\min_{\boldsymbol{\delta}\in\boldsymbol{\Delta}} \mathcal{L}_c(F_\theta(\mathbf{I}+\boldsymbol{\delta}), y)\big], \tag{1}$$

where $\mathcal{L}_c$ is the cross-entropy loss, y is the class label of $\mathbf{I}\in\mathbb{R}^{H\times W\times C}$, $\boldsymbol{\delta}\in\boldsymbol{\Delta}\subset\mathbb{R}^{H\times W\times C}$ is the imperceptible noises, and $\boldsymbol{\Delta}$ is the feasible region. Typically, the noise $\boldsymbol{\delta}$ is L_p-norm bounded, *i.e.,* $\boldsymbol{\Delta}=\{\boldsymbol{\delta}|\,||\boldsymbol{\delta}||_p\leq\sigma\}$, where σ is small such that the noise is imperceptible. The goal is for the noise $\boldsymbol{\delta}$ to make the original image $\mathbf{I}$ unlearnable, thus $\hat{\mathbf{I}} = \mathbf{I}+\boldsymbol{\delta}$ is denoted as the UEs. The formulation in (1) involves both inner and outer optimization processes. The outer optimization fixes the perturbations and updates the parameters θ of the classifier F_θ by minimizing the loss $\mathcal{L}_c$. In the inner optimization, the classifier is fixed while the perturbations $\boldsymbol{\delta}$ are updated to also minimize the loss (making each training sample "easier" for the classifier). These optimizations are performed alternately during training. In each inner optimization, EM uses Projected Gradient Descent (PGD) [49] to iteratively update the perturbations $\boldsymbol{\delta}$ over T steps. Note that there is a different perturbation $\boldsymbol{\delta}$ optimized for each sampled $\mathbf{I}$ from the dataset $\mathcal{D}_c$.

Temporal Matching Learning in VOT. Suppose the clean video training dataset consists of n clean training videos, i.e., $\mathcal{D}_v = \{(V_i, B_i)\}_{i=1}^n$, where $V_i = \{\mathbf{I}_j\}_{j=1}^k$ denotes the i-th clean video containing k video frames, and $B_i = \{\mathbf{b}_j\}_{j=1}^k$ represents the corresponding box annotations. In each frame, the annotation $\mathbf{b}_j \in \mathbb{R}^4$ specifies the top-left coordinates, width, and height of the target. Typically, the deep tracker SiamFC [4] is trained to perform template matching on randomly sampled pairs of frames within a training video:

$$\min_\theta \mathbb{E}_{(\mathbf{z},\mathbf{x})\sim\mathcal{D}_v} \mathcal{L}(f_\theta(\mathbf{z}) * f_\theta(\mathbf{x}), y), \tag{2}$$

where f_θ is the backbone in SiamFC parameterized with θ. $\mathbf{z}$ and $\mathbf{x}$ are template and search images cropped in two randomly sampled frames $\mathbf{I}_i$ and $\mathbf{I}_j$ of a training video V. $*$ is the cross-correlation operation, and y is the ground-truth response map indicating where the target is in $\mathbf{x}$. $\mathcal{L}(\cdot,\cdot)$ represents the binary-cross entropy loss.

3.2 Baselines: UEs for VOT

Despite the progress of UEs for image classification, applying UEs to the VOT task is unexplored. To bridge this gap, we build new baselines by applying off-the-shelf EM [31] to VOT [4]. Specifically, the goal is to create perturbations that reduce the tracking loss function, thus exploiting a "shortcut" matching within the tracker:

$$\begin{aligned}&\min_\theta \mathbb{E}_{(\mathbf{z},\mathbf{x})\sim\mathcal{D}_v}\big[\min_{\boldsymbol{\delta}_t\in\boldsymbol{\Delta}} \mathcal{L}(f_\theta(\hat{\mathbf{z}}) * f_\theta(\hat{\mathbf{x}}), y)\big],\\ &\text{s.t. } \hat{\mathbf{z}} = \Phi(\mathbf{z}, \phi(\boldsymbol{\delta}_t, \mathbf{b}_i), \mathbf{b}_i),\ \hat{\mathbf{x}} = \Phi(\mathbf{x}, \phi(\boldsymbol{\delta}_t, \mathbf{b}_j), \mathbf{b}_j),\end{aligned} \tag{3}$$

where $\hat{\mathbf{z}}$ and $\hat{\mathbf{x}}$ are the TUEs for the template and search images. The TUEs are created by interpolating the target perturbation $\boldsymbol{\delta}_t$ to the same size as the bounding box, and pasting it onto the template/search image. Specifically, $\phi(\boldsymbol{\delta}_t, \mathbf{b})$ is the Bilinear interpolation function, which interpolates $\boldsymbol{\delta}_t$ to the same size as bounding box $\mathbf{b}$, and $\Phi(\mathbf{x}, \hat{\boldsymbol{\delta}}_t, \mathbf{b})$ is the pasting function that pastes $\hat{\boldsymbol{\delta}}_t$ onto the target region $\mathbf{b}$ of image $\mathbf{x}$ via the additional operation. Since each video only contains one tracked instance, a single $\boldsymbol{\delta}_t$ is defined for each video, *i.e.,* shared across frames within the same video.

In (3), the outer optimization updates the tracker θ, while the inner optimization optimizes the noise $\boldsymbol{\delta}_t$ with the tracker fixed. These steps alternate during training, with the perturbation noise $\boldsymbol{\delta}_t$ updated iteratively for T steps using the PGD method [49]. The above learning of TUEs $\hat{\mathbf{z}}$ and $\hat{\mathbf{x}}$ causes the tracker to rely on the perturbation noise for temporal matching, making the training videos unexploitable. Finally, the optimized noise set $\{\boldsymbol{\delta}_t^i\}_{i=1}^n$ is obtained for each of the n videos in $\mathcal{D}_v$.

Context-Aware UEs. Existing trackers [4, 71, 84] incorporate both the central target and surrounding context regions as the template $\mathbf{z}$ to enhance tracking performance. Inspired by this, we also take context regions into considerations when generating UEs to achieve more effective data privacy protection. Our approach follows the same aforementioned learning procedure, but now introduces context noise $\boldsymbol{\delta}_c$ during optimization. More details are provided in the Supplementary.

Tracker Training w/ UEs. After obtaining the target and context perturbations $\{\boldsymbol{\delta}_t^i, \boldsymbol{\delta}_c^i\}_{i=1}^n$, we interpolate the

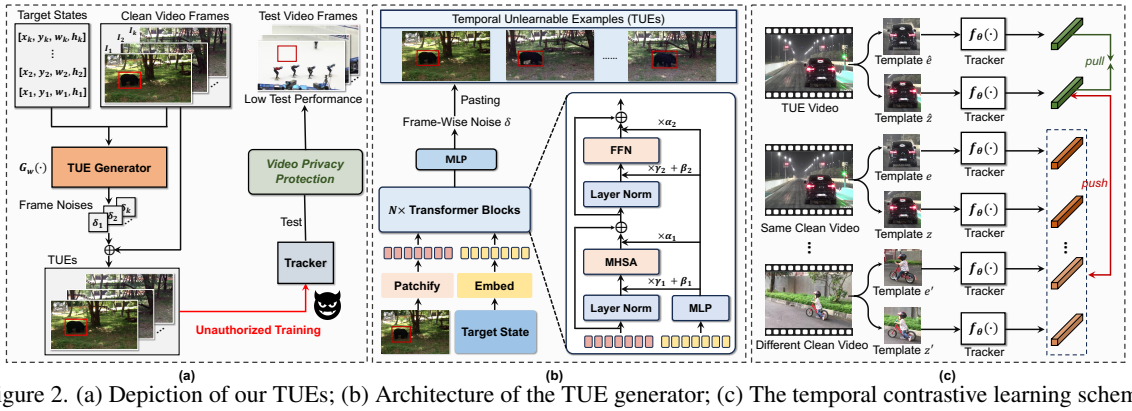

Figure 2. (a) Depiction of our TUEs; (b) Architecture of the TUE generator; (c) The temporal contrastive learning scheme.

target noise $\boldsymbol{\delta}_t^i$ and context noise $\boldsymbol{\delta}_c^i$ to match the target and context sizes in each raw frame of a given video V_i. These noises are then pasted onto the corresponding regions in every frame, resulting in a new unlearnable dataset $\mathcal{D}_u$, which is used to train standard trackers [9, 71, 84] following the official training protocols.

3.3 Generative Framework for TUE

The above baselines are based on the image-based approach EM, which has several limitations when applied to videos: 1) the iterative optimization of $\hat{\mathbf{z}}$ and $\hat{\mathbf{x}}$ is time-consuming, each optimization needs T iteration; 2) the noises $\boldsymbol{\delta}_t$ and $\boldsymbol{\delta}_c$ are pre-defined with fixed shapes, which need to be interpolated into various target scales in each frame of a video; 3) the generated noises $\{\boldsymbol{\delta}_t^i, \boldsymbol{\delta}_c^i\}_{i=1}^n$ are video specific, which cannot generalize to other unseen video datasets.

To address the above limitations, in contrast to image-based methods, where the UEs are directly optimized, we propose a new generative framework for our TUEs generation. Fig. 2 (a) depicts the overall pipeline of the proposed TUE generation process. Our TUE generation is formulated as follows:

$$\begin{aligned} &\min_{\theta} \mathbb{E}_{(\mathbf{z},\mathbf{x})\sim\mathcal{D}_v}\big[\min_{\mathbf{w}} \mathcal{L}(f_\theta(\hat{\mathbf{z}}) * f_\theta(\hat{\mathbf{x}}), y)\big], \\ &\text{s.t. } \hat{\mathbf{z}}=\mathbf{z}+G_{\mathbf{w}}(\mathbf{z}, \widetilde{\mathbf{b}}_i),\ \hat{\mathbf{x}}=\Phi(\mathbf{x}, G_{\mathbf{w}}(c(\mathbf{x}, \mathbf{b}_j), \widetilde{\mathbf{b}}_j), \mathbf{b}_j), \end{aligned} \tag{4}$$

where $\hat{\mathbf{z}}$ and $\hat{\mathbf{x}}$ are the generated TUEs for the template and search images. These TUEs are obtained by generating target-aware perturbation noises, and then pasting them onto the corresponding target regions in video frames. Specifically, $G_{\mathbf{w}}(\cdot)$ is our TUEs generator parameterized with $\mathbf{w}$, which takes a target patch $\mathbf{z}$ and normalized bounding box $\widetilde{\mathbf{b}}_i$ as input, and generates a noise perturbation $G_{\mathbf{w}}(\mathbf{z}, \widetilde{\mathbf{b}}_i)$ that is added back to $\mathbf{z}$. For the search image $\mathbf{x}$, the target is cropped using the cropping function $c(\mathbf{x}, \mathbf{b}_j)$ via the given box annotation $\mathbf{b}_j$, and then passed to the TUE generator with its corresponding normalized bounding box $\widetilde{\mathbf{b}}_j$ to generate the perturbation $G_{\mathbf{w}}(c(\mathbf{x}, \mathbf{b}_j), \widetilde{\mathbf{b}}_j)$. This perturbation is then pasted back onto the target in the search image via the $\Phi(\cdot)$ pasting function. Note that $\widetilde{\mathbf{b}}_i, \widetilde{\mathbf{b}}_j \in \mathbb{R}^4$ are the normalized target states (*i.e.,* containing the normalized top-left coordinates, width and height) of the corresponding $\mathbf{z}$ and $c(\mathbf{x}, \mathbf{b}_j)$ patches, respectively.

Algorithm 1: Optimization of TUE generator

Input : Surrogate model f_θ, TUE generator $G_{\mathbf{w}}$, learning rates α_s and α_g, number of epochs ep, clean dataset $\mathcal{D}_v = \{(V_i, B_i)\}_{i=1}^n$
Output: Optimized TUE generator $G_{\mathbf{w}}$
for $i \leftarrow 1$ ***to*** ep **do**
  for $(\mathbf{z}, \mathbf{x}) \in \mathcal{D}_v$ **do**
    # Generate perturbations
    $\hat{\mathbf{z}} = \mathbf{z} + G_{\mathbf{w}}(\mathbf{z}, \widetilde{\mathbf{b}}_i)$,
    $\hat{\mathbf{x}} = \Phi_c(\mathbf{x}, G_{\mathbf{w}}(c(\mathbf{x}, \mathbf{b}_j), \widetilde{\mathbf{b}}_j), \mathbf{b}_j)$;
    # Optimize perturbators $G_{\mathbf{w}}$ using α_s
    $\hat{\mathbf{e}} = c(\mathbf{x}, \mathbf{b}_j) + G_{\mathbf{w}}(c(\mathbf{x}, \mathbf{b}_j), \widetilde{\mathbf{b}}_j)$,
    $\mathcal{L}_f = \mathcal{L}(f_\theta(\hat{\mathbf{z}}) * f_\theta(\hat{\mathbf{x}}), y) + \lambda \mathcal{L}_{cl}(f_\theta(\hat{\mathbf{z}}), f_\theta(\hat{\mathbf{e}}))$,
    Optimize $G_{\mathbf{w}}$ via Adam to minimize $\mathcal{L}_f$;
    # Optimize the surrogate model f_θ using α_g
    $\mathcal{L}_f = \mathcal{L}(f_\theta(\hat{\mathbf{z}}) * f_\theta(\hat{\mathbf{x}}), y)$,
    Optimize f_θ via Adam to minimize $\mathcal{L}_f$;
  end
end

We employ a diffusion architecture, *i.e.,* DiT [53] as the TUE generator, which takes class label and time step as the condition for multi-step image generation. Here we adapt it to generate the target-aware TUEs in a single feed-forward step with the normalized target condition, which is more efficient. The architecture is in Fig. 2 (b).

Advantages: Our proposed generative TUE framework is more efficient than the EM baseline, in both training scalability and inference, and also transfers well to unseen videos. Specifically: 1) **High Training efficiency**: since $G_{\mathbf{w}}(\cdot)$ is lightweight and can be directly updated via the gradient back-propagation in each inner optimization, the training of $G_{\mathbf{w}}(\cdot)$ is efficient, e.g., $4\times$ faster than the EM baseline as illustrated in Table 2; 2) **Fewer learnable parameters**: The EM baseline optimizes both target and context noises for each video

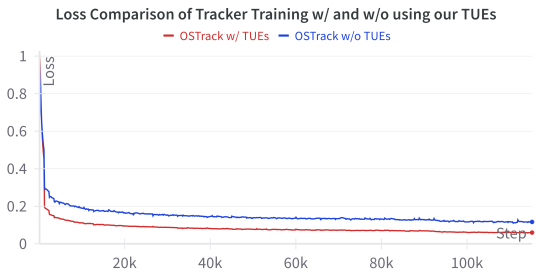

Figure 3. Training loss of OSTracker w/ and w/o our TUEs.

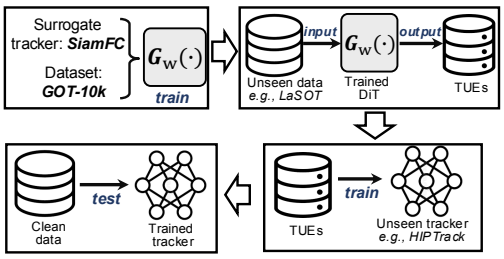

Figure 4. Overall workflow of our proposed TUEs.

(size 127×127×3), resulting in a large number of parameters on popular tracking benchmarks (e.g., 3.4GB for GOT-10k [32]). In contrast, our approach optimizes a lightweight model $G_{\mathbf{w}}(\cdot)$ with a fixed number of parameters (124MB), which is independent of dataset size; 3) **Applicable to Unseen Videos**: The EM baseline requires optimizing UEs for each dataset, which means it needs to be retrained to generate UEs for unseen datasets. As shown in Table 5, our TUE approach can generate UEs for unseen datasets directly through model inference, without the need for retraining.

3.4 Temporal Contrastive Learning

We further propose a Temporal Contrastive Loss (TCL) to encourage the tracker to rely more on the generated TUEs for temporal matching, thereby further degrading the model's tracking performance on clean test videos. Fig. 2 (c) illustrates the details of the proposed Temporal Contrastive Learning scheme. Specifically, the template TUE $\hat{\mathbf{z}}$ is used as the exemplar, treating the TUE $\hat{\mathbf{e}}$ (in the search region $\mathbf{x}$) within the same video as the positive sample. The clean templates in the same video (*i.e.,* $\mathbf{z}$ and $\mathbf{e} = c(\mathbf{x}, \mathbf{b}_j)$) and the other videos (*i.e.,* $\mathbf{z}'$ and $\mathbf{e}'$) within the same batch are regarded as the negative samples to $\hat{\mathbf{z}}$. The resulting formulation is:

$$\begin{aligned} &\min_{\mathbf{w}}[\mathcal{L}(f_\theta(\hat{\mathbf{z}}) * f_\theta(\hat{\mathbf{x}}), y) + \lambda \mathcal{L}_{cl}(f_\theta(\hat{\mathbf{z}}), f_\theta(\hat{\mathbf{e}}))], \\ &\text{s.t.} \quad \hat{\mathbf{e}} = c(\mathbf{x}, \mathbf{b}_j) + G_{\mathbf{w}}(c(\mathbf{x}, \mathbf{b}_j), \widetilde{\mathbf{b}}_j), \end{aligned} \tag{5}$$

where $\hat{\mathbf{e}}$ is the TUE in the searching region $\mathbf{x}$, and $\mathcal{L}_{cl}(\cdot)$ is the contrastive loss [6]. Note that we use the above objective to perform the inner optimization, while employing the tracker loss $\mathcal{L}(\cdot)$ for outer optimization to keep consistent with the original tracker training process. The overall optimization process of the TUE generator is outlined in Alg. 1. In each iteration, we first optimize the generator, followed by the surrogate model optimization. More details on the proposed temporal contrastive learning are in the supplementary.

Tracker training w/ TUEs. With the learned generator $G_{\mathbf{w}}(\cdot)$, we generate perturbations for existing tracking datasets and apply them to bounding boxes to create TUEs. These TUEs are then used to train various trackers following their official settings. As shown in Fig. 3, OSTrack trained with TUEs exhibits lower training loss, as it learns a "shortcut," relying on perturbations for temporal matching while ignoring the original data structure, thereby preserving training privacy.

3.5 Practical Application Pipeline

Fig. 4 illustrates our four-stage application pipeline. In Stage 1, we train the TUE generator $G_{\mathbf{w}}$ using the surrogate tracker SiamFC on an off-the-shelf tracking dataset (e.g., GOT-10k) under the supervision of Eq. 5. In Stage 2, the trained $G_{\mathbf{w}}$ generates TUEs to protect user video data. Stage 3 involves training an unauthorized tracker on the protected data. Finally, in Stage 4, the trained tracker is deployed on clean data to evaluate privacy protection performance. All experimental results are derived from Stage 4. We conduct experiments across multiple tracking applications, including VOT, VOS, and long-term point tracking, demonstrating that the protected videos effectively prevent unauthorized exploitation by these applications.

4 Experiments

4.1 Implementation Details

Our TUE generator $G_{\mathbf{w}}(\cdot)$ is implemented as a lightweight DiT-S/8 model [53] with 12 layers, 6 attention heads, and a hidden state size of 384. A fully connected layer maps the target state to the hidden space for controllable TUE generation. The generator is jointly trained with the naive SiamFC tracker [4] using Adam [34] for 50 epochs with a learning rate of 5×10^{-6} and a batch size of 16 (Algorithm 1). We set λ to 0.05. Following [4], the generator processes cropped template patches for efficiency. GOT-10k [32] is used as the source training dataset. SiamFC is chosen as the base tracker for two reasons: 1) its efficient training enables effective learning of the TUE generator, requiring only 7 hours on a single NVIDIA 4090 GPU; 2) the trained generator generalizes well to more complex trackers and datasets, avoiding time-intensive optimization with larger models. Once trained, the generator creates TUEs offline for training various trackers.

Evaluation. The trained models are evaluated on widely used tracking benchmarks, including GOT-10k [32], LaSOT [19], OTB [75], DAVIS-17 [55], and YTVOS-19 [78], using their standard evaluation metrics. Lower performance indicates stronger privacy protection.

Trackers	Variants	OTB [75] AUC	OTB [75] Prec.	GOT-10k [32] AO	GOT-10k [32] $SR_{0.5}$	GOT-10k [32] $SR_{0.75}$
SiamFC	Clean	58.6	79.2	35.5	39.0	11.8
	EM [31] Baseline	39.6	54.7	27.0	25.7	5.8
	+Context	29.5	37.4	21.4	18.2	4.3
	TUE Generator	17.6	19.9	16.1	10.4	**1.5**
	- Condi.	19.7	22.4	22.5	19.4	4.6
	+ TCL	**11.4**	**13.5**	**12.1**	**9.0**	1.9
OSTrack	Clean	67.4	89.4	71.0	80.4	68.2
	TUE Generator	45.3	60.6	41.6	47.3	30.3
	+ TCL	30.5	45.8	18.0	15.1	4.6

Table 1. Results of SiamFC and OSTrack trained on GOT-10k with clean videos (Clean); videos modified with EM baseline and further enhanced by including context optimiziation ("+Context"); and videos modified with our proposed TUE ("Ours"), and with temporal contrastive learning ("+TCL").

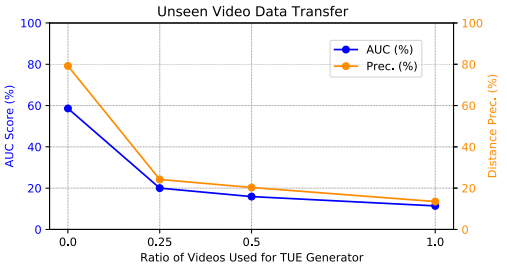

Figure 5. Ablation study on the data ratio in GOT-10k [32] used for training the TUE generator.

4.2 Ablation Studies

The effect of TCL. As illustrated in Tab. 1, using our TCL leads to larger performance degradation on both SiamFC and OSTrack. This is because TCL leads to larger distribution gap between clean samples and TUEs, which hinders the trackers' generalization to clean test data, thus ensuring training data privacy.

Incorporating context noise for optimization. In Tab. 1, we find that our EM baseline that only optimizes the target perturbation noise has limited effect. Incorporating the context noise for optimization leads to larger performance drops on both OTB and GOT-10k testing datasets. This indicates corrupting the target region only is not effective enough for SiamFC, i.e., the trackers can learn temporal matching from the context regions.

The usage of target state condition. In Eq. 4, we remove the target stage condition $\hat{\mathbf{b}}_i$ ("- Condi.") and only use the input image for TUE generation. This variant ignores the target state, achieving inferior protection performance, which shows that dynamically adapting to the target state is helpful for TUE learning.

Data ratio for training TUE generator. As shown in Fig. 5, we train our TUE generator using different video ratios from GOT-10k [32]. Specifically, we train the generator on a subset of randomly selected videos and then use it to generate TUEs for the entire dataset, which are used to fine-tune a SiamFC tracker. Results show that the TUE generator can be effectively learned with just 25% of the GOT-10k videos (about 2,300 videos) and generate effective TUEs on unseen videos.

Method	Training Time	Learnable Parameter Size
EM + Context	33 Hours	3.4GB
TUE Generator	7 Hours	124MB

Table 2. Model complexity of our TUE and EM baseline.

Variants	SeqTrack Train. Epoch	GOT-10k AO	GOT-10k $SR_{0.5}$	GOT-10k $SR_{0.75}$
Clean	500	74.7	84.7	71.8
TUE Generator	100	9.7 (65.0 ↓)	4.1 (80.6 ↓)	0.5 (71.3 ↓)
TUE Generator	200	**2.1 (72.6 ↓)**	0.9 (83.8 ↓)	0.1 (71.7 ↓)
TUE Generator	300	2.2 (72.5 ↓)	1.2 (83.5 ↓)	0.2 (71.6 ↓)
TUE Generator	400	3.8 (70.9 ↓)	1.2 (83.5 ↓)	0.1 (71.7 ↓)
TUE Generator	500	2.3 (72.4 ↓)	**0.7 (84.0 ↓)**	**0.0 (71.8 ↓)**

Table 3. Training SeqTrack [9] with TUEs generated by our TUE generator on GOT-10k for various training epochs.

Model complexity. As shown in Tab. 2, training the TUE generator takes 7 hours on a single RTX4090, significantly faster than the EM-based iterative optimization. Additionally, our approach uses fewer learnable model parameters, leading to more efficient training.

Training epochs vs. performance drop. In Tab. 3, we train SeqTrack-256 [9] on TUE-GOT10k to investigate the impact of training epochs. Training for only 200 epochs with TUEs effectively degrades its performance. Note that SeqTrack-256 originally uses 500 epochs, which demonstrates that TUEs can effectively corrupt the tracker training w/ less training epochs.

Bbox dependency. Our TUE generator requires the target bounding box as the input for target-aware TUEs generation. In practical applications, users can annotate short video clips manually or employ off-the-shelf trackers to generate reliable pseudo bboxes, similar to the annotations used in TrackingNet [51]. In addition, we also validate that our TUEs can be automatically generated (using naive unsupervised EdgeBox to generate bbox proposals) in Tab. R3 of the Supplementary, eliminating the need for user intervention.

4.3 Transfer Experiments

Transferability is crucial in real-world applications [88, 89]. We evaluate our method's transferability to various trackers, datasets, and the dense temporal matching task.

Transfer to State-of-the-art Trackers. We train our TUE generator with the naive SiamFC tracker on GOT-10k [32] and use the learned generator to produce TUE-perturbed GOT-10k (TUE-GOT10k). To evaluate the transferability, we train state-of-the-art trackers, including OSTrack-256 [84], DropTrack-384 [71], SeqTrack-256 [9], MixFormer-CvT [12], STARK-S50 [80], AQATrack-256 [77], and HIPTrack [5], on TUE-GOT10k, following their original training settings. For other UE methods: (1) EM [31] and TAP [22] are optimization-based approaches specifically optimized with SiamFC on GOT-10k; (2) LSP [85] and AR [56] are class-wise UEs. To adapt them, we randomly sam-

VOT Method	UE Method	GOT-10k [32] AO	GOT-10k [32] $SR_{0.5}$	GOT-10k [32] $SR_{0.75}$	OTB-100 [75] AUC	OTB-100 [75] P	LaSOT [19] AUC	LaSOT [19] P_{Norm}	LaSOT [19] P
SiamFC [4]	Clean	35.5	39.0	11.8	58.6	79.2	34.0	39.9	33.0
	TAP [22]	32.9	35.0	9.5	56.7	76.5	31.4	36.7	29.8
	LSP [85]	28.1	29.3	7.9	50.0	67.9	26.6	30.9	24.2
	AR [56]	34.1	37.2	11.5	56.9	76.7	33.6	38.9	32.3
	EM [31]	21.4	18.2	4.3	29.5	37.4	17.6	19.6	14.5
	TUE (Ours)	**12.1 (23.4↓)**	**9.0 (30.0↓)**	**1.9 (9.9↓)**	**11.4 (47.2↓)**	**13.5 (65.7↓)**	**9.5 (24.5↓)**	**9.5 (30.4↓)**	**6.4 (26.6↓)**
OSTrack-256 [84]	Clean	71.0	80.4	68.2	67.4	89.4	62.3	70.2	66.0
	TAP [22]	**17.0**	17.3	10.2	44.9	61.7	26.8	32.2	29.1
	LSP [85]	27.8	29.7	16.4	48.2	66.1	28.3	33.3	30.2
	AR [56]	67.0	75.9	62.4	63.8	85.1	59.2	67.0	62.0
	EM [31]	26.3	24.0	10.2	48.8	69.6	29.9	36.2	31.8
	TUE (Ours)	18.0 (53.0↓)	**15.1 (65.3↓)**	**4.6 (63.6↓)**	**30.5 (36.9↓)**	**45.8 (43.6↓)**	**22.0 (40.3↓)**	**27.7 (42.5↓)**	**23.1 (42.9↓)**
DropTrack-384 [71]	Clean	75.9	86.8	72.0	69.4	91.3	66.5	75.2	71.5
	TAP [22]	51.7	57.9	41.8	51.4	70.6	39.2	45.9	40.4
	LSP [85]	27.8	29.3	20.3	66.5	87.7	38.3	44.4	39.8
	AR [56]	66.9	75.5	61.9	66.4	87.6	60.4	68.4	63.7
	EM [31]	21.6	18.7	7.0	48.7	72.2	33.3	40.9	36.4
	TUE (Ours)	**17.1 (58.8↓)**	**12.9 (73.9↓)**	**2.7 (69.3↓)**	**36.7 (32.7↓)**	**57.7 (33.6↓)**	**25.2 (41.3↓)**	**32.2 (43.0↓)**	**28.3 (43.2↓)**
SeqTrack-256 [9]	Clean	74.7	84.7	71.8	68.1	89.9	63.6	72.4	67.9
	TAP [22]	8.1	8.6	4.4	39.4	57.0	18.1	23.1	21.6
	LSP [85]	13.5	14.3	6.8	44.5	63.8	25.0	31.4	28.1
	AR [56]	64.3	73.1	59.1	64.5	85.7	56.0	64.3	58.7
	EM [31]	8.1	6.0	1.1	34.7	55.2	15.3	21.3	19.9
	TUE (Ours)	**2.1 (72.6↓)**	**0.9 (83.8↓)**	**0.1 (71.7↓)**	**7.0 (61.1↓)**	**14.3 (75.6↓)**	**3.4 (60.2↓)**	**5.4 (67.0↓)**	**6.3 (61.6↓)**
MixFormer-CvT [12]	Clean	70.7	80.0	67.8	66.1	88.6	62.1	69.9	65.6
	TAP [22]	12.4	11.0	4.6	40.4	53.9	27.7	31.8	27.7
	LSP [85]	20.9	21.9	10.9	44.7	60.8	31.2	36.0	31.2
	AR [56]	51.4	57.9	43.9	62.0	81.6	57.2	64.7	58.6
	EM [31]	7.0	6.8	2.7	45.1	62.0	24.5	30.6	27.7
	TUE (Ours)	**1.9 (68.8↓)**	**0.1 (79.9↓)**	**0.2 (67.6↓)**	**14.7 (51.4↓)**	**22.9 (65.7↓)**	**8.1 (54.0↓)**	**10.7 (59.2↓)**	**11.4 (54.2↓)**
STARK-S50 [80]	Clean	67.2	76.1	61.2	64.1	84.7	58.2	65.7	59.5
	TAP [22]	18.6	15.6	6.6	18.7	21.7	18.3	10.8	6.0
	LSP [85]	8.7	5.0	1.5	22.7	29.4	11.4	**8.1**	**2.2**
	AR [56]	51.7	51.9	41.8	52.9	69.5	46.7	44.6	35.5
	EM [31]	14.8	15.1	8.3	43.4	56.3	28.3	31.3	26.7
	TUE (Ours)	**2.6 (64.6↓)**	**1.1 (75.0↓)**	**0.2 (61.0↓)**	**13.9 (50.2↓)**	**17.2 (67.5↓)**	**8.9 (49.3↓)**	8.9 (56.8↓)	7.3 (52.2↓)
AQATrack-256 [77]	Clean	73.2	82.6	71.5	69.1	91.5	64.3	72.7	69.0
	EM [31]	19.4	16.2	4.9	47.1	65.3	31.1	37.2	31.8
	TUE (Ours)	**16.2 (57.0↓)**	**10.6 (72.0↓)**	**1.7 (69.8↓)**	**20.8 (48.3↓)**	**28.9 (62.6↓)**	**17.4 (46.9↓)**	**20.3 (52.4↓)**	**16.7 (52.3↓)**
HIPTrack [5]	Clean	77.4	88.0	74.5	68.8	90.3	66.8	75.1	72.0
	EM [31]	63.0	71.9	52.7	66.9	88.9	55.5	64.0	59.3
	TUE (Ours)	**43.2 (34.2↓)**	**43.9 (44.1↓)**	**20.0 (54.5↓)**	**56.9 (11.9↓)**	**78.6 (11.7↓)**	**38.7 (28.1↓)**	**48.5 (26.6↓)**	**42.7 (29.3↓)**

Table 4. Methodology transfer on various trackers. We use TUEs, which are specifically optimized with SiamFC on the GOT-10k training set, to train SOTA deep trackers. The trained trackers are tested on clean GOT-10k, OTB-100, and LaSOT test sets. Performance drops of our method are shown in brackets. The best results are shown in bold.

Training Dataset	UE Method	GOT-10k [32] AO	GOT-10k [32] $SR_{0.5}$	GOT-10k [32] $SR_{0.75}$	OTB-100 [65] AUC	OTB-100 [65] P	LaSOT [20] AUC	LaSOT [20] P_{Norm}	LaSOT [20] P
LaSOT	Clean	55.6	62.1	44.6	49.2	64.0	58.2	65.7	59.5
	AR [56]	52.3	57.9	40.1	45.3	58.5	52.4	59.6	53.3
	LSP [85]	15.8	12.8	5.0	23.5	31.0	18.1	20.3	17.0
	EM [31]	11.4	9.9	7.9	21.9	29.8	18.9	22.0	19.9
	TUE (Ours)	**4.1 (51.5↓)**	**3.0 (59.1↓)**	**0.8 (43.8↓)**	**9.5 (39.7↓)**	**15.7 (48.3↓)**	**5.7 (52.5↓)**	**7.6 (58.1↓)**	**7.6 (51.9↓)**
LaSOT+GOT-10k	Clean	66.2	76.1	59.7	64.6	85.4	62.1	70.9	65.1
	AR [56]	61.2	70.0	53.9	60.9	80.5	58.2	66.1	59.3
	LSP [85]	27.0	25.8	11.6	36.6	47.5	30.2	31.6	25.9
	EM [31]	20.6	20.5	11.3	27.2	35.7	22.4	24.5	20.6
	TUE (Ours)	**4.0 (62.2↓)**	**2.1 (74.0↓)**	**0.6 (59.1↓)**	**11.0 (53.6↓)**	**16.9 (68.5↓)**	**6.8 (55.3↓)**	**8.4 (62.5↓)**	**7.8 (57.3↓)**

Table 5. Unseen dataset transfer. We use our TUE generator, which is specifically optimized with SiamFC on the GOT-10k training set, to perform zero-shot TUEs generation on the unseen LaSOT training set. The obtained TUE-perturbed LaSOT and the different combination (i.e., TUE-perturbed LaSOT + GOT-10k) are used to train the base tracker STARK-S50 [80].

ple a class-wise UE noise per video, creating perturbed videos. More details are provided in the Supplementary.

Tab. 4 shows that our TUE outperforms other UE methods in privacy protection across various tracking architectures, including CNNs (SiamFC, Stark), ViTs (OSTrack, DropTrack), and Decoders (HIPTrack, AQATrack). HIPTrack exhibits less performance degradation since it uses DropTrack as the frozen base tracker, which limits its ability to learn shortcut features. Notably, our TUEs, trained with a naive SiamFC model in about seven hours on a single NVIDIA 4090 GPU, transfer effectively to SOTA trackers, eliminating the need for costly optimization with complex tracking models.

Transfer to Unseen Datasets. To ensure scalability on large-scale video datasets, we perform zero-shot TUE generation on unseen datasets. Specifically, after training the TUE generator on GOT-10k w/ SiamFC, we use it to generate TUEs for LaSOT (TUE-LaSOT) without additional training. We then train STARK-S50 [80] on TUE-LaSOT for evaluation. As shown in Tab. 5, the optimization-based EM struggles to generalize beyond GOT-10k, requiring video-wise UEs sampled from GOT-10k for LaSOT. In contrast, our TUE achieves significant performance drops across three datasets. Training STARK-S50 on both TUE-LaSOT and TUE-GOT10k further amplifies performance degra-

VOS Method	Training Dataset	UE Method	DAVIS-17 Val [55] $\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	YTVOS-19 Val [78] $\mathcal{J}\&\mathcal{F}$	$\mathcal{J}_{seen}$	$\mathcal{J}_{unseen}$
STCN [11]	DAVIS-17	Clean	71.2	67.3	72.7	63.3	66.3	56.1
		TUE (Ours)	**50.1 (21.1↓)**	**46.8 (20.5↓)**	**53.4 (19.3↓)**	**42.4 (20.9↓)**	**42.7 (23.6↓)**	**36.9 (19.2↓)**
	DAVIS-17+YTVOS-19	Clean	82.5	79.3	85.7	82.7	81.1	78.2
		AR [56]	80.1	77.1	83.2	80.6	79.9	75.0
		LSP [85]	75.9	73.0	78.7	77.7	77.0	73.0
		EM [31]	71.5	68.4	74.6	76.0	74.8	72.4
		TUE (Ours)	**63.6 (18.9↓)**	**59.9 (19.4↓)**	**67.4 (18.3↓)**	**65.7 (17.0↓)**	**62.6 (18.5↓)**	**62.2 (16.0↓)**
XMEM [10]	DAVIS-17+YTVOS-19	Clean	84.5	81.4	87.6	84.2	83.8	78.1
		AR [56]	82.1	78.7	83.9	81.8	81.9	75.3
		LSP [85]	81.7	78.6	84.7	82.0	80.9	76.9
		EM [31]	78.9	75.7	82.1	80.5	78.3	76.6
		TUE (Ours)	**64.2 (20.3↓)**	**60.5 (20.9↓)**	**67.9 (19.7↓)**	**61.3 (22.9↓)**	**58.4 (25.4↓)**	**58.1 (20.0↓)**

Table 6. Transfer to dense temporal matching task, i.e., Video Object Segmentation (VOS). We apply our TUE generator, trained with SiamFC on GOT-10k for the VOT task, to perform zero-shot TUEs generation on DAVIS-17 and YTVOS-19 training sets. The mask annotations are firstly converted to box annotations. Lower performance indicates stronger training data privacy protection.

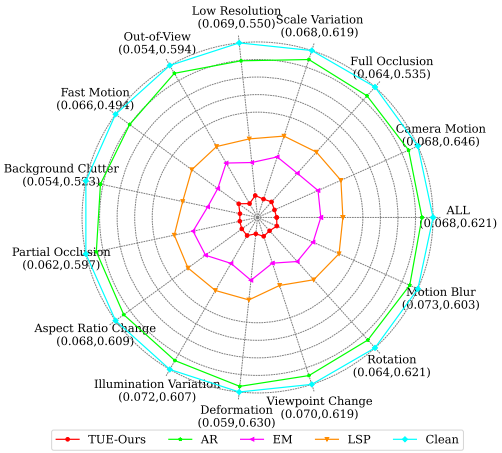

Figure 6. AUC scores of different attributes on LaSOT [19]. We use STARK-S50 [80] as the base tracker, which is trained with perturbed datasets (i.e., LaSOT + GOT-10k [32]) generated by AR, EM, LSP, and our TUE. Best viewed in color.

dation. Overall, our method efficiently generates large-scale TUEs via model inference without retraining.

Transfer to Video Object Segmentation (VOS). VOS is a dense temporal matching task that segments targets in each frame based on initial mask annotations. We evaluate our TUEs in this context by first generating bounding boxes around mask annotations and then using a generator trained on VOT to create unlearnable VOS datasets, TUE-DAVIS17 and TUE-YTVOS19. These datasets are used to train existing VOS methods following standard protocols. As shown in Tab. 6, despite differences between VOT and VOS, our TUEs cause significant performance degradation across various VOS methods and datasets, highlighting their effectiveness in disrupting temporal matching and protecting video data privacy. Further transferability to long-term point tracking is demonstrated in the supplementary.

Attribute Analysis. Fig. 6 presents the AUC scores for various attributes on LaSOT [19]. First, we apply AR, EM, LSP, and our TUE methods to the VOT dataset (i.e., LaSOT [19] + GOT-10k [32]) to generate unlearnable datasets. We then use STARK-S50 [80] as the base tracker, training it on the unlearnable datasets produced by these data privacy protection algorithms. Our method shows largest performance drop across all attributes.

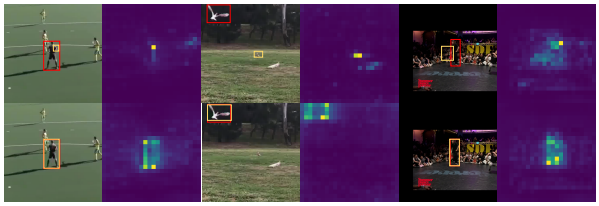

Figure 7. Template-to-search attention visualization from TUE-DropTrack on clean videos (top) and TUE-perturbed videos (bottom). Red and yellow rectangles are ground truth and predicted bounding boxes in search regions, respectively.

Visualization. Fig. 7 visualizes the template-to-search attention weights from the last layer of ViT in TUE-DropTrack. TUE-DropTrack, trained with TUEs, heavily relies on them for temporal matching. For clean videos, TUE-DropTrack generates inaccurate attention maps due to overfitting on TUEs. The supplementary provides additional visualizations demonstrating that the perturbations are imperceptible while maintaining high-quality perturbed frames, along with further visualizations of attention weights, perturbations, and TUEs.

5 Conclusion

This paper presented the first effort to address the privacy concerns about unauthorized data exploitation in VOT. We constructed a comprehensive benchmark to evaluate existing UE methods, revealing prior methods' limitations in efficiency, effectiveness, and generalizability. To overcome these issues, we introduced Temporal Unlearnable Examples (TUEs) with a lightweight generative framework, which conditions on target states to produce target-aware noise perturbations. Additionally, we designed a temporal contrastive loss to encourage trackers to rely on TUEs during training, further strengthening privacy protection. Extensive experiments show that TUEs transfer well across trackers, datasets, and temporal matching tasks. Ablation studies validated the effectiveness and efficiency of our generative framework. Qualitative results confirm that the generated perturbations are imperceptible, effectively protect VOT videos from unauthorized use. We expect our TUE to advance data privacy in the tracking community.

Acknowledgments

This work was carried out at the Rapid-Rich Object Search (ROSE) Lab, School of Electrical & Electronic Engineering, Nanyang Technological University (NTU), Singapore. This research is supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Trust Tech Funding Initiative, the Research Grants Council of the Hong Kong Special Administrative Region, China (CityU 11211624) and the National Natural Science Foundation of China under Project 62406090. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and Infocomm Media Development Authority.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016. 2

[2] Yifan Bai, Zeyang Zhao, Yihong Gong, and Xing Wei. Artrackv2: Prompting autoregressive tracker where to look and how to describe. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[3] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. Torr. Staple: Complementary learners for real-time tracking. In *CVPR*, pages 1401–1409, 2016. 2

[4] L. Bertinetto, J. Valmadre, J.F. Henriques, A. Vedaldi, and P.H.S. Vedaldi. Fully-convolutional siamese networks for object tracking. In *ECCV Workshop*, pages 850–865, 2016. 2, 3, 5, 7

[5] Wenrui Cai, Qingjie Liu, and Yunhong Wang. Hiptrack: Visual tracking with historical prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2, 6, 7

[6] T. Chen, S. Kornblith, and M. Norouzi. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, 2020. 5

[7] Xuesong Chen, Xiyu Yan, Feng Zheng, Yong Jiang, Shu-Tao Xia, Yong Zhao, and Rongrong Ji. One-shot adversarial attacks on visual tracking with dual attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10176–10185, 2020. 2

[8] X. Chen, B. Yan, J. Zhu, D. Wang, X. Yang, and H. Lu. Transformer tracking. In *CVPR*, pages 8126–8135, 2021. 2

[9] Xin Chen, Houwen Peng, Dong Wang, Huchuan Lu, and Han Hu. Seqtrack: Sequence to sequence learning for visual object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14572–14581, 2023. 2, 4, 6, 7

[10] H. K. Cheng and A G. Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV*, 2022. 8

[11] H. K. Cheng, Y. W. Tai, and C. K. Tang. Rethinking space-time networks with improved memory coverage for efficient video object segmentation. In *NeurIPS*, pages 11781–11794, 2021. 8

[12] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Mixformer: End-to-end tracking with iterative mixed attention. In *CVPR*, 2022. 2, 6, 7

[13] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg. Convolutional features for correlation filter based visual tracking. In *ICCVW*, pages 58–66, 2015. 2

[14] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *ICCV*, pages 4310–4318, 2015. 2

[15] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: learning continuous convolution operators for visual tracking. In *ECCV*, pages 472–488, 2016. 2

[16] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, pages 21–26, 2017. 2

[17] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg. Discriminative scale space tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(8): 1561–1575, 2017. 2

[18] A. Dosovitskiy, L. Beyer, and A. Kolesnikov. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2

[19] H. Fan, L. Lin, and F. Yang. Lasot: A high-quality benchmark for large-scale single object tracking. In *CVPR*, pages 5374–5383, 2019. 1, 2, 5, 7, 8

[20] H. Fan, H. Bai, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, M. Huang, J. Liu, and Y. Xu. Lasot: A high-quality large-scale single object tracking benchmark. In *IJCV*, 2021. 1, 2, 7

[21] Ji Feng, Qi-Zhi Cai, and Zhi-Hua Zhou. Learning to confuse: generating training time adversarial data with auto-encoder. *NeurIPS*, 32, 2019. 2

[22] Liam Fowl, Micah Goldblum, Ping-yeh Chiang, Jonas Geiping, Wojciech Czaja, and Tom Goldstein. Adversarial examples make strong poisons. *NeurIPS*, 34:30339–30351, 2021. 3, 6, 7

[23] Shaopeng Fu, Fengxiang He, Yang Liu, Li Shen, and Dacheng Tao. Robust unlearnable examples: Protecting data privacy against adversarial learning. In *ICLR*, 2022. 1, 3

[24] H. Galoogahi, A. Fagg, and S. Lucey. Learning background-aware correlation filters for visual tracking. In *ICCV*, 2017. 2

[25] Shenyuan Gao, Chunluan Zhou, and Jun Zhang. Generalized relation modeling for transformer tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18686–18695, 2023. 2

[26] Qing Guo, Xiaofei Xie, Felix Juefei-Xu, Lei Ma, Zhongguo Li, Wanli Xue, Wei Feng, and Yang Liu. Spark: Spatial-aware online incremental attack against visual tracking. In *European conference on computer vision*, pages 202–219. Springer, 2020. 2

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 2

[28] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015. 2

[29] Lingyi Hong, Shilin Yan, Renrui Zhang, Wanyun Li, Xinyu Zhou, Pinxue Guo, Kaixun Jiang, Yiting Chen, Jinglun Li, and Zhaoyu Chen. Onetracker: Unifying visual object tracking with foundation models and efficient tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2

[30] Bin Huang, Jiaqian Yu, Yiwei Chen, Siyang Pan, Qiang Wang, and Zhi Wang. Badtrack: a poison-only backdoor attack on visual object tracking. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[31] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. In *ICLR*, 2021. 1, 2, 3, 6, 7, 8

[32] 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. 1, 2, 5, 6, 7, 8

[33] Shuai Jia, Chao Ma, Yibing Song, and Xiaokang Yang. Robust tracking against adversarial attacks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIX 16*, pages 69–84. Springer, 2020. 2

[34] Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *arXiv:1412.6980*, 2014. 5

[35] A. Krizhevsky, S. Ilya, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Communications of the ACM*, pages 84–90, 2017. 2

[36] B. Li, W. Wu, Z. Zhu, and J. Yan. High performance visual tracking with siamese region proposal network. In *Proceedings of the CVPR*, pages 8971–8980, 2018. 2

[37] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 2019. 2

[38] Jiahao Li, Yiqiang Chen, Yunbing Xing, Yang Gu, and Xiangyuan Lan. K-space bispectrum steganography for robust unlearnable data. In *Proceedings of the 33rd ACM International Conference on Multimedia*. Association for Computing Machinery, 2025. 2

[39] Jiahao Li, Yiqiang Chen, Yunbing Xing, Yang Gu, and Xiangyuan Lan. A survey on unlearnable data, 2025. 3

[40] Yiming Li, Haoxiang Zhong, Xingjun Ma, Yong Jiang, and Shu-Tao Xia. Few-shot backdoor attacks on visual object tracking. *arXiv preprint arXiv:2201.13178*, 2022. 2

[41] Siyuan Liang, Xingxing Wei, Siyuan Yao, and Xiaochun Cao. Efficient adversarial attacks for visual object tracking. In *ECCV*, pages 34–50, 2020. 2

[42] Y. Liang, Q. Wu, Y. Liu, Y. Yan, and H. Wang. Deep correlation filter tracking with shepherded instance-aware proposals. In *IEEE Transactions on Intelligent Transportation Systems*, 2021. 2

[43] Xun Lin, Wenzhong Tang, Haoran Wang, Yizhong Liu, Yakun Ju, Shuai Wang, and Zitong Yu. Exposing image splicing traces in scientific publications via uncertainty-guided refinement. *Patterns*, 5(9):101038, 2024. 2

[44] Xun Lin, Yi Yu, Song Xia, Jue Jiang, Haoran Wang, Zitong Yu, Yizhong Liu, Ying Fu, Shuai Wang, Wenzhong Tang, et al. Safeguarding medical image segmentation datasets against unauthorized training via contour-and texture-aware perturbations. *arXiv preprint arXiv:2403.14250*, 2024. 3

[45] Xun Lin, Yi Yu, Zitong Yu, Ruohan Meng, Jiale Zhou, Ajian Liu, Yizhong Liu, Shuai Wang, Wenzhong Tang, Zhen Lei, et al. Hidemia: Hidden wavelet mining for privacy-enhancing medical image analysis. In *ACM MM*, pages 8110–8119, 2024. 2

[46] Yi Liu, Yanjie Liang, Qiangqiang Wu, Liming Zhang, and Hanzi Wang. A new framework for multiple deep correlation filters based object tracking. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1670–1674, 2022. 2

[47] Zhuoran Liu, Zhengyu Zhao, and Martha Larson. Image shortcut squeezing: Countering perturbative availability poisons with compression. *International Conference on Machine Learning*, 2023. 3

[48] A. Lukezic and T. Vojir. Discriminative correlation filter woth channel and spatial reliability. In *CVPR*, 2017. 2

[49] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018. 3

[50] Ruohan Meng, Chenyu Yi, Yi Yu, Siyuan Yang, Bingquan Shen, and Alex C Kot. Semantic deep hiding for robust unlearnable examples. *IEEE Transactions on Information Forensics and Security*, 2024. 3

[51] M. Muller, A. Bibi, and Giancola S. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, pages 300–317, 2018. 1, 6

[52] Krishna Kanth Nakka and Mathieu Salzmann. Temporally-transferable perturbations: Efficient, one-shot adversarial attacks for online visual object trackers. *arXiv preprint arXiv:2012.15183*, 2020. 2

[53] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 4, 5

[54] NhatHai Phan, Yue Wang, Xintao Wu, and Dejing Dou. Differential privacy preservation for deep auto-encoders: an application of human behavior prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016. 2

[55] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. In *arXiv:1704.00675*, 2017. 5, 8

[56] Pedro Sandoval-Segura, Vasu Singla, Jonas Geiping, Micah Goldblum, Tom Goldstein, and David Jacobs. Autoregressive perturbations for data poisoning. *NeurIPS*, 35:27374–27386, 2022. 6, 7, 8

[57] Shawn Shan, Emily Wenger, Jiayun Zhang, Huiying Li, Haitao Zheng, and Ben Y Zhao. Fawkes: Protecting privacy against unauthorized deep learning models. In *29th USENIX security symposium (USENIX Security 20)*, pages 1589–1604, 2020. 2

[58] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015.

[59] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017. 2

[60] Ye Sun, Hao Zhang, Tiehua Zhang, Xingjun Ma, and Yu-Gang Jiang. Unseg: One universal unlearnable example generator is enough against all image segmentation. *arXiv preprint arXiv:2410.09909*, 2024. 3

[61] Zhihong Sun, Jun Chen, Liang Chao, Weijian Ruan, and Mithun Mukherjee. A survey of multiple pedestrian tracking based on tracking-by-detection framework. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(5):1819–1833, 2021. 2

[62] R. Tao, E. Gavves, and A. W.M. Smeulders. Siamese instance search for tracking. In *CVPR*, pages 1420–1429, 2016. 2

[63] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr. End-to-end representation learning for correlation filter based tracking. In *CVPR*, pages 5000–5008, 2017. 2

[64] Q. Wang, J. Gao, and J. Xing. Dcfnet: Discriminant correlation filters network for visual tracking. In *arXiv:1704.04057*, 2017. 2

[65] X. Wang, X. Shu, Z. Zhang, B. Jiang, Y. Wang, Y. Tian, and F. Wu. Towards more flexible and accurate object tracking with natural language: Algorithms and benchmark. In *CVPR*, 2021. 7

[66] Xianlong Wang, Minghui Li, Wei Liu, Hangtao Zhang, Shengshan Hu, Yechao Zhang, Ziqi Zhou, and Hai Jin. Unlearnable 3d point clouds: Class-wise transformation is all you need. *arXiv preprint arXiv:2410.03644*, 2024. 3

[67] Xing Wei, Yifan Bai, Yongchao Zheng, Dahu Shi, and Yihong Gong. Autoregressive visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[68] Qiangqiang Wu and Antoni B Chan. Meta-graph adaptation for visual object tracking. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2021. 2

[69] Q. Wu, Y. Yan, Y. Liang, Y. Liu, and H. Wang. Dsnet: Deep and shallow feature learning for efficient visual tracking. In *ACCV*, pages 119–134, 2018. 2

[70] Qiangqiang Wu, Yan Yan, Yanjie Liang, Yi Liu, and Hanzi Wang. Dsnet: Deep and shallow feature learning for efficient visual tracking. In *Asian Conference on Computer Vision*, pages 119–134, 2019. 2

[71] Qiangqiang Wu, Tianyu Yang, Ziquan Liu, Baoyuan Wu, Ying Shan, and Antoni B. Chan. Dropmae: Masked autoencoders with spatial-attention dropout for tracking tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14561–14571, 2023. 2, 3, 4, 6, 7

[72] Qiangqiang Wu, Tianyu Yang, Wei Wu, and Antoni B Chan. Scalable video object segmentation with simplified framework. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13879–13889, 2023. 2

[73] Shutong Wu, Sizhe Chen, Cihang Xie, and Xiaolin Huang. One-pixel shortcut: On the learning preference of deep neural networks. In *ICLR*, 2023. 3

[74] Xugang Wu, Xiaoping Wang, Xu Zhou, and Songlei Jian. Sta: Adversarial attacks on siamese trackers. *arXiv preprint arXiv:1909.03413*, 2019. 2

[75] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015. 5, 6, 7

[76] Fei Xie, Zhongdao Wang, and Chao Ma. Diffusiontrack: Point set diffusion model for visual object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[77] Jinxia Xie, Bineng Zhong, Zhiyi Mo, Shengping Zhang, Liangtao Shi, Shuxiang Song, and Rongrong Ji. Autoregressive queries for adaptive tracking with spatio-temporal transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19300–19309, 2024. 2, 6, 7

[78] N. Xu, L. Yang, Y. Fan, D. Yue, Y. Liang, J. Yang, and T. Huang. Youtube-vos: A large-scale video object segmentation benchmark. In *arXiv:1809.03327*, 2018. 5, 8

[79] Bin Yan, Dong Wang, Huchuan Lu, and Xiaoyun Yang. Cooling-shrinking attack: Blinding the tracker with imperceptible noises. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 990–999, 2020. 2

[80] B. Yan, H. Peng, J. Fu, D. Wang, and H. Lu. Learning spatio-temporal transformer for visual tracking. In *ICCV*, pages 10448–10457, 2021. 6, 7, 8

[81] Xiyu Yan, Xuesong Chen, Yong Jiang, Shu-Tao Xia, Yong Zhao, and Feng Zheng. Hijacking tracker: A powerful adversarial attack on visual tracking. In *ICASSP*, pages 2897–2901, 2020. 2

[82] T. Yang and A. B. Chan. Learning dynamic memory networks for object tracking. In *ECCV*, pages 152–167, 2018. 2

[83] T. Yang, P. Xu, and R. Hu. Roam: Recurrently optimizing tracking model. In *CVPR*, pages 6718–6727, 2020. 2

[84] B. Ye, H. Chang, B. Ma, and S. Shan. Joint feature learning and relation modeling for tracking: A one-stream framework. In *ECCV*, pages 341–357, 2022. 2, 3, 4, 6, 7

[85] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. Availability attacks create shortcuts. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2367–2376, 2022. 3, 6, 7, 8

[86] Yi Yu, Yufei Wang, Song Xia, Wenhan Yang, Shijian Lu, Yap-Peng Tan, and Alex C Kot. Purify unlearnable examples via rate-constrained variational autoencoders. In *International Conference on Machine Learning, ICML 2024*, 2024. 1

[87] Yi Yu, Qichen Zheng, Siyuan Yang, Wenhan Yang, Jun Liu, Shijian Lu, Yap-Peng Tan, Kwok-Yan Lam, and Alex Kot. Unlearnable examples detection via iterative filtering. In *International Conference on Artificial Neural Networks*, pages 241–256. Springer, 2024. 3

[88] Yi Yu, Song Xia, Xun Lin, Chenqi Kong, Wenhan Yang, Shijian Lu, Yap-Peng Tan, and Alex C Kot. Towards model resistant to transferable adversarial examples via trigger activation. *IEEE Transactions on Information Forensics and Security*, 2025. 6

[89] Yi Yu, Song Xia, Siyuan Yang, Chenqi Kong, Wenhan Yang, Shijian Lu, Yap-Peng Tan, and Alex Kot. Mtl-ue: Learning to learn nothing for multi-task learning. In *International Conference on Machine Learning*. PMLR, 2025. 1, 6

[90] Chia-Hung Yuan and Shan-Hung Wu. Neural tangent generalization attacks. In *International Conference on Machine Learning*, pages 12230–12240. PMLR, 2021. 3

[91] L. Zhang, A. Gonzalez-Garcia, J. Weijer, M. Danelljan, and F. Khan. Learning the model update for siamese trackers. In *ICCV*, 2019. 2

[92] Xiaosong Zhang, Yunjie Tian, Wei Huang, Qixiang Ye, Qi Dai, Lingxi Xie, and Qi Tian. Hivit: Hierarchical vision transformer meets masked image modeling. *arXiv preprint arXiv:2205.14949*, 2022. 2