Supplementary Material
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

In this supplementary material, we provide more implementation details, ablation study, and qualitative visualization. Sec. A details more implementation details of our Temporal Unlearnable Example (TUE) generation, the UE baseline w/ context noise optimization, and compared off-the-shelf UE approaches. Sec. B illustrates the evaluation metrics and datasets used in the tasks of VOT, Video Object Segmentation (VOS), and long-term Point Tracking. Sec. C presents more quantitative results, including additional ablation experiments, attribute analysis on LaSOT [9], transferability to long-term Point Tracking, as well as the robustness against the annotation noise. Sec. D shows more qualitative visualization of attention weights, generated perturbation noises, and Temporal Unlearnable Examples (TUEs). Finally, we discuss the limitations and future work in Sec. E.

A Implementation Details

In this section, we introduce additional implementation details in terms of offline TUE generation, UE baseline w/ context noise optimization and compared off-the-shelf UE methods.

A.1 TUE Generation

We optimize the generator via the Algorithm 1 of the main paper. After obtaining the learned TUE generator, we apply it to perform TUEs generation on existing tracking datasets [9, 13, 14, 20], which is illustrated in Fig. F1. Specifically, following SiamFC [1], we first crop the template in each frame and use the generator to adaptively generate perturbation noise for each frame. The generated perturbation noise is bounded by $\|\boldsymbol{\delta}\|_\infty \le \frac{8}{255}$, which is imperceptible to a human observer according to previous studies in adversarial research. We then interpolate and paste the generated noises onto the target regions (i.e., indicated by bounding boxes) for each frame.

A.2 UE Baseline w/ Context Noise

SiamFC crops the template $\mathbf{z}$, capturing both the central target region and surrounding context. The EM [12] baseline in Eq. (3) of the main paper only optimizes the target noise, neglecting the context, which is crucial for temporal matching. To improve this, we propose incorporating the context for TUE generation:

$$\underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{(\mathbf{z},\mathbf{x})\sim\mathcal{D}_v}[\min_{\boldsymbol{\delta}_t,\boldsymbol{\delta}_c} \mathcal{L}(f_\theta(\hat{\mathbf{z}}) * f_\theta(\hat{\mathbf{x}}), y)], \quad (1)$$

$$\text{s.t.} \quad \begin{aligned} \hat{\mathbf{z}} &= \mathbf{z} + E(\phi(\boldsymbol{\delta}_t, \mathbf{b}_i), \boldsymbol{\delta}_c), \\ \hat{\mathbf{x}} &= \Phi_c(\mathbf{x}, E(\phi(\boldsymbol{\delta}_t, \mathbf{b}_j), \boldsymbol{\delta}_c), \mathbf{b}_j), \end{aligned} \quad (2)$$

where $\boldsymbol{\delta}_c$ is the context noise with the same shape to $\boldsymbol{\delta}_t$, $E(,)$ is the combination function to spatially combine both target and context noises. The new paste function $\Phi_c(,,)$ adds the context noise jointly with the target noise into corresponding target-context regions in $\mathbf{x}$. Finally, we optimize the noises via the alternative inner-outer optimization, and obtain the optimized target and context noise set, i.e., $\{\boldsymbol{\delta}_t^i, \boldsymbol{\delta}_c^i\}_{i=1}^n$ on all n clean videos. The noises are first interpolated via the Bicubic interpolation to meet the target and context size in each frame, and then pasted onto the frame to obtain the perturbed frame, which is similar to our TUE pipeline shown in Fig. F1.

*Equal contribution. †Corresponding author.

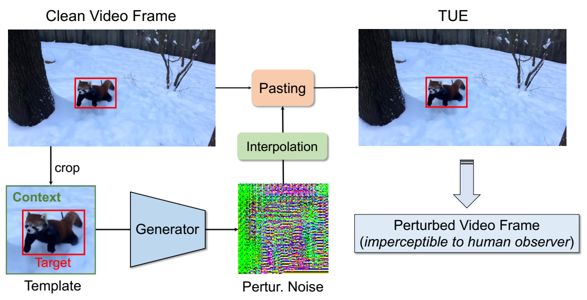

Figure F1. Details of offline TUE generation via the learned generator. We apply the TUE generator to perform TUEs generation on existing tracking datasets in an offline manner. The generated TUEs are further used to train existing deep trackers. For frames with multiple bounding boxes (i.e., the multi-object scenarios in VOS and long-term point tracking), we generate perturbations for each box and overlay them onto the original frames to create TUEs.

A.3 Temporal Contrastive Learning

In Sec. 3.4 and Fig. 2(c) of the main manuscript, we propose a Temporal Contrastive Loss (TCL) to make the tracker rely more on the generated TUEs for temporal matching. The proposed TCL $\mathcal{L}_{cl}(\cdot)$ uses $\hat{\mathbf{z}}$ as the exemplar, and treats the TUEs $\hat{\mathbf{e}}$ within the same video as the positive sample. The clean templates $\mathbf{h}$ in the same video and the other videos are regarded as the negative samples (i.e., denoted as $\{e, z, e', z'\}$ in Fig. 2(c) of the main manuscript), which can be detailed as:

$$\mathcal{L}_{cl} = -\log \frac{\exp(\hat{\mathbf{z}} \cdot \hat{\mathbf{e}}/\tau)}{\sum_{i=1}^{N} \exp(\hat{\mathbf{z}} \cdot \mathbf{h}_i/\tau)}, \tag{3}$$

where $\tau = 0.05$ and N is the number of clean templates in a mini-batch.

A.4 Off-the-Shelf UE Methods

For both LSP [23] and AR [15], we follow their default settings to generate 100 class-wise perturbations with a shape of $3 \times 127 \times 127$ and a bound of $\ell_2 = 1 \times \frac{127}{32} \times \frac{127}{32}$ to ensure comparability with other methods. For each video sequence, a perturbation tensor of size $3 \times 127 \times 127$ is randomly selected from the generated perturbations, interpolated, and applied to the area of the bounding box.

EM [12] and TAP [11] are both optimization-based UE approaches. For fair comparison, we optimize them on the VOT task using the same surrogate model and training dataset w/ our TUE. Specifically, for EM, we build it as described in Sec. 3.2 of the main paper, and optimize it w/ SiamFC on the GOT-10k dataset. For TAP, following EM, we also employ SiamFC as the surrogate model to generate uniform adversarial perturbations for each video sequence with the bound $\ell_\infty = \frac{8}{255}$. The interpolation and pasting process is identical to that used for the other approaches.

B Evaluation Metrics and Datasets

In this section, we introduce the evaluation metrics, training and testing datasets used in the tasks of VOT, VOS, and long-term Point Tracking.

B.1 VOT Metrics

In the experiments of the main paper, we use standard metrics proposed in existing tracking datasets [9, 13, 18] for evaluation. Specifically, for OTB-100 [18], we report both the distance precision rates at the threshold of 20 pixels (P) and the Area Under the Curve (AUC). For GOT-10k [13], the average overlap (AO) scores, the success rate (SR) at the threshold of 0.5 ($\text{SR}_{0.5}$) and 0.75 ($\text{SR}_{0.75}$) are employed for evaluation. Following previous works [17, 22], an additional metric of normalized P (P_{Norm}) is used for the LaSOT [9, 10] dataset.

B.2 VOS Metrics

We use the official evaluation metrics including $\mathcal{J}$ and $\mathcal{F}$ scores, to evaluate VOS approaches. Note that $\mathcal{J}$ is calculated as the average IoU between the prediction and groundtruth masks. $\mathcal{F}$ measures the boundary similarity measure between the prediction and ground-truth masks. The $\mathcal{J}\&\mathcal{F}$ score is the average of the above two metrics. For YTVOS-19 [20], $\mathcal{J}$ is calculated on testing videos containing both seen and unseen target categories, referred to as $\mathcal{J}seen$ and $\mathcal{J}unseen$, respectively. For all the evaluation in this work, lower testing performance indicates better training data privacy protection.

B.3 Point Tracking Metrics

We use the standard metrics in TAP-Vid benchmarks [7] for evaluation, including Position Accuracy (δ^x_{avg}), Occlusion Accuracy (OA) and Average Jaccard (AJ). δ^x_{avg} quantifies the average positional accuracy of visible points, while OA represents the proportion of points with correctly predicted visibility. AJ jointly calculates position and occlusion accuracy. Following [7], we use the stardard TAP-DAVIS dataset for evaluation.

B.4 Datasets

We use popular tracking datasets for training and evaluation. For VOT, we use the GOT-10k [13] training set to train our TUE generator. We then generate TUEs on the GOT-10k training set in an offline manner, in order to train existing state-of-the-art trackers following their

Variants	λ	Prec.	AUC
Clean	-	79.2	58.6
TUE	0.0	19.9 (59.3↓)	17.6 (41.0↓)
TUE	0.01	15.9 (63.3↓)	14.3 (44.3↓)
TUE	0.05	**13.5 (65.7↓)**	**11.4 (47.2↓)**
TUE	0.1	17.9 (61.3↓)	16.0 (42.6↓)

Table R1. Ablation study on the effect of λ used in the temporal contrastive loss. Note that we use SiamFC [1] as the base tracker for evaluation on OTB-100 [18]. Performance drops are shown in brackets. The best results are shown in bold.

Metrics	δ^x_{avg}	OA	AJ
Clean	80.4	88.1	64.6
TUE	72.8	81.6	57.4

Table R2. Transfer to Long-term Point Tracking and evaluation on TAP-DAVIS: TUE-DAVIS is used to train the base DINO-Tracker [16]. Drop is smaller than VOT/VOS due to the frozen DINOv2, reducing TUE impact.

UE Method	DAVIS-17 Val			YTVOS19-val			OTB-100	
	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}_{seen}$	$\mathcal{J}_{unseen}$	AUC	P
Clean	71.2	67.3	72.7	63.3	66.3	56.1	49.2	64.0
AR [51]	68.9	65.4	70.5	-	-	-	45.3	58.5
LSP [77]	65.7	62.9	68.6	-	-	-	23.5	31.0
EM [31]	64.0	61.0	67.0	57.5	59.4	51.1	21.9	29.8
TUE	**50.1**	**46.8**	**53.4**	**42.4**	**42.7**	**36.9**	**9.5**	**15.7**
TUE w/o BBox (30)	58.3	55.7	61.0	-	-	-	-	-
TUE w/o BBox (15)	<u>55.5</u>	<u>52.1</u>	<u>58.9</u>	<u>51.9</u>	<u>52.5</u>	<u>45.6</u>	<u>18.9</u>	<u>26.5</u>

Table R3. Results on VOS and VOT: respectively using STCN and STARK-50 as the base trackers trained on DAVIS17 and LaSOT with added perturbations from various UE methods. "w/o BBox (k)" uses EdgeBox to generate k pseudo bboxes per-frame.

standard training settings w/o modifications. The obtained trackers are evaluated on the GOT-10k testing set, LaSOT testing set, and OTB-100 using official toolkits and the online evaluation server. For VOS, we generate TUEs on DAVIS-17 [14] and YTVOS-19 [20] training sets, which are further used to train VOS models including STCN [5] and XMEM [4]. We evaluate the trained STCN and XMEM on popular DAVIS-17 and YTVOS-19 validation sets. We submit the results to the official online server for YTVOS-19 evaluation.

C Additional Quantitative Results

For completeness, more ablation studies and attribute performance analysis are included in this section.

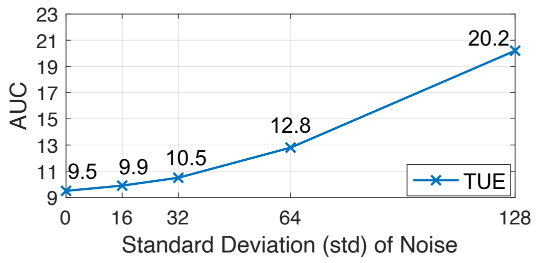

Figure F2. AUC (OTB-100) of STARK50 trained on TUE-LaSOT with varying annotation noise (Gaussian noise added to bbox coordinates) .

Trackers	Variants	Pasting Strategy	OTB [18]		GOT-10k [13]		
			AUC	Prec.	AO	$SR_{0.5}$	$SR_{0.75}$
SiamFC	Clean	-	58.6	79.2	35.5	39.0	11.8
	TUE	1) Target Only	22.2	24.5	-	-	-
	TUE	2) Target + Context	**11.4**	**13.5**	**12.1**	**9.0**	**1.9**
OSTrack	Clean	-	67.4	89.4	71.0	80.4	68.2
	TUE	1) Target Only	**30.5**	**45.8**	**18.0**	**15.1**	**4.6**
	TUE	2) Target + Context	33.7	46.5	29.3	32.8	16.9

Table R4. Ablation study of using different pasting strategies for TUE generation. 'Target Only' indicates pasting the generated perturbation noise to the target bounding box region only. 'Target + Context' indicates the pasting includes the target context region in each frame. The best results are shown in bold.

C.1 The Effect of λ

The hyper-parameter λ in (5) of the main paper controls the balance between the temporal contrastive loss (TCL) and the temporal matching loss. Here, we study the effect of λ in Table R1. An appropriate selection of λ leads to significant improvements on data-privacy protection, leading to larger testing performance drop on OTB-100 in terms of both precision and AUC metrics. This is because TCL makes larger distribution gaps between clean and TUE videos, thus causing more tracking failures on testing videos and ensuring better training video privacy.

C.2 The Effect of Pasting Strategies

In Table R4, we study two pasting strategies after obtaining the generated perturbation noise: 1) only pasting the interpolated noise to the target bounding box region in the frame; 2) the pasting covers both the target and context regions. Interestingly, for scale regression-based OSTrack [22], we find that the former strategy leads to better data privacy performance, possibly because pasting the noise to the context region may cause ambiguous scale regression, which hinders the scale regression learning. In contrast, pasting noise only in the target region may simplify scale matching across frames, leading to more severe over-fitting on scale regression learning. Therefore, for scale-regression based trackers, in-

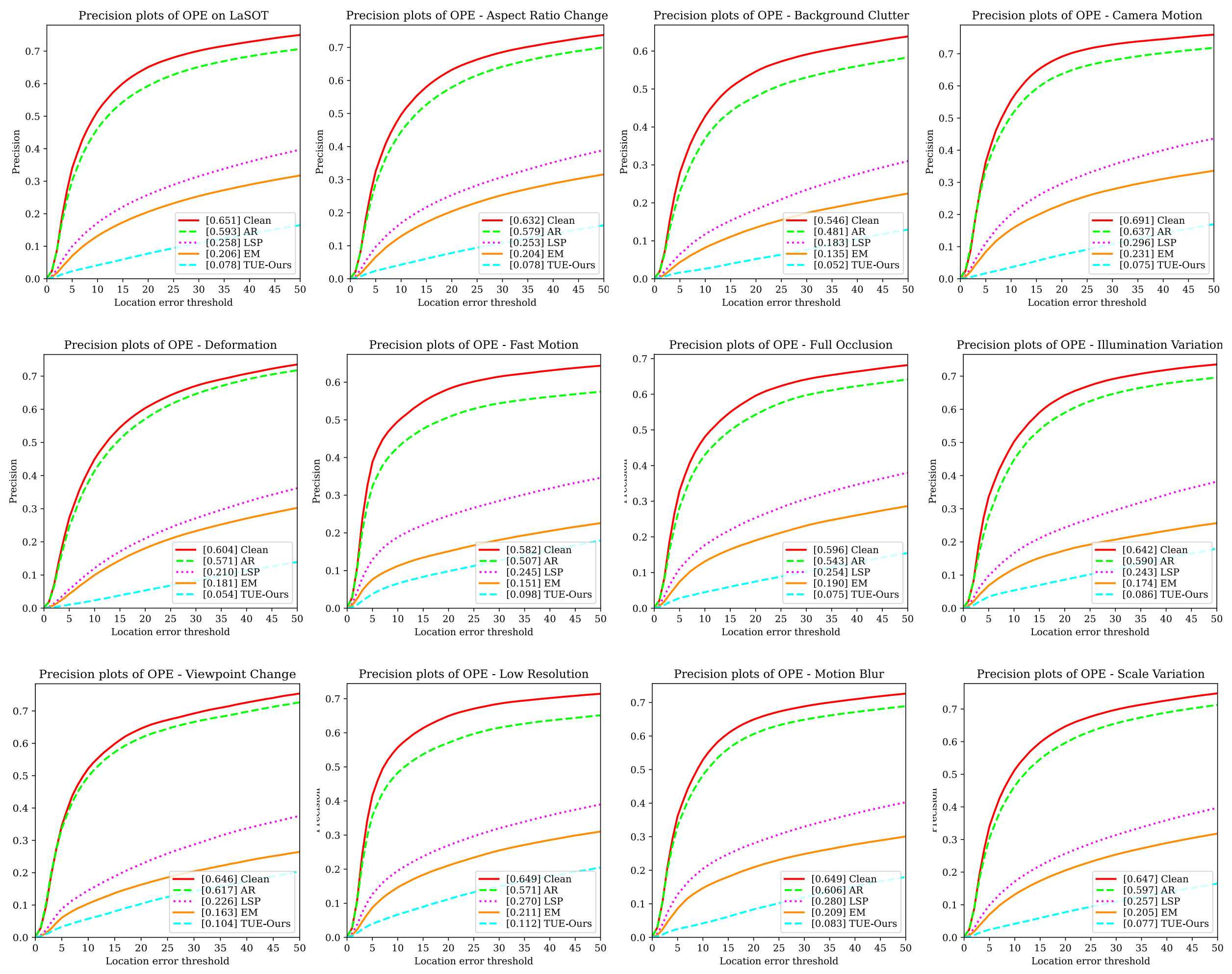

Figure F3. Precision plots of different attributes on LaSOT [9]. We use STARK-S50 [21] as the base tracker, which is trained with perturbed datasets (i.e., LaSOT + GOT-10k [13]) generated by AR [15], EM [12], LSP [23], and our TUE. Best viewed in color. Our TUE significantly degrades the tracker trained on protected training video datasets, thus achieving lowest performance on all attributes of LaSOT.

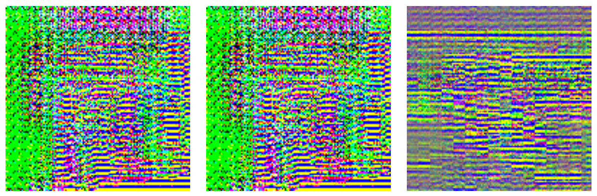

Figure F4. Visualization of the generated perturbation noises on the first (left) and last (middle) frames of a randomly sampled video (i.e., *GOT-10k_Train_000160*) from the GOT-10k training set. To emphasize the difference, the difference map is shown on the right.

cluding OSTrack [22], DropTrack [17], SeqTrack [3], MixFormer-CvT [6], STARK [21], AQATrack [19] and HIPTrack [2], we paste the perturbation noises generated by our TUE and the other compared UE approaches to the target region only in each video frame for further tracker training. For the naive SiamFC w/o scale regression, we find that pasting w/ the context region lead to larger performance drop, which is mainly because SiamFC heavily relies on the context for temporal matching.

C.3 Attribute Analysis

In this subsection, we expand the attribute analysis experiments from our main paper to provide a more comprehensive evaluation of attribute performance on LaSOT, which is shown in Fig. F3. Notably, the tracker trained with TUEs exhibits the lowest performance across all attributes of LaSOT, highlighting that TUEs significantly impair tracker training and effectively protect the privacy of the training video data.

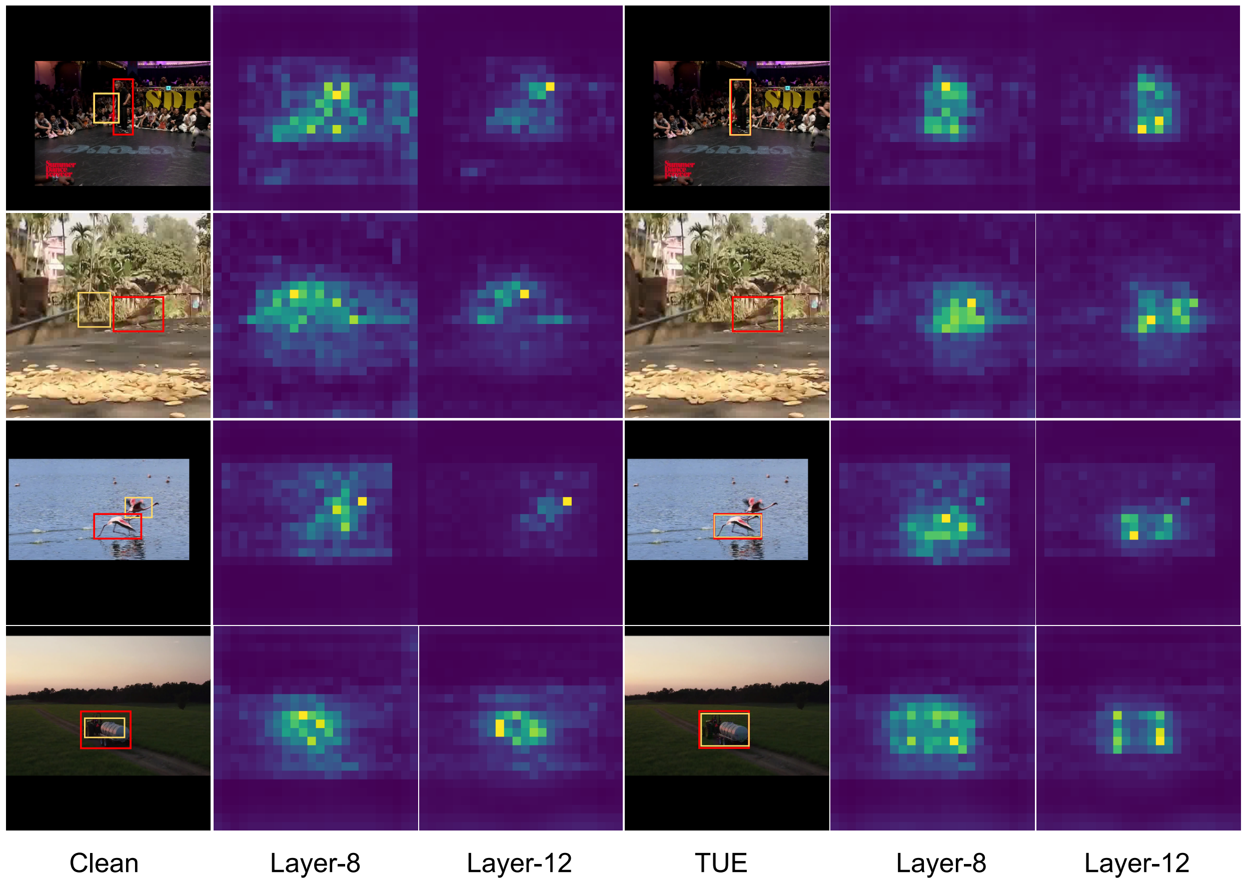

Figure F5. Template-to-search attention weights visualization from TUE-DropTrack on clean videos ('Clean') and TUE-perturbed videos ('TUE'). Red and yellow rectangles are ground truth and predicted bounding boxes, respectively. The attention weights are extracted from the 8-th and 12-th layers of the ViT in TUE-DropTrack.

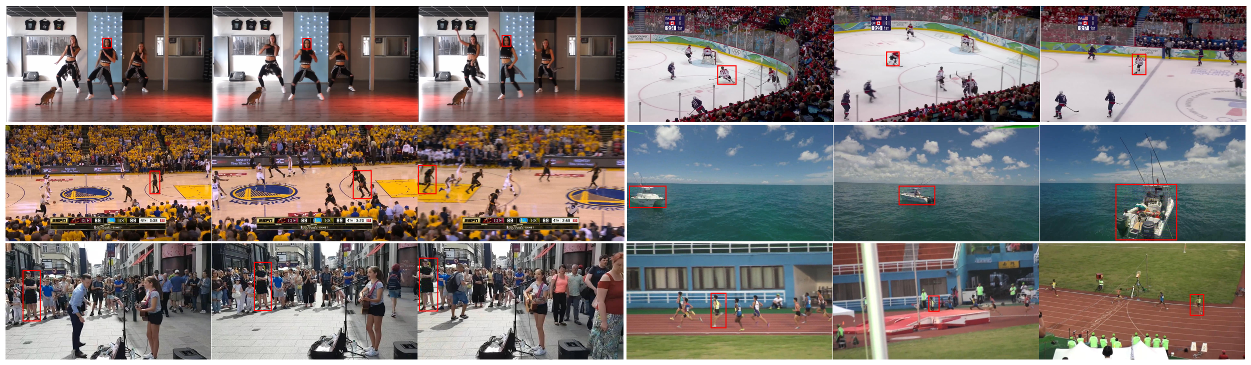

Figure F6. Visualization of TUEs on VOT datasets. The video frames are sampled from both TUE-GOT10k and TUE-LaSOT. Red boxes are ground truth bounding boxes.

C.4 Transfer to Long-term Point Tracking

Long-term point tracking is a dense tracking task which aims to accurately track dense points in long-term videos. Here we use our TUE generator, which is specifically optimized with SiamFC on the VOT training set (GOT-10k [13]), to perform zero-shot TUEs generation on the TAP-DAVIS [11] dataset (TUE-DAVIS) based on its mask annotations (similar to VOS). We then use TUE-DAVIS to train the SOTA point tracker DINO-Tracker [16], which is further evaluated on the clean TAP-DAVIS [7] dataset. Although the most parame-

Figure F7. Visualization of TUEs on VOS datasets. The video frames are sampled from TUE-DAVIS17. For multiple targets in a video frame, we apply our TUE generator to generate perturbation noise for each target.

ters in DINO-Tracker are frozen (i.e., the frozen DINOv2 model) and the trainable parameters are limited ($\sim$7.6M), we can still observe the performance degradation in Table R2 in terms of all the metrics. This demonstrates that our TUEs is also effective for protecting training video data-privacy in the dense long-term point tracking.

C.5 Bounding Box Dependency

Our model generates TUEs using the bounding boxes of protected objects within video frames as conditions. In practical scenarios, users can manually annotate some short videos. For long videos, similar to annotation strategies in TrackingNet, users may readily use open-source algorithms (e.g., using EdgeBox [24]) to generate k reliable pseudo bboxes in each video frame. Similar to the VOS transfer experiment, we then apply the TUE generator learned in VOT to generate TUEs for each bounding box and paste them on the original video frames for automatic video protection. Table R3 presents the VOT and VOS results using STCN and STARK-50 as the base trackers trained on the DAVIS17 and LaSOT datasets, respectively. The results demonstrate that our TUEs can be automatically generated (using naive EdgeBox to generate bbox proposals), eliminating the need for user intervention. This variant obtains competitive performance to our method and outperforms previous approaches.

C.6 Robustness to Annotation Noise

We further examine the robustness of our model against bounding box shift in the inference stage (Stage 2 in Fig. 4 of the main manuscript). Fig. F2 shows that our TUEs are robust to annotation noises, even under severe noise conditions (std = 128), validating the feasibility of using pseudo annotations. Future research could focus on improving automatic bbox generation or enhancing robustness to bbox noise. Nonetheless, our work represents an important first step in demonstrating the feasibility of TUEs under ideal bbox conditions.

D Qualitative Visualization

In this section, we show more visualizations of attention weights, perturbation noises and generated TUEs.

D.1 Visualization of Attention Weights

In Fig. F5, we visualize the attention weights extracted from the 8-th and 12-th layers of ViT [8] used in TUE-DropTrack trained w/ TUE-GOT10k. The tracker trained w/ TUEs heavily rely on the region with the TUE (the GT bounding box) for temporal matching, which indicates that the tracker training severely over-fits on our TUEs, which prevents the tracker from exploring original data structure, thus ensuring training data privacy.

D.2 Visualization of Perturbation Noises

The generated perturbation noise on a randomly sampled video from GOT-10k is are shown in Fig. F4. For different frames, our TUE dynamically generates adaptive perturbation noise (as shown in the difference map on the right), whereas other approaches, such as EM [12] and TAP [11], apply a fixed video-wise UE across all frames within a video.

D.3 Visualization of TUEs

We use $\ell_\infty = \frac{8}{255}$ bound for *invisibility*, and the PSNR of TUEs on LASOT is 51.8. We show our TUEs on VOT datasets in Fig. F6. The perturbation noises are pasted to the target regions (red boxes) to obtain TUEs. The TUEs on VOS datasets are shown in Fig. F7. Different from VOT, there may exist multiple tracked targets in a video frame. To adapt to these cases, we apply our TUE generator to generate perturbation noise for each target in the frame, and then paste them onto the corresponding target regions in the frame. For overlap regions, we just simply add the perturbations together.

E Limitations and Future Work

In this work, the proposed TUE generator is optimized in the single object tracking task. While we have also applied our TUE generator to the multi-object video object segmentation (VOS) task and observed performance degradation, demonstrating its task transferability, several limitations remain: 1) our approach does not explicitly address the challenges of multi-object scenarios in VOS, e.g., for the overlapped target regions in the same video frame, we naively combine their perturbation noises, which may not be optimal. The future work should consider more about overlapped target regions, since directly adding perturbation noises together may disrupt their structural integrity; 2) To generate TUEs, we convert VOS mask annotations into bounding box annotations, which may lead to a loss of fine-grained target details. Developing methods to directly generate mask-level perturbation noises would likely preserve more detailed information and improve performance.

In addition, we are the first to prevent videos from unauthorized tracker training, providing new baselines and benchmarks for the tracking community. The future work can focus on developing more robust video data privacy protection approaches. Moreover, following the attack community, some "defense" strategies can also be designed for perturbation detection or removal.

References

[1] 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. 1, 3

[2] 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. 4

[3] 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. 4

[4] H. K. Cheng and A G. Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV*, 2022. 3

[5] 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. 3

[6] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Mixformer: End-to-end tracking with iterative mixed attention. In *CVPR*, 2022. 4

[7] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adria Recasens, Lucas Smaira, Yusuf Aytar, Joao Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022. 2, 5

[8] A. Dosovitskiy, L. Beyer, and A. Kolesnikov. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 6

[9] 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, 4

[10] 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. 2

[11] Liam Fowl, Micah Goldblum, Ping-yeh Chiang, Jonas Geiping, Wojciech Czaja, and Tom Goldstein. Adversarial examples make strong poisons. *NeurIPS*, 34:30339–30351, 2021. 2, 5, 6

[12] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. In *ICLR*, 2021. 1, 2, 4, 6

[13] 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, 3, 4, 5

[14] 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. 1, 3

[15] Pedro Sandoval-Segura, Vasu Singla, Jonas Geiping, Micah Goldblum, Tom Goldstein, and David Jacobs. Autoregressive perturbations for data poisoning. *NeurIPS*, 35:27374–27386, 2022. 2, 4

[16] Narek Tumanyan, Assaf Singer, Shai Bagon, and Tali Dekel. Dino-tracker: Taming dino for self-supervised point tracking in a single video. In *ECCV*, pages 367–385. Springer, 2025. 3, 5

[17] 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, 4

[18] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015. 2, 3

[19] 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. 4

[20] 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. 1, 2, 3

[21] B. Yan, H. Peng, J. Fu, D. Wang, and H. Lu. Learning spatio-temporal transformer for visual tracking. In *ICCV*, pages 10448–10457, 2021. 4

[22] 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

[23] 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. 2, 4

[24] C. Lawrence Zitnick and Piotr Doll ́ar. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014. 6