A Robust UAV Tracking Solution in the Adversarial Environment

Mengjie Jia
University of Massachusetts Dartmouth
mjia@umassd.edu

Yanyan Li
California State University San Marcos
yali@csusm.edu

Jiawei Yuan
University of Massachusetts Dartmouth
jyuan@umassd.edu

2024 IEEE 36th International Conference on Tools with Artificial Intelligence (ICTAI) | 979-8-3315-2723-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICTAI62512.2024.00136

***Abstract*—Unmanned aerial vehicles (UAVs) are ideal platforms for object tracking due to their high mobility and advanced sensing capabilities. Recent advancements in AI have enhanced UAV tracking by integrating deep learning models with UAV systems, but they also introduce security concerns due to the vulnerabilities of deep learning models to adversarial attacks. To address this challenge, we propose a new UAV tracking solution integrating a reconstruction module with an anomaly detection module to enhance the robustness of UAV tracking systems against attacks. Our reconstruction module processes video frames to mitigate adversarial impacts without compromising tracking performance on clean frames. The anomaly detection module employs a reference generator to dynamically construct adversarial reference samples for feature map comparisons to effectively detect attacks.**

We evaluated our solution against state-of-the-art attacks on three benchmarks. The results show that our solution improves the tracking performance under attack conditions, achieving an average precision of 97.4% and a success rate of 96.3% of the original tracking. Additionally, our method achieves a 98.9% attack detection rate with a 4.23% false positive rate in anomaly detection. The evaluation results demonstrate the effectiveness of our approach in enhancing the robustness and reliability of UAV tracking systems in the adversarial environment.

I. Introduction

Featured by their high mobility and rich sensing capabilities, UAVs have been adopted as an ideal platform for object tracking. In addition, the recent advances in AI techniques have further promoted the performance of UAV tracking and attracted considerable attention from academia and industry [1]–[6]. Despite the benefits introduced by AI techniques to UAV tracking, their adoption also raises additional security concerns because many deep learning models in these AI techniques can be vulnerable to adversarial inputs. Recent research has demonstrated that deep learning (DL)-based UAV tracking algorithms can be significantly affected by adversarial perturbations applied to the inputs, which misleads the UAV to make wrong decision during the object detection and tracking [7]–[11].

Given the fact that object detection and tracking are widely integrated into various UAV tasks, it is critical to improve the robustness of UAV tracking solutions to counter potential attacks. Existing research works have shown that adversarial training [12], [13] and input reconstruction [14]–[16] can be used to improve the robustness of deep learning models. However, directly adopting them in the context of UAV tracking faces multiple challenges. Adversarial training is typically computationally expensive and can compromise the model performance. In addition, the creation of adversarial examples for training can also be a complex task. In terms of input reconstruction, most of the existing defenses focus on image classification tasks and do not consider some important operations in the context of UAV tracking, such as the prediction and update of the location and size of the target objects in a sequence of video frames. For example, the state-of-the-art Siamese trackers [3]–[6] utilize the predicted location of the target in the previous video frame to facilitate the tracking prediction of the current frame. Simply reconstructing the attacked video frames without considering the impacts from preceding frames is not sufficient to handle attacks towards UAV tracking.

In this paper, we proposed a robust UAV tracking solution that can maintain its performance in the adversarial environment. Our solution is designed with a reconstruction module coupled with an anomaly detection module, which is then integrated with state-of-the-art UAV tracking algorithms to enhance the robustness. In our solution, all video frames used for tracking will be processed by the reconstruction module first. The reconstruction process aims to minimize the negative impacts of adversarial attacks on the tracking performance while not affecting clean video frames (i.e., not being attacked). Thus, UAV tracking can be performed effectively even if it is under attack. The anomaly detection module is designed on top of our reconstruction module, which helps the system become aware of attacks and prepare for further actions. Specifically, we propose a reference generator to construct dynamic adversarial reference samples based on the reconstructed frames. The feature maps of the original input, reconstructed frame, and adversarial reference sample are extracted, which are then used for dynamic similarity measurement. The original input is considered as abnormal if it has a higher similarity with the adversarial reference compared with the reconstructed frame.

We evaluated our solution using the recently proposed attack [11] and its variants towards UAV tracking with three widely adopted UAV tracking benchmarks, including UAVTrack112 [17], UAV123 [18], and UAVDT [19]. The evaluation results demonstrate that our solution can significantly boost the tracking performance in the adversarial environment, achieving 97.4% and 96.3% performance (by average) of the original tracking in terms of precision and success rate. For the detec-

2375-0197/24/$31.00 ©2024 IEEE
DOI 10.1109/ICTAI62512.2024.00136

tion of attacks, our solution can achieve an average detection rate (DR) of 98.9% and an average false positive rate (FPR) of 4.23% under different settings.

The rest of this paper is organized as follows: We review and discuss related works in Section II. In Section III, we present the detailed construction of our solution. The evaluation of our design is presented in Section IV. We conclude this paper in Section V.

II. Related Works

A. UAV Object Tracking

Deep learning-based UAV object tracking methods have attracted much attention due to the developments of deep-learning theories and increased computational power [1]–[6]. Zhang *et al.* [1] introduced a novel coarse-to-fine deep scheme aimed at mitigating aspect ratio variations in UAV tracking, where the coarse-tracker initially generates an initial estimate of the target object, followed by the learning of a sequence of actions to finely adjust the four boundaries of the bounding box. To address the challenge of long-distance UAV detection and tracking, Li *et al.* [2] proposed an algorithm using image super-resolution that employs a saliency transformation algorithm to concentrate on the suspected area and then construct a generative adversarial network on the Region of Interest to achieve super-resolution, enhancing weak targets and restoring high-resolution details of target features.

In recent years, Siamese trackers [3]–[6] become popular due to their ability to achieve a good balance between accuracy and efficiency. Li *et al.* [3] introduced SiamRPN++ tracker that integrates a cropping residual unit and a spatial-aware sampling strategy, allowing the Siamese Region Proposal Network (SiamRPN) [4] framework to leverage modern backbones and improve the performance of the Siamese tracker. To meet real-time processing demands on resource-constrained UAV platforms, Xing *et al.* [5] proposed a Siamese Transformer Pyramid Network, which combines the advantages of Convolutional Neural Network and transformer architectures by leveraging the inherent feature pyramid of a lightweight network ShuffleNetV2 and reinforcing it with a transformer to establish a robust, target-specific appearance model. Fu *et al.* [6] introduced a novel Siamese Anchor Proposal Network (SiamAPN) that consists of two stages, where the first stage is for high-quality anchor proposal generation and the second stage is for refining the anchor proposal.

B. Adversarial Tracking Attacks

Deep learning-based object-tracking methods are vulnerable to adversarial attacks that aim to mislead the output prediction by adding imperceptible perturbations to the input video frames. The existing tracking attack approaches can be classified into two categories: 1) iterative optimization-based attacks [7], [8] and 2) deep neural network (DNN)-based attacks [9]–[11]. For iterative optimization-based attacks, both [7] and [8] apply iterative optimization algorithms such as gradient descent to generate adversarial perturbations. Wiyatno *et al.* [7] presented a Physical Adversarial Texture attack method to generate pixel perturbations via minibatch gradient descent optimization to fool the GOTURN [20] tracker. Considering the efficiency of real-time attacks on trackers, Guo *et al.* [8] proposed a spatial-aware online incremental attack algorithm that conducts spatial-temporal sparse incremental perturbations in real-time, effectively minimizing the perceptibility of the adversarial attack.

Unlike iterative optimization-based attacks that need to repeat several iterations to generate optimized perturbations, DNN-based attacks train a DNN model offline as an adversary generator to generate perturbations at one step. Liang *et al.* [9] presented a Fast Attack Network that combines drift loss and embedded feature loss as the loss function to attack the Siamese trackers. Yan *et al.* [10] designed a cooling-shrinking attack method to deceive SiamRPN-based trackers by adding perturbations to the search regions and cool hot regions where the targets appear on the heatmap, which can lead the predicted bounding box to shrink and make the target invisible to trackers. Rather than adding perturbations directly on the original video frames, Fu *et al.* [11] designed an adaptive attack approach based on the image-resampling method, which consists of downsampling the original input frame and introducing small perturbations during the upsampling process to generate the adversarial sample.

C. Robustness Enhancement Against Attacks

The strategies to enhance the robustness of DL-based object trackers against malicious adversarial attacks can be mainly classified into two categories including adversarial training [12], [13] and input reconstruction [14]–[16]. In the adversarial training approach, the object trackers are trained both on clean and adversarial data to improve their robustness. Song *et al.* [12] proposed a visual tracking algorithm via adversarial learning to address the imbalance problem between positive and negative samples in adversarial training to enhance the robustness of the visual tracking model. To improve the running speed, Zhong *et al.* [13] introduced a real-time tracking algorithm that incorporates feature map masking alongside adversarial learning with a random mechanism. However, the adversarial training approach raises the risk of decreasing accuracy on clean data due to the exposure to adversarial data during model training.

Existing input reconstruction methods are mainly designed for image classification tasks to mitigate the adversarial perturbations on images. Yuan *et al.* [14] developed the ensemble generative cleaning with feedback loops method for the effective defense of DNNs by destroying the attack pattern first and then reconstructing the clean version of the original image. Ho *et al.* [15] designed an adversarial defense with local implicit functions to remove adversarial perturbations using localized manifold projections by leveraging an encoder for per-pixel features and a local implicit module for neighborhood-based predictions. An adversarial purification method, termed DiffPure, presented in [16] leverages diffusion models for adversarial purification by adding noise to adversarial samples through a forward diffusion process and then recovering the clean images via a reverse generative process. [14]–[16] focus on enhancing the robustness of image classifiers and cannot be directly applied in object tracking tasks since their task goals are different.

III. Methods

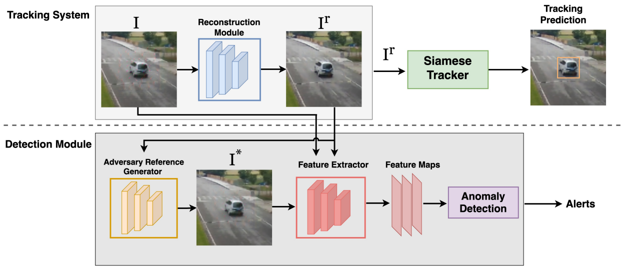

Fig. 1: Overall Framework.

The overall design of our solution is illustrated in Fig. 1. Our design focuses on the reconstruction module and anomaly detection module, which can be integrated with a state-of-the-art UAV object tracker, such as the Siamese tracker. In other words, our solution can be easily plugged into existing UAV tracking systems to enhance their robustness. Instead of directly feeding the inputs into the tracker, our solution first processes them using the reconstruction module, which addresses the impacts of adversarial attacks (if any). Meanwhile, our solution has the detection module run in parallel to identify inputs that have been manipulated by attacks.

The design of the reconstruction module adopts the idea of generative adversarial networks. We train it using newly designed loss functions based on state-of-the-art Siamese tracking algorithms, ensuring the reconstructed frames maintain high tracking accuracy for both attacked and clean inputs. The detection module is developed to identify anomalies and trigger alerts for further countermeasures. Our anomaly detection strategy involves comparing the input frame with both its reconstructed version and an adversarial reference sample using the Structural Similarity Index Measure (SSIM) to assess similarity. Due to dynamic backgrounds and moving targets in video frames, static reference samples are ineffective. Therefore, an adversary reference generator is adopted to construct a reference sample based on the current frame. Given that tracking attacks inject imperceptible perturbations making direct frame comparisons challenging, we use a pre-trained feature extractor to extract deep feature maps differentiating between clean and attacked frames.

A. Model Architecture

U-Net [21] architecture is adopted to construct the reconstruction module and adversary reference generator due to its ability to capture context at multiple scales while preserving spatial information, making it effective for pixel-level tasks. The U-Net architecture first downsamples the input search regions 8 times by a factor of 2 to capture high-level features and reduce the spatial dimensions of the input search regions and then upsamples the low-resolution feature maps to match the original input size. To handle varying search region sizes N×N used by different Siamese trackers, such as 255×255 for SiamRPN++ [3] and 287×287 for SiamAPN [6], and in long-term scenarios where sizes range from 255×255 to 831×831, we set the model's input resolution to 512×512 and the resolution gaps are managed using bilinear interpolation. The model architecture is shown in Fig. 2.

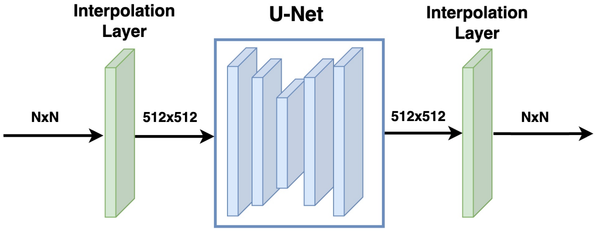

Fig. 2: Model Architecture.

B. Reconstruction Module

Following the idea of generative adversarial network [22] that two neural networks contest with each other where one agent's gain is another agent's loss, the key point of the reconstruction module is to minimize loss between the tracking predictions of reconstructed frames and clean frames, which is contrary to the goal of attackers that aims to maximize the difference between predictions of attacked frames and clean frames under some constraint. In the tracking process of the Siamese tracker, features of the template and search region are extracted through its backbone network. Based on these features, a classification score map and a regression map are generated to produce the tracking predictions. Towards minimizing the difference between these two maps of the clean frame and its reconstructed version, two loss functions L_{score} and L_{drift} are designed for reconstruction module training to achieve accurate tracking predictions for reconstructed frames. The L_2 norm distance and similarity score loss L_{sim} between reconstructed frames and clean frames are added as an additional loss to enable the module to generate frames similar to clean ones, maintaining the tracking accuracy on clean frames. The overall loss for the reconstruction module is defined as $L = L_{score} + L_{drift} + L_2 + L_{sim}$. The reconstruction module is trained with the deployed tracker weights fixed to minimize the loss L. The definitions for each loss function are provided below.

1) Score Loss: The classification score map from the Siamese tracker is reshaped to $\mathbb{R}^{H \times W \times 2}$ after applying the softmax function, which represents the target probability $\mathbf{P_t}$ and background probability $\mathbf{P_b}$ of each anchor of the predicted bounding box, i.e. the center point of the bounding box. The target area $\mathbf{T}$ and background area $\mathbf{B}$ in the clean frame $\mathbf{I}^c$ can be expressed as:

$$\begin{aligned}\mathbf{T} &= \mathbf{I}^c[\mathbf{P}_t^c > \epsilon] \\ \mathbf{B} &= \mathbf{I}^c[\mathbf{P}_b^c < -\epsilon]\end{aligned} \tag{1}$$

where ϵ is a preset threshold, $\mathbf{P}_t^c$ and $\mathbf{P}_b^c$ are the target probability and background probability of each anchor in the clean frame.

The score loss function is defined as follows:

$$L_{score} = \frac{1}{N}(|\mathbf{P}_t^r[\mathbf{T}] - \mathbf{P}_t^c[\mathbf{T}]| + |\mathbf{P}_b^r[\mathbf{B}] - \mathbf{P}_b^c[\mathbf{B}]|) \tag{2}$$

where N is the batch size, $\mathbf{P}_t^r$ and $\mathbf{P}_b^r$ are the target probability and background probability of each anchor in the reconstructed frame. The L_{score} function aims to reduce the difference between the confidence scores of the target and background area in clean and reconstructed frames.

2) Drift Loss: The regression map $\boldsymbol{R} \in \mathbb{R}^{H \times W \times 4}$ has four dimensions $\mathbf{R}(x)$, $\mathbf{R}(y)$, $\mathbf{R}(w)$, $\mathbf{R}(h)$. $\mathbf{R}(x)$ and $\mathbf{R}(y)$ represent the center position of the bounding box. $\mathbf{R}(w)$ and $\mathbf{R}(h)$ represent the size of the bounding box. The drift loss function is defined as follows:

$$\begin{aligned} L_{drift} = \frac{1}{N}\{\sum_{\mathbf{T}} |\mathbf{R}^r(w)\mathbf{R}^r(h) - \mathbf{R}^c(w)\mathbf{R}^c(h)| \\ + \sum_{\mathbf{T}}((\mathbf{R}^r(x) - \mathbf{R}^c(x))^2 + (\mathbf{R}^r(y) - \mathbf{R}^c(y))^2)\} \end{aligned} \tag{3}$$

where $\mathbf{R}^r$ and $\mathbf{R}^c$ are regression maps of the reconstructed frame and clean frame, respectively.

L_{drift} function is designed to make the bounding box size and center position in the reconstructed frame close to the ones in the clean frame. Here, we only consider the potential bounding boxes within the activated target area in the frame.

3) L_2 Norm Loss: The L_2 norm loss function aims to make the reconstructed frame similar to the clean frame, which is defined as follows:

$$L_2 = \frac{\gamma}{N}\|\mathbf{I}^c - \mathbf{I}^r\| \tag{4}$$

where γ is the weight of L_2 loss, $\mathbf{I}^c$ and $\mathbf{I}^r$ are the clean and reconstructed frame, respectively.

4) Similarity Loss: The L_{sim} loss is computed using the SSIM between the feature maps of the clean frame and its reconstructed version. Since SSIM ranges from -1 to 1, where 1 denotes identical feature maps, the L_{sim} loss function is defined as 1 minus the SSIM to convert the SSIM score into a similarity loss for model training. The formula for L_{sim} loss is given below:

$$L_{sim} = \frac{1}{N}(1 - sim(\mathbf{S}^r, \mathbf{S}^c)) \tag{5}$$

where $\mathbf{S}^r$ and $\mathbf{S}^c$ are the feature maps of the reconstructed frame and clean frame, respectively.

L_{sim} loss function aims to improve the similarity score between clean and reconstructed frames to reduce the false positive rate in anomaly detection.

C. Feature Extraction

To differentiate between clean and attacked frames, we analyze feature spaces from various intermediate layers of pre-trained ResNet-50 [23], which serves as the backbone for state-of-the-art Siamese trackers. Since feature spaces at different layers affect anomaly detection performance [24]–[27] and deep layers of networks pre-trained on ImageNet are biased towards natural image classification [28], we aim to select feature maps that balance effectiveness and computational efficiency in real-time UAV tracking. As shown in Table I, feature maps from layers conv1 and conv5_x of ResNet-50 effectively represent differences between clean and attacked frames. Considering the computational costs on resource-constrained UAV tracking platforms, feature maps from conv1 layer are selected for similarity calculations in anomaly detection.

TABLE I: Pixel Distributions at Different Feature Levels

Clean Pixel Distribution					
	conv1	conv2_x	conv3_x	conv4_x	conv5_x
Mean	36.41	46.19	56.74	75.59	96.63
Median	24.0	42.0	54.0	65.0	86.0
Std.	38.03	29.01	29.27	49.93	51.90
Attacked Pixel Distribution					
	conv1	conv2_x	conv3_x	conv4_x	conv5_x
Mean	21.06	45.89	54.44	78.53	83.72
Median	16.0	44.0	53.0	75.0	78.0
Std.	15.42	25.36	29.14	41.89	41.18
Difference Δ					
	conv1	conv2_x	conv3_x	conv4_x	conv5_x
Δ Mean	15.35	0.30	2.30	-2.94	12.91
Δ Median	8.0	-2.0	1.0	-10.0	8.0
Δ Std.	22.61	3.65	0.13	8.04	10.72

D. Adversary Reference Generator

The adversary reference generator shares the same model architecture as the reconstruction module but utilizes different loss functions to generate adversarial reference samples for anomaly detection. It takes the reconstructed frame from the reconstruction module as input to construct its corresponding adversarial sample. We aim to develop an adversary reference generator that constructs adversarial samples similar to the attacked frames to improve the anomaly detection rate while differing from the clean frames to decrease the false positive rate. To achieve this goal, we design two similarity-based loss functions and L_2 norm loss function for the training of the adversary reference generator.

1) L_2 Norm Loss: The L_2 norm loss function is to make the generated adversarial samples similar to the attacked frames, which is defined as follows:

$$L_2 = \frac{\gamma}{N}\|\mathbf{I}^a - \mathbf{I}^*\| \tag{6}$$

where γ is the weight of L_2 loss, $\mathbf{I}^a$ is the attacked frame and $\mathbf{I}^*$ is the generated adversarial sample.

2) Similarity Loss 1: The L_{sim1} loss is calculated based on the SSIM between the feature maps of the adversarial sample and the attacked frame.

$$L_{sim1} = \frac{\alpha}{N}(1 - sim(\mathbf{S}^a, \mathbf{S}^*)) \quad (7)$$

where α is the weight of L_{sim1} loss, $\mathbf{S}^a$ and $\mathbf{S}^*$ are the feature maps of the attacked frame and adversarial sample, respectively.

3) Similarity Loss 2: The L_{sim2} loss is the SSIM between the feature maps of the adversarial sample and the clean frame. The L_{sim2} loss is defined as follows:

$$L_{sim2} = \frac{\beta}{N} sim(\mathbf{S}^*, \mathbf{S}^c) \quad (8)$$

where β is the weight of L_{sim2} loss, $\mathbf{S}^*$ and $\mathbf{S}^c$ are the feature maps of adversarial sample and clean frame, respectively. During the training, the L_{sim2} loss is minimized to make the adversarial samples different from clean frames, contributing to decreasing the false positive rate in anomaly detection.

IV. Evaluation

A. Experiment Setup

In our experiments, the weight γ in Equations 4 and 6 is set to 700, α in Equation 7 to 3, and β in Equation 8 to 2. The batch size N is 128. We use the state-of-the-art SiamRPN++ [3] both for training our reconstruction module and for testing. The GOT-10K [29] dataset is downsampled by selecting one frame every ten frames from each video. A subset of 180 videos from this downsampled dataset is used as the validation set, and the remaining videos form the training dataset. To evaluate our method under various attacks, we trained a new variant of the Adaptive Adversarial (Ad^2) attack [11] using the U-Net architecture with same loss functions. Both Ad^2 attack and our attack are used to produce attacked frames for training the reconstruction module and adversarial reference generator. The effectiveness of our method is evaluated by implementing Ad^2 attack and our attack on three UAV benchmarks: UAVTrack112 [17], UAV123 [18], and UAVDT [19]. The Ad^2 attack and our attack are implemented continuously across all frames, except the first frame as the initial template. Additionally, we test two attack strategies: continuous attacks on 50% of the frames over one period and over two periods within the video.

B. Evaluation Metrics

To evaluate anomaly detection performance, we use true positive rate (TPR), false positive rate (FPR), false negative rate (FNR), and true negative rate (TNR) to measure the accuracy of anomaly detection. We evaluate the tracking performance with precision and success rate as metrics [30]. Precision is determined by the center location error (CLE), which is the Euclidean distance between the predicted center location of the target object and the ground truth. CLE is averaged over all frames of a video to summarize precision performance. Success rate focuses on the intersection over union (IoU) between the ground truth and predicted bounding box. Success rate performance is measured by counting the number of successful frames where the IoU is greater than a given threshold.

To better assess the anomaly detection performance, we include the detection rate (DR) [31], which is calculated as the ratio of successfully detected attacks to the total number of successful attacks. In our experiments, a successful attack is defined as one that reduces the IoU between the ground truth and predicted bounding box by at least 10% compared to the clean prediction.

C. Threshold Selection

The countereffects between the reconstruction module and adversary reference generator on clean frames can cause the adversarial reference samples to appear similar to the original clean frames, resulting in high FPR in anomaly detection. To address this issue and improve detection reliability, we implement a two-stage checking strategy for anomaly detection with a similarity threshold, based on validation set results.

Let $sim1$ represent the similarity between the input frame and its reconstructed version, and $sim2$ denote the similarity between the input frame and the adversarial reference. In our experiments on the validation set, 89% of clean frames have $sim1$ and $sim2$ both over 0.9, with 32% showing $sim2 > sim1$ and a mean difference of 0.01, leading to false alarms if detection relies solely on higher similarity. For attacked frames, 56% have both $sim1$ and $sim2$ over 0.9, with 99.8% showing $sim2 > sim1$ and a mean difference of 0.05. Thus, FPR primarily arises when $sim1$ and $sim2$ are close and above 0.9. Driven by this observation, we adopt a two-stage checking strategy based on $sim1$ and $sim2$ values in the anomaly detection process. If not both $sim1$ and $sim2$ are greater than 0.9 simultaneously, detection is made in the first stage based on the higher similarity. If both exceed 0.9, a second stage checks if $sim2 - sim1 \geq k$. If this condition is met, the frame is considered as abnormal; otherwise, it is normal.

To determine the optimal threshold that balances TPR and FPR, we evaluated various thresholds on the validation set using $TPR \times (1 - FPR)$, a standard metric for binary classification performance [32]. Thresholds ranging from 0.01 to 0.05 in 0.01 increments were evaluated, with 0.01 and 0.05 representing the average similarity differences for clean and attacked frames, respectively. The threshold of $k = 0.04$ provided the highest $TPR \times (1 - FPR)$ value of 0.876. Therefore, we set the threshold k to 0.04 for evaluating our method on test datasets.

D. Results

1) Attack Recovery Performance: The overall tracking performance of three datasets is summarized by the success and precision plots as shown in Fig. 3. Precision plots and success plots are widely adopted for the evaluation of tracking performance [33], [34]. The precision plots in the first row

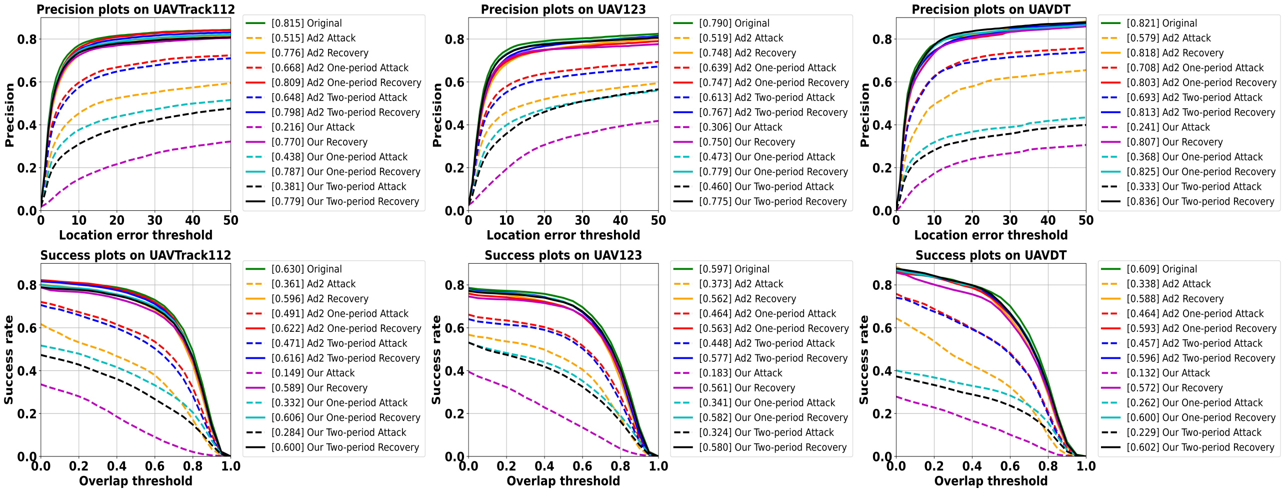

Fig. 3: Overall tracking performance of SiamRPN++ tracker under attacks (dashed lines) and recovery (solid lines) on UAVTrack112, UAV123, and UAVDT.

TABLE II: Summary of recovery performance on precision

Precision			
	Ad^2 **Attack [11]**		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.815	0.515 (63.2%)	0.776 (95.2%)
UAV123	0.790	0.519 (65.7%)	0.748 (94.7%)
UAVDT	0.821	0.579 (70.5%)	0.818 (99.6%)
	Ad^2 **One-Period Attack**		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.815	0.668 (82.0%)	0.809 (99.3%)
UAV123	0.790	0.639 (80.9%)	0.747 (94.6%)
UAVDT	0.821	0.708 (86.2%)	0.803 (97.8%)
	Ad^2 **Two-Period Attack**		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.815	0.648 (79.5%)	0.798 (97.9%)
UAV123	0.790	0.613 (77.6%)	0.767 (97.1%)
UAVDT	0.821	0.693 (84.4%)	0.813 (99.0%)
	Our Attack		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.815	0.216 (26.5%)	0.770 (94.5%)
UAV123	0.790	0.306 (38.7%)	0.750 (94.9%)
UAVDT	0.821	0.241 (29.4%)	0.807 (98.3%)
	Our One-Period Attack		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.815	0.438 (53.7%)	0.787 (96.6%)
UAV123	0.790	0.473 (59.9%)	0.779 (98.6%)
UAVDT	0.821	0.368 (44.8%)	0.825 (100%)
	Our Two-Period Attack		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.815	0.381 (46.7%)	0.779 (95.6%)
UAV123	0.790	0.460 (58.2%)	0.775 (98.1%)
UAVDT	0.821	0.333 (40.6%)	0.836 (102%)

TABLE III: Summary of recovery performance on success rate

Success Rate			
	Ad^2 **Attack [11]**		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.630	0.361 (57.3%)	0.596 (94.6%)
UAV123	0.597	0.373 (62.5%)	0.562 (94.1%)
UAVDT	0.609	0.338 (55.5%)	0.588 (96.6%)
	Ad^2 **One-Period Attack**		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.630	0.491 (77.9%)	0.622 (98.7%)
UAV123	0.597	0.464 (77.7%)	0.563 (94.3%)
UAVDT	0.609	0.464 (76.2%)	0.593 (97.4%)
	Ad^2 **Two-Period Attack**		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.630	0.471 (74.8%)	0.616 (97.8%)
UAV123	0.597	0.448 (75.0%)	0.577 (96.6%)
UAVDT	0.609	0.457 (75.0%)	0.596 (97.9%)
	Our Attack		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.630	0.149 (23.7%)	0.589 (93.5%)
UAV123	0.597	0.183 (30.7%)	0.561 (94.0%)
UAVDT	0.609	0.132 (21.7%)	0.572 (93.9%)
	Our One-Period Attack		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.630	0.332 (52.7%)	0.606 (96.2%)
UAV123	0.597	0.341 (57.1%)	0.582 (97.5%)
UAVDT	0.609	0.262 (43.0%)	0.600 (98.5%)
	Our Two-Period Attack		
	Ori.	**Att. (% of Ori.)**	**Reconstruct (% of Ori.)**
UAVTrack112	0.630	0.284 (45.1%)	0.600 (95.2%)
UAV123	0.597	0.324 (54.3%)	0.580 (97.2%)
UAVDT	0.609	0.229 (37.6%)	0.602 (98.9%)

show the percentage of frames whose estimated location is within a given threshold of pixel distance from the ground truth. The location error thresholds range from 0 to 50 are used for our evaluation. For each tracking condition, the representative precision score shown next to the plot is taken at a location error threshold of 20 pixels. The success plots in the second row display the ratio of successful frames across IoU thresholds ranging from 0 to 1. Since evaluating the tracking performance using a single success rate at a specific threshold may not be fair or representative, the area under the curve of the success plot is used as the representative score to provide a more comprehensive assessment across varying overlap thresholds. The solid lines representing the recovery tracking performance are close to the original performance in

TABLE IV: Tracking performance without attack

Precision		
	Ori.	**Reconstruct (% of Ori.)**
UAVTrack112	0.815	0.808 (99.1%)
UAV123	0.790	0.772 (97.7%)
UAVDT	0.821	0.827 (100%)
Success Rate		
	Ori.	**Reconstruct (% of Ori.)**
UAVTrack112	0.630	0.625 (99.2%)
UAV123	0.597	0.580 (97.2%)
UAVDT	0.609	0.613 (100%)

the green line, indicating that our method can bring back the tracking performance close to the original scenarios.

Table II and Table III present the reconstruction tracking performance recovering from different attacks. The results show that our attack as a variant of Ad^2 attack can reduce the precision and success rate of SiamRPN++ by over 50%. Our solution can restore the tracking performance with an average precision of 97.4% and an average success rate of 96.3% of original scenarios. The tracking performance with reconstructed frames without attack is provided in Table IV. The reconstructed frames of original clean frames without attack can maintain an average precision of 98.9% and an average success rate of 98.8% of original scenarios. The results in Table II, III, and IV indicate that our reconstruction module can eliminate the adversarial impacts for attacked frames while maintaining the tracking accuracy for clean frames, demonstrating the effectiveness of our method in enhancing the robustness of UAV tracking systems.

TABLE V: Anomaly Detection Accuracy

Anomaly Detection on UAVTrack112					
	FPR	FNR	TNR	TPR	DR
Ad^2 Attack [11]	N/A	0.088	N/A	0.912	0.985
Ad^2 One-Period Attack	0.047	0.093	0.953	0.907	0.994
Ad^2 Two-Period Attack	0.045	0.094	0.955	0.906	0.992
Our Attack	N/A	0.052	N/A	0.948	0.991
Our One-Period Attack	0.058	0.051	0.942	0.949	0.997
Our Two-Period Attack	0.051	0.053	0.949	0.947	0.994
Anomaly Detection on UAV123					
	FPR	FNR	TNR	TPR	DR
Ad^2 Attack [11]	N/A	0.072	N/A	0.928	0.992
Ad^2 One-Period Attack	0.051	0.068	0.949	0.932	0.996
Ad^2 Two-Period Attack	0.044	0.073	0.956	0.927	0.995
Our Attack	N/A	0.058	N/A	0.942	0.992
Our One-Period Attack	0.052	0.053	0.948	0.947	0.996
Our Two-Period Attack	0.044	0.051	0.956	0.949	0.996
Anomaly Detection on UAVDT					
	FPR	FNR	TNR	TPR	DR
Ad^2 Attack [11]	N/A	0.112	N/A	0.888	0.954
Ad^2 One-Period Attack	0.034	0.110	0.966	0.890	0.989
Ad^2 Two-Period Attack	0.027	0.122	0.973	0.878	0.974
Our Attack	N/A	0.046	N/A	0.954	0.987
Our One-Period Attack	0.032	0.042	0.968	0.958	0.991
Our Two-Period Attack	0.026	0.045	0.974	0.955	0.992

2) Anomaly Detection Accuracy: Table V presents the detection accuracy of our method across various attacks. The Ad^2 attack from [11] and our attack are applied to all video frames. Since all frames are attacked, FPR and TNR are not applicable for evaluating detection accuracy in these cases. The results in Table V show that our detection method achieves a high detection rate above 0.95 and a false positive rate below 0.06 across all test datasets, demonstrating the reliability of our detection module.

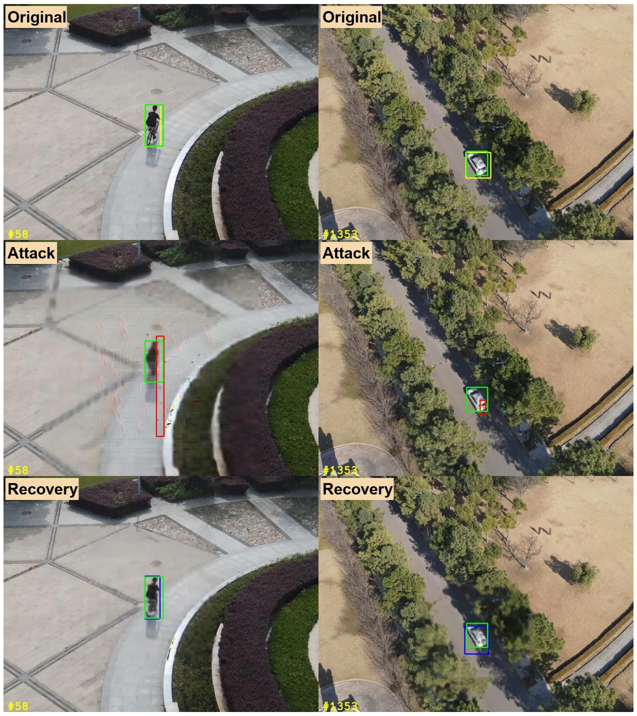

Fig. 4: Tracking Examples.

3) Tracking Examples: We take frames from two videos in the UAVTrack112 dataset to show the effectiveness of our solution in restoring tracking performance under the Ad^2 attack in real-world tracking scenarios. The first row in Fig. 4 displays the original tracking results for bike and car objects in the two videos: green boxes represent ground truths, and yellow boxes denote original tracking predictions. In the second row, red boxes indicate tracking predictions deviating from the ground truths under attack. The third row shows recovery tracking predictions by our solution using reconstructed frames, depicted by blue boxes overlapping with the ground truths.

V. Conclusion

In this paper, we present a reconstruction-based solution designed to enhance the robustness of UAV tracking systems against adversarial attacks. Our approach involves a reconstruction module to reconstruct input frames to mitigate adversarial impacts on tracking performance and an anomaly detection module to identify attacks and raise alerts. We evaluate our solution against the recently proposed Ad^2 attack and its variants across three different datasets. The results of our comprehensive evaluations demonstrate that our solution is highly effective in mitigating adversarial attacks, significantly improving the reliability and security of UAV tracking systems.

Acknowledgment

This work is supported by the US National Science Foundation awards OAC-2229976 and CNS-2318711.

References

[1] W. Zhang, K. Song, X. Rong, and Y. Li, "Coarse-to-fine uav target tracking with deep reinforcement learning," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 4, pp. 1522–1530, 2018.

[2] B. Li, S. Qiu, W. Jiang, W. Zhang, M. Le *et al.*, "A uav detection and tracking algorithm based on image feature super-resolution," *Wireless Communications and Mobile Computing*, vol. 2022, 2022.

[3] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4277–4286, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:57189581

[4] 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.

[5] D. Xing, N. Evangeliou, A. Tsoukalas, and A. Tzes, "Siamese transformer pyramid networks for real-time uav tracking," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2022, pp. 2139–2148.

[6] C. Fu, Z. Cao, Y. Li, J. Ye, and C. Feng, "Siamese anchor proposal network for high-speed aerial tracking," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 510–516.

[7] R. R. Wiyatno and A. Xu, "Physical adversarial textures that fool visual object tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4822–4831.

[8] 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," in *European conference on computer vision*. Springer, 2020, pp. 202–219.

[9] S. Liang, X. Wei, S. Yao, and X. Cao, "Efficient adversarial attacks for visual object tracking," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16*. Springer, 2020, pp. 34–50.

[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] C. Fu, S. Li, X. Yuan, J. Ye, Z. Cao, and F. Ding, "Ad 2 attack: Adaptive adversarial attack on real-time uav tracking," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5893–5899.

[12] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, R. W. Lau, and M.-H. Yang, "Vital: Visual tracking via adversarial learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8990–8999.

[13] H. Zhong, X. Yan, Y. Jiang, and S.-T. Xia, "Improved real-time visual tracking via adversarial learning," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 1853–1857.

[14] J. Yuan and Z. He, "Ensemble generative cleaning with feedback loops for defending adversarial attacks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 581–590.

[15] C.-H. Ho and N. Vasconcelos, "Disco: Adversarial defense with local implicit functions," *Advances in Neural Information Processing Systems*, vol. 35, pp. 23 818–23 837, 2022.

[16] W. Nie, B. Guo, Y. Huang, C. Xiao, A. Vahdat, and A. Anandkumar, "Diffusion models for adversarial purification," *arXiv preprint arXiv:2205.07460*, 2022.

[17] C. Fu, Z. Cao, Y. Li, J. Ye, and C. Feng, "Onboard real-time aerial tracking with efficient siamese anchor proposal network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–13, 2021.

[18] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 445–461.

[19] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li, W. Zhang, Q. Huang, and Q. Tian, "The unmanned aerial vehicle benchmark: Object detection and tracking," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 370–386.

[20] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 749–765.

[21] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[23] 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 (CVPR)*, June 2016.

[24] Y. Cao, X. Xu, Z. Liu, and W. Shen, "Collaborative discrepancy optimization for reliable image anomaly localization," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 11, pp. 10 674–10 683, 2023.

[25] D. Gudovskiy, S. Ishizaka, and K. Kozuka, "Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2022, pp. 98–107.

[26] O. Rippel, P. Mertens, and D. Merhof, "Modeling the distribution of normal data in pre-trained deep features for anomaly detection," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 6726–6733.

[27] L. Heckler, R. König, and P. Bergmann, "Exploring the importance of pretrained feature extractors for unsupervised anomaly detection and localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2917–2926.

[28] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, "Towards total recall in industrial anomaly detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 14 318–14 328.

[29] 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*, vol. 43, no. 5, pp. 1562–1577, 2019.

[30] 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.

[31] A. Aldahdooh, W. Hamidouche, S. A. Fezza, and O. Déforges, "Adversarial example detection for dnn models: A review and experimental comparison," *Artificial Intelligence Review*, vol. 55, no. 6, pp. 4403–4462, 2022.

[32] X. Liu, "Classification accuracy and cut point selection," *Statistics in medicine*, vol. 31, no. 23, pp. 2676–2686, 2012.

[33] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 8, pp. 1619–1632, 2010.

[34] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part IV 12*. Springer, 2012, pp. 702–715.