A Pluggable Solution For Robust UAV Tracking Against Attacks

Mengjie Jia, *Student Member, IEEE,* Yanyan Li, *Member, IEEE,* Houbing Herbert Song, *Fellow, IEEE,* Jiawei Yuan, *Senior Member, IEEE*

***Abstract*—Unmanned Aerial Vehicles (UAVs) are increasingly integrated into vehicular systems for applications such as autonomous surveillance, traffic monitoring, and navigation. These applications rely on real-time object tracking, which has been significantly enhanced by deep learning (DL)-based models. However, DL-based trackers remain highly vulnerable to adversarial attacks, where imperceptible perturbations can severely degrade tracking accuracy and reliability. To address these challenges, we propose a pluggable defense solution designed to enhance the robustness of UAV tracking systems without modifying existing tracking architectures. Our approach leverages a dual-level optimization strategy to mitigate adversarial perturbations at both feature and decision levels, ensuring resilient tracking performance. Implemented as a pre-processing stage, our solution can be seamlessly integrated with various UAV tracking systems. We evaluate our approach against multiple adversarial attacks across three widely used UAV tracking benchmarks: UAVTrack112, UAV123, and UAVDT. Experimental results demonstrate that our pluggable solution effectively restores tracking accuracy and improves robustness under various adversarial attacks without sacrificing tracking performance in original (attack-free) scenarios. Real-world tests on a UAV platform validate the efficiency and practicality of our method. Comprehensive results indicate our solution can strengthen UAV tracking in real-world applications and ensure reliable performance in adversarial environments.**

***Index Terms*—UAV Object Tracking, UAV Security, Deep Learning.**

I. Introduction

Unmanned Aerial Vehicles (UAVs) have been widely integrated into vehicular systems to enhance intelligent transportation technologies due to their high mobility and advanced sensing capabilities [2]–[5]. As part of modern vehicular networks, UAVs provide a mobile and adaptable sensing platform that supports critical applications such as autonomous surveillance [6]–[11], traffic monitoring and management [12]–[16] in smart cities, and pedestrian safety monitoring [17]–[19]. A fundamental capability enabling these UAV applications is object tracking, which allows UAVs to detect, follow, and analyze the movement of objects in real time. In autonomous surveillance, object tracking enables UAVs to continuously monitor dynamic targets such as vehicles and pedestrians [6]–[11]. For traffic monitoring, UAV-based tracking assists in vehicle movement analysis, congestion detection, and law enforcement support [12]–[16]. Additionally, UAV-based autonomous navigation depends on object tracking to avoid obstacles, maintain stable flight paths, and interact safely with dynamic environments [20]–[23].

With the rapid advancement of AI techniques, deep learning (DL)-based trackers have been integrated with UAVs to enhance the performance of UAV-based tracking systems [24]–[31]. DL-based trackers can process complex patterns from various onboard UAV sensors, such as cameras, LiDAR, and thermal sensors. They can effectively handle variations in lighting, scale, orientation, and background clutter, making them well-suited for dynamic and diverse environments, even in challenging scenarios involving occlusions or rapid movements. However, AI and deep learning models are vulnerable to adversarial attacks and data manipulation due to their sensitivity to input perturbations [32]–[36]. These vulnerabilities arise from the data-driven nature of their learning processes and the high-dimensional complexity of their feature spaces. While integrating AI into UAV tracking systems brings significant advancements and capabilities, it also introduces critical security challenges since DL-based trackers employed in UAV tracking systems share these vulnerabilities, which can compromise tracking accuracy and reliability in real-world applications. A compromised UAV tracking system could result in security failures, inaccurate surveillance, failed deliveries, or unreliable autonomous flight. Recent studies have demonstrated that adversarial perturbations applied to input data can significantly compromise the effectiveness of DL-based UAV tracking algorithms [37]–[42]. Most existing tracking attacks aim to mislead the tracking predictions by manipulating video frames with pixel-level adversarial perturbations taking state-of-the-art Siamese-based trackers as victim trackers. The perturbations are generated either by inverting the original training loss functions at tracking decision-level [37]–[40] or disrupting the intermediate feature space by maximizing the difference between feature maps of clean and adversarial frames to deceive the tracker at feature-level [41], [42].

Given the fact that object tracking is widely integrated into various UAV applications, it is crucial to improve the robustness of UAV tracking systems against potential attacks. Once the tracking system is disabled by external attacks,

The preliminary version of this paper appeared in the IEEE 36th International Conference on Tools with Artificial Intelligence (ICTAI 2024) [1].

This work was supported in part by the US National Science Foundation awards OAC-2309760, OAC-2229976, CNS-2318710, and CNS-2318711.

Mengjie Jia and Jiawei Yuan are with the Department of CIS, University of Massachusetts Dartmouth, emails: mjia@umassd.edu and jyuan@umassd.edu.

Yanyan Li is with the Department of CSE, California State University San Marcos, email: yali@csusm.edu.

H. H. Song is with the Department of Information Systems, University of Maryland, Baltimore County, email: songh@umbc.edu.

UAV tasks could fail and even result in severe accidents since the entire system is exposed to adversarial environments. Previous studies have demonstrated that methods such as adversarial training [43], [44] and input reconstruction [45]–[49] can effectively enhance the robustness of deep learning models. However, directly applying these approaches to UAV tracking systems poses several challenges. Adversarial training is computationally intensive and can degrade model performance due to the exposure to adversarial data during training. Meanwhile, most input reconstruction methods are tailored to image classification tasks and have limited effectiveness in object tracking scenarios. Furthermore, existing defense methods predominantly focus on mitigating decision-level attacks, often overlooking feature-level disruptions in UAV-based tracking systems. However, feature-level attacks can propagate through the network, leading to cascading vulnerabilities that affect both feature extraction and the final predictions. Therefore, addressing only decision-level attacks is insufficient for ensuring the robustness and reliability of UAV tracking systems. A comprehensive defense strategy should account for both feature-level and decision-level vulnerabilities to provide a more effective and resilient solution.

In this paper, we propose a robust pluggable tracking solution compatible with various Siamese-based trackers to enhance the performance of UAV tracking systems in adversarial environments. To mitigate the impact of attacks, we develop an input reconstruction module that purifies video frames by effectively eliminating adversarial perturbations and restoring them to their clean versions. The key motivation behind our design is to prevent feature variance and distortion caused by adversarial perturbations from propagating through the tracking pipeline to the prediction stage. To achieve this, our module employs a dual-level optimization strategy that operates at both feature and decision levels. This approach ensures effective mitigation of adversarial perturbations during intermediate feature extraction and final decision-making stages. Consequently, the reconstructed output serves as a robust input for the subsequent tracker, enabling accurate and reliable tracking predictions across diverse and challenging adversarial scenarios. Our reconstruction module is designed as a pluggable input pre-processing stage to be easily integrated into existing tracking systems, making it adaptable to different tracking architectures without modifications.

We evaluated our solution against various recently proposed tracking attacks towards UAV tracking systems, using three widely adopted UAV tracking benchmarks: UAVTrack112 [50], UAV123 [51], and UAVDT [52]. To test the robustness of our solution, we implemented two offline DNN-based decision-level attacks [39], [40], one online optimized decision-level attack [38], and one offline feature-level attack [42]. For assessing compatibility, our reconstruction module was integrated with three representative Siamese-based trackers: SiamRPN++ [27], SiamAPN [30], and SiamMask [31]. Our solution successfully restored the tracking performance of these trackers in various adversarial environments, achieving 95.8% and 93.2% of the original performance (on average) across three datasets for SiamRPN++ in terms of precision and success rate, 95.4% and 90.6% for SiamAPN, and 94.2% and 91.2% for SiamMask. Compared with recently proposed solutions [1], [49], our approach achieves comparable performance on decision-level attacks and significantly outperforms when handling feature-level attacks in terms of precision and success rate. These results demonstrate that our dual-level optimization strategy significantly improves recovery effectiveness. We conducted real-world tests on a UAV platform to evaluate the efficiency and practicality of our solution. Overall, the evaluation results indicate that our method substantially enhances tracking performance against different types of adversarial attacks and can be transferred to various Siamese-based trackers, achieving consistent and significant recovery improvements.

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. The transferability of our solution to trackers with different architectures is discussed in Section V. We conclude this paper in Section VI.

II. Related Works

A. UAV Object Tracking

The developments of deep-learning theories and computational power have encouraged the integration of deep-learning object tracking methods with UAV systems [24]–[31]. A novel coarse-tracker was introduced by Zhang *et al*. [24] to mitigate the effects of aspect ratio variations in UAV tracking. A coarse-to-fine deep scheme is adopted to finely adjust the boundaries of the bounding box by generating initial estimates of target objects and learning the sequential actions in the following frames. To tackle the challenge of long-distance UAV tracking, Li *et al*. [25] introduced an approach integrating image super-resolution with a saliency transformation algorithm. This method focuses on suspected regions by applying saliency transformation and subsequently employs a generative adversarial network on the identified Region of Interest to achieve super-resolution for enhancing weak targets and recovering high-resolution details of target features.

In recent years, Siamese trackers [26]–[31] have been widely used in applications due to their ability to achieve a good balance between accuracy and efficiency in real-time tasks. Siamese trackers adopt deep neural networks such as AlexNet[53] and ResNet50 [54] as the backbone to extract features of the template and search region patch for similarity learning and then employ a head network for feature fusion to produce the tracking predictions. Siam-FC [26] takes a fully convolutional network as the backbone and a cross-correlation layer as the head network for object localization. Region Proposal Network (RPN) is introduced in SiamRPN [28] to perform target region proposal extraction on the feature maps from the backbone network to generate two-branch response maps with tracking information including foreground-background classification score map and object bounding box regression map. SiamRPN++ [27] is an enhanced version of SiamRPN that employs a spatial-aware sampling strategy to aggregate feature representations from multiple layers

to realize cross-correlation operations and further improve tracking performance. To support the semi-supervised video segmentation tasks, SiamMask [31] extends the two-branch head network to three-branch by adding a segmentation mask branch to SiamRPN that encodes the feature information to generate a pixel-wise binary mask. Siamese Transformer Pyramid Network (SiamTPN) [29] is proposed to meet real-time processing requirements on resource-constrained UAV platforms. SiamTPN combines the strengths of Convolutional Neural Networks and transformers by leveraging ShuffleNetV2's lightweight feature pyramid and integrating a transformer to build a robust, target-specific appearance model. Siamese Anchor Proposal Network (SiamAPN) [30] consists of a two-stage architecture where the first stage generates high-quality anchor proposals adaptively, and the second stage refines these proposals for enhanced precision.

B. Adversarial Tracking Attacks

Existing tracking attacks [37]–[42] can be divided into different categories based on their perturbation learning strategies and adversarial objective functions targeting level. Based on the learning strategies, the tracking attacks can be classified into online interactive optimization-based [37], [38] and offline deep neural network (DNN)-based attacks [39], [40]. Both [37] and [38] implement online attacks by applying iterative optimization algorithms such as gradient descent to generate perturbations. To fool the GOTURN [55] tracker, Wiyatno *et al*. [37] proposed a Physical Adversarial Texture attack method utilizing a minibatch gradient descent algorithm to optimize the pixel perturbations. Guo et al. [38] proposed a spatial-aware online incremental attack algorithm to improve the efficiency of real-time attacks on trackers. This approach performs spatial-temporal sparse incremental perturbations in real time, minimizing the perceptibility of adversarial attacks while preserving their impact. Different from online iterative attacks, offline DNN-based attacks pre-train a DNN model as the adversary generator to generate perturbations at one step. Cooling-Shrinking Attack (CSA) [39] method is designed to deceive SiamRPN-based trackers by cooling hot regions where targets appear on the heatmap to make the target invisible and shrinking the predicted bounding box. Instead of adding perturbations directly on the original video frames, Adaptive Adversarial Attack (Ad^2Attack) [40] first downsamples the input frames and introduces perturbations during the upsampling process to generate adversarial samples.

In terms of the targeting level of adversarial objective functions, adversarial attacks can be classified into decision-level [37]–[40] and feature-level [41], [42]. [37]–[40] are decision-level attacks that generate perturbations to mislead tracking predictions towards confusing the response maps from the tracker's head network. SPARK [38] and CSA [39] aim to reduce the score gap between object and background in the foreground-background classification response map. Ad^2Attack [40] deceives the tracker by reversing the object and background scores. To force the predicted bounding box to drift away from the object, CSA [39] introduces shrinking loss, and Ad^2Attack [40] adds noisy offsets to disrupt the regression response map that includes the information to adjust the bounding box. Inspired by the fact that features from intermediate layers of the DNN model can affect the task-oriented decision, feature-level attacks have attracted attention recently by distorting feature maps of normal samples to generate adversarial perturbations. Motivated by the observation that features extracted from intermediate layers of different DNNs on different data for various tasks share strong similarities, Transferable Adversarial Perturbations (TAP) [42] introduces a loss function to maximize the relative distance between normal features of a sample and its adversarial counterparts extracted from intermediate layers of a DNN model. On the other hand, Pluggable Attack [41] corrupts the feature space from the backbone of trackers by disordering the feature distributions.

C. Robustness Enhancement Against Attacks

Enhancing the robustness of DL-based object trackers against adversarial attacks can generally be achieved through two main approaches: adversarial training [43], [44] and input reconstruction [45]–[49]. Adversarial training strengthens model resilience by exposing object trackers to both clean and adversarial data during training. Song et al. [43] introduced a visual tracking framework that integrates adversarial learning to address class imbalance issues in training, thereby enhancing robustness. To improve inference speed, Zhong et al. [44] developed a real-time tracking algorithm that combines adversarial learning with a feature map masking strategy and a randomized mechanism. Despite its effectiveness, adversarial training poses a risk of compromising accuracy on clean data due to exposure to adversarial examples during model training.

Input reconstruction focuses on mitigating adversarial perturbations by restoring corrupted inputs, which is a technique predominantly developed for image classification rather than object tracking. Yuan et al. [45] proposed a feedback-based ensemble generative model that disrupts adversarial patterns before reconstructing clean images. Ho et al. [46] introduced a local implicit function approach that projects adversarially perturbed inputs back onto a learned manifold using per-pixel feature encoding. Another method, DiffPure [47], employs diffusion models to remove adversarial noise through a forward perturbation process, followed by a reverse generative step to reconstruct clean images. However, these methods are designed for image classification and encounter difficulties when applied directly to object tracking, where maintaining temporal consistency and ensuring real-time performance are crucial.

To bridge the gap between images and videos in reconstruction methods, LRR [49] leverages semantic text guidance from language-image models such as CLIP [56] by taking continuous frames to create spatial-temporal implicit representations, which become the input of their proposed language-driven resample network to reconstruct incoming frames, preserving both semantic and visual consistency with the target object and its clean counterparts. Although LLR can achieve promising defense performance against decision-level attacks [38]–[40], it overlooks the feature-level attacks [41], [42] and remains vulnerable to them, which can significantly degrade tracking

accuracy by distorting feature extraction in the early stages of the tracking pipeline.

III. Methods

The overall design of our solution is illustrated in Fig. 1. Our solution is designed to enhance the robustness of UAV tracking systems in adversarial environments by integrating a pluggable dual-level reconstruction module to mitigate adversarial attacks without compromising tracking accuracy on clean inputs. The key idea is to filter adversarial perturbations from input frames before they reach the tracking pipeline to ensure accurate and reliable tracking. The reconstruction module operates as an independent input pre-processing component, making it easily pluggable into existing tracking systems without requiring modification to the underlying trackers. We focus on state-of-the-art Siamese-based trackers that are widely adopted in UAV tracking systems due to their balance of efficiency and accuracy.

The training pipeline of our dual-level reconstruction module is shown in Fig. 2. The common backbone-head architecture of Siamese-based trackers makes them suitable for our dual-level optimization strategy to address perturbations at both the feature and decision levels. During training, it takes adversarial frames $\mathbf{I}^{adv}$ as input to generate reconstructed versions $\mathbf{I}^{rec}$ using three key loss functions: decision-level loss, feature-level loss, and L_2 loss. These losses are combined to achieve comprehensive mitigation of adversarial perturbations while maintaining consistency with clean inputs. The tracker remains frozen during training and serves as a fixed reference with $\mathbf{I}^{ori}$ as inputs to optimize the reconstruction module. The backward optimization pipeline (indicated by the dashed line) iteratively updates the module to minimize the combined losses. We only use adversarial samples during training to guide the reconstruction module to learn features that effectively counter perturbations. The training objective is to reconstruct frames that are close to their clean versions. By learning from these challenging adversarial inputs, the reconstruction model can also handle clean frames effectively during testing without disrupting them. Although our approach is tailored for Siamese-based trackers, the modular nature of our design makes it extensible to other trackers with similar architectures.

A. Model Architecture

U-Net [57] architecture is adopted to construct the reconstruction module 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++ [27] and 287×287 for SiamAPN [30], 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.

B. Decision Loss

The Siamese-based trackers with backbone-head architecture produce common task-oriented response maps at the decision level, including a classification map and a regression map from the head network based on the extracted features. The classification map represents the confidence score of the proposed bounding box being a target or background and the regression map includes the information to adjust the bounding box. By minimizing the difference between these two decision-level response maps of adversarial and reconstructed frames, our reconstruction module is optimized for accurate prediction in the decision-making process. The overall decision loss is defined as $L_d = L_{score} + L_{drift}$. 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 original clean frame $\mathbf{I}^{ori}$ can be expressed as:

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

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

The score loss function is defined as follows:

$$L_{score} = \frac{\alpha}{N}(\left|\mathbf{P}_t^{rec}[\mathbf{T}] - \mathbf{P}_t^{ori}[\mathbf{T}]\right| + \left|\mathbf{P}_b^{rec}[\mathbf{B}] - \mathbf{P}_b^{ori}[\mathbf{B}]\right|) \tag{2}$$

where α is the weight of L_{score}, N is the batch size, $\mathbf{P}_t^{rec}$ and $\mathbf{P}_b^{rec}$ 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 original clean and reconstructed frames.

2) Drift Loss: The regression map $\mathbf{R} \in \mathbb{R}^{H\times W\times 4}$ 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:

$$\begin{aligned} L_{drift} = \frac{\beta}{N}\{&\sum_{\mathbf{T}} \left|\mathbf{R}^{rec}(w)\mathbf{R}^{rec}(h) - \mathbf{R}^{ori}(w)\mathbf{R}^{ori}(h)\right| \\ &+ \sum_{\mathbf{T}}((\mathbf{R}^{rec}(x) - \mathbf{R}^{ori}(x))^2 + (\mathbf{R}^{rec}(y) - \mathbf{R}^{ori}(y))^2)\} \end{aligned} \tag{3}$$

where β is the weight of L_{drift}, $\mathbf{R}^{rec}$ and $\mathbf{R}^{ori}$ are regression maps of the reconstructed frame and original frame, respectively.

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

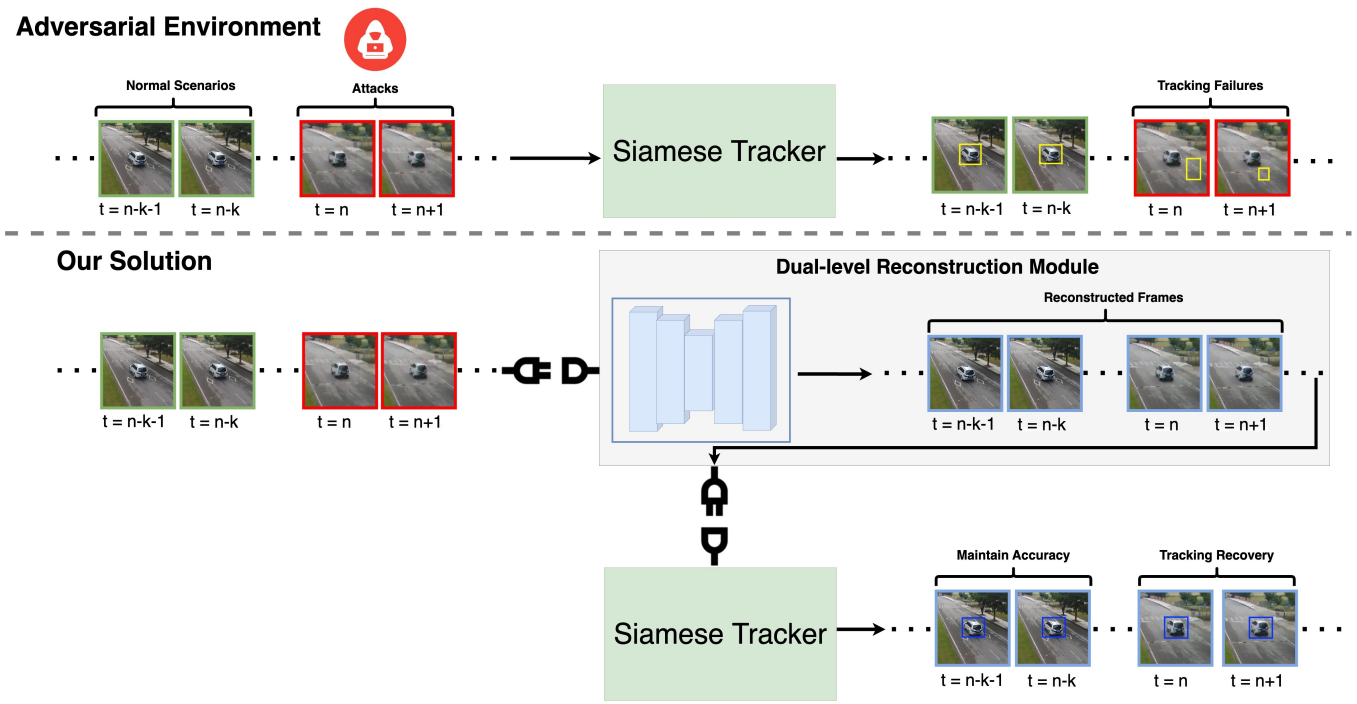


Fig. 1: Overall design of our solution - A pluggable design with dual-level reconstruction module that enables robust UAV tracking against adversarial attacks.

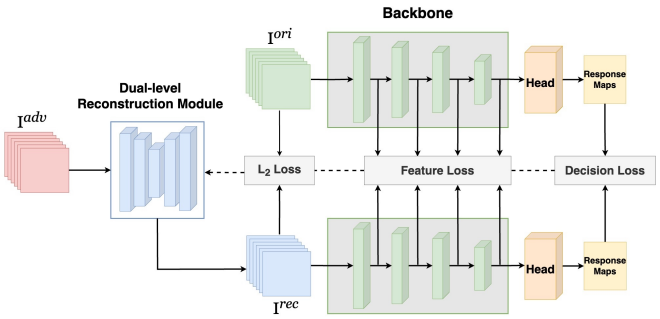


Fig. 2: Training pipeline of reconstruction module. The dual-level reconstruction module takes adversarial frames $\mathbf{I}^{adv}$ as input and outputs purified frames $\mathbf{I}^{rec}$. The model is trained using three losses: L_2 loss between $\mathbf{I}^{rec}$ and clean frames $\mathbf{I}^{ori}$, feature loss comparing intermediate features from the backbone, and decision loss based on the final response maps.

C. Feature Loss

Feature-level attacks target the intermediate feature representations extracted by the backbone network of a tracker. These attacks exploit the fact that neural networks process information through multiple layers, where low-level features are gradually transformed into high-level semantic representations used for prediction. When adversarial perturbations are introduced at the feature level, the feature maps extracted from the adversarial frames become misaligned with those of the original frames. This misalignment can propagate through the network, causing cascading disruptions that result in significant errors in decision-level tracking predictions.

To address these challenges, our reconstruction module incorporates a feature-level loss design that directly mitigates feature-level disturbances. The feature-level loss L_f is defined in Equation 4, where k is the total number of selected layers producing feature maps, γ is the weight of L_f, and i is the layer index. L_f is the sum of the l_2 norm distances between the feature maps of the original and reconstructed frames across selected layers of the backbone network. By minimizing L_f, the reconstruction module learns to generate frames whose feature maps closely match those of clean frames. This alignment effectively reduces the impact of adversarial perturbations on the extracted features, ensuring the downstream prediction process operates on high-quality semantic information.

$$L_f = \frac{\gamma}{N} \sum_{i=1}^{k} \|\mathbf{FM}^{ori}{}_i - \mathbf{FM}^{rec}{}_i\| \tag{4}$$

D. L_2 Norm Loss

The L_2 norm loss is designed to minimize the pixel-wise difference by calculating the Euclidean distance between the reconstructed frame and the clean frame. This loss function encourages the reconstruction module to generate frames that are visually similar to the original. It is defined as follows:

$$L_2 = \frac{\lambda}{N} \|\mathbf{I}^{ori} - \mathbf{I}^{rec}\| \tag{5}$$

where λ is the weight of L_2 loss, $\mathbf{I}^{ori}$ and $\mathbf{I}^{rec}$ are the clean and reconstructed frame, respectively.

IV. Evaluation

A. Experiment Setup

In our experiments, we trained the reconstruction module with different combinations of loss weights, but the performance on the validation set showed minimal variation. This indicates that the loss weights have little impact on the model's performance. To maintain consistency in loss magnitude, we set the weight parameters as follows: $\alpha = 1$ in Equation 2, $\beta = 10$ in Equation 3, $\gamma = 500$ in Equation 4, and $\lambda = 700$ in Equation 5. The batch size N is 128. We use the state-of-the-art SiamRPN++ [27] and its backbone ResNet-50 [58] to train our reconstruction module. The GOT-10K [59] dataset is downsampled by selecting one frame for 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. Two decision-level attacks including Ad^2Attack [40] and CSA [39] and a feature-level attack TAP [42], are implemented to generate adversarial samples for training. The effectiveness of our method is evaluated by implementing four attack strategies including two offline decision-level attacks Ad^2Attack [40] and CSA [39], one online decision-level attack SPARK [38], and one offline feature-level attack TAP [42] on three UAV benchmarks: UAVTrack112 [50], UAV123 [51], and UAVDT [52]. To evaluate the transferability of our reconstruction module, we also integrated our reconstruction module pre-trained on SiamRPN++ with two other representative Siamese-based trackers, including SiamAPN and SiamMask, to run the test experiments. We also compare our solution with the recently proposed solutions [49] and [1] to assess recovery effectiveness. In addition, we retrained the approach proposed in [1] that only considers decision loss with the same training set to show improvements in recovery performance by our new method against feature-level attacks.

B. Evaluation Metrics

We evaluate the tracking performance with precision and success rate as metrics [60]. Precision measures the proportion of frames where the predicted target center is within a specified location error threshold T_p from the ground truth center. The center location error (CLE) for each frame $\mathbf{I_i}$ is computed as

$$\mathrm{CLE}_i = \sqrt{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2}, \tag{6}$$

where (x_i, y_i) and $(\hat{x}_i, \hat{y}_i)$ represent the ground truth and predicted center locations, respectively. A prediction is considered precise if its CLE is less than T_p, which is defined as

$$P_i = \begin{cases} 1, & \text{if } \mathrm{CLE}_i \leq T_p, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

The overall precision is calculated as the proportion of predictions that satisfy the threshold T_p constraint:

$$\text{Precision} = \frac{1}{N} \sum_{i=1}^{N} P_i, \tag{8}$$

where N is the total number of frames.

Success rate evaluates tracking performance based on the intersection over union (IoU) between the predicted and ground truth bounding boxes. A prediction is considered successful if the IoU is equal to or greater than the predefined threshold T_{IoU}, expressed as

$$\mathrm{IoU}_i \geq T_{IoU}, \tag{9}$$

Based on this criterion, the success indicator for each frame $\mathbf{I_i}$ is defined as

$$S_i = \begin{cases} 1, & \text{if } \mathrm{IoU}_i > T_{IoU}, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

The overall success rate is then computed as the proportion of successful predictions:

$$\text{Success Rate} = \frac{1}{N} \sum_{i=1}^{N} S_i, \tag{11}$$

where N is the total number of frames.

These metrics provide a comprehensive evaluation of tracking performance by assessing precision for target localization accuracy and success rate for bounding box overlap quality.

C. Loss Weights Selection

Due to the varying magnitudes of the raw loss values, we rescale them to the range [1, 10] to improve training stability and ensure balanced influence. Fig. 3 illustrates the convergence of training and validation loss over epochs. Early stopping is employed to prevent overfitting, with the best model selected at epoch 14.

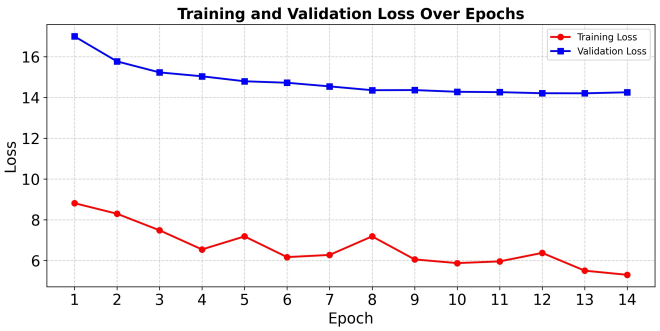


Fig. 3: Training and validation loss convergence.

TABLE I: Different loss weight configurations

	Loss Weights				Validation Performance	
	α	β	γ	λ	Precision	Success Rate
Raw	1	1	1	1	0.518	0.324
Selected	1	10	500	700	0.617	0.497
Comb.1	5	15	550	750	0.609	0.472
Comb.2	3	12	450	650	0.598	0.470
Decision-only	1	10	0	0	0.532	0.358
Feature-only	0	0	500	0	0.552	0.410
L_2-only	0	0	0	700	0.520	0.344

Table I compares different loss weight configurations on validation performance. The results show that applying scaled

loss values via appropriate weights significantly improves model performance over the unweighted (Raw) setting. The selected configuration achieves the best validation performance, showing the importance of balancing different losses. Comb.1 and Comb.2 achieve comparable performance, indicating that moderate variations in weights have limited impact as long as the losses remain in the same magnitude. In contrast, using only decision-level, feature-level, or L_2 loss results in degraded performance, demonstrating that no single loss alone is sufficient. These results highlight the importance of jointly optimizing multiple losses to guide training effectively.

D. Loss Interaction

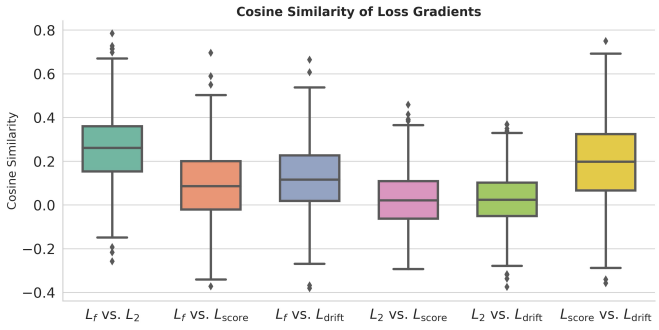


Fig. 4: Cosine similarity of loss gradients.

Fig. 4 illustrates the cosine similarity of gradients between different loss pairs during training. A cosine similarity of 1 indicates aligned gradients (i.e., losses optimize in the same direction), 0 indicates orthogonal gradients (independent influence), and –1 indicates opposing directions. Each box represents the distribution of loss values across training iterations, with the median (central line), interquartile range (box), and outliers. The feature-level loss (L_f) shows moderate similarity with L_2 and each decision-level loss (L_{score} and L_{drift}), suggesting complementary guidance during optimization. In contrast, L_2 has low similarity with the decision losses, reflecting separate optimization directions. The decision losses L_{score} and L_{drift} align in gradient direction, which indicates they collaborate with each other during model training. These observations indicate that no single loss dominates training, and proper weight balancing is essential for effective joint optimization.

E. Results

The evaluation results are summarized in Table II to Table V. The performance of our solution with different thresholds (T_p and T_{IoU}) is also detailed in the Appendix. Table II, Table III, and Table IV present the recovery tracking performance achieved by our solution under various adversarial attacks with comparison to the defense methods in [1] and [49]. Precision values are taken at $T_p = 20$, and success rates are represented by the AUC across T_{IoU} from 0 to 1 in success plots. For decision-level attacks including Ad^2Attack, CSA, and SPARK, our method restores tracking performance with an average precision of 96.8% and an average success rate of 95.2% relative to the original scenarios across three benchmarks, which is comparable to the recovery performance of [1] and [49]. This indicates our approach can mitigate disruptions at the decision level and ensure stable tracking performance.

For feature-level attacks like TAP, our solution achieves an average precision of 92.7% and an average success rate of 87.4% relative to the original scenarios for SiamRPN++ across three benchmarks, 88.9% and 76.5% for SiamAPN, and 88.9% and 83.4% for SiamMask. Our approach outperforms the recovery method in [1] and improves precision and success rate by an average of 8.9% and 6.7% for SiamRPN++, 8.8% and 11.7% for SiamAPN, and 4.8% and 3.3% for SiamMask. Compared to [49], our approach improves precision and success rate by 25.3% and 27.8% for SiamRPN++, 45.4% and 39.9% for SiamAPN, and 19.6% and 20.8% for SiamMask. These improvements result from our dual-level reconstruction process that addresses perturbations at both the feature extraction and decision-making stages. This capability makes our approach more resilient to attacks targeting feature representations.

Table V shows the performance of the solution under normal conditions, i.e., without attacks. For all three evaluated trackers, plugging in our solution maintains their tracking performance. Therefore, our solution can enhance the robustness of these trackers against adversarial attacks without affecting the operation under normal conditions.

Fig. 5: Tracking examples.

F. Tracking Examples

We take frames from two videos in the UAV123 dataset to show the effectiveness of our solution in restoring tracking performance under the Ad^2Attack. The first row in Fig. 5 displays the original tracking results for people 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

TABLE II: Recovery performance on SiamRPN++ under four attacks

SiamRPN++						
	Precision			Success Rate		
Dataset	**UAVTrack112**	**UAV123**	**UAVDT**	**UAVTrack112**	**UAV123**	**UAVDT**
Ori.	0.815	0.790	0.821	0.630	0.597	0.609
Ad^2**Attack [40] (% of Ori.)**	0.515 (63.2%)	0.519 (65.7%)	0.579 (70.5%)	0.361 (57.3%)	0.373 (62.5%)	0.338 (55.5%)
Recovery by [1] (% of Ori.)	0.776 (95.2%)	0.748 (94.7%)	0.818 (99.6%)	0.596 (94.6%)	0.562 (94.1%)	0.588 (96.6%)
Recovery by [49] (% of Ori.)	0.691 (84.8%)	0.654 (82.8%)	0.677 (82.5%)	0.485 (77.0%)	0.476 (79.7%)	0.454 (74.5%)
Our Solution (% of Ori.)	0.770 (94.5%)	0.741 (93.8%)	0.800 (97.4%)	0.591 (93.8%)	0.555 (93.0%)	0.568 (93.3%)
CSA [39] (% of Ori.)	0.403 (49.4%)	0.440 (55.7%)	0.452 (55.1%)	0.271 (43.0%)	0.293 (49.1%)	0.281 (46.1%)
Recovery by [1] (% of Ori.)	0.792 (97.2%)	0.779 (98.6%)	0.815 (99.3%)	0.607 (96.3%)	0.586 (98.2%)	0.592 (97.2%)
Recovery by [49] (% of Ori.)	0.755 (92.6%)	0.756 (95.7%)	0.783 (95.4%)	0.540 (85.7%)	0.553 (92.6%)	0.540 (88.7%)
Our Solution (% of Ori.)	0.787 (96.6%)	0.795 (100%)	0.822 (100%)	0.597 (94.8%)	0.593 (99.3%)	0.589 (96.7%)
SPARK [38] (% of Ori.)	0.405 (49.7%)	0.578 (73.2%)	0.307 (37.4%)	0.303 (48.1%)	0.424 (71.0%)	0.221 (36.3%)
Recovery by [1] (% of Ori.)	0.795 (97.5%)	0.750 (94.9%)	0.782 (95.2%)	0.603 (95.7%)	0.563 (94.3%)	0.571 (93.8%)
Recovery by [49] (% of Ori.)	0.809 (99.3%)	0.801 (101%)	0.830 (101%)	0.616 (97.8%)	0.611 (102%)	0.620 (102%)
Our Solution (% of Ori.)	0.781 (95.8%)	0.747 (94.6%)	0.805 (98.1%)	0.599 (95.1%)	0.559 (93.6%)	0.591 (97.0%)
TAP [42] (% of Ori.)	0.543 (66.6%)	0.479 (60.6%)	0.503 (61.3%)	0.395 (62.7%)	0.357 (59.8%)	0.319 (52.4%)
Recovery by [1] (% of Ori.)	0.650 (79.8%)	0.676 (85.6%)	0.706 (86.0%)	0.513 (81.4%)	0.484 (81.1%)	0.484 (79.5%)
Recovery by [49] (% of Ori.)	0.568 (69.7%)	0.554 (70.1%)	0.512 (62.4%)	0.398 (63.2%)	0.404 (67.7%)	0.303 (47.8%)
Our Solution (% of Ori.)	0.783 (96.1%)	0.718 (90.9%)	0.749 (91.2%)	0.577 (91.6%)	0.530 (88.8%)	0.498 (81.8%)
Δ(Ours−[1]) (% of Ori.)	0.133 (16.3%)	0.042 (5.3%)	0.043 (5.2%)	0.064 (10.2%)	0.046 (7.7%)	0.014 (2.3%)
Δ(Ours−[49]) (% of Ori.)	0.215 (26.4%)	0.164 (20.8%)	0.237 (28.8%)	0.179 (28.4%)	0.126 (21.1%)	0.195 (34%)

TABLE III: Recovery performance on SiamAPN under four attacks

SiamAPN						
	Precision			Success Rate		
Dataset	**UAVTrack112**	**UAV123**	**UAVDT**	**UAVTrack112**	**UAV123**	**UAVDT**
Ori.	0.812	0.765	0.708	0.617	0.574	0.517
Ad^2**Attack [40] (% of Ori.)**	0.313 (38.5%)	0.248 (32.4%)	0.291 (41.1%)	0.133 (21.6%)	0.114 (19.9%)	0.121 (23.4%)
Recovery by [1] (% of Ori.)	0.775 (95.4%)	0.693 (90.1%)	0.691 (97.6%)	0.566 (91.7%)	0.509 (88.7%)	0.468 (90.5%)
Recovery by [49] (% of Ori.)	0.535 (65.9%)	0.549 (71.8%)	0.569 (80.4%)	0.286 (46.4%)	0.330 (57.5%)	0.284 (54.9%)
Our Solution (% of Ori.)	0.765 (94.2%)	0.716 (93.6%)	0.708 (100%)	0.561 (90.9%)	0.527 (91.8%)	0.493 (95.4%)
CSA [39] (% of Ori.)	0.762 (93.8%)	0.717 (93.7%)	0.680 (96.0%)	0.525 (85.1%)	0.517 (90.1%)	0.397 (76.8%)
Recovery by [1] (% of Ori.)	0.799 (98.4%)	0.750 (98.0%)	0.742 (105%)	0.601 (97.4%)	0.562 (97.9%)	0.533 (103%)
Recovery by [49] (% of Ori.)	0.779 (95.9%)	0.732 (95.7%)	0.629 (88.8%)	0.575 (93.2%)	0.542 (94.4%)	0.442 (85.5%)
Our Solution (% of Ori.)	0.790 (97.3%)	0.741 (96.9%)	0.735 (104%)	0.593 (96.1%)	0.557 (97.0%)	0.513 (99.2%)
SPARK [38] (% of Ori.)	0.240 (29.6%)	0.337 (44.1%)	0.161 (22.7%)	0.154 (25.0%)	0.209 (35.9%)	0.096 (18.6%)
Recovery by [1] (% of Ori.)	0.776 (95.6%)	0.721 (94.2%)	0.726 (103%)	0.580 (94.0%)	0.534 (93.0%)	0.523 (101%)
Recovery by [49] (% of Ori.)	0.813 (100%)	0.771 (101%)	0.729 (103%)	0.620 (100%)	0.571 (99.5%)	0.523 (101%)
Our Solution (% of Ori.)	0.782 (96.3%)	0.718 (93.9%)	0.722 (102%)	0.583 (94.5%)	0.533 (92.9%)	0.519 (100%)
TAP [42] (% of Ori.)	0.307 (37.8%)	0.337 (44.1%)	0.231 (32.6%)	0.188 (30.5%)	0.218 (38.0%)	0.133 (25.7%)
Recovery by [1] (% of Ori.)	0.648 (79.8%)	0.591 (77.3%)	0.590 (83.3%)	0.397 (64.3%)	0.400 (69.7%)	0.313 (60.5%)
Recovery by [49] (% of Ori.)	0.345 (42.5%)	0.388 (50.7%)	0.264 (37.3%)	0.209 (33.9%)	0.263 (45.8%)	0.153 (30.0%)
Our Solution (% of Ori.)	0.728 (89.7%)	0.655 (85.6%)	0.648 (91.5%)	0.486 (78.8%)	0.460 (80.1%)	0.365 (70.6%)
Δ(Ours−[1]) (% of Ori.)	0.080 (9.9%)	0.064 (8.3%)	0.058 (8.2%)	0.089 (14.5%)	0.060 (10.4%)	0.052 (10.1%)
Δ(Ours−[49]) (% of Ori.)	0.383 (47.2%)	0.267 (34.9%)	0.384 (54.2%)	0.277 (44.9%)	0.197 (34.3%)	0.212 (40.6%)

shows recovery tracking predictions by our solution using reconstructed frames, depicted by blue boxes overlapping with the ground truths.

G. Real-World Tests

Real-world tests were conducted on a UAV platform powered by NVIDIA Jetson AGX Xavier (32GB) to evaluate the efficiency and practicality of our solution. SiamRPN++ was deployed as the tracking model, and Ad^2Attack was implemented as the adversarial attack method. The integration of our solution introduces additional computation cost for better robustness, and hence reduces the tracking speed on our UAV platform from 40 frames per second (fps) to 12 fps, i.e., 233% computation overhead. As a comparison, the recent research [49] that is compared in our evaluation for tracking performance requires a similar computation overhead, i.e., 243%. Note that, although our prototype implementation successfully supported our real-world tests, its efficiency and computational overhead could be further optimized with optimization strategies such as those proposed in recent research [61].

Tracking results from two real-world tests are shown in Fig. 6 and Fig. 7, where the targets are a runner and a scooter rider. These tests evaluate our method in real-world conditions by comparing original tracking, adversarial attacks, and recovery with our solution. Ad^2Attack degrades tracking accuracy with significant prediction deviations. Our solution restores performance effectively by producing bounding box predictions that closely align with the ground truth. The IoU curves in Fig. 6 and Fig. 7 show a drop to zero under adversarial attack. After applying our method, IoU values return to levels comparable to the original scenario.

Fig. 8 presents a tracking sequence from real-world UAV scenarios. Under adversarial attacks, the tracker produces unstable predictions (red boxes) that drift away from the target. In contrast, our solution effectively restores accurate and stable tracking outputs (blue boxes) that remain consistent with the

TABLE IV: Recovery performance on SiamMask under four attacks

SiamMask						
	Precision			Success Rate		
Dataset	**UAVTrack112**	**UAV123**	**UAVDT**	**UAVTrack112**	**UAV123**	**UAVDT**
Ori.	0.794	0.790	0.803	0.599	0.589	0.598
Ad^2**Attack [40] (% of Ori.)**	0.504 (63.5%)	0.577 (73.0%)	0.521 (64.9%)	0.298 (49.7%)	0.375 (63.7%)	0.256 (42.8%)
Recovery by [1] (% of Ori.)	0.756 (95.2%)	0.744 (94.2%)	0.797 (99.3%)	0.560 (93.5%)	0.555 (94.2%)	0.564 (94.3%)
Recovery by [49] (% of Ori.)	0.678 (85.4%)	0.640 (81.0%)	0.684 (85.2%)	0.476 (79.5%)	0.465 (78.9%)	0.459 (76.8%)
Our Solution (% of Ori.)	0.749 (94.3%)	0.763 (96.6%)	0.796 (99.1%)	0.554 (92.5%)	0.564 (95.8%)	0.567 (94.8%)
CSA [39] (% of Ori.)	0.327 (41.2%)	0.445 (56.3%)	0.332 (41.3%)	0.190 (31.7%)	0.263 (44.7%)	0.170 (28.4%)
Recovery by [1] (% of Ori.)	0.779 (98.1%)	0.772 (97.7%)	0.772 (96.1%)	0.577 (96.3%)	0.572 (97.1%)	0.553 (92.5%)
Recovery by [49] (% of Ori.)	0.764 (96.2%)	0.756 (95.7%)	0.801 (99.8%)	0.546 (91.2%)	0.552 (93.7%)	0.556 (93.0%)
Our Solution (% of Ori.)	0.773 (97.4%)	0.788 (99.7%)	0.765 (95.3%)	0.569 (95.0%)	0.582 (98.8%)	0.542 (90.6%)
SPARK [38] (% of Ori.)	0.304 (38.3%)	0.484 (61.3%)	0.182 (22.7%)	0.220 (36.7%)	0.337 (57.2%)	0.125 (20.9%)
Recovery by [1] (% of Ori.)	0.761 (95.8%)	0.738 (93.4%)	0.743 (92.5%)	0.568 (94.8%)	0.551 (93.5%)	0.531 (88.8%)
Recovery by [49] (% of Ori.)	0.810 (102%)	0.807 (102%)	0.815 (101%)	0.608 (101%)	0.601 (102%)	0.618 (103%)
Our Solution (% of Ori.)	0.767 (96.6%)	0.731 (92.5%)	0.738 (91.9%)	0.570 (95.2%)	0.548 (93.0%)	0.533 (89.1%)
TAP [42] (% of Ori.)	0.541 (68.1%)	0.538 (68.1%)	0.479 (59.7%)	0.372 (62.1%)	0.386 (65.5%)	0.281 (47.0%)
Recovery by [1] (% of Ori.)	0.689 (86.8%)	0.661 (83.7%)	0.656 (81.7%)	0.499 (83.3%)	0.491 (83.4%)	0.439 (73.4%)
Recovery by [49] (% of Ori.)	0.577 (72.7%)	0.559 (70.8%)	0.516 (64.3%)	0.403 (67.3%)	0.407 (69.1%)	0.306 (51.2%)
Our Solution (% of Ori.)	0.757 (95.3%)	0.689 (87.2%)	0.675 (84.1%)	0.538 (89.8%)	0.504 (85.6%)	0.447 (74.7%)
Δ(Ours−[1]) (% of Ori.)	0.068 (8.5%)	0.028 (3.5%)	0.019 (2.4%)	0.039 (6.5%)	0.013 (2.2%)	0.008 (1.3%)
Δ(Ours−[49]) (% of Ori.)	0.180 (22.6%)	0.130 (16.4%)	0.159 (19.8%)	0.135 (22.5%)	0.097 (16.5%)	0.141 (23.5%)

TABLE V: Tracking performance without attack on three trackers

		Precision			Success Rate		
	Dataset	**UAVTrack112**	**UAV123**	**UAVDT**	**UAVTrack112**	**UAV123**	**UAVDT**
SiamRPN++	**Ori.**	0.815	0.790	0.821	0.630	0.597	0.609
	Our Solution (% of Ori.)	0.788 (96.7%)	0.773 (97.8%)	0.805 (98.1%)	0.606 (96.2%)	0.577 (96.6%)	0.594 (97.5%)
SiamAPN	**Ori.**	0.812	0.765	0.708	0.617	0.574	0.517
	Our Solution (% of Ori.)	0.800 (98.5%)	0.728 (95.2%)	0.715 (100%)	0.603 (97.7%)	0.547 (95.3%)	0.518 (100%)
SiamMask	**Ori.**	0.794	0.790	0.803	0.599	0.589	0.598
	Our Solution (% of Ori.)	0.790 (99.5%)	0.782 (99.0%)	0.831 (103%)	0.588 (98.2%)	0.574 (97.5%)	0.610 (102%)

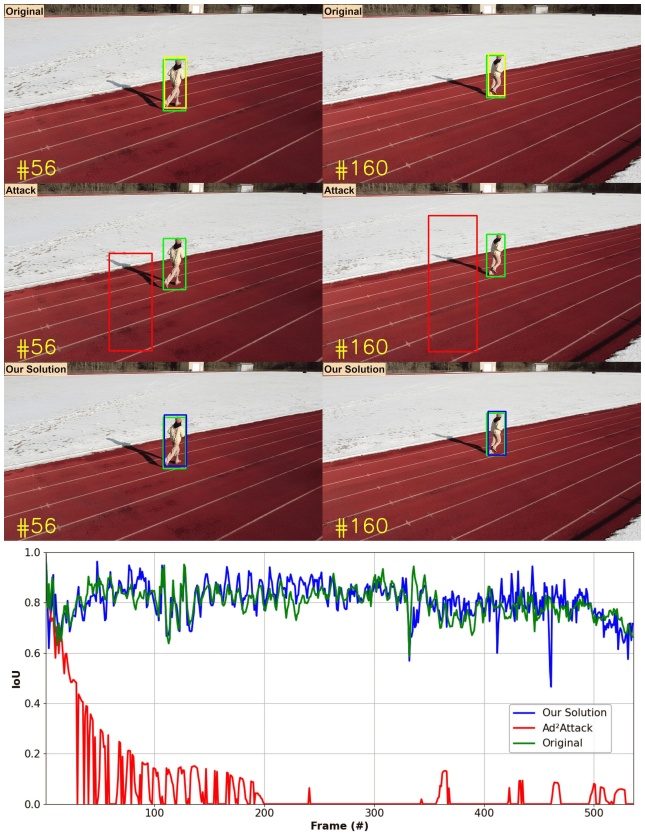


Fig. 6: Sample tracking frames and IoU curve - Runner.

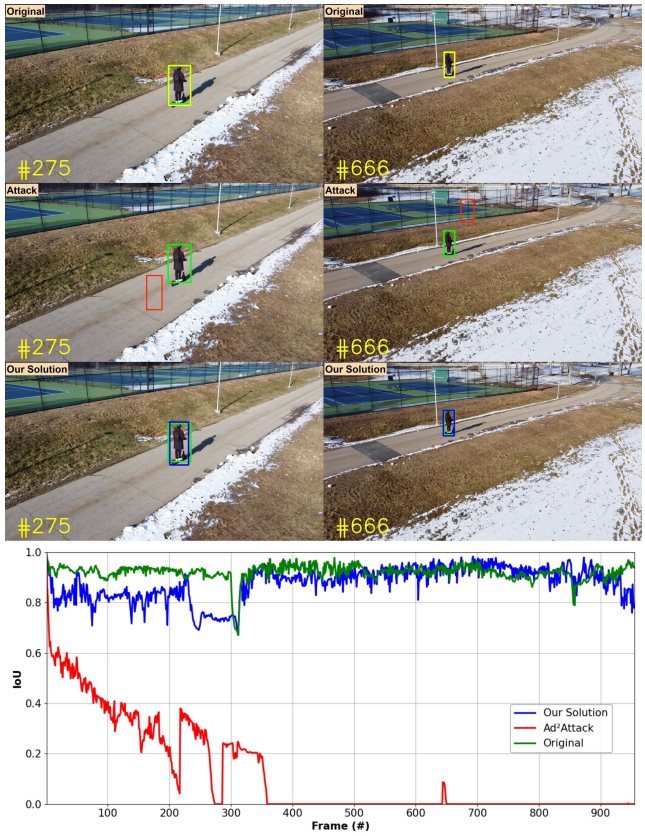


Fig. 7: Sample tracking frames and IoU curve - Scooter rider.

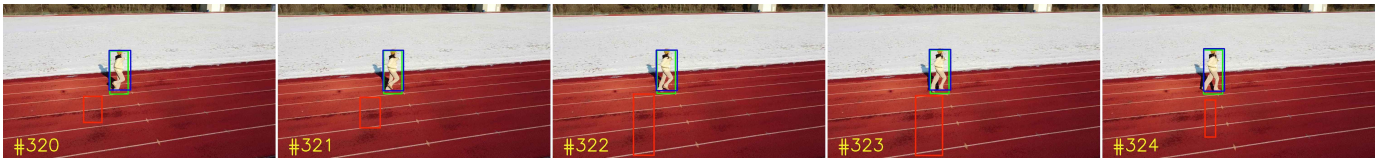


Fig. 8: Sample tracking sequence in real-world test.

object's trajectory.

V. Discussion and Future Work

Our solution currently focuses on the Siamese tracker family, which has been widely adopted for UAV tracking solutions. In our solution, loss functions are tailored to their typical backbone-head architectures. In these trackers, the backbone extracts feature representations, and the head networks, including classification and regression branches, produce response maps for tracking predictions. Our decision-level losses are task-specific and defined based on the output response maps from these head networks. Recently proposed transformer-based trackers, such as STARK [62], employ corner heads that directly regress the coordinates of the top-left and bottom-right corners of the target bounding box. Therefore, when adapting our approach to transformer-based trackers, the decision-level loss needs to be redesigned to align with the outputs of corner heads. In our future research, we will aim to address such challenges and further improve the generality of our solution for more tracking systems.

VI. Conclusion

This paper presents a pluggable tracking solution designed to enhance the resilience of UAV tracking systems against adversarial attacks. Our approach functions as an input reconstruction module and employs a dual-level optimization strategy to restore video frames at both the feature and decision levels, ensuring accurate and reliable tracking. As an input pre-processing component, our solution can be seamlessly integrated with various Siamese-based trackers without requiring modifications to their architectures and can easily plug into existing UAV tracking systems. Extensive evaluations on multiple UAV tracking benchmarks show that our solution effectively eliminates adversarial impacts and restores tracking accuracy across diverse attack scenarios. Real-world tests were conducted on a UAV platform to assess the deployment feasibility of our solution. The results indicate the practicality and efficiency of our approach, demonstrating its applicability for robust real-time UAV tracking in adversarial environments.

VII. Appendix

Fig. 9, Fig. 10, and Fig. 11 show the precision and success rates of our solution under different thresholds. The first row of each Figure displays the precision plots, where location error thresholds (T_p) range from 0 to 50. For each tracking condition, the representative precision score shown next to the plot is taken at $T_p = 20$ pixels. The second row presents the success plots, which show the proportion of successful predictions across IoU thresholds (T_{IoU}) from 0 to 1, with values next to the plot indicating the area under the curve (AUC). The AUC of each success plot is used as the representative score for a more comprehensive assessment across different T_{IoU} values.

References

[1] M. Jia, Y. Li, and J. Yuan, "A robust uav tracking solution in the adversarial environment," in *2024 IEEE 36th International Conference on Tools with Artificial Intelligence (ICTAI)*, 2024, pp. 945–952.

[2] N. S. Labib, M. R. Brust, G. Danoy, and P. Bouvry, "The rise of drones in internet of things: A survey on the evolution, prospects and challenges of unmanned aerial vehicles," *IEEE Access*, vol. 9, pp. 115 466–115 487, 2021.

[3] Y. Wang, Z. Tang, A. Huang, H. Zhang, L. Chang, and J. Pan, "Placement of uav-mounted edge servers for internet of vehicles," *IEEE Transactions on Vehicular Technology*, 2024.

[4] Z. Chen, Z. Qu, N. Xiong, A. Liu, M. Dong, T. Wang, and S. Zhang, "Uitde: A uav-assisted intelligent true data evaluation method for ubiquitous iot systems in intelligent transportation of smart city," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[5] A. Telikani, A. Sarkar, B. Du, and J. Shen, "Machine learning for uav-aided its: A review with comparative study," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[6] X. Yan, T. Fu, H. Lin, F. Xuan, Y. Huang, Y. Cao, H. Hu, and P. Liu, "Uav detection and tracking in urban environments using passive sensors: A survey," *Applied Sciences*, vol. 13, no. 20, p. 11320, 2023.

[7] A. V. Savkin and H. Huang, "Multi-uav navigation for optimized video surveillance of ground vehicles on uneven terrains," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 9, pp. 10 238–10 242, 2023.

[8] W. Zheng, H. Xu, P. Li, R. Wang, and X. Shao, "Sac-rsm: A high-performance uav-side road surveillance model based on super-resolution assisted learning," *IEEE Internet of Things Journal*, 2024.

[9] S. Park, S. B. Son, S. Jung, and J. Kim, "Dynamic quantum federated learning for uav-based autonomous surveillance," *IEEE Transactions on Vehicular Technology*, 2025.

[10] J. Alikhanov and H. Kim, "Online action detection in surveillance scenarios: A comprehensive review and comparative study of state-of-the-art multi-object tracking methods," *IEEE Access*, 2023.

[11] M. Choubisa, V. Kumar, M. Kumar, and S. Khanna, "Object tracking in intelligent video surveillance system based on artificial system," in *2023 International Conference on Computational Intelligence, Communication Technology and Networking (CICTN)*. IEEE, 2023, pp. 160–166.

[12] X. Kong, C. Ni, G. Duan, G. Shen, Y. Yang, and S. K. Das, "Energy consumption optimization of uav-assisted traffic monitoring scheme with tiny reinforcement learning," *IEEE Internet of Things Journal*, 2024.

[13] B. Cherif, H. Ghazzai, and A. Alsharoa, "Lidar from the sky: Uav integration and fusion techniques for advanced traffic monitoring," *IEEE Systems Journal*, 2024.

[14] M. Bakirci and I. Bayraktar, "Integrating uav-based aerial monitoring and ssd for enhanced traffic management in smart cities," in *2024 Mediterranean Smart Cities Conference (MSCC)*. IEEE, 2024, pp. 1–6.

[15] V. Desai, S. Degadwala, and D. Vyas, "Multi-categories vehicle detection for urban traffic management," in *2023 Second International Conference on Electronics and Renewable Systems (ICEARS)*. IEEE, 2023, pp. 1486–1490.

[16] A. A. Rafique, A. Al-Rasheed, A. Ksibi, M. Ayadi, A. Jalal, K. Alnowaiser, H. Meshref, M. Shorfuzzaman, M. Gochoo, and J. Park, "Smart traffic monitoring through pyramid pooling vehicle detection and filter-based tracking on aerial images," *IEEE Access*, vol. 11, pp. 2993–3007, 2023.

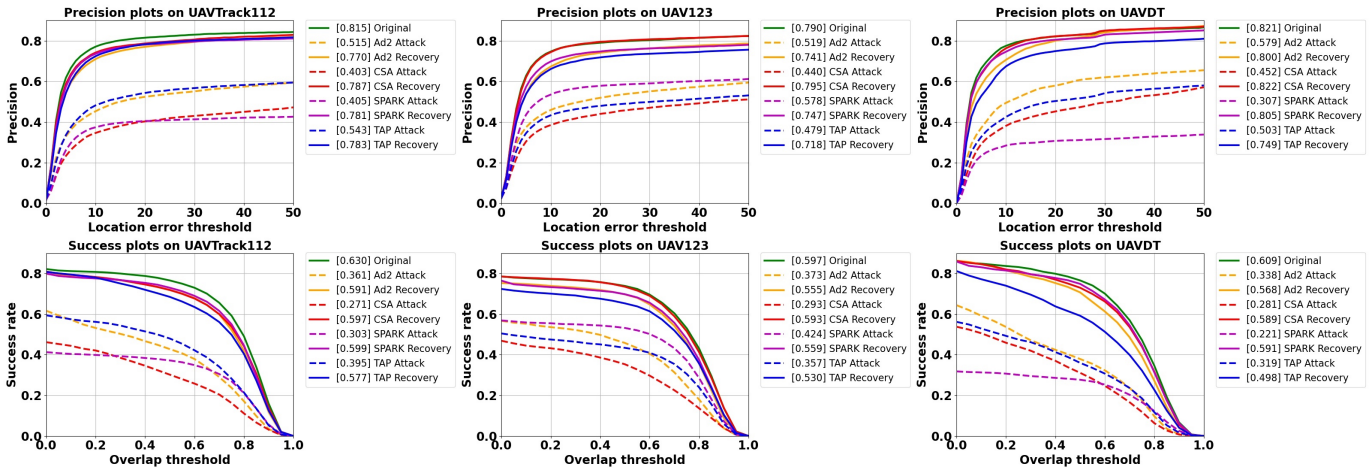


Fig. 9: Overall tracking performance of SiamRPN++ tracker under attacks (dashed lines) and recovery by our solution (solid lines) on UAVTrack112, UAV123, and UAVDT.

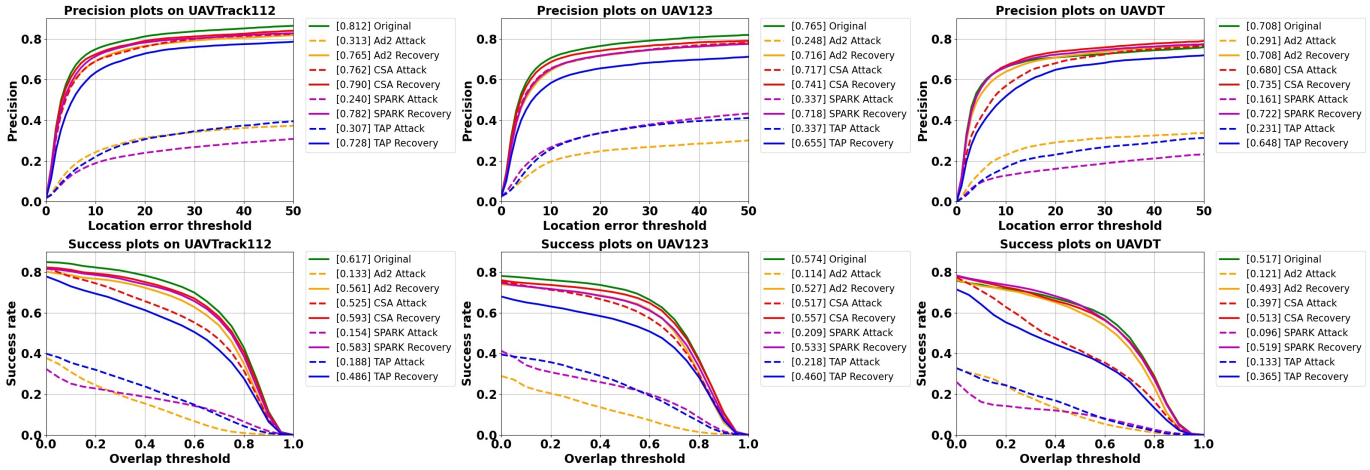


Fig. 10: Overall tracking performance of SiamAPN tracker under attacks (dashed lines) and recovery by our solution (solid lines) on UAVTrack112, UAV123, and UAVDT.

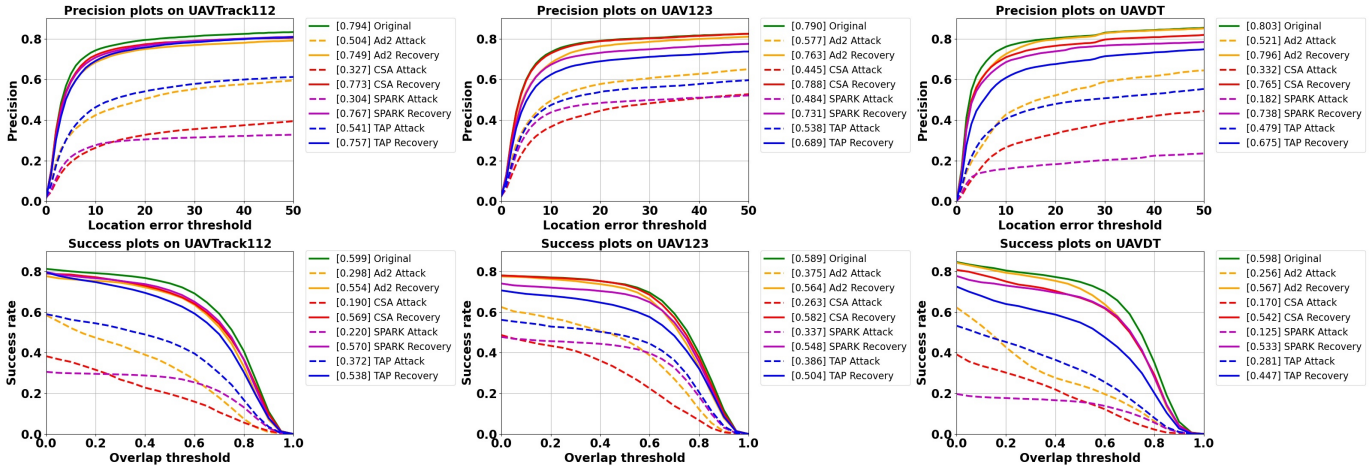


Fig. 11: Overall tracking performance of SiamMask tracker under attacks (dashed lines) and recovery by our solution (solid lines) on UAVTrack112, UAV123, and UAVDT.

[17] H. Huang and A. V. Savkin, "Navigating uavs for optimal monitoring of groups of moving pedestrians or vehicles," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 4, pp. 3891–3896, 2021.

[18] S. Bouassida, N. Neji, L. Nouveliere, J. Mohamed, and J. Neji, "System of unmanned aerial vehicles for road safety improvement," in *2023 Integrated Communication, Navigation and Surveillance Conference (ICNS)*. IEEE, 2023, pp. 1–8.

[19] H. Tang and Z. Li, "Pedestrian detection algorithm based on improved yolo v5," in *2024 9th International Conference on Intelligent Computing and Signal Processing (ICSP)*. IEEE, 2024, pp. 83–86.

[20] R. Ravindran, M. J. Santora, and M. M. Jamali, "Multi-object detection and tracking, based on dnn, for autonomous vehicles: A review," *IEEE Sensors Journal*, vol. 21, no. 5, pp. 5668–5677, 2020.

[21] H.-k. Chiu, J. Li, R. Ambruş, and J. Bohg, "Probabilistic 3d multi-modal, multi-object tracking for autonomous driving," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 14 227–14 233.

[22] Z. Xu, X. Zhan, Y. Xiu, C. Suzuki, and K. Shimada, "Onboard dynamic-object detection and tracking for autonomous robot navigation with rgb-d camera," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 651–658, 2023.

[23] M. Ahmed, A. B. Bakht, T. Hassan, W. Akram, A. Humais, L. Seneviratne, S. He, D. Lin, and I. Hussain, "Vision-based autonomous navigation for unmanned surface vessel in extreme marine conditions," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 7097–7103.

[24] 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.

[25] 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.

[26] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part II 14*. Springer, 2016, pp. 850–865.

[27] 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

[28] 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.

[29] 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.

[30] 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.

[31] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr, "Fast online object tracking and segmentation: A unifying approach," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, 2019, pp. 1328–1338.

[32] C. Szegedy, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[33] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*. Springer, 2013, pp. 387–402.

[34] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 427–436.

[35] A. Madry, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[36] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," *arXiv preprint arXiv:1712.04248*, 2017.

[37] 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.

[38] 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.

[39] 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.

[40] 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.

[41] S. Zhao, T. Xu, X.-J. Wu, and J. Kittler, "Pluggable attack for visual object tracking," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 1227–1240, 2023.

[42] M. Salzmann *et al.*, "Learning transferable adversarial perturbations," *Advances in Neural Information Processing Systems*, vol. 34, pp. 13 950–13 962, 2021.

[43] 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.

[44] 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.

[45] 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.

[46] 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.

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

[48] R. Yu, Z. Wu, Q. Liu, S. Zhou, M. Gou, and B. Xiang, "Cmdn: Pre-trained visual representations boost adversarial robustness for uav tracking," *Drones*, vol. 8, no. 11, p. 607, 2024.

[49] J. Chen, X. Ren, Q. Guo, F. Juefei-Xu, D. Lin, W. Feng, L. Ma, and J. Zhao, "LRR: Language-driven resamplable continuous representation against adversarial tracking attacks," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=3qo1pJHabg

[50] 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.

[51] 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.

[52] 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.

[53] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[55] 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.

[56] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

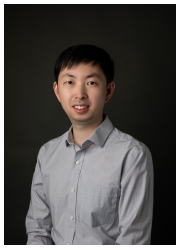
[57] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[58] 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.
[59] 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.
[60] 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.
[61] T. P. Swaminathan, C. Silver, and T. Akilan, "Benchmarking deep learning models on nvidia jetson nano for real-time systems: An empirical investigation," *arXiv preprint arXiv:2406.17749*, 2024.
[62] B. Yan, H. Peng, J. Fu, D. Wang, and H. Lu, "Learning spatio-temporal transformer for visual tracking," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 448–10 457.

Mengjie Jia received the B.Eng. degree in Computer Science from Shanghai University, China, in 2021, and the M.S. degree in Computer Science from the University of Massachusetts Dartmouth in 2023. She is currently pursuing her Ph.D. degree in Computer Science at the University of Massachusetts Dartmouth. Her research interests include AI-driven UAV security, deep learning, and robust object tracking.

Yanyan Li is an Associate Professor in Computer Science and Information Systems at California State University San Marcos. He received his Ph.D. degree from the University of Arkansas at Little Rock in 2018. His research focuses on mobile and IoT security, UAV security, AI and security, and cybersecurity education. He has published over 30 peer-reviewed papers in international journals and conferences, and received four best paper awards and one best poster award in IEEE CNS 2016.

Houbing Herbert Song received the Ph.D. degree in electrical engineering from the University of Virginia, Charlottesville, VA, in August 2012. He is currently a Professor, the Founding Director of the NSF Center for Aviation Big Data Analytics (Planning), the Associate Director for Leadership of the DOT Transportation Cybersecurity Center for Advanced Research and Education (Tier 1 Center), and the Director of the Security and Optimization for Networked Globe Laboratory (SONG Lab, www.SONGLab.us), University of Maryland, Baltimore County (UMBC), Baltimore, MD. He is a Distinguished Visiting Fellow of the Scottish Informatics and Computer Science Alliance (SICSA). Prior to joining UMBC, he was a Tenured Associate Professor of Electrical Engineering and Computer Science at Embry-Riddle Aeronautical University, Daytona Beach, FL. He serves as an Associate Editor for IEEE Transactions on Artificial Intelligence (TAI) (2023-present), IEEE Internet of Things Journal (2020-present), IEEE Transactions on Intelligent Transportation Systems (2021-present), and IEEE Journal on Miniaturization for Air and Space Systems (J-MASS) (2020-present). He was an Associate Technical Editor for IEEE Communications Magazine (2017-2020). Dr. Song is an IEEE Fellow, an Asia-Pacific Artificial Intelligence Association (AAIA) Fellow, an ACM Distinguished Member, and a Full Member of Sigma Xi. Dr. Song has been a Highly Cited Researcher identified by Web of Science since 2021. He is an ACM Distinguished Speaker (2020-present), an IEEE Computer Society Distinguished Visitor (2024-present), an IEEE ComSoc Distinguished Lecturer (2024-present), an IEEE Intelligent Transportation Systems Society (ITSS) Distinguished Lecturer (2024-present), an IEEE Vehicular Technology Society (VTS) Distinguished Lecturer (2023-present) and an IEEE Systems Council Distinguished Lecturer (2023-present).



Jiawei Yuan is an Associate Professor in Computer and Information Science at the University of Massachusetts Dartmouth. Previously, he served as an Assistant Professor at Embry-Riddle Aeronautical University from 2015 to 2019. Dr. Yuan earned his Ph.D. from the University of Arkansas at Little Rock in 2015 and his BEng in 2011 from the University of Electronic Science and Technology of China. His research primarily focuses on cybersecurity, with a recent emphasis on UAV security, the integration of AI and cybersecurity, and cybersecurity education.