CVPR 2025 Anti-UAV Tracking and Detection Challenge

Tomasz Frelek
Department of Computer Science
The Ohio State University
Columbus, OH 45210
frelek.1@osu.edu

Sammy Aidja
Department of Computer Science
The Ohio State University
Columbus, OH 45210
aidja.3@osu.edu

Raagul Sundaralingam
Department of Computer Science
The Ohio State University
Columbus, OH 45210
sundaralingam.6@osu.edu

Abstract

The rapid growth of unmanned aerial vehicles (UAVs) has introduced significant security risks, necessitating the development of detection and tracking systems capable of operating under real-world conditions. In this project, we investigate two approaches to the UAV detection and tracking problem as part of the Anti-UAV Challenge. A baseline tracking method utilizing Siamese Fully-Convolutional Networks (SiamFC) and an advanced detection method based on the YOLOv11 architecture. To facilitate training and evaluation, we curated a representative subset of the Anti-UAV dataset and adapted existing model implementations to accommodate the dataset's format. Our experimental results demonstrate that SiamFC suffers from substantial limitations in the presence of occlusions, scale variation, and re-identification scenarios. In contrast, the YOLO-based detection approach consistently achieved superior performance, demonstrating higher precision, recall, and robustness across diverse environments. These findings highlight the advantages of detection-based frameworks for reliable UAV monitoring.

1 Introduction

The proliferation of UAVs has brought serious security concerns across various sectors. UAVs misuse poses risks ranging from privacy invasion to threats in restricted airspaces. Detecting and tracking unauthorized UAVs, especially in complex and cluttered environments, remains a challenging task due to factors such as occlusions, varying object scales, rapid motion, and dynamic backgrounds.

The Anti-UAV Challenge[1], hosted as part of CVPR 2025, aims to advance research in this critical area by providing a standardized benchmark for evaluating UAV detection and tracking algorithms. The challenge involves identifying and following UAVs in a series of annotated videos that simulate real-world complexities, including small object sizes, motion blur, and environmental noise. The provided Anti-UAV dataset[2] contains diverse scenarios designed to rigorously test the robustness and generalization capabilities of proposed methods.

In this project, we experiment with two methods to develop an effective algorithm to participate in the Anti-UAV Challenge. Specifically, we employ SiamFC[3] tracking and we implement YOLO11[4].

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

By experimenting with SiamFC as the baseline approach, and YOLO as our advanced approach, we seek to address the unique challenges posed by UAV detection and tracking.

2 Motivation

The detection and tracking of UAVs has become an increasingly critical problem as UAVs are now widely accessible and capable of causing substantial disruption[5]. UAVs present security threats ranging from physical attacks to significant economic losses. These growing risks highlight the urgent need for UAV detection systems capable of operating reliably in complex, real-world environments.

However, detecting UAVs is a unique challenge. UAVs are typically small, fast-moving objects. Traditional detection methods often struggle to handle these challenges, necessitating the development of advanced solutions.

Working on this problem provides an opportunity to apply and extend concepts learned in class. Topics such as convolutional neural networks (CNNs), object detection, data preparation, and handling occlusions are central to the UAV detection problem. This project allows us to push these foundational techniques into realistic scenarios.

3 Approach

To address the UAV detection and tracking task posed by the Anti-UAV Challenge, we evaluated two distinct approaches: a tracking-based baseline, SiamFC, and a more sophisticated detection-based advanced method, YOLO. This section outlines the methodology, strengths and limitations of both approaches, highlighting how each contributes to solving the core objectives of the challenge.

3.1 Baseline Approach

For our baseline approach, we implemented a Siamese Fully-Convolutional Network (SiamFC) for UAV tracking. The baseline approach was adapted from an implementation of SiamFC from an open source GitHub repo by Rafael Eller[9]. SiamFC is a relatively lightweight, end-to-end tracker that formulates tracking as a similarity learning problem between an exemplar image (the UAV in the initial frame) and subsequent search frames. The network outputs a response map indicating the UAV's location in each frame.

SiamFC was selected as the baseline due to its simplicity, efficiency, and minimal training requirements. However, as a tracker-only method, it requires manual initialization of the UAV in the first frame and struggles with long-term tracking challenges like occlusion, appearance changes, and re-identification after disappearance. Furthermore, since it lacks a detection component, SiamFC cannot autonomously detect new UAVs entering the scene.

3.2 Advanced Approach

Our advanced approach focused on object detection using YOLO, specifically leveraging the YOLOv11 architecture for UAV detection. YOLO performs frame-wise detection, identifying UAVs in each frame independently. This enables automatic detection and localization of UAVs without prior manual input.

We specifically utilized the YOLO11s model, developed by Ultralytics[6]. This approach allowed us to detect UAVs across varying scenes and re-detect them after occlusion or disappearance, a significant advantage over pure tracking methods. YOLOs fast inference speed and high detection accuracy make it a strong candidate for real-time UAV monitoring.

3.3 Computational Resources

Training both the SiamFC and YOLO models is computationally demanding due to the large datasets and complex architectures involved[7][8]. To efficiently train these models, we leveraged external computational resources with GPU acceleration, which significantly reduced training time and allowed for experimentation with different configurations.

3.3.1 Baseline Approach Resources

For the baseline approach, the implementation of generating a PyTorch Dataset object and the process of training the model on this dataset was modified to fit the provided drone dataset. The training process was then run using 4 CPUs and 2 GPUs on the Ohio Supercomputer[10]. The model was trained for 5 epochs which took approximately 10 hours.

3.3.2 Advanced Approach Resources

For the advanced approach, we trained a YOLO11s model, a lightweight version of the most capable YOLO version with approximately 9.4 million parameters[11], optimized for a balance of speed and accuracy. The model was obtained from Ultralytics, and was run using Ultralytics' inbuilt pipeline. Due to the computational load of training YOLO on the Anti-UAV dataset, we utilized Google Colab[12], which provided access to NVIDIA T4 GPUs (16GB VRAM). Training was limited by Colab's runtime quotas (about 4 hours per session), and we trained the model for 30 epochs, at a rate of a little over an hour per 10 epochs.

4 Experiments

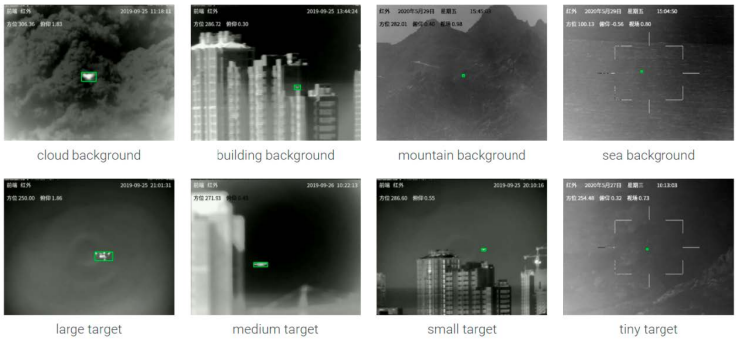

Figure 1: Examples different types of single-UAV target against various backgrounds contained within the dataset

4.1 Data

Our experiments were conducted on the Anti-UAV Challenge dataset, which contains more than 300,000 individual frames extracted from 224 unique video sequences. These videos depict UAVs in diverse and challenging environments, including varying weather conditions, lighting changes, occlusions, complex backgrounds, and varying UAV amounts. Each frame is annotated with bounding boxes for UAVs and a visibility flag that indicates whether the UAV is visible or occluded.

Given the substantial size of the dataset and the high redundancy between consecutive frames in video sequences, we performed data cleaning and subsampling to reduce computational overhead during training. Specifically, we filtered out frames with incomplete annotations and subsampled the dataset to approximately 20,000 representative frames. We created two different subsets of data, reflecting the different requirements of SiamFC and YOLO. These subsets retained a balanced distribution of UAV appearances across different conditions, ensuring that the models are exposed to sufficient variability while keeping the training process tractable.

These curated subsets of images were then used for both training and evaluation across our baseline and advanced approaches.

4.1.1 Baseline Approach Data Modification

For SiamFC training, we transformed the dataset into template-search pairs. From each video sequence, we extracted the initial frame's bounding box to create a 127×127 template of the target drone, then paired it with 255×255 search regions from subsequent frames (within 10-frame separation). This process generated approximately 400,000 training pairs from our 20,000 representative frames.

We implemented a balanced loss function to address foreground-background class imbalance in the response maps, and evaluated performance using 4,000 frames from separate test sequences.

4.1.2 Advanced Approach Data Modification

To subsample a training set for YOLO, 20,000 frames were randomly subsampled across all sequences. This ensured a broad coverage of different UAV appearances, backgrounds, and conditions, allowing the model to generalize well across various scenarios.

Furthermore, the original annotation format was converted from pixel valued $[left, top, width, height]$ to the normalized coordinates of $[x - center, y - center, width, height]$ compatible with YOLO11.

4.2 Metrics

Both the SiamFC and YOLO models were primarily evaluated using the mAP50 statistic, which is a standard metric used in computer vision. Additionally, SiamFC was also evaluated by other complementary statistics such as calculating IoU (Intersection over Union), success rate (percentage of frames with IoU > 0.5), and precision (percentage of frames with IoU > 0.2). YOLO, in turn, was evaluated with precision, recall (as well as the corresponding F1 curve), and higher mAP thresholds of 50-95.

4.3 Results

Due to the CVPR competition challenge ending on March 9th, the results couldn't be uploaded to be evaluated on official test data. Instead, the original dataset was subsampled (ensuring no overlap with either training set), and evaluated the models on 4000 UAV images it hadn't seen before.

To evaluate the performance of our models, we used different evaluation strategies tailored to the nature of each model and the availability of tools.

For SiamFC, a custom evaluation script was developed since no public evaluations existed for our use-case. Our script initializes the SiamFC tracker with the first frame's ground truth bounding box, then allows it to track the drone through subsequent frames without further guidance. For each frame, we compare the predicted bounding box with the ground truth annotation, calculating IoU, success rate, precision, and mAP50. This approach allowed us to quantitatively assess SiamFC's ability to maintain accurate drone tracking across challenging scenarios.

For the YOLO detection model, we utilized the Ultralytics platform, which provides a comprehensive evaluation suite for object detection tasks[13]. This includes the computation of standard metrics such as: mAP@50, Precision, Recall, and other IoU metrics.

4.3.1 Baseline Approach Results

Mean IoU	Success Rate	Precision	mAP50
0.299	0.391	0.409	0.364

Table 1: Performance of the SiamFC baseline model on the drone-tracking test set

Our SiamFC baseline model, trained on the drone tracking dataset, demonstrated varied performance across different test sequences.

Performance analysis revealed significant variations across different sequences. The tracker performed exceptionally well on some sequences where the model maintained consistent tracking with high precision.

However, the model struggled significantly with other sequences where the tracker essentially failed to maintain any meaningful overlap with the target drone.

These disparate results highlight the context-dependent nature of the SiamFC tracker. When conditions are favorable - with consistent drone appearance, minimal background clutter, and moderate motion - the tracker can achieve excellent results. However, the model's performance deteriorates

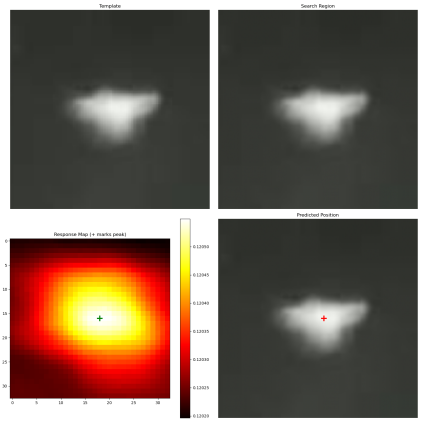

Figure 2: Template (top-left), search region (top-right), response map with peak marked (bottom-left), and predicted position (bottom-right) for one test frame of the drone-tracking dataset.

dramatically in more challenging scenarios involving rapid motion, significant appearance changes, or complex backgrounds.

While the average metrics appear modest, the strong performance on several sequences demonstrates the potential of the approach when environmental conditions are favorable. These results provide valuable insights for our advanced approaches, particularly the need to develop more robust feature representations that can handle the extreme variability in drone appearance and movement patterns across different environments.

4.3.2 Advanced Approach Results

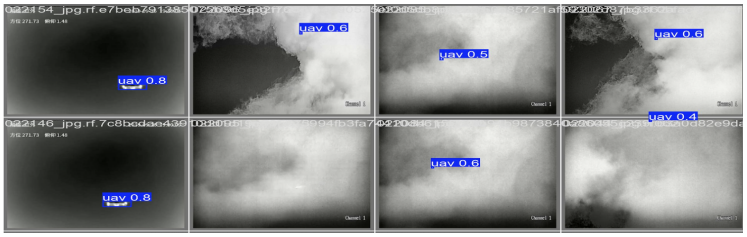

Figure 3: Predicted bounding boxes for UAV's by YOLO model.

Class	Images	Instances	P	R	mAP@50	mAP@50-95
UAV	2084	2083	0.96	0.834	0.88	0.529

Table 2: Evaluation metrics for YOLO detection model on the Anti-UAV validation set.

The performance of the YOLO-based detection model was evaluated on the Anti-UAV Challenge validation set using the Ultralytics evaluation framework. The results are summarized in Table 2, which reports key detection metrics, including Precision (P), Recall (R), mean Average Precision at IoU threshold 0.5 (mAP@50), and mean Average Precision across IoU thresholds from 0.5 to 0.95 (mAP@50-95).

The model was tested on 2084 images, containing 2083 UAV instances (note that not all images contained a UAV instance, while others contained multiple). The Precision of 0.96 indicates that the model makes very few false positive detections, while the Recall of 0.834 reflects a strong ability to correctly detect UAVs present in the frames. The mAP@50 score of 0.88 demonstrates high detection accuracy at the standard IoU threshold of 0.5, while the mAP@50-95 score of 0.529 shows a decrease in accuracy across stricter IoU thresholds.

In addition to these metrics, we analyzed the F1-Confidence curve (shown in Figure 4) to understand the model's trade-off between precision and recall across varying confidence thresholds. The F1

score, which balances precision and recall, peaked at 0.89 at a confidence threshold of approximately 0.296. This suggests that the optimal balance between detecting true positives and avoiding false positives occurs at a relatively low confidence level, reflecting the challenging nature of the UAV detection task (e.g., small object size, occlusions).

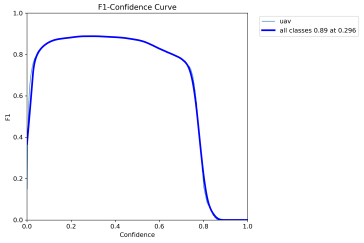

Figure 4: F1-Confidence curve for YOLO detection model on the Anti-UAV validation set. The peak F1 score of 0.89 occurs at a confidence threshold of 0.296.

5 Conclusion and Discussion

Detecting and tracking UAVs is a challenging problem due to their small size, fast motion, and tendency to blend into environments. In this project, we explored two approaches to address this challenge. A basic tracking-based method using SiamFC, and an advanced detection-based method using YOLO. While SiamFC struggled with occlusions and re-identification, YOLO consistently detected UAVs and handled challenging scenarios better. Overall, detection-based methods proved more effective for real-world UAV tracking.

5.1 Insights

Our experiments revealed clear differences between the baseline, SiamFC, and advanced, YOLO, approaches in addressing the UAV detection task. SiamFC struggled significantly with occlusions, scale changes, and re-identification after the UAV disappeared from view. These limitations were reflected in its modest performance, achieving a mean IoU of 0.299 and a success rate of 39.1%. In contrast, the advanced YOLO-based approach effectively addressed these deficiencies by detecting UAVs independently in each frame, enabling automatic re-detection after occlusion or disappearance. This resulted in significantly better performance, with the YOLO model achieving a precision of 96%, recall of 83.4%, and mAP@50 of 88%. The ability of YOLO to localize UAVs consistently, even under challenging conditions, demonstrated the advantage of detection-based methods.

5.2 Workload

In general, the workload included preparing a manageable version of the dataset, modifying existing model implementations for compatibility, training both models, and evaluating their results. Sammy was responsible for subsampling and setting up the Anti-UAV dataset for the models. Raagul focused on implementing the baseline approach by modifying an open-source SiamFC implementation to make it compatible with the Anti-UAV dataset, adapting it to create template-search pairs, and training the model. Tomasz worked on the advanced approach by converting the data set into YOLO compatible format, setting up the training pipeline for YOLOv11, and evaluating its performance using the Ultralytics framework.

5.3 Challenges

One of the primary challenges we faced was the sheer size of the Anti-UAV dataset, which required substantial preprocessing to make the training process feasible within our computational constraints. Another major challenge was modifying both the SiamFC and YOLO implementations to work correctly with the dataset, as the original models were not directly compatible with the provided formats. Additionally, training the SiamFC model posed a challenge due to the long training times. Even with access to GPUs, training for just five epochs took approximately 10 hours.

References

[1] Anti-UAV Workshop Challenge – CVPR 2025. anti-uav.github.io, https://anti-uav.github.io/. Accessed 26 Apr. 2025.

[2] Anti-UAV Dataset. Google Drive, https://drive.google.com/drive/folders/1hEGq14WnfPstYrI$_9 Ogsc R1 V sW c5_X Dl. Accessed 26 Apr. 2025.

[3] Bertinetto, Luca, et al. Fully-Convolutional Siamese Networks for Object Tracking. arXiv, 2021, https://arxiv.org/abs/1606.09549.

[4] YOLOv11 Models Documentation. Ultralytics, https://docs.ultralytics.com/models/yolo11/. Accessed 26 Apr. 2025.

[5] Campanile, Carl. "Mysterious Drones Shut Down Runways of NY Airport, Causing Hochul to Demand Feds Step in: 'Gone Too Far.'" New York Post, 14 Dec. 2024, https://nypost.com/2024/12/14/us-news/mysterious-drones-shut-down-runways-of-ny-airport-causing-hochul-to-demand-feds-step-in-gone-too-far/. Accessed 26 Apr. 2025.

[6] Ultralytics. https://www.ultralytics.com. Accessed 26 Apr. 2025.

[7] Fabbri, Filippo, et al. YOLOv11: A Vision Transformer Extension for Real-Time Object Detection. arXiv, 2024, https://arxiv.org/abs/2410.17725v1. Accessed 26 Apr. 2025.

[8] Bertinetto, Luca, et al. Fully-Convolutional Siamese Networks for Object Tracking. University of Oxford, 2016, https://www.robots.ox.ac.uk/ vedaldi/assets/pubs/bertinetto16fully.pdf. Accessed 26 Apr. 2025.

[9] Eller, Rafael. Pytorch-SiamFC. GitHub, 2019, https://github.com/rafellerc/Pytorch-SiamFC. Accessed 26 Apr. 2025.

[10] Ohio Supercomputer Center. https://www.osc.edu. Accessed 26 Apr. 2025.

[11] YOLOv11s vs YOLOv11n: Compare Model Sizes. Roboflow, https://roboflow.com/compare-model-sizes/yolo11s-vs-yolo11n. Accessed 26 Apr. 2025.

[12] Google Colaboratory. Google, https://colab.research.google.com/. Accessed 26 Apr. 2025.

[13] Ultralytics Command Line Interface (CLI) Documentation. Ultralytics, https://docs.ultralytics.com/usage/cli/. Accessed 26 Apr. 2025.