WILEY

Computational Intelligence

ORIGINAL ARTICLE OPEN ACCESS

# Vision-Based UAV Detection and Tracking Using Deep Learning and Kalman Filter

Nancy Alshaer[1] | Reham Abdelfatah[2] | Tawfik Ismail[2, 3] | Haitham Mahmoud[4]

[1]Department of Electronics and Electrical Communication, Faculty of Engineering, Tanta University, Gharbiya, Egypt | [2]National Institute of Laser Enhanced Sciences, Cairo University, Giza, Egypt | [3]Energy, Industry, and Advanced Technologies Research Center, Taibah University, Madinah, Saudi Arabia | [4]College of Computing, Birmingham City University, Birmingham, UK

**Correspondence:** Haitham Mahmoud (Haitham.mahmoud@bcu.ac.uk)

**Received:** 7 October 2024 | **Revised:** 1 January 2025 | **Accepted:** 19 January 2025

**Funding:** This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

**Keywords:** computer vision | deep learning | Kalman filter | UAV Detection | UAV tracking | YOLO

## ABSTRACT

The rapid increase in unmanned aerial vehicles (UAVs) usage across various sectors has heightened the need for robust detection and tracking systems due to safety and security concerns. Traditional methods like radar and acoustic sensors face limitations in noisy environments, underscoring the necessity for advanced solutions such as deep learning-based detection and tracking. Hence, this article proposes a two-stage platform designed to address these challenges by detecting, classifying, and tracking various consumer-grade UAVs. The tracking efficacy of the proposed system is assessed using a combination of deep learning and Kalman filter techniques. Specifically, we evaluate models such as YOLOv3, YOLOv4, YOLOv5, and YOLOx to identify the most efficient detector for the initial detection stage. Moreover, we employ both the Kalman filter and the Extended Kalman filter for the tracking stage, enhancing the system's robustness and enabling real-time tracking capabilities. To train our detector, we construct a dataset comprising approximately 10,000 records that capture the diverse environmental and behavioural conditions experienced by UAVs during their flight. We then present both visual and analytical results to assess and compare the performance of our detector and tracker. Our proposed system effectively mitigates cumulative detection errors across consecutive video frames and enhances the accuracy of the target's bounding boxes.

## 1 | Introduction

In recent years, the demand for unmanned aerial vehicles (UAVs) has surged across diverse sectors in social, commercial, military, and entertainment applications [1, 2]. This increases the significance of the pressing need for robust detection and tracking systems, driven by concerns over safety and security [3]. The potential risks associated with UAV misuse, including the possibility of collisions with aircraft, highlight the urgency of developing reliable detection and tracking methods [4].

Despite their small size and moderate speed, UAVs can cause significant damage in negligent or malicious scenarios. Therefore, accurately identifying and tracking these fast-moving objects is crucial [5, 6]. Traditional tracking methods like radar and acoustic sensors have limitations, particularly in noisy environments [7, 8]. On the other hand, advancements in deep learning and camera technology offer promising solutions [9]. Within deep learning, Deep Convolutional Neural Networks (DCNNs) have emerged as powerful tools for object detection. Two-stage detectors, such as Faster R-CNN and region-based fully convolutional network R-FCN, have demonstrated promise in efficiently locating UAVs in real-time scenarios [10–12]. Moreover, one-stage detectors like You Only Look Once (YOLO) and Single Shot Detector (SSD) provide rapid UAV detection using global image features [13, 14]. These methods offer essential capabilities for safety and security applications. However, they come with limitations of the inefficiency in addressing background confusion and blur, which led to the motivation to employ a two-stage method. This approach improves the accuracy and reliability of UAV identification and monitoring by including independent stages for object detection and tracking [15, 16].

Tracking by detection approaches utilising deep learning for detection and data association methods such as the Kalman filter (KF) offers real-time solutions crucial for maintaining visual contact with UAVs [17, 18]. The KF plays a vital role in predicting the trajectory of UAVs based on noisy measurements obtained during detection, enhancing overall tracking accuracy. The KF algorithm is considered a no-prior knowledge method that uses a series of noisy measurements containing uncertainty to predict the upcoming state. KF is an algorithm that predicts the future position of a UAV by using a series of noisy and uncertain measurements to estimate the UAV's current state. It operates by iteratively refining these predictions through a feedback loop that minimizes the estimation error, thereby enhancing the accuracy of tracking. On the other hand, the Extended Kalman Filter (EKF) is an adaptation of the standard KF designed to handle non-linear models by linearizing them around the current estimate. Unlike the KF, which is limited to linear systems, the EKF extends its applicability to more complex, real-world scenarios where the UAV's motion and sensor readings exhibit non-linear behavior. Both the KF and extended Kalman filter (EKF) [19] are presented to predict the trajectory of our target from the noisy sequence of detector measurements. In the tracking process, the bounding boxes obtained from the detection network represent the tracked targets. Moreover, trackers based on correlation filters, such as Kernel Correlation Filter (KCF), Tracking-Learning-Detection (TLD), and Structured Output Tracking with Kernels (Struck), provide robust performance in tracking UAVs [20, 21].

However, these methods have a deficiency in learning comparatively simple models since they extract the examples from the video itself, so the data is derived from the current video exclusively [9, 22, 23]. Moreover, some trackers require prior knowledge of the template matching method and the mean shift method. Some of those require a particular template to match the target image in steady and dynamic states [24, 25]. In this case, prior knowledge is defined as a template that is needed by the algorithm initially.

Detection and tracking of UAVs as well as distinguishing them from other flying objects based on classification into consumer-grade categories have been introduced in the literature but have not been implemented [3, 26, 27]. UAV classification is proposed depending on differentiating UAV from other flying objects, introducing three consumer-grade UAVs [26]. Moreover, the results of this survey [3] highlight that one of the problems in drone detection is specifying the drone type. Another noticed problem in this research domain is the lack of data for evaluation of the work. Hence, we created our dataset to verify the presence of a UAV distinguishing it from other flying objects.

This article is concerned with the visual detection of UAVs in adverse weather or environments with frequently appearing obstacles where detection models lose the target object. For this purpose, a reliable platform is developed for detecting, classifying, and tracking the UAV in real-time scenarios with a fast and accurate algorithm using a two-stage approach. The proposed platform integrates a detection system (YOLOv5) and a tracking system (KF or EKF) to guarantee a precise continual-based estimated output. The system performance is investigated visually and analytically in terms of root mean square error (RMSE), in three cases standalone YOLOv5, YOLOv5 with KF, and YOLOv5 with EKF.

The main contribution of this article can be summarised as follows:

1. Enhanced Tracking Approach: A development of a two-stage platform to enhance the tracking approach of UAVs. The proposed system utilises a combination of various deep learning algorithms (YOLOv3, YOLOv4, YOLOv5, and YOLOx) for the detection stage and two Kalman filter techniques (KF and EKF) for the tracking stage. This platform is designed to detect, classify, and track various consumer-grade UAVs with an emphasis on (a) real-time tracking capabilities, (b) minimising errors introduced by the detection system through rectifying its output; (c) compensating for dropouts resulting from occlusion, blur or other obstructions; and (d) verifying the presence of a UAV while distinguishing it from other flying objects.
2. Construction of Comprehensive Dataset: To train the detector, a dataset comprising approximately 10,000 records is constructed. This dataset captures the diverse environmental and behavioural conditions experienced by UAVs during their flight, ensuring the effectiveness of the detection system across various scenarios. Three videos have also been generated for evaluating the KF and EKF algorithms with the detection. This dataset aims to verify the presence of UAVs, distinguishing them from other flying objects, and classifying UAVs based on their type. This dataset is made available as open-source data, ensuring accessibility and reproducibility for further research and development[1].
3. Performance Evaluation: The system's performance is rigorously evaluated using root mean square error (RMSE) across various test scenarios to assess tracking accuracy and reliability. Comparisons are drawn between different configurations (YOLOv5 alone, YOLOv5 with KF, and YOLOv5 with EKF), demonstrating significant improvements in tracking precision when integrating deep learning with advanced filtering techniques.

The remainder of this article is organised as follows. Section 2 addresses the evolution to computer vision for UAV tracking as well as discusses the existing studies. Section 3 proposes the framework and its functions, including detection approaches, and tracking approaches. Section 4 discusses the generated dataset, and discusses the detection and tracking results based on visual and analytical. Section 5 concludes the work and highlights the future work.

## 2 | Background and Related Works

### 2.1 | Evolution to Computer Vision for UAV Detection

In the early stages of UAV detection, the field relied heavily on heuristic algorithms and traditional image-processing techniques. These methods were effective in controlled environments but struggled with the complexities of dynamic and varied scenarios. They relied on handcrafted features and rule-based systems, which limited their ability to adapt to changing environmental conditions and complex background clutter. As a result, early UAV detection systems faced challenges in accurately identifying UAVs in real-world applications. The landscape of UAV detection underwent a transformative shift with the introduction of deep learning, particularly Convolutional Neural Networks (CNNs). CNNs enabled systems to automatically learn and extract meaningful features from raw visual data. This breakthrough allowed for the development of more robust and accurate UAV detection systems. By leveraging large-scale datasets, CNN-based approaches could learn to differentiate between UAVs and other objects in various environmental conditions with higher accuracy and reliability.

Further advancements in real-time object detection techniques, such as You Only Look Once (YOLO) pushed the boundaries of UAV detection capabilities. These methods enabled faster and more efficient identification of UAVs in diverse scenarios. YOLO, in particular, introduced a paradigm shift by providing real-time object detection capabilities with a balance between speed and accuracy. This was crucial for applications requiring immediate responses to detected UAVs, such as surveillance and security. As deep learning models continued to evolve and datasets diversified and expanded, computer vision systems for UAV detection became more resilient and adaptable. Modern systems can now handle complex scenarios, including infrastructure management, military operations, security surveillance, and civilian applications. These advancements have paved the way for enhanced situational awareness and decision-making capabilities in UAV operations.

Looking forward, ongoing research is focused on further improving the efficiency and accuracy of UAV detection systems. This includes advancements in deep learning techniques, such as attention mechanisms and transformer-based models, to enhance object detection and tracking capabilities. Additionally, integrating multi-sensor data fusion and AI-based decision-making processes will play a critical role in enhancing the overall performance and reliability of UAV detection systems.

### 2.2 | Related Works

Tracking objects can be challenging due to obstacles, crowded surroundings, and changes in appearance caused by variations in illumination and perspective. This study aims to develop a model capable of monitoring objects with minimal labeled data, dynamically learning their appearance, and re-acquiring them if they disappear [28]. The authors train two classifiers on conditionally independent perspectives of the same data using a co-training strategy with a limited number of exemplars. They use an SVM classifier for the discriminative model, trained online using gradient histograms, and a Multiple Linear Subspace Generative Model. The subspaces are progressively updated with new samples, and LASVM, an incremental SVM method, is employed for the discriminative model [29]. Previous models do not handle partial obstruction well, despite strong results compared to existing literature. Combining offline and online training may improve tracking performance for some objects.

Another approach to track the kinematic state and purpose of highly maneuverable devices like drones uses a heavy-tailed $\alpha$-stable Levy process within a state-space model, which is effective in capturing abrupt direction changes [30]. The model integrates the kinematic state and purpose into a vector solution of a stochastic differential equation (SDE), where the stochastic term consists of Levy processes computed using a truncated Poisson series. A normal distribution calculates the transition density for a single $\alpha$-stable noise source, except for the part of the stochastic integral directly linked to Gaussian noise, which is manually computed. This study explores the combination of detection and tracking algorithms. Algorithms that simultaneously detect and track typically include point, primitive geometric shapes, skeletal models, and silhouettes in shape representation. Techniques in appearance representation involve probability density, templates, multi-view appearances, and active appearance models. Object detection methods are categorized into single-frame object methods and methods using temporal information. In tracking tasks, object detection and correspondence across sequences of frames can be performed either jointly or separately. Three families of tracking methods, including point tracking, kernel tracking, and silhouette tracking, are also discussed. Additionally, various tracking challenges such as cross-camera tracking, handling occlusions, and managing non-target objects are addressed. This study proposes a classification model using a two-stage decision tree approach for detection, classification, and tracking [31]. The model enhances performance in real-time scenarios by leveraging kinematics and micro-Doppler components. Field trials using the Aveillant/Thales GameKeeper 16U radar system showed a significant reduction in false positives compared to a single-stage approach.

In a related work, this study introduces Single Integrated Air Picture (SIAP) metrics to evaluate staring radar systems in drone surveillance [32]. These metrics include completeness, clarity (both ambiguity and spuriousness), continuity, and kinematic accuracy, essential for assessing system effectiveness. Results from the SESAR SAFIR program demonstrated improved performance metrics when using the two-stage decision tree model discussed earlier, emphasizing its positive impact on system clarity.

Another paper evaluates six state-of-the-art CNN object detectors for drone detection and tracking using a Pan-Tilt-Zoom (PTZ) camera system [32]. The study provides a comparative analysis, identifying YOLOv2 as suitable due to its balance of speed and accuracy, contributing to low-cost surveillance solutions. Additionally, this paper focuses on the ground-truthing process for supervised machine learning in drone classification

**TABLE 1** | The existing studies of UAV tracking.

| Ref | Algorithm | Purpose | Pros | Cons |
|---|---|---|---|---|
| [35] | Track While Scan (TWS) | Simultaneously track existing targets and scan for new ones | Optimizes active track tasks and detect abrupt changes | Complex implementation and high computational cost |
| [36] | Interactive-EKF | Enhance velocity predictions for small aircraft during turns | Improves accuracy in critical air traffic scenarios | Limited by assumptions of linear models |
| [37] | Interactive KF using Neural Networks | Trajectory planning in autonomous vehicles based on social interactions | Increases accuracy in trajectory predictions | Requires large training data and computationally intensive |
| [31] | Two-stage decision-tree classifier | Distinguishing drones from other targets | Improved classification accuracy | High training time and limited to radar data |
| [38] | SDD | Real-time object detection; balancing speed and accuracy | High detection accuracy; real-time performance | Low accuracy and slow detection |
| [32] | CNN | Identifying YOLOv2 as optimal for speed and accuracy | Accurate real-time drone tracking | May struggle with occlusion and complex environments |
| [39] | Neural network-enhanced Kalman filter | Improving accuracy in 3D target tracking | Superior performance compared to traditional filters | Requires large training datasets and complex model integration |
| This paper | KF/EKF+ YOLOv3, v4, v5 and YOLOx | Detecting, classifying, and tracking consumer-grade UAVs | Comprehensive dataset with diverse environmental conditions; open-source data | Real-time tracking capabilities and minimizes errors |

using multi-beam staring radars [33]. It highlights the need for accurately labeled data to differentiate drones from other targets. The study employs "Theodolite" on an iPad to gather comprehensive ground truth data, facilitating the training of a decision tree classifier with high-accuracy in distinguishing drones from non-drones.

Furthermore, this paper proposes a multi-agent system (MAS) architecture for real-time coordination of sensors onboard Remotely Piloted Aircraft Systems (RPAS) [34]. The MAS dynamically manages resources allocated to sensors, enhancing flexibility and efficiency in mission management. Permanent sensors function as agents, while tactical agents are created dynamically to handle detected targets. The MAS architecture optimizes resource allocation through real-time scheduling, ensuring effective data collection and mission performance for airborne platforms.

In terms of combining the detection with the UAV tracking. Several algorithms have been employed for UAV tracking in various studies as shown in Table 1. Track While Scan (TWS) is used to simultaneously track existing targets and scan for new ones, optimizing active tracking tasks and detecting abrupt changes, albeit being complex to implement and computationally intensive. Interactive Extended Kalman Filter (IEKF) enhances velocity predictions for small aircraft during turns, improving accuracy in critical air traffic scenarios, though it is limited by assumptions of linear models. Interactive Kalman Filter using Neural Networks is applied for trajectory planning in autonomous vehicles, incorporating social interactions to improve trajectory prediction accuracy, though it requires large training datasets and is computationally intensive.

A Two-stage Decision-Tree Classifier is introduced for distinguishing drones from other targets, offering improved classification accuracy, albeit with high training times and limited to radar data. For real-time object detection, Single Shot Detector (SSD) identifies YOLOv2 as the optimal choice, balancing speed and accuracy, providing high detection accuracy and real-time performance, though it may suffer from low accuracy and slow detection in certain conditions. Convolutional Neural Networks (CNNs) are employed to identify the best detector for real-time drone tracking, ensuring accurate real-time tracking, but facing challenges with occlusion and complex environments. Neural Network-Enhanced Kalman Filter improves accuracy in 3D target tracking, offering superior performance compared to traditional filters, though demanding large training datasets and complex model integration.

However, this field has been widely discussed as previously mentioned, none has managed to utilise KF with state-of-the-art detection such as YOLO algorithms. Hence, this article proposes a combination of various YOLO versions (v3, v4, v5, and YOLOx) with Kalman Filter (KF) and Extended Kalman Filter (EKF) for detecting, classifying, and tracking consumer-grade UAVs. This approach leverages a comprehensive dataset with diverse environmental conditions and open-source data, designed for real-time tracking capabilities and minimizing errors.

## 3 | The Proposed Framework

The proposed system consists of two stages, the detection stage, and the tracking stage. The detection is sensitive to a range of errors due to blurring, occlusion, noisy environmental factors, and the fast movement of the UAV changing its pose and orientation. The tracker based on KF or EKF is proposed to model the motion of the UAV among consecutive frames, where each frame is successfully detected with minimum localization errors.

A prediction stage in the tracker is employed to predict the trajectory of the UAV, which results in a previous estimation to give a posterior estimated value from the KF. The measurement process of the KF is responsible for calculating the posterior estimate of the current state, which is then sent to the prediction step again for trajectory prediction in a recursive loop that covers all video frames. The first stage of detection is achieved by the YOLO detector. It has the ability to classify the target UAV into a certain class to be followed by a tracking stage using KF and EKF.

### 3.1 | Detection Approaches

The proposed detection model was trained with positive and negative samples to differentiate between UAVs and other flying objects, or empty backgrounds (without objects), in order to reduce the false positives. It is also trained with different resolutions to increase the accuracy of the model, and the geometric distortion method for data augmentation is applied to increase the variability of the input images so that our proposed model has higher robustness to the dataset samples obtained from different environments. The observation models represented in this study for the detection stage are from the YOLO family of detectors. Given that quickly detecting the UAVs is more important than precise detection, YOLO is applied since it is the fastest architecture with lightweight and convenient for systems with limited computational power and memory.

YOLOv3 [40] is utilized as an observation model, where the framework used for training is Darknet open source [41]. It predicts bounding boxes depending on the highest Intersection over Union (IOU) between detected and ground truth boxes and applies non-maximum suppression in cases of multiple boxes appearance. Logistic regression is utilized to predict each bounding box occurrence confidence. In (1), the ground truth input data to the model is demonstrated, $m$ is the annotated samples, where to improve speed in real-time scenarios, various resolutions with a maximum of $416 \times 416$ are used to train the model. $n$ and $c$ represent the bounding box, and the three-class labels, respectively, where $M^s$ are the seen images and $N^s$ are the seen labels by the detector. At the last layer, a 3D tensor is predicted to encode the bounding box, object probability, and the class scores as $N \times N \times [3 \times (4 + 1 + 3)]$.

The original YOLO tensor is trained within 80 classes, here, instead, three classes are required. Also, YOLOv4 [13] from the YOLO family of detectors is trained using a CSPDarknet53 [42] backbone that improved DarkNet53 used in YOLOv3 [40]. It shares the same head as YOLOv3. The algorithm utilized the bag of freebies and the bag of specials to enhance the performance, adding slight inference costs. YOLOv4 utilizes SPP [43] to increase the receptive field over the backbone network and exploits PANet [44] to combine different network levels instead of employing FPN [45]. Since YOLOv3 and YOLOv4 were implemented in c++, YOLOv5 is being investigated for development in the PyTorch framework, written in Python. Various models of YOLOv5 architecture are provided in [46] including YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. The complexity of the architectures increases in ascending order, starting with the simplest at $s$ and ending with the most complicated at $x$. However, during the training stage, we initialized the weights using YOLOv5s pre-trained weights to save training time and reduce computational costs. In order to accommodate the flexibility of the model, the input image is prepared with a size of $640 \times 640$. It similarly utilizes the CSPDarknet53 architecture with an SPP layer as the backbone, PANet as the Neck, and YOLO detection head.

Similar to YOLOv5, YOLOX [47] utilized the Pytorch framework for implementation to ease the practical use of researchers. YOLOX detects objects in an anchor-free manner similar to YOLOv1 to lower the number of design parameters. Inconsistent with YOLOv5, YOLOx has models of YOLOX-s, YOLOX-m, YOLOX-l, and YOLOX-x. It utilizes a decoupled head instead of the coupled head used from YOLOv3 to YOLOv5 to prevent the classification and localization tasks from competing during training. We initialized the weights using pre-trained YOLOXs with $640 \times 640$ input image. According to the task criteria, the optimum detector is chosen, where the optimum efficiency of the detectors is determined in terms of mean average precision and inference time. The previously discussed detectors are investigated on our particular task and dataset to determine the suitable detector to be integrated with the tracking network.

$$D_t : (m, n, c) | m \in M^s, \left( n \in N^s,\ n = \begin{bmatrix} x \\ y \\ w \\ h \end{bmatrix} \right), c \in R^3 \tag{1}$$

The detection approach for processing a single input frame is illustrated in Figure 1. When verifying the presence of a particular UAV, the model specifies the location of the target object

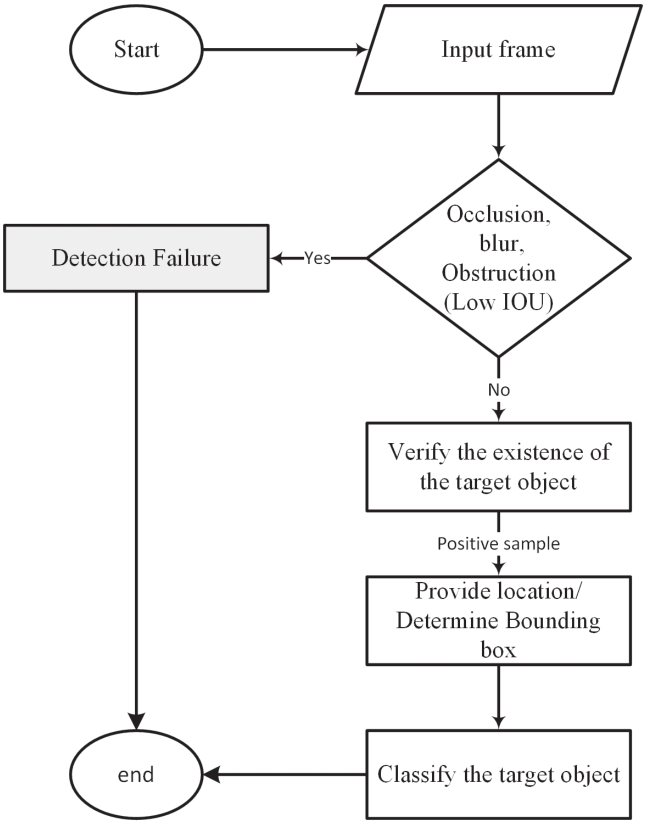
Start
Input frame
Occlusion, blur, Obstruction (Low IOU)
Yes
Detection Failure
No
Verify the existence of the target object
Positive sample
Provide location/ Determine Bounding box
Classify the target object
end

**FIGURE 1** | Flow chart of a single frame processing through YOLO detection network.

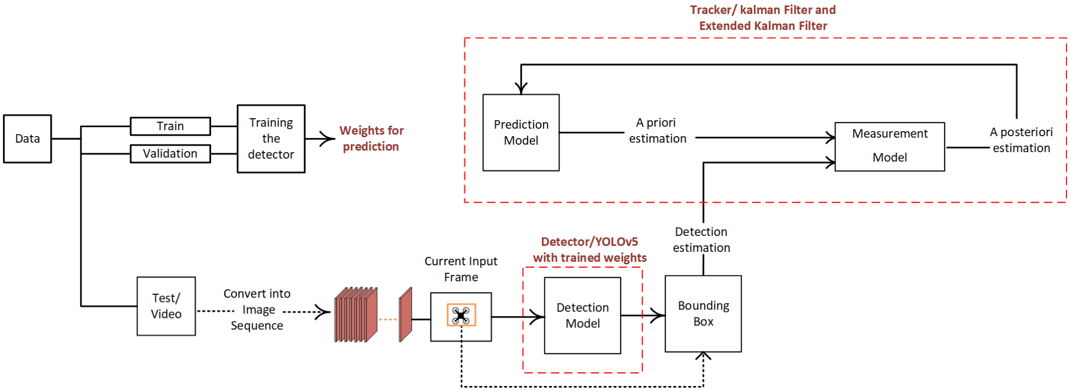

**FIGURE 2** | The proposed system diagram. A priori estimation and a posteriori estimations are the outputs of the prediction step and measurement step respectively.

through a four-dimensional vector representing the center, aspect ratio, and proportion of the target object. It is categorized into a certain class to establish the tracking stage. Unfortunately, any sort of obstruction will direct the detection approach into losing sight of the UAV in this frame, leading to the failure of the detection. So, in this work, the detection network, the YOLO model, is integrated with the tracking network in order to maintain the UAV trajectory. This is achieved by considering the correlation between neighboring frames during the tracking stage, as discussed in detail in the following subsection.

## 3.2 | Tracking Approaches

The framework of the proposed system is presented in Figure 2, where the detection stage is enhanced by a tracking stage in a recursive loop. The Kalman filter algorithm is proposed here as the tracker, where it predicts the position and velocity of the UAV. The sequence of measurements taken by the detector is used to adjust the bounding box precision. The position of the object is estimated using prediction and measurement models, which are then used to evaluate the trajectory of the UAV. However, because the motion is not completely linear, we developed the Extended Kalman filter to model the non-linearity of motion by substituting the Jacobian matrix for the Kalman filter constants.

### 3.2.1 | Linear Kalman Filter Estimation Model

To employ location estimation using the Kalman filter algorithm, a motion state variable is defined as $v = [x, \dot{x}, y, \dot{y}, w, \dot{w}, h, \dot{h}]$, where $[x, y]$ are the coordinates of the center of the target position $(p_x, p_y)$, and their following values are the derivatives representing the velocity of the target along the x-axis and y-axis $(v_x, v_y)$ respectively. Also, $[w, h]$ describes the proportion and aspect ratio of the target object, which is also representing the width and height with their derivatives, where the vector $s$ is utilized for describing the motion route for the moving target using state equation and observation equation in (2a) and (2b), respectively.

$$s_k = F s_{k-1} + B u_{k-1} + w_{k-1} \tag{2a}$$

$$y_k = H s_k + v_k \tag{2b}$$

where $s_k$ and $y_k$ denote the state vector and the measurement vector at the $k^{th}$ instant, respectively. $F$ is known as the state transition matrix that is utilized to propagate the state at instant $k-1$ to instant $k$ to describe how each state moves over one time instant, by capturing all related internal dynamics of the system. The matrices $B$ and $u$ are neglected in such a non-controllable system. The transformational matrix $H$ maps state to measurement value, and $w_k$ represents prediction noise with covariance $Q$, and $v_k$ represents measurement noise with covariance $R$. In the prediction step, the Kalman filter predicts the current motion state estimate $\hat{s_k}^-$ in (3a), and $P_k^-$ in (3b) which is the prediction error covariance matrix between the predictive value and true value, for the first frame the Kalman filter is initialized through ground truth or the detector's output of high confidence value.

$$\hat{s_k}^- = F \hat{s}_{k-1} \tag{3a}$$

$$P_k^- = F P_{k-1} F^T + Q \tag{3b}$$

The following step, which is the measurement step, aims to estimate the state through prediction and measurement of its current state, where Kalman gain is computed as in (4a) to reflect the uncertainty in prediction in regards to the measurement of the detector, giving $\hat{s_k}$ in (4b) and $P_k$ in (4c) as state estimation and error covariance matrix, respectively, which are prepared for the recursion process.

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \tag{4a}$$

$$\hat{s_k} = \hat{s_k}^- + K_k (y_k - H \hat{s_k}^-) \tag{4b}$$

$$P_k = (I - K_k H) P_k^- \tag{4c}$$

### 3.2.2 | Extended Kalman Filter Estimation Model

As shown in Figure 3, in the EKF algorithm, instead of modeling the motion of the UAV by a random acceleration in $x$ and $y$−axis, it is modeled by its speed along with the diagonal $v(t)$ and heading direction $\theta(t)$. It first linearizes the model by applying first-order Taylor series approximation around $s_{n-1}$ (5a). By substituting matrices, we obtain the Jacobian matrix $J_s$ as in (5b) of $f(s(t))$, then it applies the prediction and measurement steps as discussed earlier. While $F$ and $H$ are not fixed as in the Kalman filter, they are adjusted by computing the Jacobian as in (5c) and (5d) giving $F_k$ and $H_k$ that are updated at each time step $k$.The procedure is summarized in Algorithm 1.

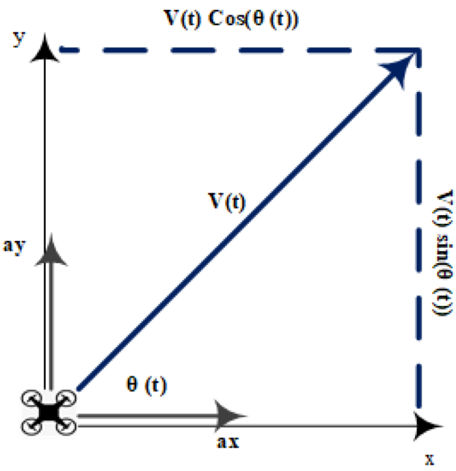

**FIGURE 3** | Motion model of UAV in linear and non-linear states.

$$f(s(t)) \approx f(s_{n-1}) + J_s(s(t) - s_{n-1}) \tag{5a}$$

$$J_s = \begin{bmatrix} 0 & 0 & \cos(\theta_{k-1}) & -v_{k-1}\sin(\theta_{k-1}) \\ 0 & 0 & \sin(\theta_{k-1}) & v_{k-1}\cos(\theta_{k-1}) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5b}$$

$$F_k = \left(\frac{\partial f}{\partial s}\right)_{\hat{s}_{k-1}} \tag{5c}$$

$$H_k = \left(\frac{\partial h}{\partial s}\right)_{\hat{s}_k} \tag{5d}$$

As previously stated, the while loop iterates through $N$ frames to cover all of the video frames extracted. The threshold is adjusted to accept the output of the detector depending on the confidence value, and in the instance of low confidence, the tracker does not lose the target since it corrects the measured value from the detector using a previously successful frame. The measurement step in the KF and EKF combines the detection network output with the prior estimation of the prediction step, to get a posteriori accurate estimated output. Assuming EKF is adopted, the previous approach updates the state vector $s$ values in each iteration, and the parameters $F$ and $H$ are updated in each iteration. The detection task pursues the tradeoff between accuracy and optimal speed. The detection of a UAV in every frame is subjected to various obstacles that ravel the detection, where the tracking task supports a continuous framing to the target object. The tracking task should be robustly integrated with the best performance detector.

## 3.3 | Evaluation Metrics

### 3.3.1 | Detection

The performance evaluation metrics to detect UAVs include numerous crucial factors. First, the Mean Average Precision (mAP) at an Intersection over Union (IoU) criterion of 0.25 measures the detection model's average precision across various confidence levels. It estimates the average precision for each class and then averages them to get an overall measure of detection accuracy as follows:

**ALGORITHM 1** | Detector-Tracker algorithm for UAV detection and tracking system.

```
$k = 1$
Set Threshold ← 0.5 IOU for detection
$P^- \leftarrow$ Initialize State Error Covariance Matrix
$p(w) \sim N(0, Q)$ ←Initiate Process Covariance Matrix
$p(v) \sim N(0, R)$ ←Initiate Measurement Covariance Matrix
while $k < N$ do
    if $k == 1$ then
        initialize state vector $x$ in first frame.
    else
        $\hat{s}_k^- = Apply\ (2a)$
    end if
    $P_k^- = Apply\ (2b)$ : Update state covariance matrix.
    $y_k, confidence_k \leftarrow throughDetector$ : Get measurements from proposed detector.
    if confidence < Threshold then
        Correct prediction through previous frame.
    end if
    $K_k = Apply\ (3a)$ : Update Kalman Gain to calculate $\hat{s}_k$ and $P_k$.
    $\hat{s}_k = Apply\ (3b)$ : Update State vector.
    $P_k = Apply\ (3c)$ : Update state covariance.
end while
```

$$mAP@0.25 = \frac{1}{n}\sum_{i=1}^{n} AP_i \tag{6}$$

Similarly, at an IoU threshold of 0.5, the mAP refines the evaluation by selecting only the predictions that overlap more with the ground truth bounding boxes. This stronger criterion provides insight into the model's precision under more demanding settings. The formula for mAP @0.5 is similar to that of mAP @0.25, but with a different IoU threshold.

In addition to precision measures, inference time is critical for determining the efficiency of the detection model. It refers to the time it takes for the model to process an input image and make predictions, which is commonly measured in seconds or milliseconds. Furthermore, Frames Per Second (FPS) complements inference time by measuring the model's speed in processing frames or images per second. It is computed as the inverse of the inference time.

$$FPS = \frac{1}{\text{Inference Time}} \tag{7}$$

### 3.3.2 | Tracking

Root Mean Square Error (RMSE) measures the average magnitude of the errors between predicted values and actual values. RMSE is used to evaluate and compare the performance of both tracking algorithms. It is calculated by taking the square root of the average of the squared differences between predicted and actual values as follows:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{z}_i - z_i)^2} \tag{8}$$

where $z_i$ and $\hat{z}_i$ are the ground truth and the system output for the bounding box center, aspect ratio and proportion, respectively.

## 4 | Simulation Results and Analysis

A dataset of about 10,000 samples is annotated for the training and testing stages to address the detection problem, given that the detector cannot see the entire dataset during the training or testing stages. Alternatively, unseen data is retained to test and evaluate the system performance, which is sampled at a higher sampling rate to account for consecutive frames adoption. The proposed two-stage system is analyzed using visual results, the introduced error by each stage is reduced for precise tracking and trajectory prediction.

### 4.1 | Measurable Dataset

Detecting UAVs is a challenging task due to several environmental and behavioral conditions surrounding their movement, which are considered during the dataset creation. UAVs have a high motion and rotation, causing their aspect ratio to change rapidly. The dataset statistics will show various sizes of UAVs in multiple locations to address the issue of UAVs appearing as small objects. It consists of training data that the detector sees and testing data that the detector can not see (unseen).

The dataset consists of 7,620 images of which 79.8% of 6,080 images are for training, 0.2% of 20 images for testing and 20% of 1,520 images for validation. All images are captured in high resolution. Some of the images are very challenging where only the shadow of the UAV is visible. The created dataset can be accessed through this link: https://github.com/HaithamHmahmoud/UAV-CDT.

#### 4.1.1 | Training Dataset

Three drone types are detected and classified in order to evaluate the proposed platform (F450, Husban, and Phantom). The introduced dataset has balanced data that can be used to verify the class label successfully. The drones were detected in [27] without considering the UAVs classification into different classes and introduced these types as consumer-grade types. This dataset consists of videos taken from YouTube [48, 49], where it contains multiple setups of the three UAVs with a diverse background to overcome the challenge of the landscape variety that faces the detection of the UAV. Also, the UAV appears at varying distances to address the detection challenges due to the motion behavior of the UAV. It comprises multiple videos (10,000 images), where 80% were used for training, and 20% were used for validation. Figure 4 shows the dataset statistics as it reveals that UAVs in this dataset appear with immense variation across the image plane, which is clear in the UAV location figure. The UAV size illustration demonstrates that the UAV is captured at long distances, appearing in the UAV to the entire image ratio. Thus, the balance of the raw data can be deduced from this figure. The dataset was manually generated and labelled. The accurate label information of the image is recorded in a text file format using Labelimg [50]. The text file mainly contains the position of the UAV coordinate data and its label. Three categories of UAVs exist: F450, Husban, and Phantom drones. Figure 5 illustrates positive data samples on the right that contain F450, Husban, and Phantom, respectively. The left side represents the negative background, with a positive to negative ratio of 7:3. Also, noisy

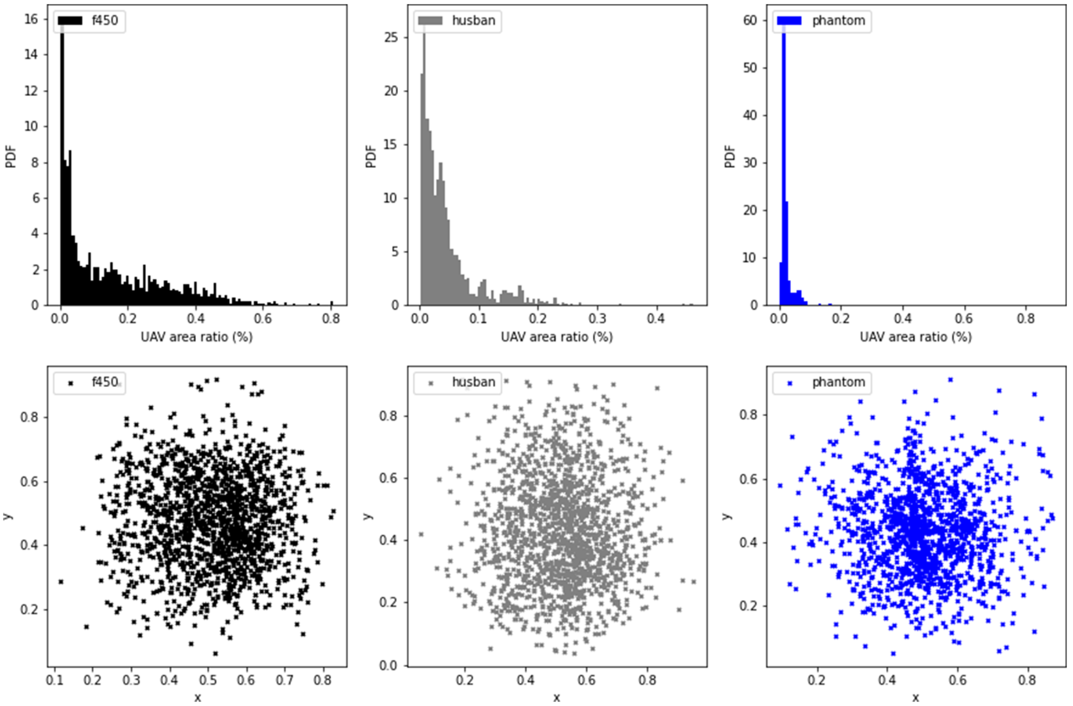

**FIGURE 4** | Dataset Statistics. The upper three figures for the UAV to image ratio for F450, Husban, and Phantom from left to right, respectively. The bottom three for the UAV's location in the image.

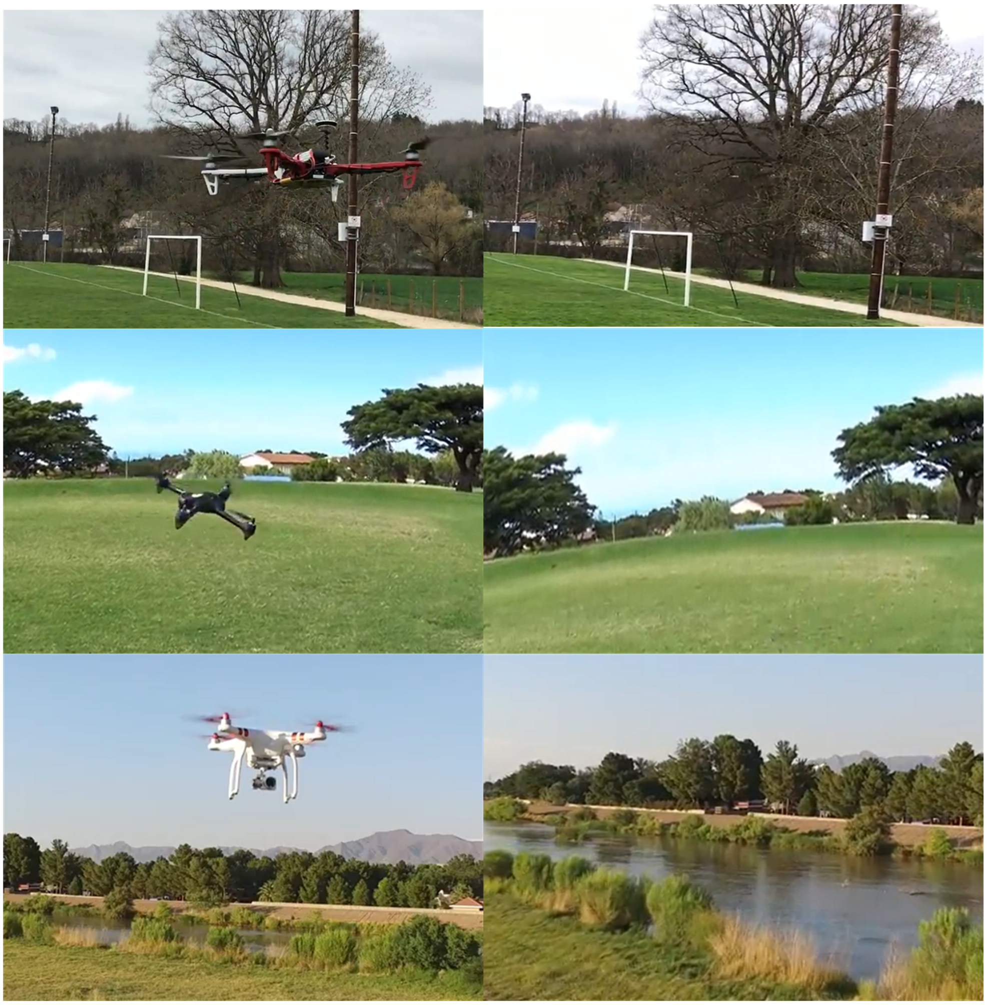

**FIGURE 5** | Positive samples with drones and negative samples with empty backgrounds.

backgrounds are considered to confirm the presence of the UAV for further detection and classification.

### 4.1.2 | Testing Dataset

The dataset used for testing is distinct from the dataset used for training, in which the frames are collected with a higher sampling rate in order to provide more consecutive frames for the tracking stage. Several videos with a total of 500 frames are utilized to test the mechanism, which is not seen by the detector. The frames are continuous for each clip in order to validate the concept of adjacent frames. Ground truth labeled annotations were constructed for these frames that were not included in the training but were utilized to evaluate both the detection and tracking networks.

## 4.2 | Detection Network Visual and Analytical Results

A proper detection model is established for the tracking system, where the detectors are evaluated through accuracy and speed to utilize the one with the best performance. The evaluation indices for detectors are the mean average precision (mAP), inference time, and frames per second (FPS) conducted on the validation data. As shown in Table 2, for a single detector, various IOU affects the detection mAP, whereby increasing the required IOU threshold, the corresponding mAP decreases. The mAP at the IOU threshold of 0.5 is the commonly used matrix for detection evaluation. IOU thresholds are demonstrated to compare the detectors. YOLOv4 has a slightly higher mAP score than YOLOv3 at 0.25 IOU, which results in a small improvement in inference time and detection speed. YOLOv5 achieves better results than YOLOv4 and YOLOX at 0.25 and 0.5 IOUs. Inference time shows the notability of YOLOv5 where it decreased by 59% compared to YOLOv3 and by 57% compared to YOLOv4 and by 7.1% compared to YOLOX. Moreover, it achieves higher FPS than YOLOv3, YOLOv4 and YOLOX by 141%, 127%, and

**TABLE 2** | Performance evaluation of detectors.

| Detection model | mAP @0.25 | mAP @0.5 | Inference time | FPS |
|---|---|---|---|---|
| YOLOv3 | 98.91% | 94.99% | 32 ms | 31 |
| YOLOv4 | 99.19% | 98.42% | 30 ms | 33 |
| YOLOv5 | 99.3% | 99% | 13 ms | 75 |
| YOLOX | 96.86% | 93.88% | 14 ms | 71 |

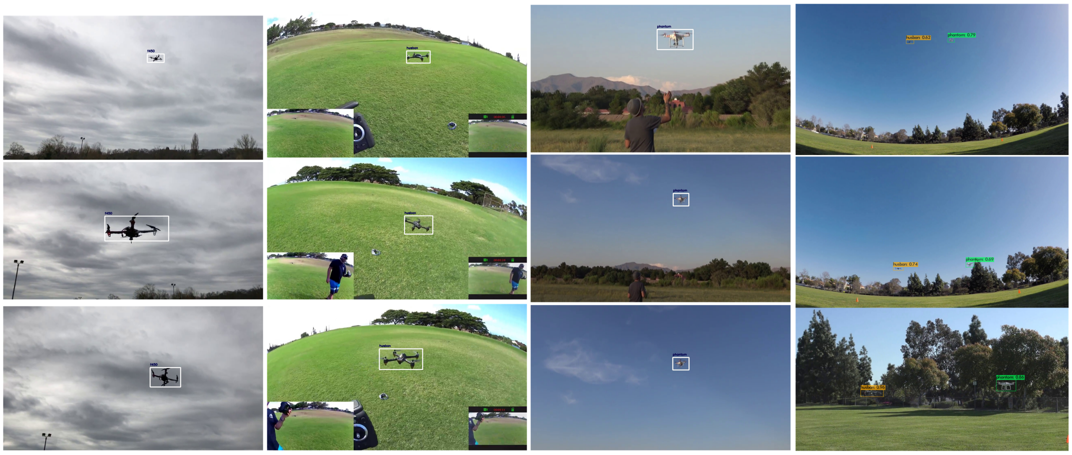

**FIGURE 6** | Snapshot of predicted results from the YOLO detector. The prediction class is stated within the bounding box. UAV types of F450, Husban, and Phantom are horizontally shown, respectively.

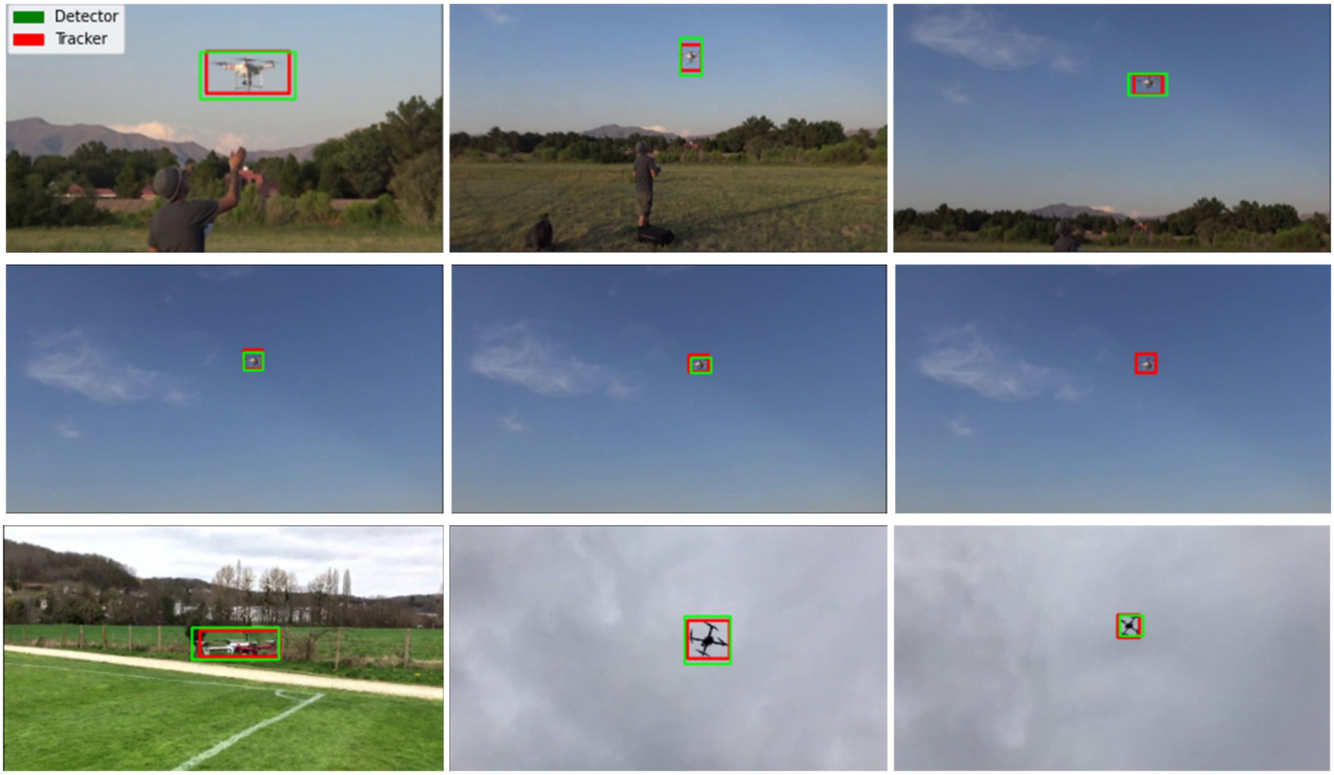

**FIGURE 7** | Snapshot of prediction using YOLO in green boxes vs. the estimated output using Kalman filter algorithm in red boxes.

5.63% respectively. Since YOLOv5 is significantly faster than prior detectors, it is integrated with the tracking system to provide higher robustness. The experiment is run on a single NVIDIA TESLA K80 GPU.

Several snapshots from various videos were taken as shown in Figure 6, where the performance of the YOLOv5 detector was evaluated using a threshold of 0.5. This stage involves detecting and categorizing any passing UAV into its corresponding class, where it is differentiated from the background, benefiting the negative sampled data.

### 4.3 | Tracking Network Analysis and Simulated Results

As stated earlier, the prediction on the current frame without considering the previous drone trajectory might result in losing sight of the UAV. The proposed tracker takes the initiative in situations of occlusion or blur, allowing the UAV to be spotted within the frame. In Figure 7 the green box represents the detector output, and the red box represents the tracker output, where the KF is used to estimate the output. The last frame in the second row shows that the detector captured the position of the UAV with low confidence, and rather than losing track of the UAV, the tracker succeeded in maintaining its trajectory. In this frame, the only thing that can be seen is a red bounding box, which indicates the output of the tracker. To evaluate the performance of the detector and tracker simultaneously, the RMSE between the system output and the ground truth is calculated.

In Table 3, we measured the detector and tracker where the system output is the YOLO prediction, estimation from the Kalman filter, and an extended Kalman filter, respectively. The ground truth used here is generated specifically for the evaluation task and was not included in the training stage. The frames in the three videos were generated with a 30 FPS sampling rate, with 50,100, and 295 Frames, respectively. These frames are used to evaluate the tracker, where they were kept aside during the training process.

The RMSE in the table represents the errors along with the bounding box four boundaries. It is visible that the detector suffers from a high error rate due to the missing frames, and the bold values are for the Kalman filter, which has the lowest error, which enhances the system performance. On the other hand, lost detector frames contributed a significant amount of inaccuracy to the system. In Figure 8, we compared the performance of the trackers across three different video streams. The Kalman filter shows smaller error values, but the EKF still gives acceptable results, implying that the linear motion model has a more appropriate motion representation and that the EKF introduces complexity to the system. Finally, the trajectory of two different UAVs is captured through different colors in Figure 9. It shows the prediction step output value detected in consecutive frames. The camera utilized in this video is a fixed camera, which allows a successful appearance of the changeable UAV path in relation to each video frame.

**TABLE 3** | RMSE (Pixels) comparison between detector and tracker.

| Video stream | YOLOv5-Detector | YOLOv5+ KF tracker | YOLOv5+ EKF tracker |
|---|---|---|---|
| Video 1 | 57.491 | **6.231** | 7.495 |
| Video 2 | 38.484 | **9.833** | 11.934 |

## 5 | Conclusion and Future Work

This paper presented a two-stage detection, classification, and tracking system for UAVs. The proposed system integrates the YOLO detector with the Kalman filter as a tracker. The tracker exploits the consecutive frames to locate the UAV in available frames and to overcome background confusion, occlusion, blur, or any obstacles causing low detection confidence when using only the YOLO detector. To include all of the challenges associated with UAV detection, a specially selected dataset is annotated and classified into various balanced UAV categories. Using the proposed data, a comparison of the trained detectors (YOLOv3, YOLOv4, YOLOv5, and YOLOX) showed that the YOLOv5 detector had a higher mAP of 99% at an IOU of 0.5, compared to the other detectors. Furthermore, the inference time of YOLOv5 is around 57% less than that of YOLOv4, making it suitable for real-time scenarios. Hence, our system performance is compared visually and analytically, in terms of RMSE, in three cases YOLOv5 only, YOLOv5 with KF, and YOLOv5 with EKF. The

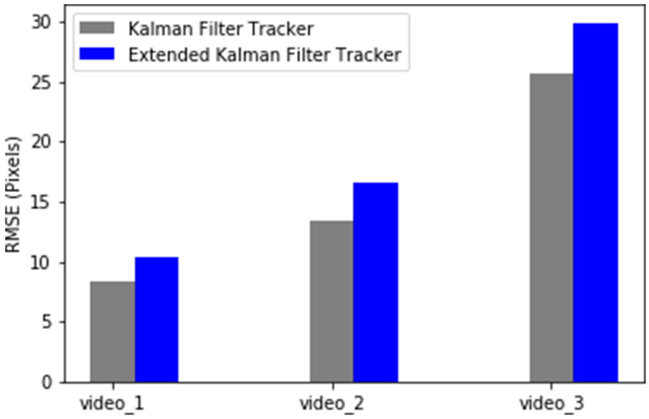

**FIGURE 8** | RMSE Comparison between KF Tracker and EKF Tracker.

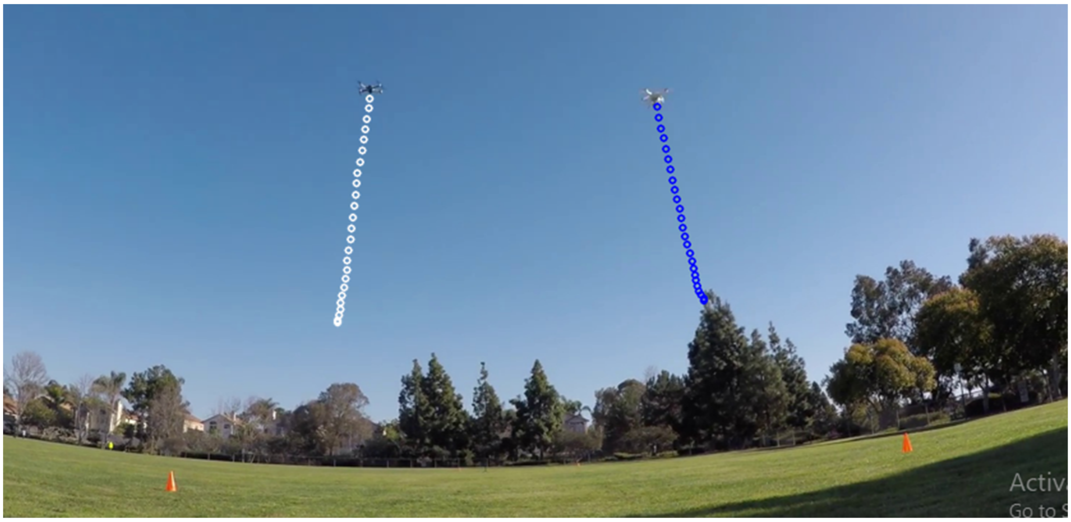

**FIGURE 9** | Tracker prediction & trajectory estimation results using Kalman filter.

results show that the bounding box representing the UAV location in a particular frame is refined using the KF and EKF algorithms, and the dropouts due to detection failure are compensated. In addition, it is found that the KF algorithm introduces better performance with linear systems, in this work, contrary to the EKF algorithm that is recommended in non-linear systems. We believe that the proposed platform helps improve the UAVs' visual detection. In addition, the efficiency of the system can be improved by investigating the possibility of integrating additional trackers with KF and EKF.

Future studies on UAV tracking and detection may focus on a number of important issues to improve the functionality and flexibility of the system. First, to increase tracking accuracy in challenging environments like urban settings with high occlusion rates, more sophisticated deep learning models need to be integrated with conventional filtering approaches. To reduce the effect of errors caused by occlusion, blur, or other obstacles, research efforts will be focused on enhancing the combination of Kalman filtering and its variations with real-time object identification algorithms like YOLO.

Making computer vision algorithms more resilient and flexible is a significant area of future studies. This involves looking at methods that can better manage dynamic environmental factors, enabling improved UAV tracking and detection in a range of lighting and weather conditions. It will be essential to move toward more robust algorithms that can adjust on their own as the look and surroundings of the UAV change. Furthermore, the scalability of UAV tracking systems will be further investigated in future studies. This entails creating techniques that can effectively manage large-scale situations, including tracking several UAVs concurrently over vast geographic regions. Methods for handling non-target objects, controlling tracking between cameras, and strengthening tracking resilience in challenging situations are within the future work.

## Author Contributions

N.A. conducted the design and implementation. R.A. and T.I. supervised the work. N.A., R.A. and T.I. drafted the work. R.A. and T.I. assisted in the simulation of the work. R.A., T.I. and H.M. verified the work. H.M. revised and reviewed the manuscript. All authors provided critical feedback and helped shape the research, analysis, and manuscript.

## Ethics Statement

This work did not involve human participants, animal subjects, or any sensitive data, and therefore no ethical approval was required.

## Data Availability Statement

The datasets generated and/or analysed during the current study are available in this link: https://github.com/HaithamHmahmoud/UAV-CDT.

## Endnotes

[1] https://github.com/HaithamHmahmoud/UAV-CDT.

## References

1. Y. Ko, J. Kim, D. G. Duguma, P. V. Astillo, I. You, and G. Pau, "Drone Secure Communication Protocol for Future Sensitive Applications in Military Zone," *Sensors* 21, no. 6 (2021): 1–25.

2. S. J. Kim, Y. Jeong, S. Park, K. Ryu, and G. Oh, "A Survey of Drone Use for Entertainment and Avr Augmented and Virtual Reality," in *Augmented Reality and Virtual Reality* (Springer, 2018), 339–352.

3. B. Taha and A. Shoufan, "Machine Learning-Based Drone Detection and Classification: State-Of-The-Art in Research," *IEEE Access* 7 (2019): 138669–138682.

4. H. Mahmoud, I. F. Kurniawan, A. Aneiba, and A. T. Asyhari, "Enhancing Detection of Remotely-Sensed Floating Objects via Data Augmentation for Maritime Sar," *Journal of the Indian Society of Remote Sensing* 52 (2024): 1–11.

5. X. Wu, W. Li, D. Hong, R. Tao, and Q. Du, "Deep Learning for Uav-Based Object Detection and Tracking," *A Survey* 2021 (2021): 1–24.

6. 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* 16, no. 4 (2018): 1522–1530.

7. S. Jeon, J.-W. Shin, Y.-J. Lee, W.-H. Kim, Y. Kwon, and H.-Y. Yang, "Empirical Study of Drone Sound Detection in Real-Life Environment With Deep Neural Networks," in *25th European Signal Processing Conference (EUSIPCO)* (IEEE, 2017), 1858–1862.

8. W. D. Scheller, *Detecting Drones Using Machine Learning*Ph.D. thesis, (Iowa State University, 2017).

9. B. K. Isaac-Medina, M. Poyser, D. Organisciak, C. G. Willcocks, T. P. Breckon, and H. P. Shum, "Unmanned Aerial Vehicle Visual Detection and Tracking Using Deep Neural Networks: A Performance Benchmark," in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (IEEE, 2021), 1223–1232.

10. R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2014), 580–587.

11. S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-Cnn: Towards Real-Time Object Detection With Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, no. 6 (2016): 1137–1149.

12. J. Dai, Y. Li, K. He, and J. Sun, "R-Fcn: Object Detection via Region-Based Fully Convolutional Networks," in *Advances in Neural Information Processing Systems* (Springer, 2016), 379–387.

13. A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal Speed and Accuracy of Object Detection," 2020.arXiv:2004.10934.

14. W. Liu, D. Anguelov, D. Erhan, et al., "Ssd: Single Shot Multibox Detector," in *European Conference on Computer Vision (ECCV)* (Springer, 2016), 21–37.

15. A. Jadhav, P. Mukherjee, V. Kaushik, and B. Lall, "Aerial Multi-Object Tracking by Detection Using Deep Association Networks," in *National Conference on Communications (NCC)* (IEEE, 2020), 1–6.

16. W. Li, J. Mu, and G. Liu, "Multiple Object Tracking With Motion and Appearance Cues," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops* (IEEE, 2019), 1–9.

17. S. Chopra, R. Hadsell, and Y. LeCun, "Learning a Similarity Metric Discriminatively, With Application to Face Verification," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1 (IEEE, 2005), 539–546.

18. J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-Speed Tracking With Kernelized Correlation Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, no. 3 (2014): 583–596.

19. G. Bishop and G. Welch, "An Introduction to the Kalman Filter," 1995.

20. Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-Learning-Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, no. 7 (2011): 1409–1422.

21. S. Hare, S. Golodetz, A. Saffari, et al., "Struck: Structured Output Tracking With Kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, no. 10 (2015): 2096–2109.

22. L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-Convolutional Siamese Networks for Object Tracking," in *European Conference on Computer Vision* (Springer, 2016), 850–865.

23. Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object Detection With Deep Learning: A Review," *IEEE Transactions on Neural Networks and Learning Systems* 30, no. 11 (2019): 3212–3232.

24. P. Fieguth and D. Terzopoulos, "Color-Based Tracking of Heads and Other Mobile Objects at Video Frame Rates," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (IEEE, 1997), 21–27.

25. J. Jiao, X. Wang, Z. Deng, J. Cao, and W. Tang, "A Fast Template Matching Algorithm Based on Principal Orientation Difference," *International Journal of Advanced Robotic Systems* 15, no. 3 (2018): 1–9.

26. F. Svanström, C. Englund, and F. Alonso-Fernandez, "Real-Time Drone Detection and Tracking With Visible, Thermal and Acoustic Sensors," in *25th International Conference on Pattern Recognition (ICPR)* (IEEE, 2021), 7265–7272.

27. F. Svanström, "Drone Detection and Classification using Machine Learning and Sensor Fusion," Master's thesis, Kristian IV:s väg 3, 301 18 Halmstad, Sweden 2020.

28. Q. Yu, T. B. Dinh, and G. Medioni, "Online Tracking and Reacquisition Using Co-Trained Generative and Discriminative Trackers," in *10th European Conference on Computer Vision(ECCV)* (Springer, 2008), 678–691.

29. A. Bordes, S. Ertekin, J. Weston, L. Botton, and N. Cristianini, "Fast Kernel Classifiers With Online and Active Learning," *Journal of Machine Learning Research* 6, no. 9 (MIT Press, 2005).

30. R. Gan, B. I. Ahmad, and S. J. Godsill, "Lévy State-Space Models for Tracking and Intent Prediction of Highly Maneuverable Objects," *IEEE Transactions on Aerospace and Electronic Systems* 57, no. 4 (2021): 1–17.

31. M. Jahangir, B. I. Ahmad, and C. J. Baker, "Robust Drone Classification Using Two-Stage Decision Trees and Results From Sesar Safir Trials," in *IEEE International Radar Conference (RADAR)* (IEEE, 2020), 636–641.

32. J. Park, D. H. Kim, Y. S. Shin, and S.-h. Lee, "A Comparison of Convolutional Object Detectors for Real-Time Drone Tracking Using a Ptz Camera," in *17th International Conference on Control, Automation and Systems (ICCAS)* (IEEE, 2017), 696–699.

33. J. Sim, M. Jahangir, F. Fioranelli, C. J. Baker, and H. Dale, "Effective Ground-Truthing of Supervised Machine Learning for Drone Classification," in *International Radar Conference (RADAR)* (IEEE, 2019), 1–5.

34. L. Grivault, A. El Fallah-Seghrouchni, R. Girard-Claudon, Multi-Agent System to Design Next Generation of Airborne Platform, *Intelligent Distributed Computing XI* (Springer, 2018), 103–113.

35. M. Pilté, S. Bonnabel, and F. Barbaresco, "Maneuver Detector for Active Tracking Update Rate Adaptation," in *19th International Radar Symposium (IRS)* (IEEE, 2018), 1–10.

36. P. Marion, J. Sami, B. Silvère, B. Frédéric, F. Marc, and H. Nicolas, "Invariant Extended Kalman Filter Applied to Tracking for Air Traffic Control," in *International Radar Conference (RADAR)* (IEEE, 2019), 1–6.

37. C. Ju, Z. Wang, C. Long, X. Zhang, and D. E. Chang, "Interaction-Aware Kalman Neural Networks for Trajectory Prediction," in *IEEE Intelligent Vehicles Symposium (IV)* (IEEE, 2020), 1793–1800.

38. A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal Speed and Accuracy of Object Detection," arXiv:2004.10934.

39. S. Jouaber, S. Bonnabel, S. Velasco-Forero, and M. Pilte, "Nnakf: A Neural Network Adapted Kalman Filter for Target Tracking," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2021), 4075–4079.

40. J. Redmon and A. Farhadi, "Yolov3: An Incremental Improvement," 2018.arXiv:1804.02767.

41. J. Redmon, "Darknet: Open Source Neural Networks in C," 2013–2016, http://pjreddie.com/darknet/.

42. C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "Cspnet: A New Backbone That Can Enhance Learning Capability of Cnn," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (IEEE, 2020), 390–391.

43. K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, no. 9 (2015): 1904–1916.

44. S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path Aggregation Network for Instance Segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2018), 8759–8768.

45. T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2017), 2117–2125.

46. G. Jocher, "Yolov5 Algorithm," https://github.com/ultralytics/yolov5.

47. Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding Yolo Series in 2021," 2021.arXiv:2107.08430.

48. T. Noble, "Dji f450 Naza v2Url," https://www.youtube.com/watch?v=jdvLV6dYyRU.

49. D. Dunnil, "Hubsan x4 h501s Advanced - Full Review - [Unbox, Inspection, setup, Flight Test, Pros & Cons," https://www.youtube.com/watch?v=ERUWJYsU9c8.

50. D. Tzutalin, "Tzutalin/Labelimg," *Github* (2015). https://github.com/HumanSignal/labelImg/tree/master.