STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 2026, pp 1–20.
Published online in International Academic Press (www.IAPress.org)

An Image-based Obstacle Detection Approach for Drones Using Deep Learning

Afnan Abdelfattah [1,2,*], Dina ElMenshawy [1], Hatem ElKadi [1], Ahmed Abdelhafeez [3,4]

[1] *Information Systems Department, Faculty of Computers and Artificial Intelligence, Cairo University, Egypt*
[2] *Information Systems Department, Faculty of Information Systems and Computer Science, October 6 University, Giza, Egypt*
[3] *Faculty of Computer and Information Technology, Innovation University, Cairo, Egypt*
[4] *Applied Science Research Center, Applied Science Private University, Amman, Jordan*

Abstract Drones have been used in multiple sectors, particularly in the civilian and military sectors, including agriculture, healthcare, security, and surveillance. This expansion has raised additional concerns about the structural safety of drones due to their prohibitive cost. As a result, developing onboard obstacle detection systems for drones has become an important research area. Most existing systems face significant challenges in detecting and classifying obstacles, particularly in dynamic environments which include various objects such as birds, trees, power pylons, and even other drones. Overcoming these challenges requires developing robust and effective models that are trained in a variety of multi-obstacle images with diverse backgrounds. This research proposes an obstacle detection approach for drones based on deep learning. The proposed approach was applied in complex, low-visibility scenarios using the You Only Look Once (YOLOv11s) model and ByteTrack tracking algorithm. In addition, we propose a newly curated multi-source dataset of images containing multiple objects namely drones, birds, trees, and power pylons. The images were collected from different environments to ensure diversity. Using Roboflow and other datasets, the dataset was systematically processed, optimized, and developed. YOLOv11s achieved an overall precision of 94.2%, recall of 89.6%, mAP@0.5 of 93.7%, and mAP@0.5:0.95 of 73.0% on the evaluation set. In contrast to prior research where only one or two classes have been considered, this paper considers the performance of a unified framework for four classes of obstacles that are relevant for drones in complicated visual scenarios. Hence, the results must be regarded as demonstrating superior performance in the presented multi-class scenario and not in general terms over all past approaches. In addition, the YOLOv11s detector has a measured throughput rate of 65.7 frames per second at an input resolution of 640 x 640 pixels, which corresponds to an average processing time of approximately 15.22 milliseconds per image.

Keywords UAV, obstacle detection, YOLOv11, ByteTrack, power pylons, multi-object tracking

DOI: 10.19139/soic-2310-5070-4440

1. Introduction

Unmanned Aerial Vehicles (UAVs) have seen much wider use in civilian and military applications. UAVs are used in agriculture, infrastructure monitoring, surveillance, environmental monitoring, and deliveries. This increase in the application of UAVs has created a higher demand for obstacle detection systems, especially in low altitude areas where there can be a threat to safety from obstacles such as birds, other UAVs, trees, and infrastructure [2], [6].

Detection of Small UAVs in Varying Target Scale, Background, Illumination, and View Point has been shown to be very challenging by Recent Anti-UAV Benchmarks & Challenges [3].

*Correspondence to: Afnan Abdelfattah (Email: afnanabdelfattah17@gmail.com). Information Systems Department, Faculty of Information Systems and Computer Science, October 6 University, Giza, Egypt.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

The introduction of drones in areas such as agriculture, security, surveillance, delivery, etc. has led to several challenges, including obstacles that may cause the drones to malfunction. Unfortunately, there are several issues with the currently used obstacle-detection algorithms. For example, when capturing photos or videos from a distance, it can be difficult to distinguish between drones and birds. Furthermore, these algorithms often experience poor detection performance due to varying lighting and weather conditions, as well as blurry backgrounds. These issues have not been adequately addressed in the literature. Some literature work applied various versions of the You Only Look Once (YOLO) model but had limitations in terms of performance and accuracy. To address the problem of background variations, new and improved drone detection algorithms need to be developed.

In this paper, an enhanced image-based obstacle detection approach is proposed using YOLOv11s and a tracking tool to identify moving objects, such as birds and drones, in low-visibility environments, including those with changing lighting or birds entering trees.

The main contributions of this paper are as follows:

1. Construction of the dataset: The UAV obstacle dataset comprising four classes of objects (bird, drone, tree, and power pylon) is composed from three open-source datasets. The experimental dataset consists of various background, scale, view angle, lighting, and partial occlusion samples of the above four classes of objects.

2. Evaluation of two YOLOv11 variants: Two light-weight versions of YOLOv11 were compared empirically to study the trade-off between performance and inference speed. It turned out that YOLOv11s demonstrated better performance, whereas YOLOv11n demonstrated substantially better inference speed.

3. Baseline for Tiny-YOLOv7 and YOLOv8: Tiny-YOLOv7 and YOLOv8 models were trained and evaluated on the same four-class dataset, with the same evaluation metrics. This gives more accurate comparison than numerical comparison of papers that use different datasets and evaluation methods.

4. Qualitative demonstration of ByteTrack: ByteTrack algorithm was integrated with YOLOv11 and used to track moving obstacles through video frame sequences. Qualitative experiment was performed with a video sequence consisting of 170 frames. Due to the absence of ground truth trajectories, the tracking task can be evaluated only qualitatively.

The research is divided into the following sections: Section Two provides background on some of the models used to detect obstacles in general; Section Three mentions previous work in this field; Section Four details the model used and the data sets used in this study; Section Five presents the experimental results of the study, analyzes them, and measures their effectiveness; and finally, Section Six concludes this research.

2. Background

Due to the recent widespread use of drones, several technologies were used to detect obstacles encountered by drones in various fields. One of the most famous technologies was radar [4]. Drone-detection systems may rely on radar, acoustic sensing, radio-frequency analysis, and vision-based techniques [4], [6]. but these proved to be inefficient and expensive. More recently, machine learning such as support vector machine and artificial neural networks have been used to detect obstacles encountered by drones from images, leading to satisfactory results. This has demonstrated the effectiveness of YOLO in detecting obstacles at high speed, surpassing Region-based Convolutional Neural Networks (R-CNN), and Single Shot MultiBox Detector (SSD) [6]. Therefore, YOLO has been relied upon heavily in this field for detecting objects quickly and accurately [8][13].

Obstacle detection systems for drones employ a variety of techniques. This study focuses on studies relevant to our proposed approach. Obstacle detection is divided into two main approaches: a two-stage detection and a single-stage detection [9]. There are many examples of two-stage detection techniques, such as R-CNN [10], and examples of single-stage detection techniques, are YOLO [11] and SSD [12]. The general workings of object detection techniques were explained in this research [13]. Based on this research, it was discovered that single-stage algorithms are faster than two-stage algorithms and are characterized by working in real time, which is required

in our field of research. YOLO is characterized by the ease and speed of training, and its efficiency compared to others. This method divides the image into several sections and then determines the probability of each category, placing bounding boxes around any obstacle in the images, and then uses a single convolutional neural network for prediction.

A single frame-level approach for drone obstacle avoidance is not enough since the same bird or drone could be detected as separate objects in two adjacent frames due to motion speed, motion blur, light changes, or even partial occlusion by trees. For this reason, the suggested system employs YOLOv11s to produce bounding boxes and class confidence levels before using ByteTrack for matching between frames. ByteTrack would be a good choice because it would not ignore all low confidence detections but use both high and low confidence detections while matching the objects[1].

3. Related Work

This section focuses on reviewing the related work that explored obstacle detection approaches facing drones.

Shi and Li. (2020) [5], explored a vision-based system that detects UAVs based on the YOLOv4 algorithm and its application in real-time anti-UAV situations. This system revealed the possibility of deep learning object detection in real-time visual surveillance.

Abu-Jassar et al. (2025) [14] developed and experimentally evaluated a computer-vision system for a small-sized mobile humanoid robot operating in a smart environment. Their design replaced a computationally heavier VGG-16-based feature extractor with a lightweight Tiny-YOLO configuration to reduce the number of calculations while preserving object-recognition capability. Static and dynamic experiments demonstrated the relevance of lightweight YOLO-based perception for resource-constrained mobile platforms, which is closely related to the real-time deployment requirements of UAV obstacle-detection systems.

Zhu et al. (2023) [15], proposed two novel UAV platforms and a particular light source (silicon-based golden LED) which were used to test a real-time object detection method (YOLO-Drone). The researchers achieved a maximum mean average precision (mAP) of 87.71%, which outperformed the YOLO series when exposed to standard light sources.

Hanif et al. (2025) [16] proposed a real-time ambulance-detection and priority-passage framework based on transfer-learned YOLOv8 and NorFair tracking. The detector achieved an mAP@0.5 of 0.860, while the tracking component maintained temporal continuity across video frames. The study also identified reduced reliability under night-time and fog conditions, reinforcing the importance of low-visibility samples and temporal association when object-detection systems are deployed in dynamic outdoor environments.

K. Sricharan et al. (2023) [17], proposed a YOLOv5-based system for real-time drone detection. After manually annotating 6,820 photos of various UAVs, the researchers trained YOLOv5 on Google Colab for one hundred epochs. Their proposed model achieved an F1-score of 0.93 and an approximate mAP of 92.8%.

Shandilya et al. (2023) [18], applied the YOLO model on a dataset of 20,925 images, including 8,451 bird images and 12,474 drone images. Working on the Roboflow data had some problems namely, data imbalance, heterogeneity, and lack of diversity of categories within the dataset.

Burchan Aydin et al. (2023) [19], presented a YOLOv5-based drone detection system that uses a bespoke two-class dataset of 2,395 photos to distinguish drones from birds in real time. After improving these photos to more than 5,700 samples, the authors used transfer learning with the pretrained weights from YOLOv5. With an mAP of 90.40%, the final model outperformed an earlier YOLOv4 solution using the same dataset by 21.57%.

Suvaris et al. (2025) [20] introduced PRISM, a real-time multi-scale object-detection framework that combines YOLOv8 feature extraction with Transformer-based contextual modelling. The architecture uses a Context-Aware Feed-Forward Network and Cross-Scale Attention Skip Connections to improve the representation of small, distant, and partially occluded objects in complex surveillance scenes. The reported evaluation showed 96% classification accuracy, illustrating the value of combining local multi-scale features with broader contextual information for real-time visual detection.

Rehman et al. (2026) [21] proposed Group Sparse YOLOv8 for real-time differentiation between real animals and toy replicas. The study focused on visual variations caused by viewpoint, occlusion, illumination, and texture, and integrated group-sparsity regularization, frame selection, and parallel processing to reduce computational overhead while maintaining detection performance. These design considerations are relevant to UAV obstacle detection, where birds may appear at different scales and under rapidly changing visual conditions.

Farhaoui et al. (2025) [22] developed a fine-tuned object-detection framework for mask recognition in IoT environments using Faster R-CNN with a ResNet-50 backbone. The study used 2,000 training images and 400 test images and incorporated model-compression and quantization techniques to support resource-efficient edge deployment. Its reported robustness to illumination changes, occlusion, and object-orientation variation provides useful evidence for the design of real-time visual detectors operating under constrained computational and environmental conditions.

Zamri et al. (2024) [23, improved the detection head by adding a tiny detection head and incorporated the attention module into the neck. The goal of the study was to enhance the YOLOv8n model and boost its capacity to recognize small objects. Although the YOLOv8n + ResCBAM + high-resolution detection head produced the intended outcomes, this added complexity to the model.

Muhammad K. Kabir et al. (2024) [24], developed a YOLOv7-based system for detecting drones in video streams. The researchers compiled a massive dataset of over 57,000 drone images. They reported a recall rate of 96.56% and an accuracy of 95.09%. They also compared their approach with YOLOv8, YOLO-NAS, and Faster R-CNN.

Laroca et al. (2025)[25], used the YOLOv11 model to detect drones. This model had difficulties in detecting small drones; this is because it was trained only on drones, not on similar objects like birds.

Due to the features of YOLO, such as compatibility and faster training times, numerous versions of YOLO have been proposed, and these versions have been enhanced based on various usage scenarios, including drones.

Table 1. Contextual comparison of selected published studies; reported values are not directly comparable because the datasets and evaluation protocols differ.

Reference	Method	Dataset	Classes	Reported Metric(s)
[20]	PRISM: Hybrid Transformer-YOLOv8	Surveillance dataset	Five object categories	Accuracy = 96%
[26]	YOLOv5	1,359 images	Drone	mAP@0.5 = 94.1%
[17]	YOLOv5	6,820 images	Drone	mAP@0.5 = 92.8%
[27]	YOLOv9	10,000 images	Drone	mAP@0.5 = 95.7%
[25]	YOLOv11	77 videos	Drone	mAP@0.5= 73.9%
[19]	YOLOv5	2,395 images expanded to 5,700	Drone, Bird	mAP@0.5 = 90.4%
[24]	YOLOv7	More than 57,000 images	Drone	Recall = 96.56%; Precision = 95.09%
[28]	YOLOv3 + DeepSORT	1,300 images	Drone, bird-shaped drone	mAP@0.5 = 96.98%

As shown in the previous table several studies relevant to our research area were compiled to compare the proposed approach with the most recent deep learning-based object detection methods for drones. Several factors were compared, including the algorithm used, the type and size of data used, the number of classes, as detailed in Table 1, and finally, the final model efficiency score. These studies were selected because of their relevance to our research and their importance in clarifying the proposed approach and its improvements over previous work. The algorithms used to detect obstacles facing drones are diverse. Some of these algorithms yielded high results and good efficiency.

It needs to be noted that the above-mentioned works used in Table 1 have been carried out with a number of different datasets, definitions of classes, imaging data, configurations of the model, and evaluation methodology. Thus, their numerical results cannot be seen as a full-fledged comparison of one another. Indeed, there have been several studies, which have achieved higher mAP but examined only one or two classes, whereas our work examines

four obstacle classes with the use of a single detection system. The idea behind Table 1 is to place the proposed study into the context of the previous research.

4. Proposed Approach

This section will present the proposed approach including the process of preparing the dataset, the model structure and the evaluation measures used in this research. Figure 1 provides an overview of the proposed approach.

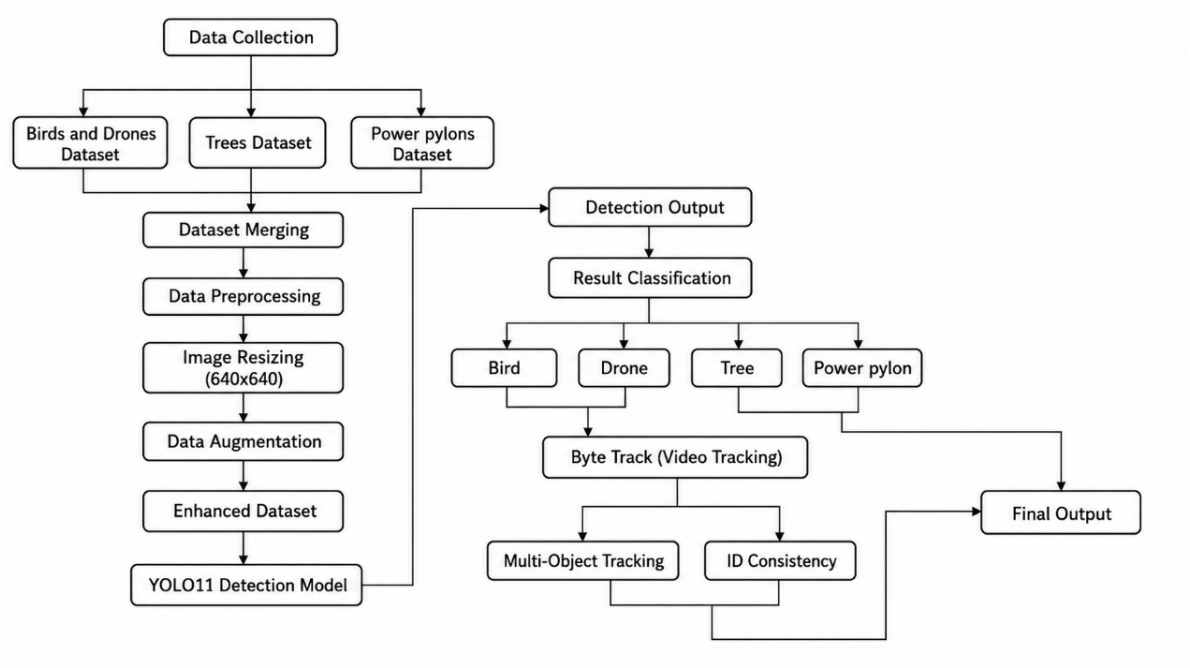

Figure 1. Proposed Approach of Image-based Obstacle Detection

To maintain a clear structure, the suggested methodology will be presented in three consecutive stages. Stage one is related to the creation and preprocessing of the dataset along with its augmentation. Stage two will involve object detection based on YOLOv11 and model training. Stage three will include video tracking based on ByteTrack of dynamic objects.

4.1. Dataset Description

Detection of obstacles facing drones requires models' training using large and diverse datasets to increase the efficiency of obstacles' detection. In addition, to avoid losses resulting from drone collisions with objects while using drones to capture images, such as for plant pest detection. Therefore, a large and varied dataset was created, coming from multiple datasets. The dataset was constructed through optimizing the image variety, increasing the dataset size, and varying image brightness to achieve the best results while maintaining the model's obstacle detection speed. This also included tracking moving objects, such as birds, to locate them in dark environments like trees or in adverse weather conditions where rapid detection is difficult. Three datasets were collected containing trees, birds, power pylons, and drones. Unclassified images were extracted from various sources, including Roboflow and Kaggle, as well as other scientific research data in the same field. This contributed to increasing data diversity and covering all aspects of the obstacles a drone might encounter. Figure 2 refers to a set of diagrams that illustrate label analysis in general and are divided into four sections. The first section (a) shows the rate of class division in the dataset used, which consists of birds, drones, Power pylons, and trees. As for section (b), it

shows the extent of object distribution within the image, where it was observed that most of the objects are close to the center of the image. It also shows the extent of the difference in object sizes based on the varied sizes of the boxes, which are also called bounding boxes. As for section (c), it shows the concentration of objects' locations and shows that they are mostly in the middle of the images; these are called heatmap objects. As for the last section (d), it indicates the extent of the difference in object sizes in the images, which shows that they tend towards small and medium sizes slightly.

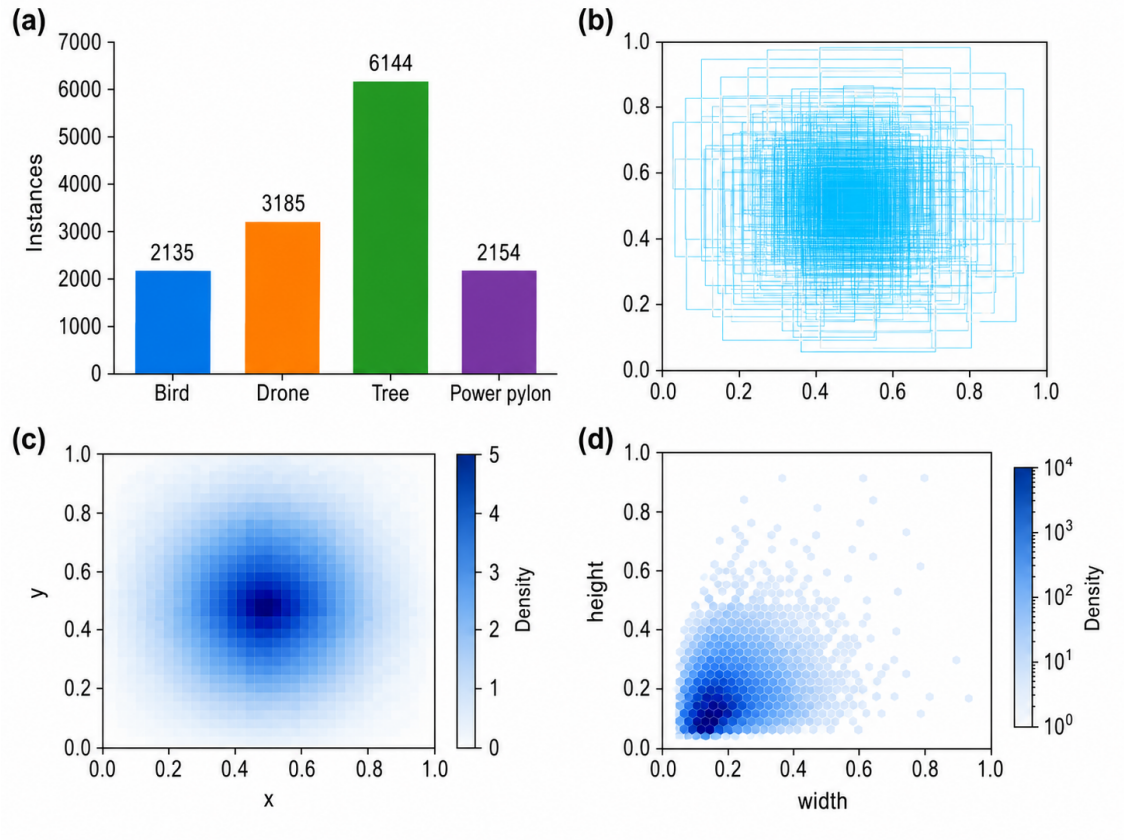

Figure 2. Distribution and spatial characteristics of the final obstacle annotations.

The first set of data was created using three publicly available datasets. Data pertaining to bird and drone images was collected using the Drone-Bird-Detection database available at Roboflow Universe. Data for tree images was taken from the Urban Tree Detection Dataset provided by Kaggle while that of power pylon images came from the Transmission System dataset at Roboflow Universe, Only annotations belonging to the original tower class were retained and mapped to the final power pylon class. Annotations belonging to the insulator class were excluded these are detailed in Table 2 below.

The source datasets contained different predefined split structures; therefore, class-wise split ratios were not identical

Table 3 Shows that the dataset consists of four categories: birds, drones, trees, and power pylons. The initial dataset comprised 10836 images. After Validation, cleaning, resizing, and enlargement, the training set was expanded to include 20,000 images. A clear difference between the first and second datasets is now evident.

Figure 3 shows the number of source images which were retained before augmentation with the total number of images and annotated instances after preprocessing and augmentation. The retained dataset consisted of 2,928 images of birds, 4,284 images of drones, 2,716 images of trees, and 908 images of power pylons. After preprocessing and augmentation, the total number of images was 5,874 images of birds, 8,574 images of drones,

Table 2. Dataset distribution before augmentation.

Class	Final images	Train images	Validation images	Test images	Source link
Bird	2928	2046	575	307	https://universe.roboflow.com/myworkspace-0p4nk/drone-bird-detec
Drone	4284	2972	904	408	https://universe.roboflow.com/power-tower/transimission-system
Power pylon	908	665	153	90	
Tree	2716	2376	227	113	https://www.kaggle.com/datasets/mcii34/urbantree-subset-public/data
Total	10836	8059	1859	918	

Table 3. Dataset distribution after augmentation.

Class	Final images	Final instances	Train images	Train instances	Validation images	Validation instances	Test images	Test instances
Bird	5,874	5,874	4137	4137	1116	1116	621	621
Drone	8,574	8,998	5938	6230	1795	1871	841	897
Tree	4,487	5,634	3901	4242	424	1179	162	213
Power pylon	1,065	9,143	724	7603	322	1191	19	349
Total	20,000	29,649	14700	22212	3657	5357	1643	2080

4,487 images of trees, and 1,065 images of power pylons. The corresponding number of annotated instances was 5,874, 8,998, 5,634, and 9,143, respectively. The number of annotated instances could be higher than that of images since an image can contain more than one instance of the same class.

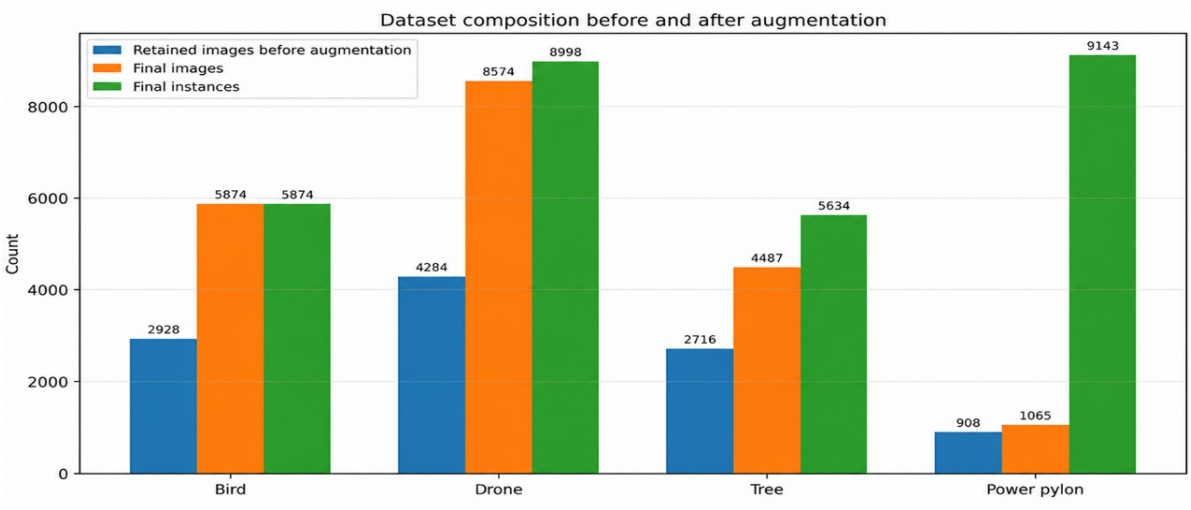

Figure 3. Dataset composition before and after augmentation.

4.2. Data Preprocessing and Annotation

Initially, Images that have been manually reviewed images used in other scientific research related to the field, images used to detect objects such as birds and other drones were within the field of view. In addition, we examined sets used in similar research projects and identified several factors that contributed to the inaccurate and ineffective

detection of obstacles. This included image quality: a model like YOLO requires appropriate resolution to provide accurate, fast, and efficient object detection. Other factors included lighting and its variations, climate changes, and the presence of obscure or unclear environments, such as birds in trees. Insufficient image variety also hindered the model's ability to train effectively and led to learning cessation. Additionally, some images lacked annotations, significantly affecting the model's performance. Finally, the limited number of classes and the insufficient number of images per class also affect the detection accuracy. Therefore, all images have been resized to a fixed resolution compatible with the YOLOv11 architecture. The bounding boxes were manually created and checked to ensure the accurate identification of the location of each object in the dataset was created, which was compiled from several datasets. The number of classes and the number of images in each class have been increased to provide better training for the model and increase its efficiency and accuracy in object detection while maintaining its detection speed. Incorrect and blurry bounding boxes were removed to improve the quality of the dataset.

4.3. Data Augmentation

Augmentation was done to make the models more robust against variations in scale, viewpoint, illumination, weather, and partial occlusions.

Geometric and photometric augmentations were performed with the help of albumentations version 2.0.8 [8] while maintaining bounding box consistency. The details of the actual parameters used for augmentation are described in Table 4.

The augmentation process involves image rescaling, cropping and scaling, image horizontal flipping, brightness and contrast adjustments, hue and saturation adjustments, and image validation to maintain the consistency of the transformed bounding boxes. These augmentations were chosen due to the UAV image characteristics that are related to altitude, camera angle, motion blurring, shadows, clouds, and scale.

Table 4. Data preprocessing and augmentation configuration.

Augmentation/process	Purpose	Final setting
Resize to 640 x 640	Standardizes YOLOv11 input size and batch processing.	640 x 640
Horizontal flip	Simulates opposite flight/view directions.	Probability = 0.50
Random crop/scale	Improves robustness to object size and position changes.	Scale range = 0.50–1.50; probability = 0.50
Brightness/contrast	Simulates low-light, glare, and cloudy scenes.	Variation range = ±20%; probability = 0.50
HSV/saturation	Improves robustness to color and weather variation.	Hue = ±1.5%; saturation = ±30%; probability = 0.50
Annotation validation	Removes invalid or unclear bounding boxes.	Manual + export validation

4.4. Experimental Setup

To achieve optimal results and train the model to its full potential without increasing model complexity or detection time, several experiments were conducted with different components to ensure optimal results while maintaining a diverse range of categories. Initially, data consisting of only two categories, birds and drones, was used with the YOLOv11s model. Large dataset sizes were avoided due to their incompatibility with drone capabilities. This resulted in an unsatisfactory precision of only 70%. This was due to insufficient image enhancement and model training. Subsequently, two additional categories were added: power lines and trees. By combining these datasets, an precision rate of 68% was achieved, as shown in Table 5. This increase in the number of categories was achieved without improving image quality, and the model stopped learning after only 300 epochs. Afterward, image quality was improved, as previously described in the section on data preprocessing, annotation, and the integration of multiple datasets to enhance the model's learning capacity and increase the number of categories.

The squeezing process involved changing the batch size and increasing the total data volume, as described in the Data Scaling section, until the total data volume increased from 10,836 images to 20,000 images. This resulted in an efficiency rate increase to 94.2%, and an accuracy increase in the drone category to 99.2% and in the bird category to 98.2%. Figure 4 illustrates a model for training the collected data batches and optimizing them to increase model efficiency and achieve the results. It also shows the boxes surrounding potential obstacles in the image to improve the annotation process.

Table 5. Summary of the sequential dataset-development experiments.

Experiment	Classes	Images	Training images	Validation images	Test images	Overall precision
Experiment 1	Bird, drone	7,212	5,018	1,479	715	70.0%
Experiment 2	Bird, drone, tree, and power pylon	10,836	8,059	1,859	918	68.0%
Experiment 3	Bird, drone, tree, power pylon	20,000	14,700	3,657	1,643	94.2%

4.5. Model Architecture

Selection of YOLOv11 as a detector for objects has been done considering the one-stage detection network that gives a proper trade-off between accuracy and speed of detection. This trade-off becomes significant in case of UAV images where birds and drones may cover small areas on the image whereas trees and power lines show variations in size, form, texture, and background. The input image is processed in a single pass by the model and gives class predictions, confidences, and object location on the image.

The multi-scale feature representation of YOLOv11 enables the detection of objects with varying sizes and observation distances. It is critical for aerial images since the size of the obstacles varies depending on the height of the UAV, angle of the camera, distance of the object, and surrounding environment. In this research work, all the images have been scaled to 640×640 pixels.

ByteTrack was incorporated following the detection process in YOLOv11 as an auxiliary temporal association module. It uses the output bounding boxes and confidences from YOLOv11 for association between detections in successive video frames. ByteTrack does not change the class predictions and bounding box regression outputs from the detector. In this research work, the tracking module was shown qualitatively as no ground truth object trajectories existed. Thus, the precision, recall, F1-score, mAP, and throughput values presented quantitatively in this paper belong to the object detection models.

4.5.1. YOLOv11 Model Variants Since eventual UAV usage necessitates a trade-off between computation cost, latency, and the accuracy of detection, two light-weighted YOLOv11 models—namely, YOLOv11n and YOLOv11s—were tested. YOLOv11n was chosen as the one with a higher throughput while YOLOv11s as the one with a better accuracy. YOLOv11s was demonstrated to have superior accuracy in general and mAP@0.5 while YOLOv11n had a higher throughput. Consequently, YOLOv11s was chosen as the primary model for the class-wise testing while YOLOv11n was kept as an alternative one due to its lightweight design.

The quantitative accuracy–throughput comparison is reported in Table 6.

Table 6. Comparison between the YOLOv11 Nano and Small variants.

Variant	**Input**	**Precision**	**Recall**	**mAP@0.5**	**mAP@0.5:0.95**	**Detector FPS**
YOLOv11n	640×640	92.2%	89.7%	92.2%	72.9%	183.9
YOLOv11s	640×640	94.2%	89.6%	93.7%	73.0%	65.7

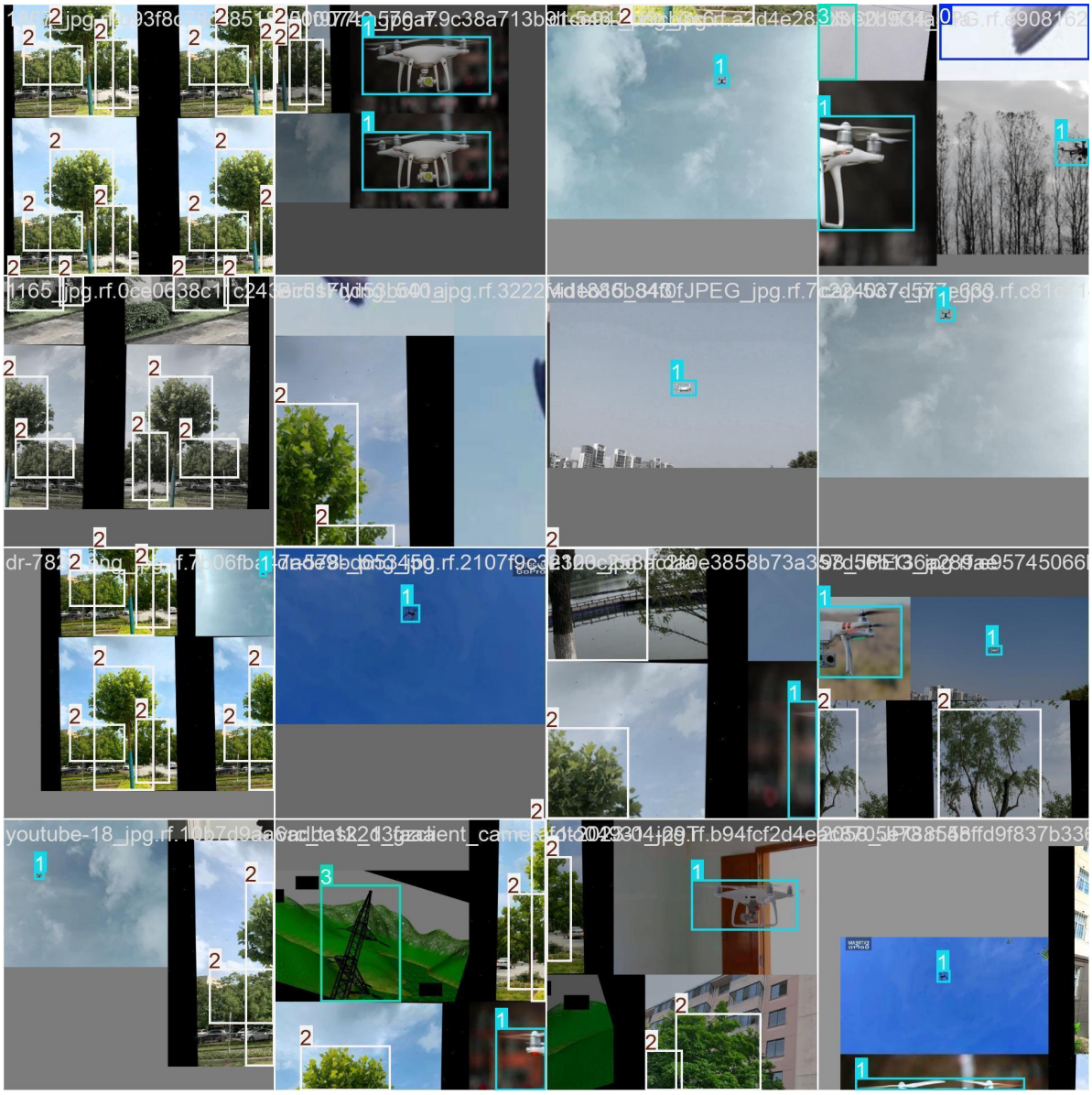

Figure 4. Representative training batches after resizing and augmentation.

Figure 5 shows that measurements using YOLOv11n showed a data transfer rate of 183.9 frames per second, while YOLOv11s achieved a data transfer rate of 65.7 frames per second. Higher accuracy and a higher average accuracy at the 0.5 threshold (mAP@0.5) were obtained using YOLOv11s. In contrast, the data transfer rate was significantly higher with YOLOv11n.

4.5.2. Dataset Scaling Increasing the size and variety of data is crucial in our research and in the field of aeronautics in general. This is due to significant changes in images, which have several causes. These include variations in lighting levels caused by climate change, affecting object detection in low-light conditions. Altitude and angle variations also impact detection accuracy, necessitating data diversity for the model to learn different patterns. Furthermore, size variation is a factor; the size of birds differs depending on the distance of the aircraft.

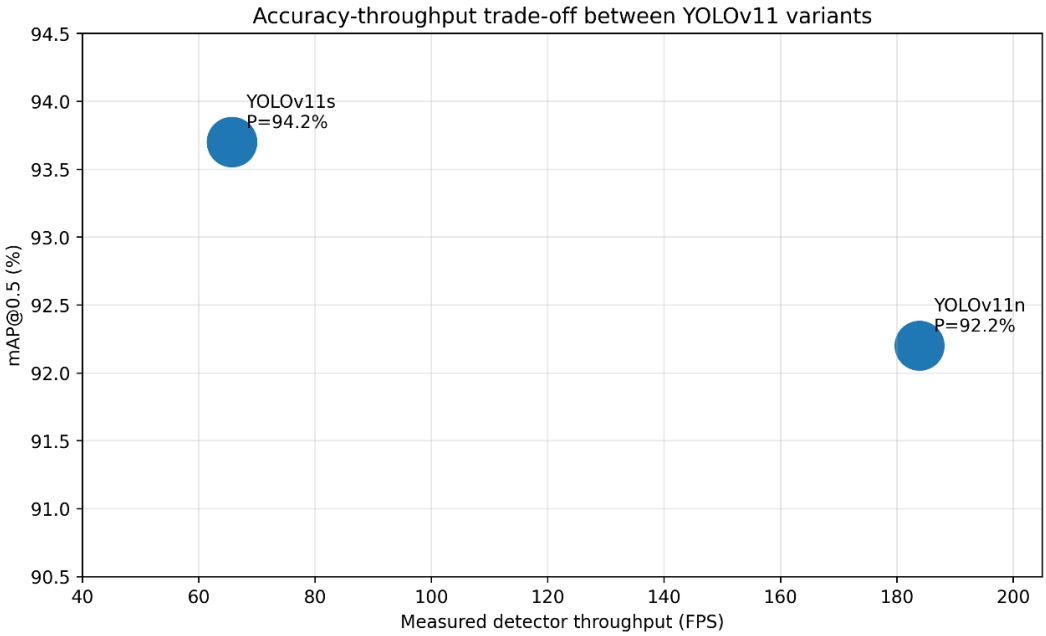

Figure 5. Accuracy–throughput trade-off between YOLOv11n and YOLOv11s. Marker size represents detector precision.

Since both birds and drones are moving, there are variations in size between birds and aircraft, and between trees and power pylons. These variations must be considered when training the model to quickly detect objects before any damage occurs to the drone. Object interference, such as birds entering trees, must also be considered, as this can partially obscure the birds and slow down the detection. Therefore, it is essential to reduce overfitting. Increasing the number of images from 10836 to 20,000 reduces overfitting and minimizes the discrepancy between training and validation data.

Therefore, several different data sources were collected and tested, namely, Robflow and Kaggle, combined multiple datasets, and increased the number of categories while trying to maintain a balance between categories to obtain the best results.

4.5.3. Video-Based Detection and Tracking It is important to understand that relying on image results may not be enough for a fully effective system. Video is needed alongside images. This is because each image on its own does not have a fixed timeframe and there is no relationship between one frame and the next. Drones and birds are things that move around a lot and are not stable. They can interact with things around them like trees when they are moving. The use of video and images was employed to make a complete system. Video and images can help us understand drones and birds better. Since moving objects may appear and disappear from the frame, then reappear, the model might recognize them as new objects. To address this issue, ByteTrack technology was used to maintain the object's identity during its disappearance and reappearance by assigning it to a unique identifier to mitigate the problem of temporary loss and treat it as a new object during the detection process.

4.5.4. Multi-Object Tracking with ByteTrack Drones and birds are highly mobile and unstable objects, so work has been done to develop a system that combines video and images. Video and images help us better understand the movement of drones and birds. Because moving objects may appear and disappear from the frame and then reappear, the model might treat them as new objects. To address this issue, ByteTrack was used to maintain the object's identity during its disappearance and reappearance by assigning it to a unique identifier. This reduces the problem of temporary loss and treats it as a new object during the detection process. Why was this tracking method chosen? Because it offers several advantages that support our research. First, tracking is detection-based,

and while some tracking systems rely on high-reliability detection, ByteTrack operates on both high and low-reliability detection. For example, when using YOLO, an object disappears into an indistinct environment like trees, resulting in low reliability. On the other hand, ByteTrack is considered here since it can match detections both of high and low confidence, which could help track identity even when detection confidence becomes lower due to partial occlusion. Tracking in this research is considered qualitatively as there is no ground truth trajectory [1].

4.6. Training Configuration

The model was developed to enable obstacle detection and moving object tracking, overcoming the limitations of relying solely on detection. Two lightweight YOLOv11 variants, YOLOv11n and YOLOv11s, were evaluated. YOLOv11s was selected as the main detector because it achieved the highest mAP@0.5 of 93.7%, whereas YOLOv11n achieved the highest detector throughput of 183.9 FPS. To achieve this, the data were initially integrated and optimized, as described in previous sections, and the dataset was divided into three parts: training, validation, and testing. The overall image resolution was fixed at 640 x 640 pixels. Sixty-four images were used per batch. The maximum training duration was configured as 5,000 epochs. However, the checkpoint that achieved the best validation performance was obtained at epoch 332 and was selected for final testing.

4.7. Hardware and Software Environment for Model Evaluation

The hardware and software environment used for model testing is outlined in Table 7 below. All experiments were executed on a paid Google Colab with GPU capabilities. The assigned accelerator was NVIDIA Tesla T4 with 15,095 MiB of memory. The testing environment was configured with Python 3.12.12, PyTorch 2.9.0+cu126, and Ultralytics 8.3.224. Models were tested using an image resolution of 640 × 640 pixels and a batch size of 1.

Table 7. Hardware and software environment used for model evaluation.

Parameters	**values**
Experimental platform	Google Colab
GPU	NVIDIA Tesla T4
GPU memory	15,095 MiB
Python	3.12.12
PyTorch	2.9.0+cu126
Ultralytics	8.3.224
Input resolution	640 × 640 pixels
Evaluation batch size	1

4.8. Evaluation Metrics

The performance of the detector was calculated using measures such as precision, recall, F1-score, mAP@0.5 and mAP@0.5:0.95. Precision is a ratio of the number of predictions which are true detections. Recall on the other hand is a ratio of all the actual objects which have been detected by the system. F1-score is the balanced score between precision and recall. mAP@0.5 is the mean average precision score calculated at IoU of 0.5. mAP@0.5:0.95 is the mean AP score calculated across IoUs from 0.5 to 0.95 with an interval of 0.05.

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F_1 = 2\,\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{IoU} = \frac{|B_p \cap B_{gt}|}{|B_p \cup B_{gt}|}$$

$$\text{AP} = \int_0^1 p(r)\, dr$$

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i$$

$$\text{FPS} = \frac{1000}{t_{\text{frame}}}$$

TP denotes a true-positive detection, FP denotes a false-positive detection, and FN denotes a false-negative detection.

5. Results

This part describes the quantitatively estimated outcomes of YOLOv11s, YOLOv11n, Tiny-YOLOv7, and YOLOv8s assessment in regard to the gathered four-class dataset. The model performance was assessed with the help of precision, recall, F1 score, mAP@0.5, and mAP@0.5:0.95 measures. The best checkpoint of YOLOv11s was achieved during the epoch 332. The aforementioned metrics reflect the ability of the models to detect objects reliably and accurately localize bounding boxes, while their speed of inference is measured in frames per second separately. Tables 8, 9, and 10 illustrate the results of applying the model to dataset using YOLOv11s, Tiny-YOLOv7, and YOLOv8s, demonstrating the overall model accuracy and specifying the accuracy of each category individually. According to the current experiment setup, YOLOv11s demonstrated better results compared to the two baseline models Tiny-YOLOv7 and YOLOv8s. Nonetheless, numerical comparisons with the results of other studies that used another dataset can be considered only as a context-based comparison.

Table 8. Detection performance of YOLOv11s on the four-class dataset.

Class	**Precision**	**Recall**	**mAP@0.5**	**mAP@0.5:0.95**	**F1-score**
All classes	94.2%	89.6%	93.7%	73.0%	91.8%
Bird	98.2%	98.5%	99.4%	82.3%	98.3%
Drone	99.2%	98.2%	99.3%	77.3%	98.7%
Tree	81.2%	66.3%	78.6%	43.2%	73.0%
Power pylon	98.0%	95.5%	97.5%	89.1%	96.7%

Figure 6 provides a class-wise visual summary of precision, recall, and mAP@0.5.

While the bird, drone, and power-pylon classes achieved high precision, recall, and mAP values, the tree class yielded relatively poor results. As shown in Table 8, the tree class had lower precision, recall, and mAP than any other classes. The reasons for such reduction in accuracy are multiple. On the one hand, tree objects are visually irregular and consist of different elements such as branches, leaves, shadows, and texture. On the other hand, tree regions tend to be similar to the background because they contain vegetation, sky, shadows, and distant landscapes of the same colors and textures. Moreover, under occlusion and low visibility conditions, the object is prone to be confused with the background. Thus, most mistakes in relation to the tree class are false negative detection cases.

The qualitative tracking test was carried out using a video clip that contained 170 frames with a duration of 5.67 seconds and had an original frame rate of 30 FPS. The YOLOv11–ByteTrack system proved to be adequate for performing the task of target detection and tracking since it managed to process the entire sequence adequately. The YOLOv11s detector showed an detector throughput rate of 65.7 FPS on the test set of videos, which is significantly higher than the original 30 FPS of the considered video. However, the full tracking throughput rate

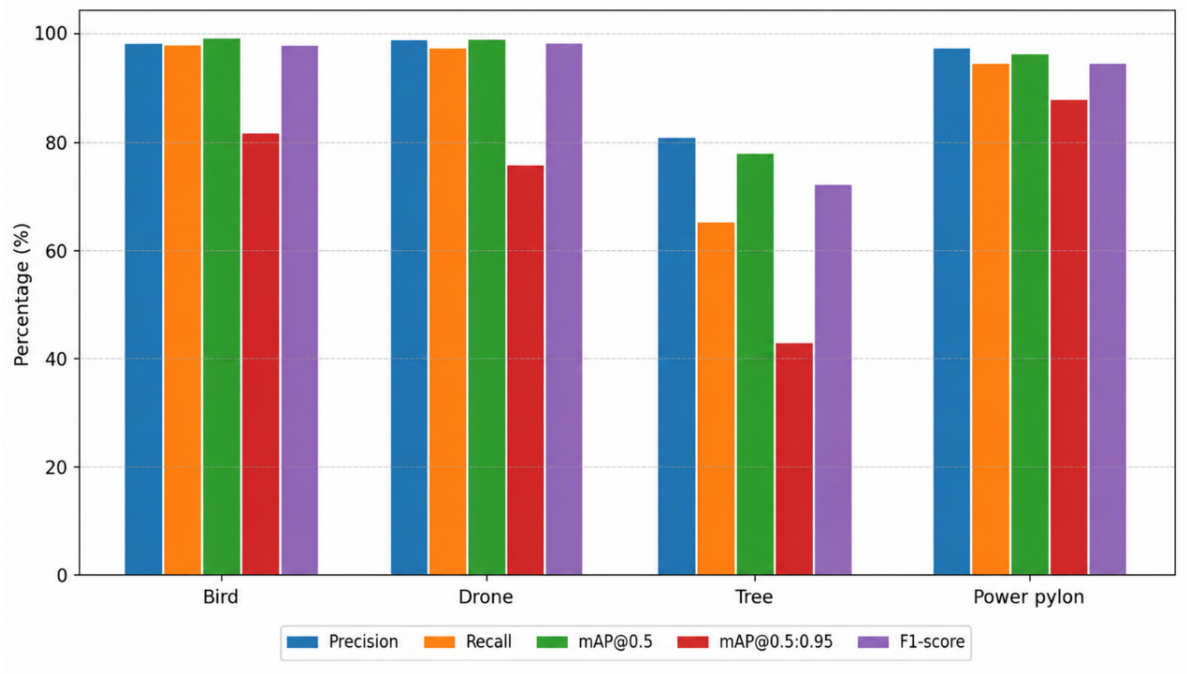

Figure 6. YOLOv11s performance by class.

Table 9. Results of Tiny-YOLOv7 on the collected dataset.

Class	Precision	Recall	MAP(50)	MAP (50-95)	F1-score
All	85.9%	75.9%	82.6%	46.6%	80.6%
Bird	91.0%	91.1%	93.6%	54.8%	91.0%
Drone	92.2%	84.5%	92.0%	57.2%	88.2%
Tree	76.1%	51.8%	62.6%	27.7%	61.6%
Power pylon	83.9%	76.4%	82.2%	46.8%	80.0%

Table 10. Results of YOLOv8s on the collected dataset.

Class	Precision	Recall	MAP(50)	MAP (50-95)	F1-score
All	82.1%	77.1%	79.5%	53.3%	79.5%
Bird	91.8%	93.1%	97.1%	72.1%	92.4%
Drone	92.4%	87.9%	93.8%	60.8%	90.1%
Tree	54.0%	39.0%	36.9%	14.8%	45.3%
Power pylon	90.2%	88.3%	90.4%	65.5%	89.2%

was not measured separately. It was also impossible to calculate any quantitative tracking metrics due to a lack of proper data.

Figure 7 shows the F1-confidence curves for birds, drones, trees, and pylons on a per-class basis. These curves show the trade-off between precision and recall with respect to various confidence levels. The combined curve attained its maximum value for F1-score at a confidence level of 0.37.

Figure 8 presents confidence threshold vs. precision for each obstacle class. An increase in confidence threshold results in better precision since less confident detection points are gradually ruled out, but this might also lead to lower recall rates.

Figure 9 shows the precision-recall curves for the four classes of obstacles. Area under the curve for the individual class indicates the Average Precision (AP) score for that particular class. The mean of AP scores for

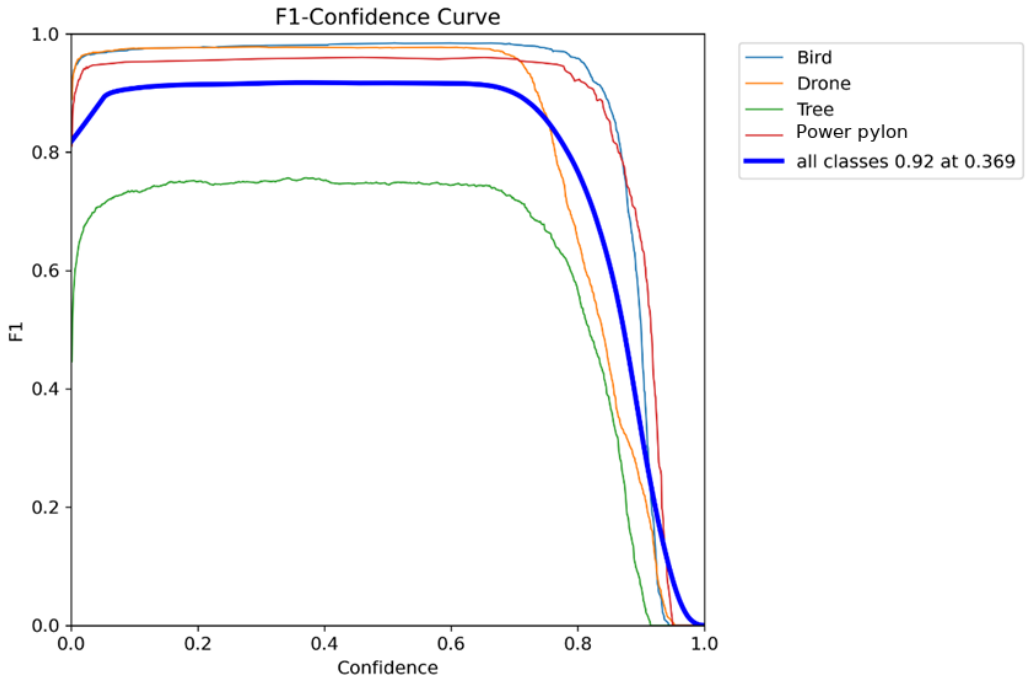

Figure 7. F1–confidence curves for the four obstacle classes.

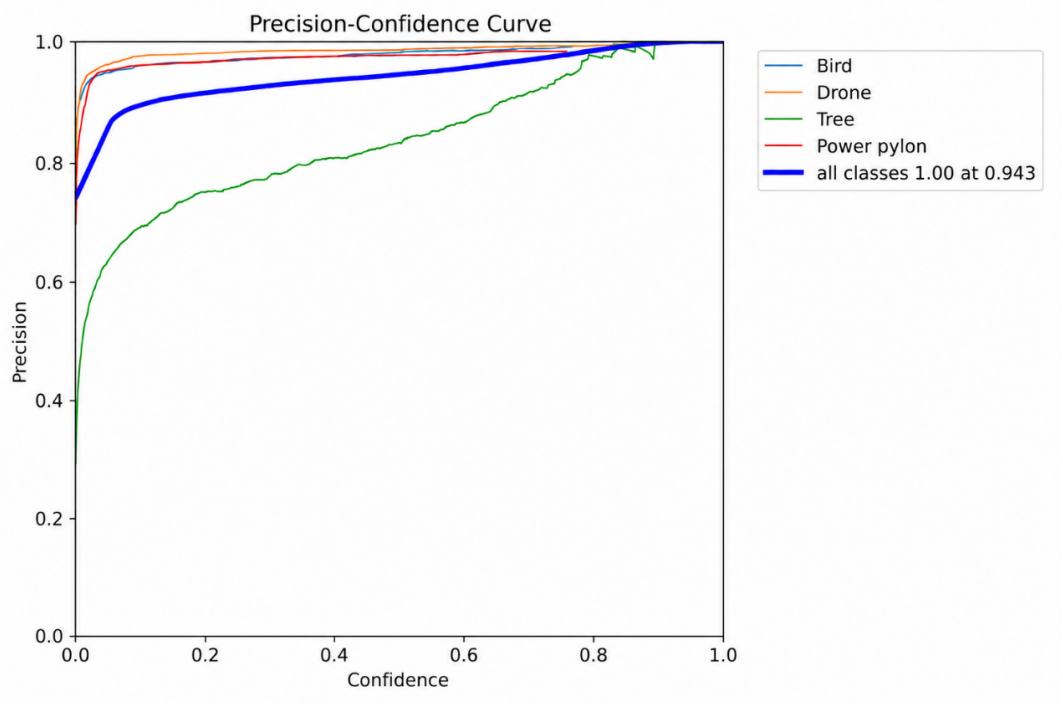

Figure 8. Precision–confidence curves for the four obstacle classes.

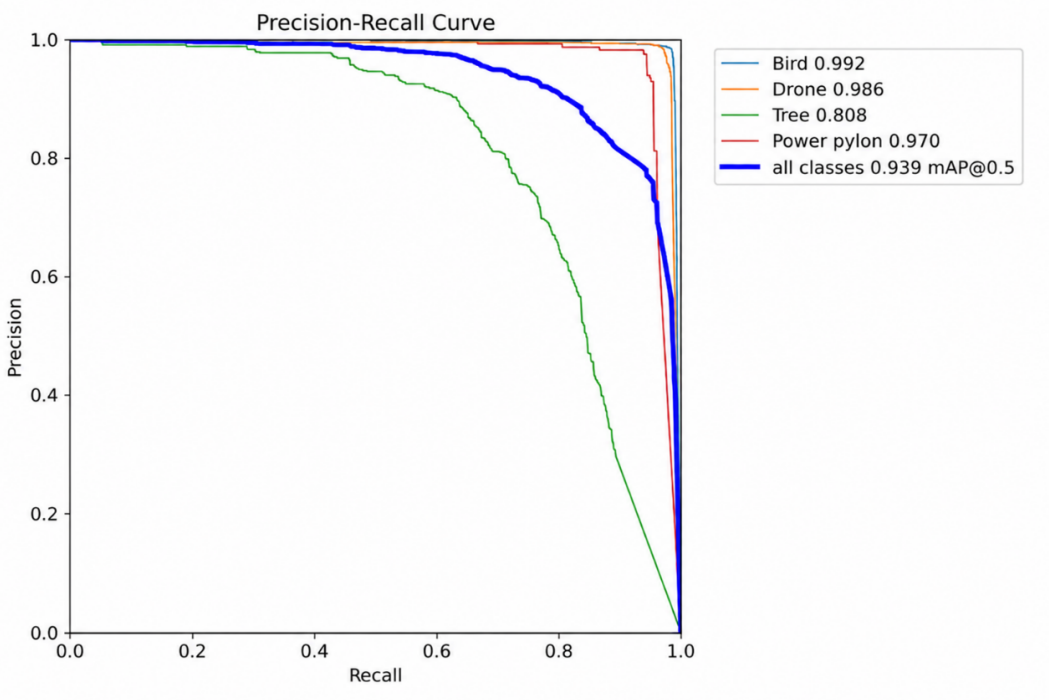

Figure 9. Precision–recall curves for the four obstacle classes.

all classes at a threshold IoU value of 0.5 indicates the mAP@0.5 score. YOLOv11s scored an mAP@0.5 of 93.7%.

Figure 10 Provides the confusion matrix normalized for YOLOv11s on the evaluation dataset. There is high consistency among the classes birds, drones, and power pylons, while the tree class is less consistent with background objects.

Figure 11 shows the extent to which the model's performance improves after training on the data. The horizontal line represents the number of epochs, and the vertical line represents loss. The drawing consists of more than one part. The first part shows the results of training, and the other part refers to the results of validation. As for the upper part, the figure (box_loss) shows the relationship between the prediction of the box and its real location. As the number of epochs increases, the number of losses decreases. The second part of the figure (cls_loss) shows the classification of obstacles in each class. It shows the dfl_loss that represents the accuracy of the bounding box, and the recall and precision that represents the accuracy of the model detection. It depicts the increase in accuracy with the increase in the number of epochs.

Figure 12 shows the validation batch results after applying the YOLOv11 object detection algorithm. It further illustrates the diversity of the images captured, reflecting variations in lighting between environments, object sizes, and the size of the enclosure boxes around obstacles. The numbers on the images represent the confidence scores for detection, demonstrating the model's ability to efficiently detect obstacles under varying conditions, distances, and sizes.

6. Conclusion

In this research, a deep learning-based approach using YOLOv11 has been proposed to detect obstacles encountered by drones, such as birds, other drones, trees, and power pylons. The proposed approach also tracks moving obstacles like birds and drones to address the problem of these objects interfering with complex

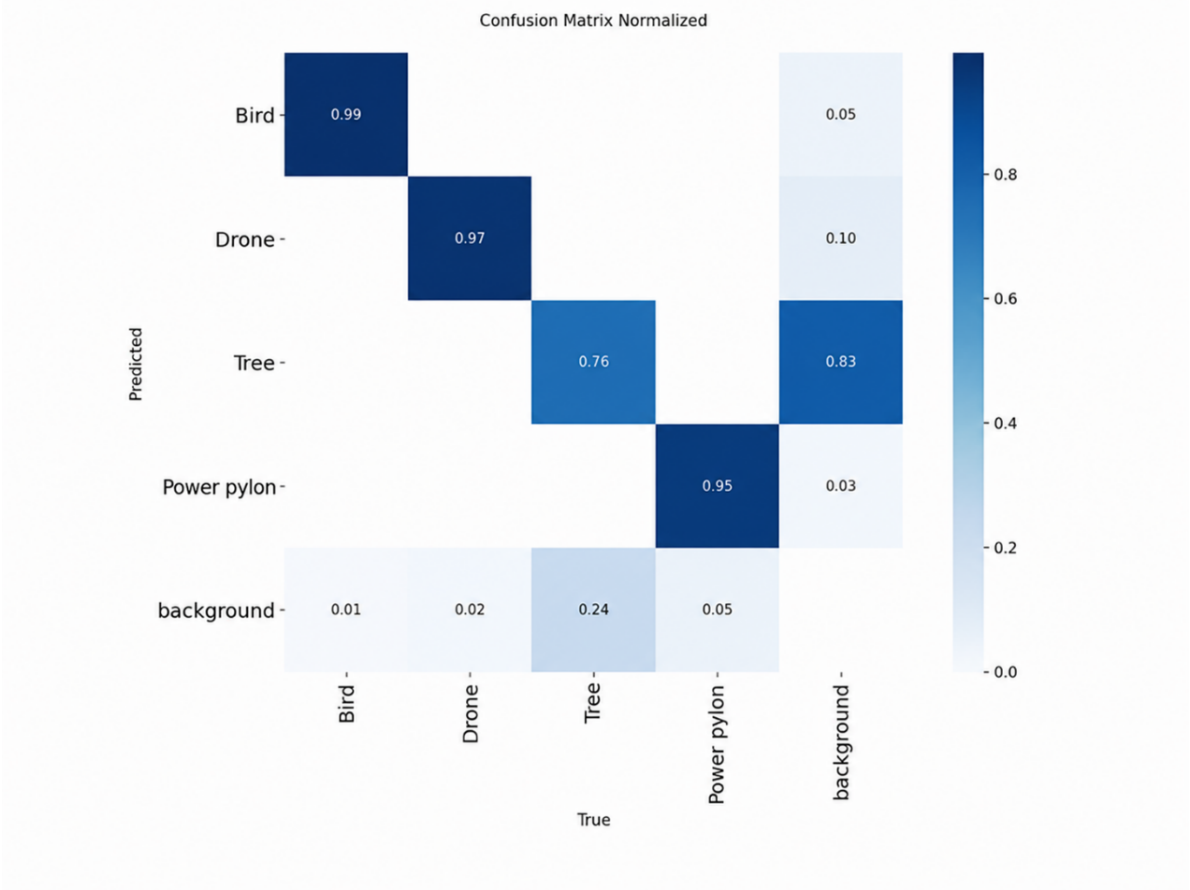

Figure 10. Normalized confusion matrix of the YOLOv11s detector on the final evaluation set.

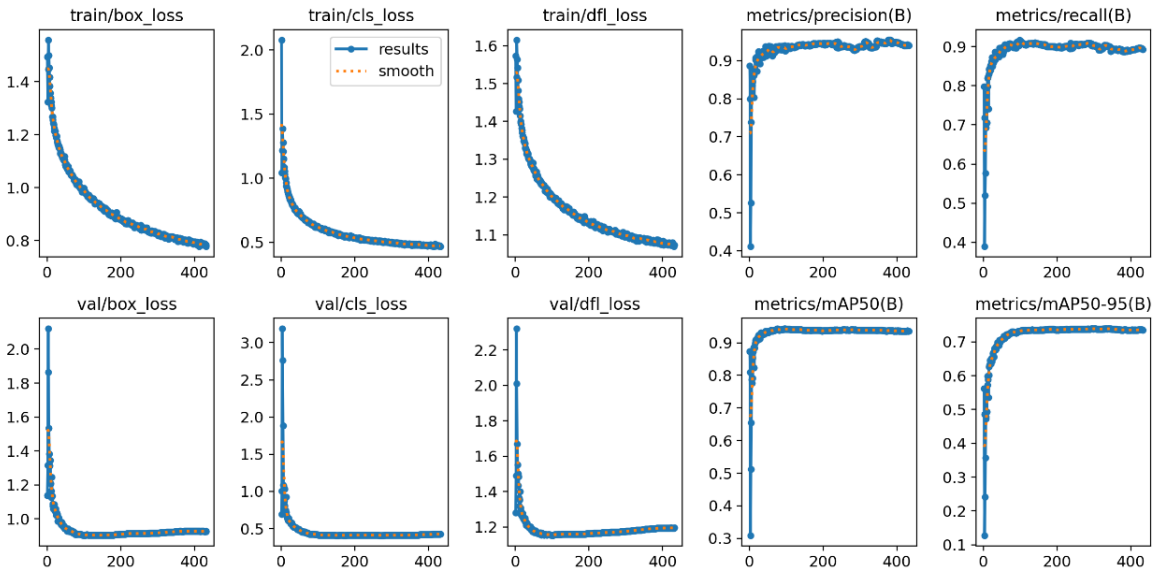

Figure 11. Training and validation losses and detection metrics of YOLOv11s across the training epochs.

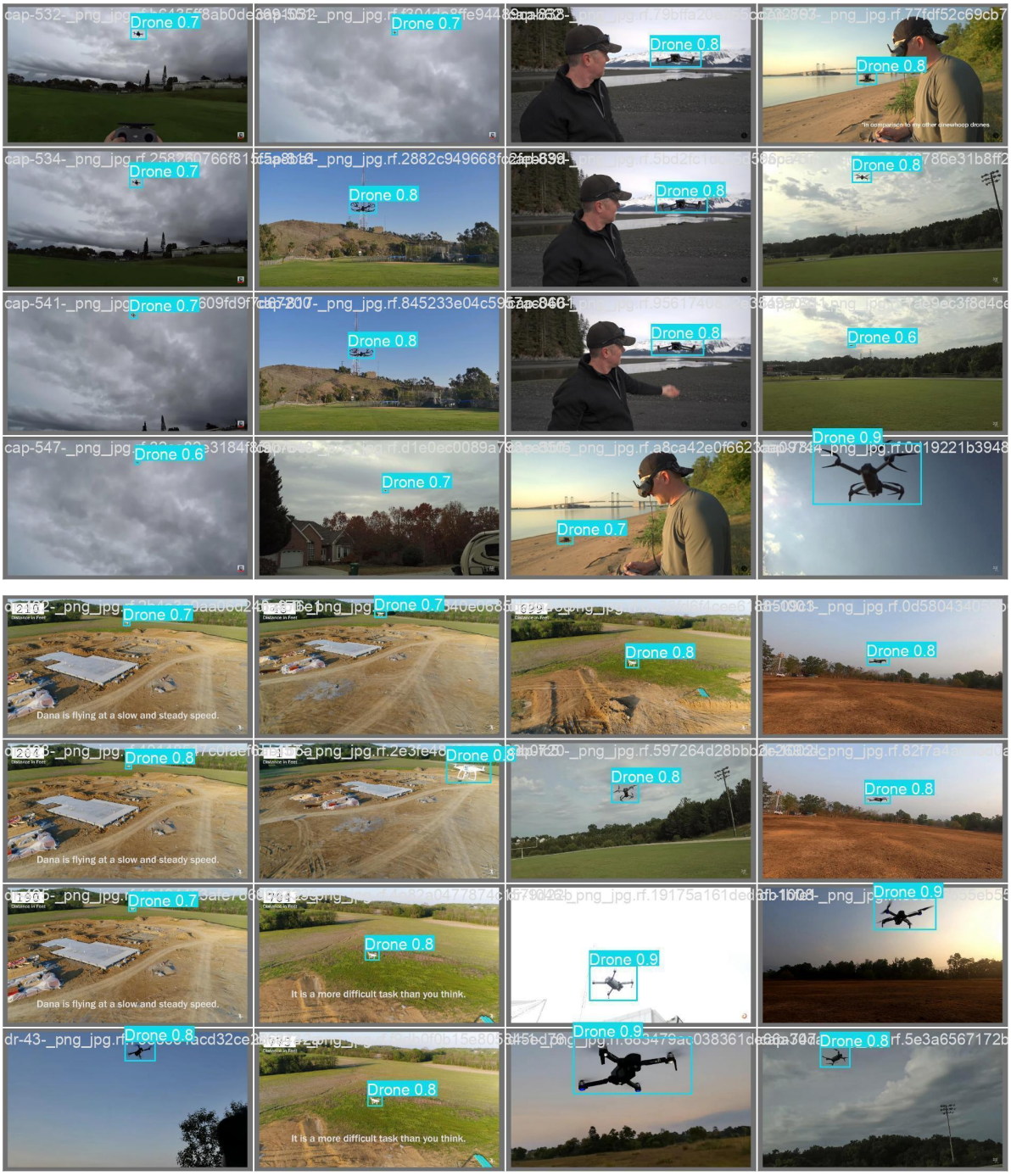

Figure 12. Qualitative predictions on representative validation batches.

environments, such as trees. To accomplish this, Creation and availability of a new four-category UAV obstacles dataset compiled using three existing publicly available datasets. The three datasets were combined through class label normalization, annotation validation, invalid sample elimination, pre-processing, and training data augmentation to give a total of 20,000 images and 29,649 annotated objects. The YOLOv11 (Nano and small)

algorithm was then applied to detect and classify obstacles including the four categories. Finally, ByteTrack was integrated to track moving objects in complex environments. The results from the quantitative analysis illustrate the performance of the YOLOv11s object detector on the four obstacles. ByteTrack algorithm is independently integrated as a qualitative time association part for moving objects; however, the claim of being better than any previous approach cannot be made without training the alternative algorithms on the same data set and under the same conditions. It achieved a precision of 94.2% in the combined categories. It also achieved high scores in the bird category (98.2%), the other drone category (99.2%), the power pylon category (98%), and the tree category (81.2%). These results confirm the model's effectiveness and efficiency in detecting obstacles faced by drones.

7. Data Availability

Images and annotations used for the generation of the four-class UAV obstacle dataset were sourced from three public repositories. Images of birds and drones were sourced from the Drone-Bird-Detection dataset on Roboflow Universe, images of trees were sourced from the Urban Tree Detection Dataset on Kaggle, while images of power pylons were sourced from the Transmission System dataset on Roboflow Universe.

Through image quality filtering, annotation verification, and class normalization, 10,836 source images have been retained. They consist of 2,928 images of birds, 4,284 images of drones, 2,716 images of trees, and 908 images of power pylons. Details about preprocessing and data augmentation can be found in Sections 4.2 and 4.3.

Original source datasets are available through the links provided in Table 2 below. The cleaned annotations, mapping of classes, training configuration file, and train/validation/test split manifests are available at the following link:
https://www.kaggle.com/datasets/afnanabdelfattah/an-image-based-obstacle-detection.

The original images are still subject to the licensing and usage policies in the source repositories. No private or personal data was used in this study.

REFERENCES

1. Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "ByteTrack: Multi-Object Tracking by Associating Every Detection Box," in Computer Vision–ECCV 2022, LNCS 13682, pp. 1–21, 2022, doi: 10.1007/978-3-031-20047-2_1.
2. A. Yasmeen and O. Daescu, "Recent research progress on ground-to-air vision-based anti-UAV detection and tracking methodologies: A review," Drones, vol. 9, no. 1, Art. no. 58, Jan. 2025, doi: 10.3390/drones9010058.
3. J. Zhao et al., "The 3rd Anti-UAV Workshop & Challenge: Methods and results," arXiv preprint, arXiv:2305.07290, 2023.
4. M. Jian, Z. Lu, and V. C. Chen, "Drone detection and tracking based on phase-interferometric Doppler radar," in Proc. IEEE Radar Conf. (RadarConf), 2018, pp. 1146–1149.
5. Q. Shi and J. Li, "Object detection of UAV for anti-UAV based on YOLOv4," in Proc. IEEE Int. Conf. Civil Aviation Safety and Information Technology (ICCASIT), 2020, pp. 1048–1052.
6. B. Taha and A. Shoufan, "Machine learning-based drone detection and classification: State-of-the-art in research," IEEE Access, vol. 7, pp. 138669–138682, 2019 doi: 10.1109/ACCESS.2019.2942944.
7. X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios," in Proc. IEEE/CVF Int. Conf. Computer Vision Workshops (ICCVW), 2021, pp. 2778–2788.
8. A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and Flexible Image Augmentations," Information, vol. 11, no. 2, Art. no. 125, 2020, doi: 10.3390/info11020125.
9. Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object Detection in 20 Years: A Survey," Proceedings of the IEEE, vol. 111, no. 3, pp. 257–276, 2023, doi: 10.1109/JPROC.2023.3238524.
10. R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 580–587, 2014, doi: 10.1109/CVPR.2014.81.
11. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.
12. W. Liu et al., "SSD: Single Shot MultiBox Detector," in Computer Vision–ECCV 2016, Lecture Notes in Computer Science, vol. 9905, pp. 21–37, 2016, doi: 10.1007/978-3-319-46448-0_2.
13. M. Saqib et al., "A study on detecting drones using deep convolutional neural networks," in Proc. IEEE Int. Conf. Advanced Video and Signal-Based Surveillance (AVSS), 2017, pp. 1–5.

14. A. T. Abu-Jassar, H. Attar, A. Amer, V. Lyashenko, V. Yevsieiev, and A. Solyman, "Development and Investigation of Vision System for a Small-Sized Mobile Humanoid Robot in a Smart Environment," International Journal of Crowd Science, vol. 9, no. 1, pp. 29–43, 2025, doi: 10.26599/IJCS.2023.9100018.
15. L. Zhu et al., "YOLO-Drone: Airborne real-time detection of dense small objects from high-altitude perspective," arXiv preprint, arXiv:2304.06925, 2023.
16. M. Hanif, T. Muhammad, A. Ikram, S. Yousaf, M. Abu-Zanona, A. M. A. Al Khateeb, B. Elzaghmouri, S. M. A. R. Ahmed, and L. H. Rahamatalla, "Ambulance Detection and Priority Passage at Urban Intersections Using Transfer Learning and Explainable AI," International Journal of Advanced Computer Science and Applications, vol. 16, no. 10, pp. 285–292, 2025, doi: 10.14569/IJACSA.2025.0161030.
17. K. Sricharan and M. Venkat, "Real-Time Drone Detection Using Deep Learning," in Proc. Second International Conference on Emerging Trends in Engineering, Advances in Engineering Research, vol. 223, pp. 905–918, 2023, doi: 10.2991/978-94-6463-252-1_91.
18. S. K. Shandilya et al., "YOLO-Based Segmented Dataset for Drone vs. Bird Detection for Deep and Machine Learning Algorithms," Data in Brief, vol. 50, Art. no. 109355, 2023, doi: 10.1016/j.dib.2023.109355.
19. B. Aydin and S. Singha, "Drone Detection Using YOLOv5," Eng, vol. 4, no. 1, pp. 416–433, 2023, doi: 10.3390/eng4010025.
20. R. D. Suvaris, R. Suryodai, S. Narayanasamy, A. Saravanan, R. Kumar, P. N. V. S. Rao M., and E. Muniyandy, "Real-Time Multi-Scale Object Detection in Surveillance Using Hybrid Transformer Architecture," International Journal of Advanced Computer Science and Applications, vol. 16, no. 10, pp. 762–774, 2025, doi: 10.14569/IJACSA.2025.0161077.
21. Z. U. Rehman, A. Syed, A. Tayab, G. G. Tejani, D. S. Khafaga, and E.-S. M. El-Kenawy, "Improving Real-Time Animal Detection Using Group Sparsity in YOLOv8: A Solution for Animal-Toy Differentiation," Computers, Materials & Continua, vol. 86, no. 2, pp. 1–25, 2026, doi: 10.32604/cmc.2025.070310.
22. Y. Farhaoui, A. E. Allaoui, J. Rasheed, and O. Osman, "Fine-Tuned Object Detection for Mask Recognition Using Green Computing in IoT Systems," Journal of Image and Graphics, vol. 13, no. 5, pp. 561–569, 2025, doi: 10.18178/joig.13.5.561-569.
23. F. N. M. Zamri et al., "Enhanced Small Drone Detection Using Optimized YOLOv8 with Attention Mechanisms," IEEE Access, vol. 12, pp. 90629–90643, 2024, doi: 10.1109/ACCESS.2024.3420730.
24. M. Kabir et al., "Drone detection from video streams using image processing techniques and YOLOv7," Int. J. Imag21e Graph. Signal Process., vol. 16, pp. 83–95, 2024.
25. R. Laroca, M. dos Santos, and D. Menotti, "Improving Small Drone Detection Through Multi-Scale Processing and Data Augmentation," in Proc. 2025 International Joint Conference on Neural Networks (IJCNN), pp. 1–8, 2025, doi: 10.1109/IJCNN64981.2025.11227421.
26. N. Al-Qubaydhi et al., "Detection of Unauthorized Unmanned Aerial Vehicles Using YOLOv5 and Transfer Learning," Electronics, vol. 11, no. 17, Art. no. 2669, 2022, doi: 10.3390/electronics11172669.
27. R. Hakani and A. Rawat, "Edge Computing-Driven Real-Time Drone Detection Using YOLOv9 and NVIDIA Jetson Nano," Drones, vol. 8, no. 11, Art. no. 680, 2024, doi: 10.3390/drones8110680.
28. R. Gandhi, "UAV Object Detection and Tracking in Video Using YOLOv3 and DeepSORT," in Proc. 2024 International Conference on Emerging Technologies in Computer Science for Interdisciplinary Applications (ICETCS), pp. 1–6, 2024, doi: 10.1109/ICETCS61022.2024.10543307.