Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 9999596, 15 pages
https://doi.org/10.1155/2021/9999596

Research Article

Multitarget Real-Time Tracking Algorithm for UAV IoT

Tao Hong,[1,2] **Qiye Yang**,[2] **Peng Wang**,[3] **Jinmeng Zhang**,[4] **Wenbo Sun**,[5] **Lei Tao**,[2] **Chaoqun Fang**,[2] **and Jihan Cao**[2]

[1]*Yunnan Innovation Institute·BUAA, Kunming, China*
[2]*Beijing Key Laboratory for Microwave Sensing and Security Applications, Beihang University, Beijing, China*
[3]*Joint War College, National Defense University of PLA, Beijing, China*
[4]*Beijing University of Agriculture, Beijing 102206, China*
[5]*Aerospace Hi-Tech Holding Group Co. Ltd., Beijing, China*

Correspondence should be addressed to Tao Hong; hongtao@buaa.edu.cn

Received 8 March 2021; Revised 23 June 2021; Accepted 17 July 2021; Published 26 August 2021

Academic Editor: Bo Rong

Unmanned aerial vehicles (UAVs) have increased the convenience of urban life. Representing the recent rapid development of drone technology, UAVs have been widely used in fifth-generation (5G) cellular networks and the Internet of Things (IoT), such as drone aerial photography, express drone delivery, and drone traffic supervision. However, owing to low altitude and low speed, drones can only limitedly monitor and detect small target objects, resulting in frequent intrusion and collision. Traditional methods of monitoring the safety of drones are mostly expensive and difficult to implement. In smart city construction, a large number of smart IoT cameras connected to 5G networks are installed in the city. Captured drone images are transmitted to the cloud via a high-speed and low-latency 5G network, and machine learning algorithms are used for target detection and tracking. In this study, we propose a method for real-time tracking of drone targets by using the existing monitoring network to obtain drone images in real time and employing deep learning methods by which drones in urban environments can be guided. To achieve real-time tracking of UAV targets, we employed the tracking-by-detection mode in machine learning, with the network-modified YOLOv3 (you only look once v3) as the target detector and Deep SORT as the target tracking correlation algorithm. We established a drone tracking dataset that contains four types of drones and 2800 pictures in different environments. The tracking model we trained achieved 94.4% tracking accuracy in real-time UAV target tracking and a tracking speed of 54 FPS. These results comprehensively demonstrate that our tracking model achieves high-precision real-time UAV target tracking at a reduced cost.

1. Introduction

The application of 5G and the Internet of Things (IoT) represents the development of future drone technology. The development of 5G technology has facilitated the emergence of smart cities [1]. The recent construction of smart cities in China has been elevated to a national strategy with the strong support of the state, and considerable progress has been made. As of February 2019, 100% of subprovincial cities and 93% of prefecture-level cities of China—consisting of more than 700 cities in total (including county-level cities)—have proposed or constructed smart cities. Regardless of the scale of the smart city market or smart city information technology investment, both exhibit a rapid growth trend and a large future market space. IDC (Internet Data Center), an international data company, predicts that investments related to global smart city technology will reach 189.46 billion US dollars in 2023. Meanwhile, the same investment in China will reach 38.92 billion US dollars.

Smart cities comprehensively promote the development of modern living through information technologies, such as the IoT, cloud computing, and geospatial infrastructure, in Figure 1. Cities that want high-quality development need smarter infrastructure, in addition to roads, viaducts, hydropower, and so on. Smart city infrastructure, such as the IoT, tends to become increasingly popular with the development of 5G networks. Smart cities present new changes on multiple levels, and the construction of smart cities provides

Figure 1: Drone in IoT smart cities.

unprecedented opportunities for development [2]. To support the information infrastructure of smart cities, the focus is diverted to full coverage of the 5G network and object recognition cameras, among others, which can be gradually integrated with the IoT infrastructure to achieve a comprehensive IoT. From the perspective of technological development, smart city construction requires the realization of full perception, interconnection, universal computing, and integrated applications through the IoT, cloud computing, and other modern information technology applications represented by mobile technology.

The advantage of a smart city lies in its safety and security. The steady progress of 5G technology has also facilitated the continued optimization of city monitoring systems. Traditional security surveillance system cameras can be replaced by smarter 5G cameras. 5G technology has three characteristics: (i) high throughput, which solves the bandwidth transmission problem of video upstream and downstream; (ii) low-latency, which can achieve a theoretical value of 1 ms or 10 ms in the 5G era; and (iii) ultra-large-scale. 5G inherently supports ultra-large-scale device access and can support more cameras and other IoT devices for the security industry [3]. Only a city surveillance system with a wider bandwidth and higher video stability can be combined with an intelligent video surveillance cloud platform to achieve true intelligent security. Moreover, compared with wired video transmission, 5G wireless transmission is easier to deploy, more convenient, and lower in cost. The high transmission, high broadband, and high reliability of 5G can provide a UAV identification system with more high-definition monitoring data at a faster speed. Using a surveillance system composed of a network of urban surveillance cameras to monitor, identify, and warn drones in designated areas in real time can effectively solve the problems in drone surveillance. The key to this method is to determine how to effectively detect whether a drone to track and locate from the video exists. In this study, we attempt to solve this problem by machine learning.

With the continuous industrial and technological progress, low-altitude, slow-speed, and small-target UAVs have been rapidly developed and are widely used. UAVs fly at low altitudes; thus, radar waves may not reach these targets as they are affected by the curvature of the earth and the shelter of buildings. In addition, a large amount of ground clutter will enter the radar receiver while it is working, which impedes radar from distinguishing the echo signal of a UAV target. The flying speed of UAVs is slow, and some are even lower than the radar speed detection threshold, preventing the pulse-Doppler radar from detecting the target. Slow-flying UAVs are easily confused with slow-moving clutter, such as weather clutter and bird swarm, rendering target t recognition difficult. Their small size and radar reflection area, in addition to the weak echo signal, weaken w the UAV detection ability of radar. However, owing to difficulties in target detection and effective supervision, such UAVs have incurred major security threats to countries in recent years. In 2017, UAV interference occurred at Kunming Changshui International Airport in China, resulting in 35 flights being forced to divert to alternate routes and 28 flights being delayed at the airport. Some flights were delayed for 4 h, and the airport runway was forced to close for 45 min. In 2018, "black flight" and "disturbing flight" UAV events in Germany showed rapid growth trends. As of August 12, 2018, more than 100 UAV interference incidents occurred in major airports in Germany, which exceeded the total number reported in 2017. Thus, methods for detecting such targets have significant and immediate application requirements. Various recognition methods for UAV detection currently exist, including radar technologies, audio signal analysis, trajectory analysis, and image recognition.

Radar technology has been widely used in UAV detection and classification because of its fast-remote sensing ability. In radar technology, frequency-modulated continuous-wave (FMCW) radar is the most common choice because it can obtain sufficient information about targets with a short dwell time. When the position of the UAV target changes relative to the radar, the rotation of the wing can cause the radar echo to modulate and produce a micro-Doppler effect. Analysis of the micro-Doppler characteristics of radar echo can extract detailed information such as the number of rotors of the UAV target. The use of FMCW in surveillance systems is limited because it cannot detect the distance from the target to the radar, requiring manual intervention [4]. For UAV recognition based on audio signals, the multirotor UAV can produce FM (Frequency Modulation) noise when flying. The current study proposes the RPM (rotation per minute) speed wave correction method, which corrects the classical noise prediction method of rotorcrafts, considers the frequency modulation effect, and identifies UAVs. Large-scale public activity venues, airports, government agencies, and other places requiring UAV detection have large background noise; thus, the noise emitted by UAVs for identification is difficult to capture [5]. To address this concern, a UAV flight path recognition network based on radar monitoring data is constructed in accordance with the conditions identified by research on UAV flight path identification and the characteristics of radar data information of low-altitude surveillance systems. A UAV identification method based on radar monitoring data and recurrent neural network (RNN) flight path

identification network is then proposed. Modeling with RNN-Long Short-Term Memory networks after training can identify the target trajectory in airspace, efficiently identify the trajectory data of the UAV, and achieve the expected classification effect. However, collecting trajectory information entails time, suggesting that a UAV is impossible to identify and track in real time.

Traditional UAV detection techniques are often limited in public places, government agencies, airports, and other areas. Moreover, radar systems experience difficulty operating effectively in environments with many shelters, where the RPM cannot operate because excessive background noise and track recognition cannot meet the requirements of real-time monitoring. For the past few years, the rapid improvement of deep learning and computing abilities have greatly enhanced the accuracy and speed of image target recognition and classification, allowing the tracking of algorithms based on target detectors. Relevant research has determined that existing methods employing a target detection algorithm to identify UAVs can only be used to identify targets but not to target the identified drones. Meanwhile, the existing UAV tracking algorithm has deficiencies in robustness and real-time performance. To address these issues, this study introduces the target tracking algorithm into the UAV recognition algorithm.

2. Related Work

The basic idea of multitarget real-time tracking for UAV positioning in IoTs is illustrated in Figure 2. The IoT camera is arranged in the city, and the captured images are transmitted to the cloud via 5G at a high speed. The real-time machine learning algorithm is used to track multiple targets of the drone. To achieve real-time target tracking, research on related machine learning algorithms is also necessary.

Currently, the most widely used deep learning target detection method is the target detection algorithm based on CNNs (Convolutional Neural Networks). The development of CNNs has a long history. In 1962, Hubel and Wiesel used the brain of a cat to explore the visual system. In 1980, Japanese scientist Kunihiko Fukushima proposed a neural network structure with a convolutional layer and a pooling layer. In 1998, Yann LeCun proposed LeNet-5 and applied the backpropagation algorithm for training this type of neural network structure, forming a prototype for the contemporary CNN. In the 2012 ImageNet Image Recognition Competition Challenge, AlexNet proposed a deep structure and dropout method discussed in the study by Hinton et al. in which the error rate decreased from more than 25% to 15%, revolutionizing the field of image recognition. Following the idea of AlexNet, LeCun et al. proposed DropConnect in 2013, further reducing the error rate to 11%. Network in Network was proposed by Yan Shuicheng et al. of NUS (National University of Singapore), significantly altering the structure of CNN. Based on these methods, Inception and VGG architectures deepened the network in 2014 to about 20 layers and significantly improved the image recognition error rate to 6.7%, indicating its similarity to the human error rate of 5.1%. Ren Shaoqing and He Kaiming et al. of MSRA

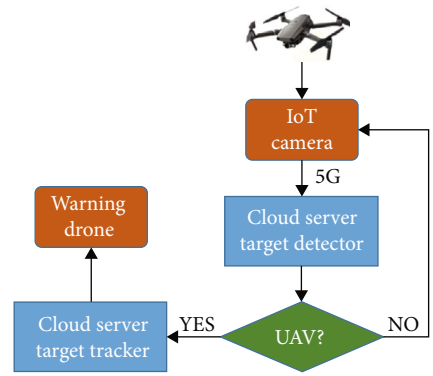

Figure 2: Drone tracking flowchart.

(Microsoft Research Asia) optimized the original R-CNN (region-based CNN) and Fast R-CNN with the development of Faster R-CNN. The main contributions of Faster R-CNN are the use and image recognition of the same CNN features in which features can not only recognize the category of objects in the image but also record their positions. He Kaiming subsequently introduced Mask R-CNN and added a mask head to Faster R-CNN. By using the mask head only in training, the mask head message was passed back to the original CNN feature, allowing the original feature to contain more detailed information. Currently, the structure of CNN is becoming increasingly complex, and the research direction is to find algorithms that can automatically optimize its structure. Supported by a deep coproduct neural network, the target detection method can be divided into two categories: the two-stage method and the one-stage method [6–8].

2.1. Target Detection Algorithms

2.1.1. Two-Stage Approaches. The two-stage approach divides the target detection task into two phases: RoI (region of interest) extraction, followed by RoI classification and regression. R-CNN, SPPNet, Fast R-CNN, Faster R-CNN, Mask R-CNN, and Cascade R-CNN, among others, are all two-stage approaches. Fast R-CNN before object detection is driven by the region proposal approach and the region-based CNN. Fast R-CNN uses a very deep network to achieve near real-time rates when the time spent on regional proposals is ignored; meanwhile, the proposal is also the bottleneck of inference in the detection system. Regional proposal methods often rely on inexpensive functions and economic reasoning schemes. Selective search is a widely used technique but is slower than effective detection networks by about one order of magnitude [9].

2.1.2. One-Stage Approaches. The one-stage method eliminates RoI extraction and then directly classifies and regresses the candidate anchor boxes. As its name suggests, this approach follows a completely different objective to apply a single neural network to the entire image. The representative algorithms are YOLO, R-SSD, RefineNet, and so on. Contrary to the classifier-based method, YOLO is trained on the loss function directly corresponding to the detection performance, while the whole model is trained under joint training [10]. As a representative one-stage method, YOLO has

significant advantages in computational speed over the two-stage method. Subsequent versions of YOLO (YOLOv2 and YOLOv3) exhibit improved detection accuracy while maintaining a high detection speed.

The development of anchor-based approaches has involved several issues. In the anchor mechanism, solid prior knowledge is needed to set the appropriate super parameters of scale and aspect ratio, and the number of targets in an image is often limited. Setting a large number of anchor boxes based on each anchor can produce a large number of easy samples and lead to an out-of-balance state of positive and negative samples. When target classification is based on the anchor box, the threshold of intersection over union (IOU) is also difficult to set. By contrast, the anchor-free method can avoid the anchor. While it cannot provide high detection stability, its calculation time is significantly reduced, allowing real-time high-precision detection and segmentation.

2.1.3. Anchor-Based. These methods typically require numerous anchors to ensure a sufficiently high IOU rate on the ground. Various hyperparameters and design choices are also possible with anchor boxes, and these methods can become more complex when these choices are used in conjunction [11].

2.1.4. Anchor-Free. Generally, the nonanchor detector belongs to a first-class detector. Although the performance of the one-stage approach CornerNet remains limited by its relatively weak ability to reference global information, its productivity can be increased by its ability to perceive visual patterns within each proposed area. Thus, it can independently identify the correctness of each bounding box.

2.2. Target Tracking Algorithms. Early tracking patterns used temporal and spatial points of interest, which could not be separated from some low-level features, such as corners and intensity peaks [12, 13]. Although early classical algorithms were able to achieve high-precision single-target tracking, they could not meet the high-speed multitarget tracking problem. Subsequently, because of the rising interest and financial support of target detection in recent years [14], detection by tracking has gradually led the research and application of multitarget detection. The current target tracking algorithm completes detection and tracking simultaneously, transforming the existing detector into a tracker.

2.2.1. Classic Algorithms. The improved classic MHT (Multiple Hypothesis Tracking) algorithm of the 1990s has achieved performance close to that of the most advanced methods in the standard benchmark dataset [15]. Using accurate object detectors simplifies a small number of possible assumptions. The appearance model can be learned efficiently using a regularized least-squares framework, and each assumed branch requires only several additional actions to take advantage of MHT when using higher-order information. JPDA (Joint Probabilistic Data Association) has also been improved to compute for high target and clutter density applications. Many experiments have also shown that when embedded in the simple tracking framework, the JPDA algorithm performs competitively, with the most advanced global tracking methods in both applications, while significantly reducing the processing time [16]. While MHT and JPDA retain their advantages, uncertainty and latency are high when they are faced with difficult targets. The composite complexity of these methods increases exponentially with the number of trace lines, making it impractical to apply such traditional methods in highly dynamic environments.

2.2.2. Multitarget Tracking. The latest research trend in multitarget tracking is the use of the same framework to include detection and tracking and the combination of the previous two-stage methods into a one-stage multitasking process. This technique uses the current and previous frames as input and then predicts the target frame offset position of the next frame. CenterTrack applies the detection model to a pair of images and detects from previous frames. With minimal input and the previous frame, CenterTrack can locate objects and predict their association. Simple, online, and live, CenterTrack can also be easily extended to single-eye 3D tracking by tracking other 3D attributes [17]. The detector based on the keypoint has shown satisfactory performance. However, incorrect key matching still commonly occurs and can seriously inhibit the performance of the detector. Compared with traditional embedding methods, CentripetalNet combines location information with match corner points more accurately. The corner pool extracts the information from the bounding box to the boundary. Feature matching of the star-shaped deformable convolution network endows the corner information with clarity. In addition, by equipping CentripetalNet with a mask prediction module, it can explore instance segmentation on an anchor-free detector [18].

2.2.3. Tracking by Detection. Mainstream target tracking algorithms are currently designed according to the detection and tracking mechanism. First, the target detection algorithm is used to detect the target in each frame and obtain the corresponding position coordinates, classification, confidence, and other data. Data association is then employed to correlate previous detection results with the detection results of the previous frame. However, when the target moves rapidly, traditional tracking by detecting matching fails. The tracking method based on trajectory prediction can successfully solve this problem. By adding a one-step Kalman filter to predict the tracking state of the next frame, the predicted target state is compared with the detected target to link the fast-moving objects.

Compared with detection and tracking combined into multitasking, the traditional tracking-by-detection tracking algorithm requires no high-cost video streaming data training and only needs to make several improvements on the basis of the existing target detector and thereby obtain robust and real-time compliance tracking.

The advantage of the algorithm used in this study is that it does not rely on the video streaming dataset to train the model. Only image datasets can be used to train effective target tracking models to realize real-time tracking of UAV models.

3. Detection and Tracking Method of UAV

This study primarily is aimed at achieving a simple and effective drone tracking method. In the target detector

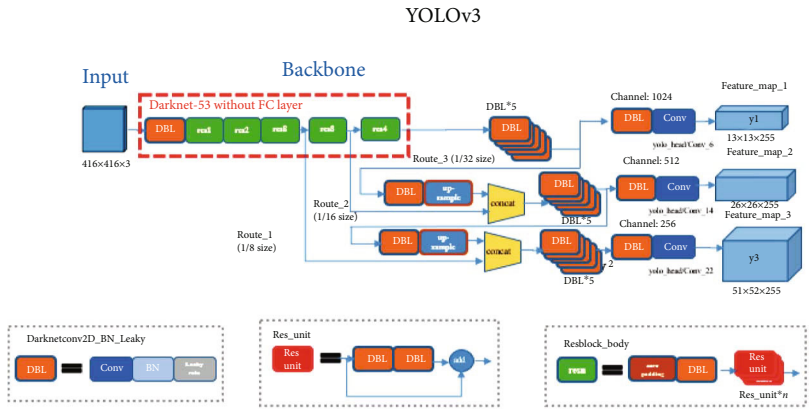

Figure 3: YOLOv3 network structure.

component, YOLOv3 [19] and CenterNet are selected to satisfy the speed and performance requirements of the target tracking model. As a representative of a single-stage network, YOLOv3 has no complicated network structure and can be well applied in industrial landing. As a representative of the new-generation anchor-free detection model, CenterNet can provide a good comparison with the algorithm under the anchor mechanism. The strategy in this study is to add data association between image frames on the existing target detector to achieve target tracking. Moreover, the most current detection and tracking methods require video streaming data as training data. However, the dataset entails considerably high costs. Consequently, we abandoned this tracking algorithm and turned to the mainstream tracking-by-detection tracking algorithm. This approach provides a clever strategy to complete the training of high-precision detection and tracking models with only image datasets.

3.1. Detector

3.1.1. Target Detector YOLOv3 and Its Improved Design. YOLOv3 is the third-generation version of the YOLO series. Compared with the two previous versions, the third-generation version integrates newly proposed techniques in target detection. The backbone partly draws on the structural design of the residual network. In the Darknet-53 [20] network, the neck part introduces the feature pyramid network (FPN) [21] structure. The superior techniques of the previous version are simultaneously retained, such as batch normalization (BN) [22] and k-means clustering algorithm [23]. The network structure of YOLOv3 is presented in Figure 3.

In YOLOv3, the network uses an improved backbone, upgrading Darknet-19, which was used by YOLOv2, to Darknet-53. Darknet-53 uses a 3×3 convolution kernel to replace the pooling layer, cuts the feature map, reduces the dimensionality, and performs a 1×1 convolution operation on the cut result to control the final output channel number. This approach not only reduces the amount of data brought by pooling and accelerates the calculation but also increases the nonlinearity and robustness of the network.

Recognizing objects at different scales is a basic challenge of Computer Vision, hence the proposal of the FPN structure. YOLOv3 introduces a structure similar to that of FPN to perform information fusion on detection frames with three different sizes. The FPN structure is aimed at using the inherent multiscale pyramid layer of the deep CNN to construct the feature pyramid, and its implementation combines low-level features with enhanced resolution and positioning information, together with high-level features with strong semantic information. This approach helps the network in detecting multiscale targets, particularly small ones. The information fusion of FPN includes three routes. The first route is a bottom-up pathway, which generates multiple feature maps of different layers through the forward propagation of the backbone CNN. For the feature pyramid, each stage is defined as a pyramid level. It is a top-down pathway, which is the core of the FPN structure, as shown in Figure 4. Its main function is the upsampling of the high-order pyramid layer to the same size as that of the secondary feature map. The lateral connection, which is mainly the high semantic feature map and high positioning information feature map obtained from the previous two roads, is combined by interpolation. Three sizes of prediction results—13×13, 26×26, and 52×52—are output to the network.

To accelerate the convergence speed during network training and prevent gradient explosion or disappearance in the backpropagation of the deep network, BN technology was integrated with YOLOv2 and retained in YOLOv3. In training the neural network, the input distribution of each layer of the network is always changing at each stochastic gradient descent for each minibatch. Owing to the sensitivity interval of the activation function being fixed, when continuous multilayer input distributions are present, the input distribution is in the activation function. In nonsensitive areas, network convergence tends to stall. To solve this problem, BN is added. The BN operation restricts the input distribution to a standard normal distribution with a value of 0 and a variance of 1 via a normalized operation; thus, the gradient change moves toward the optimal value of the loss function. This approach accelerates network convergence.

YOLOv3 is a representative of the anchor mechanism in the single-stage detection model. The method of setting set a suitable anchor is a finishing touch to enhance the detection performance of the model. In the anchor setting of YOLO, k

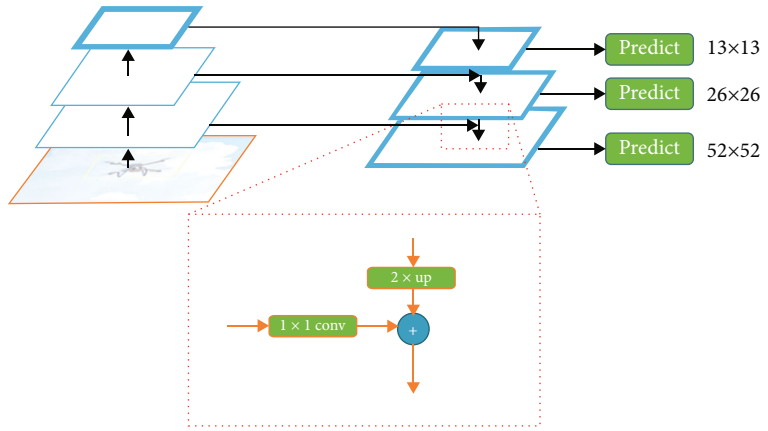

Figure 4: YOLOv3 feature pyramid network structure.

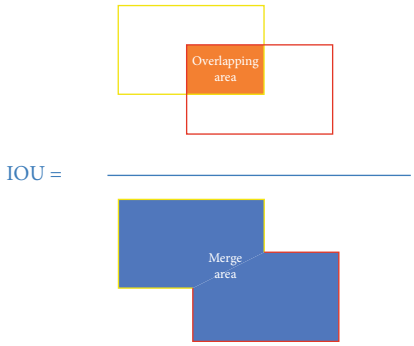

Figure 5: IOU calculation diagram.

-means clustering is still used for data processing to obtain the distribution setting of the anchor. Unlike that in Faster R-CNN, the anchor is manually generated through a preset scale and coordinate relationship. In YOLOv3, cluster analysis needs to be performed on the bounding boxes in the training set to generate appropriately sized anchors. In this study, we use a modified k-means clustering algorithm.

Performing standard k-means clustering on the anchor can induce the large box to generate a larger error during clustering, and this error is expected to be almost unrelated to the size of the box. The key to solving this problem is the measurement of the distance between different anchors. We use the maximum IOU (intersection over union) to replace the Euclidean distance measure in the standard k-means clustering algorithm. The IOU calculation is presented in Figure 5. The formula is as follows:

$$d(\text{box}, \text{centroid}) = 1 - \text{IOU}(\text{box}, \text{centroid}). \quad (1)$$

The brief process of k-means clustering for anchor boxes is summarized:

(i) The coordinates of all labeled ground truth frames in the training set are extracted

(ii) The coordinates of all ground truth (GT) boxes are converted into the height information of the boxes

(iii) K GT boxes are randomly selected as anchor boxes; with these k anchor boxes and the remaining GT boxes, IOU is determined and d = 1-IOU is calculated

(iv) The GT boxes are classified. The distance $\{d(i, 1), d(i, 2) \cdots d(i, k)\}$ of each GT box is compared with each anchor box, and the smallest distance d is selected. The GT box owned by the k anchor boxes is ultimately recorded

(v) The anchor box is updated. For each anchor box, the average value of the frame height of its GT box is calculated, and the value is used as the new size of the anchor box

(vi) Steps (i) to (iv) are repeated until the size of the anchor box no longer changes

To perform the k-means clustering operation on the anchor section, we set $k = 9$ because YOLOv3 has three feature maps. Each feature map requires three anchors of different scales. To perform k clustering training on the anchor data information in the training set and then ultimately obtain nine anchor scales, (36, 17), (54, 28), and (81, 40) are first used on the smallest 13×13 feature map. For the 26×26 feature map of the intermediate size, the three-scale anchors (123, 71), (156, 128), and (259, 111) are used, and (206, 172), (275, 217), and (380, 283) are prepared for the largest 52×52 feature map.

The SPP-Net [24] network is used to improve the YOLOv3 network. After the first layer feature map of YOLOv3, a structure similar to the SPP network is added. This addition is intended to effectively avoid the problems of image area clipping and image distortion caused by zoom operations and to solve the problem of CNN for image repeated feature extraction, which greatly improves the speed of generating candidate frames and reduces computational costs. Adding the SPP module to YOLOv3 strengthens the feature pyramid and semantic information and realizes the extraction of local and global features on the smallest feature map, which is favorable for addressing the problem of large differences in target size. The structure of YOLOv3-SPP is presented in Figure 6.

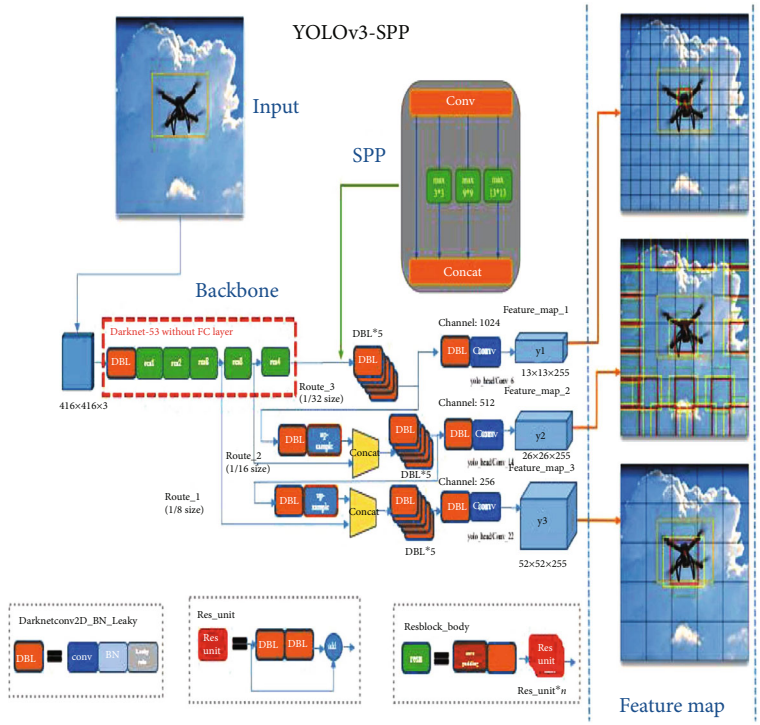

Figure 6: Modified YOLOv3-SPP network structure.

The loss function of the YOLO model must then be modified. The modified YOLO loss function mainly consists of box loss, confidence loss, and category loss:

$$\begin{aligned}\lambda = \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} T_{ij}^{\text{obj}} \left[(x_i - x_i)^2 + (y_i - y_i)^2 + (w_i - w_i)^2 \right. \\ \left. + (h_i - h_i)^2 \right] + \sum_{i=0}^{S^2} \sum_{j=0}^{B} T_{ij}^{\text{obj}} (C_i - C_i)^2 \\ + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} T_{ij}^{\text{obj}} (C_i - C_i)^2 \\ + \sum_{i=0}^{S^2} T_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - p\wedge_i(c))^2.\end{aligned} \tag{2}$$

T_i^{obj} defines whether there is a target in the i -th grid; T_{ij}^{obj} defines whether the j -th *a priori* box in the i -th grid is the *a priori* box responsible for target prediction; and λ_{coord} is the frame loss. The coefficient λ_{noobj} is the coefficient of no target confidence loss; s is the side length of the feature map; and B is the number of anchors for each grid. The original YOLO loss function divides the width and height of the detection frame with a root sign and then calculates the loss. The purpose is to reduce the proportion of the width and height loss occupied by the target frame size and prevent the coordinate loss of the central point from being overwhelmed. However, in the k mean value after clustering, the loss of the target frame size is reduced. Consequently, the square root operation is no longer performed on the loss of the target frame size. With this approach, the increased accuracy of anchor initialization is ensured, and the initial error exerts less effect and is easier to converge. The final loss category uses a cross-entropy loss function design.

3.1.2. Target Detector CenterNet and Its Improved Design. In recent years, the research on object detection has been redirected improving the detection accuracy of target detectors to focusing on the speed of the detector and then to the current weighting of the accuracy and real-time performance of the target detector. In this process, the two-stage detector method—for instance, the first stage of Faster R-CNN—proposes possible regions of interest (RoIs) via the region proposal network (RPN), combining the screening and calibration of RoIs. The first one is similar to the effect of coarse classification + fine classification. Owing to this design, Faster R-CNN achieves high detection accuracy, but simultaneously, the two-stage network runs at a low speed (5 FPS) and produces larger memory occupation, hindering its application in many real-world scenarios. The two-stage method is a slow process and thus has been reduced to a one-stage technique. The detection speed of one-stage detectors represented by SSD [25] and YOLO can generally be increased to more than 30 FPS, which basically meets the real-time requirements of detection and also achieves good detection accuracy. The most crucial problem at present is the poor detection efficiency of small target objects in multiscale detection tasks. Since the one-stage method does not have the same proposal stage as the RPN network layer, only a small part of the boxes can be selected as anchors when the anchor is preset, and few boxes match small objects. Many one-stage solutions have also been proposed in response to

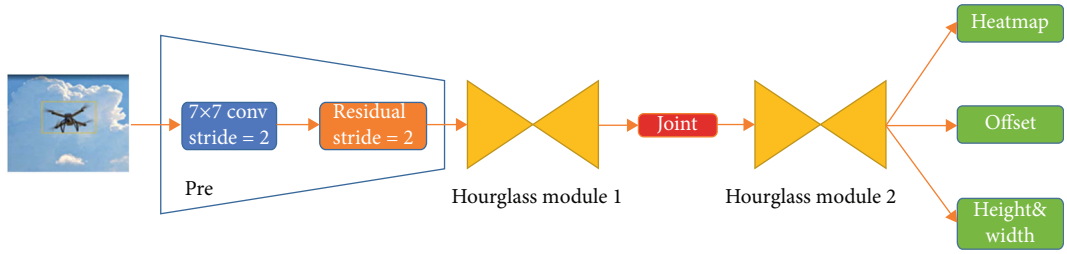

Figure 7: CenterNet network structure based on Hourglass backbone.

this situation. For instance, in the YOLO series, a k-means clustering algorithm is used for the dataset to generate a preset anchor. Moreover, an FPN-like structure is added to the train to enhance the positioning information of each layer, including semantic information. Since the development of target detectors, the anchor-based detection mechanism has gradually hindered the improvement of the detection performance. Therefore, a new method of solving object detection has emerged using the anchor-free technique.

The anchor-free method was originally proposed by CornerNet. Different from their previous target detector, the anchor is preset to predict the offset in order to complete the regression correction of the target frame. CornerNet directly converts the object detection problem into a keypoint detection problem. The network directly predicts the coordinates of the upper left point and the lower right point of the target frame and uses the embedding vector to match them and thereby complete the target detection. This approach abandons the anchor mechanism and no longer requires anchor setting; in addition, no subsequent screening and nonmaximum suppression (NMS) operations of a large number of anchors are needed. However, CornerNet also has deficiencies because some errors inevitably occur when the upper left corner and the lower right corner of the target box are matched, resulting in the reduced accuracy of the network. Moreover, the corner pooling it uses only relies on the edge information of the object and consequently uses no internal information, weakening its capability for global information acquisition. Therefore, we used CenterNet, a more advanced anchor-free detector (Figure 7).

CenterNet is further simplified in the idea of CornerNet. The network directly predicts whether each pixel is the target center; if so, it predicts the bounding box for the central point. This method is closely related to the anchor-based technique because each pixel in the feature map A pixel can be regarded as a shape-agnostic anchor. However, the "anchor" is only related to the position, and further predicting the offset and using NMS for postprocessing is unnecessary.

Among Figure 7, Pre is a 7×7 residual error unit with a step size of two. After this structure, the size of the picture is compressed to 1/4 of the original picture, and two hourglass modules perform keypoint detection. The final output has three branches: (i) Heatmap with the dimensions (W/4, H/4, C), which outputs the center point positions of the objects in C categories; (ii) Offset, with the dimensions (W/4, H/4, 2), which supplements and corrects the output result of Heatmap to improve the accuracy of positioning; and (iii) Height & Width, with the dimensions of (W/4, H/4, 2), which predicts the width and height of the detection frame centered on the keypoint.

The loss function of CenterNet also consists of three parts (target category loss, target center point offset loss, and target frame size loss):

$$L_{\text{det}} = L_k + \lambda_{\text{size}} L_{\text{size}} + \lambda_{\text{off}} L_{\text{off}}, \tag{3}$$

$$L_k = \frac{-1}{N} \sum_{xyc} \begin{cases} \left(\left(1 - \hat{Y}_{xyc} \right)^{\alpha} \log \left(\hat{Y}_{xyc} \right) \right), & \text{if } Y_{xyc} = 1, \\ \left(\left(1 - Y_{xyc} \right)^{\beta} \left(\hat{Y}_{xyc} \right)^{\alpha} \log \left(1 - \hat{Y}_{xyc} \right) \right), & \text{otherwise,} \end{cases} \tag{4}$$

$$L_{\text{off}} = \frac{1}{N} \sum_{p} \left| \hat{O}_{\tilde{p}} - \left(\frac{p}{R} - \tilde{p} \right) \right|, \tag{5}$$

$$L_{\text{size}} = \frac{1}{N} \sum_{p} \left| \hat{S}_{p_k} - s_k \right|. \tag{6}$$

L_k is the target category loss function. For a $W \times H$ picture $I \in R^{W \times H \times 3}$, the keypoint heatmap generated after going through the network is $\hat{Y}_{xyc} \in [0, 1]^{(W/R) \times (H/R) \times C}$; R is the output stride; C is the number of keypoints; and N is the number of keypoints in the heatmap. In the heatmap, $\hat{Y}_{xyc} = 1$ indicates that the point is the center point of the target; $\hat{Y}_{xyc} = 0$ indicates that the point is the background; Y_{xyc} is the labeled GT information; α and β are the hyperparameters of focal loss [26]; and the default settings are 2 and 4. The category loss function in this part draws on the idea of focal loss. The central point of the training weight in the easy example is appropriately reduced, which is the loss value. When $Y_{xyc} = 1$, $(1 - Y^{\wedge}_{xyc})^{\alpha}$acts as a correction function; if $\hat{Y}_{xyc}$ is close to 1, point detection is relatively easy, and $(1 - Y^{\wedge}_{xyc})^{\alpha}$is correspondingly lower. When $\hat{Y}_{xyc}$ is close to 0, the proportion of training should be increased because the center point has not been learned. Therefore, $(1 - Y^{\wedge}_{xyc})^{\alpha}$tends to be considerably large. The purpose is to balance the training process, that is, to solve the problem of imbalance between positive and negative samples. L_{off} is the offset loss of the center point, and $\hat{O}_{p^{\sim}} \in R^{(W/R) \times (H/R) \times C}$ is the local offset of each prediction center point, the target center point. The bias function must account for the accuracy loss of the input picture after the downsampling operation of the backbone network to render the prediction result closer to the target center point. In

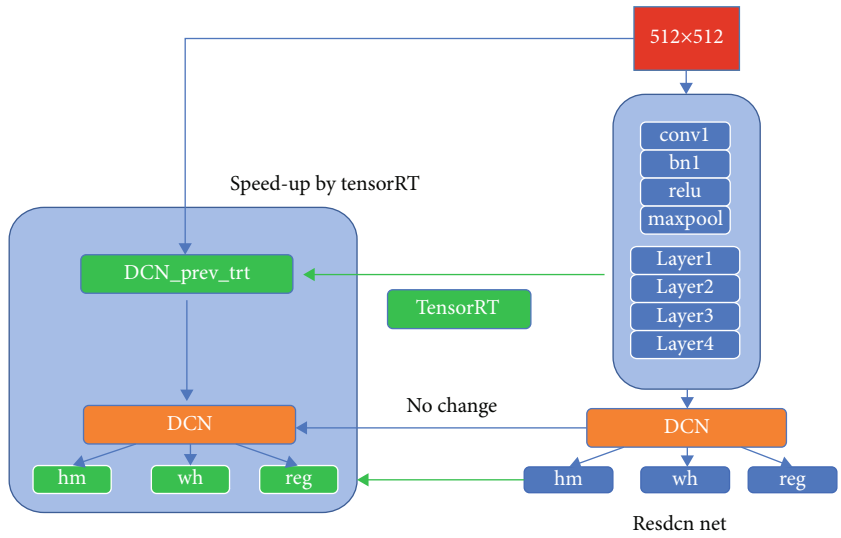

Figure 8: Split acceleration design.

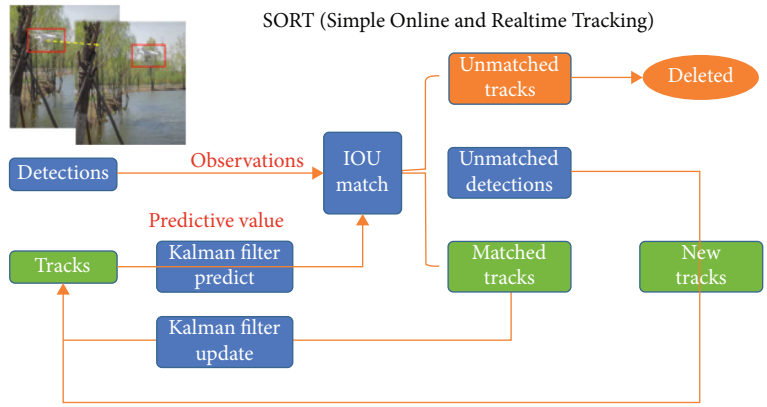

Figure 9: SORT algorithm flow.

this case, L_{size} is the loss function of the width and height of the target frame for the frame k with coordinates $(x_1^k, y_1^k, x_2^k, y_2^k)$. The size of the target frame $s_k = (x_2^k - x_1^k, y_2^k - y_1^k)$ and the loss function of the center point error are then designed by the loss function of L1 loss.

The backbone network of CenterNet consists of ResNet [27] and DCN [28]. However, upsampling convolutional layers, such as DCNv2, greatly limit the deployment of the model. To improve the performance of the CenterNet network and enable it to achieve real-time detection, we perform TensorRT acceleration processing on CenterNet. Limited by the design of the DCNv2 network, TensorRT does not currently support the acceleration operation of the DCN network. The network used appears in Figure 8.

The CenterNet network is separated to achieve TensorRT acceleration. The network is divided into two parts of the backbone network, ResNet and DCN, and ResNet is separately accelerated using TensorRT. This design can theoretically increase the running time of the backbone network by 10–20 times.

3.2. Tracker. After determining the detectors YOLOv3-SPP and CenterNet, we use Deep SORT [29] as the follow-up tracking algorithm. The Deep SORT algorithm is improved on the basis of the Simple Online and Realtime Tracking (SORT) algorithm [30].

The core of the SORT algorithm presented in Figure 9 consists of the Kalman filter and the Hungarian algorithm. The Kalman filter algorithm is divided into two processes: prediction and update. The algorithm defines the motion state of the target as eight normally distributed vectors. When the target starts to move, the position and speed of the target detection frame of the current frame are predicted from the target detection frame and target speed of the previous frame. This process describes the prediction approach of the Kalman filter. The update process of the Kalman filter is based on the predicted value of the previous frame and the observed value of the current frame (the predicted value and the observed value are in accordance with the normal distribution), which are linearly weighted to obtain the current prediction state of the system. The Hungarian algorithm then solves the matching

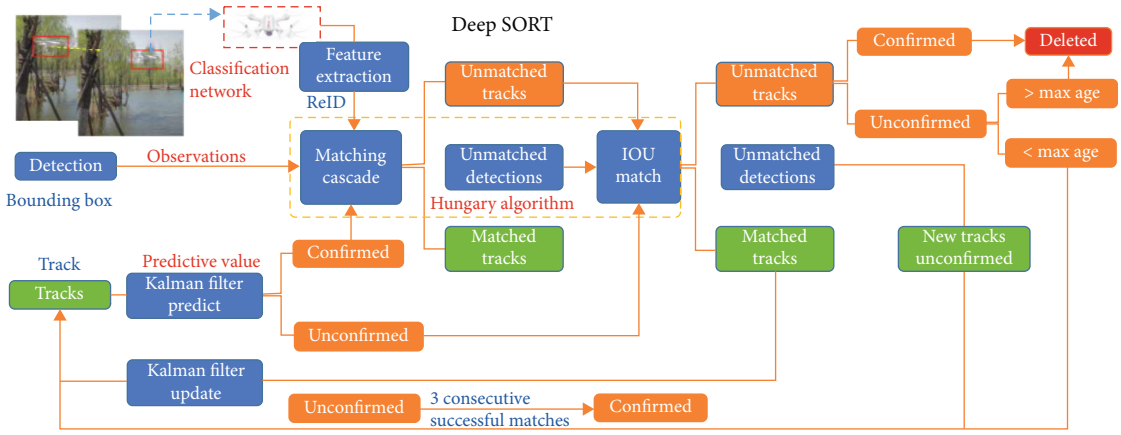

Figure 10: Deep SORT algorithm flow.

problem. After the similarity matrix, the Hungarian algorithm can solve the problem of matching the target of the two frames before and after the similarity matrix.

The most significant improvement of the Deep SORT algorithm over the SORT algorithm is the introduction of deep networks for appearance feature extraction and the use of models in pedestrian rerecognition for feature extraction. This operation also substantially reduces the amount of ID switching in the tracking algorithm. The Deep SORT algorithm can be divided into three steps: (i) predicting the trajectory tracks by Kalman filter; (ii) matching the predicted trajectory tracks with the detection frame in the current frame, including cascade matching and IOU matching, by using the Hungarian algorithm; and (iii) the third step which is to update the Kalman filter (Figure 10).

In matching the detection frame and the predicted trajectory, the situations shown in Figure 11 are expected to occur.

(i) Matched Tracks in which the detection frame and the track match: ordinary continuous tracking targets are classified under this situation, and the targets in the previous and subsequent frames exist and can be matched.

(ii) Unmatched Detections in which the detection box does not find a matching track: if the detector suddenly detects a new target in the image, the detection frame cannot find a matching target in the previous trajectory.

(iii) Unmatched Tracks in which the track does not match the detection frame: the continuously tracked target disappears from the video or flies out of the shooting range of the camera, and the predicted trajectory does not find the matching detection frame information.

(iv) Another situation occurring when two targets overlap: that is, when one target is occluded by another target, the trajectory of the occluded target cannot find a matching detection frame, and the target temporarily disappears from the image. When the occluded target reappears, the ID assigned by the occluded target should not change as much as possible and can be recognized by the algorithm as the target corresponding to the previous ID. This problem cannot be solved using the SORT algorithm. Thus, cascade matching in the Deep SORT algorithm is needed to solve it.

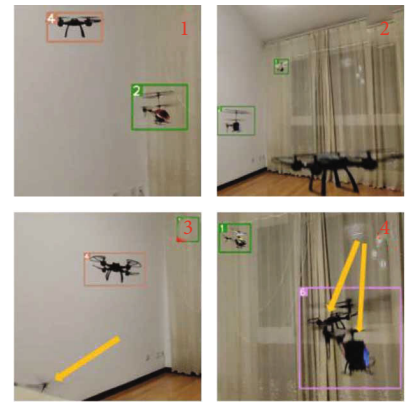

Figure 11: Four tracking situations.

4. Experiments

4.1. Dataset. We generated a drone target dataset because of the lack of a publicly available one. To ensure the diversity of data sources and process the drone pictures downloaded from the Internet, we also prepared four drones for shooting: two quad-rotor drones and two single-rotor drones. We obtained drone flight shots in both indoor and outdoor scenarios. The video was shot at 30 frames per second to prevent the pictures from appearing too similar between the data. One of 10 video frames was selected as the dataset, and 2459 pictures were obtained. We also crawled 341 pictures from the Internet as a supplement. The dataset composition is listed in Table 1.

After the photos were acquired, the corresponding datasets were built for the YOLO and CenterNet networks. The YOLO dataset had its format requirements, and the CenterNet network used the COCO dataset format.

Table 1: UAV dataset.

Black four-rotor	White four-rotor	Yellow single rotor	Red single rotor	Total
823	678	500	799	2800

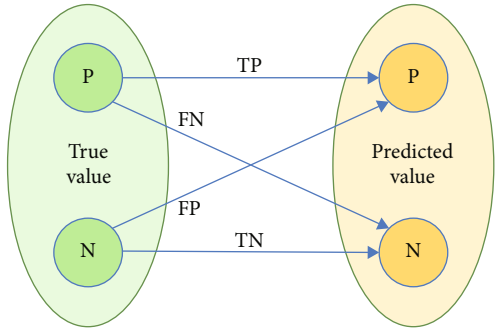

Figure 12: Four predicted results.

In the YOLO dataset format, a picture corresponds to a text annotation file, which contains the category and coordinate information of all target frames in the corresponding picture. The YOLO format requires that the coordinate values of all target frames be normalized. The conversion formula is as follows:

$$\begin{aligned} x' &= \frac{(x_{\min} + x_{\max})}{2W}, \\ y' &= \frac{(y_{\min} + y_{\max})}{2H}, \\ w' &= \frac{w}{W}, \\ h' &= \frac{h}{H}, \end{aligned} \tag{7}$$

where W and H are the width and height of the picture; x' and y' are the coordinates of the center point of the target frame, respectively; and w' and h' are the width and height of the target frame, respectively, and are normalized based on the width and height of the picture operation.

The COCO dataset format requires the coordinates of the upper left corner of the target frame and the width and height of the target frame. They are converted using the following formula:

$$\begin{aligned} x' &= x_{\min}, \\ y' &= y_{\min}, \\ w' &= x_{\max} - x_{\min}, \\ h' &= y_{\max} - y_{\min}, \end{aligned} \tag{8}$$

where x' and y' are the coordinates of the center point of the target frame and w' and h' are the width and height of the target frame, respectively.

Figure 13: Image data splicing.

After obtaining the labeled data files, we divided the dataset into two parts with a train set : test set ratio of 9 : 1. The training set contained 2520 images, and the test set contained 280 images.

4.2. Experimental Performance Index

4.2.1. Target Detection Index

(i) True positives (TP): the true value is a positive example, and the predicted value is a positive example.

(ii) True negatives (TN): the true value is a negative example, and the predicted value is a negative example.

(iii) False positives (FP): the true value is a negative example, and the predicted value is a positive example.

(iv) False negatives (FN): the true value is a positive example, and the predicted value is a negative example.

As shown in Figure 12, the main indicators used are as follows:

(i) Precision, the proportion of TP in the recognition result

$$\text{Precision} = \frac{\text{tp}}{\text{tp} + \text{fp}} \tag{9}$$

(ii) Recall, where TP accounts for the proportion of all positive samples in the dataset

$$\text{Recall} = \frac{\text{tp}}{\text{tp} + \text{fn}} \tag{10}$$

(iii) AP, where the size of the area is enclosed by the precision-recall curve

(iv) Mean (mAP), the average of multiple categories of AP

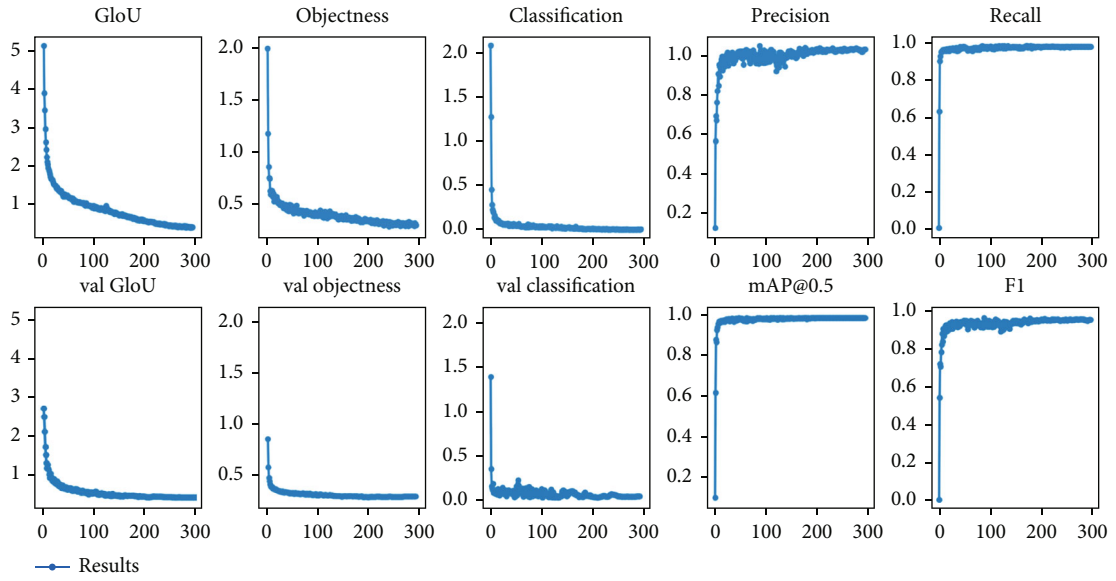

Figure 14: YOLOv3 loss curves.

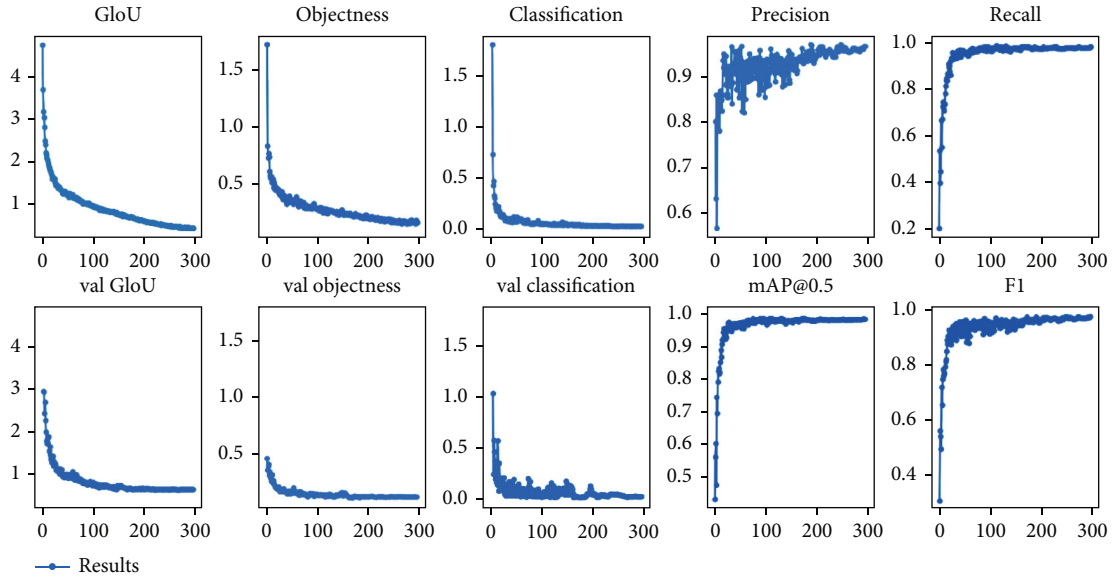

Figure 15: YOLOv3-SPP loss curves.

4.2.2. Target Tracking Indicator. To measure the target tracking algorithm performance, the following are the main indicators:

(i) ID switch (IDSW), target ID switch total

(ii) Fragmentation (FM), the total number of interrupted target tracking

(iii) Multiple object tracking accuracy (MOTA):

$$\mathrm{MOTA} = 1 - \frac{\sum_t (\mathrm{FN} + \mathrm{FP} + \mathrm{IDSW})}{\sum_t \mathrm{GT}} \tag{11}$$

where GT is the ground true box of each frame.

4.2.3. Training Result

(1) Detection Results. YOLOv3, YOLOv3-SPP, and CenterNet algorithms were used to train the previously constructed drone dataset. The training machine environment was the i7-9700K CPU and the single card 1080 Ti GPU. The deep learning framework used by YOLOv3 was Pytorch1.4.0, and that by CenterNet was PyTorch 1.2.0. YOLOv3 and YOLOv3-SPP used the same training parameters. A batch size of 8300 epochs was trained, and CenterNet was trained for 150 epochs with a batch size of 16. By using this approach, the training volume of the three models was identical. Moreover, in the training process of YOLOv3, a data enhancement operation of randomly splicing data pictures was used (Figure 13), which could achieve multiscale training of the

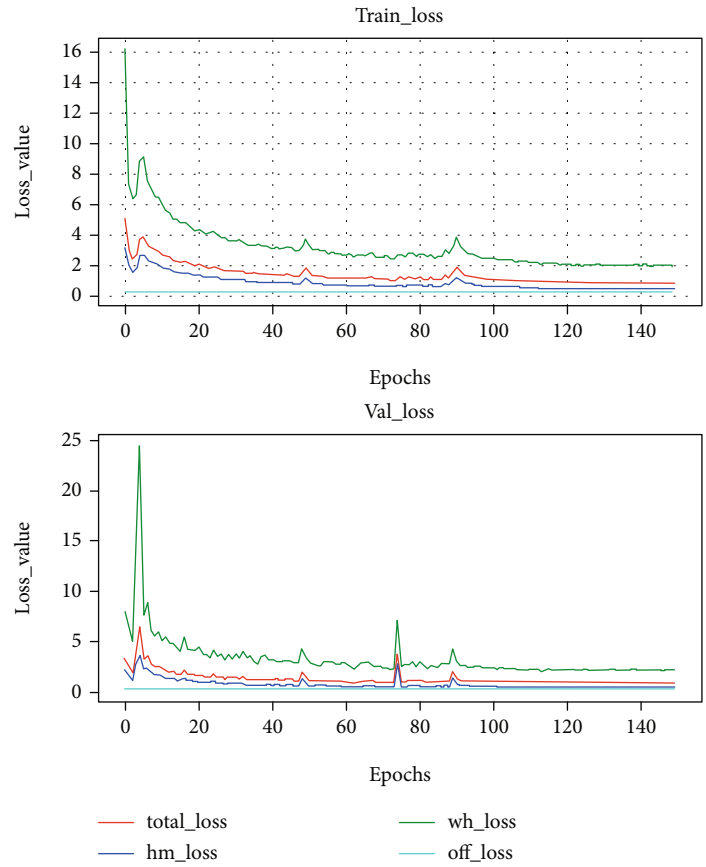

Figure 16: CenterNet loss curves.

Table 2: A comparison of model training time and detection performance.

Algorithm model	YOLOv3	YOLOv3-SPP	CenterNet-DLA34
Training time	12.5 h	9 h	8.5 h
mAP (IOU = 0.5)	0.988	0.993	0.958
Precision	0.971	0.982	0.961
Recall	0.962	0.973	0.942
Detection speed	69.5 FPS	69 FPS	23 FPS

target and also simulate the appearance of multiple targets in the same picture.

Figures 14–16 present the loss function curves during training of the three models:

All loss curves presented in Figure 16 meet the training expectations. Comparison and observation of the data in Table 2 reveal that the performances of YOLOv3 and YOLOv3-SPP do not considerably vary, although that of YOLOv3-SPP is slightly better. The speed of CenterNet can only reach one-third than that of YOLO, and the accuracy is lower than that of the YOLO model. Compared with YOLOv3, CenterNet eliminates the anchor setting and NMS operation calculation, and the final output feature map of CenterNet has only one heatmap layer. Therefore, CenterNet should theoretically exhibit superior real-time performance than the YOLO series. However, CenterNet only needs one layer of feature maps; for such a layer to obtain sufficient feature information, a larger backbone network and a more complex feature fusion neck layer need to be used to extract features. Thus, the amount of calculation is not comparable to the YOLOv3 series. In summary, the single-stage detection model YOLOv3 under the anchor mechanism is evidently superior to CenterNet under the anchor-free mechanism on our UAV dataset.

(2) Tracking Algorithm Experiment Results. The core of Deep SORT matching tracking is to extract effective image feature information by using deep networks. The existing deep network is a suitable network model for pedestrian rerecognition. This model was originally a 751 classification network. The previous training data used 751 images of different positions and different angles as the training set for classification model training. The remaining problem was pedestrian reidentification.

In order to apply the algorithm to our drone information feature extraction, we used the detector to collect the drone image data, crop the image part in the detection frame, and classify and store it according to the categories of the four drones. We then used the previous pedestrian rerecognition model for classification training. The model obtained by

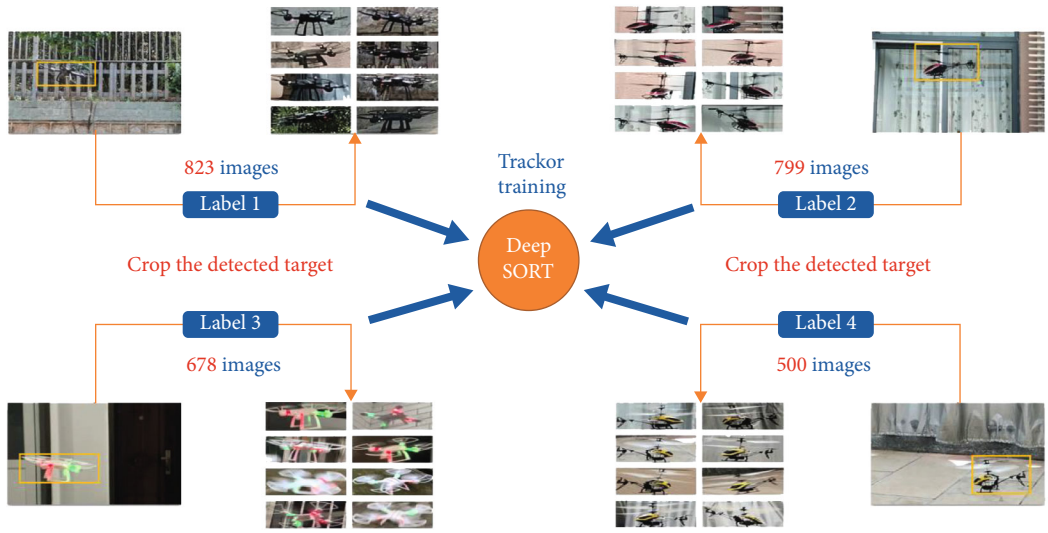

Figure 17: Generating data for Deep SORT.

Figure 18: Target tracking results in IoT.

Table 3: Target tracking performance.

Tracking model	YOLOv3-SPP + Deep SORT	CenterNet + Deep SORT
FP	0	0
FN	85	503
ID switch	4	31
FM	13	28
GT	1591	1591
MOTA	0.94406	0.66436
Speed (FPS)	54	25

classification training was employed as the feature extraction model of our tracking algorithm. The operation is illustrated in Figure 17.

Finally, the two detectors were combined with the tracker to obtain the following tracking results. The left picture in Figure 18 is the tracking effect of YOLOv3-spp + Deep SORT, and the right picture is the tracking effect of CenterNet + Deep SORT.

For the same online drone video tracking, the results are obtained as shown in Table 3.

5. Conclusion

In this study, we introduce a new method, referred to as tracking-by-detection, in tracking UAV targets. The approach can achieve real-time high-precision tracking. This method used YOLOv3 as the detector and Deep SORT as the tracking mode to achieve a detection speed of 54 FPS and MOTA reaching 94.4%, which meets the requirements of real-time tracking of multiple targets for drones. On this basis, we also modify the YOLOv3 network by changing the loss function of the model in accordance with the characteristics of the drone target and adding the SPP module to collect the drone data to generate the initial anchor. These operations improve the MOTA of the modified YOLOv3-SPP network by 2% with high detection speed, which is only 5 FPS. For comparison, we also attempt tracking UAV targets by using the anchor-free mode. This mode is currently widely used in target tracking, with CenterNet as the target detector and Deep SORT as the tracker. The final detection speed of this model is 25 FPS, and the MOTA is 66.4%, only slightly enhancing real-time tracking. The experimental results indicate the effectiveness of the proposed approach and confirm that YOLOv3-SPP + Deep SORT is highly applicable for multitarget real-time tracking of UAVs.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declares that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (Project No. 2018YFB0505100). The work by Tao Hong is supported by the National Natural Science Foundation of China under Grant No. 61827901.

References

[1] A. Fotouhi, M. Ding, and M. Hassan, "Understanding autonomous drone maneuverability for Internet of Things applications," in *2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 1–6, Macau, 2017.

[2] F. Qi, X. Zhu, G. Mang, M. Kadoch, and W. Li, "UAV network and IoT in the sky for future smart cities," *IEEE Network*, vol. 33, no. 2, pp. 96–101, 2019.

[3] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile unmanned aerial vehicles (UAVs) for energy-efficient internet of things communications," *IEEE Transactions on Wireless Communications*, vol. 16, no. 11, pp. 7574–7589, 2017.

[4] B. S. Oh, X. Guo, and Z. Lin, "A UAV classification system based on FMCW radar micro-Doppler signature analysis," *Expert Systems with Application*, vol. 132, pp. 239–255, 2019.

[5] D. Han, D. Y. Gwak, and S. Lee, "Noise prediction of multirotor UAV by RPM fluctuation correction method," *Journal of Mechanical Science and Technology*, vol. 34, no. 4, pp. 1429–1443, 2020.

[6] J. Wang, K. Chen, S. Yang, C. Loy, and D. Lin, "Region proposal by guided anchoring," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2965–2974, 2019.

[7] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "Yolov4: optimal speed and accuracy of object detection," 2020, https://arxiv.org/abs/2004.10934/.

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[10] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," 2019, https://arxiv.org/abs/1904.07850/.

[11] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, "Multiple hypothesis tracking revisited," in *IEEE International Conference on Computer Vision (ICCV)*, pp. 4696–4704, 2015.

[12] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, "Tracking without bells and whistles," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 941–951, 2019.

[13] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Detect to track and track to detect," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3057–3065, 2017.

[14] S. H. Rezatofighi, A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid, "Joint probabilistic data association revisited," in *Proceedings of the IEEE international conference on computer vision*, pp. 3047–3055, 2015.

[15] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking Objects as Points," in *European Conference on Computer Vision*, pp. 474–490, 2020.

[16] Z. Dong, G. Li, Y. Liao, F. Wang, P. Ren, and C. Qian, "Centripetalnet: Pursuing high-quality keypoint pairs for object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10519–10528, 2020.

[17] J. Shi, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pp. 593–600, Seattle, WA, 1994.

[18] C. Tomasi and T. Kanade, *Detection and tracking of point features. Technical Report CMUCS-91-132*, Carnegie Mellon University, 1991.

[19] J. Redmon and A. Farhadi, "Yolov3: an incremental improvement," 2018, http://arxiv.org/abs/1804.02767.

[20] Q. C. Mao, H. M. Sun, Y. B. Liu, and R. S. Jia, "Mini-YOLOv3: real-time object detector for embedded applications," *IEEE Access*, vol. 7, pp. 133529–133538, 2019.

[21] R. A. Sturm, D. L. Duffy, Z. Z. Zhao et al., "A single SNP in an evolutionary conserved region within intron 86 of the HERC2 gene determines human blue-brown eye color," *The American Journal of Human Genetics*, vol. 82, no. 2, pp. 424–431, 2008.

[22] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pp. 448–456, 2015.

[23] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1no. 14, pp. 281–297, 1967.

[24] 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*, vol. 37, no. 9, pp. 1904–1916, 2015.

[25] W. Liu, D. Anguelov, D. Erhan et al., "SSD: Single shot multibox detector," in *European Conference on Computer Vision*, pp. 21–37, 2016.

[26] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

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

[28] J. Dai, H. Qi, Y. Xiong et al., "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 764–773, 2017.

[29] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*, pp. 3645–3649, 2017.

[30] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE international conference on image processing (ICIP)*, pp. 3464–3468, 2016.