Received 16 May 2023, accepted 1 June 2023, date of publication 6 June 2023, date of current version 5 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3283411

RESEARCH ARTICLE

Toward Real-Time UAV Multi-Target Tracking Using Joint Detection and Tracking

TINNAKORN KEAWBOONTAN[1] **AND MASON THAMMAWICHAI**[2], **(Member, IEEE)**
[1]Graduate School, Navaminda Kasatriyadhiraj Royal Air Force Academy, Bangkok 10220, Thailand
[2]Electrical Engineering Department, Navaminda Kasatriyadhiraj Royal Air Force Academy, Bangkok 10220, Thailand

Corresponding author: Mason Thammawichai (mason@rtaf.mi.th)

This work was supported by the Royal Thai Air Force under Grant 65186-81100181-001.

ABSTRACT Multiple object tracking (MOT) of unmanned aerial vehicle (UAV) systems is essential for both defense and civilian applications. As drone technology moves towards real-time, conventional tracking algorithms cannot be directly applied to UAV videos due to limited computational resources and the unstable movements of UAVs in dynamic environments. These challenges lead to blurry video frames, object occlusion, scale changes, and biased data distribution of object classes and samples, resulting in poor tracking accuracy for non-representative classes. Therefore, in this study, we present a deep learning multiple object tracking model for UAV aerial videos to achieve real-time performance. Our approach combines detection and tracking methods using adjacent frame pairs as inputs with shared features to reduce computational time. We also employed a multi-loss function to address the imbalance between the challenging classes and samples. To associate objects between frames, a dual regression bounding box method that considers the center distance of objects rather than just their areas was adopted. This enables the proposed model to perform object ID verification and movement forecasting via single regression. In addition, our model can perform online tracking by predicting the position of an object within the next video frame. By exploiting both low- and high-quality detection techniques to locate the same object across frames, more accurate tracking of objects within the video is attained. The proposed method achieved real-time tracking with a running time of 77 frames per second. The testing results have demonstrated that our approach outperformed the state-of-the-art on the VisDrone2019 test-dev dataset for all ten object categories. In particular, the multiple object tracking accuracy (MOTA) score and the F1 score both increased in comparison to earlier work by 8.7 and 5.3 percent, respectively.

INDEX TERMS Multi-object detection, UAV, deep learning, target tracking.

I. INTRODUCTION

The advancement in computer vision and machine learning over the past decade has brought benefits to all cyber-physical systems, including unmanned aerial vehicles (UAVs). UAVs have been extensively used in various applications, i.e., reconnaissance, search and rescue, target recognition, security surveillance, monitoring, and inspection. One of the most attractive problems within the UAV research community is object detection and target tracking from UAV images/videos. Multiple object tracking (MOT) tasks involve identifying and classifying objects in a sequence of images without prior knowledge of the targets. These objects can be of various types, such as pedestrians, vehicles, or animals. To enable detection correlation over time, each object must also be assigned a tracking ID. However, conventional multi-object tracking algorithms have shown to be ineffective in handling UAV videos due to various factors such as irregular target movements, camera movement, and perspective shifting in 3D. UAV video datasets typically contain several types of moving objects, and the number of objects in each category is often highly unbalanced, making it challenging to train the detection model. Furthermore, most objects in UAV videos are tiny due to the altitude of the flight, which can further complicate the detection task. The challenge of object tracking lies in the inconsistent appearance and movement

The associate editor coordinating the review of this manuscript and approving it for publication was Bing Li.

data of the target, which are caused by abnormal and rapid camera movements. Resulting in an alternate assignment of the object movement ID, which is more intricate than the traditional MOT. Additionally, due to the complexity of the network and the massive volume of data, it is difficult to increase the speed of MOT in practice. These issues lead to missing and false detections, resulting in poor tracking performance as well as real-time constraints on the UAV. Several algorithms have been proposed to solve multi-target tracking in complex environments, including the Kalman filter [1], [2], particle filters [3], [4], multi-hypothesis tracking (MHT) [5], [6], joint probabilistic data association filter (JPDAF) [7], [8], [9], graph-based multi-object tracking (GBMOT) [10], [11], and random finite set (RFS) filtering [12]. These algorithms are designed to estimate the position of multiple targets in difficult environments, using techniques such as non-linear motion models, probability-based estimation, and graph theory. However, to enhance the accuracy and robustness of UAV multi-target tracking, a hybrid approach that combines multiple tracking algorithms has been proposed [13]. A data-driven approach is another method used to address the MOT problem, in which object properties (features) are extracted from the dataset. Deep learning-based features are extracted through a hierarchical process of neural network layers that learn representations of the input data. These features are learned automatically from the data without requiring human intervention and are often highly effective in tasks such as image recognition and natural language processing. In contrast, handcrafted features are designed manually by human experts and are often based on domain-specific knowledge. The handcrafted features can be effective in certain applications, but they require significant expertise and effort to design and may not generalize well to new tasks [14]. The architecture can be either a two-step or one-step process. In a two-step process, objects are first detected in each frame of a video and then linked to the detected objects in the previous frame, which is referred to as tracking by detection (TBD). In contrast, a one-step process combines detection and appearance feature learning, which is known as joint detection and tracking (JDT) [15]. The suitability of either approach for real-time tracking depends on the specific requirements of the application, with the JDT approach being potentially faster owing to a single model, whereas the TBD approach is potentially more accurate and efficient with a high-quality detector and tracker.

Therefore, in this study, we apply deep learning (DL) techniques to address the issue of UAV multi-target tracking. Specifically, we propose an enhanced "Multi-target Tracking using Joint Detection and Tracking (MTTJDT)" model. Our methods employ object detection, feature extraction, and joint detection and tracking of the linkage between adjacent frames to simultaneously detect and track the objects. The aim of this research is to develop a UAV multi-target tracking model that is suitable for real-time applications. The main contributions of this study are as follows:

1) We used the joint detection and tracking method, which uses adjacent frame pairs and shared features to reduce the computational time and enable real-time performance. This method is suitable for resource-constrained systems, while leveraging the temporal information provided by adjacent frame pairs to better resolve occlusions and appearance changes. Our method can process data at a rate of 77 frames per second (fps), making it suitable for real-time applications.
2) We propose a multi-loss function that consists of a combination of classifications using the focal loss function and localization loss employing the CIoU loss function, with a loss-scale parameter used to balance the two functions. This approach enables the prioritization of specific factors, such as object type or position, while also accounting for imbalances in challenging classes and samples.
3) A dual-regression bounding box was proposed to associate objects between adjacent frames by considering the distance between their centers. This allowed us to simultaneously verify the IDs and predict the movements of objects using a single regression.
4) To improve the performance of object re-identification (re-ID) in video tracking, we employed detection techniques that considered objects with both high and low detection scores according to their predicted trajectories. This approach allows for more comprehensive and accurate tracking of objects within a video.

The remainder of this paper is organized as follows. Section II reviews closely related work in the literature. The proposed network architecture is explained in Section III. The performance metrics and model training are described in Section IV. Section V demonstrates the experimental results and discussion. The conclusions and future work are presented in Section VI.

II. RELATED WORK

A. OBJECT DETECTION

Object detection is a fundamental problem in computer graphics for predicting the position and class of an object. Determining the position of an object in an image captured by a CCTV camera or UAV is challenging. Identifying multiple objects in a class with distinct features, particularly small objects where the majority of images have a large computational area, is not trivial. Object detection is typically categorized into two approaches: anchor-based and anchor-free.

Anchor-based detection utilizes a predefined Convolutional Neural Network (CNN) [16] to transform an image into a feature map and a sliding window to traverse the map. The point at the center of each sliding window frame is known as the anchor point. Reference [17] proposed a well-known two-stage anchor-based detection algorithm called R-CNN: Regions with CNN features. This method uses region proposal algorithms to generate numerous instances for CNN-based classification. In [18], a fast region-based convolutional network method (Fast R-CNN) was proposed in which a region-of-interest (RoI) aggregation layer was added to the CNN. This method significantly improved the training and testing times. By sharing all the CNN core calculations, Faster R-CNN [19] specifies a Region Proposal Network (RPN) for proposal generation and core recalculation, allowing the model to be trained without prior data. Reference [20] presented a Feature Pyramid Network (FPN) architecture to exploit the multi-scale and pyramidal hierarchies of deep convolutional networks. This method uses top-down architecture with lateral connections to build high-level semantic feature maps at all scales at an extra computational cost.

Another approach is a one-stage detection architecture [21], [22], [23], which does not utilize the regional proposal method. The idea is to learn the regressed bounding-box coordinates and class probabilities directly from image pixels. Therefore, they could significantly accelerate the detection process. Reference [24] proposed a single deep neural network called the Single-Shot MultiBox Detector (SSD), which employs multiple aspect ratios and scales per feature-map location. The algorithm creates many convolutional layers of diminishing size, in which each level has a set of anchor boxes for each cell. The model then identifies and predicts the suitable anchor boxes. This approach is suitable for detecting objects of different sizes and minimizing the computational time.

Unlike the anchor-based approach, anchor-free detection, such as FCOS [25] and FoveaBox [26], can eliminate intricate calculations involving anchor boxes, such as calculating intersection over union (IoU) and matching between the anchor box and ground-truth bounding box during training, and may alleviate the aspect ratio issue and the need for a large number of anchor boxes. It is possible to reduce the difficulty in identifying objects posed by the presence of diversely shaped objects, particularly small ones. Generally, an anchor-free CNN generates a crude feature map to indicate the position of an object. Reference [27] proposed a single convolution neural network, CornerNet, to detect objects as paired keypoints. Extremenet [28] applied position estimation to locate the center of an object. To improve both precision and recall, CenterNet [29] uses the information of the object's center as another reference point.

B. MULTI-OBJECT TRACKING

There are a number of works in the literature that address the MOT problem. However, object detection and tracking in real-time are difficult. Both algorithm complexity and processing time are trade-offs. The tracking process relies on data association. The similarity indicators are based on the location, motion, and features of an object. Additionally, the dynamic nature of the environment, including the varying number of goals between frames, occlusions, and other complex conditions, further complicates the MOT problem [30], [31]. Object detection linkages in video sequences can be classified as online or offline linkages. Because the online method considers information from the previous and current frames, it is ideal for real-time applications. The offline method consists of a predetermined set of frames. Data association challenges arise when the target changes its aspect ratio; there are sudden changes in the appearance of objects, occlusion occurs, and rapid object movement occurs. Some objects may enter or leave the scene. As previously stated, the two common schemes for solving the MOT problem are the TBD and JDT approaches.

In the TBD approach, objects can be tracked easily by combining the detection models with object associations. The object bounding outputs are correlated across frames to form a tracking trajectory. The advantage of this approach is that it allows for the selection of resilient detectors with sufficient frame rates. However, for this approach to be effective, the object positions between two successive frames should not fluctuate significantly. Reference [32] presented an IOU-Tracker that uses object overlap (IoU) over time to determine which tracks to link. For an effective tracking, the target object must be clearly visible. Reference [33] proposes utilizing the integration of five Siamese networks to generate similarity scores for the objects being tracked. The final trait score is generated by the data-binding function to create an effective tracer for the subject in the thermal image. Another method for resolving the association problem is to anticipate the trajectory of all the object locations in the current frame using data collected from previous frames. The Kalman filter [34] is often used to estimate the trajectory of an object. SORT [35] uses the Kalman filter and Hungarian algorithm to predict the location of the bounding box and associate it with the current detection with the greatest IoU value. Deep SORT [36] integrates a deep associate metric in the visual appearance space to track objects in an occlusion. Reference [37] proposed a tracker, called the good old Hungarian simple tracker (GHOST), which utilizes both appearance and movement cues to enhance the accuracy of re-ID models. Reference [38] proposed FineTrack, a precise representation tracker that utilizes flow alignment FPN (FAFPN) and multi-head part mask generator (MPMG). FAFPN aligns and aggregates feature maps, while MPMG generates fine part masks for accurate global-local appearance embedding representation. Additionally, shuffle-group sampling (SGS) is introduced to balance training data and improve model performance. Reference [39] proposed a method named Triplet, which combines tracking information and visual features to accurately differentiate objects that share similar appearances. By employing an attention-based re-identification model, this approach significantly enhances the process of associating objects based on their distinctive visual characteristics. Tracking by detection may fail in difficult situations such as crowded scenes, abrupt object movement, or camera motion. Tracking performance also depends on the efficiency of the detectors. The processing time is often slow because it comprises two distinct tasks. Therefore, frame rates for real-time applications are difficult to obtain.

The joint detection and tracking approach integrates object detection and tracking into a single framework. There are two main approaches for solving this problem. The first method is joint detection and re-ID, which embeds the re-ID feature into the NMS and uses the re-ID feature similarity to relate it to an object. Reference [40] proposed a general framework for object instance segmentation called Mask R-CNN. Reference [41] proposed a multi-object tracking and segmentation (MOTS) algorithm that takes advantage of temporally consistent mask-based tracking information. The authors in [52] proposed a joint detection and embedding model that integrates the detector and embedding model into a single network to save on computation and increase efficiency. Reference [43] presents an approach to object detection and tracking by utilizing an anchor-based detection model that is trained in a combined manner, known as RetinaTrack. Reference [48] presented a FairMOT model that consists of two homogeneous branches capable of predicting pixel-wise objectness scores and re-ID features to achieve fairness between tasks. Reference [51] presented a method for multi-object tracking of pedestrians that extends the existing FairMOT approach by leveraging HRNet32 and incorporating the Polarized Self-Attention mechanism to enhance person re-identification. Additionally, the ReID branch is trained using circle loss, a technique that effectively improves the discrimination of features. The second approach is joint detection and motion prediction, which jointly detects and predicts the movement of objects. Examples of this scheme can be found in [44], [53], [54], and [55]. Tracktor [42] used bounding box regression of an object detector to predict the position of the target in the next frame. CenterTrack [44] uses a pair of images as inputs and predicts their associations with the previous frame. Reference [45] proposed a dual regression tracking framework for object tracking using a classification-based convolutional neural network (CNN) and a correlation filter (CF) module to improve tracking performance and reduce computational costs. Reference [49] demonstrated the effectiveness of using a dual-level deep representation model for thermal IR tracking and highlighted the importance of considering both image and motion information for accurate tracking. Reference [46] discussed a new depth correlation tracker called the self-supervised deep correlation tracker (self-SDCT), which uses a pre-trained feature extraction network and self-supervised learning to improve the tracking performance despite the limitations in labeled training data. Reference [47] proposed a chained-tracker (CTracker), which is an end-to-end model that adopts a chained structure and pairs attentive regression. Reference [50] presented a simple generic association model called ByteTrack. By including low-score detection boxes, it can improve the IDF1 score.

A summary of several multiple object detection and tracking models reported in the literature is provided in Table 1. In this study, we employ a joint detection and tracking approach. However, two adjacent frames were fed into the model instead of one frame. We integrated ResNet-18 with an FPN to isolate the current frame features. The temporal features are shared with the next frame pair. Our model employs the FCOS method, which is an enhanced anchor-free detector capable of analyzing two frames simultaneously. For target tracking, we applied an improved CTracker to determine the same object's position in both frames by employing the CIoU technique to store uncertain or missing objects from frames with a defined number of frame buffers. Object tracking trajectories are generated using the ByteTrack algorithm instead of matching the same object between frame pairs. The details of the proposed framework are presented in the next section.

III. METHODOLOGY

A. DATASET OVERVIEW

This study used a multi-object tracking dataset from VisDrone2019 [56]. The dataset included training, validation, and test sets. The dataset consists of video sequences, each of which contains images accompanied by annotations in the form of text files (.txt) for each object within a frame. The annotations included the frame number, object ID, object class, bounding box coordinates, and other features. The bounding box coordinates of the object are (x, y, w, h), where x and y are the coordinates of the upper-left corner of the bounding box and w and h are the width and height of the bounding box, respectively. Each bounding box was assigned a unique ID that was used to track the objects in different frames of the video. In addition to the border box and ID tag, the dataset also contains additional information, such as the object type, consisting of ten categories: pedestrians, people, cars, vans, buses, trucks, motors, bicycles, awning-tricycles, and tricycles, and the visibility of the object (e.g., fully visible, partially visible, or occluded). This information helps the model better understand the context of objects within the frame and improves its ability to detect and track objects in real-world situations. These sequences present several challenges, such as viewpoint, scale variations, and motion blur. Unlike CCTV with a fixed view, UAV cameras are captured from arbitrary angles [57]. Furthermore, because UAV cameras view objects from varying heights, the scales of the objects in an image vary. A camera mounted on the UAV records the video as it moves, resulting in significant motion blur in the recorded video [58].

B. ARCHITECTURE OVERVIEW

In this section, we present our multi-target tracking using the joint detection and tracking (MTTJDT) model designed for use in UAV multi-target tracking tasks. The MTTJDT model builds upon the concepts and methods of the CTracker model,

TABLE 1. Summary of multiple object detection and tracking models.

Model	Approach	Method	Year
R-CNN [17]	Anchor-based	CNN-based classification with selective search techniques	2014
Fast R-CNN [18]	Anchor-based	CNN with regional features	2015
Faster R-CNN [19]	Anchor-based	Region proposal network with classification and regression	2015
YOLO [21]	Anchor-based	Grid-based approach with regression	2016
SSD [24]	Anchor-based	Default box priors with multi-scale feature maps	2016
FPN [20]	Anchor-based	Multi-scale feature learning	2017
FCOS [25]	Anchor-free	Center-ness and scale awareness to predict object recognition points with FCN	2019
CornerNet [27]	Anchor-free	Paired keypoint regression	2019
ExtremeNet [28]	Anchor-free	Four corner point estimation and center point grouping	2019
CenterNet [29]	Anchor-free	Object detection using keypoint triplets	2019
FoveaBox [26]	Anchor-free	Foveation with multi-level feature aggregation	2020
SORT [35]	TBD	Kalman filter and Hungarian algorithm	2016
Deep SORT [36]	TBD	Kalman filter, Hungarian algorithm, and deep appearance feature	2017
IOU-Tracker [32]	TBD	IOU based tracking	2017
Siamese Networks [33]	TBD	Integration of five Siamese networks	2021
GHOST [37]	TBD	Bipartite matching via the Hungarian algorithm	2023
FineTrack [38]	TBD	Flow alignment FPN (FAFPN) and multi-head part mask generator (MPMG)	2023
Triplet [39]	TBD	Triplet sampling for object differentiation with anchor, positive, negative images.	2023
Mask R-CNN [40]	JDT	Visual appearance matching for identification	2017
MOTS [41]	JDT	Segmentation with joint detection and Kalman filtering	2019
Tracktor [42]	JDT	Tracking with dynamic bounding boxes and motion model	2019
RetinaTrack [43]	JDT	Siamese Network with RetinaNet	2020
CenterTrack [44]	JDT	CenterNet with deep association embedding	2020
Dual-regression [45]	JDT	FPN with fine-grained correlation filter and regression-based visual tracking	2020
Self-SDCT [46]	JDT	Self-supervised representation learning	2020
Chained-Tracker [47]	JDT	Paired attentive regression and chained regression for tracking	2020
FairMOT [48]	JDT	Joint detection and embedding learning and fairness regularization	2021
MMNet [49]	JDT	Multi-modal fusion and graph convolutional network	2022
ByteTrack [50]	JDT	Temporal information and byte-level features	2022
FairMOT+Circle Loss [51]	JDT	FairMOT with Polarized Self-Attention and Circle Loss	2023
MTTJDT (Ours)	JDT	FCOSx with dual-frame features for detection and ByteTrack for tracking	2023

which utilizes two adjacent frames as inputs and combines the ResNet-50 and Feature Pyramid Network (FPN) to extract high-level semantic features for predicting matching boxes and binding targets in adjacent frames using anchor-based detectors. To enhance the performance of the CTracker model, the MTTJDT model utilizes smaller networks to extract object features, implements an improved anchor-free detector capable of analyzing two frames simultaneously, and implements a trajectory prediction method for generating tracking paths for objects in the video sequence. An overview of the architecture used in this study is illustrated in Fig.2. We adopted ResNet-18 in combination with an FPN for the feature-extraction process using a multi-level feature pyramid. We propose a combined feature method in which features from the current and previous frames are blended to decrease the feature losses. The multi-loss function is used to reduce imbalanced categories and improve the ability to identify objects between two adjacent frames. The combined features are then fed into two separate branches: classification and regression networks. Finally, the pairs of bounding boxes are filtered based on their confidence in the classification and matching box pairings. Online tracking is achieved using the node ChainingX technique to improve the ability to connect targets across frames, allowing it to track objects with both high and low detection scores. This enables the detection of complex object motion in a moving UAV video.

C. CLASSIFICATION AND REGRESSION PREDICTION NETWORKS

The UAV video input is split into frames $\{F_1, F_2, \ldots, F_N\}$ and fed into the core. ResNet-18 is used to extract high-level semantic features and is integrated with the FPN to generate multi-scale feature representations for associating adjacent frames targets. To minimize the calculations, the extracted features of the current frame F_t, that is, $\{f_t^1, f_t^2, \ldots, f_t^i\}$, are temporarily stored using a Memory Sharing Mechanism (MSM) for memory sharing between two consecutive frames. This enables our method to accept frame-by-frame inputs while using feature data from the two adjoining frames. The related features of the two adjacent frames F_{t-1} and F_t are combined to construct object bounding boxes of the same ID using double-box regression. This method can significantly improve the ID check of the regression branch when it identifies objects that may be occluded or warped by other objects and the background data in neighboring frames. Therefore, the identity switching (IDS) problem caused by an object's occlusion, overlap, or blending in the current frame is significantly reduced.

The detection and regression frameworks are illustrated in Fig.2. The network contained two branches: object classification (Cls_h) and object boundary box regression (Reg_h). Both branches were trained concurrently with no weight-sharing. The Cls_h branch was used to classify the ten target categories. The localization branch outputs the

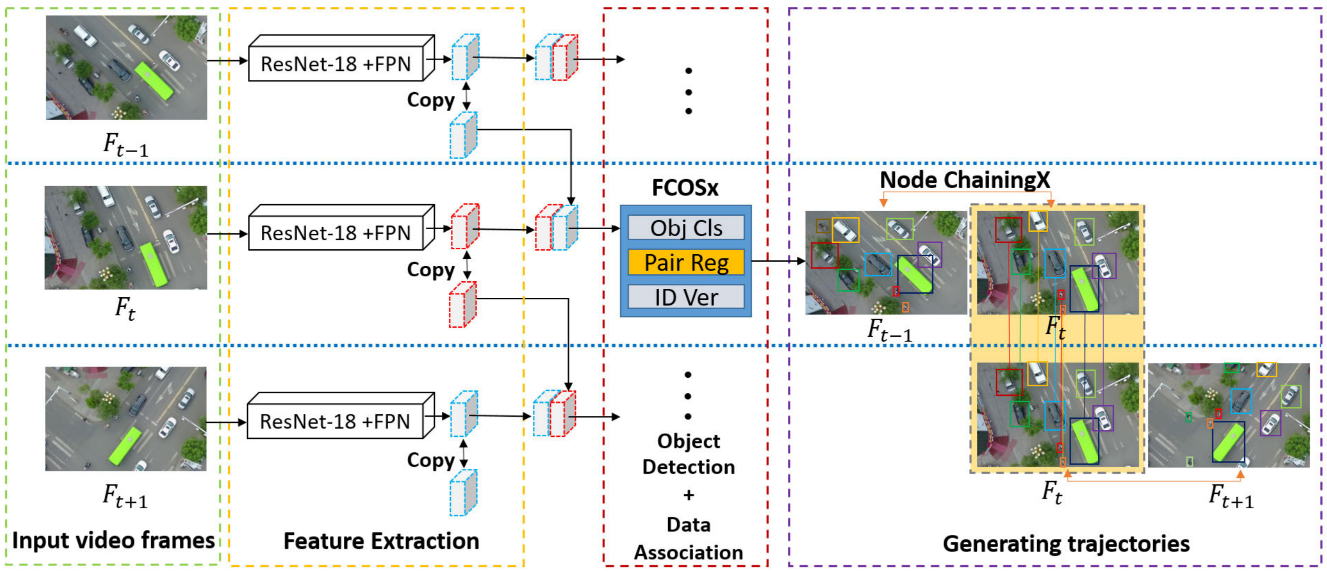


FIGURE 1. A pipeline of our MTTJDT target tracking framework. The procedure is as follows: 1) Input video frames: The current video frame (F_t) is fed into the system for processing. 2) Feature Extraction: A combination of ResNet-18 and FPN is used to extract relevant features (*feature$_t$*) from the current frame (F_t). This feature is then saved for use in subsequent processing and with the next frame. 3) Object Detection and Data Association: The features of both the previous (*feature$_{t-1}$*) and current frames (*feature$_t$*) are analyzed using FCOSx, with 3 subnets for classification (Obj. Cls), ID verification (ID Ver), and paired boxes (Pair Reg). The final output (*final_bboxes*) comprises detection scores and bounding box locations of all objects present in the current frame, which are calculated in conjunction with information from the preceding frame. 4) Generating Trajectories: The node ChainingX method is employed to associate object boxes (*final_bboxes*) and generate trajectories for all identified targets.

location of each positively sampled bounding box in both frames for ID inspection, thereby enabling the regression to concentrate on areas that match the same ID target. The method was improved from that of CTracker, a joint detection and tracking algorithm that utilizes adjacent frame pairs as inputs. CTracker employs an object detection method called RetinaNet, which utilizes anchor boxes and a paired-box regression branch to identify matched boundary boxes of targets appearing in adjacent frames. The algorithm consists of two branches: a classification branch, which predicts the object-type confidence score, and an ID verification branch, which corresponds to the same target. However, this method has been further enhanced by incorporating an anchor-free object detection method called FCOS, which comprises two branches: classification and regression. This modification enables the FCOS to detect objects from two frames simultaneously, thus increasing its detection and tracking capabilities. Additionally, the ID verification branch was updated to use the CIoU loss functions for faster convergence on the box boundaries of the same object. In brief, the proposed method, FCOSx combines the strengths of both CTracker and FCOS, resulting in an improved joint detection and tracking performance.

For the object prediction process, FCOSx, which is an upgraded version of FCOS, was adopted. It takes the features of two adjacent frames as inputs. The FPN is applied to every location on the feature map and highlights locally informative regions for paired box regression. In the regression procedure, any point in the ground-truth bounding boxes was considered a positive sample, whereas a point in multiple boxes was considered a fuzzy sample. When ambiguity exists, the smallest boundary is selected as the target regression. Unlike normal detection models, the bounding box coordinates are denoted by (l^*, t^*, r^*, b^*), so that FCOSx can select more positive samples [25] as shown in (1) and (2).

$$l^* = x - x_0^{(i)}, \quad t^* = y - y_0^{(i)}, \tag{1}$$

$$r^* = x_1^{(i)} - x, \quad b^* = y_1^{(i)} - y. \tag{2}$$

where the ground-truth bounding boxes for an input image are defined as $x_0^{(i)}, y_0^{(i)}, x_1^{(i)}, y_1^{(i)}$, denoting the coordinates of the top left and bottom right corners of the bounding box, respectively. Location (x, y) is considered a positive sample, and l^*, t^*, r^*, b^* are the distances from location to the four sides of the bounding box. Points inside the ground-truth bounding boxes were considered positive samples because they were created using points far from the center of the object. This leads to the generation of numerous low-quality prediction boundary boxes. To avoid this issue, we employ the Complete Intersection Over Union (CIoU) metric, which is used to measure the overlap between bounding boxes in an object detection algorithm. It considers the size and shape of the boxes as well as the distance between them. The soft non-maximum suppression (soft-NMS) technique, as in [59], is a boundary box suppression approach that assigns scores to each box depending on its overlap with other boxes and progressively reduces the scores of the overlapping boxes before suppressing them. In this study, we combined the CIoU and soft-NMS to assess the overlap more precisely and suppress the overlapping bounding boxes in the final output.

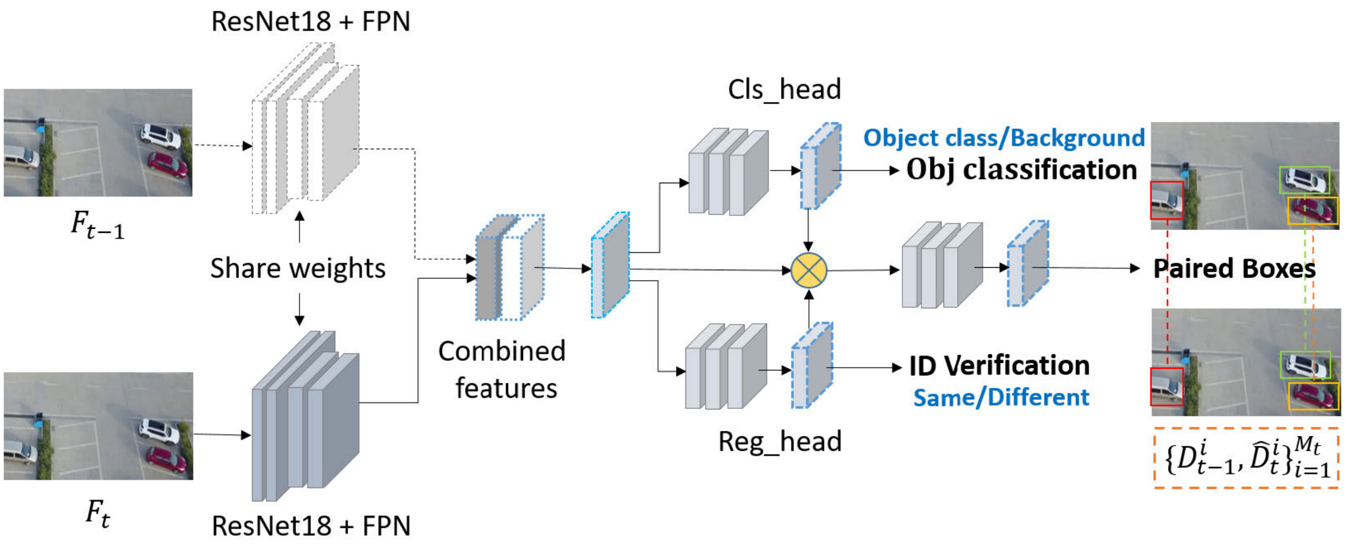


FIGURE 2. Two continuous frame, F_t and F_{t-1} are inputs to the network architecture of MTTJDT, which consists of the following steps: 1) utilizes the ResNet-18 core and FPN to extract current frame features and combine them with previously saved frame features; 2) associates the values of both frames such that combined features are used to predict boxes that match; and 3) detects and predicts the movement of objects for later inter-frame associations.

D. LEARNING TO JOIN DETECT AND TRACK

Our proposed network is trained in an end-to-end manner, which requires independent export of target categories and positions. It is a multi-task detection and tracking model that is trained using a multi-loss function composed of classification and localization losses. A loss-scale parameter is used to weigh the two loss functions.

1) CLASSIFICATION LOSS

Cross-entropy loss is often employed in classification loss calculations for detection. The balanced cross-entropy approach compensates for losses when samples in each category are imbalanced. However, they cannot differentiate between simple and hard samples. Therefore, in this study, we adopt the focal loss (FL) presented in [23] to increase the loss contribution of challenging samples while decreasing the loss contribution of simple samples. The focal loss is defined as follows:

$$FL(p_t) = -\alpha(1-p_t)^{\gamma}\log(p_t) \tag{3}$$

where p_t denotes the category prediction probability, α and γ are constants, typically set to 0.25 and 2, respectively.

2) LOCALIZATION LOSS

Our detection objective was to locate the target in the drone video dataset. Boxes are frequently used for depicting objects. The intersection over union (IoU) [60] is a primary indicator for determining the efficacy of target detection. However, this form of loss function is only applicable when the bounding boxes of the objects overlap. Target identification using UAV cameras often varies during the flight. If IoU is calculated directly, it affects the location of the object boundary box. For this study, we chose a complete IoU (CIoU) loss function presented in [61] that considers three geometric factors: the overlap area, normalized central-point distance, and shape of the boxes. Our CIoU loss function, L_{CIoU} is defined as

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \tag{4}$$

where b^{gt} and b denote the center coordinates of the target and the predicted bounding boxes, respectively. ρ is the Euclidean distance between the center coordinates and c is the diagonal length of the smallest enclosing box covering the two boxes. v is the consistency of the aspect ratio defined in (5), and α measures the positive and negative parameters of IoU defined in (6), as presented in [61].

$$v = \frac{4}{\pi^2}(\arctan\left(\frac{w^{gt}}{h^{gt}}\right) - \arctan\left(\frac{w}{h}\right))^2 \tag{5}$$

$$\alpha = \begin{cases} 0, & \text{if } IoU < 0.5 \\ \dfrac{v}{(1-IoU)+v}, & \text{if } IoU \geq 0.5 \end{cases} \tag{6}$$

where w and h are the width and height of the intersection box.

3) MULTI-LOSS FUNCTION

In this study, a coefficient loss scale was applied to the weight between the two loss functions. Our multi-loss function is expressed as follows:

$$L(\{p_{x,y}\}, \{t_{x,y}\}) = \frac{1}{\alpha^2 \cdot N_{pos}} \sum_{x,y} L_{cls}(p_{x,y}, c^*_{x,y}) + \frac{1}{\beta^2 \cdot N_{pos}} \sum_{x,y} L_{reg}(t_{x,y}, t^*_{x,y}) \tag{7}$$

where L_{cls} denotes the classification loss, and L_{reg} the CIoU loss. N_{pos} denotes the number of positive samples; $p_{x,y}$ represents the prediction probability of the target; and $c^*_{x,y}$ represents the ground-truth box. If it is a positive sample, $c^*_{x,y}$ is equal to one; otherwise, it is equal to zero. $t_{x,y}$ and $t^*_{x,y}$ represent the location of the prediction box and ground-truth box, respectively. The weight scales of L_{cls} and L_{reg} are denoted by α and β, respectively. It should be noted that $\alpha^2 \times \beta^2 \geq 1$.

E. MULTI-TARGET TRACKING USING JOINT DETECTION AND TRACKING (MTTJDT)

In this section, we describe the proposed MTTJDT model. A pseudocode is presented in Algorithm 1, where the inputs are the UAV video frames F and the output is the object trajectory T. The algorithm begins by extracting features through the core network (Lines 6-8), and then merges previous features in the MSM of frame f^i_{t-1} with the current frame features f^i_t (Lines 9-10). The combined features are then fed into the next process, that is, object detection and ID verification (Lines 12 × 16). The FCOSx technique was adopted for this process, as described in Section III-C. The network is split into two branches: classification and regression. The FCOSx algorithm computes the characteristics of two adjacent frames simultaneously. The result consists of an object classification score ($cls\_s_{t,t-1}$) for both frames (Line 12). The regression branch outputs the regression features of both frames (reg_f) (Line 13). Next, the algorithm computes the object box center reference points (p_t) based on the relevant features (reg_f) obtained in the preceding step. During the training phase, the distance to the center of the object box in the previous and current frames was computed. If the result is greater than the threshold, then the paired points are considered untrustworthy and must be disregarded. Only matches with a high dependability were retained. In this work, the distance between two predicted points that is more than 24 pixels is considered unreliable (Line 14). Using the reference points (p_t) and the predicted area of the object (reg_f), the algorithm determines the area around the object by calculating the projected distance from a bounding box. The smallest box area ($b_{t,t-1}$) that could represent the object region between two consecutive frames was obtained using the distance2bbox function (Line 15). To filter out the remaining low-quality boxes, we employed the soft-NMS method with classification confidence scores to filter out object boxes ($b_{t,t-1}$) to ensure that the remaining boxes were not negative samples. In this work, we use a Gaussian function of the box overlap score threshold set to 0.7 to determine whether two overlapped boxes are the same object box. The method verifies the ID of the same target box in two consecutive frames. The results of soft_nms are the detection score and bounding boxes of the same target boundary frame in both frames D_t, D_{t-1} (Line 16). To solve the problem of losing object attributions between frames, we propose an algorithm called "Node ChainingX,"

Algorithm 1 MTTJDT
```
1:  Input: UAV video frames $F = \{F_1, F_2, \ldots, F_N\}$
2:  Output: Object Trajectories $T = \{T_1, T_2, \ldots, T_M\}$
3:  procedure Initialize
        // Initialize set of trajectories.
4:      $T \leftarrow \phi$
5:  procedure Extract and combine features
6:      for frame $F_t$ in $F$ do
7:          $f^i_t$ ←extract feature from $F_t$
8:          $f_t$ ←copy feature from $f^i_t$ to MSM
9:      $f_{t-1}$ ←features from previous frame in MSM
10:     $f^i_{t,t-1}$ ←combined features $f^i_t$ and $f^i_{t-1}$
11: procedure object detection and data association
        // Classification and localization of objects.
12:     $cls\_s_{t,t-1} \leftarrow cls\_h(f^i_{t,t-1})$
13:     $reg\_f \leftarrow reg\_h(f^i_{t,t-1})$
14:     $p_t$ = get_point($reg\_f$)
        // Data association and paired boxes for ID verification.
15:     $b_{t,t-1} \leftarrow$ distance2bbox($p_t$, $reg\_f$)
16:     $D_t, D_{t-1} \leftarrow$ soft_nms($b_{t,t-1}$, $cls\_s_{t,t-1}$)
17: procedure Online tracking
        // Initialize bounding box for high/low score detection.
18:     $D_{high} \leftarrow \phi$
19:     $D_{low} \leftarrow \phi$
        // Compute trajectories for all objects under tracking.
20:     $T$ = ByteTrack($D_t$)
21: Return: $T$
```

which associates data between frames by combining the node chaining method of the chained-tracker [47] and the ByteTrack algorithm [50], as shown in Figure 2. ByteTrack takes (D_t) as the input and ignores (D_{t-1}) because the object detection location from the previous frame has already been tracked in the algorithm. The detection boxes (D_t) were divided into D_{high} and D_{low} boxes, based on the detection score threshold values. The Kalman filter is then applied to tracklet T to predict the next location of each tracked object. First, an IoU is calculated between the high-score detection boxes and tracks. Subsequently, a Hungarian algorithm [62] was used to match objects based on similarity. Because of occlusions, blurring, or size changes, certain tracklets may not match the high-score detection box. The algorithm then links the low-score detection box to the mismatched tracklet to separate objects from the background.

IV. IMPLEMENTATION

A. DATASET

The VisDrone2019 MOT dataset [56] was used to evaluate the performance of the MTTJDT model. The dataset included training (56 sequences), validation (7 sequences), and test (33 sequences; test challenge: 16 sequences;

test-dev: 17 sequences) sets. Each video sequence has distinct scenarios and objectives such as different image sizes and filming techniques. Each object in a frame has a bounding box, category, and tracking identification (ID). The dataset contains ten classes (class ID ranges from 1 to 10), including pedestrians, people, bicycles, cars, vans, buses, trucks, tricycles, awning-tricycles, and motors. In this study, we consider all ten object categories. The corresponding label format of the utilized dataset was a coco format. The background object class (ID 0) and other object classes (ID 11) are excluded from the training process.

B. PERFORMANCE METRICS

Our performance evaluations are typical multiple object tracking metrics, as presented in the literature [63], that is, multiple object tracking accuracy (MOTA), multiple object tracking precision (MOTP), mostly tracked (MT) and mostly lost (ML) tracks, false positives (FP), false negatives (FN), and identity switches (IDSW). MOTA represents the overall performance of the model in a multiple object tracking task [63], as shown in (8)

$$\mathrm{MOTA} = 1 - \frac{\sum_t \mathrm{FP}_t + \mathrm{FN}_t + \mathrm{IDSW}_t}{\sum_t \mathrm{GT}_t} \tag{8}$$

where t is the frame index, FP_t, FN_t, IDSW_t, and GT_t denote the number of false positive samples, false negative samples, identity switches, and ground truth objects of frame index t, respectively. MOTA ranged from $-\infty$ to 100 percent. The value is negative when the number of errors exceeds the number of ground-truth objects. The MOTA is composed of three error ratios. First, the false-positive rate ($\overline{\mathrm{FP}}$) [64], defined in (9), indicates the number of detected objects that are missing from the target detection.

$$\overline{\mathrm{FP}} = \frac{\sum_t \mathrm{FP}_t}{\sum_t \mathrm{GT}_t} \tag{9}$$

Second, the false negative ratio ($\overline{\mathrm{FN}}$) [64], defined in (10), indicates the number of objects incorrectly detected during the detection process.

$$\overline{\mathrm{FN}} = \frac{\sum_t \mathrm{FN}_t}{\sum_t \mathrm{GT}_t} \tag{10}$$

Finally, the mismatch ratio ($\overline{\mathrm{IDSW}}$), which represents the number of ID switches for the same object [63], is defined by (11).

$$\overline{\mathrm{IDSW}} = \frac{\sum_t \mathrm{IDSW}_t}{\sum_t \mathrm{GT}_t} \tag{11}$$

The MOTP represents the average overlap between all tracked targets [63] defined in (12).

$$\mathrm{MOTP} = \frac{\sum_{i,t} d_i^t}{\sum_t c_t} \tag{12}$$

where c_t denotes the total matches between the ground truth and detection output, and d_t^i is the bounding box overlap between target i and the ground truth at frame index t. Finally, MT and ML denote the number of successfully tracked trajectories and unsuccessfully followed routes, respectively.

To evaluate the object association in real-time, we adopted the metrics from [65], that is, identification precision (IDP), identification recall (IDR), and F1 score (IDF1). These standards are based on the number of false-positive identities (IDFP), false-negative identities (IDFN), and true-positive identities (IDTP), which are defined as follows:

$$\mathrm{IDP} = \frac{\mathrm{IDTP}}{\mathrm{IDTP+IDFP}} \tag{13}$$

$$\mathrm{IDR} = \frac{\mathrm{IDTP}}{\mathrm{IDTP+IDFN}} \tag{14}$$

$$\mathrm{IDF1} = \frac{2\mathrm{IDTP}}{2\mathrm{IDTP+IDFP+IDFN}} \tag{15}$$

IDF1 is a performance evaluation of object association. The truth-to-result mapping is not frame-by-frame but identity-by-identity throughout the entire sequence. This approach enables a more comprehensive evaluation of tracking performance as it accounts for any potential changes in object appearance or movement over time. For instance, an object tracked correctly in one time frame, but not in the next frame, is identified as an error in the identity-based evaluation, whereas this would not be captured in a frame-by-frame comparison.

C. MODEL TRAINING

For data augmentation, we applied random cropping (between 0.3 and 0.9), random scaling (between 0.5 and 1.2), photometric distortion (which changes an image's brightness, contrast, saturation, and hue), and random flipping of the node's location along a specified axis with a specified probability. The initial learning rate was set as $5e^{-5}$. The network was trained for 50 epochs, with the learning rate decaying five times after 10, 20, 30, 40, and 50 epochs. The model was trained using a GeForce Tesla T4 GPU with a batch size of 1. Focal loss (FL) was used to adjust for imbalance in the object classes. The weight scales of the classification and regression losses were set to 0.33 and 3.33, respectively. For the extraction and combination of feature procedures, the threshold for randomly importing two nearby frames for model training was set within three frames of the current frame. For the object detection and data association procedure, the detection score threshold was set to 0.3, and computations were performed using the distance2bbox module, as described previously. The center reference location area of the object was set to 24 pixels, both vertically and horizontally, from the center of the box. The filtering process around the final object was performed using the soft_nms module with the Gaussian method, and the IoU threshold was set to 0.7. For the online tracking procedure, the threshold for the number of frames used to save mismatched tracks during association was set to 10, and the minimum detection score threshold for object tracking was set to 0.3. The first association, the second association, and initial new

TABLE 2. **Ablation study on VisDrone2019 validation set.**

Models	MOTA↑	MOTP↑	IDF1↑	Prcn↑	Rcll↑	IDP↑	IDR↑	FN↓	FP↓	IDSW↓
BM	10.5%	69.7%	30.0%	60.7%	38.0%	38.9%	24.4%	70,730	28,126	3,295
BM+M1	24.6%	72.4%	35.2%	78.1%	36.9%	54.9%	25.9%	72,012	11,798	2,285
BM+M1+M2	16.4%	70.9%	34.0%	66.2%	38.0%	46.6%	26.7%	70,768	22,107	2,531
BM+M1+M3	24.9%	72.5%	35.7%	78.6%	36.9%	56.0%	26.3%	72,016	11,461	2,185
BM+M1+M3+M4	25.1%	71.9%	42.3%	77.4%	36.3%	66.2%	31.1%	72,667	12,104	698

track detection thresholds using IoU were set to 0.80, 0.50, and 0.65, respectively.

V. EXPERIMENTAL RESULTS

A. ABLATION STUDY

In this section, an ablation study of the MTTJDT, which is divided into four sub-modules, is performed. Our base model (BM) is a center-based anchor-free network comprising two sub-branches: classification and regression. We utilized ResNet-18 and FPN as backbone networks. The details of each sub-module are as follows:

- Sub-module 1 (M1): applying the "get_point" module to find the waypoints by searching where the same object is within the two frames using use_stride_channel function together with IoU, and figuring out how far away the object is from its center via the CIoU loss function.
- Sub-module 2 (M2): using the "cls_attention" module to focus on the image areas containing objects of a specified class by multiplying the class feature metric from the classification branch by the combined feature metric before feeding into the regression branch to find the location of an object.
- Sub-module 3 (M3): employing the "soft_nms" method as a filter for low-quality bounding boxes that considers both the detection score and IoU score.
- Sub-module 4 (M4): adding "ChainingX" method to connect objects across frames by regression and takes into account both objects with low and high detection scores. This helps improve the object-ID tracking performance.

The results of the ablation study on the VisDrone2019 validation set with 10-epoch training models are presented in Table 2. Our base model (BM) started with 10.5% MOTA, 30% IDF1, and 3,295 IDSW. With the addition of M1 to the base model, MOTA improved to 24.6%, IDF1 increased to 35.2%, and IDSW decreased to 2,285. When M1 and M2 were added to the base model, MOTA decreased to 16.4%, IDF1 decreased to 34.0%, and IDSW increased to 2,531.

It can be seen that M2 reduces the overall performance of the model. Consequently, M2 was removed in the subsequent experiments. Next, we appended M1 and M3 to the base model, and the overall performance was improved; that is, MOTA increased to 24.9%, IDF1 increased to 35.7%, and IDSW decreased to 2,185. Finally, our MTTJDT model (BM+M1+M3+M4) achieved 25.1% for MOTA, 42.3% for IDF1, and 698 for IDSW. Next, a detailed discussion on the performance of each model is provided. The objective of M1 is to focus more on the region of the object between the two frames and the object's center location. This allows it to successfully maintain the features and locations of objects across the frames of perspective shift in the UAV video. M1 utilizes a CIoU loss function, which incorporates the concept of intersection over union (IoU) as well as the center distance and aspect ratio, as opposed to utilizing the "center-ness" method from FCOS in the base model. Despite its complexity, M1 demonstrated the capability of converging on the boundary box of the same object at a faster rate. For this topic, we focus on the object positioning indications and four major object tracking metrics, i.e., MOTA, MOTP, Prcn, and FP. As shown in Table 2, FP decreased from 28,126 to 11,798, whereas MOTA, MOTP, and Prcn increased from 10.5%, 69.7%, and 60.7% to 24.6%, 72.4%, and 78.1%, respectively. The results indicate that M1 helps achieve a more accurate position of the object between frames and significantly minimizes false positives, resulting in a considerable increase in the overall efficiency of object tracking, as shown in Fig.3. M2 is applied to an object prediction network that simultaneously analyzes the features from two adjacent frames. This effect can be observed when objects are in motion, or when objects of the same class overlap. When the location of the bounding box of an object is incorrect, the efficacy of the object detection between frames decreases. Note that for the values of MOTA, Prcn, FN, and FP listed in the first and second rows of Table 2, FN decreased from 72,012 to 70,768, indicating a more accurate object detection. However, FP increased from 11,798 to 22,107, whereas Prcn decreased from 78.1% to 66.2%, indicating incorrect object position. Moreover, there is a decrease in the MOTA from 24.6% to 16.4%, which drastically reduces the tracking performance. An example of the M2 effect is illustrated in Figure.4.

M3 utilizes a technique known as soft-NMS, which effectively filters out low-quality bounding boxes by considering both the detection score and intersection over union (IoU) score. This results in a reduction in the number of bounding boxes for low-quality objects following the initial object detection round. Importantly, the incorporation of this technique did not significantly increase the overall calculation time because the parameters were derived from the previous process. To evaluate the performance of M3, we must focus on the ID association indicators, that is, IDP, IDR, and IDSW, and the detection indicators, that is, FN and FP between models BM+M1 and BM+M1+M3. As shown

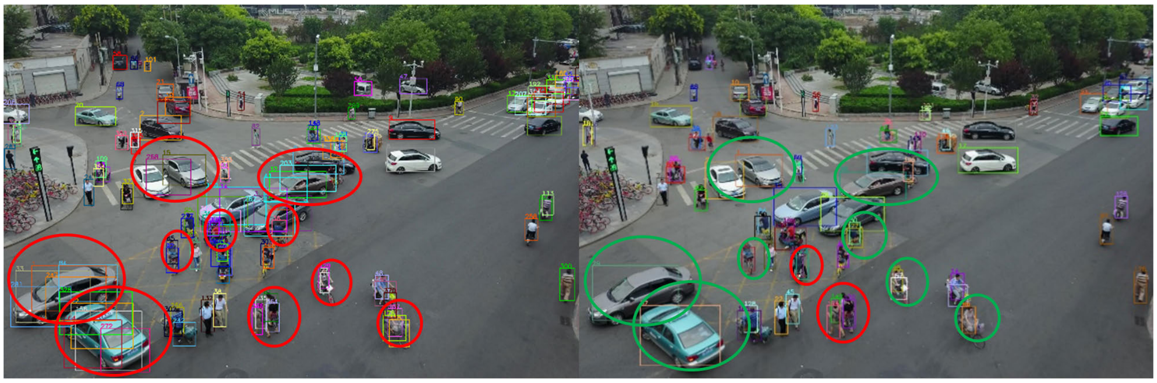


FIGURE 3. Shows how M1 helps improve the results of identifying the bounding box. The image on the left shows the location of the object as determined by BM, and the image on the right shows the result of the base BM + M1. Red circles indicate that there are several bounding boxes around a single object. Green circles signify that the object's bounding boxes have been correctly identified.

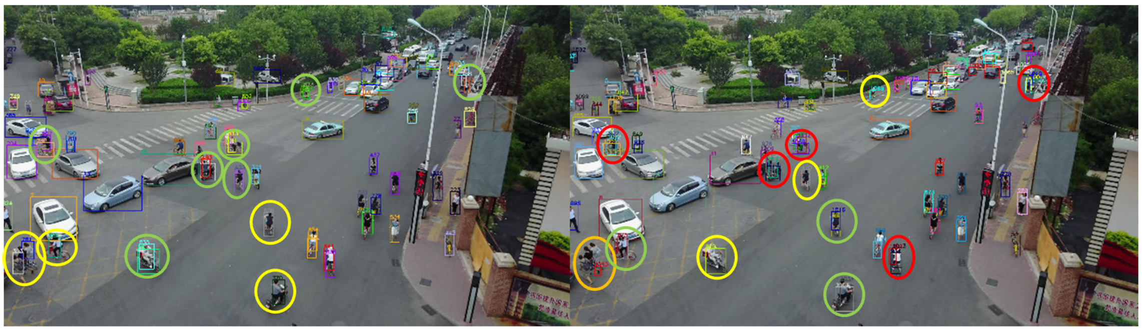


FIGURE 4. Shows the experimental outcomes for BM+M1 (left) vs. BM+M1+M2 (right). Red circles indicate that one single object has several overlapped bounding boxes. Yellow circles indicate the losses from object detections. Green circles indicate that the object's bounding boxes have been accurately detected.

in Table 2, 1DP increased from 54.9% to 56.0%, IDR increased from 25.9% to 26.3%, IDSW decreased from 2,285 to 2,185, and FP detection indicator decreased from 11,798 to 11,461. This module does not improve object detection; it merely filters bounding boxes from the objects. Therefore, the FN values did not change significantly. These results indicate that M3 has a positive effect on ID matching, mainly because of low-quality, false-positive object bounding box filtering, as illustrated in Fig.5. Our MTTJDT Framework utilizes the FCOSx method, which employs M1 and M3 modules as the primary means of implementing object detection and data association to enhance the precision of object detection. The FCOSx method is initiated by identifying the approximate object region using the get_point and use_stride_channel functions. The object box positioning is further refined by considering the center distance and aspect ratio from the CIoU, as determined by M1. Additionally, module M3 filters out low-quality boxes by evaluating both the detection score and the object box overlap area with soft_nms. The results of implementing FCOSx (BM+M1+M3) demonstrated a substantial improvement in the object positioning accuracy when compared to the base method (BM). This was demonstrated by a decrease in FP from 28,126 to 11,461 and an increase in MOTA from 10.5% to 24.9%.

To evaluate M4, the performance metrics of online object tracking, that is, IDF1, IDP, IDR, IDSW, and the overall object tracking performance indicator, MOTA, were considered. Observe the models BM+M1+M3 and BM+M1+M3+M4 from Table 2, the IDF1 increased from 35.7% to 42.3%, the IDP increases from 56.0% to 66.2%, the IDR increased from 26.3% to 31.1%, and the IDSW decreased from 2,185 to 698. The total tracking efficiency of MOTA increased from 24.9% to 25.1%. As indicated by the higher IDF1 value, the M4 method has been shown to effectively address the limitations of traditional re-identification methods by incorporating the locations of momentarily missing objects stored in the buffer, including those with low detection scores. This method employs additional memory to store the locations of mismatched objects while linking with a threshold of 10 frames. This has no effect on the overall storage space; however, M4 uses the position of these objects in relation to a Kalman filter-predicted trajectory, which is capable of strengthening the ID linkages across frames over time. This is the outcome of comparing low-detection-score objects with object location predictions that consider their movements. Figure 6 shows the benefits of adding M4 to the object tracking phase. The results illustrate that the model can track objects from frames

70 to 110, with almost every object maintaining its original ID.

Finally, we trained our MTTJDT model for up to 50 epochs to evaluate the performance of the VisDrone2019 test-dev dataset with 10 object categories. The results of the test dataset for all scenarios are listed in Table 3. Our MTTJDT utilizes TensorRT, a library developed by NVIDIA, to optimize deep learning models for deployment in NVIDIA graphics processing units (GPUs). The utilization of TensorRT allowed for a significant increase in the processing speed, with the ability to execute 77 frames per second (fps) from the point of execution of the main function to the point of exit. This processing speed is suitable for real-time application. From Table 3, the top two sequences with the lowest object tracking performance are uav0000188_00000_v and uav0000073_04464_v, with MOTA values of 14.40% and 16.10%, respectively. These datasets consist of objects that are both in motion and rather small. In addition, there are low-light scenarios. The two subsets for which our MTTJDT model achieved high performance are uav0000249_02688_v and uav0000009_03358_v. These sequences are scenarios in which the light level is high and most objects are large. However, because of the sufficient amount of light in the sequences, even tiny objects that are partially obscured can be reliably recognized and tracked, as shown by the MOTA, which are 49.80% and 49.30%, respectively.

To further analyze the factors that affect the performance, we investigated the characteristics of each subset of the VisDrone2019 test-dev dataset. The characteristic scores for each aspect of the test sequences were defined as follows:

- "size" defines the average area of the bounding box (in pixels) for all objects in a subset.
- "center_move" denotes the rate at which the object's center bounding box (pixel) moves between each frame pair.
- "density" denotes the density (in decimal) of the number of objects in the image, determined by comparing the total number of objects to the total number of frames in each subset.
- "brightness" defines the average brightness of each subset's images. The Python module ImageStat-Pillow was used to calculate the brightness of each pixel in the image.
- "people" and "vehicles" denote the percentages of the two main types of objects in each subset, i.e., the people and the vehicles, respectively.
- "occlusion" defines as the average occlusion of all objects in the frame of each subset.
- "SCDB" defines the multiplication of "size," "center_move," "density," and "brightness".

Figure 7 displays the normalized plots of the characteristic scores of each subset. It can be inferred from Fig.7a that when the "size" is small, the "center_move" and the "density" are high, which considerably decreases the performance of MOTA and IDF1. As for the "brightness," when the scene is very dark at night or extremely bright during the day, the tracking performance decreases. Figure7b indicates that MOTA and IDF1 are linear to the "SCDB" metric, i.e., "size," "center_move," "density," and "brightness" are the main factors that affect multiple object tracking capability. Figure 8 shows that the size of the object and intensity of the light in the image have a significant impact on the performance of our MTTJDT model.

B. COMPARISON WITH STATE-OF-THE-ARTS

In this section, we compare the performance of our MTTJDT model with that of related works in the literature using the VisDrone2019 dataset. Details of each comparison are provided in this section. The MOTDT model [66] solves unreliable detection by accumulating candidates from both detection and tracking and constructing a CNN-based scoring mechanism to deliver the optimal selection. The SORT method [35] uses the Kalman filter and Hungarian algorithm to predict the location of the bounding box and matches it with the current detection that has the greatest IoU value. The MOTR model [67] executes a sequence prediction series by updating the track query information. The Transformer [68] method automatically begins a new track based on a query for a specified item and follows an existing track in space and time. The search for new tracks is predicated on retaining the identification of the previous track queries. The IOUT method [32] measures the similarity of detections over successive frames using intersection-over-union (IoU). Finally, the GOG model [69] correlates the detection inputs in video frames using a minimum-cost flow algorithm and a cost function computed from appearance and motion data to estimate the number of trajectories, and birth and death stages.

The proposed MTTJDT model incorporates the advantages and disadvantages of each method into the core concept of constructing an object detection and tracking network. The MTTJDT is based on a JDT network that is appropriate for online object tracking. We trained the model using two adjacent frames to prevent loss of features between frames. Our method processes only one frame, where the current frame features are copied to the next frame to avoid doubling up on processing, whereas other approaches input a single frame at a time. Similar to the SORT approach, we correlated data across frames using the Kalman filter and Hungarian method. We adopted a similar approach to MOTDT to use IoU to predict the track locations. We followed the MOTR approach to save all track locations, including the unmatched tracks, to be employed in the next frame. Therefore, as illustrated in Table 4, our method outperforms previous methods in the VisDrone2019 test-dev set, attaining success rates of 31.2% for MOTA and 43.6% for IDF1. Table 5 presents a runtime comparison between our MTTJDT and other multi-object tracking methods on the VisDrone2019 dataset reported in the literature. It appears that our model is superior to the others in terms of achieving a running time of 77 fps on a Tesla T4 GPU.

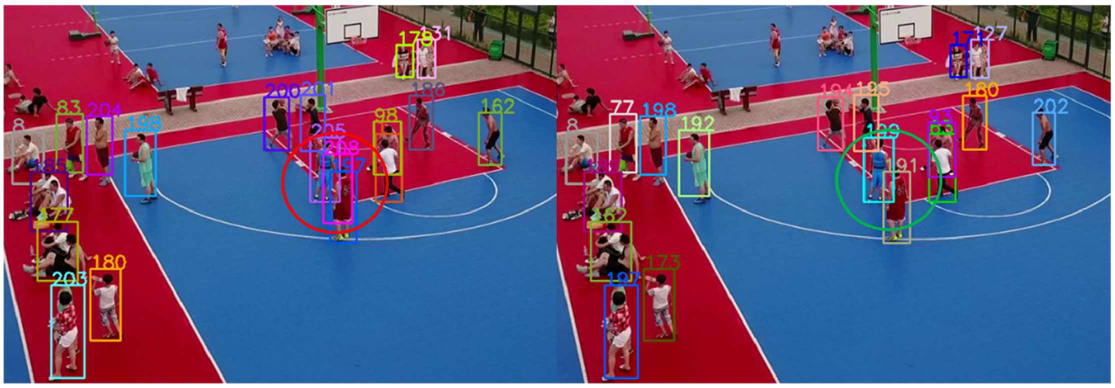
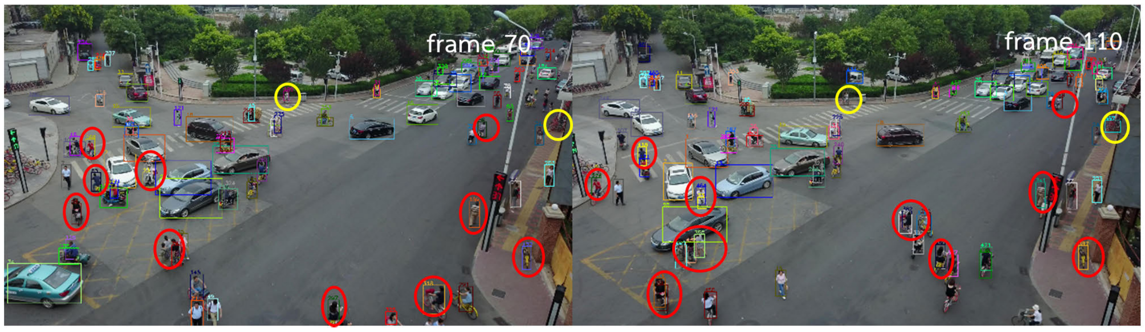
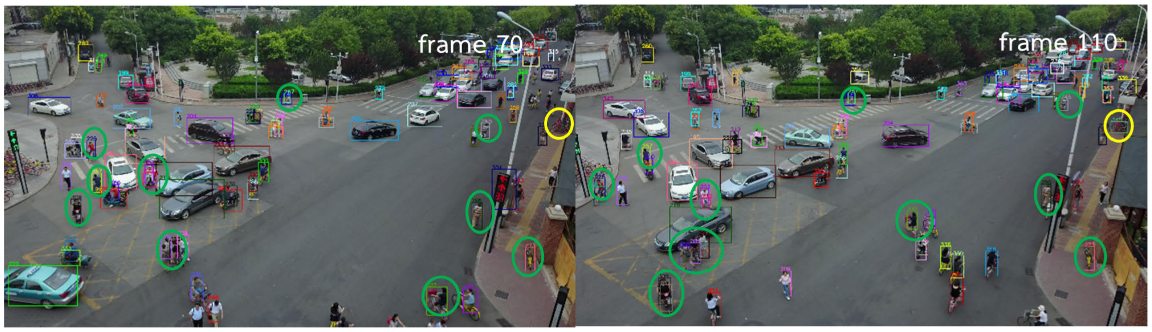


FIGURE 5. Shows the improvement resulting from filtering of low-quality object bounding boxes via M3. The image on the left represents the NMS filtering results of the BM+M1 model, while the image on the right represents the model BM+M1+M3 filtering outcomes. The red circle indicates that there are still boxes of low qualities within the object area. The green circle represents the correct object's bounding box.



(a)



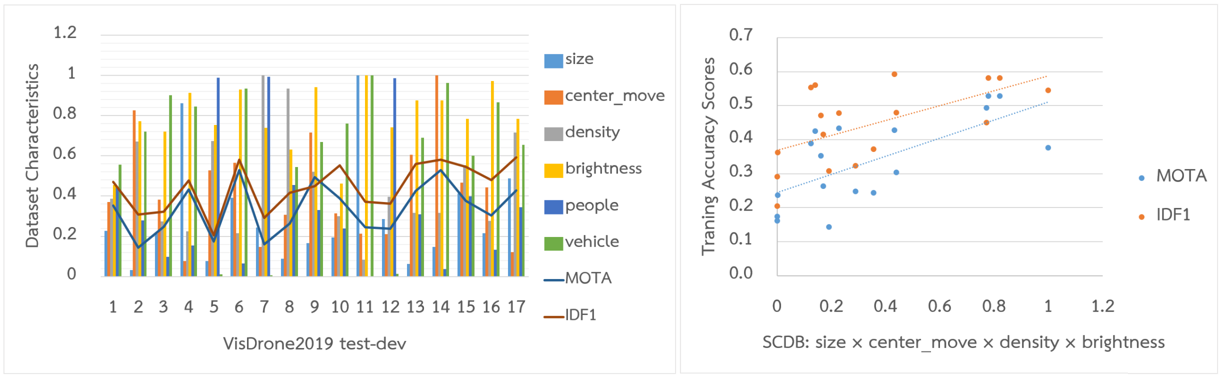
(b)

FIGURE 6. Depicts the outcome of enhancing object tracking using the ChainingX technique of M4 module. Figure 6a demonstrates the result of object tracking utilizing node chaining to link objects across frames from the BM+M1+M3. Figure 6b shows the outcome of tracking objects using low-score detection and their expected positions to validate object identifications. Red circles indicate that object IDs have been switched, while yellow circles indicate that objects have lost track, and green circles indicate that object IDs are identified correctly.

After thorough analysis and testing, our proposed model was found to be suitable for tracking UAV aerial video objects during both daytime and nighttime conditions, as well as during inclement weather. However, it should be noted that there is still a limitation to the capability of our model to detect and accurately track small objects in challenging environments. These limitations can be attributed to the following factors: First, our method involves identifying the smallest boundary that can represent the boundary of the object from two consecutive frames using a reference point (the center point). However, if the distance between the reference points exceeds 24 pixels, as determined by the algorithm, the position of the object may be deemed unreliable. This limitation can lead to inaccurate locating of small objects in densely populated scenes, particularly when the bounding boxes of multiple small objects overlap. Second, multiple trajectories of small objects, such as people and pedestrians, may overlap because the reference

TABLE 3. Evaluation on VisDrone2019 test-dev dataset with 10 object categories.

Datasets	IDF1	IDP	IDR	Rcll	Prcn	GT	MT	ML	FP	FN	IDSW	MOTA	MOTP	# objects
uav0000120_04775_v	47.10%	68.80%	35.90%	43.90%	84.30%	329	55	159	3,482	23,916	227	35.30%	73.10%	42,665
uav0000188_00000_v	30.80%	67.30%	19.90%	22.20%	74.80%	132	5	90	1,434	14,954	63	14.40%	68.10%	19,211
uav0000201_00000_v	32.30%	52.40%	23.40%	35.10%	78.60%	88	10	39	1,949	13,300	157	24.80%	70.80%	20,478
uav0000077_00720_v	47.80%	69.20%	36.50%	48.30%	91.50%	90	27	25	866	9,947	89	43.30%	77.90%	19,244
uav0000088_00290_v	20.50%	48.00%	13.00%	23.00%	84.80%	119	3	76	901	16,884	329	17.40%	68.90%	21,928
uav0000249_00001_v	58.10%	63.70%	53.50%	69.30%	82.50%	173	78	52	1,251	2,612	148	52.80%	75.60%	8,504
uav0000073_04464_v	29.10%	66.30%	18.70%	22.30%	79.20%	161	9	105	2,013	26,688	114	16.10%	68.80%	34,344
uav0000161_00000_v	41.60%	68.90%	29.80%	35.20%	81.40%	193	22	97	2,555	20,538	216	26.40%	70.80%	31,691
uav0000009_03358_v	45.00%	55.80%	37.70%	59.80%	88.50%	118	48	32	975	5,041	339	49.30%	73.90%	12,531
uav0000119_02301_v	55.30%	89.00%	40.10%	42.00%	93.30%	51	13	27	178	3,430	8	38.90%	76.90%	5,915
uav0000370_00001_v	37.20%	79.30%	24.30%	27.60%	90.20%	25	7	10	74	1,791	6	24.40%	73.80%	2,475
uav0000073_00600_v	36.30%	69.30%	24.60%	29.90%	84.20%	103	16	60	803	10,038	75	23.70%	73.30%	14,312
uav0000306_00230_v	56.00%	77.60%	43.80%	49.80%	88.20%	124	26	44	981	7,368	78	42.50%	74.00%	14,666
uav0000249_02688_v	51.50%	56.80%	47.10%	67.70%	81.60%	168	74	41	1,119	2,366	192	49.80%	75.20%	7,322
uav0000297_00000_v	54.50%	61.50%	49.00%	59.80%	75.00%	175	66	49	1,773	3,577	208	37.60%	76.10%	8,903
uav0000355_00001_v	48.00%	85.60%	33.40%	34.80%	89.20%	76	27	32	598	9,284	29	30.40%	77.50%	14,238
uav0000297_02761_v	59.20%	67.80%	52.50%	60.30%	77.90%	168	44	42	5,024	11,647	137	42.80%	73.10%	29,372
OVERALL	43.60%	66.40%	32.40%	40.40%	82.70%	2,293	530	980	25,976	183,381	2,415	31.20%	73.20%	307,799



(a) (b)

FIGURE 7. (a) Depicts the rate of change of MOTA and IDF1 relative to object categories (people and vehicles), average object size (size), and object movement. (center_move), the density of the number of objects in the image (density), and the brightness of the backdrop (brightness) of each subset of data in VisDrone2019 test-dev (1 to 17). (b) Shows the MOTA and IDF1 of each subset against its dataset characteristics score (SCDB).

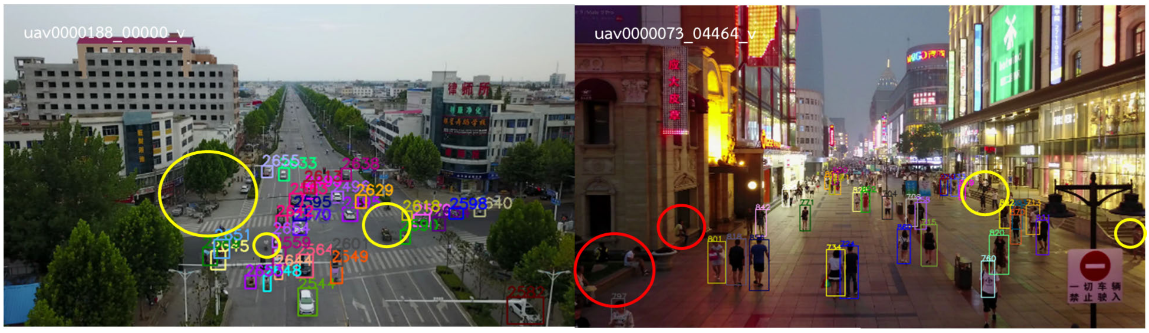
TABLE 4. A comparison between MTTJDT and other approaches for MOT on the VisDrone2019 test-dev [56], [70].

Models	Methods	MOTA↑	MOTP↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓	FM↓
MOTDT [66]	JDT	-0.8%	68.5%	21.6%	87	1,196	44,548	185,453	1,437	3,609
SORT [35]	TBD	14.0%	73.2%	38.0%	506	545	80,845	112,945	3,629	4,838
MOTR [67]	JDT	22.8%	72.8%	41.4%	272	825	28,407	147,937	959	3,980
TrackFormer [68]	JDT	25%	73.9%	30.5%	385	770	25,856	141,526	4,840	4,855
IOUT [32]	TBD	28.1%	74.7%	38.9%	467	670	36,158	126,549	2,393	3,829
GOG [69]	TBD	28.7%	76.1%	36.4%	346	836	17,706	144,657	1,387	2,237
MTTJDT (Ours)	JDT	**31.2%**	73.2%	**43.6%**	**530**	980	25,976	183,381	2,415	6,514

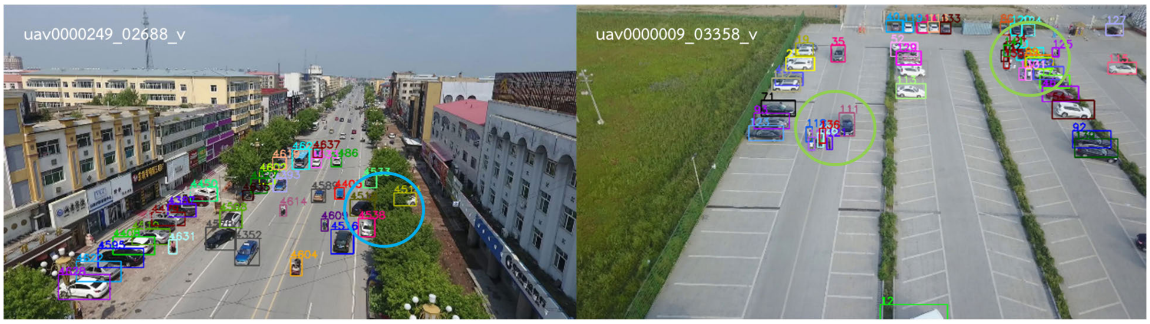
TABLE 5. Speed comparison of MTTJDT and other approaches for MOT on the VisDrone2019 dataset [71].

Models	GPU	CPU	PL	Framework	Speed (fps)
HMTT [72]	GTX TITAN X	Intel i7-4790K@4.00GHz	Python	CenterNet [73]+IOU tracker [32]	0.4
IITD_DeepSort [74]	Tesla K80	Intel Xeon 1.70GHz×16	Python	RetinaNet [75]+DeepSORT [36]	0.3
T&D-OF [71]	TITAN X MAXWELL	Intel i7-7700(48GB)	Python	R-FCN [76]+MOTDT [66]	0.3
TNT_DRONE [77]	Quadro GV100/Titan Xp×2	Intel i7-7700K@4.20GHz	Python, Matlab	Faster R-CNN [19] +TrackletNet [78], [79]	3.2
OS-MOT [71]	GTX980	Intel i7-6700K@4.00GHz×8(16GB)	Matlab	auction assign [80]	5
VCLDAN [81]	GTX 1080Ti	Intel Xeon E5-2640@2.40GHz	Python	DAN [81]	6.3
Flow-Tracker [82]	GTX 1080Ti	Intel Xeon E5-1650v4@3.60GHz×12	Python	Cascade R-CNN [83]+IoU Tracker [32]	5
TrackKITSY [81]	NVS5200M	Intel i7-6700@3.40GHz (16GB)	C++	Cascade R-CNN [83]+TrackCG [77]	10
SGAN [71]	Titan X Pascal	Intel i7-6700@3.40GHz	Python	Social-LSTM [84]	1.5
DBAI-Tracker [71]	Tesla V100	Intel Xeon Platinum 8160	Python	Cascade R-CNN [83]+GOG [69]	20 ∼ 50
MTTJDT (Ours)	Tesla T4	Intel(R) Xeon(R)@2.20GHz	Python	FCOSx + ByteTrack [50]	**77**

points are too close to one another. Consequently, the model may assign an invalid ID to the object. Moreover, challenging environments, including cluttered backgrounds, shadows, and lighting conditions, can significantly affect



(a)



(b)

FIGURE 8. Illustrates our MTTJDT method's test results. Figure 8a demonstrates the object tracking results of the two subsets with the lowest performance. Yellow circles illustrate tiny or moving objects. A red circle denotes objects that are in low-light environments. Figure 8b presents example results of multiple object tracking on the two highest-performing subsets. The blue circle indicates objects that are concealed by the background, and green circles indicate tiny and occluded objects. Nonetheless, they are correctly identified when there is enough light in the scenario.

object tracking performance because it is more difficult to differentiate between the object of interest and other elements in the scene.

VI. CONCLUSION

This paper presents a framework for online MOT based on joint detection and tracking. By using adjacent frame pairs as inputs and sharing the features, the efficiency of the feature extraction increases. In addition, using a multi-loss function to train our model, it is feasible to rectify the disparity between challenging classes and samples. By associating objects between frames, we focus on ID verification using dual regression bounding boxes that stress the center distance of objects between frames rather than the object area only and utilize a single regression for movement forecasting. Online tracking is also possible by predicting the position of an object in the next frame and by exploiting detection techniques that consider objects with both high and low detection scores corresponding to their predicted trajectories to find the same object across frames. Our proposed model can achieve multiple object tracking in real-time. The experimental results show that our method receives the highest MOTA and IDF1 scores for MOT on the VisDrone2019 test-dev for all ten object categories of interest compared with previous work in the literature. Although our model is less efficient in detecting and tracking small objects in challenging environments, we will consider using a larger network for object feature separation to accomplish this goal in the future. In future work, one can consider ensemble learning, which combines the functionalities of different deep learning models. Multi-camera multi-object tracking is also a challenge.

REFERENCES

[1] G. S. Phadke, "Robust multiple target tracking under occlusion using fragmented mean shift and Kalman filter," in *Proc. Int. Conf. Commun. Signal Process.*, Mar. 2011, pp. 517–521.

[2] B. Sahbani and W. Adiprawita, "Kalman filter and iterative-Hungarian algorithm implementation for low complexity point tracking as part of fast multiple object tracking system," in *Proc. 6th Int. Conf. Syst. Eng. Technol. (ICSET)*, Oct. 2016, pp. 109–115.

[3] M. H. Jaward, L. S. Mihaylova, N. Canagarajah, and D. R. Bull, "Multiple object tracking using particle filters," in *Proc. IEEE Aerosp. Conf.*, Jul. 2006, p. 8.

[4] J. U. Cho, S. Jin, X. D. Pham, and J. W. Jeon, "Multiple objects tracking circuit using particle filters with multiple features," in *Proc. IEEE Int. Conf. Robot. Automat.*, Apr. 2007, pp. 4639–4644.

[5] Z. Zhang, K. Fu, X. Sun, and W. Ren, "Multiple target tracking based on multiple hypotheses tracking and modified ensemble Kalman filter in multi-sensor fusion," *Sensors*, vol. 19, no. 12, p. 2180, 2019.

[6] S. P. Coraluppi, "Robustness in multiple-hypothesis tracking," in *Proc. 25th Int. Conf. Inf. Fusion (FUSION)*, Jul. 2022, pp. 1–8.

[7] B. K. Habtemariam, R. Tharmarasa, T. Thayaparan, M. Mallick, and T. Kirubarajan, "A multiple-detection joint probabilistic data association filter," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 3, pp. 461–471, Jun. 2013.

[8] A. F. Tchango, V. Thomas, O. Buffet, A. Dutech, and F. Flacher, "Tracking multiple interacting targets using a joint probabilistic data association filter," in *Proc. 17th Int. Conf. Inf. Fusion (FUSION)*, Jul. 2014, pp. 1–8.

[9] E. Fan, W.-X. Xie, J. Pei, K. Hu, X. Li, and V. Podpecan, "Improved joint probabilistic data association (JPDA) filter using motion feature for multiple maneuvering targets in uncertain tracking situations," *Information*, vol. 9, p. 322, 2018.

[10] C.-Y. Chong, "Graph approaches for data association," in *Proc. 15th Int. Conf. Inf. Fusion*, Jul. 2012, pp. 1578–1585.

[11] J. He, Z. Huang, N. Wang, and Z. Zhang, "Learnable graph matching: Incorporating graph partitioning with deep feature learning for multiple object tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Mar. 2021, pp. 5295–5305.

[12] S. Reuter, B. Wilking, J. Wiest, M. Munz, and K. C. J. Dietmayer, "Real-time multi-object tracking using random finite sets," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, no. 4, pp. 2666–2678, Oct. 2013.

[13] W. Ng, J. Li, S. J. Godsill, and J. Vermaak, "A hybrid approach for online joint detection and tracking for multiple targets," in *Proc. IEEE Aerosp. Conf.*, Mar. 2005, pp. 2126–2141.

[14] E. Akleman, "Deep learning," *Computer*, vol. 53, no. 9, p. 17, Sep. 2020.

[15] Y. Lin, M. Wang, W. Chen, W. Gao, L. Li, and Y. Liu, "Multiple object tracking of drone videos by a temporal-association network with separated-tasks structure," *Remote Sens.*, vol. 14, no. 16, p. 3862, Aug. 2022.

[16] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *Proc. Int. Conf. Eng. Technol. (ICET)*, Aug. 2017, pp. 1–6.

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[18] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[19] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[20] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[22] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525.

[23] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020.

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2016, pp. 21–37.

[25] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9626–9635.

[26] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li, and J. Shi, "FoveaBox: Beyound anchor-based object detection," *IEEE Trans. Image Process.*, vol. 29, pp. 7389–7398, 2020.

[27] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," *Int. J. Comput. Vis.*, vol. 128, no. 3, pp. 642–656, Mar. 2020.

[28] X. Zhou, J. Zhuo, and P. Krähenbühl, "Bottom-up object detection by grouping extreme and center points," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 850–859.

[29] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6568–6577.

[30] M. Jiang, T. Hai, Z. Pan, H. Wang, Y. Jia, and C. Deng, "Multi-agent deep reinforcement learning for multi-object tracker," *IEEE Access*, vol. 7, pp. 32400–32407, 2019.

[31] L. Ren, J. Lu, Z. Wang, Q. Tian, and J. Zhou, "Collaborative deep reinforcement learning for multi-object tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 586–602.

[32] E. Bochinski, V. Eiselein, and T. Sikora, "High-speed tracking-by-detection without using image information," in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug. 2017, pp. 1–6.

[33] M. P. Muresan, S. Nedevschi, and R. Danescu, "Robust data association using fusion of data-driven and engineered features for real-time pedestrian tracking in thermal images," *Sensors*, vol. 21, no. 23, p. 8005, Nov. 2021.

[34] G. Welch and G. Bishop, "An introduction to Kalman filter," in *Proc. Int. Conf. Comput. Graph. Interact. Techn.*, 1995, pp. 1–15.

[35] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 3464–3468.

[36] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 3645–3649.

[37] J. Seidenschwarz, G. Brasó, V. C. Serrano, I. Elezi, and L. Leal-Taixé, "Simple cues lead to a strong multi-object tracker," 2022, *arXiv:2206.04656*.

[38] H. Ren, S. Han, H. Ding, Z. Zhang, H. Wang, and F. Wang, "Focus on details: Online multi-object tracking with diverse fine-grained representation," 2023, *arXiv:2302.14589*.

[39] W.-J. Ahn, K.-S. Ko, M.-T. Lim, D.-S. Pae, and T.-K. Kang, "Multiple object tracking using re-identification model with attention module," *Appl. Sci.*, vol. 13, no. 7, p. 4298, Mar. 2023.

[40] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[41] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "MOTS: multi-object tracking and segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7934–7943.

[42] P. Bergmann, T. Meinhardt, and L. Leal-Taixé, "Tracking without bells and whistles," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 941–951.

[43] Z. Lu, V. Rathod, R. Votel, and J. Huang, "RetinaTrack: Online single stage joint detection and tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 14656–14666.

[44] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 474–490.

[45] X. Li, Q. Liu, N. Fan, Z. Zhou, Z. He, and X.-Y. Jing, "Dual-regression model for visual tracking," *Neural Netw.*, vol. 132, pp. 364–374, Dec. 2020.

[46] D. Yuan, X. Chang, P. Huang, Q. Liu, and Z. He, "Self-supervised deep correlation tracking," *IEEE Trans. Image Process.*, vol. 30, pp. 976–985, 2021.

[47] J. Peng, C. Wang, F. Wan, Y. Wu, Y. Wang, Y. Tai, C. Wang, J. Li, F. Huang, and Y. Fu, "Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2020, pp. 145–161.

[48] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "FairMOT: On the fairness of detection and re-identification in multiple object tracking," *Int. J. Comput. Vis.*, vol. 129, no. 11, pp. 3069–3087, Nov. 2021.

[49] Q. Liu, D. Yuan, N. Fan, P. Gao, X. Li, and Z. He, "Learning dual-level deep representation for thermal infrared tracking," *IEEE Trans. Multimedia*, vol. 25, pp. 1269–1281, 2023.

[50] Y. Zhang, P. Sun, Y. Jiang, D. Yu, Z. Yuan, P. Luo, W. Liu, and X. Wang, "ByteTrack: Multi-object tracking by associating every detection box," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 1–21.

[51] J. Che, Y. He, and J. Wu, "Pedestrian multiple-object tracking based on FairMOT and circle loss," *Sci. Rep.*, vol. 13, no. 1, p. 4525, Mar. 2023.

[52] Z. Wang, L. Zheng, Y. Liu, and S. Wang, "Towards real-time multi-object tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 107–122.

[53] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Detect to track and track to detect," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3057–3065.

[54] B. Pang, Y. Li, Y. Zhang, M. Li, and C. Lu, "TubeTK: Adopting tubes to track multi-object in a one-step training model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6307–6317.

[55] P. Sun, J. Cao, Y. Jiang, R. Zhang, E. Xie, Z. Yuan, C. Wang, and P. Luo, "TransTrack: Multiple object tracking with transformer," 2020, *arXiv:2012.15460*.

[56] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, and H. Ling, "Detection and tracking meet drones challenge," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 7380–7399, Nov. 2022.

[57] M. Mueller, N. G. Smith, and B. Ghanem, "A benchmark and simulator for UAV tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 445–461.

[58] I. Kalra, M. Singh, S. Nagpal, R. Singh, M. Vatsa, and P. B. Sujit, "DroneSURF: Benchmark dataset for drone-based face recognition," in *Proc. 14th IEEE Int. Conf. Autom. Face Gesture Recognit. (FG)*, May 2019, pp. 1–7.

[59] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-NMS—Improving object detection with one line of code," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5562–5570.
[60] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 658–666.
[61] Z. Zheng, P. Wang, D. Ren, W. Liu, R. Ye, Q. Hu, and W. Zuo, "Enhancing geometric factors in model learning and inference for object detection and instance segmentation," *IEEE Trans. Cybern.*, vol. 52, no. 8, pp. 8574–8586, Aug. 2022.
[62] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. Logistics*, vol. 52, no. 1, pp. 7–21, Feb. 2005.
[63] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The CLEAR MOT metrics," *EURASIP J. Image Video Process.*, vol. 2008, pp. 1–10, Jan. 2008.
[64] K. Smith, D. Gatica-Perez, J. Odobez, and S. Ba, "Evaluating multi-object tracking," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)-Workshops*, Sep. 2005, p. 36.
[65] E. Ristani, F. Solera, R. S. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2016, pp. 17–35.
[66] L. Chen, H. Ai, Z. Zhuang, and C. Shang, "Real-time multiple people tracking with deeply learned candidate selection and person re-identification," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2018, pp. 1–6.
[67] F. Zeng, B. Dong, T. Wang, C. Chen, X. Zhang, and Y. Wei, "MOTR: End-to-end multiple-object tracking with transformer," in *Proc. Eur. Conf. Comput. Vis.*, 2021, pp. 659–675.
[68] T. Meinhardt, A. Kirillov, L. Leal-Taixé, and C. Feichtenhofer, "Track-Former: Multi-object tracking with transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8834–8844.
[69] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *Proc. CVPR*, Jun. 2011, pp. 1201–1208.
[70] S. Liu, X. Li, H. Lu, and Y. He, "Multi-object tracking meets moving UAV," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8866–8875.
[71] L. Wen, P. Zhu, D. Du, X. Bian, H. Ling, Q. Hu, and J. Zheng, "VisDrone-MOT2019: The vision meets drone multiple object tracking challenge results," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 189–198.
[72] S. Pan, Z. Tong, Y. Zhao, Z. Zhao, F. Su, and B. Zhuang, "Multi-object tracking hierarchically in visual data taken from drones," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 135–143.
[73] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," 2019, *arXiv:1904.07850*.
[74] A. Jadhav, P. Mukherjee, V. Kaushik, and B. Lall, "Aerial multi-object tracking by detection using deep association networks," in *Proc. Nat. Conf. Commun. (NCC)*, Feb. 2020, pp. 1–6.
[75] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2999–3007.
[76] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 379–387.
[77] W. Tian, M. Lauer, and L. Chen, "Online multi-object tracking using joint domain information in traffic scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 1, pp. 374–384, Jan. 2020.
[78] G. Wang, Y. Wang, H. Zhang, R. Gu, and J.-N. Hwang, "Exploit the connectivity: multi-object tracking with TrackletNet," in *Proc. 27th ACM Int. Conf. Multimedia*, Oct. 2019, pp. 482–490.
[79] H. Zhang, G. Wang, Z. Lei, and J.-N. Hwang, "Eye in the sky: Drone-based object tracking and 3D localization," in *Proc. 27th ACM Int. Conf. Multimedia*, Oct. 2019, pp. 899–907.
[80] D. P. Bertsekas, "Auction algorithms for network flow problems: A tutorial introduction," *Comput. Optim. Appl.*, vol. 1, no. 1, pp. 7–66, Oct. 1992.
[81] S. Sun, N. Akhtar, H. Song, A. Mian, and M. Shah, "Deep affinity network for multiple object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 1, pp. 104–119, Jan. 2021.
[82] W. Li, J. Mu, and G. Liu, "Multiple object tracking with motion and appearance cues," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 161–169.
[83] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6154–6162.
[84] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 961–971.

TINNAKORN KEAWBOONTAN received the B.E. degree in computer engineering from the King Mongkut's Institute of Technology Ladkrabang, Thailand, in 2010. He is currently pursuing the M.Eng. degree in defense technology with Navaminda Kasatriyadhiraj Royal Air Force Academy. He is also an Information Technology and Educational Equipment Officer with Royal Thai Air Force, Bangkok, Thailand. His research interests include machine learning, deep learning, multi-object target tracking, and computer vision.

MASON THAMMAWICHAI (Member, IEEE) was born in Bangkok, Thailand, in 1983. He received the B.S. degree in computer engineering from the University of Wisconsin-Madison, USA, the M.Sc. degree in avionic systems from the University of Sheffield, U.K., and the Ph.D. degree from the Imperial College London, in 2016. He has been a Faculty Member of the Graduate School, Navaminda Kasatriyadhiraj Royal Air Force Academy, since 2016. From 2017 to 2021, he was an assistant professor. Since 2021, he has been an associate professor. His current research interests include mathematical optimization, optimal control, UAV, swarm robotics, intelligence systems, cubesat, satellites, machine learning, deep learning, and cybersecurity.