remote sensing

MDPI

Article

Extracting High-Precision Vehicle Motion Data from Unmanned Aerial Vehicle Video Captured under Various Weather Conditions

Xiaohe Li * and Jianping Wu

Department of Civil Engineering, Tsinghua University, Beijing 100084, China
* Correspondence: lixiaohe20@mails.tsinghua.edu.cn

Abstract: At present, there are many aerial-view datasets that contain motion data from vehicles in a variety of traffic scenarios. However, there are few datasets that have been collected under different weather conditions in an urban mixed-traffic scenario. In this study, we propose a framework for extracting vehicle motion data from UAV videos captured under various weather conditions. With this framework, we improve YOLOv5 (you only look once) with image-adaptive enhancement for detecting vehicles in different environments. In addition, a new vehicle-tracking algorithm called SORT++ is proposed to extract high-precision vehicle motion data from the detection results. Moreover, we present a new dataset that includes 7133 traffic images (1311 under sunny conditions, 961 under night, 3366 under rainy, and 1495 under snowy) of 106,995 vehicles. The images were captured by a UAV to evaluate the proposed method for vehicle orientation detection. In order to evaluate the accuracy of the extracted traffic data, we also present a new dataset of four UAV videos, each having 30,000+ frames, of approximately 3K vehicle trajectories collected under sunny, night, rainy, and snowy conditions, respectively. The experimental results show the high accuracy and stability of the proposed methods.

Keywords: vehicle orientation detection; vehicle tracking; deep learning; vehicle motion data; unmanned aerial vehicles

Citation: Li, X.; Wu, J. Extracting High-Precision Vehicle Motion Data from Unmanned Aerial Vehicle Video Captured under Various Weather Conditions. *Remote Sens.* **2022**, *14*, 5513. https://doi.org/10.3390/rs14215513

Academic Editors: Hemanth Venkateswara, Chengcai Leng and Anup Basu

Received: 15 September 2022
Accepted: 29 October 2022
Published: 1 November 2022

1. Introduction

In the last few years, the use of unmanned aerial vehicles (UAVs) has been increasing, especially for traffic data collection. UAVs that have high-resolution cameras onboard can record traffic from a so-called bird's-eye perspective effectively and flexibly. Furthermore, UAVs can be easily deployed and operated with a low cost of acquisition.

In a traffic scenario, a UAV is usually used to capture traffic video from a bird's eye view. The vehicle motion data extracted from these UAV videos can be used for traffic flow analysis, vehicle motion prediction, behavior analysis and modeling, and driving scenario generation, as well as autonomous driving decision making, algorithm development, and verification [1]. Table 1 lists aerial-view vehicle motion datasets that contain motion data from vehicles in a variety of traffic scenarios. These datasets were collected using different vehicle detection and tracking methods.

In summary, although various aerial-view vehicle motion datasets have been collected using different vehicle detection and tracking algorithms, a few problems remain:

- These datasets were all collected in clear-weather conditions to maximize recording quality and are not focused on weather factors. At present, there are no datasets that contain vehicle motion data collected from UAV video captured under various weather conditions in an urban mixed-traffic scene. The weather and the ambient illumination conditions change constantly in a real traffic scenario, and this can greatly affect the quality of the images. The accuracy and robustness of the vehicle detection and tracking methods significantly decrease under various inclement weather conditions

such as rain, fog, and snow, and at nighttime because of darkness, blurring, and partial occlusion.

- Another major issue with the aforementioned datasets is that the vehicle detection and tracking methods used for these datasets do not obtain highly accurate and stable vehicle motion data, because vehicle detection and tracking in UAV images and videos is challenging for the following reasons: random disturbance of the camera, obstruction from buildings and trees, the existence of many objects against intricate background, and perspective distortion. Raw data extracted from the UAV video need to be smoothed and refined based on other data collected by additional sensors, and they may contain erroneous coordinate, speed, and acceleration information.

Table 1. Different aerial view vehicle motion datasets.

Database	Motion Parameters	Scenarios	Road User Types	Weather	Trajectories and FPS	Locations
NGSIM [2]	Trajectory, speed, acceleration	Freeway, arterial corridor	Car, truck	Sunny	9206; 10 Hz	4
Stanford Drone [3]	Trajectory	Campus	Pedestrian, bicycle, car, cart, bus	Sunny	10,300; 25 Hz	8
highD [4]	Trajectory, speed, acceleration	Highway	Car, truck	Sunny	110,500; 25 Hz	4
DUT [5]	Trajectory, speed, yaw angle	Campus	Pedestrian, vehicles	Sunny	1862; 23.98 Hz	2
INTERACTION [1]	Trajectory, velocity, yaw angle	Intersection	Car, pedestrian, bicycle	Sunny	18,642; 10 Hz	4
inD [6]	Trajectory, speed, acceleration, yaw angle	Intersection	Pedestrian, bicycle, car, truck, bus	Sunny	13,599; 25 Hz	4
roundD [7]	Trajectory, speed, acceleration, yaw angle	Roundabout	Pedestrian, bicycle, car, truck, bus	Sunny	13,746; 25 Hz	3
exiD [8]	Trajectory, speed, acceleration, yaw angle	Highway	Car, van, truck	Sunny	69,172; 25 Hz	7
CitySim [9]	Trajectory, speed, yaw angle	Highway, roundabout, intersection	Vehicles	Sunny	10,000+; 30 Hz	12

To solve these problems, some methods have been proposed to detect vehicles under various weather conditions, which can be summarized as follows:

Some methods based on image preprocessing can improve image quality and remove noise to transform images captured under adverse conditions into normal images. A vehicle detection model based on weather conditions is then selected to detect vehicles in the processed image. The common algorithms for image processing are histogram equalization (HE) [10] and median fil1tering [11], which can improve the contrast of images captured under low illumination conditions. However, some detailed texture information is lost during image preprocessing, and there is no generic solution that works for all adverse weather conditions. Furthermore, supervised learning methods such as the convolutional neural network (CNN) [12] and generative adversarial network (GAN) [13] can be used for UAV image enhancement and denoising but may reduce the real-time performance of vehicle detection.

Although many deep learning-based models have been proposed for detecting vehicles in UAV videos, adverse weather conditions are seldom tested by these models. In order to adapt to different weather conditions, three aspects of these models should be improved [14]. First, some models are divided into two parts: the first part is used to judge the weather type of the input data, and the second part selects the corresponding

model for vehicle detection [15]. Second, some models combine different deep learning networks, such as DSNet [16], RCNN [17], faster RCNN [18], SSD [19], and YOLO [20–22], with image processing methods to directly train on datasets collected under different weather conditions [23,24]. Third, some features of vehicles may disappear in adverse weather conditions. In order to improve accuracy and robustness, vehicle detection models can be combined with multiple feature extraction channels to extract different features from images [25]. The extracted vehicle features include appearance features, local binary patterns (LBPs), Haar-like features, histograms of oriented gradient (HOGs), and speeded up robust features (SURF). Then, classifiers such as support vector machine (SVM) and Bayesian theory are used to train the feature matrix to obtain the vehicle [26].

Although many existing methods are not designed for vehicle detection in adverse weather conditions, they can adapt to different weather conditions through retraining and learning vehicle features from a dataset collected under corresponding weather conditions. Domain adaptation is a representative method of transfer learning [27]. Domain-adapted detectors were trained on a dataset collected under normal weather conditions and then used in a network such as the regional proposal network (RPN) [28], prior estimation network [29], and CycleGAN [30] to transfer the style of the original dataset to generate a new training dataset under different weather conditions. The new training dataset is used to train vehicle detection methods such as faster RCNN so they perform well on cross-domain datasets. However, methods trained on rendered synthetic images generated by adaptive detectors do not perform as well as methods trained on real images. In addition, many large UAV datasets have been collected for training and testing deep learning algorithms, such as the DOTA [31], UA-DETRAC [32], UAVDT [33], and VisDrone [34]. However, these datasets contain few manually labeled ground truth images captured in adverse weather conditions.

To sum up, few driving scenes in multi-weather conditions have been included in published aerial-view vehicle motion datasets, and the existing vehicle detection and tracking methods that can be applied to special scenarios have shown limited performance. Therefore, we propose a new framework for extracting high-precision vehicle motion data from UAV video captured under various weather conditions.

We used a camera-equipped drone to record real-world traffic UAV videos under different weather conditions to create new datasets to test the proposed framework. To summarize, the main contributions of this paper are as follows:

- To the best of our knowledge, this study is the first work that extracts high-precision vehicle motion data, including vehicle trajectory, vehicle speed, and vehicle yaw angle, from UAV video captured under various weather conditions.
- Two new aerial-view datasets, named the Multi-Weather Vehicle Detection (MWVD) and Multi-Weather UAV (MWUAV) datasets, are collected and manually labeled for vehicle detection and vehicle motion data estimation, respectively. The MWVD dataset includes 7133 traffic images (1311 taken under sunny conditions, 961 under night conditions, 3366 under rainy conditions, and 1495 under snowy conditions) of 106,995 vehicles. The data were captured by a camera-equipped drone and will be used to evaluate the proposed vehicle orientation detection method. The MWUAV dataset for data estimation contains four UAV videos having over 30,000 frames of approximately 3000 vehicle trajectories. The speed and yaw angle were collected under sunny, night, rainy, and snowy conditions. This is the first and the largest vehicle motion dataset collected from UAV videos captured at an urban intersection under multi-weather conditions.
- We propose a new vehicle orientation detection method based on YOLOv5 with image-adaptive enhancement to improve vehicle detection performance under various weather conditions. Our method significantly outperforms state-of-the-art methods on the collected MWVD dataset.
- A new vehicle tracking method called SORT++ is proposed to extract high-precision and reliable vehicle motion data from the vehicle orientation detection results. Our

tracking method also achieves state-of-the-art performance on the collected MWUAV dataset.

2. Methodology

2.1. Overall Framework

As shown in Figure 1, our framework contains four steps: video stabilization, vehicle detection, vehicle tracking, and data extraction. Firstly, any random disturbance in the UAV videos is eliminated using a homography matrix obtained by a scale-invariant feature transform (SIFT) operator [35] and K-nearest neighbor (KNN) algorithm. Then, a dual-weight vehicle detection algorithm, namely You Only Look Once version 5 Oriented Bounding Box (YOLOv5-OBB) [36], with a contrast-limited adaptive histogram equalization (CLAHE) method is applied for vehicle orientation detection as the foundation for trajectory construction. After that, the SORT++ algorithm is proposed to track vehicles and obtain motion data from the detection results over time. Finally, vehicle trajectory, vehicle speed, and vehicle yaw angle are extracted from the tracking results based on the vehicle steering model. The details of each step are introduced as follows.

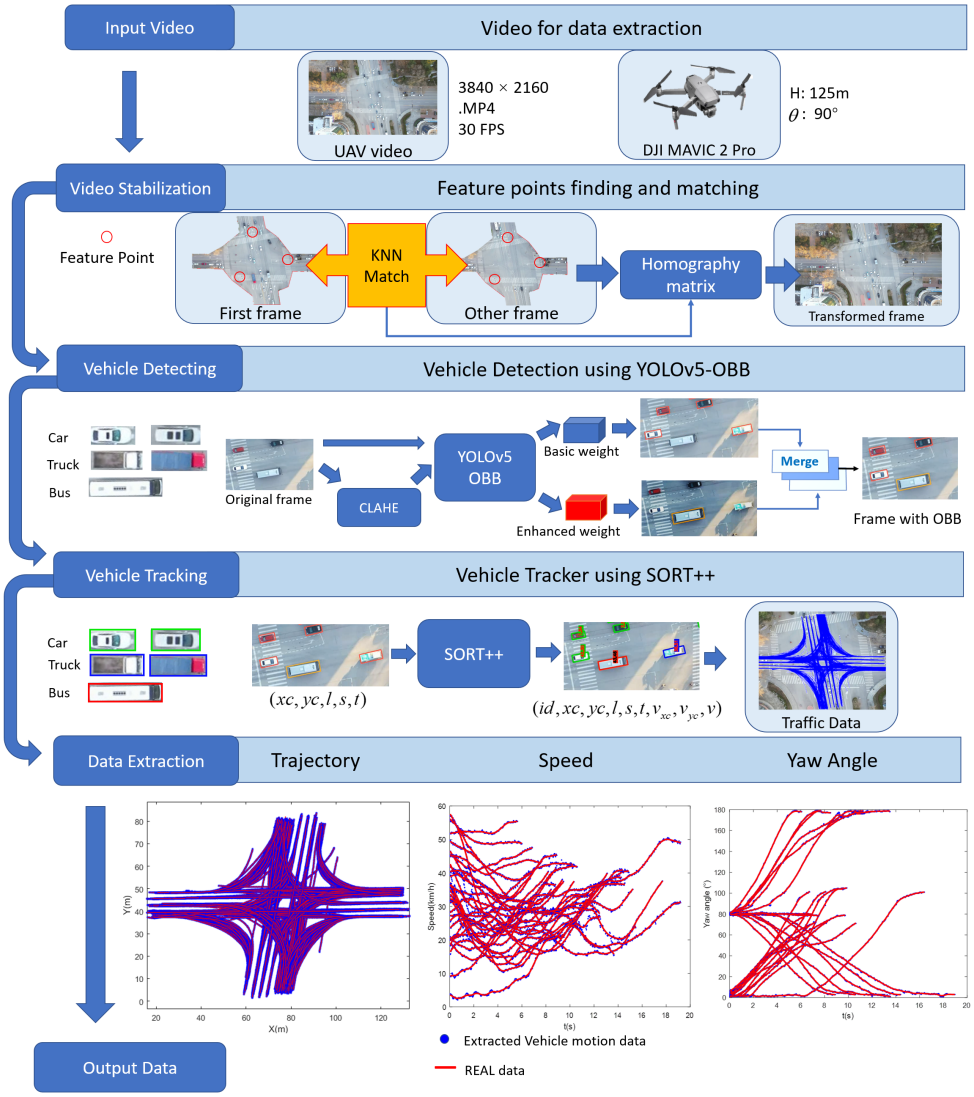

Figure 1. Framework flowchart for extracting vehicle motion data from UAV video.

2.2. Image Stabilization

The camera may shake when a UAV is used to record traffic video because of the effects of airflow, motor vibration, human manipulation, and other factors. The unavoidable movement of the UAV may result in rotation, scaling, and jitter in the video frame, which could then interfere with the movement of the objects in the video and affect the accuracy of the traffic data that are extracted.

In order to solve these problems, the video should be stabilized. First, the start and end frame of the video are extracted, and the roadside area with the buildings and trees is masked. The SIFT operator is then used to find feature points of the first frame and the last frame, respectively. After that, the KNN algorithm is used to match these feature points and obtain good key-point pairs of background in the same geographical location. The feature template is an affine transformation matrix searched for all good key-point pairs according to the minimum sum of errors. Finally, we choose the first frame as the reference frame, and all video frames are affine-transformed using the feature template and quickly aligned to the reference frame. We can obtain a stabilized video by connecting each transformed frame.

2.3. Structure of the Vehicle Orientation Detection Method

YOLO is the name of a popular object detection algorithm, and the most common version at present is YOLOv5, which obtains high accuracy and is able to detect objects in real time [37]. However, it uses the horizontal bounding boxes (HBB) detector, which is not suitable for vehicle orientation detection in a UAV video. Therefore, we improved the YOLOv5 algorithm by using the OBB detectors, which we called YOLOv5-OBB. Moreover, in order to improve the accuracy of vehicle detection in different environments, we proposed a novel dual-weight method consisting of CLAHE-based image enhancement and the YOLOv5-OBB algorithm. As shown in Figure 2, we first use the original UAV dataset and the UAV dataset enhanced by CLAHE to train the YOLOv5-OBB algorithm and obtain the basic weight and enhanced weight files as the training results, respectively. We then select a region of interest (ROI) in the UAV video. We use the original ROI and the same ROI enhanced by CLAHE as the input data for the YOLOv5-OBB algorithm with basic weight and enhanced weight files. In the end, we perform basic and enhanced vehicle detection. We merge these two types of detections, and we obtain the final vehicle detection results after non-maximum suppression (NMS).

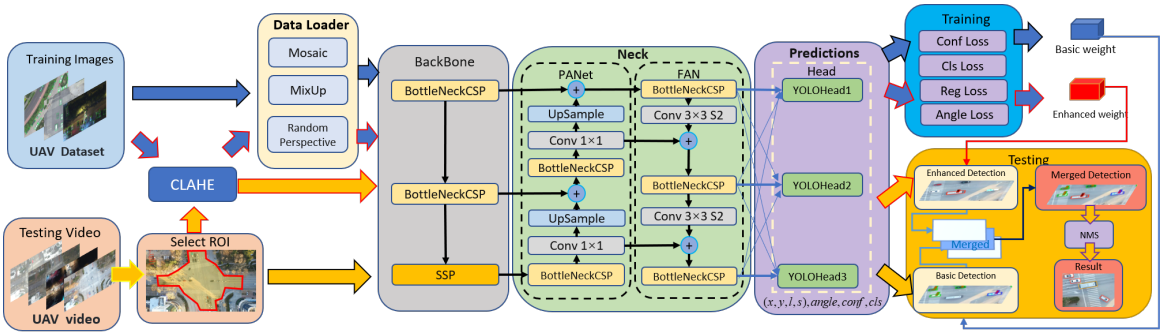

Figure 2. Structure of the dual-weight YOLOv5-OBB algorithm with CLAHE.

2.3.1. CLAHE

The UAV images captured under different weather conditions have different light intensities. As shown in Figure 3, we can clearly see that the gray level with the peak frequency in the gray-level histogram of the UAV image under sunny conditions is within the range (100, 150), the gray level with the peak frequency in the gray-level histogram of the UAV image under foggy conditions is within the range (150, 255), the gray level with the peak frequency in the gray-level histogram of the UAV image under night conditions with strong light is within the range (50, 100), and the gray level with the peak frequency in

the gray-level histogram of the UAV image under weak light conditions at night is within the range (0, 50). Therefore, we can use the gray-level histogram to divide the UAV images into four categories: bright, foggy, weak light at night, and strong light at night.

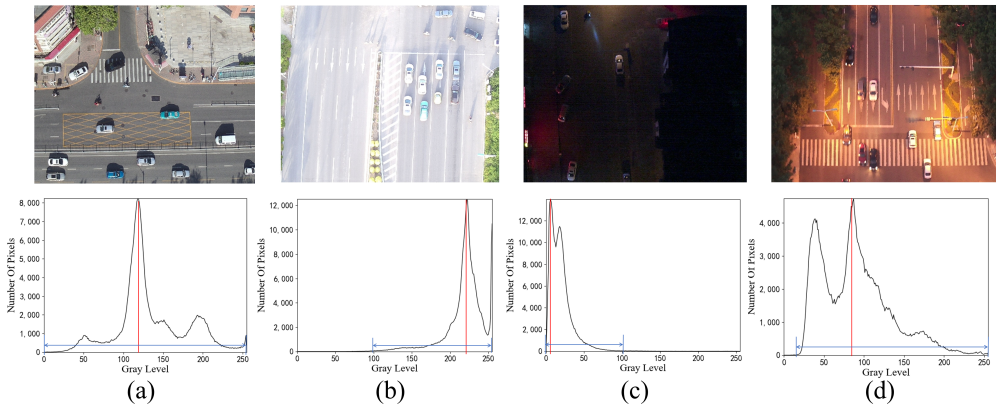

Figure 3. First row shows the UAV images under different illumination conditions: from (**a**–**d**) are bright, foggy, weak light at night, and strong light at night; the second row shows the corresponding gray-level histograms: the red line represents the gray level at peak frequency and the blue line shows the range of gray levels having a frequency greater than 0.

According to the range of gray levels having frequencies greater than 0 in the gray-level histograms of the UAV images, we can see that UAV images in the categories of foggy and weak light at night have poor contrast, and the vehicles in the image are difficult to distinguish from the background. In order to improve detection performance, the contrast in the UAV image should be enhanced by distributing the gray levels over as many ranges as possible to minimize the effects of weather and illumination.

The CLAHE is a local HE method to enhance the contrast in the image, and it is usually combined with an object detection algorithm [38]. CLAHE can solve the problems of the classical HE algorithm, such as excessive contrast enhancement, increased background noise, and the loss of image details.

The implementation process of the CLAHE algorithm can be divided into four steps [39]: (1) separate the image into MxN local tiles; (2) compute the gray-level histogram of each tile separately, and each histogram is redistributed in such a way that its height does not exceed the threshold; (3) traverse each tile and calculate the transformed value of each pixel to compute the linear difference between tiles, and then redistribute the tile pixels; (4) merge all tiles and eliminate the artifacts between tiles by using the bilinear interpolation.

As shown in Figure 4, the four categories of UAV images are enhanced by the CLAHE algorithm. By observing the RGB image, we can see that the contrast in the UAV image is enhanced, and the vehicles in the enhanced images have become more obvious. In contrast, the gray level distribution range in the histogram that is enhanced by the CLAHE algorithm is expanded, and the frequency of the gray levels is homogenized.

2.3.2. YOLOv5-OBB Algorithm

As shown in Figure 2, the YOLOv5-OBB algorithm consists of four parts: data loader, backbone, neck, and prediction. The data loader is the input terminal, which enhances the training images by performing data augmentation. Subsequently, the preprocessed data are sent to the backbone network to extract the general features. The backbone network is made up of several different modules. The cross-stage partial bottleneck (BottleneckCSP) [40] serves as the backbone for obtaining bounding boxes and extracting deep features, and the spatial pyramid pooling technique (SPP) pools the input feature maps to fuse multiple receptive fields and enhance the performance for small target detection. Then, multi-scale characteristics are extracted via a neck network consisting of feature pyramid network

(FPN) and path aggregation network (PANet). The FPN transmits strong semantic features from top to bottom, while the PANet transmits reliable positioning features from bottom to top. The parameters of various detection layers are aggregated from various backbone layers, and they further enhance the capacity of the model for feature extraction. Finally, the category, anchor confidence, yaw angle, and anchor box of the detected objects are predicted by three YOLOHeads in the prediction network.

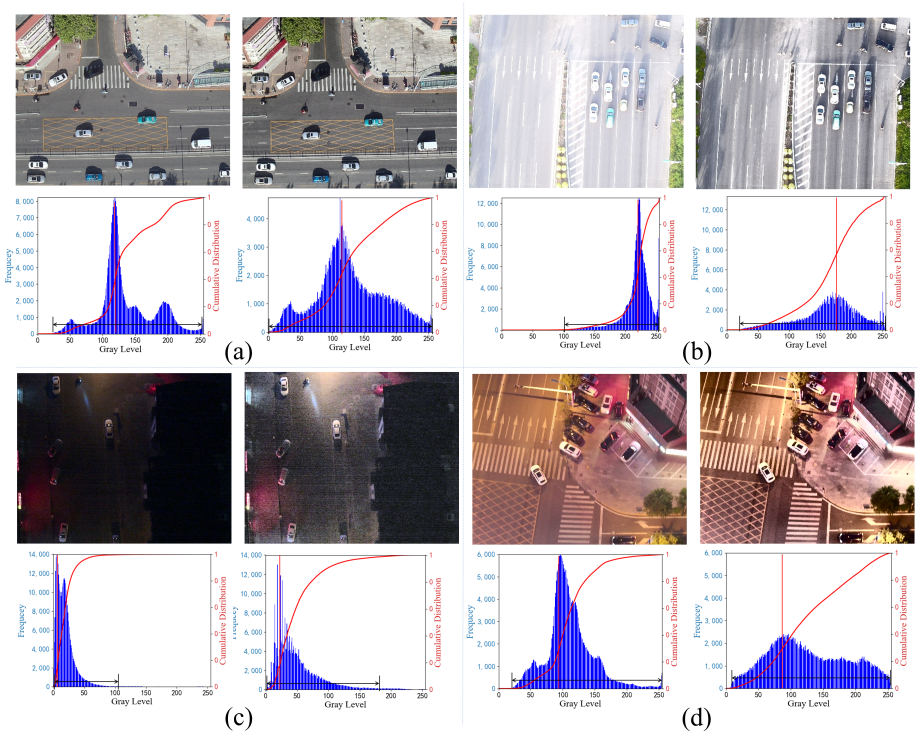

Figure 4. Original image (left) and result of contrast enhancement by CLAHE (right) for four categories of images with their histograms below: (**a**) bright, (**b**) foggy, (**c**) weak light at night, (**d**) strong light at night.

During the training phase, the YOLOHeads calculate the multi-task losses and feedback for the network. The classification loss of categories L_{cls}, regression loss of bounding boxes L_{reg}, and confidence loss of object L_{conf} make up the majority of the loss function for object detection. To obtain the rotation angle of objects, we add angular classification layers to the YOLOv5 prediction head and the classification loss of angle L_{angle} for the YOLOv5-OBB vehicle detection method. The multi-task loss function is given as (1):

$$L_{\text{All}} = \alpha_1 L_{\text{reg}} + \alpha_2 L_{\text{cls}} + \alpha_3 L_{\text{conf}} + \alpha_4 L_{\text{angle}} \tag{1}$$

where hyperparameters $\alpha_1, \alpha_2, \alpha_3$ and α_4 are used to control these losses, and the four losses are defined as follows. Generally, the confidence loss α_3 is the maximum weight, which is followed by the weight of classification loss of categories α_2 and angle α_4, and the regression loss of bounding boxes α_1 is the minimum weight. In our experiments, α_1 = 0.05, α_2 = 0.3, α_3 = 0.7, and α_4 = 0.3. The four losses are defined as follows.

In order to obtain accurate orientation and scale information of the detected objects, we use five parameters (x_c, y_c, l, s, θ) to describe the oriented bounding box, where (x_c, y_c) represent the center point of the oriented bounding box and θ is defined as the acute angle

between long side l and x-axis, and the other short side of the box was defined as short side s. The regression loss of the bounding boxes L_{reg} is calculated by CIoU Loss [41] as follows:

$$L_{\text{CIoU}} = 1 - \left(IOU - \frac{\rho^2(b, b^{gt})}{c^2} - \frac{v}{(1 - IOU) + v} \right), v = \frac{4}{\pi^2} \left(\arctan \frac{s^{\text{gt}}}{l^{\text{gt}}} - \arctan \frac{s}{l} \right)^2 \quad (2)$$

where IOU is defined as the ratio of overlap area and union area between two bounding boxes, which can be estimated according to the percent of interior pixels. c represents the diagonal length of the smallest enclosing box covering the two boxes. $\rho(\cdot)$ is the Euclidean distance between the center points b and b^{gt} of two bounding boxes, and v measures the consistency of the aspect ratio, where the longer side of bounding box is defined as l and the short side is defined as s.

Additionally, we used the idea of classification to regard θ as the category of the detected object and divide θ into 180 classes according to the angular range, and we used the circular smooth label (CSL) in [42] to code the angle ground truth before calculating the angle classification loss, which has a high error tolerance for adjacent angles and alleviates the problem of angle–class imbalance. The binary cross-entropy (BCE) logit loss was used as the classification loss of angle, the confidence loss of the object and the classification loss of categories, and these losses are defined as follows:

$$L_{\text{BCEWithLogits}} = -\sum_{n=1}^{\text{N}} [y_i \cdot \log \sigma(x_i) + (1 - y_i) \cdot \log(1 - \sigma(x_i))] \quad (3)$$

$$\text{CSL}(x) = \begin{cases} \exp\left(-\frac{(x-\theta)^2}{2r^2}\right), \theta - r < x < \theta + r \\ 0, \text{ otherwise} \end{cases} \quad (4)$$

$$L_{\text{cls}} = \sum_{i}^{\text{N}} L_{\text{BCEWithLogits}}(P_{\text{cls}}, T_{\text{cls}}) \quad (5)$$

$$L_{\text{conf}} = \sum_{i}^{\text{N}} L_{\text{BCEWithLogits}}\left(P_{\text{obj}}, T_{\text{obj}}\right) \quad (6)$$

$$L_{\text{angle}} = \sum_{i}^{\text{N}} L_{\text{BCEWithLogits}}\left(P_{\text{angle}}, \text{CSL}(\theta)\right) \quad (7)$$

where N denotes the total number of input samples, while x_i and y_i represent the vector of predicted and ground truth, respectively. θ represents the true angle of the bounding box, and r represents the window radius. The window radius controls the error tolerance between the adjacent angles. P_{cls} and T_{cls} are the predicted probability distribution and corresponding truth probability distribution for all object classes, respectively. T_{obj} is the IoU between the predicted and the ground-truth bounding box and P_{obj} represents the predicted confidence of the bounding box. P_{angle} and $\text{CSL}(\theta)$ indicate the prediction of angle and angle labels encoded by CSL, respectively.

After the YOLOv5-OBB algorithm is trained on the original dataset and the corresponding enhanced dataset, we obtain the basic and enhanced weight files. In the testing phase, we obtain vehicle detections from the UAV video by using these weight files. The prediction network can generate the most trusted detection predictions of objects through NMS.

2.4. The Structure of the Tracking Method

The vehicle tracking task is processed after vehicle detection. The simple online and real-time tracking (SORT) [43] algorithm is generally used with a Kalman filter and Hungarian algorithm. However, SORT is not suitable for vehicle tracking in complicated traffic scenarios and leads to ineffective tracking and inaccurate data. To obtain high-precision vehicle motion data such as the vehicle trajectory, vehicle speed, and vehicle yaw angle, we improve the SORT algorithm based on the detection results to extract high-

precision vehicle motion data from the UAV video. We call our algorithm SORT++. The overall flow of this algorithm is shown in Figure 5.

First, the SORT++ algorithm uses the boxes filter (BF) to calculate the rotated intersection over union (RIoU) of oriented detection bounding boxes obtained from the YOLOv5-OBB algorithm. These boxes are divided into high score detection boxes, low score detection boxes, and overlapped detection boxes according to the anchor confidence and RIoU values. Afterward, the second RIoU matching is used to match the high score detection boxes over the current frame with the predicted tracks from the Kalman filter. Subsequently, the unmatched tracks are matched with low score detection boxes by the third RIoU matching, and unmatched detection boxes after the second and third RIoU matching are sent to the appearance feature extractor (AFE) to determine whether they are bounding boxes of vehicles. Then, all the unmatched tracks, matched tracks, and new unconfirmed tracks are classified as vehicle tracks or other tracks by the motion feature extractor (MFE). Finally, the parameters of the vehicle tracks are updated and divided into confirmed vehicle tracks and unconfirmed vehicle tracks by the Kalman filter. We can obtain vehicle data such as ID, image coordinates, speed, vehicle size, and yaw angle from the confirmed vehicle tracks.

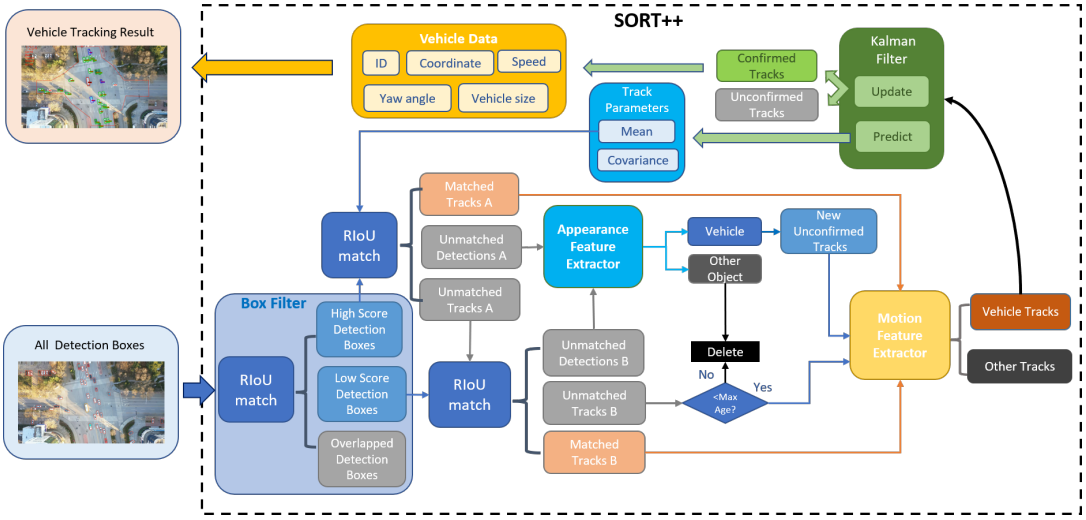

Figure 5. Vehicle tracking process using the SORT++ algorithm.

2.4.1. RIoU Matching

In order to determine whether two detected objects are overlapped and to match the detected objects with tracks, we use RIoU matching [44]. As shown in Figure 6, RIoU is the ratio of the overlap to the union of the two OBBs. RIoU represents the accuracy of the prediction model, and the RIoU value of the same vehicle is approximately 1, which shows the least prediction error. A threshold is set for RIoU matching. If the RIoU values exceed the threshold, the Kalman predicted track and unmatched detections of the current frame have been successfully associated.

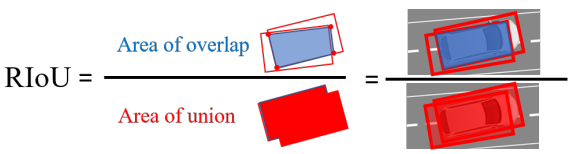

Figure 6. Calculation of RIoU.

2.4.2. Appearance Feature Extractor (AFE)

The output of the vehicle detection algorithm usually contains the false object detection results, such as the shadows and highlights of objects in the scene, non-vehicle objects that are vehicle-like in appearance, and some non-motor vehicles. In order to eliminate

these false detections effectively, we propose the appearance feature extractor based on the Canny edge detector [45].

As shown in Figure 7, we first calculate the length–width ratio and area of the detection boxes. We remove the boxes having a length–width ratio larger than the maximum length–width ratio $R_{\max}$ of vehicles and an area smaller than the value of the heavy vehicle $A_{\max}$, because the detection boxes of vehicles are generally rectangular. Then, we calculate the gray value of the image blocks clipped by the detection boxes and remove the detection boxes with a gray value smaller than the minimum gray value of vehicles $G_{\min}$. After that, we use the Canny edge detector to find vehicle contours in the image blocks clipped by the detection boxes. The boxes having an area of detected contours that is larger than the minimum contour area of vehicle $A_{\min}$ are true vehicle detection boxes. All of these parameters are set according to the input data and vehicle parameters shown in Table 2.

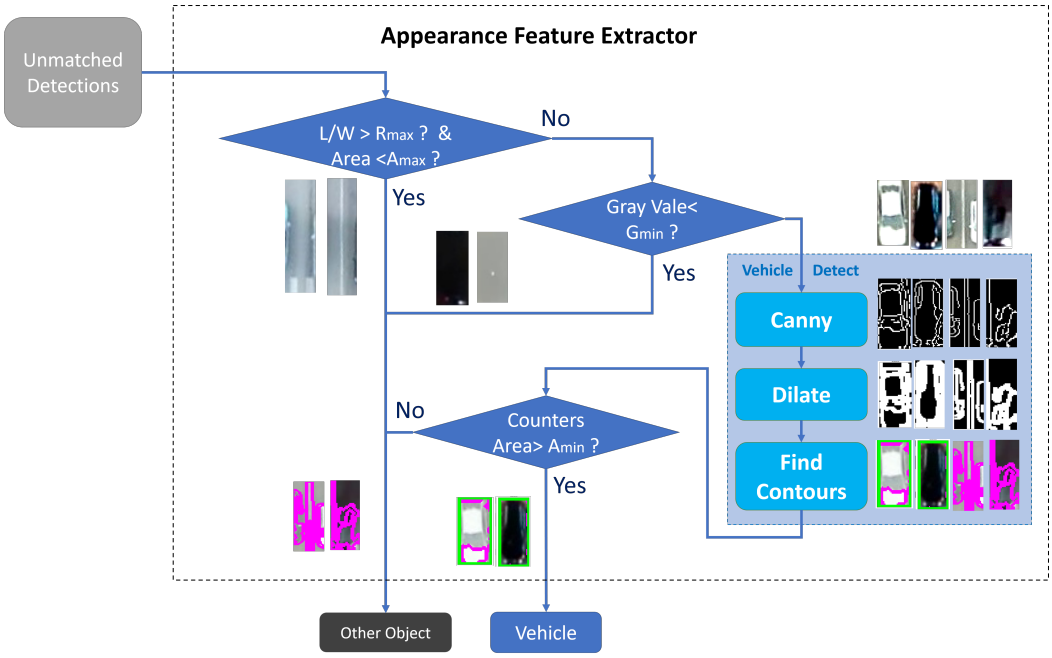

Figure 7. Flowchart for using the appearance feature extractor.

2.4.3. Motion Feature Extractor (MFE)

Vehicle motion is constrained by fixed parameters, as shown in Table 2. Lateral motion and velocity cannot change abruptly. Therefore, the motion feature of vehicle tracks should satisfy the following constraints:

$$d_i = \sqrt{d_{\mathrm{x},i}^2 + d_{\mathrm{y},i}^2} < d_{\max} = v_{\max} \cdot t_{\mathrm{frame}} \tag{8}$$

$$d_{\mathrm{x},i} = |x_i - x_{i-1}| < d_{\mathrm{x,max}} = d_{\max} \cdot \cos\varphi \tag{9}$$

$$d_{\mathrm{y},i} = |y_i - y_{i-1}| < d_{\mathrm{y,max}} = d_{\max} \cdot \sin\varphi \tag{10}$$

where φ represents the acute angle between the x-axis and the velocity of the vehicle. x_i and y_i represent the center coordinates of the vehicle at time i, while $d_{\mathrm{x},i}$ and $d_{\mathrm{y},i}$ represent the displacement of the vehicle along the x-axis and y-axis, which should smaller than the maximum displacement $d_{\mathrm{x,max}}$ and $d_{\mathrm{y,max}}$ along the x-axis and y-axis, respectively. The total displacement d_i is also smaller than the maximum displacement of the vehicle, $d_{\max}$, and $d_{\max}$ is the product of the maximum vehicle speed $v_{\max}$ and frame time t_{frame}.

$$\Delta\theta_i = |\theta_i - \theta_{i-1}| < \theta_{\max} = \mathrm{FPS} \cdot \arctan\left(\frac{l}{R_{\min}}\right) \tag{11}$$

When the vehicle is turning, the change in yaw angle $\Delta\theta_i$ of the vehicle between time i and time $i-1$ should be smaller than the maximum change in vehicle yaw angle $\theta_{\max}$. l is the wheelbase of the vehicle, and $R_{\min}$ represents the minimum turning radius of the vehicle. Frame time is the inverse of frame per second (FPS), which is 1/30 s in our experiment.

$$\Delta A_i = 1 - \left| \frac{\text{Area}_i}{\text{Area}_{i-1}} \right| = 1 - \left| \frac{l_i \cdot s_i}{l_{i-1} \cdot s_{i-1}} \right| < 0.1 \tag{12}$$

Vehicle body dimensions are fixed. The area Area_i of the OBB at time i should be approximately equal to the area Area_{i-1}at previous time $i-1$, so the change range ΔA is smaller than 10%.

Table 2. Main parameters of vehicles.

Parameters	Mini	Car	Truck	Bus
Vehicle length (mm)	<4000	<5200	<12,000	<18,000
Vehicle width (mm)	<1600	<2000	<2500	<2500
Vehicle length–width ratio	<2.5	<3	<6	<6
Minimum turning radius (mm)	3.5∼5.0	4.5∼7.5	4.0∼10.5	4.0∼11.0
Speed limit (km/h)	<50	<70	<60	<60

2.4.4. Kalman Filtering

Kalman filtering is one of the core algorithms in the SORT++ algorithm, and it is used to predict and update the tracking target parameters based on the target's motion state [46]. The target model refers to the motion model that transfers the identity information of the target to the next frame. The Kalman filtering algorithm uses a linear isokinetic model that approximates the displacement of each object from the current to the next frame, independently of other targets.

A nine-dimensional space vector $\left(x_c, y_c, a, l, \theta, \dot{x}_c, \dot{y}_c, \dot{a}, \dot{l}\right)$ is used to represent the state quantity x of the target at a particular instant, where (x_c, y_c) and a represent the center coordinates and the area of the predicted OBB, respectively. l denotes the length of the predicted target frame of a vehicle. θ indicates the yaw angle between l and the x-axis. (x_c, y_c, a, l, θ) is the observed quantity z, and $\left(\dot{x}_c, \dot{y}_c, \dot{a}, \dot{l}\right)$ represent the velocity information of (x_c, y_c, a, l) relative to the image coordinates.

The standard Kalman filter for the SORT algorithm is based on the linear motion hypothesis. At each step t, the time update equation of the Kalman filter algorithm is as follows:

$$\hat{x}_{t|t-1} = \mathbf{F}_t \hat{x}_{t-1|t-1} + \gamma_t, \mathbf{F} = \begin{bmatrix} \mathbf{I}_{5\times5} & \begin{bmatrix} \mathrm{d}t \cdot \mathbf{I}_{4\times4} \\ \mathbf{O}_{1\times4} \end{bmatrix} \\ \mathbf{O}_{4\times5} & \mathbf{I}_{4\times4} \end{bmatrix} \tag{13}$$

$$z_t = \mathbf{H}_t x_t, \mathbf{H} = \begin{bmatrix} \mathbf{I}_{5\times5} & \mathbf{O}_{5\times4} \end{bmatrix} \tag{14}$$

$$\mathbf{P}_{t|t-1} = \mathbf{F}_t \mathbf{P}_{t-1|t-1} \mathbf{F}_t^{\mathrm{T}} + \mathbf{Q}_t \tag{15}$$

The state transition equation predicts the prior estimate of state x. The observation equation obtains the observation matrix z, $\mathbf{P}$ is the covariance state matrix, $\mathbf{F}$ is the state transition model, $\mathbf{H}$ is the measurement matrix, γ_t is the random error of process at time k, and $\gamma_k \sim \mathcal{N}(0, Q)$ is subject to zero-mean Gaussian distribution. $\mathbf{Q}$ represents the covariance of process excitation noise.

If an observation z_t is provided on step t, the Kalman filter algorithm calculates the posteriors with the following state update equation:

$$\mathbf{K}_t = \mathbf{P}_{t|t-1} \mathbf{H}_t^{\mathrm{T}} \left(\mathbf{H}_t \mathbf{P}_{t|t-1} \mathbf{H}_t^{\mathrm{T}} + \mathbf{R}_t \right)^{-1} \tag{16}$$

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + \mathbf{K}_t \left(z_t - \mathbf{H}_t \hat{x}_{t|t-1} \right) \tag{17}$$

$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\mathbf{P}_{t|t-1} \tag{18}$$

where **R** represents the covariance of measurement error; to improve the accuracy of tracking when the vehicle detection method has high accuracy, we set the value of measurement error **R** close to zero in the experiment.

2.4.5. Traffic Data Calculation

As shown in Figure 8, we extract traffic data from the OBBs of the matched detected objects having the same ID. The parameters of the OBB are $(x_c, y_c, a, l, \theta, \dot{x}_c, \dot{y}_c)$. We consider the center of the bounding box (x_c, y_c) as the center of gravity of the vehicle, and by connecting the centers of gravity of all identified vehicles having the same ID, the vehicle trajectories can be obtained. l, s, and θ represented the length, width, and yaw angle of the vehicle, respectively.

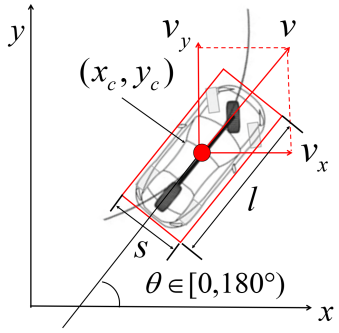

Figure 8. Vehicle steering model and OBB.

To extract the instantaneous speed of a vehicle, we use the velocity information $(\dot{x}_c, \dot{y}_c)$ of the OBB as the vehicle speed components (v_x, v_y) along the x- and y-axes. Considering the correlation between the extracted data points, we use linear regression to eliminate the detection error and improve the accuracy of the data. The instantaneous speed of the vehicle is calculated by the following formula:

$$v_{x,i} = \frac{\sum_{k=i-C}^{i+C}(x_k - \bar{x}) \cdot (t_k - \bar{t})}{\sum_{k=i-C}^{i+C}(t_k - \bar{t})^2} \tag{19}$$

$$v_{y,i} = \frac{\sum_{k=i-C}^{i+C}(y_k - \bar{y}) \cdot (t_k - \bar{t})}{\sum_{k=i-C}^{i+C}(t_k - \bar{t})^2} \tag{20}$$

$$v_i = \frac{\sqrt{v_{x,i}^2 + v_{y,i}^2}}{\text{ppm}} \tag{21}$$

where $v_{x,i}$ and $v_{y,i}$ are the vehicle speed components along the x-axis and y-axis at time t_i. v_i is the vehicle's instantaneous speed at time t_i. C is the fixed time interval, and ppm is the number of pixels per meter.

3. Experiments and Analysis

3.1. MWVD and MWUAV Datasets

At present, there are many aerial-view datasets that have been manually collected for training and evaluating deep learning-based methods in vehicle detection and tracking, such as CARPK [47], COWC [48], CyCAR [49], DOTA [31], EAGLE [50], UA-DETRAC [32], UAV123 [51], UAVDT [33], UAVid [52], VEDAI [53], VisDrone [34], and DLR 3K [54]. However, these datasets have few manually labeled ground truth images captured under different weather conditions in an urban mixed-traffic scene. Therefore, we used a camera-equipped drone to record real-world traffic under different weather conditions to test

the proposed framework. The entire process of the data collection is shown in Figure 9. The UAV videos were collected at urban intersections in China under different weather conditions. An experiment vehicle installed with the high-precision differential positioning sensor WTRTK-M using the real-time kinematic difference global positioning system (RTK-GPS) was selected to collect high-precision vehicle GPS differential positioning data.

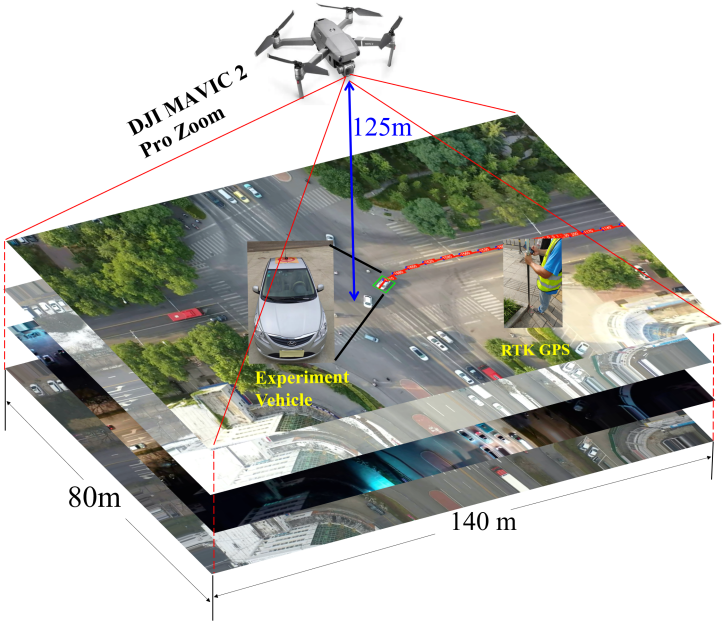

Figure 9. Process of collecting UAV video at urban intersections under different weather conditions.

During the data collection, the research team captured UAV videos for testing using a high-definition camera mounted on a UAV (DJI Mavic 2 Professional Zoom). To obtain accurate plane coordinates, the UAV hovered 125 m overhead and the camera angle was set perpendicular to the ground. The related parameters of the UAV are shown in Table 3. The test videos were captured at 30 frames per second (fps) and with a resolution of 3840 × 2160 pixels under different weather conditions. In the actual experiment of collecting UAV videos, we found that the rainwater and snowflake affect the video image and motors of the UAV, so the UAV videos are collected when rain or snow stops.

Table 3. Parameters of DJI MAVIC 2 Pro Zoom.

Parameter	Value
Weight	430 g
Maximum endurance mileage	10 km
Maximum hover time	20 min
Maximum take-off altitude	5000 m
Maximum wind speed	10 m/s
Operative temperature	From 0 °C to 40 °C
Camera resolution	3840 × 2160, 24/25/30 fps
Equivalent focal length	24 mm

In order to evaluate the proposed vehicle orientation detection method, we selected 7133 frames from the UAV videos collected under different weather conditions to build a dataset. This dataset was named the Multi-Weather Vehicle Detection (MWVD) dataset. All vehicles in the dataset were manually labeled for training and testing the vehicle detection of the deep learning-based algorithm. The MWVD dataset includes 7133 traffic images

(1311 under sunny conditions, 961 under night conditions, 3366 under rainy conditions, and 1495 under snowy conditions) of 106,995 vehicles (3399 of mini cars, 94,610 of cars, 4205 of trucks, 4781 of buses), captured by a UAV in five traffic scenes, including four intersections and one roundabout. The details of the MWVD dataset used in the experiment are shown in Table 4.

Table 4. Details of the MWVD dataset used in the experiment.

Video Data	Vehicle Number	Image Number	Date	Image Size
Sunny	28,773	1311	7:30 01/12/2021	1920 × 1080
Night	35,905	961	17:30 01/12/2021	1920 × 1080
Rainy	19,501	3366	13:00 10/01/2022	1920 × 1080
Snowy	22,816	1495	13:30 22/01/2022	1920 × 1080

Another new dataset for vehicle motion data estimation was collected from the UAV videos captured at three intersections under different weather conditions, and it is denoted as the Multi-Weather UAV (MWUAV) dataset in this paper. As shown in Table 5, four 30,000+ frame UAV videos of approximately 3K vehicle trajectories were collected under sunny, night, rainy, and snowy conditions, respectively. For the ground-truth vehicle trajectory, we have used an experiment car with RTK GPS to compare trajectories. As shown in Figure 9, a test vehicle installed with the high-precision global positioning system (GPS) differential positioning sensor WTRTK-M was selected to collect high-precision vehicle GPS differential positioning data, including the longitude and latitude. For the ground-truth vehicle yaw angle and speed, the change in yaw angle and moving distance of each vehicle on the road were manually labeled over a time interval of five consecutive frame pairs [55]. The time interval was 0.167 s in the collected UAV videos, so the vehicle yaw angle and speed of each vehicle could be considered as a constant in this instant period.

Table 5. Details of four UAV videos from the MWUAV dataset that were used in the experiment.

Video Data	Vehicle Number	Frame Number	Date	Time Length	Video Size	Frame Rate
Sunny	893	30,303	7:30 01/12/2021	16 min 51 s	1920 × 1080	29.97 fps
Night	1021	36,010	18:00 01/12/2021	20 min 1 s	1920 × 1080	29.97 fps
Rainy	709	34,585	13:30 10/01/2022	19 min 13 s	1920 × 1080	29.97 fps
Snowy	470	36,026	13:00 22/01/2022	20 min 2 s	1920 × 1080	29.97 fps

In order to test the performance of the proposed vehicle tracking algorithm, we also select four test video clips from the MWUAV dataset. The details of these videos are shown in Table 6.

Table 6. Details of the four test video clips in the collected MWUAV dataset.

Video Data	Vehicle Number	Frame Number	Date	Time Length	Video Size	Frame Rate
TEST0 (Sunny)	57	1311	7:30 01/12/2021	43 s	1920 × 1080	29.97 fps
TEST1 (Night)	61	961	17:30 01/12/2021	32 s	1920 × 1080	29.97 fps
TEST2 (Rainy)	37	3366	13:00 10/01/2022	112 s	1920 × 1080	29.97 fps
TEST3 (Snowy)	33	1495	13:30 22/01/2022	49 s	1920 × 1080	29.97 fps

3.2. Detection Algorithm Evaluation Metrics

Precision, recall, and Mean Average Precision (mAP) are metrics to evaluate the performance of object detection algorithms, and they can be calculated as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{22}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{23}$$

$$\text{mAP} = \frac{1}{n} \sum_{m=0}^{n} \int_0^1 P_m(R_m) dR_m \tag{24}$$

where TP (True Positive), FP (False Positive), and FN (False Negative) represent the total number of correct detections of vehicles, incorrect detections of non-vehicle objects, and non-detections of vehicles for each frame in the UAV video, respectively. The mAP was calculated using the precision and recall metrics of each detection classification, and n is the number of classes used for object detection.

3.3. Metrics for Multiple Object Tracking (MOT)

For single-camera MOT, we use the same evaluation metric as the MTT challenge [56]. The multiple object tracking accuracy (MOTA) is a common metric for evaluating vehicle tracking accuracy, which focuses on detection performance and is calculated as follows:

$$\text{MOTA} = 1 - \frac{\sum_{t=1}^{\text{T}} (FP_t + FN_t + IDSW_t)}{\sum_{t=1}^{\text{T}} GT_t} \tag{25}$$

where T is the total number of video frames, GT_t, FP_t, FN_t and $IDSW_t$ are the number of real vehicles, non-vehicles, missed vehicles, and the number of identity switches in frame t, respectively.

The Identification F-Score (IDF1) is another commonly used metric for examining the continuity of tracking and the accuracy of recognition, and it focuses on the length of time that the tracking algorithm tracks a target. After the vehicle track is established and assigned an ID, we calculate the number of correct tracks for vehicles: IDTP (True Positive IDs); missed tracks for vehicles: IDFN (False Negative IDs); and wrong tracks for vehicles: IDFP (False Positive IDs). Based on these parameters, we calculate Identification Precision (IDP) as:

$$\text{IDP} = \frac{\text{IDTP}}{\text{IDTP} + \text{IDFP}} \tag{26}$$

and Identification Recall (IDR) as:

$$\text{IDR} = \frac{\text{IDTP}}{\text{IDTP} + \text{IDFN}} \tag{27}$$

IDF1 is the harmonic mean of IDP and IDR:

$$\text{IDF1} = \frac{2}{\frac{1}{\text{IDP}} + \frac{1}{\text{IDR}}} = \frac{2 \cdot \text{IDTP}}{2 \cdot \text{IDTP} + \text{IDFP} + \text{IDFN}} \tag{28}$$

Furthermore, in order to evaluate the percentage of the ground-truth vehicle trajectory that is recovered by the proposed vehicle tracking algorithm, we use mostly tracked (MT) to count the number of mostly tracked (more than 80% of the frames) trajectories for the vehicles.

3.4. Results of Vehicle Detection

To obtain a better training model, we combined our MWVD dataset with the DroneVehicle dataset. The DroneVehicle dataset is made up of 56,878 RGB and infrared images that were collected by the drone in various traffic scenes, including highways, parking lots, intersections, and other places. In addition, the size of each image is 840 × 712. There are five categories of vehicles in this dataset, including car, van, truck, bus, and freight car [57]. We selected 9830 aerial-view UAV images from the combined DroneVehicle and MWVD dataset to build a mixed dataset.

We compared our proposed vehicle detection method with state-of-the-art methods, including RetinaNet by [58], Faster R-CNN by [59], Mask R-CNN by [60], and RoITrans-

former by [61], on the mixed dataset under the same settings. We implemented these methods on a server with one GeForce RTX 2070s and 16 GB total memory. The results are shown in Table 7. Compared with other methods, our dual-weight YOLOv5-OBB method achieves superior performance with the highest mAP and accuracy for each category, and the mAP of the YOLOv5-OBB method using our basic weight file and enhanced weight file is also higher than other methods. To summarize, the proposed dual-weight YOLOv5-OBB method obtained higher vehicle detection performance than the above methods in UAV images captured under various weather conditions.

Table 7. Evaluation on mixed dataset (MWVD and DroneVehicle datasets).

Method	Car AP	Truck AP	Bus AP	mAP
RetinaNet (OBB)	64.27	30.53	61.02	51.94
Faster R-CNN (OBB)	67.09	42.98	66.72	58.93
Mask R-CNN (OBB)	68.50	44.31	66.91	59.91
RoITransformer (OBB)	71.28	49.02	71.15	63.82
YOLOv5-OBB with basic weight	82.20	62.21	81.52	75.31
YOLOv5-OBB with enhanced weight	85.12	69.07	84.10	80.43
Dual weight YOLOv5-OBB (Ours)	**89.10**	**72.07**	**87.55**	**82.91**

The visualized detection results of the YOLOv5-OBB algorithm with basic weight and enhanced weight on the MWVD dataset are shown in Figure 10. The area selected by the red dashed box in Figure 10a,b represents the white-colored vehicle on a snowy day, and some black-colored vehicles in a dark region are missed with basic weight but detected with enhanced weight. One reason for this may be that some vehicles are visually similar to the background road in low-contrast UAV images, which makes it difficult for the vehicle detection algorithm to distinguish them. In Figure 10d, the area selected by the blue dashed box in the figure represents missed detection with enhanced weight because the illumination conditions in different areas are also different and some bright areas are excessively enhanced by CLAHE, which blurs the vehicle outline. Therefore, we use the proposed dual-weight YOLOv5-OBB to merge these detections and reduce false negative errors.

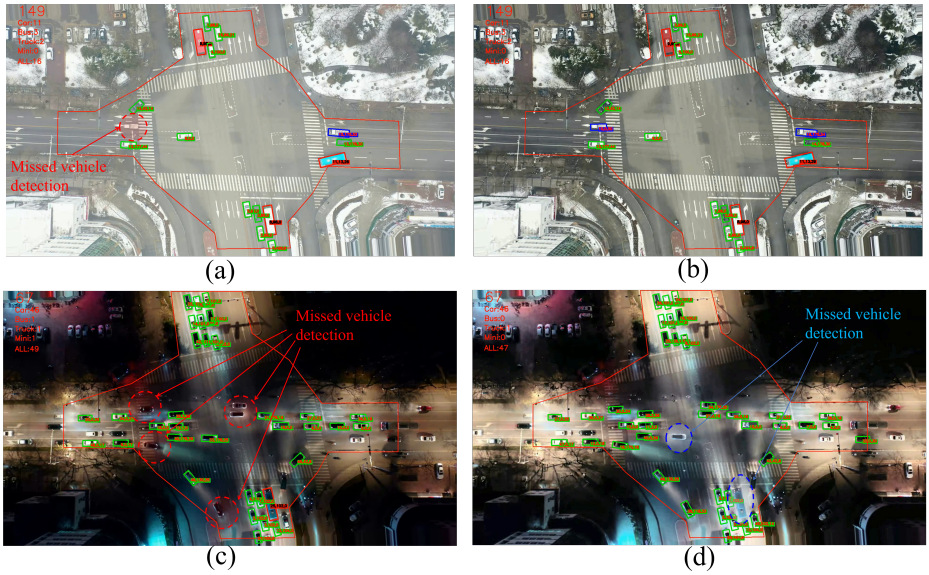

Figure 10. Detection results of the YOLOv5-OBB algorithm with basic weight and enhanced weight on MWVD dataset. (**a**,**c**) show the detection results of the YOLOv5-OBB with basic weight in UAV images captured on a snowy day and at night, respectively. (**b**,**d**) show the detection results of the YOLOv5-OBB with enhanced weight in the enhanced same images.

3.5. Results of Vehicle Tracking

To evaluate the tracking accuracy, we manually marked the position and category of vehicles in the four test videos as test sets, and we calculated MOT metrics. We compared the proposed vehicle tracking method SORT++ with other popular object tracking methods, including SORT by [43], DeepSORT by [62], ByteTrack by [63], and OC-SORT by [64] on the four test sets Test0 to Test4 in the MWUAV dataset. To evaluate the contributions of the box filter (BF), appearance feature extractor (AFE), and motion feature extractor (MFE) in SORT++, we apply them to other methods and test them using the four test sets. The results are shown in Tables 8–11, respectively. We select SORT as our baseline method because all these methods adopt the Kalman filter to predict object motion, and SORT uses the simplest construct. Our proposed SORT++ method achieved superior performance with the highest MOTA and IDF1 metric on all test videos, and it reduced FP and IDSW to 0, which indicated that the proposed vehicle tracking method was stable and accurate under different weather conditions. Furthermore, we can see that BF significantly decreases the FP and IDSW of SORT and ByteTrack, and MFE further decreases the FP and IDSW. AFE in SORT++ can decrease the FP and IDSW to nearly 0.

Table 8. Comparison of the state-of-the-art methods on the Test0 (Sunny) set.

Method	MOTA↑	IDF1↑	FP↓	FN↓	IDSW↓	MT↑
SORT	85.62	91.87	2620	1,494	24	48
DeepSORT	85.16	80.30	783	3302	186	43
ByteTrack	84.12	91.97	4471	80	19	56
OC-SORT	88.36	93.25	1815	1509	25	48
OC-SORT+ByteTrack	80.33	95.39	5527	112	20	55
SORT+BF	94.16	96.65	232	1445	3	48
ByteTrack+BF	99.13	99.16	248	0	1	57
SORT+MFE+BF	94.95	97.04	2	1448	3	48
ByteTrack+MFE+BF	100.00	99.59	0	0	1	57
SORT++ (Ours)	**100.00 (+14.38)**	**99.59 (+7.72)**	**0**	**0**	**0**	**57**

Table 9. Comparison of the state-of-the-art methods on the Test1 (Night) set.

Method	MOTA↑	IDF1↑	FP↓	FN↓	IDSW↓	MT↑
SORT	86.20	92.46	4231	698	26	58
DeepSORT	87.96	92.94	2577	1704	42	55
ByteTrack	79.54	89.18	7249	56	43	61
OC-SORT	87.12	93.27	3891	713	22	58
OC-SORT+ByteTrack	84.08	91.41	5621	61	35	61
SORT+BF	96.87	98.23	205	917	3	57
ByteTrack+BF	98.99	99.27	352	0	1	61
SORT+MFE+BF	97.32	98.46	30	932	1	57
ByteTrack+MFE+BF	99.82	99.91	60	4	0	61
SORT++ (Ours)	**99.99 (+13.79)**	**99.99 (+7.54)**	**0**	**4**	**0**	**61**

Moreover, we tested the proposed SORT++ method on the MWUAV dataset and calculated the MOT metrics. The visual results are shown in Figure 11, and the MOT metrics are shown in Table 12. The MOTA and IDF1 of our proposed SORT++ method were higher than 99.6%, and the MT was basically equal to GT, which indicates that our proposed vehicle-tracking method can track vehicles accurately and robustly in UAV videos captured under various weather conditions.

Table 10. Comparison of the state-of-the-art methods on the Test2 (Rainy) set.

Method	MOTA↑	IDF1↑	FP↓	FN↓	IDSW↓	MT↑
SORT	90.22	94.98	1392	498	17	35
DeepSORT	81.51	68.20	750	2693	163	26
ByteTrack	83.36	91.81	3108	109	29	37
OC-SORT	91.30	95.48	1174	508	15	35
OC-SORT+ByteTrack	85.51	92.74	2689	109	29	37
SORT+BF	95.92	97.95	273	522	0	35
ByteTrack+BF	94.84	97.48	1007	0	0	37
SORT+MFE+BF	96.57	98.27	139	530	0	35
ByteTrack+MFE+BF	95.65	97.87	849	0	0	37
SORT++ (Ours)	**99.97 (+9.75)**	**99.98 (+5.01)**	**0**	**6**	**0**	**37**

Table 11. Comparison of the state-of-the-art methods on the Test3 (Snowy) set.

Method	MOTA↑	IDF1↑	FP↓	FN↓	IDSW↓	MT↑
SORT	91.93	96.06	1473	369	0	30
DeepSORT	80.59	87.85	821	3600	8	26
ByteTrack	89.44	94.72	2355	45	9	33
OC-SORT	93.55	96.83	1100	372	0	30
OC-SORT+ByteTrack	93.02	96.33	1535	50	8	33
SORT+BF	95.97	97.96	213	706	0	29
ByteTrack+BF	98.22	99.12	407	0	0	33
SORT+MFE+BF	96.84	98.39	9	713	0	29
ByteTrack+MFE+BF	99.36	99.68	136	11	0	33
SORT++ (Ours)	**99.99 (+8.06)**	**99.99 (+3.93)**	**0**	**3**	**0**	**33**

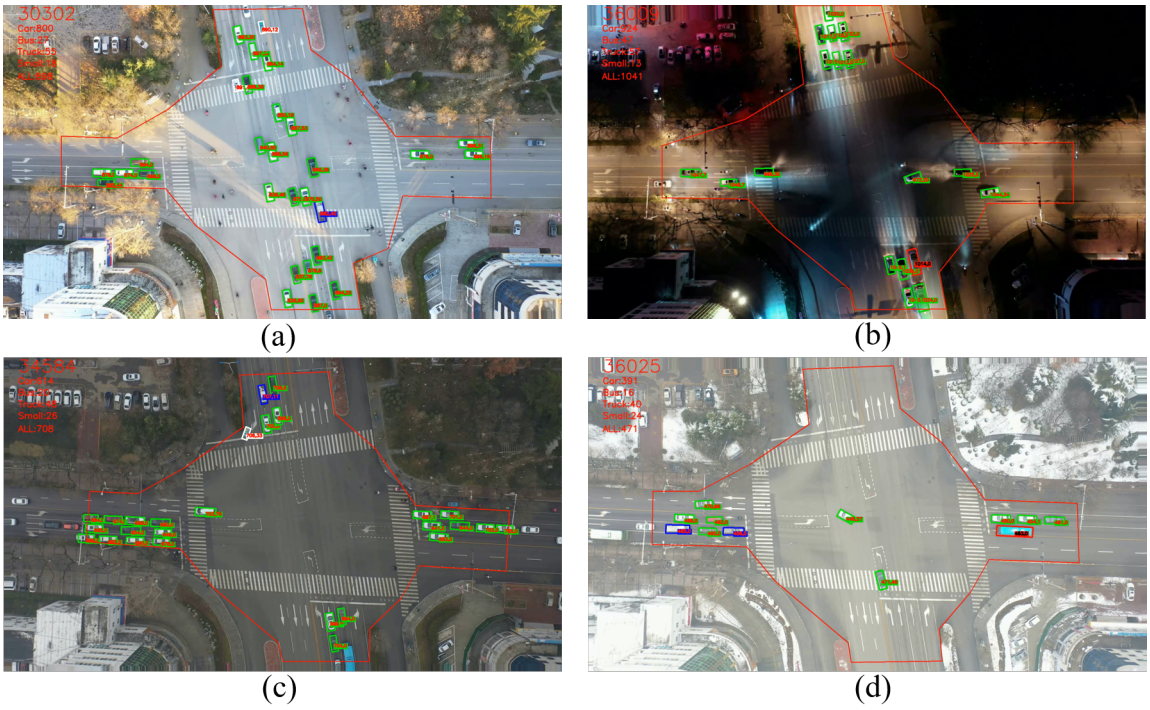

Figure 11. Tracking results of SORT++ on the MWUAV dataset. (**a**) Sunny Data. (**b**) Night Data. (**c**) Rainy Data. (**d**) Snowy Data.

Table 12. MOT Results of SORT++ on the MWUAV dataset.

Data Type	MOTA	IDF1	FP	FN	IDSW	MT	GT	FPS
Sunny Data	100.00	99.94	5	11	4	893	893	8.57
Night Data	99.97	99.61	139	128	5	1021	1021	9.21
Rainy Data	99.99	99.86	14	25	0	708	709	9.73
Snowy Data	99.95	99.89	1	211	2	469	470	10.02

3.6. Results of Vehicle Motion Data Estimation

The precision and continuity of these extracted vehicle motion data are primarily dependent on the performance of the vehicle detection and tracking method. Based on the results of the aforementioned vehicle detection and tracking method discussed in Sections 3.4 and 3.5, we extracted the vehicle motion data, such as vehicle trajectories, vehicle speed, and vehicle yaw angle, by using the proposed framework. The precision of these extracted vehicle motion data was calculated using MATLAB 2020b.

To evaluate the precision of the extracted data, we calculated absolute errors between the test value x_i and true value y_i using Root Mean Square Error (RMSE) [65], and the formula for RMSE is as follows:

$$\text{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(x_i - y_i)^2} \tag{29}$$

In addition, the margin of relative error between the average test value x_i and true value y_i is measured using Mean Absolute Percentage Error (MAPE), and the formula for its calculation is as follows:

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - x_i}{y_i}\right| \times 100\% \tag{30}$$

We calculated the RMSE and MAPE of the extracted vehicle motion data, including vehicle trajectory, speed, and yaw angle from the UAV videos in the collected and labeled MWUAV dataset.

As shown in Figure 12, due to the high accuracy of the proposed vehicle detection and tracking methods, the RMSE and MAPE of the extracted vehicles trajectories are less than 0.13 m and 0.98%, respectively, and the mean values of the RMSE and MAPE are less than 0.025 m and 0.142%, which indicates that the framework had high precision and strong robustness.

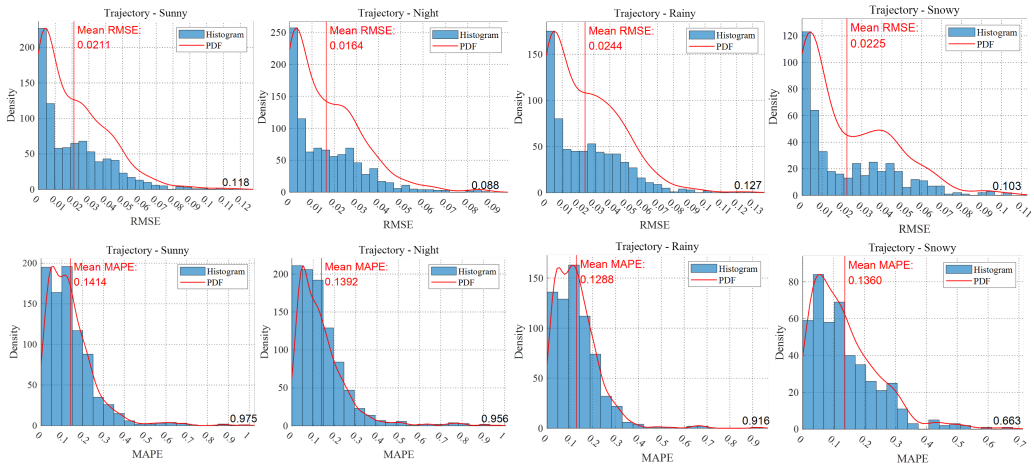

Figure 12. RMSE (first row) and MAPE (second row) histograms of extracted vehicle trajectories under four different weather conditions in the labeled MWUAV dataset.

As shown in Figure 13, we compared the extracted vehicle speed by the proposed method with the manual ground truth. The RMSE and MAPE of the extracted vehicle speed data under four different weather conditions are less than 0.12 km/h and 1.53%, respectively, and the mean values of RMSE and MAPE are less than 0.023 km/h and 0.312%, which indicates the high reliability of the proposed framework.

As shown in Figure 14, we compare the yaw angle of vehicles extracted using the proposed method with the manual ground truth. The RMSE and MAPE of the extracted vehicle yaw angle data are less than ±0.19 degrees and 1.70%, respectively, and the mean value of RMSE and MAPE are less than 0.021 degrees and 0.301%, which indicates the high stability of the framework.

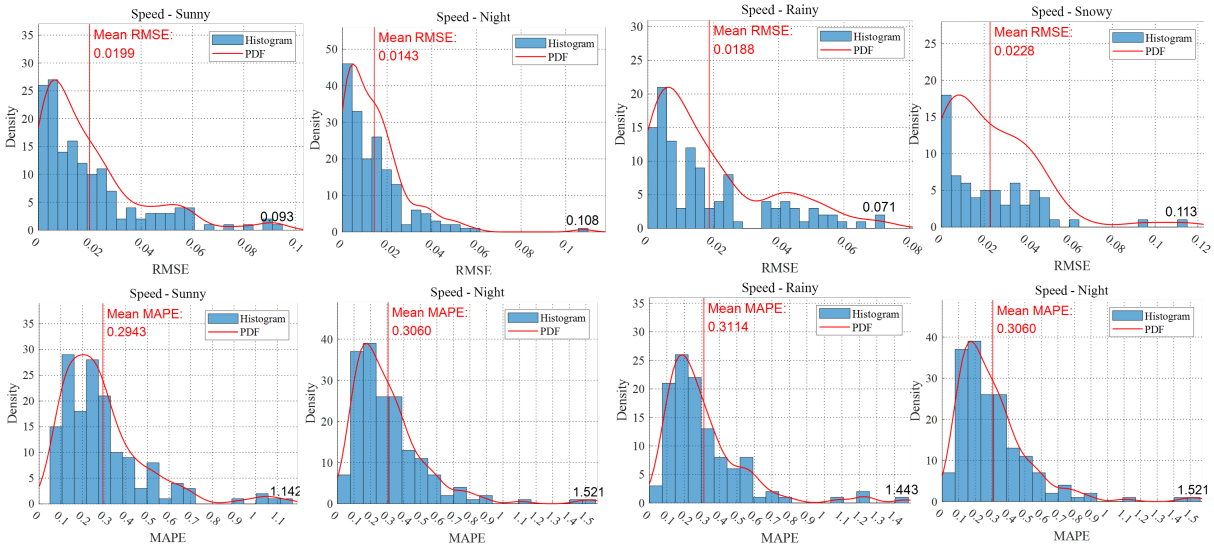

Figure 13. RMSE (first row) and MAPE (second row) histograms of extracted vehicle speed under different weather conditions in the labeled MWUAV dataset.

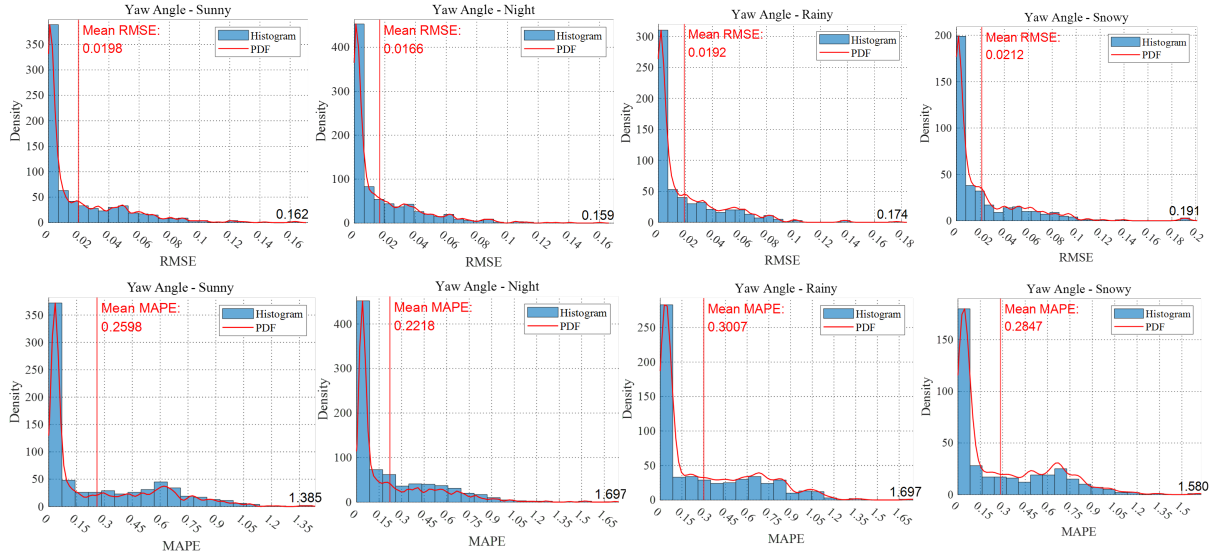

Figure 14. RMSE (first row) and MAPE (second row) histograms of extracted vehicle yaw angle under different weather conditions in the labeled MWUAV dataset.

According to the result of the aforementioned extracted vehicle motion data estimation, our proposed framework can extract high-precision vehicle motion data from UAV video captured under various weather conditions such as sunny, night, rainy, and snowy conditions.

4. Conclusions

In this study, we proposed a new framework for extracting high-precision vehicle motion data from UAV video captured under various weather conditions. For the input data, the traffic video captured by the camera-equipped UAV under different weather conditions such as rainy, sunny, snowy, and night condition was stabled. Then, the dual-weight YOLOv5-OBB object detection algorithm was used to detect vehicles from the input data. The detection results were processed using the object tracking algorithm SORT++ to obtain traffic data as the output. Finally, two new collected and manually labeled datasets were used to evaluate the accuracy of the extracted vehicle motion data. This study offers the following findings:

1. In our experiments on the two UAV datasets collected under various weather condition, the proposed vehicle detection and tracking method significantly outperforms the state-of-the-art methods on collected datasets.

2. According to the results of the data estimation, the proposed framework can extract accurate and reliable vehicle motion data such as vehicle trajectory, vehicle speed, and vehicle yaw angle from UAV videos captured under different weather conditions.

For future work, the UAV image or video could be enhanced using a more advanced image enhancement method to reduce the influence of different weather conditions. In addition, more advanced vehicle detection and vehicle tracking methods could be used for the proposed framework to further improve the accuracy of the extracted vehicle motion data.

Author Contributions: Conceptualization, X.L.; methodology, X.L.; software, X.L.; validation, X.L.; formal analysis, X.L.; investigation, X.L.; resources, X.L.; data curation, X.L.; writing—original draft preparation, X.L.; writing—review and editing, X.L., J.W.; visualization, X.L.; supervision, X.L., J.W.; project administration, X.L.; funding acquisition, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhan, W.; Sun, L.; Wang, D.; Shi, H.; Clausse, A.; Naumann, M.; Kummerle, J.; Konigshof, H.; Stiller, C.; de La Fortelle, A.; et al. Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. *arXiv* **2019**, arXiv:1910.03088.
2. Alexiadis, V.; Colyar, J.; Halkias, J.; Hranac, R.; McHale, G. The next generation simulation program. *Inst. Transp. Eng. ITE J.* **2004**, *74*, 22.
3. Robicquet, A.; Sadeghian, A.; Alahi, A.; Savarese, S. Learning social etiquette: Human trajectory understanding in crowded scenes. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2016; pp. 549–565.
4. Krajewski, R.; Bock, J.; Kloeker, L.; Eckstein, L. The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2118–2125.
5. Yang, D.; Li, L.; Redmill, K.; Özgüner, Ü. Top-view trajectories: A pedestrian dataset of vehicle-crowd interaction from controlled experiments and crowded campus. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, 9–12 June 2019; pp. 899–904.
6. Bock, J.; Krajewski, R.; Moers, T.; Runde, S.; Vater, L.; Eckstein, L. The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections. In Proceedings of the 2020 IEEE Intelligent Vehicles Symposium (IV), Las Vegas, NA, USA, 19 October–13 November 2020; pp. 1929–1934.
7. Krajewski, R.; Moers, T.; Bock, J.; Vater, L.; Eckstein, L. The round dataset: A drone dataset of road user trajectories at roundabouts in germany. In Proceedings of the 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), Rhodes, Greece, 20–23 September 2020; pp. 1–6.

8. Moers, T.; Vater, L.; Krajewski, R.; Bock, J.; Zlocki, A.; Eckstein, L. The exiD Dataset: A Real-World Trajectory Dataset of Highly Interactive Highway Scenarios in Germany. In Proceedings of the 2022 IEEE Intelligent Vehicles Symposium (IV), Aachen, Germany, 4–9 June 2022; pp. 958–964.
9. Zheng, O.; Abdel-Aty, M.; Yue, L.; Abdelraouf, A.; Wang, Z.; Mahmoud, N. CitySim: A Drone-Based Vehicle Trajectory Dataset for Safety Oriented Research and Digital Twins. *arXiv* **2022**, arXiv:2208.11036.
10. Wu, B.F.; Juang, J.H. Adaptive vehicle detector approach for complex environments. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 817–827. [CrossRef]
11. El-Khoreby, M.A.; Abu-Bakar, S.; Mokji, M.M.; Omar, S.N. Vehicle detection and counting using adaptive background model based on approximate median filter and triangulation threshold techniques. *Autom. Control. Comput. Sci.* **2020**, *54*, 346–357. [CrossRef]
12. He, S.; Chen, Z.; Wang, F.; Wang, M. Integrated image defogging network based on improved atmospheric scattering model and attention feature fusion. *Earth Sci. Inform.* **2021**, *14*, 2037–2048. [CrossRef]
13. Lin, C.T.; Huang, S.W.; Wu, Y.Y.; Lai, S.H. GAN-based day-to-night image style transfer for nighttime vehicle detection. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 951–963. [CrossRef]
14. Wang, Z.; Zhan, J.; Duan, C.; Guan, X.; Lu, P.; Yang, K. A review of vehicle detection techniques for intelligent vehicles. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**. [CrossRef]
15. Abdullah, M.N.; Ali, Y.H. Vehicles Detection System at Different Weather Conditions. *Iraqi J. Sci.* **2021**, *62*, 2040–2052. [CrossRef]
16. Huang, S.C.; Le, T.H.; Jaw, D.W. DSNet: Joint semantic learning for object detection in inclement weather conditions. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 2623–2633. [CrossRef]
17. Han, X. Modified cascade RCNN based on contextual information for vehicle detection. *Sens. Imaging* **2021**, *22*, 1–19. [CrossRef]
18. Arora, N.; Kumar, Y.; Karkra, R.; Kumar, M. Automatic vehicle detection system in different environment conditions using fast R-CNN. *Multimed. Tools Appl.* **2022**, *81*, 18715–18735. [CrossRef]
19. Cao, J.; Song, C.; Song, S.; Peng, S.; Wang, D.; Shao, Y.; Xiao, F. Front vehicle detection algorithm for smart car based on improved SSD model. *Sensors* **2020**, *20*, 4646. [CrossRef] [PubMed]
20. Hassaballah, M.; Kenk, M.A.; Muhammad, K.; Minaee, S. Vehicle detection and tracking in adverse weather using a deep learning framework. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 4230–4242. [CrossRef]
21. Humayun, M.; Ashfaq, F.; Jhanjhi, N.Z.; Alsadun, M.K. Traffic Management: Multi-Scale Vehicle Detection in Varying Weather Conditions Using YOLOv4 and Spatial Pyramid Pooling Network. *Electronics* **2022**, *11*, 2748. [CrossRef]
22. Chen, X.Z.; Chang, C.M.; Yu, C.W.; Chen, Y.L. A real-time vehicle detection system under various bad weather conditions based on a deep learning model without retraining. *Sensors* **2020**, *20*, 5731. [CrossRef]
23. Al-Haija, Q.A.; Gharaibeh, M.; Odeh, A. Detection in Adverse Weather Conditions for Autonomous Vehicles via Deep Learning. *AI* **2022**, *3*, 303–317. [CrossRef]
24. Walambe, R.; Marathe, A.; Kotecha, K.; Ghinea, G. Lightweight object detection ensemble framework for autonomous vehicles in challenging weather conditions. *Comput. Intell. Neurosci.* **2021**, *2021*, 5278820. [CrossRef]
25. Rezaei, M.; Terauchi, M.; Klette, R. Robust vehicle detection and distance estimation under challenging lighting conditions. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2723–2743. [CrossRef]
26. Baghdadi, S.; Aboutabit, N. Illumination correction in a comparative analysis of feature selection for rear-view vehicle detection. *Int. J. Mach. Learn. Comput.* **2019**, *9*, 712–720. [CrossRef]
27. Nguyen, K.; Nguyen, P.; Bui, D.C.; Tran, M.; Vo, N.D. Analysis of the Influence of De-hazing Methods on Vehicle Detection in Aerial Images. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*. [CrossRef]
28. Chen, Y.; Li, W.; Sakaridis, C.; Dai, D.; Van Gool, L. Domain adaptive faster r-cnn for object detection in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 3339–3348.
29. Sindagi, V.A.; Oza, P.; Yasarla, R.; Patel, V.M. Prior-based domain adaptive object detection for hazy and rainy conditions. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2020; pp. 763–780.
30. Li, J.; Xu, Z.; Fu, L.; Zhou, X.; Yu, H. Domain adaptation from daytime to nighttime: A situation-sensitive vehicle detection and traffic flow parameter estimation framework. *Transp. Res. Part C Emerg. Technol.* **2021**, *124*, 102946. [CrossRef]
31. Xia, G.S.; Bai, X.; Ding, J.; Zhu, Z.; Belongie, S.; Luo, J.; Datcu, M.; Pelillo, M.; Zhang, L. DOTA: A large-scale dataset for object detection in aerial images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3974–3983.
32. Wen, L.; Du, D.; Cai, Z.; Lei, Z.; Chang, M.C.; Qi, H.; Lim, J.; Yang, M.H.; Lyu, S. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *Comput. Vis. Image Underst.* **2020**, *193*, 102907. [CrossRef]
33. Du, D.; Qi, Y.; Yu, H.; Yang, Y.; Duan, K.; Li, G.; Zhang, W.; Huang, Q.; Tian, Q. The unmanned aerial vehicle benchmark: Object detection and tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 370–386.
34. Du, D.; Zhu, P.; Wen, L.; Bian, X.; Lin, H.; Hu, Q.; Peng, T.; Zheng, J.; Wang, X.; Zhang, Y.; et al. VisDrone-DET2019: The vision meets drone object detection in image challenge results. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, Seoul, Korea, 27–28 October 2019.
35. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]

36. Li, X.; Cai, Z.; Zhao, X. Oriented-YOLOv5: A Real-time Oriented Detector Based on YOLOv5. In Proceedings of the 2022 7th International Conference on Computer and Communication Systems (ICCCS), Wuhan, China, 22–25 April 2022; pp. 216–222.
37. Feng, J.; Yi, C. Lightweight Detection Network for Arbitrary-Oriented Vehicles in UAV Imagery via Global Attentive Relation and Multi-Path Fusion. *Drones* **2022**, *6*, 108. [CrossRef]
38. Reza, A.M. Realization of the contrast limited adaptive histogram equalization (CLAHE) for real-time image enhancement. *J. VLSI Signal Process. Syst. Signal Image Video Technol.* **2004**, *38*, 35–44. [CrossRef]
39. Kuran, U.; Kuran, E.C. Parameter selection for CLAHE using multi-objective cuckoo search algorithm for image contrast enhancement. *Intell. Syst. Appl.* **2021**, *12*, 200051. [CrossRef]
40. Wang, C.Y.; Liao, H.Y.M.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 390–391.
41. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12993–13000.
42. Yang, X.; Yan, J. On the arbitrary-oriented object detection: Classification based approaches revisited. *Int. J. Comput. Vis.* **2022**, *130*, 1340–1365. [CrossRef]
43. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.; Upcroft, B. Simple online and realtime tracking. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3464–3468.
44. Zhou, D.; Fang, J.; Song, X.; Guan, C.; Yin, J.; Dai, Y.; Yang, R. Iou loss for 2d/3d object detection. In Proceedings of the 2019 International Conference on 3D Vision (3DV), Quebec, QC, Canada, 15–18 September 2019; pp. 85–94.
45. Rong, W.; Li, Z.; Zhang, W.; Sun, L. An improved CANNY edge detection algorithm. In Proceedings of the 2014 IEEE International Conference on Mechatronics and Automation, Tianjin, China, 3–6 August 2014; pp. 577–582.
46. Welch, G.; Bishop, G. An Introduction to the Kalman Filter. 1995. Available online: https://www.researchgate.net/publication/200045331_An_Introduction_to_the_Kalman_Filter (accessed on 21 October 2022).
47. Hsieh, M.R.; Lin, Y.L.; Hsu, W.H. Drone-based object counting by spatially regularized regional proposal network. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 4145–4153.
48. Mundhenk, T.N.; Konjevod, G.; Sakla, W.A.; Boakye, K. A large contextual dataset for classification, detection and counting of cars with deep learning. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 785–800.
49. Kouris, A.; Kyrkou, C.; Bouganis, C.S. Informed region selection for efficient uav-based object detectors: Altitude-aware vehicle detection with cycar dataset. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 51–58.
50. Azimi, S.M.; Bahmanyar, R.; Henry, C.; Kurz, F. Eagle: Large-scale vehicle detection dataset in real-world scenarios using aerial imagery. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 6920–6927.
51. Mueller, M.; Smith, N.; Ghanem, B. A benchmark and simulator for uav tracking. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 445–461.
52. Lyu, Y.; Vosselman, G.; Xia, G.S.; Yilmaz, A.; Yang, M.Y. UAVid: A semantic segmentation dataset for UAV imagery. *ISPRS J. Photogramm. Remote Sens.* **2020**, *165*, 108–119. [CrossRef]
53. Razakarivony, S.; Jurie, F. Vehicle detection in aerial imagery: A small target detection benchmark. *J. Vis. Commun. Image Represent.* **2016**, *34*, 187–203. [CrossRef]
54. Liu, K.; Mattyus, G. Fast multiclass vehicle detection on aerial images. *IEEE Geosci. Remote. Sens. Lett.* **2015**, *12*, 1938–1942.
55. Ke, R.; Li, Z.; Kim, S.; Ash, J.; Cui, Z.; Wang, Y. Real-time bidirectional traffic flow parameter estimation from aerial videos. *IEEE Trans. Intell. Transp. Syst.* **2016**, *18*, 890–901. [CrossRef]
56. Dendorfer, P.; Osep, A.; Milan, A.; Schindler, K.; Cremers, D.; Reid, I.; Roth, S.; Leal-Taixé, L. Motchallenge: A benchmark for single-camera multiple target tracking. *Int. J. Comput. Vis.* **2021**, *129*, 845–881. [CrossRef]
57. Sun, Y.; Cao, B.; Zhu, P.; Hu, Q. Drone-based RGB-Infrared Cross-Modality Vehicle Detection via Uncertainty-Aware Learning. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 6700–6713. [CrossRef]
58. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
59. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
60. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
61. Ding, J.; Xue, N.; Long, Y.; Xia, G.S.; Lu, Q. Learning RoI transformer for oriented object detection in aerial images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 2849–2858.
62. Wojke, N.; Bewley, A.; Paulus, D. Simple online and realtime tracking with a deep association metric. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 3645–3649.

63. Zhang, Y.; Sun, P.; Jiang, Y.; Yu, D.; Yuan, Z.; Luo, P.; Liu, W.; Wang, X. Bytetrack: Multi-object tracking by associating every detection box. *arXiv* **2021**, arXiv:2110.06864.
64. Cao, J.; Weng, X.; Khirodkar, R.; Pang, J.; Kitani, K. Observation-Centric SORT: Rethinking SORT for Robust Multi-Object Tracking. *arXiv* **2022**, arXiv:2203.14360.
65. Chai, T.; Draxler, R.R. Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature. *Geosci. Model Dev.* **2014**, *7*, 1247–1250. [CrossRef]