remote sensing

MDPI

Article

An Adaptive Framework for Multi-Vehicle Ground Speed Estimation in Airborne Videos

Jing Li [1,*], Shuo Chen [1], Fangbing Zhang [2,3], Erkang Li [1], Tao Yang [2,3,*] and Zhaoyang Lu [1]

1 School of Telecommunications Engineering, Xidian University, Xi'an 710000, China; shuochen@stu.xidian.edu.cn (S.C.); ekli@stu.xidian.edu.cn (E.L.); zhylu@xidian.edu.cn (Z.L.)

2 School of Computer Science, Northwestern Polytechnical University, Xi'an 710000, China; fangbing_zhang@mail.nwpu.edu.cn

3 National Engineering Laboratory for Integrated Aero-Space-Ground-Ocean Big Data Application Technology, SAIIP, Xi'an 710000, China

* Correspondence: jinglixd@mail.xidian.edu.cn (J.L.); tyang@nwpu.edu.cn (T.Y.); Tel.: +86-139-9132-0168 (J.L.); +86-150-0291-9079 (T.Y.)

Received: 27 March 2019; Accepted: 17 May 2019; Published: 24 May 2019

Abstract: With the rapid development of unmanned aerial vehicles (UAVs), UAV-based intelligent airborne surveillance systems represented by real-time ground vehicle speed estimation have attracted wide attention from researchers. However, there are still many challenges in extracting speed information from UAV videos, including the dynamic moving background, small target size, complicated environment, and diverse scenes. In this paper, we propose a novel adaptive framework for multi-vehicle ground speed estimation in airborne videos. Firstly, we build a traffic dataset based on UAV. Then, we use the deep learning detection algorithm to detect the vehicle in the UAV field of view and obtain the trajectory in the image through the tracking-by-detection algorithm. Thereafter, we present a motion compensation method based on homography. This method obtains matching feature points by an optical flow method and eliminates the influence of the detected target to accurately calculate the homography matrix to determine the real motion trajectory in the current frame. Finally, vehicle speed is estimated based on the mapping relationship between the pixel distance and the actual distance. The method regards the actual size of the car as prior information and adaptively recovers the pixel scale by estimating the vehicle size in the image; it then calculates the vehicle speed. In order to evaluate the performance of the proposed system, we carry out a large number of experiments on the AirSim Simulation platform as well as real UAV aerial surveillance experiments. Through quantitative and qualitative analysis of the simulation results and real experiments, we verify that the proposed system has a unique ability to detect, track, and estimate the speed of ground vehicles simultaneously even with a single downward-looking camera. Additionally, the system can obtain effective and accurate speed estimation results, even in various complex scenes.

Keywords: ground vehicle speed estimation; intelligent airborne video surveillance; unmanned aerial vehicle; object detection and tracking; motion compensation

1. Introduction

With the popularization of cars, the number of vehicles, as well as the traffic pressure, have rapidly increased. Therefore, improving traffic conditions has become an important issue for traffic management in various countries, and the intelligent transportation system (ITS) has been rapidly developed [1]. Vehicle ground speed estimation is one fundamental and important part of the intelligent transportation system, which is critical to improving the efficiency and safety of roadway transportation [2]. Moreover, it has many applications in many fields, such as the autonomous landing of UAVs [3], social anti-terrorism, and public security protection.

The existing vehicle speed estimation system includes buried induction loop technology [4], radar technology [5], ultrasonic technology [6,7], laser technology [8], video-based technology [2], and so on [9–11]. Among these, geomagnetic coil technology is widely used, but these technologies require extensive installation and maintenance of equipment and the use of professional knowledge for configuration and calibration. Microwave radar and laser meters avoid these problems, but the equipment they need is usually more expensive and requires frequent maintenance. Ultrasonic technology uses reflection characteristics for speed measurement at a lower cost. However, the measurement reaction time is long, and the effective distance is small. In contrast, video-based technology does not require complex and expensive hardware, is easy to install and maintain, and has a large monitoring range. It is a low-cost, information-rich method. Moreover, owing to the development and maturity of video processing technology, this approach has grown rapidly.

During recent decades, various vehicle ground speed estimation methods based on vision have been proposed. At present, there are two main methods used at home and abroad [12], as follows: (1) The speed estimation method based on the virtual loop [13,14]: The method is similar to the induction coil speed estimation method, the advantage of this method is the shortened processing time. However, this method obtains the average speed of the vehicle within the virtual loop. Furthermore, it cannot measure a wide range of vehicle speeds, therefore wasting image resources. In addition, the method can only detect the speed of a vehicle in one lane at a time and cannot detect the speed when two or more vehicles pass at the same time. (2) The method based on vehicle tracking [15,16]: As the vehicle is moving, its position in the image captured by the camera will change continuously. In this method, a series of positions of the vehicle in the video sequence can be obtained by computer vision tracking technology. Then, the time intervals between corresponding frames are used to calculate the vehicle's speed. This method can obtain the speeds of all vehicles in the field of view with high measurement accuracy, and the appearance and current location of a vehicle can be obtained. With the development of computer vision technology in recent years, this method is becoming more and more popular in intelligent monitoring systems.

Most video-based speed estimation systems are based on fixed cameras [17]. These cameras are usually placed in specific locations such as intersections to monitor coverage. For example, in [18], researchers presented a vehicle speed estimation algorithm for video surveillance. The method acquires the positional change of the vehicle on the image by the moving target detection and tracking technique, and then restores the vehicle position change in the real world by using the calibration of the fixed camera to realize the speed measurement. A very interesting project is mentioned in [19], which is a novel two-camera-based vehicle speed detection system. It estimates the speed of a vehicle using pairs of distance measurements obtained from two cameras and demonstrates that there is a specific geometry between the cameras to minimize the speed error. This fixed camera-based surveillance system has a limited monitoring range and may not be cost effective, because a large number of these devices are required for a single road segment.

With the rapid development of UAVs in recent years [20], a method of traffic data acquisition using UAVs equipped with cameras has gradually emerged. UAV is a cost-effective and flexible platform, which can realize large-scale monitoring at high altitudes and accurate small-scale monitoring at low altitudes by freely adjusting different flight altitudes. Moreover, it can be used to monitor continuous areas by moving and can also hover over specific road sections. In addition, the use and maintenance of UAVs are convenient, and it can provide rapid assessment and reconnaissance of the accident site as an emergency response. For the aforementioned reasons, UAV-based intelligent monitoring systems have attracted more and more attention [21], and the vehicle speed estimation technology in airborne videos has gradually developed. For instance, Yamazaki et al. [22] suggested deriving the velocity of cars by exploiting their shadows. This work explored the use of a set of invariant features (i.e., SIFT features) to locate the vehicles and to estimate their speeds in two successive aerial images. Similarly, Moranduzzo et al. [23] developed an approach for vehicle speed estimation by finding correspondence between scale-invariant feature transform (SIFT) features across video frames.

Ke et al. [24] presented a method to estimate the average speed of traffic streams from aerial UAV footage. The method extracts and tracks interest points with the Kanade–Lucas algorithm and the motion vectors of optical flow are used to cluster the interest points in the velocity space to determine the average speed of traffic flow. Bruin et al. [25] proposed an autonomous drone-based traffic flow estimation system. The system first creates and saves road profiles that can store all location-specific information. Then, the displacement (distance) of the vehicles between consecutive frames is obtained to determine the velocity of passing vehicles. Guido et al. [26] presented a methodology used to extract data from a traffic stream though UAV application. This method first obtains a set of location GPS points using a desktop GIS. These points can improve the quality of the georeferencing process of the frames acquired by UAV and allow the mapping of video pixels to resolve the association of every pixel to real-world coordinates. Then, a series of vehicle trajectories are obtained by the target tracking algorithm, and the vehicle's velocity is calculated from the space–time diagrams of the target vehicle.

However, there are still many difficulties in vehicle speed estimation in airborne videos [27], as shown in Figure 1. First, there are vibration and blur in the images acquired by UAVs, and the airborne video environment is complex, as shown in Figure 1d. There are vegetation, road facilities, buildings, roadside landscape trees, water bodies, non-motor vehicles, and pedestrians, which will increase the difficulty of subsequent image processing. Second, vehicle speed estimation in airborne videos is difficult due to the dynamic background in the image caused by UAV motion [28]. Moreover, there can be many vehicles in a scene and their sizes are small, as shown in Figure 1c, so it is very difficult to locate the target accurately [29]. Third, when calculating the vehicle speed, it is necessary to recover the pixel scale to obtain the true displacement. However, the mapping relationship between the pixels and the actual distance varies with the change of flight altitude. Many methods recover the pixel scale by setting reference marks or ground control points [24,25,30], or by determining the flying height and camera internal parameters [23]. This process is complicated and consumes a lot of time and effort, which makes the system inconvenient to use. Finally, vehicle speed estimation system from airborne videos requires fast algorithm processing, which can be achieved with real-time processing.

To address the above problems, this paper presents a novel adaptive framework for multi-vehicle ground speed estimation in airborne videos. This framework takes the actual size of a car as the prior information and has a unique ability to detect, track, and estimate the speed of ground vehicle simultaneously. Moreover, the framework can obtain effective and robust vehicle speed estimation results in various complex environments without using auxiliary equipment such as GPS. Vehicle speed estimation results for different scenarios are shown in Figure 1. The main contributions of this work are as follows:

- First, we built a traffic training dataset for vehicle detection in airborne videos, including aerial images which were captured by the UAV, and images of the public traffic dataset (UA-DETRAC). We used this training dataset to generate a neural network model and used the deep learning method to improve the vehicle detection rate in aerial video. Moreover, we established a test dataset for system performance evaluation, which consisted of five aerial scenarios, each with more than 5000 images. Then, we obtained a series of positions for each vehicle in the image through the tracking-by-detection algorithm.
- Second, we formed an adaptive framework for multi-vehicle ground speed estimation in airborne videos. In this framework, based on the detection and tracking results, we first eliminated the influence of the detected target and utilized the motion compensation method based on homography to obtain the vehicle trajectories in the current frame and the real pixel displacements. Then, we designed a mapping relationship between the pixel distance and the actual distance to estimate the real displacements and motion speeds of the vehicles. This used the actual size of the car as the prior information and realized the adaptive recovery of the pixel scale at the current time by estimating the vehicle size in the image.

- Finally, we built an adaptive vehicle speed estimation system for airborne video in simulations and in a real environment. The structure of the real system is very simple. It consists of a UAV DJI-MATRICE 100, a Point Grey monocular camera, and a computer. To prove the effectiveness and robustness of the system, we first conducted a large number of simulation experiments on the AirSim platform. The quantitative analysis results were used to verify that the system has a high-speed measurement accuracy. Then, we carried out experiments in real environments, showing that the proposed system has a unique ability to detect, track, and estimate the speed of ground vehicles simultaneously. Moreover, the quantitative experimental results and analysis proved that the system could quickly obtain effective and robust speed estimation results without using auxiliary equipment such as GPS in various complex environments.

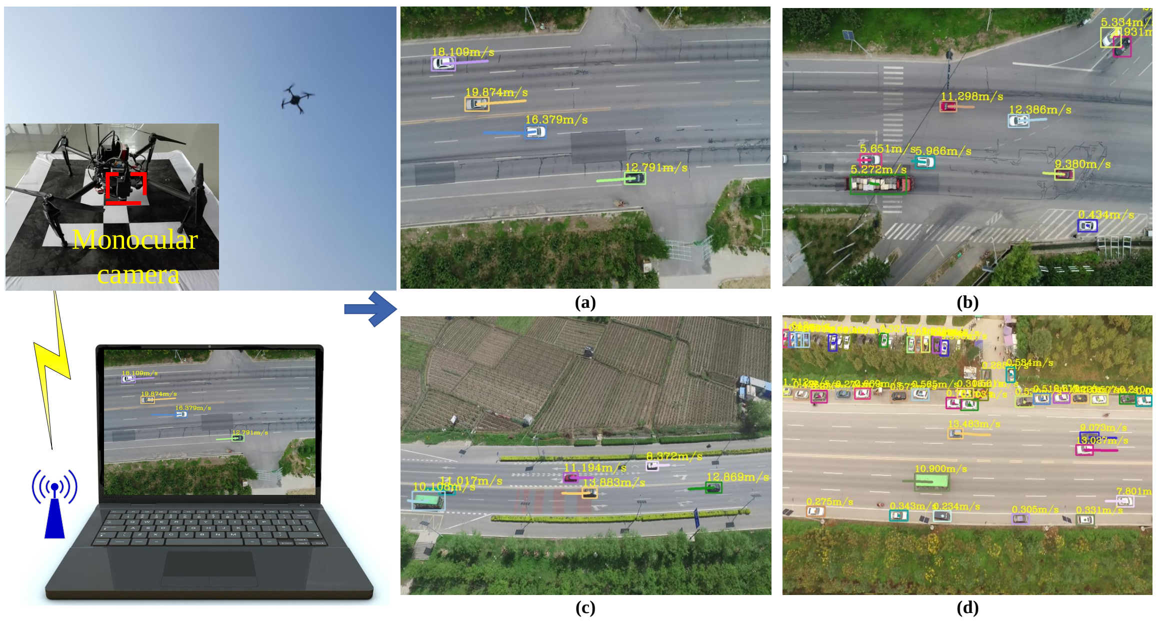


Figure 1. Vehicle speed estimation results of our system in different scenarios. Among them, scene (**a**) is a highway, scene (**b**) is an intersection, scene (**c**) is a road, and scene (**d**) is a parking lot entrance. The backgrounds of these scenes are complex, the number of vehicles is large, and the size is small, which makes vehicle speed estimation difficult.

2. The Proposed Method

The framework of the proposed adaptive vehicle speed estimation system for airborne videos is illustrated in Figure 2. It can be segmented into three separate parts, which are described in the sub-sections below. The first part is vehicle detection and tracking, which includes the deep leaning detection algorithm YOLOv3 [31] and the tracking-by-detection method. In this part, a series of positions of vehicles in the image can be obtained. The second part is the motion compensation of the trajectory. In this part, the dense optical flow method accelerated by GPU is first utilized to obtain effective matching points. On this basis, we can calculate the homography matrix more accurately, and the real trajectory is estimated by the homography in the current frame. In the third part, a mapping relationship is designed between the pixel distance and the actual distance to estimate the real displacements and motion speed of the vehicles. Specifically, the actual size of the car is taken as the prior information, and the diagonal lengths of cars in the image are used to design a mapping relationship, which determines the mapping ratio by estimating the diagonal pixel length of each car in the current frame. Meanwhile, a Gaussian model is established by using the diagonal lengths of cars in the image and is constantly updated over time, and the mean of the model is the cars' average diagonal length. Then, the real moving distance is recovered, and the vehicle speed is calculated.

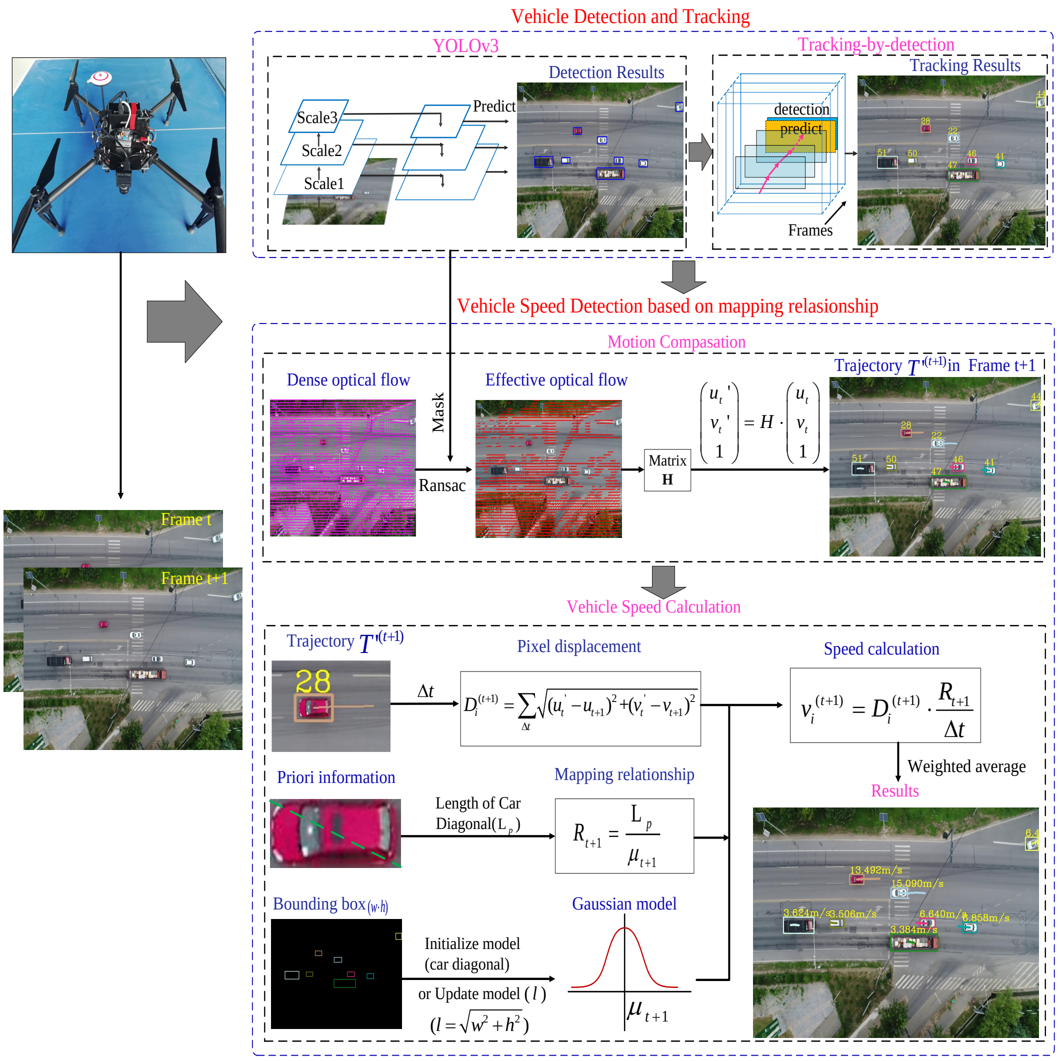


Figure 2. An overview of the proposed multi-vehicle ground speed estimation system in airborne videos. The hardware structure of the system is very simple. It is composed of a UAV DJI-MATRICE 100, a Point Grey monocular camera, and a computer, and it transmits data using wireless technology. The vehicle speed estimation algorithm mainly consists of three parts: multi-vehicle detection and tracking, motion compensation, and adaptive vehicle speed calculation.

2.1. Vehicle Detection And Tracking

In order to obtain the displacement of the vehicle over a period of time, we first used the detection algorithm and tracking algorithm to obtain a series of positions of the target in the image, as shown in the first part of Figure 2. In this section, we focus on the deep learning detection algorithm and the tracking-by-detection algorithm, the vehicle detection, and tracking are shown in Figure 3.

Because of the high position of UAV, the airborne video has a wide field of view, and there is no target occlusion problem. However, the target in the image is small, the background in the scene is complex, and the motion blurring will be caused by the vibration of the UAV, which brings new problems for target detection. Currently, there are many popular target detection algorithms, such as background modeling methods like the visual background extractor (ViBe) [32], stereo vision methods [33], and deep learning methods like YOLOv3 [31]. In the system, we used YOLOv3 [31] to obtain an accurate detection bounding box of the vehicle in the image. It is noted that the network model provided by the YOLOv3 official website can detect the vehicle, but it is not good for the detection of small-sized vehicles captured by multiple angles in the airborne video. Therefore, we built a traffic training dataset $Train\_TrafficData$ to train the YOLOv3 network model. This dataset contained 13,900 images, including our own traffic dataset based on UAV and a public dataset (UA-DETRAC). Moreover, we used the $LabelImg$ tool [34] to label vehicles in these images to train the network model.

YOLOv3 [31] is a good detector, which outperforms some state-of-the-art methods like faster regions with convolutional neural networks (RCNN) with ResNet [35] and single shot multibox detector (SSD) [36], while still running significantly faster. This method solves object detection as a regression problem. The output from the input of the original image to the position and category of the object is done based on a single end-to-end network. More specifically, the first step is to calculate the feature map of the image. As can be seen from the first part of Figure 2, the system extracts features from three different scales using a similar concept to feature pyramid networks, which helps to detect small objects better. The second step is to predict the suggestion box. Based on the feature map, the system predicts bounding boxes using dimension clusters as anchor boxes. The network predicts four coordinates and the level of confidence for each bounding box, t_x, t_y, t_w, t_h. If the cell is offset from the top left corner of the image by (c_x, c_y) and the bounding box prior has a width and height of p_w and p_h, then the predicted bounding box corresponds to

$$\begin{aligned} b_x = \sigma(t_x) + c_x \quad b_y = \sigma(t_y) + c_y \\ b_w = p_w e^{t_w} \quad b_h = p_h e^{t_h} \end{aligned} \tag{1}$$

where (b_x, b_y) represents the coordinates of the upper left corner of the predicted bounding box, and b_w and b_h represent the width and height of the bounding box. In this way, a series of bounding boxes are obtained. In addition, YOLOv3 predicts an objectness score for each bounding box using logistic regression and only assigns one bounding box prior for each ground truth object. In the third step, each box predicts the classes by using independent logistic classifiers. Then, non-maximus suppression (NMS) [37] is used to select the final detection $\{x, y, w, h\}$ from multiple overlapping candidate proposals, x, y represents the coordinates of the upper left corner of the detection bounding box, and w and h represent the width and height of the bounding box. By using the large-scale airborne video data training detection model, this target detection algorithm has good detection performance for small target vehicles from airborne videos in many complex environments. From the detection results in Figure 3, we can see that even if the background of the surveillance scene is complex, the number of targets is large and their sizes are small, we can still get great detection results.

After that, the system uses the tracking-by-detection method [38–41] to track the target and get a series of vehicles positions in the image sequence. In order to associate the targets with the trajectories correctly, the position information of the detection bounding box is used and the similarity is measured with the Kalman filter [42] and IOU calculation to achieve target tracking; the tracking process is shown in Figure 3. This method assumes that the state of the detection target T_i is $S_i = \{x_i, y_i, w_i, h_i\}$, which is the position of its bounding box. The bounding box predicted by the Kalman filtering method is used as the state of the trajectory T'_j, and the state is represented as $S'_j = \{x'_j, y'_j, w'_j, h'_j\}$. We assume that there are M detection targets and N trajectories in Frame t. Then, the state set of detection targets is $S^{(t)} = \{S_1^{(t)}, ..., S_i^{(t)}, ..., S_M^{(t)}\}$, and the state set of trajectories is $S'^{(t)} = \{S_1'^{(t)}, S_2'^{(t)}, ..., S_j'^{(t)}, ..., S_N'^{(t)}\}$. Then, the IOU values of the detection boxes are calculated in $S^{(t)}$ and $S'^{(t)}$ to generate the data association matrix in Frame t. The formula for calculating the IOU value $IOU_{i,j}^{(t)}$ of the target $T_i^{(t)}$ and trajectory $T_j'^{(t)}$ is

$$IOU_{i,j}^{(t)} = \frac{(area(S_i^{(t)} \cap S_j^{\prime(t)}))}{(area(S_i^{(t)} \cup S_j^{\prime(t)}))}. \tag{2}$$

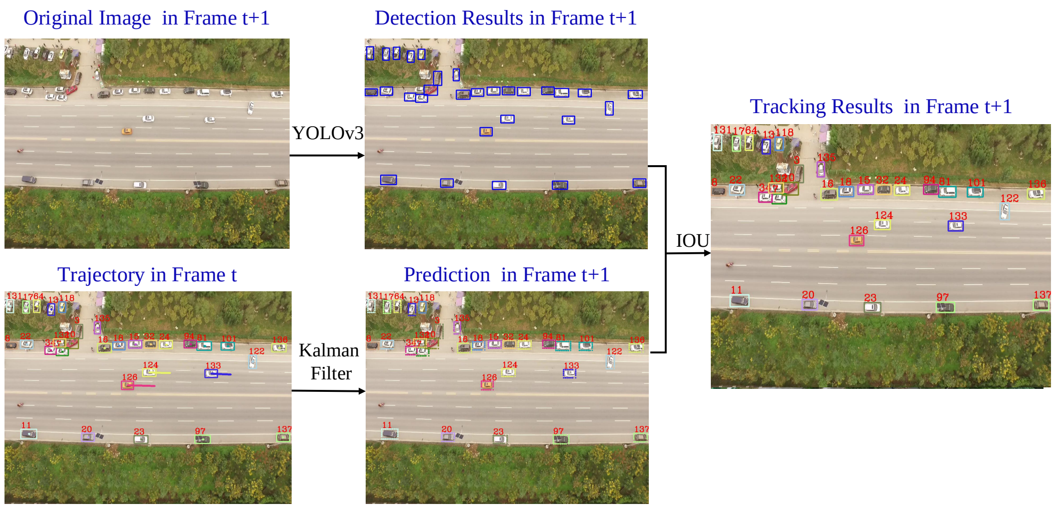

Original Image in Frame t+1
Detection Results in Frame t+1
YOLOv3
Tracking Results in Frame t+1
IOU
Trajectory in Frame t
Prediction in Frame t+1
Kalman Filter

Figure 3. Vehicle detection and tracking in a complex scene. In this scenario, there are lots of small-sized vehicles and complicated backgrounds. The system can achieve good detection results through using the YOLOv3 algorithm, and effective and robust target tracking can be realized by using the Kalman filter and intersection over union (IOU) calculation.

In the formula, $area(S_i^{(t)} \cap S_j^{\prime(t)})$ represents the area of the intersection of bounding boxes $S_i^{(t)}$ and $S_j^{\prime(t)}$, and $area(S_i^{(t)} \cup S_j^{\prime(t)})$ represents the area of the union of bounding boxes $S_i^{(t)}$ and $S_j^{\prime(t)}$. Then, $NIOU(i,j) = 1 - IOU_{i,j}^{(t)}$, and the data association matrix is given by

$$D^{(t)} = \begin{bmatrix} NIOU(1,1) & NIOU(1,2) & \cdots & NIOU(1,N) \\ NIOU(2,1) & NIOU(2,2) & \cdots & NIOU(2,N) \\ \vdots & \vdots & \ddots & \vdots \\ NIOU(M,1) & NIOU(M,2) & \cdots & NIOU(M,N). \end{bmatrix}, \tag{3}$$

Thereafter, the Hungarian algorithm is adopted to solve the assignment problem between detected targets and trajectories. When target $T_i^{(t)}$ and trajectory $T_j^{\prime(t)}$ match each other, we need to judge whether the value of $IOU_{i,j}^{(t)}$ is greater than the threshold TH. If $IOU_{i,j}^{(t)} > TH$, we think that target $T_i^{(t)}$ is associated with the trajectory $T_j^{\prime(t)}$ and update the trajectory state. Otherwise, they are not related and a new target appears. In this system, $TH = 0.5$. This tracking algorithm only uses the position information of the detection bounding box to achieve fast target tracking and obtain effective and robust tracking results.

2.2. Motion Compensation of the Vehicle Trajectory

After using the above vehicle detection and tracking algorithm, a series of vehicle positions in the image can be obtained. In a UAV surveillance system, the onboard camera moves with the UAV. Therefore, both the background and the targets in the video are moving, and the pixel displacement cannot be estimated directly by the position change of the target in the image. In this part, we introduce a motion compensation method based on homography to adjust the position of the vehicle, so as to eliminate the background motion caused by camera movement and ensure the consistency of the target's spatial coordinates in successive frames. Motion compensation is shown in the second part of Figure 2.

It is assumed that I_t and I_{t+1} are the images of Frame t and Frame $t+1$ in the airborne video. In the system, the background motion is estimated by calculating the homography matrix between I_t and I_{t+1}, which requires the preprocessing step of background feature point matching. From the image sequence in Figure 4, it can be seen that there is less road texture in the background of the image. Therefore, the commonly used feature extraction and matching algorithms may extract very few feature points on the highway, resulting in inaccurate motion estimation of the background. In our system, the GPU-accelerated Gunnar Farneback algorithm [43] is adopted to extract image matching points. The Gunnar Farneback algorithm approximates each pixel in two frames by polynomial expansion transform and then deduces the displacement field from the polynomial expansion coefficient by observing how a polynomial is accurately transformed under the translation, thus forming a dense optical flow field. This process does not depend entirely on the texture characteristics of the image. Through this dense optical flow field, the matching points of all pixels in the image can be obtained, and pixel-level image registration can be performed, even on textureless backgrounds. It should be noted that not all matching point pairs are used, but some matching points on the rows and columns of the image are evenly sampled to calculate the homography matrix.

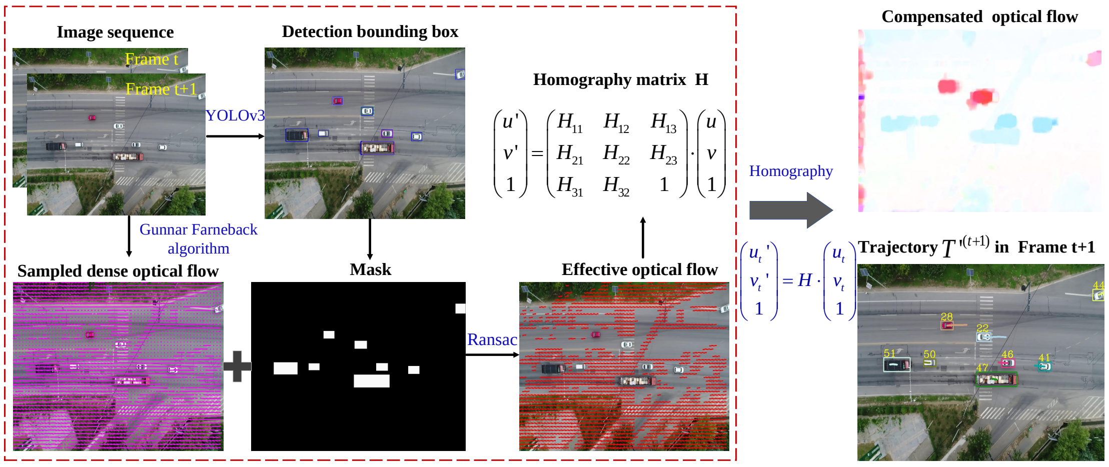


Image sequence
Frame t
Frame t+1
YOLOv3
Detection bounding box
Homography matrix H
$\begin{pmatrix}u'\\v'\\1\end{pmatrix}=\begin{pmatrix}H_{11}&H_{12}&H_{13}\\H_{21}&H_{22}&H_{23}\\H_{31}&H_{32}&1\end{pmatrix}\cdot\begin{pmatrix}u\\v\\1\end{pmatrix}$
Homography
Compensated optical flow
Gunnar Farneback algorithm
Sampled dense optical flow
Mask
Ransac
Effective optical flow
$\begin{pmatrix}u_t'\\v_t'\\1\end{pmatrix}=H\cdot\begin{pmatrix}u_t\\v_t\\1\end{pmatrix}$
Trajectory $T'^{(t+1)}$ in Frame t+1

Figure 4. Motion compensation of the vehicle trajectory. The method firstly utilizes the dense optical flow method to obtain the image matching points and eliminates the foreground matching points by means of the detection result to more accurately calculate the homography matrix **H**. Then, the trajectories of the previous frame are subjected to a homography transformation to get trajectories $T_j'^{(t+1)}$ in the current frame.

Furthermore, in order to estimate background motion more accurately, the vehicle detection result is used as the mask to eliminate the influence of the target pixel. Then, the effective optical flow is obtained with the RANSAC method. The MASK and effective optical flow are shown in Figure 4. It is assumed that there are m pairs of matching points between I_t and I_{t+1}, and $m >> 4$. Then, the elements of the homography matrix **H** of the two images I_t and I_{t+1} used for the motion compensation are derived from the following formula:

$$\begin{pmatrix}u'\\v'\\1\end{pmatrix}=\mathbf{H}\cdot\begin{pmatrix}u\\v\\1\end{pmatrix}=\begin{pmatrix}H_{11}&H_{12}&H_{13}\\H_{21}&H_{22}&H_{23}\\H_{31}&H_{32}&1\end{pmatrix}\cdot\begin{pmatrix}u\\v\\1\end{pmatrix} \tag{4}$$

Pixel (u,v) and Pixel (u',v') are an arbitrary pair of background matching points for images I_t and I_{t+1}, where H_{11}, H_{21}, H_{12}, and H_{22} represent the scaling and rotation factors, H_{13} and H_{23}

represent the translation factors, and H_{31} and H_{32} are the transformation factors. There are m pairs of matching points. In order to solve **H**, the overdetermined equation is formed and the results are obtained by the least-squares method. To verify the accuracy of this matrix **H**, **H** is used to perform motion compensation on the previous frame and recalculate the optical flow. The compensated optical flow is shown in Figure 4. We can see that after the motion compensation, the background motion is eliminated, and there is only the motion of the foreground target. On this basis, we use **H** to map the trajectory of the vehicle in Frame t to Frame $t+1$. For motion compensation of the vehicle trajectory, we only focus on the center of the detected target. We define $p_t = (u_t, v_t)$ as the center of the detected target $T_i^{(t)}$ in the image I_t, and $p'_t = (u'_t, v'_t)$ represents the mapping of pixel p_t to the image I_{t+1}. The mapping equation is similar to formula (4).

In this way, we can map the trajectories of all targets in the previous frame t to the current frame $t+1$ and obtain their real trajectories. Figure 4 shows the trajectories in the Frame $t+1$. We assume $p'_{t+1} = (u_{t+1}, v_{t+1})$ is the center of the detected target $T_i^{(t+1)}$ in the image I_{t+1}, and the length of the trajectory of the target T_i in Frame $t+1$ is N. Then, this trajectory is defined as $T_i'^{(t+1)} = \{p'_{t-N+2}, ..., p'_t, p'_{t+1}\}$.

2.3. Adaptive Vehicle Speed Estimation

The last step of the system is the speed estimation of each vehicle in the current frame $t+1$, as shown in the third part of Figure 2. In this section, we showcase an adaptive speed estimation algorithm based on the mapping relationship between the pixel distance and the actual distance. According to the real trajectory of the target in the current frame, this method can calculate the pixel displacement of the vehicle in a period of time and then estimate the real displacement and the instantaneous speed by determining the mapping relationship in Frame $t+1$. The flow of this algorithm is shown in Algorithm 1.

Calculation of vehicle pixel displacement: When calculating the vehicle speed of the current frame $t+1$, we do not only count the pixel displacement of the target in the adjacent frame, but estimate the speed by calculating the total displacement of the previous Q frame. This means that the time difference of the pixel displacement is $\Delta t = Q \cdot Fps$, and Fps indicates the time interval between two adjacent frames. In the system, Q is set to 16. We know that the trajectory of the target T_i in Frame $t+1$ is $T_i'^{(t+1)} = \{p'_{t-N+2}, ..., p'_t, p'_{t+1}\}$. Then, the Euclidean distance is used to determine the pixel displacement $D_i^{(t+1)}$ of the target T_i from Frame $t-Q+2$ to Frame $t+1$. If the length of the trajectory is less than Q, the target is considered to have just appeared, the pixel displacement is not calculated, and the velocity is initialized to 0. The calculation formula is given by

$$Ed(p'_{t+1} - p'_t) = \sqrt{(u'_t - u_{t+1})^2 + (v'_t - v_{t+1})^2} \tag{5}$$

$$D_i^{(t+1)} = \begin{cases} \sum_{k=t-Q+2}^{t} Ed(p'_{k+1} - p'_k), & if \quad N >= Q \\ 0, \quad if \quad N < M \end{cases} \tag{6}$$

$Ed(p'_{k+1} - p'_k)$ represents the Euclidean distance between pixel p'_{k+1} and pixel p'_k in the image I_{t+1}.

Mapping relationship: From the above process, we can get the pixel displacement $D_i^{(t+1)}$ of the target T_i in Δt. In order to get the real moving distance $D_i'^{(t+1)}$ of the target, the mapping relationship between the pixel and the actual distance needs to be determined. It is assumed that the mapping relationship is 1 pixel representing R meters. The mapping ratio is R, which is the pixel scale. In this system, considering that the camera is fixed under the UAV with the lens facing down and the camera field of view angle is not large (about 80°), so the observation field of view of this system is limited, and the pixel scale difference between the image center and borders is generally small. Therefore, in order to facilitate calculation, we assumed that the pixel scale R of all the pixels on the image was the same. We tested the actual pixel scales at the center of the image and close to the borders at different altitudes. The flight altitude of the UAV is generally 50–80 m. The test results showed that with an increase in flight altitude (50 m $\rightarrow$ 80 m), the pixel scale difference at the center of the image and close to the borders increased gradually (0.000 cm/pixel $\rightarrow$ 0.171 cm/pixel), but the difference was generally less than 0.2 cm/pixel, and far less than the actual pixel scale. Consequently, it is acceptable to assume that the pixel scale of all the pixels on the image is the same when estimating the vehicle speed in the proposed system.

Algorithm 1: Vehicle speed calculation method

```
Input: Target detection results $T_i^{(t+1)} = \{xi, y_i, w_i, h_i\}, i = 1...M$ and the trajectory
    $T_j'^{(t+1)}, j = 1, ...N$ in the image $I_{t+1}$
Output: Vehicle speed $V_i^{(t+1)}, i = 1...M$ within the monitored range
1  $i = 1$;
2  if $I_{t+1}$ is the first frame then
3      while $i <= M$ do
4          if $\beta_1 \cdot h_i > w_i > \beta_2 \cdot h_i || \beta_1 \cdot w_i > h_i > \beta_2 \cdot w_i$ then
5              Calculate the car's pixel diagonal length $l_i$;
6          end
7          $i = i + 1$;
8      end
9      Calculate car's average pixel diagonal length $l^{(0)}$;
10     Initialize the $\mu_0$ and $\sigma_0^2$ of the Gaussian model;
11     vehicle speed $V_i^{(t+1)} = 0, i = 1, ..., M$
12 else
13     while $i <= M$ do
14         Calculate the vehicle pixel displacement $D_i^{(t+1)}$ for a period of time $\delta t$;
15         Count the car's pixel diagonal length $l_i$, and update the $\mu_{t+1}$ and $\sigma_{t+1}^2$ of the Gaussian
             model;
16         Determine the mapping ratio $R_{t+1} = \frac{L_p}{\mu_{t+1}}$ ;
17         Calculate the vehicle speed $v_i^{(t+1)}$ according to the formula (14);
18         Weighted average of vehicle speed $V_i^{(t+1)}$;
19         $i = i + 1$;
20     end
21 end
```

Because the height of the UAV changes when monitoring traffic scenes, the mapping ratio R also changes during the monitoring process, and as the height of the UAV increases, the mapping ratio R increases. From the original image in Figure 2, we can know that most of the vehicles in the surveillance scene are domestic cars, such as the red car in Figure 2. Therefore, we used the diagonal length L_p of the car as a priori information to adaptively estimate the mapping ratio R_{t+1} of the current frame by determining the pixel length μ_{t+1} of the car's diagonal length in the image. The R_{t+1} is given by

$$R_{t+1} = \frac{L_p}{\mu_{t+1}}. \tag{7}$$

In this formula, the selection of a priori information L_p is very important. In order to determine the prior information, we checked the size of the car on the Internet; it was generally 4.5 m in length, 1.7 m in width, and about 4.8 m in the diagonal distance. In addition, we measured some domestic cars in real life, and the average result was about 4.8 m in the diagonal distance. Therefore, we set L_p to 4.8 m in our real system.

Then, in order to measure the car's diagonal pixel length μ, we built a single Gauss model by counting the diagonal length of the car's bounding boxes and updated the model over time, as shown in the third part in Figure 2. The mean value of the Gaussian model in the current frame $t+1$ is the diagonal pixel length μ_{t+1} of the car in the image I_{t+1}. Specifically, we first initialized the single Gauss model in the first frame. It should be noted that among all the bounding boxes in the first frame, the height h and width w of the car's box should satisfy $\beta_1 \cdot h > w > \beta_2 \cdot h$ or $\beta_1 \cdot w > h > \beta_2 \cdot w$, and $\beta_1 = 2.7, \beta_2 = 1.2$, so as to ignore the impact of some non-car targets, such as large trucks. The model assumes that there are n cars, which are $T_1, ..., T_i, ..., T_n$. The bounding box of T_i is $\{x_i, y_i, w_i, h_i\}$, and the diagonal pixel length l_i can be calculated as follows:

$$l_i = \sqrt{w_i \cdot w_i + h_i \cdot h_i}. \tag{8}$$

Then, the average diagonal pixel length $l^{(0)}$ in the first frame is given by

$$l^{(0)} = \frac{\sum_{i=1}^{n} l_i}{n}. \tag{9}$$

We set the initial mean μ_0 of the Gauss model to $l^{(0)}$ and set the initial variance σ_0^2 to 20 × 20. Thereafter, we calculated the diagonal pixel length l_i of each target in the current frame $t+1$ to update the mean and variance of the single Gauss model. When updating the model with l_i, it is necessary to determine whether the diagonal pixel length l_i belongs to this Gauss model. If it belongs to this model, it will be updated to the model. The updated formula is as follows:

$$\begin{cases} \mu_{t+1} = \mu_t; \quad \sigma_{t+1}^2 = \sigma_t^2, \quad if \quad |l_i - \mu_t| > \lambda \cdot \sigma_t \\ \mu_{t+1} = (1-\alpha) * \mu_t + \alpha * l_i; \quad \sigma_{t+1}^2 = (1-\alpha) * \sigma_t^2 + \alpha * (l_i - \mu_t)^2, \quad ELSE \end{cases} \tag{10}$$

where λ is a threshold to determine whether it belongs to the model, α indicates the speed of updating the model. In the system, $\lambda = 2.55$ and $\alpha = 0.5$. In this way, the diagonal pixel length μ_{t+1} of the car can be obtained in the current frame $t+1$. Then, the R_{t+1} can be calculated, and the speed $v_i^{(t+1)}$ of the target $T_i^{(t+1)}$ is given by

$$v_i^{(t+1)} = D_i^{(t+1)} \cdot \frac{R_{t+1}}{\Delta t}. \tag{11}$$

In addition, in order to ensure that the speed of the vehicle changes smoothly, we used the weighted average method to calculate the average speed of the target $T_i^{(t+1)}$ as the final vehicle speed $V_i^{(t+1)}$; the calculation formula is as follows:

$$V_i^{(t+1)} = \begin{cases} \omega_0 \cdot v_i^{(t+1)} + \omega_1 \cdot V_i^{(t)} + \omega_2 \cdot V_i^{(t-1)} + \omega_3 \cdot V_i^{(t-2)} + \omega_4 \cdot V_i^{(t-3)}, \quad if \quad t >= 3 \\ v_i^{(t+1)}, \quad ELSE \end{cases} \tag{12}$$

In the formula, $\omega_0 + \omega_1 + \omega_2 + \omega_3 + \omega_4 = 1$, and we set $\omega_0, \omega_1, \omega_2, \omega_3, \omega_4 = 0.8, 0.07, 0.05, 0.04, 0.03$.

This speed estimation algorithm uses the real size of the car as a priori information to establish a mapping relationship and only uses the image information from a single camera to adaptively recover the pixel scale and estimate the accurate speed of the vehicle in the scene.

3. Experiments

In this section, in order to evaluate the performance of the presented adaptive framework for multi-vehicle ground speed estimation in airborne videos, we describe the building of a traffic dataset based on UAV and conduct simulation experiments and real experiments. In simulations, in order to improve the authenticity, both the simulation environment and the simulation system used were very similar to the real experiment. With the help of the simulation environment, we performed quantitative and qualitative analyses of the experimental results and evaluated the vehicle speed estimation performance of our system. In real experiments, we first tested the performance of the system on the established dataset, including the vehicle positioning performance and speed estimation performance. Then, we had the drone fly over a road with landmarks of known distances and then conducted a quantitative velocity measurement experiment by comparing with true values. Additionally, we carry out an a priori information evaluation experiment to verify the correctness of the prior values selected in the real environment.

3.1. Traffic Dataset Based On Uav

In this part, we introduce our traffic training dataset for vehicles detection in airborne videos. This dataset contains 13,900 images, including our own traffic dataset based on UAV and a public traffic dataset UA-DETRAC, as shown in Figure 5.

The self-built UAV-based training dataset contains 3000 aerial images, which are captured by the UAV DJI-MATRICE 100 with a Point Grey monocular camera in six scenes, and the image resolution is 1280 × 960. By changing the camera angle and UAV flying height during the shooting process, the diversity of the appearance and size of vehicles in the dataset is increased. Some images in the self-built UAV-based training dataset are shown in Figure 5a. We can see that the background of the dataset is complex; there are a large number of trees, shadows, large buildings, etc. Moreover, there are numerous small vehicles in the image are small. We used the $LabelImg$ tool to label these images for the training network. In addition to this aerial dataset, we also selected 10,900 pictures of the public traffic dataset UA-DETRAC to supplement the training set, as shown in Figure 5b. This dataset was taken by a high fixed surveillance camera. Due to the high position of the camera, we know that many vehicles are observable in the image, and the size of the vehicles are moderate, as seen by the figure. These images can be regarded as data taken by the drone at low altitude. In this paper, we used the deep learning method YOLOv3 to detect ground vehicles, and the dataset $Train\_TrafficData$ was used to train the YOLOv3 network model.

In order to evaluate the vehicle detection and speed estimation performance of the proposed system, we established a UAV-based test dataset $Test\_TrafficData$. This test set consisted of five scenarios, each with more than 5000 images. These five scenes were intersections, parking entrances, roads, highways, and streets. They were common traffic monitoring areas with generally complicated backgrounds, a large number of vehicles, and the potential for violations such as vehicle overspeed. At present, we have posted this dataset on a website: https://shuochen365.github.io/. Furthermore, due to the constant research in this project, we will continue to expand this dataset.

3.2. Simulation in Airsim

3.2.1. Simulation Platform

In the simulation experiments, we used the AirSim to build the simulation environment in which the working process of the proposed system was simulated. AirSim is an open-source, cross-platform simulator for drones, built on Unreal Engine. It provides physically and visually realistic simulations. Specifically, we first chose the urban highway as our simulation environment, as shown in Figure 6a. It can be seen that the background of the environment is complex, and there are shadows, trees, and so on. Thereafter, we added a UAV and a car to the environment.

The AirSim simulation environment provides us with various types of visual data such as RGB images, depth maps, disparity maps, and color segmentation maps, and these can be acquired in all directions of the UAV. Then we selected RGB images as the input of the simulation system and set the image resolution to 960 × 540 at 5 fps. The input image is shown in Figure 6a, and we can see that the target is small in the acquired image. The components of the simulation system are shown in Figure 6b, including two computers and an ethernet crossover cable. Among them, a computer (Client) was used to build a simulation environment, and output scene images. Another computer (Server) acted as a processor to implement the algorithm and obtain the speed estimation result of the vehicle.

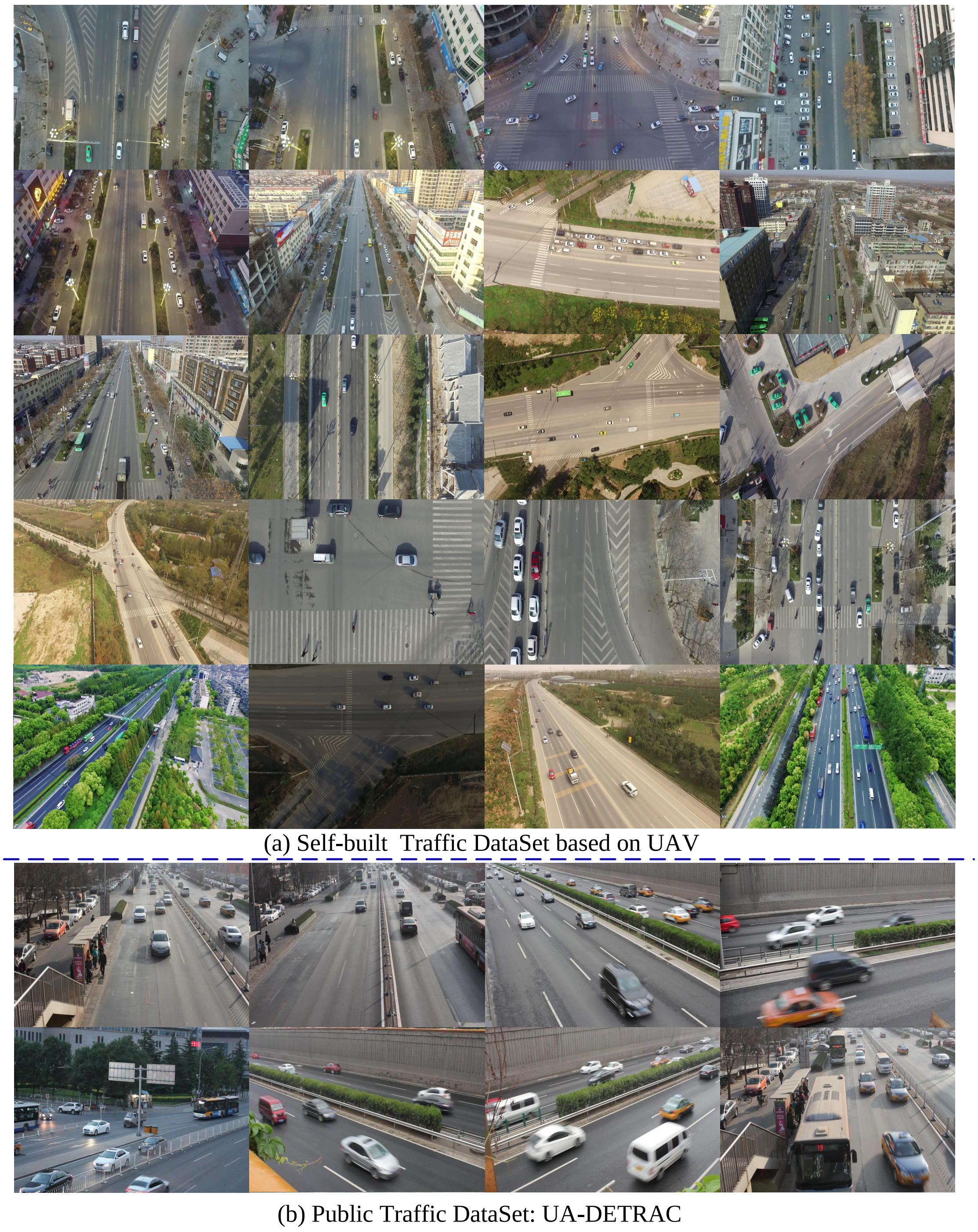

(a) Self-built Traffic DataSet based on UAV
(b) Public Traffic DataSet: UA-DETRAC

Figure 5. Some images of our training dataset, including (**a**) a self-built traffic dataset based on an unmanned aerial vehicle (UAV), where the images were captured by DJI-MATRICE 100 with a Point Grey monocular camera; (**b**) the public traffic dataset UA-DETRAC [44].

3.2.2. Simulation Results

On the constructed simulation platform, we designed six vehicle speed measuring processes to test the performance of our system. These included variable motion of the vehicle when the UAV and vehicle moved in the same direction, uniform motion when the UAV and vehicle moved in the same direction, variable motion when the UAV was moving in the same direction as the vehicle and the altitude of the UAV was constantly changing, variable motion when the UAV and vehicle moved in opposite directions, uniform motion when the UAV and vehicle moved in opposite directions, and variable motion when the UAV was stationary. We conducted a quantitative analysis of the speed estimation results in these six cases.

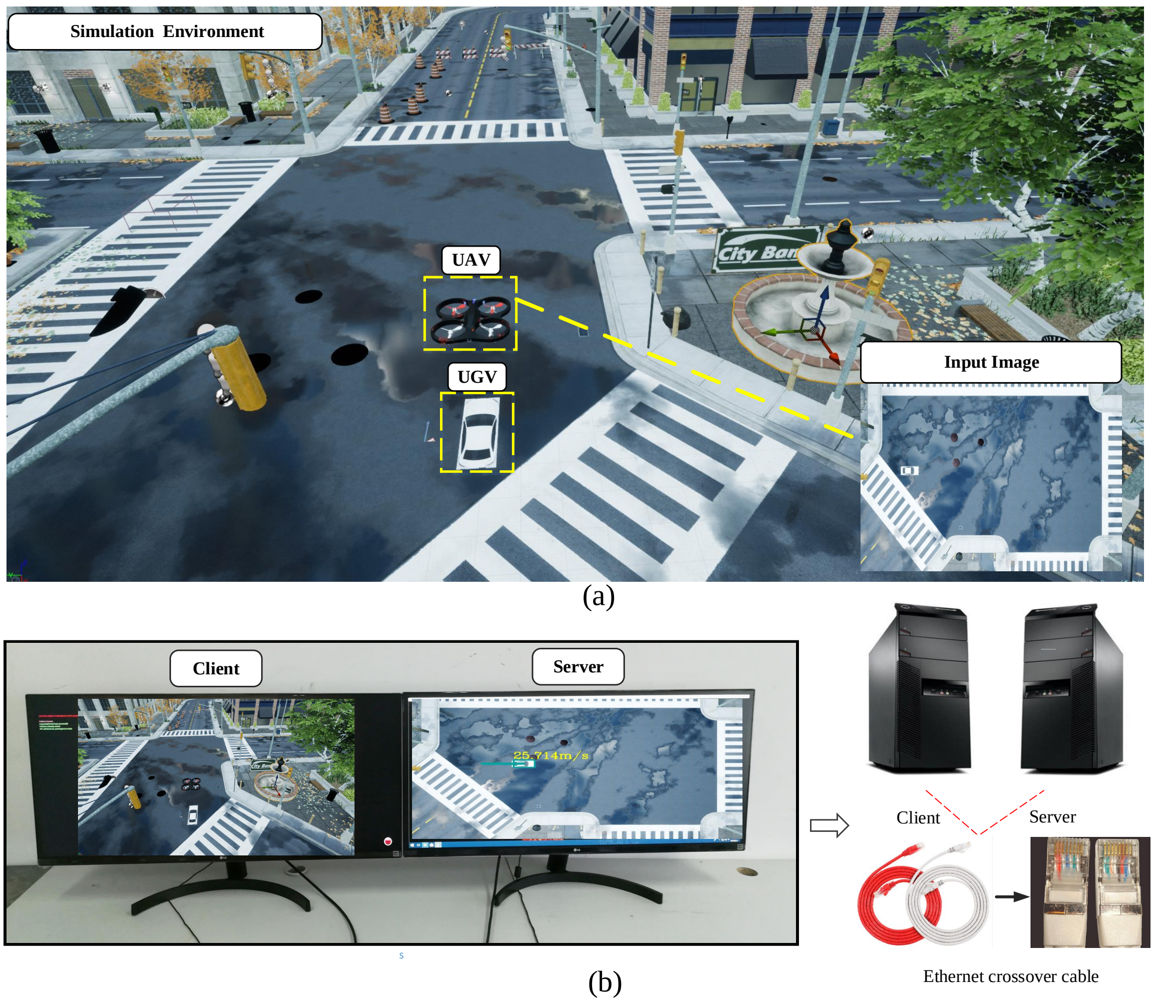


Figure 6. The demonstration of the simulation platform: (**a**) construction of the simulation environment, including monitoring scenes, a car, and a UAV; (**b**) the components of the simulation system, including two computers and an ethernet crossover cable.

It should be noted that in the simulations, we did not directly use the YOLOv3 network model trained with real data to detect the vehicle with simulation data. Although the simulation environment was very realistic, the appearance of the simulation car was different from that of a real car. If the YOLOv3 network model trained with real data was used directly in simulation, the detection effect would be reduced and the speed measurement process would be affected. Therefore, for the sake of ensuring the accuracy of vehicle detection in simulation, we especially built a training dataset and trained a YOLOv3 model for simulating car detection. This simulation training dataset included 6000 images, which were acquired in the simulation environment. During the acquisition process, we increased the diversity of the dataset by changing the camera angle and UAV flying height. After testing, the trained network model has good vehicle detection performance and could be used for vehicle speed measurement in the simulations. In addition, the vehicle size in the simulation environment was different from that in the real environment. The actual size of the simulated vehicle was 2.9 m in length, 1.1 m in width, and about 3.1 m in the diagonal distance. Therefore, in the simulation, the prior information was set to 3.1 m, which was used to adaptively estimate the mapping ratio of the current frame by determining the diagonal pixel length of the car in the image. Although this prior information is different from the system in the real environment, the two prior values were determined according to the actual vehicle size in the experimental environment.

The experimental results are shown in Figures 7 and 8. In the quantitative analysis, the performance of the algorithm was evaluated by comparing the simulation data with the true values provided by the simulation environment; the average error of speed measurement is shown in Table 1. Note that our system has an initialization process when measuring the speed of the vehicle appearing in the scene. During the initialization process, we set the speed to 0. This process mainly consists of two stages. First, when the vehicle is detected and tracked to obtain its trajectory, we believe that a target with more than three frames in succession is valid.

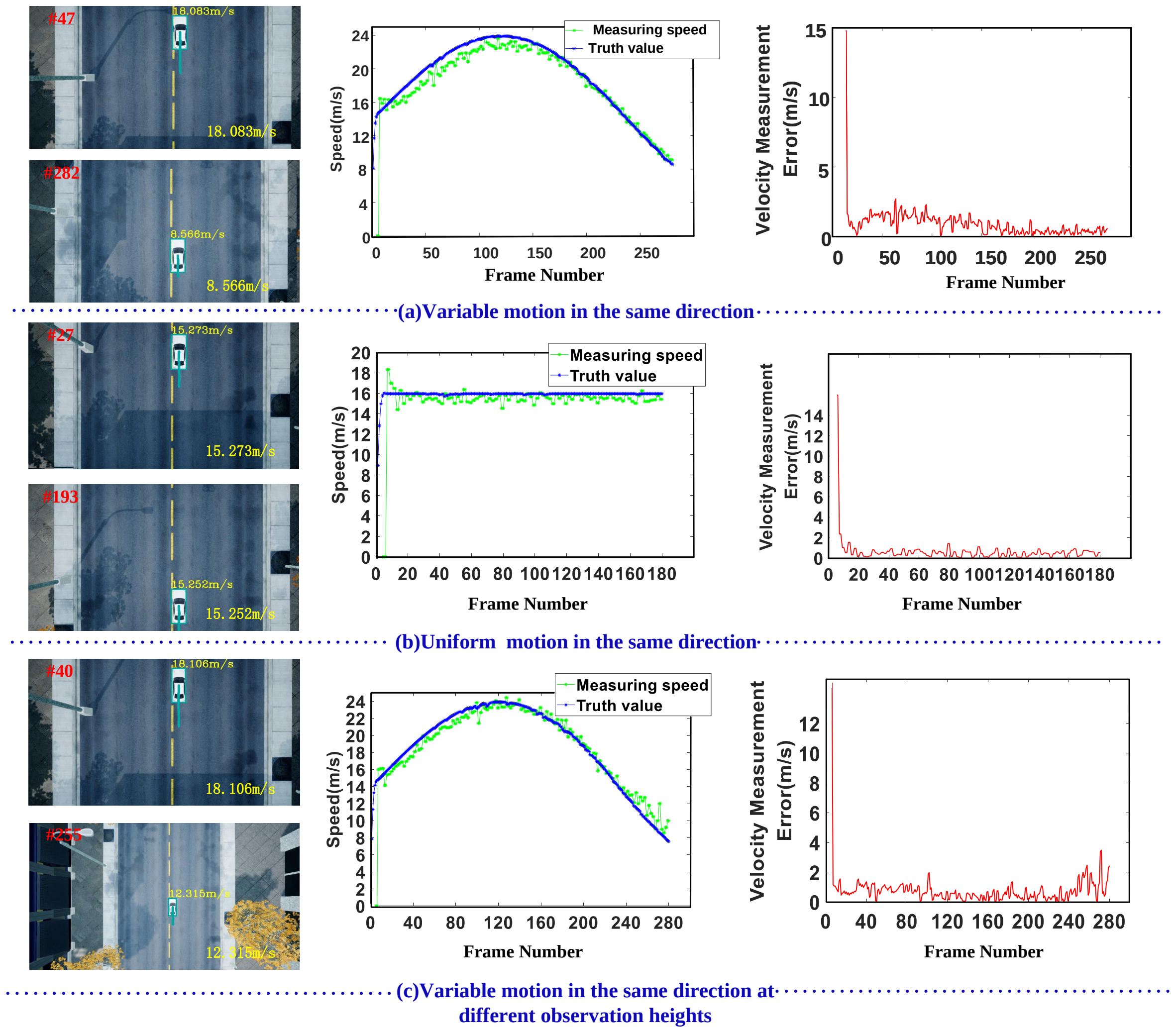


Figure 7. Vehicle speed estimation under different monitoring conditions, including (**a**) variable motion when the UAV and vehicle moved in the same direction; (**b**) uniform motion when the UAV and vehicle moved in the same direction; (**c**) variable motion when the UAV was moving in the same direction as the vehicle, and the altitude of the UAV was constantly changing.

Second, when calculating the vehicle speed, we estimated the average speed for a short period of time ($\Delta t = Q \cdot Fps$). Thus, the effective speed can only be calculated if the target has been valid for longer than Δt. Otherwise, we set the speed to 0. In the simulation experiment, we set $Q = 4$. This initialization process can be seen from the measuring speed curves in Figures 7 and 8.

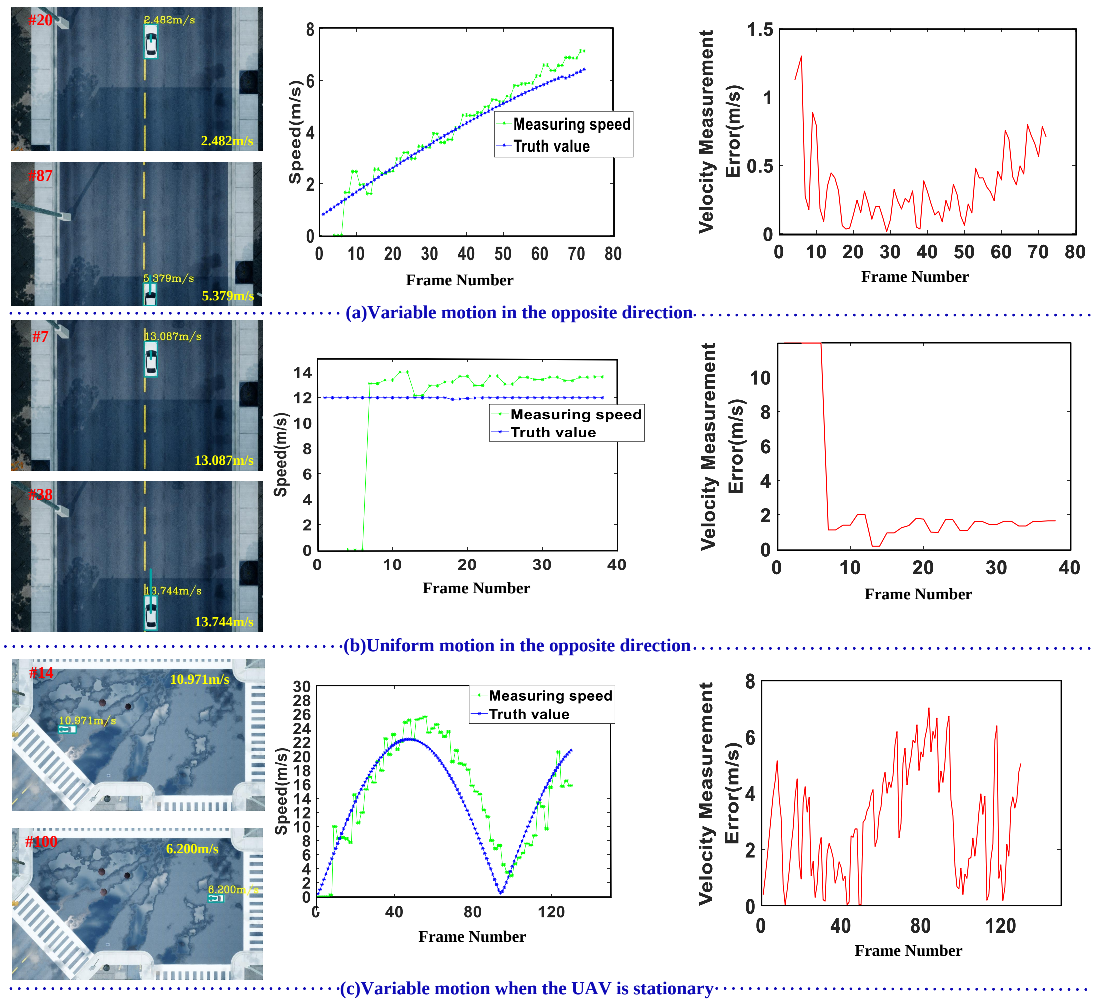


Figure 8. Vehicle speed estimation under different monitoring conditions, including (**a**) variable motion when the UAV and vehicle moved in opposite directions; (**b**) uniform motion when the UAV and vehicle moved in opposite directions; (**c**) variable motion when the UAV was stationary.

Figure 7 shows three situations when the UAV moved in the same direction as the vehicle. In (a) and (b), the monitoring height of the UAV was set to 10 m, which is constant. From Figure 7a, we can see that the vehicle is moving at a variable speed (8 m/s $\rightarrow$ 24 m/s $\rightarrow$ 8m/s). In the process of acceleration (Frame 0$\sim$120), the measuring speed is generally slightly lower than the true value, and the average error is about 1.5. In the process of deceleration, the measurement speed is very close to the true value, and the average error is about 0.5. Table 1 shows that throughout the speed measurement process, the average true speed is 14.168 m/s, the average measurement speed is 13.532 m/s, and the average measurement error is 0.777 m/s. From Figure 7b, we can see that when the vehicle is moving at a constant speed (16 m/s), and the measuring speed fluctuates around the true value and tends to be stable (about 16 m/s) during the speed measurement process (excluding initialization). The measurement error decreases gradually, averaging no more than 1. Table 1 shows that the average measurement error is approximately 0.9 m/s. In Figure 7c, the monitoring height of the UAV is variable, and the range of variation is 10 m$\sim$20 m. In this process, the altitude change of the UAV mainly occurs from Frame 200 to Frame 280. We can see that when the height is constant, the speed measurement accuracy is high, and the average error is about 0.5. When the height changes, we can see that although the error increases, the error average is about 1.5. In the process of speed measurement, we can see from Table 1 that the average true speed is 18.780 m/s, the average measurement speed is 18.739 m/s, and the average measurement error is about 0.7 m/s. In short, the measurement accuracy is acceptable. Moreover, this process shows that the proposed system can indeed adjust the mapping ratio adaptively according to the size of the target in the image.

Figure 8a,b shows two situations when the UAV moved in the opposite direction to the vehicle. From Figure 8a, we can see that when the vehicle moved at an incremental speed (1 m/s $\rightarrow$ 6 m/s), the measurement error average was about 0.5. Table 1 shows that the average true speed was 5.557 m/s, the average measurement speed was 5.876 m/s, and the average measurement error was about 0.4 m/s. From Figure 8b, we can see that when the vehicle was moving at a uniform speed (12 m/s), the measuring speed was generally slightly lower than the true value, and the measurement error fluctuated around 1.5. Table 1 shows that throughout the speed measurement process, the average true speed was 11.965 m/s, the average measurement speed was 13.320 m/s, and the average measurement error was about 1.355 m/s. Figure 8c shows the situation where the UAV was stationary, and the monitoring height of the UAV was set to 20 m. In the process, the vehicle speed changed greatly (0 m/s $\rightarrow$ 23 m/s $\rightarrow$ 0 m/s $\rightarrow$ 23 m/s). The measurement speed fluctuated around the true value during acceleration, and the average measurement error was about 2. During deceleration, it was generally greater than the true value, and the measurement error average was around 4. Table 1 shows that the average measurement error was approximately 3.5 m/s. In this process, because the target size in the image was small, there was a small jump in the position of the detection bounding box, which affected the measurement accuracy.

In short, the measurement error average was basically less than 1.5 m/s, and the measurement accuracy was acceptable. These quantitative results and analyses of the simulation experiment prove that the proposed adaptive framework for multi-vehicle ground speed estimation in airborne videos can obtain effective and robust vehicle speed estimation results without auxiliary equipment such as GPS under various conditions. That is to say, in this framework, it is feasible to estimate the vehicle speed by adaptively recovering the pixel scale with the actual vehicle size as a priori information.

Table 1. Average speed measurement error in the simulation environment

Motion Mode	Average True Speed (m/s)	Average Measured Speed (m/s)	Average Measurement Error (m/s)
Variable motion in the same direction	14.168	13.532	0.777
Uniform motion in the same direction	15.952	15.081	0.924
Variable motion in the same direction and flight altitude is changing.	18.780	18.739	0.692
Variable motion in the opposite direction	5.557	5.876	0.419
Uniform motion in the opposite direction	11.965	13.320	1.355
Variable motion when the UAV is stationary	23.197	25.196	3.481

3.3. Real Experiments

In order to further validate the effectiveness of the vehicle speed estimation system, we conducted experiments in the real environment. The hardware structure of the system was very simple. In this experiment, DJI-MATRICE 100 was used as the UAV, the monocular camera was a Point Grey camera, and the computer used had an Intel i5-8400 CPU, 8GB RAM, and a Navida Geforce gtx1060 GPU, 6 GB video memory. The monocular camera was fixed under the UAV with the lens facing down, and the camera parameters are shown in Table 2. As can be seen from Table 2, the field of view angle of the system equipped with the Point Grey camera was 84°. In this section, we first tested the detection and tracking performance of the system, which is the basis of the system. Then, the vehicle speed estimation performance was evaluated on the established UAV-based traffic dataset, and the time performance of the system was analyzed. Thereafter, we had the drone fly over a road with landmarks of known distances and conducted a quantitative velocity measurement experiment by comparing with true values in the real environment. Finally, in order to verify the correctness of the prior values selected in the real system, we conducted an a priori information evaluation experiment.

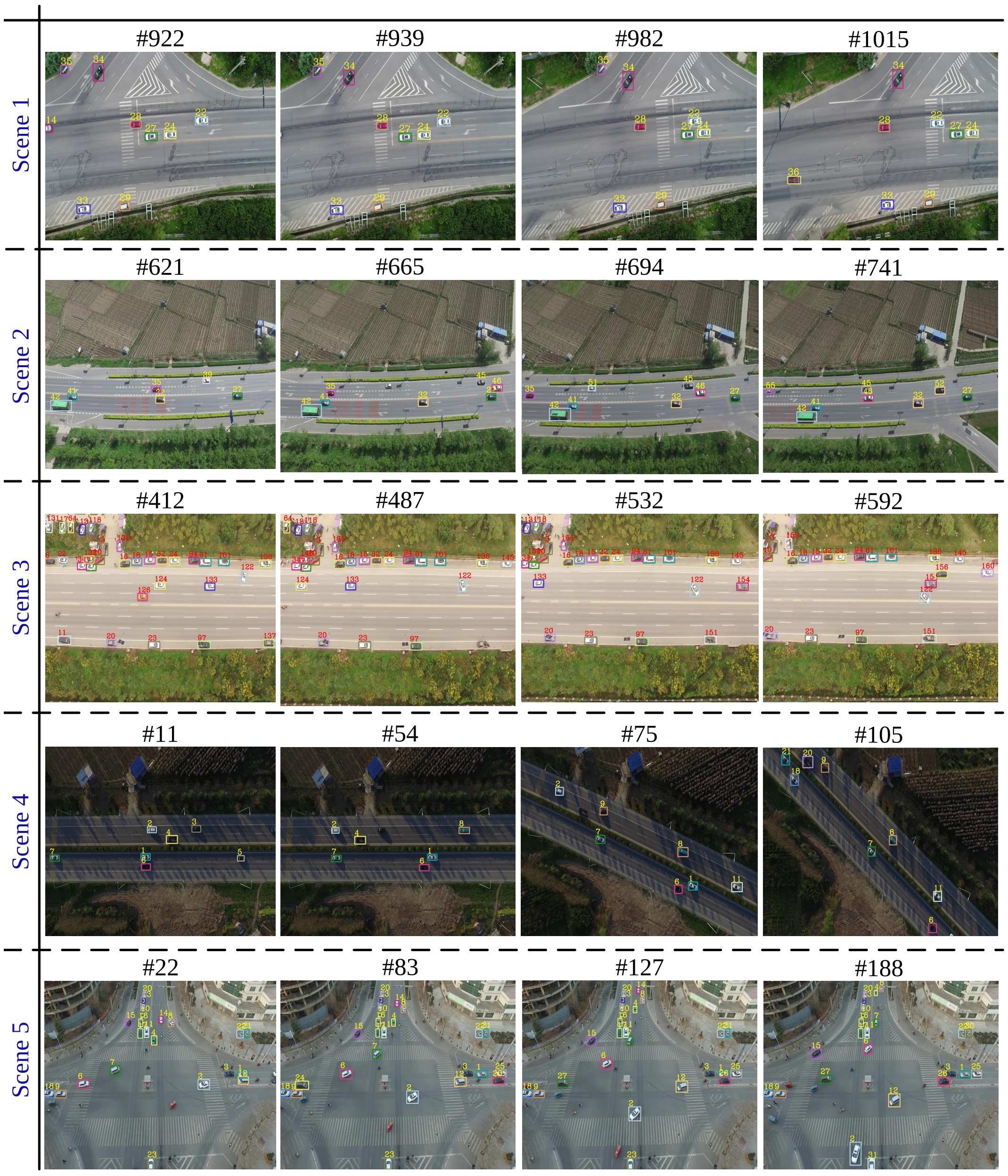


Figure 9. Vehicle positioning results for our UAV-based traffic dataset. The test dataset contains five common traffic monitoring scenarios: an intersection, country road, parking entrance, highways, and crossroads. The background of these scenes is complex, with shadows, buildings, trees, etc., and there are numerous small-sized vehicles in the scene.

Table 2. The camera and lens parameters.

Point Grey Camera	Specification	Parameter
	Sensor	CMV4000-3E12
	Angle of View	84°
	Resolution	1920 × 1080 pixels
	Frame rate	60 fps
	Power consumption	4.5 W
	Operating temperature	−20 °C∼50 °C

3.3.1. Vehicle Detection and Tracking Experiments

In the system, in order to estimate the position change of the vehicle over a period of time, we obtained a series of positions for the vehicles in the UAV video through the detection and tracking algorithm. The detection algorithm used was YOLOv3, and the network model was trained on a self-built dataset $Train\_TrafficData$ to achieve vehicle detection in the aerial scene. In the course of training, we did not directly use the self-built training dataset $Train\_TrafficData$ to re-train the YOLOv3 network model. Instead, we used the $Train\_TrafficData$ dataset to fine-tune the model on the pre-trained network model to improve the accuracy of vehicle detection in aerial images. To be specific, first, we chose the YOLOv3-608 model on the YOLOv3 website as the pre-training model. We know that this model is trained with COCO datasets (tens of thousands of images), which can detect more than 9000 kinds of objects, and the detection effect is very good. Then, the pre-trained network parameters were used as the initialization parameters of the new training model, and the new network model and weight parameters were trained by using the training dataset $Train\_TrafficData$. This not only reduced the training time but also improved the vehicle detection accuracy of the network model in aerial images.

Based on this, we used a tracking-by-detection algorithm to get the trajectory of the target. This method measures the similarity by measuring the IOU of the detection bounding box; then, it achieves fast and effective target tracking. Our system showed good positioning performance on the established dataset $Test\_TrafficData$; the experimental results are shown in Figure 9. The test dataset contained five common traffic monitoring scenarios: an intersection, country road, parking entrance, highways, and crossroads. As can be seen from the figure, there were few partial occlusions and complete occlusions, which is beneficial for target detection and tracking. However, their backgrounds were complicated; there were many trees in scenes 1, 2, and 3 and shadows in scene 4. In addition, the backgrounds of the images changed due to the UAV movement. Despite these difficulties, our vehicle detection and tracking algorithms were still able to obtain good vehicle positioning results in various aerial videos. Even in the case where the number of vehicles was large and the size was small, such as scenes 3 and 5, effective positioning results were still obtained. Specifically, stationary targets in the motion background, such as target 22 of scene 1 and targets 97 and 101 of scene 3, were correctly detected and tracked. For the moving targets in the motion background, such as the targets 41 and 32 of scene 2, target 7 of scene 4, and targets 2 and 6 of scene 5, we generally obtained stable positioning results. There were some missing targets, such as target 3 of Frame 54 in scene 4. The black car was very similar to the background with shadows, so it was difficult to detect, resulting in the failure of target tracking and positioning.

In general, the detection and tracking algorithm in this system was able to obtain effective and robust positioning results in various complex scenarios. These results are very important for the vehicle speed measurement system.

3.3.2. Vehicle Speed Estimation Experiments

The accuracy of vehicle speed estimation in the system is not only related to the vehicle detection and tracking performance but is also related to the motion compensation effect and the estimation of the mapping ratio. Based on the good positioning performance, we carried out further qualitative experiments on our UAV-based test dataset $Test\_TrafficData$ to test the effect of motion compensation, to estimate the car's diagonal pixel length, and to calculate the speed. The experimental results are shown in Figure 10.

From Figure 10, we can see the optical flow after motion compensation, the mean changes of the Gaussian model, the speed measurement results, and the speed changes of some targets throughout the monitoring process. First, we know that the motion compensation algorithm can eliminate the background motion in adjacent frames caused by camera movement. From the compensated optical flow in Figure 10, we can see that when there was more texture on the road surface, such as in scenes 1 and 5, the compensation effect was better and the compensated optical flow points were basically in the vehicles. When there was less texture on the road surface and more background texture, the compensation effect reduced and some compensated optical flow points appeared on the road surface, such as in scenes 2 and 3. Overall, although there were some errors, the effect of motion compensation was obvious. Thereafter, the mean changes in the single Gaussian model in Figure 10 show the changes in the car's diagonal pixel length. It can be seen that during the monitoring of each scene, the mean fluctuated greatly at the initial time and gradually stabilized at a fixed value over time. For example, in scene 1, the mean was stable at around 73.8. Therefore, the mapping ratio of the scene was always the same. Since the height of the drone was constant in the process of video capture, the mapping ratio was indeed constant. This indicates that it is feasible to adaptively recover the pixel scale by estimating the Gaussian model.

Table 3. Time performance of the proposed system **(ms/frame)**.

Scene	**YOLOv3 [31]**	**Speed Measurement**			**System**	**Fps**
		Tracking	**Motion Compensation**	**Total**		
Scene1	45.234	0.412	42.792	23.548	57.565	**17.372**
Scene2	44.890	0.317	42.207	23.554	55.023	**18.174**
Scene3	50.674	0.700	46.565	26.712	60.369	**16.565**
Scene4	46.055	0.256	43.327	23.312	59.402	**16.834**
Scene5	43.334	0.450	41.715	24.098	56.220	**17.787**
Average	46.037	0.427	43.321	24.245	57.716	**17.346**

Finally, from the speed estimation results in Figure 10, we can see that for each vehicle that appeared in the scene, there was a process of speed initialization. For stationary targets in the scene, such as target 4 of scene 1 and target 4 of scene 3, their measurement speed was about 0.5 m/s on average, and the measurement accuracy was high. For the moving target in the scene, such as target 4 in scene 2, it can be seen that the initial speed of the target when entering the monitoring range was about 16 m/s, and deceleration occurred in Frame 600 to 900. In Frame 900–1000, the target stopped for a moment because of encountering a red light and then accelerated to 12 m/s and moved out of the surveillance field of view. It is obvious that the proposed UAV-based vehicle speed estimation system is very meaningful for monitoring traffic safety. In the process of speed measurement, the UAV of the system can hover to continuously monitor a fixed scene. It can also move horizontally or up and down to expand the monitorable area. Moreover, the speed measuring system does not depend on any auxiliary equipment such as GPS or reference markers, which makes the system easy to use and suitable for practical application.

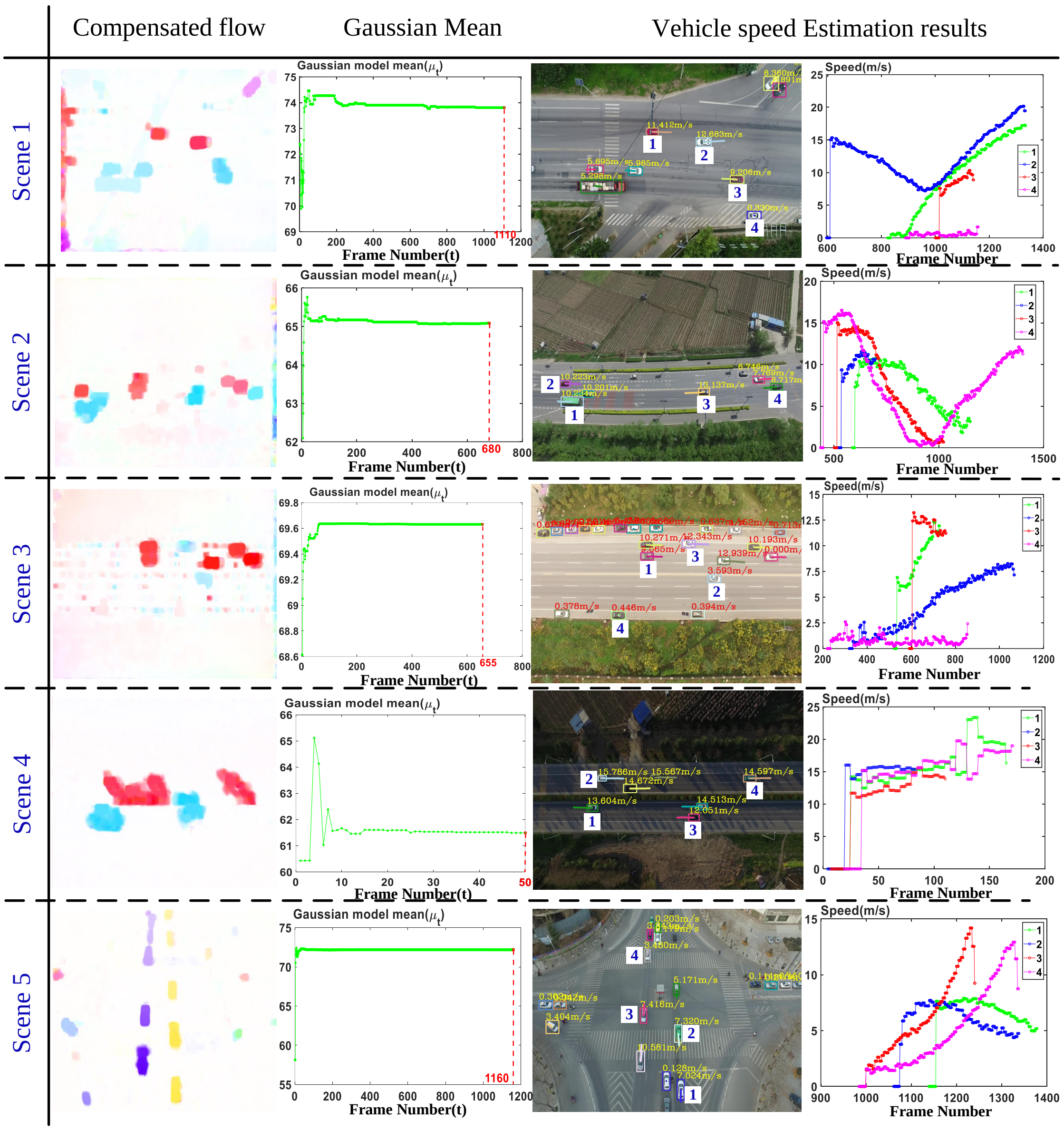


Figure 10. Vehicle speed estimation process and results for our UAV-based traffic dataset, including motion compensated optical flow, the mean changes of the Gaussian model, speed measurement results, and speed changes of some targets throughout the monitoring process.

Additionally, we tested the time performance of the proposed system on 2000 images from each scene, and the computing time is shown in Table 3. In the system, the computer used had an Intel i5-8400 CPU, 8GB RAM, and a Nvidia Geforce gtx1060 GPU, 6 GB video memory. The algorithm was implemented using two processes. One process was the YOLOv3 algorithm, and its average calculation time was 46.037 ms. Another process was speed measurement, including target tracking and motion compensation. Although motion compensation is time-consuming (about 43.321 ms), we set the system to perform motion compensation every five frames, and the average computation time of this process was 24.245 ms. The average calculation time of the system was 57.716 ms. Therefore, our system can quickly obtain effective and robust vehicle speed estimation results and is suitable for application in actual scenarios.

In addition to the experimental results presented in the paper, we also tested several scenes of our dataset and produced a video demo to demonstrate the effectiveness of our system.

The dataset is posted online at https://shuochen365.github.io/. The video demo is displayed in the Supplementary Materials. In this demo, we showed the effective and robust speed measurement results of ground vehicles in five different environments (Video S1).

3.3.3. Vehicle Speed Quantitative Experiments

In order to get the true value of velocity in a real environment, we had the drone fly over a road with landmarks of known distances and obtained the actual speed of all the vehicles in the image using this along with the time information from the video. In the experiment, the monitoring range of UAV is shown in Figure 11, where road segment 1 and road segment 2 are used as landmarks in the scene, and the actual distances of these road segments were measured as 33.06 m. We divided these road segments more carefully and used the time information from the video to calculate the actual speed at each segment point as the true value of the vehicle speed at that location. On the basis, we designed four vehicle speed measuring processes to test the performance of our system: a UAV and a vehicle moving in the same direction, a UAV and a vehicle moving in opposite directions, a UAV hovering in the scene, and a UAV flying from low to high in the scene. We conducted a quantitative analysis of the speed estimation results of these four cases. The performance of the algorithm was evaluated by comparing the true values of segment points, and the experimental results are shown in Figure 12.

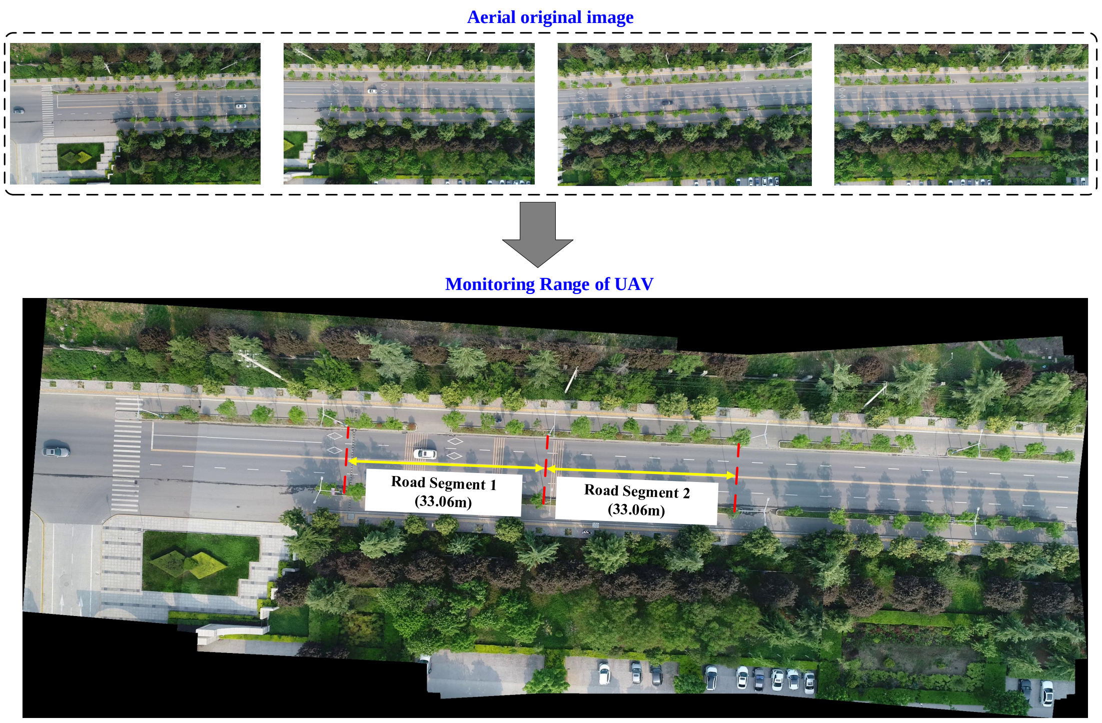
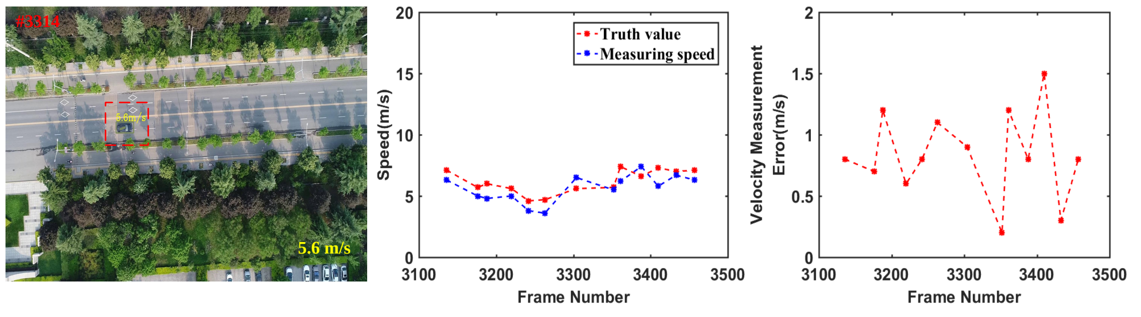
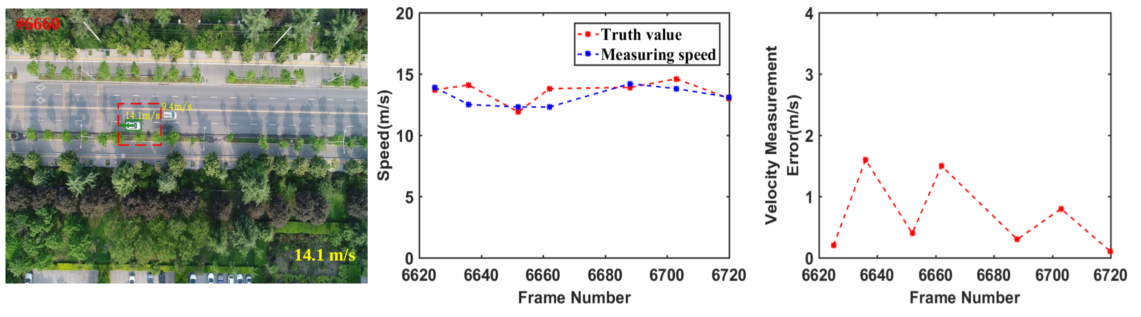


Figure 11. The figure shows the monitoring range of the UAV in the qualitative experiment of the real environment, where road segment 1 and road segment 2 are used as landmarks in the scene, and the actual distances of these road segments were measured to be 33.06 m.

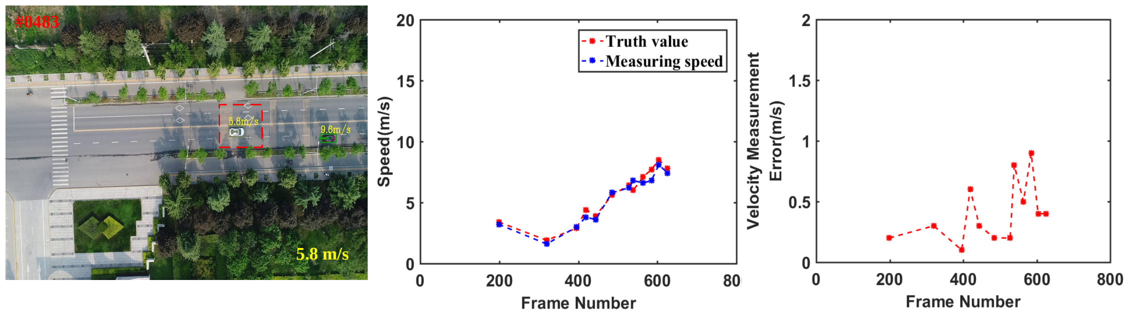
Figure 12a shows the speed estimation results of segment points and the measurement error when the UAV and vehicle moved in the same direction. In this case, we set up 13 segment points on the road and select a car in the scene to test the system performance, as shown in the red rectangular box in the figure. The speed change curve shows that the true value of vehicle speed at these 13 points was about 6 m/s, and the measured speed of the system fluctuated near the true value. The velocity measurement error curve shows that the average error was about 0.7 m/s. From Figure 12b, we can see the speed estimation results of segment points and the measurement error when the UAV and vehicle moved in opposite directions. In this case, we set up seven segment points on the road. The speed change curve shows that the true value of the vehicle speed was about 14 m/s on the road segment with known distances. The measuring speed fluctuated around the true value, and the fluctuation became smaller and smaller, which meant that the measurement error was also reduced (about 1 m/s → 0.4 m/s). The velocity measurement error was about 0.8 m/s throughout the speed measurement process.



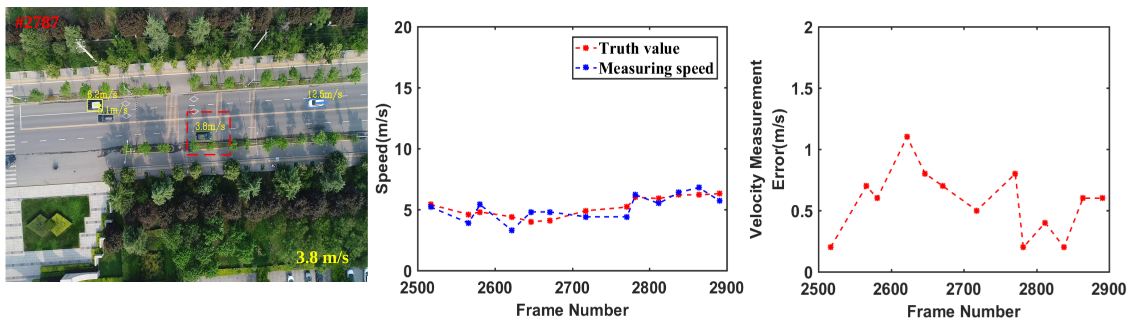
(a) UAV and vehicle move in the same direction



(b) UAV and vehicle move in the opposite direction



(c) UAV hovering in the scene



(d) UAV flies from low to high in the scene

Figure 12. Vehicle speed estimation under different monitoring conditions in the real environment, including (**a**) a UAV and a vehicle moving in the same direction; (**b**) a UAV and a vehicle moving in opposite directions; (**c**) a UAV hovering in the scene; and (**d**) a UAV flying from low to high in the scene.

Figure 12c shows the speed estimation results of the segment points and the measurement error when the UAV hovered in the scene, and the flight height of the UAV was about 80 m. In this case, we set up 12 segment points on the road and selected a car in the scene to test the system's performance. From the speed change curve, we can see that the vehicle was moving at a variable speed (3 m/s $\rightarrow$ 9 m/s), and the measurement speed was very close to the true value; the average measurement error was about 0.4 m/s. In Figure 12d, the speed estimation results of segment points and the measurement error when the UAV flew from low to high (50 m $\rightarrow$ 80 m) in the scene are shown. In this case, we set up 13 segment points on the road. The speed change curve shows that vehicle speed changed slightly, and the true value and measured speed were basically 5 m/s. The velocity measurement error change curve shows that the error fluctuated around 0.5 m/s, and the average error was about 0.6 m/s.

In summary, under these four different experimental conditions, we found that the system performance was the best and the average velocity measurement error was the smallest, about 0.4 m/s, when the UAV hovered in the scene. When the UAV moved horizontally, the average measurement error was about 0.75 m/s. The reason for the increase in the error may be due to the existence of a certain error in motion compensation. When the UAV moved up and down, the measurement error did not change dramatically with the change of altitude, and the average measurement error was about 0.6 m/s, which proves that the proposed system can adjust the pixel scale adaptively to ensure measurement accuracy. Therefore, the quantitative analysis results in the real environment prove that the proposed system can obtain effective and robust vehicle speed estimation results under various conditions. The average measurement error was less than 1 m/s, and the measurement accuracy was acceptable for practical applications.

3.3.4. A Priori Information Evaluation Experiments

In this system, we used the car's diagonal length as a priori information to adaptively estimate the mapping ratio of the current frame. In the real experiments, the prior information was set to 4.8 m. To prove the correctness of the prior values selected in the real system, we had the drone fly over a road with landmarks of known distances to obtain the true value of vehicle speed in the image and conducted an a priori information evaluation experiment. Specifically, in the experiment, the monitoring range of UAV is shown in Figure 11. In order to obtain the velocity measurement error distribution with respect to the prior values, we used 0.2 as the step size and selected 11 prior values around 4.8 m to obtain their respective average velocity measurement errors. These prior values fell within the range of [3.8 m, 5.8 m]. Then, we designed three different experimental conditions: a UAV hovering at a height of 50 m in the scene, a UAV hovering at a height of 70 m in the scene, and a UAV and vehicle moving in the same direction. The distribution of velocity measurement errors with different prior values under three monitoring conditions in a real environment are shown in Figure 13.

From Figure 13, we can see that under the three different experimental conditions, the velocity measurement error distribution with different prior values was an open-up parabola. On the distribution curve, the measurement error was smallest with a prior value of 4.8 m, and the farther the prior value was from 4.8 m, the bigger the measurement error was. Moreover, the distribution had some symmetry for a prior value of 4.8 m. For a prior value with the same distance at 4.8 m, such as 4.6 m and 5.0 m, the corresponding measurement error was approximately equal. According to this rule, in many practical scenarios, the speed measurement effect should be the best with a prior value of 4.8 m.

Comparing the error distribution under three experimental conditions and further analysis, we found that the measurement error when UAV was hovering was smaller than the error when the UAV was moving. Figure 13a,b shows that when the UAV hovered in the scene, the measurement error at a prior value of 4.8 m was about 0.5 m/s, regardless of the flight altitude (50 m or 70 m). This indicates that the measurement error is independent of the UAV's flight altitude. Figure 13c shows that the measurement error at a prior value of 4.8 m was about 0.7 m/s when the UAV was moving. The reason for the increase in error may be due to the existence of a certain error in motion compensation. In short, the experimental results and analysis prove that, in most cases, the speed measurement effect of the system should be the best when the prior value is 4.8 m and the error is generally less than 1 m/s.

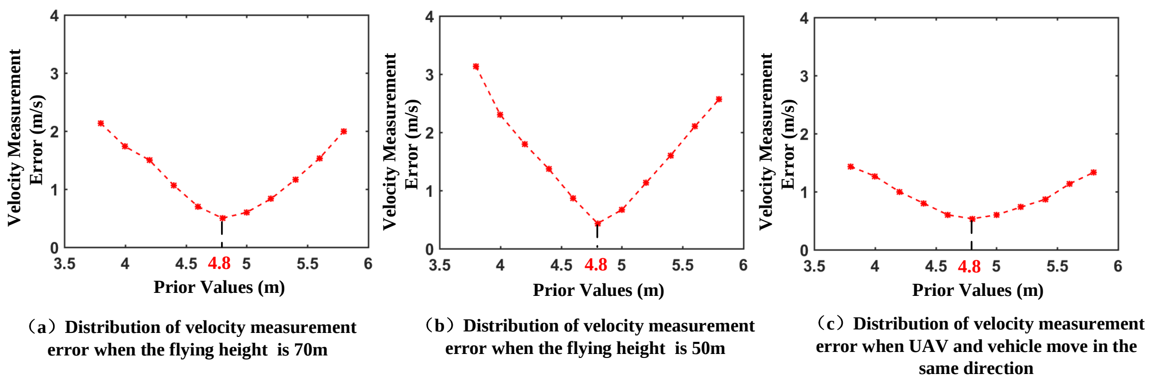


Figure 13. Distribution of velocity measurement errors with different prior values under different monitoring conditions in a real environment, including (**a**) a UAV hovering at a height of 50 m in the scene; (**b**) a UAV hovering at a height of 70m in the scene; and (**c**) a UAV and a vehicle moving in the same direction.

4. Conclusions

In this paper, we proposed an adaptive framework for multi-vehicle ground speed estimation in airborne videos. This framework has a unique ability to detect, track, and estimate the speed of ground vehicles simultaneously. Moreover, it takes the actual size of a car as the prior information and obtains effective and robust vehicle speed estimation results in various complex environments without using auxiliary equipment such as GPS. More specifically, we first built a UAV-based traffic dataset and utilized it to train the YOLOv3 network model to achieve great vehicle detection results in aerial images. Then, a series of vehicle positions were obtained using a tracking-by-detection algorithm. Meanwhile, we showed a motion compensation method based on homography. This model explores the bounding boxes to accurately estimate the homography matrix and then obtain the real trajectory of the current frame. Thereafter, we presented an adaptive vehicle speed estimation method based on the mapping relationship. This method takes the actual size of the car as prior information and estimates the vehicle size in the image by establishing a Gaussian model to achieve adaptive pixel scale recovery and speed calculation.

Finally, in order to test the performance of the proposed system, we carried out a large number of simulations on the AirSim platform as well as real experiments. In the simulations, the quantitative analysis proved that our system can measure speed with high accuracy under various conditions. Moreover, the system can adjust the mapping ratio adaptively according to the vehicle sizes on the image. In real experiments, the good positioning performance of the system was proved by the vehicle detection and tracking experiment. The results and analysis of velocity measurement experiment and quantitative experiment not only showed the validity of the whole velocity measurement process but also further proved that the system could quickly obtain effective and robust vehicle speed estimation results in various complex environments. Thereafter, we evaluated the prior value selected to prove its correctness, and the value corresponding to the speed measurement effect was deemed to be the best for many practical scenarios. There are also some problems with the system. It is not very accurate to use the statistical information of the bounding box size to estimate the pixel scale in the whole image adaptively. In the future, we will consider using SLAM to estimate the 3D information of the scene to improve the accuracy of vehicle speed estimation.

Supplementary Materials: The following are available online at www.mdpi.com/xxx/s1, Video S1: FHY-XD-UAV-DEMO.

Author Contributions: J.L., S.C., F.Z., E.L., and T.Y. designed the overall system and wrote the source code. J.L., S.C., F.Z., and E.L. participated in the research data collection, analysis and interpretation. T.Y. guided the experiments and the statistical analysis, and Z.L. supplied help with the experiments and paper revision. Additionally, they jointly designed and performed the experiments.

Funding: This work is supported by the National Natural Science Foundation of China (No. 61672429, No.61502364, No.61272288).

Acknowledgments: We would like to thank the academic editor and anonymous reviewers' valuable suggestions on this paper, and thanks to Dongdong Li, Bolin Xie, Fenglin Liu for their support of UAV aerial data collection.

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

References

1. Liu, Y. Big data technology and its analysis of application in urban intelligent transportation system. In Proceedings of the International Conference on Intelligent Transportation, Big Data Smart City, Xiamen, China, 25–26 January 2018; pp. 17–19.
2. Luvizon, D.C.; Nassu, B.T.; Minetto, R. A video-based system for vehicle speed measurement in urban roadways. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 1393–1404. [CrossRef]
3. Yang, T.; Ren, Q.; Zhang, F.; Ren, B.X.H.; Li, J.; Zhang, Y. Hybrid camera array-based uav auto-landing on moving ugv in gps-denied environment. *Remote. Sens.* **2018**, *10*, 1829. [CrossRef]
4. El-Geneidy, A.M.; Bertini, R.L. Toward validation of freeway loop detector speed measurements using transit probe data. In Proceedings of the 7th International IEEE Conference on Intelligent Transportation Systems, Washington, WA, USA, 3–6 October 2004; pp. 779–784.
5. Sato, Y. Radar speed monitoring system. In Proceedings of the Vehicle Navigation and Information Systems Conference, Yokohama, Japan, 31 August–2 September 1994; pp. 89–93.
6. Lobur, M.; Darnobyt, Y. Car speed measurement based on ultrasonic doppler's ground speed sensors. In Proceedings of the 2011 11th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM), Polyana-Svalyava, Ukraine, 23–25 February 2011; pp. 392–393.
7. Odat, E.; Shamma, J.S.; Claudel, C. Vehicle classification and speed estimation using combined passive infrared/ultrasonic sensors. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 1593–1606. [CrossRef]
8. Musayev, E. Laser-based large detection area speed measurement methods and systems. *Opt. Lasers Eng.* **2007**, *45*, 1049–1054. [CrossRef]
9. Hussain, T.M.; Baig, A.M.; Saadawi, T.N.; Ahmed, S.A. Infrared pyroelectric sensor for detection of vehicular traffic using digital signal processing techniques. *IEEE Trans. Veh. Technol.* **1995**, *44*, 683–689. [CrossRef]
10. Cevher, V.; Chellappa, R.; Mcclellan, J.H. Vehicle speed estimation using acoustic wave patterns. *IEEE Trans. Signal Process.* **2009**, *57*, 30–47. [CrossRef]
11. Zhang, W.; Tan, G.; Ding, N. *Vehicle Speed Estimation Based on Sensor Networks and Signal Correlation Measurement*; Springer: Berlin/Heidelberg, Germany, 2014.
12. Liang, W.; Junfang, S. The speed detection algorithm based on video sequences. In Proceedings of the International Conference on Computer Science Service System, Nanjing, China, 11–13 August 2012; pp. 217–220.
13. Yung, N.H.C.; Chan, K.C.; Lai, A.H.S. Vehicle-type identification through automated virtual loop assignment and block-based direction-biased motion estimation. *IEEE Trans. Intell. Transp. Syst.* **1999**, *1*, 86–97.
14. Couto, M.S.; Monteiro, J.L.; Santos, J.A. Improving virtual loop sensor accuracy for 2d motion detection. In Proceedings of the 2002 Proceedings of the Bi World Automation Congress, Shanghai, China, 10–14 June 2002; pp. 365–370.
15. Alefs, B.; Schreiber, D. Accurate speed measurement from vehicle trajectories using adaboost detection and robust template tracking. In Proceedings of the IEEE Intelligent Transportation Systems Conference, Seattle, WA, USA, 30 September–3 October 2007; pp. 405–412.
16. Luvizon, D.C.; Nassu, B.T.; Minetto, R. Vehicle speed estimation by license plate detection and tracking. In Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2014; pp. 6563–6567.

17. Wu, J.; Liu, Z.; Li, J.; Gu, C.; Si, M.; Tan, F. An algorithm for automatic vehicle speed detection using video camera. In Proceedings of the International Conference on Computer Science Education, Nanning, China, 25–28 July 2009; pp. 193–196.
18. Wang, J.X. Research of vehicle speed detection algorithm in video surveillance. In Proceedings of the International Conference on Audio, Language and Image Processing, Shanghai, China, 11–12 July 2016; pp. 349–352.
19. Llorca, D.F.; Salinas, C.; Jimenez, M.; Parra, I.; Morcillo, A.G.; Izquierdo, R.; Lorenzo, J.; Sotelo, M.A. Two-camera based accurate vehicle speed measurement using average speed at a fixed point. In Proceedings of the IEEE International Conference on Intelligent Transportation Systems, Rio de Janeiro, Brazil, 1–4 November 2016; pp. 2533–2538.
20. Yang, T.; Li, Z.; Zhang, F.; Xie, B.; Li, J.; Liu, L. Panoramic uav surveillance and recycling system based on structure-free camera array. *IEEE Access* **2019**, *7*, 25763–25778. [CrossRef]
21. Kanistras, K.; Martins, G.; Rutherford, M.J.; Valavanis, K.P. A survey of unmanned aerial vehicles (uavs) for traffic monitoring. In Proceedings of the International Conference on Unmanned Aircraft Systems, Atlanta, GA, USA, 28–31 May 2013; pp. 221–234.
22. Yamazaki, F.; Liu, W.; Vu, T.T. Vehicle extraction and speed detection from digital aerial images. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, IGARSS, Boston, MA, USA, 7–11 July 2008; pp. 1334–1337.
23. Moranduzzo, T.; Melgani, F. Car speed estimation method for uav images. In Proceedings of the 2014 IEEE Geoscience and Remote Sensing Symposium, Quebec City, QC, Canada, 13–18 July 2014; pp. 4942–4945.
24. 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.* **2017**, *18*, 890–901. [CrossRef]
25. Bruin, A.D.; (Thinus) Booysen, M.J. Drone-based traffic flow estimation and tracking using computer vision. In Proceedings of the South African Transport Conference, Pretoria, South Africa, 6–9 July 2015.
26. Guido, G.; Gallelli, V.; Rogano, D.; Vitale, A. Evaluating the accuracy of vehicle tracking data obtained from unmanned aerial vehicles. *Int. J. Transp. Sci. Technol.* **2016**, *5*, 136–151. [CrossRef]
27. Liu, X.; Yang, T.; Li, J. Real-time ground vehicle detection in aerial infrared imagery based on convolutional neural network. *Electronics* **2018**, *7*, 78. [CrossRef]
28. Xin, Z.; Chang, Y.; Li, L.; Jianing, G. Algorithm of vehicle speed detection in unmanned aerial vehicle videos. In Proceedings of the International Conference on Wireless Communications, NETWORKING and Mobile Computing, Washington DC, USA, 12–16 January 2014; pp. 3375–3378.
29. Li, J.; Dai, Y.; Li, C.; Shu, J.; Li, D.; Yang, T.; Lu, Z. Visual detail augmented mapping for small aerial target detection. *Remote. Sens.* **2018**, *11*, 14. [CrossRef]
30. Shastry, A.C.; Schowengerdt, R.A. Airborne video registration and traffic-flow parameter estimation. *IEEE Trans. Intell. Transp. Syst.* **2005**, *6*, 391–405. [CrossRef]
31. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
32. Olivier, B.; Marc, V.D. Vibe: A universal background subtraction algorithm for video sequences. *IEEE Trans. Image Process.* **2011**, *20*, 1709–1724.
33. Li, J.; Zhang, F.; Wei, L.; Yang, T.; Lu, Z. Nighttime foreground pedestrian detection based on three-dimensional voxel surface model. *Sensors* **2017**, *17*, 2354. [CrossRef] [PubMed]
34. Tzutalin. Labelimg. Available online: https://github.com/tzutalin/labelImg (accessed on 2 March 2019).
35. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2015; pp. 91–99.
36. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In *Computer Vision—ECCV 2016*; Springer: Cham, Switzerland, 2016; pp. 21–37.
37. Hosang, J.; Benenson, R.; Schiele, B. Learning non-maximum suppression. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6469–6477.
38. 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.

39. Bae, S.H.; Yoon, K.J. Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1218–1225.
40. Bochinski, E.; Eiselein, V.; Sikora, T. High-speed tracking-by-detection without using image information. In Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance, Lecce, Italy, 29 August–1 September 2017.
41. Long, C.; Haizhou, A.; Zijie, Z.; Chong, S. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In Proceedings of the 2018 IEEE International Conference on Multimedia and Expo (ICME), San Diego, CA, USA, 23–27 July 2018.
42. Sorenson, H.W. *Kalman Filtering: Theory and Application*; The Institute of Electrical and Electronics Engineers, Inc.: Piscataway Township, NJ, USA, 1985.
43. Farneback, G. Two-frame motion estimation based on polynomial expansion. In Proceedings of the Scandinavian Conference on Image Analysis, Halmstad, Sweden, 29 June–2 July 2003; pp. 363–370.
44. Ua-detrac. Available online: http://detrac-db.rit.albany.edu/ (accessed on 5 February 2019).