International Journal of *Geo-Information*

MDPI

Article

Speed Estimation of Multiple Moving Objects from a Moving UAV Platform

Debojit Biswas [1], Hongbo Su [1,*], Chengyi Wang [2] and Aleksandar Stevanovic [1]

1 Department of Civil, Environmental and Geomatics Engineering, Florida Atlantic University, Boca Raton, FL 33431, USA; dbiswas2015@fau.edu (D.B.); astevano@fau.edu (A.S.)
2 Lab of Remote Sensing Image Processing, Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing 100101, China; wangcy@radi.ac.cn
* Correspondence: hongbo@ieee.org; Tel.: +1-561-2973936

Received: 14 March 2019; Accepted: 26 May 2019; Published: 31 May 2019

Abstract: Speed detection of a moving object using an optical camera has always been an important subject to study in computer vision. This is one of the key components to address in many application areas, such as transportation systems, military and naval applications, and robotics. In this study, we implemented a speed detection system for multiple moving objects on the ground from a moving platform in the air. A detect-and-track approach is used for primary tracking of the objects. Faster R-CNN (region-based convolutional neural network) is applied to detect the objects, and a discriminative correlation filter with CSRT (channel and spatial reliability tracking) is used for tracking. Feature-based image alignment (FBIA) is done for each frame to get the proper object location. In addition, SSIM (structural similarity index measurement) is performed to check how similar the current frame is with respect to the object detection frame. This measurement is necessary because the platform is moving, and new objects may be captured in a new frame. We achieved a speed accuracy of 96.80% with our framework with respect to the real speed of the objects.

Keywords: multiple object speed detection; faster R-CNN; discriminative correlation filter; feature based image alignment; structural similarity index measure

1. Introduction

Real-time traffic monitoring is a challenging task. Even with all the best technologies we have, we are still struggling with insufficient information on the road to solve traffic problems. Traffic monitoring systems have become more intelligent than ever before. Traffic signaling systems are adaptive, car counting is automated [1,2], and car density estimation on the road is also becoming automated [3,4]. Every day, new technologies are introduced into this field to make intelligent traffic monitoring systems (ITMS) better. One of the latest technological additions to this field is the use of UAVs (unmanned aerial vehicles). UAVs could be very effective to monitor traffic where traffic cameras are not available, especially for long driveways, forests, mountains and desert highways, and public events [5]. In addition, UAVs could be very effective in tracking and monitoring a vehicle for law enforcement and crime prevention. Static traffic cameras have a limited field of view, whereas drone cameras can overcome this limitation. High efficiency in collecting data in remote areas as well as for small and narrow lane data can be achieved by using a UAV platform. With this research, we are now more capable of collecting detailed vehicle-level data instead of only road and lane-level data. Vehicle-level data give us driving trajectories, which would be beneficial in monitoring driving behavior and patterns [6].

UAVs can provide a large field of view with more mobility and lower cost compared to traditional ground-based transportation sensors or low-angle cameras. UAVs can record and transmit data for investigation from the field directly but are rarely dedicated only to monitoring transportation and roadways. The short battery lives of the UAVs and a lack of efficient algorithms to detect and track moving vehicles [5] are the main factors preventing this technology from being widely used. It is extremely difficult to predict accurate motions when both the platform and target are moving at variable speeds. Since there are a total of six degrees of freedom for a UAV platform, compensating for these movements is challenging. Moreover, vibration and natural weather can cause negative effects during the detection and tracking processes.

In this study, we estimated the speed of multiple vehicles from a moving UAV platform using video streams of optical cameras. Detecting speeding vehicles or stopped vehicles in unsafe positions in real time will be a lifesaving solution. Locating trapped vehicles after natural disasters, such as snow, rain, flooding, and hurricanes, will be a potential application of this research. This study will be beneficial not only for civilian purposes but also for military purposes. In particular, tracking moving vehicles from a UAV platform will be extremely effective if the speed of the moving vehicles can be accurately estimated.

In this paper, the related work is introduced in Section 2. We describe the methodology in detail in Section 3 and explain how the framework is built. Section 4 documents the dataset information used for the experiments. The results of the framework and accuracy are discussed in Section 5. The conclusion and future scope of the study are discussed in Section 6.

2. Related Work

Vehicle detection and tracking is the building block of this study to detect the speed of the vehicles. It is very important to detect and track the vehicles precisely and accurately to get accurate speed detection results. Kanistras et al. [7] presented a survey on vehicle detection and tracking using UAVs. According to the research community, detection of vehicles can be characterized into two categories: (i) optical-flow and (ii) feature extraction and matching. Objects can be tracked and detected in contiguous frames by optical flow, but the movement is a conjugate of both camera and vehicular motion. Motions of the object and camera were separately calculated by a method designed by Rodríguez-Canosa et al. [8]. They developed a system to generate a vehicle's dynamic information, such as positions and velocities, by using consecutive video frames. They developed four major modules to generate the trajectory of a vehicle. These are feature extraction, image registration, vehicle shape detection, and vehicle tracking. Alpatov et al. [2] developed vehicle detection and counting systems considering road situation analysis tasks for traffic control and ensuring safety. A vehicle detection and counting algorithm, and a road marking detection algorithm were proposed. The algorithms were designed to process images obtained from a stationary camera. The vehicle detection and counting algorithm was implemented and also tested on an embedded platform of smart cameras. Eamthanakul et al. [4] designed a Traffic Congestion Investigating System by Image Processing from closed circuit television (CCTV) Camera to check a traffic condition from a traffic image on the road. Then, a user can check in advance for a traffic condition in a specific time. Seenouvong et al. [9] proposed a computer vision-based video detection method for vehicle detection and counting. The method found foreground objects using a background subtraction algorithm. Several computer vision techniques, such as thresholding, hole filling, and adaptive morphology operations, were applied to detect moving vehicles. Qin et al. [10] implemented a dynamic target detection method based on the fusion of optical flow and neural network. Xu et al. [11] used low-angle camera videos for detection and tracking and used a cellular neural network for background subtraction. Then, optical flow was used on the refined video. Elloumi et al. [12] proposed a UAV-based real-time monitoring (RTM) system with multiple UAVs to monitor traffic within city roads. To this end, the authors studied the impact of mobile UAV trajectories and the number of controlled vehicles on the event detection rate.

To make a detection, feature extraction methods for object detection extract features from a region of interest are used and then matched with feature banks (also called training features). Gleason et al. [13] applied multiple feature extraction with vehicle features, such as edge orientation, histogram of oriented gradients (HOG), and color, to increase accuracy. CNNs are very popular these days because of their accuracy. Current high-performance GPUs and unlimited storage make them successful. An OverFeat framework was used in [1] to detect and to count the cars. Through-traffic cameras, Single-shot detection, and MobileNet convolutional neural networks (CNN) were used to detect the front and back sides of cars in [3]. Leifloff et al. [14] used improved Haar-like features and line features of the vehicles to detect cars in city areas from high-resolution satellite images. Tuermere et al. [15] used a sophisticated blob detector, HOG, and Haar-like features to detect vehicles. Leitloff et al. [16] used extended Haar-like features and support vector machines (SVM) for vehicle detection. Hardjono et al. [17] proposed a solution to traffic monitoring and planning by using nonintrusive traffic sensors only, namely the virtual detection zone (VDZ) and closed circuit television (CCTV). When VDZ is combined with CCTV snap shots as a system, it can replace the traditional system based on inductive loop detector sensors.

Photo-based vehicle detection can obtain only static information, such as the positions of vehicles, the gap between two vehicles, and the lane position of the vehicles. However, video-based vehicle detection can provide dynamic information, including speed and acceleration of the vehicles. That is why video-based vehicle detection gives an advantage over photo-based detection. Cao et al. [18] used the Kanade–Lucas–Tomasi (KLT) feature-based framework for multi-motion layer analysis, which is effective and robust, in the detection and tracking of moving vehicles. A feature extraction matching method was used by numerous researchers for vehicle tracking. Lingua et al. [19] implemented an auto-adaptive version of the scale invariant feature transform (SIFT) used in photogrammetry.

Ibrahim et al. [20] implemented a speed detection system using video streams. They used an adaptive background subtraction technique for object detection. For object tracking, they monitored the object's entrance and exit of the scene. They counted the number of frames an object takes in the travel to measure the speed. Wu et al. [21] and Rad et al. [22] presented an automatic speed detection algorithm using video cameras. They first mapped the coordinates from the image domain to the real-world domain. The second part of their framework was vehicle detection using the segmentation and adaptive background subtraction algorithm. Finally, they calculated the speed of the vehicles by calculating the number of frames a vehicle takes to travel from one scene to another. Ranjit et al. [23] implemented a motion vector technique to determine the speed of the vehicles. Wang [24] presented a vehicle speed detection algorithm using video surveillance. In this approach, they used three frame differencing and background differencing to extract features from moving vehicles. Then, they implemented vehicle centroid feature extraction for tracking and positioning. Finally, the speed of the vehicle was detected using the differencing mapping algorithm between pixel distance and actual distance. Hua eta al. [25] implemented a tracking and detection based vehicle speed estimation system coupled with an optical-flow-based approach.

Image registration is another important factor in calculating the accurate movement of the vehicles. Zitova and Flusser [26] presented a review of image registration methods. They separated the registration methods according to the image registration procedure. Aicardi et al. [27] presented an image-based approach to co-register multitemporal UAV dataset. They implemented an automated method to register the images using anchor images. Sheng et al. [28] used hydrological images to implement an automatic way to register based on the extraction of lakes centroids from satellite images. Using the alignment of orthorectified data from optical satellite remote sensing images, Behling et al. [29] detected landslides.

Finally, the data acquisition plan is also a significant factor for this study. The acquisition of data from a UAV platform needs to consider several factors. These are flight planning strategies, ground control points, checkpoint distribution, and measurement, etc. Aicardi et al. [30] evaluated the possibility of acquiring and use of oblique images for the 3D reconstruction of a historical building, for the realization of the high-level-of-detail architectural survey, obtained by a UAV and traditional COTS (Commercial Off-the-Shelf) digital cameras. Jiang et al. [31] exploited the use of a UAV for outdoor data acquisition and conducts accuracy assessment tests to explore potential usage for offsite inspection of transmission lines. Vacca et al. [32] studied the accuracy of 3D building modeling based on both nadir and oblique UAV images. They demonstrated that the integration of oblique UAV images can substantially increase the achievable accuracy compared with traditional modeling using the terrestrial point cloud.

3. Methodology

Vehicle detection and tracking using a UAV platform are attracting numerous researchers. This type of research can open many new avenues beyond our imaginations. Still, some issues need to be addressed more carefully. First, (a) increasing the detection rate when vehicle size is very small needs attention. Faster region-based convolutional neural network (R-CNN) [33] addresses this problem, and that is why we have adopted this CNN algorithm in our study to solve detection problems. Another existing issue in this research area is (b) maintaining tracking for a longer time while both the platform and the object are moving. The final problem is (c) extracting the speed information of the vehicle. Solving the last problem is challenging because reference objects to measure the exact distance traveled by the vehicles are not always available. Furthermore, a UAV platform does not move with the same elevation or direction at all times. We handled this problem very tactically in our study. We created a database for different sizes of vehicles, and when a particular category of vehicle was detected, we used the vehicle as a reference object to measure the distance traveled. After that, we converted the distance information into speed with respect to time. In this section, we discussed Faster R-CNN as a vehicle detection algorithm, channel and spatial reliability tracking (CSRT) as a tracking algorithm, feature-based image alignment (FBIA) as an image alignment algorithm, and structural similarity index measurement (SSIM) as a similarity index measure algorithm.

3.1. Faster R-CNN

Faster R-CNN is used for vehicle detection in this research. The R-CNN family was originally introduced by Girshick et al. in 2013. After several iterations and improvements, Girshick published a second paper, Fast R-CNN, in 2015. A few months later, Girshick published the third paper of this series, Faster R-CNN [33]. The Faster R-CNN algorithm can be performed in three steps. Figure 1 explains the basic functionality of Faster R-CNN. It takes a raw image as input.

Step 1 performs feature extraction using pre-trained CNN. In Step 2, the extracted features are passed in parallel to two different components of the Faster R-CNN. (a) A region proposal network (RPN) determines the potential objectness, i.e., whether any object is present at a certain location. (b) A region of interest (ROI)—pooling layer, pools and scales down the image and passes proposed ROI bounding boxes to two different CNN networks. Step 3 determines the object type and the location of the object, using these two CNN networks.

Implementation of Faster R-CNN: Vehicles are labeled using the ‘Labeling’ [34] software from the training images. Against every labeled image, an annotation file is created. Tensorflow object detection (TFOD) API is used as a training platform. Three different kinds of record files are created that store the location and class information about the datasets.

The record directory stores these three files:

i. Training.record: The serialized image dataset of images, bounding boxes, and labels used for training.
ii. Testing.record: The images, bounding boxes, and labels used for testing.
iii. Classes.pbtxt: A plaintext file containing the names of the class labels and their unique integer IDs.

A special configuration file is used to instruct the TFOD API how to train the model. This file contains information about the location of training data, size of the images, and hyper-parameters, such as learning rate, number of iterations. After the training is done, frozen models (model files) are created with weights. The model is later used for vehicle detection purposes. The number of iterations performed is 50 k, and the learning rate 0.0001 is used for this experiment.

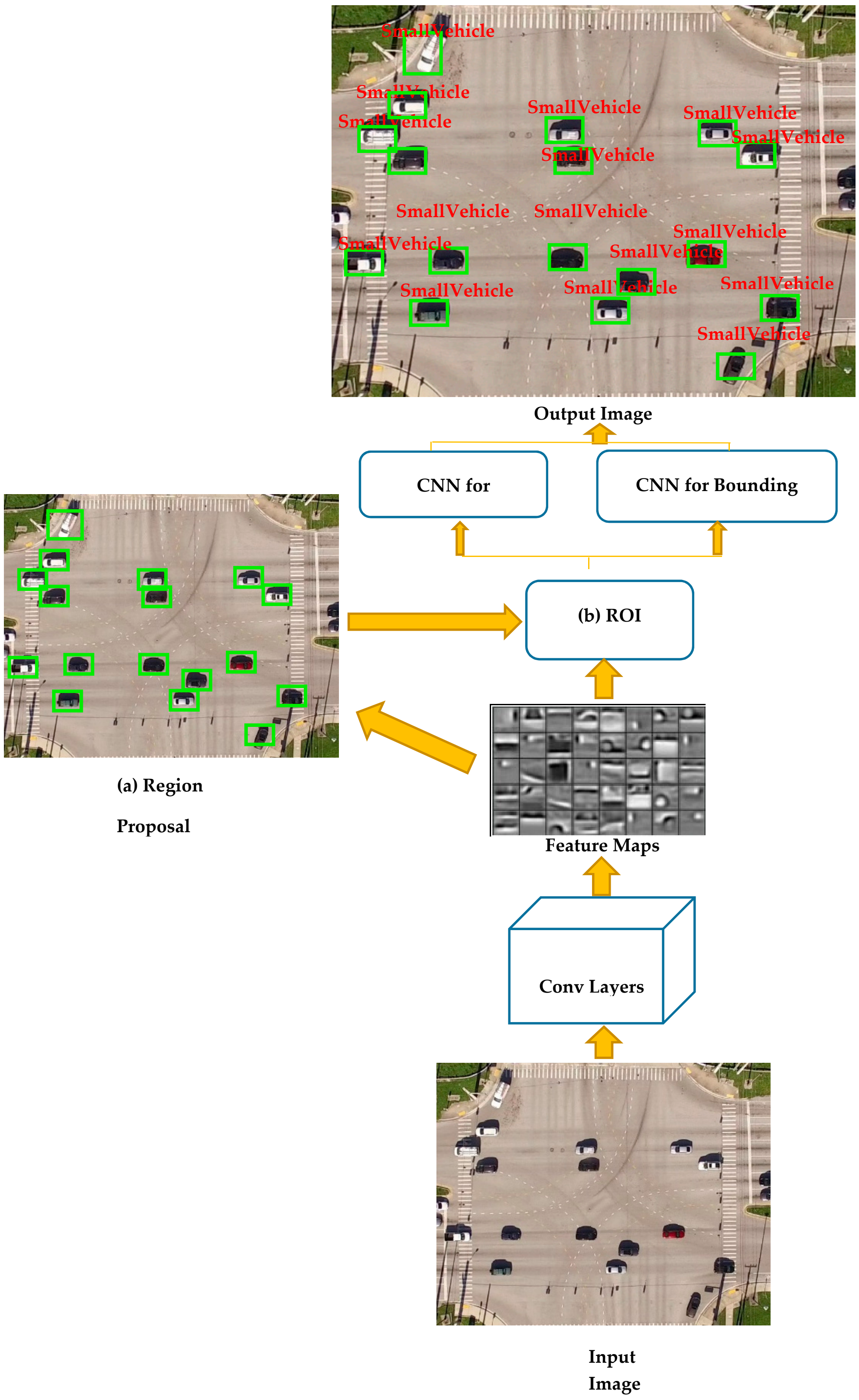


Figure 1. Faster region-based convolutional neural network (R-CNN) architecture.

3.2. *CSRT*

Object detection does not preserve the information about the detection of previous frames. In object tracking, the algorithm gathers lots of information from each object's previous location, such as direction and motion of the objects. Moreover, before tracking, detection is mandatory. From the detection, it is possible to know the appearance of the objects. A good tracking algorithm will use all the information it has to predict the next location of the objects. Several common tracking algorithms include (a) BOOSTING Tracker is slow and does not work well. (b) MIL Tracker gives better accuracy than BOOSTING Tracker but does a poor job reporting failure. (c) KCF Tracker is faster than BOOSTING and MIL Trackers but is not able to handle occlusion. (d) MedianFlow Tracker does a nice job of reporting failure but is unable to handle too fast-moving objects. (e) TLD Tracker does best under multiple occlusion conditions, but it has a higher false-positive rate. (f) MOSSE Tracker is very fast but not very accurate. We chose CSRT Tracker [35] for our experiment, as this algorithm gives good accuracy, and we were not concerned about occlusions.

CSRT uses a spatial reliability map to adjust filter support to the part of the selected region from the frame for tracking. CSRT uses only two standard features, HOGs and color names, to achieve correlation responses among channels.

Pseudocode for CSRT

```
Inputs:

Image, object position on the previous frame, filter, scale, color histogram, channel reliability.
Localization and scale estimation:

1. Create a new target position using the position of the maximum in the correlation between
   filter and image patch features extracted on the previous position and weighted by the channel
   reliability scores.
2. Estimate detection reliability using per-channel responses.
3. Use the new location to get a new scale.

Update:

1. Extract foreground and background histograms.
2. Update foreground and background histograms.
3. Estimate reliability map.
4. Estimate new filter.
5. Estimate learning channel reliability.
6. Calculate channel reliability.
7. Update filter.
8. Update channel reliability.

Repeat:

Perform (i) Localization and scale estimation and do (ii) Update for every new detection.
```

3.3. FBIA

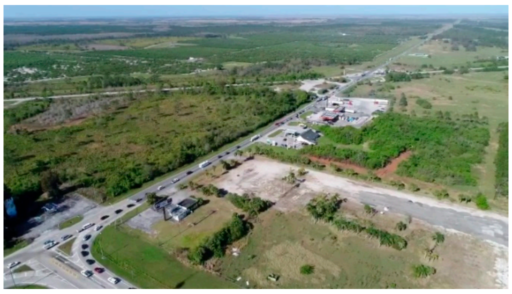
Image alignment is a necessary step for speed estimation. Image alignment is done for all the consecutive frames with respect to the detection frame to get the exact distance a car moves in a particular time. Image alignment, or image registration, is a technique to find out common features between two images where one image is lined up with respect to the other image.

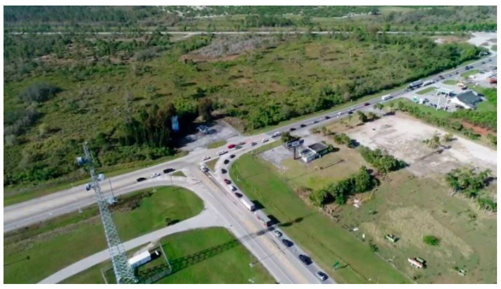
The core of the image alignment techniques consists of a 3×3 matrix. This core matrix is called homography. If (x_1, y_1) is a point at first frame and (x_2, y_2) is the same point at the second frame, then homography relates them in the following way:

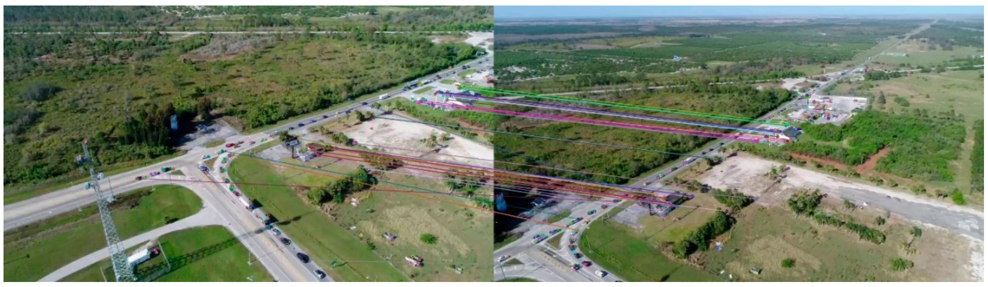
$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = H \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}. \quad (1)$$

We need four or more points to find the homography (H). The ORB feature detector is applied to find the same points for two comparable frames. ORB stands for "oriented FAST and rotated BRIEF". The FAST algorithm locates the coordinates that are stable under image transformation, such as translation, scale, and rotation. The BRIEF algorithm works as a descriptor that encodes the appearance of the points to find the features. The same descriptor is used for the same physical point in two images. Hamming distance is implemented to measure the similarity between two features, but it is not uncommon for 20% to 30% of the matches to be incorrect. A robust estimation technique called random sample consensus (RANSAC), which produces the right result even in the presence of a large number of bad matches, is applied to remove the outliers. After calculating the homography, the transformation is applied to all the pixels in one frame to map it to the other frame.

Figure 2 shows the result of a frame alignment. Figure 2a,b show a reference frame and a frame to align. A match result is shown using Hamming distance in Figure 2c, and finally, the aligned image is displayed in Figure 2d.


(**a**) Reference frame


(**b**) Frame to align


(**c**) Match result

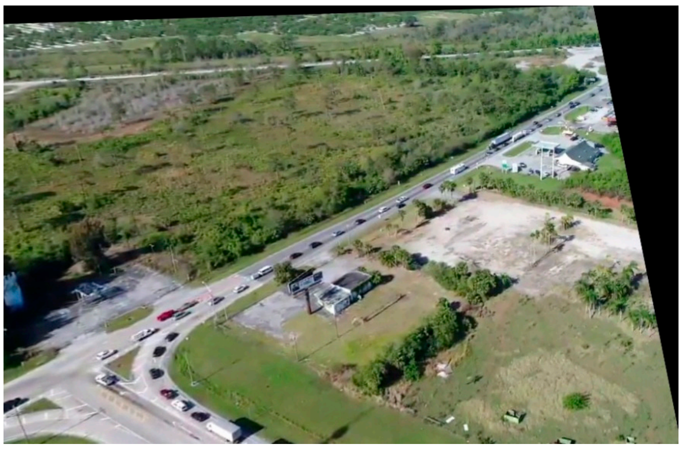

(**d**) Aligned frame

Figure 2. Feature-based image alignment result.

3.4. SSIM

Structural similarity index measure (SSIM) is used in this experiment to make sure that contiguous frames have enough similarity. The SSIM threshold for this study is set to 0.50. That means the detection frame and the current frame should have at least 0.50 SSIM. This index is a very important part of the experiment because SSIM is used to determine when the detection algorithm (Faster R-CNN) needs to perform based on its value. The tracking algorithm (CSRT) is performed when SSIM is greater than 0.50. In Figure 3, SSIM is 0.21, which means it is time to perform the detection algorithm. The optimal SSIM index is set to 0.50 after performing several experiments to make sure there is enough overlapping between the frames. At the same time, duplicated detection can be avoided. The detection algorithm is much slower with respect to the tracking algorithm, which turned out to be a bottle-neck to limit the real-time performance.

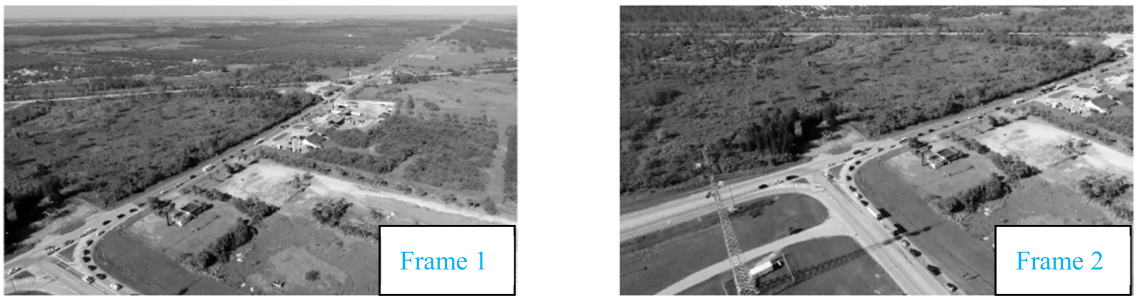


Figure 3. Structural similarity index measure (SSIM) result with a value of 0.21.

SSIM attempts to model the perceived change in the structural information of the image. Rather than comparing for the entire image, Equation (2) compares two windows (small sub-samples). In addition to the perceived changes, this approach accounts for changes in the structure of the images. The value of SSIM can vary between −1 and 1, where 1 indicates perfect similarity.

$$SSIM(x,y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)} \tag{2}$$

Equation (2) indicates the (x,y) location of a N X N window in each image, where μ indicates the mean of the pixels at x and y direction, σ_x and σ_y indicates the variance of x and y. σ_{xy} denotes the covariance of x and y. c_1 and c_2 are two stabilizers to act on weak denominator.

3.5. Speed Estimation

Two categories of vehicles are detected for this study. These are small vehicles and large vehicles. Sedan cars from the category of small vehicles and eighteen-wheel trucks from the category of large vehicles are considered as a reference object to measure the speed of the vehicles. The average length of the sedan cars are 186 inches [36], and the average length of the multi axel eighteen-wheel trucks are 840 inches [37]. The number of pixels is calculated against the length of the vehicle when a sedan car or an eighteen-wheel truck is detected in a frame. Then the traveled distance of a vehicle is calculated by measuring the centroid displacement of the vehicle from the reference frame to the current frame. Finally, the speed of the vehicles is measured by the traveled distance with respect to time (time is calculated from the frame rate).

Figure 4 explains the complete framework of the study. The object detection algorithm (Faster R-CNN) must be applied under the three conditions. (a) for the first frame, (b) frame number multiple of 100 or c) SSIM index is less than 0.5. Then FBIA is applied for image alignment, and CSRT is applied for object tracking when SSIM index is more than 0.5. The speeds of the vehicles are calculated from the gathered statistics during the tracking process. Image alignment (FBIA) process can distort the video frames; hence, applying inverse homography is important to get undistorted video frames with tracking information.

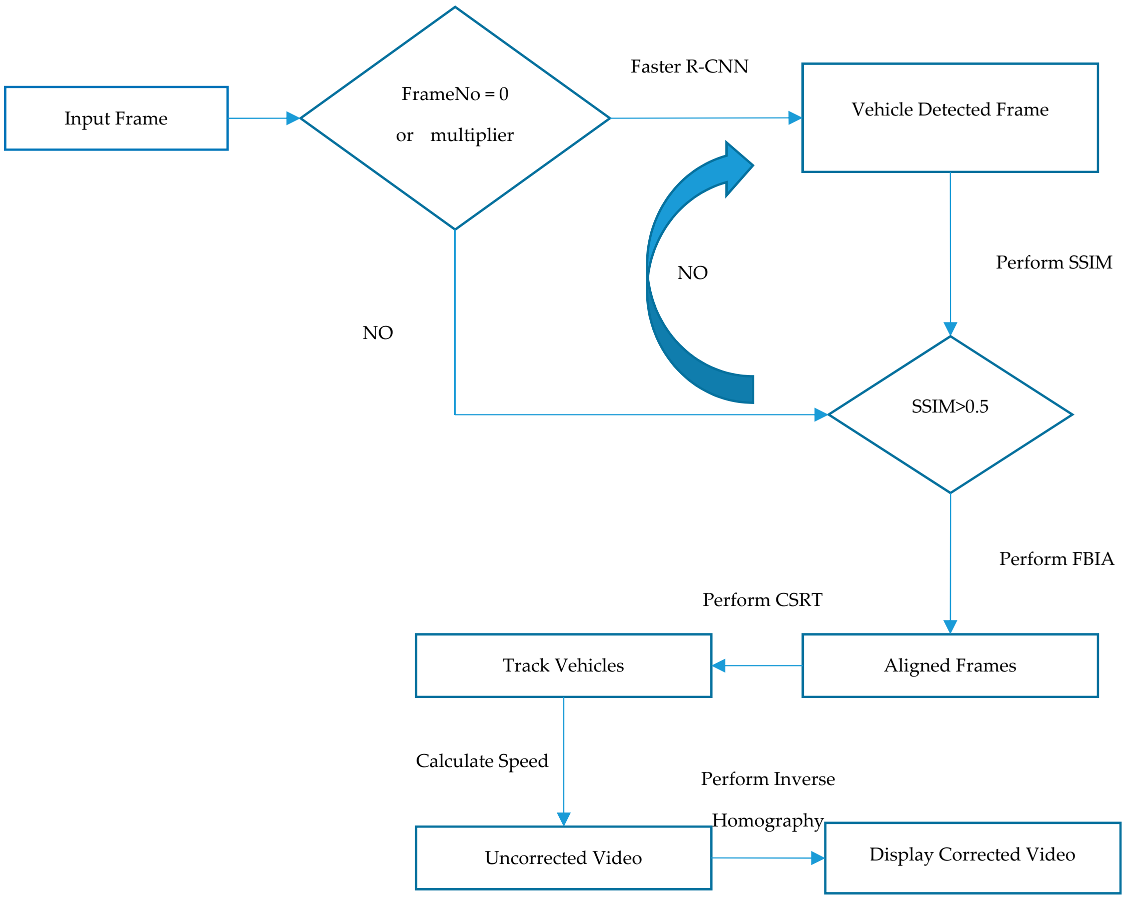


Figure 4. Complete flow chart of the framework.

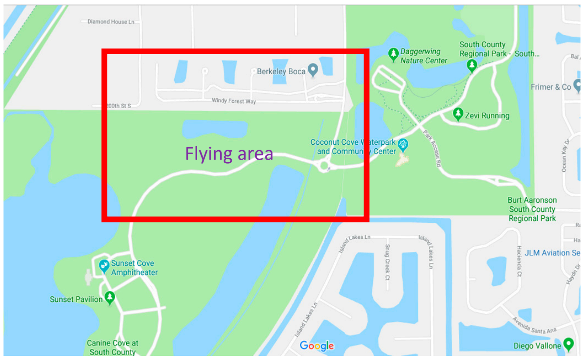
4. Datasets

An Inspire I V2.0 UAV with a ZENMUSE X3 optical camera is used for this study. The camera is equipped with 4 k video, 16 MP, 3-axis gimbal. Two types of videos are captured for the experiment: while the drone is *static* and while the drone is *moving*. The videos are collected from three different locations for this study: (a) the intersection of Glades Road and State Road 441 in Boca Raton, FL (Static video), (b) Burt Aaronson South County Regional Park, Boca Raton, FL (static and moving), and (c) Bluegrass Yeehaw junction, Okeechobee, FL (moving). Figure 5 shows their locations in Google Maps.

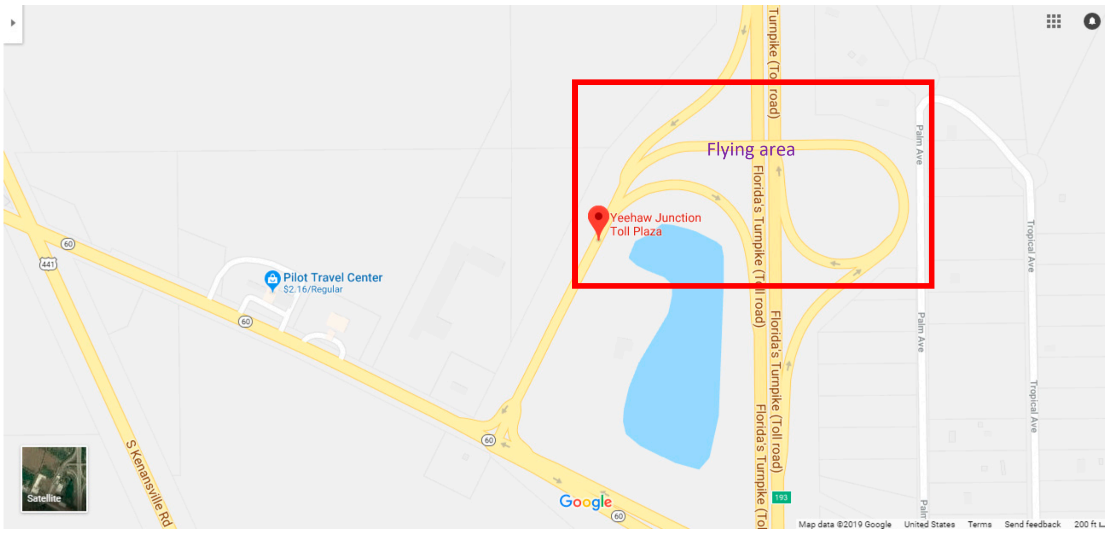
From these videos, two classes are prepared for this study, "SmallVehicle" and "LargeVehicle". All the small cars, sedan, SUVs, and small trucks are labeled as "SmallVehicles". The big vans, buses, and eighteen-wheel trucks are labeled as "LargeVehicle" A total of 17,304 small vehicles and 784 large vehicles are used as training and validation for Faster R-CNN. Among these data, 70% is used for training, 26% is used for validation, and 4% is used for manual testing. Open source software called "Labelimg" [34] is used for image labeling. The images are collected in different light conditions.

Flying area
Glades Rd
US 441 S
Best Buy
Home Depot
Walgreens
Starbucks
Google

(**a**) Glades Road and 441 highway intersection in Boca Raton, FL


Flying area
Berkeley Boca
Coconut Cove Waterpark and Community Center
Daggerwing Nature Center
South County Regional Park
Burt Aaronson South County Regional Park
Sunset Cove Amphitheater
Sunset Pavilion
Google

(**b**) Burt Aaronson South County Regional Park, Boca Raton, FL


Flying area
Yeehaw Junction Toll Plaza
Florida's Turnpike (Toll road)
Palm Ave
Tropical Ave
Pilot Travel Center
$2.16/Regular
S Kenansville Rd
Satellite
Google
Map data ©2019 Google United States Terms Send feedback 200 ft

(**c**) Bluegrass Yeehaw junction, Okeechobee, FL

Figure 5. Google Maps data collection points.

5. Results and Discussion

The results of this study are presented in two tables below. Table 1 shows the detection results, and Table 2 represents the speed estimation results. Table 1 compares manual detection and Faster R-CNN detection. We collected sample frames from all three video capture locations. Glades Road is a city road, so we obtained fewer sample data for large vehicles at that location. Burt Aaronson is a public park, so big trucks are not allowed there. We obtained some good sample data for large vehicles on the highway turnpike at Bluegrass Yeehaw junction. We achieved 90.77% average accuracy for small vehicles and 96.15% average accuracy for large vehicle detection. F score is also calculated following the equation $\left(\frac{recall^{-1}+precision^{-1}}{2}\right)^{-1}$. F score for large vehicle is 98.04% and for small vehicle, it is 95.16%, since there is no misclassification among the two classes although we found a number of missing detections.

Table 2 reports the speed accuracy of the framework. In the static video of Glades Road, we observed 43 vehicles with an average speed of 0 mph, 10 vehicles with an average speed of 30 mph, and 15 vehicles with an average speed of 25 mph. Our framework gives 100% accuracy for 0 mph vehicles, 98.33% for 30 mph, and 96% for 25 mph. The average RMSE is 0.564 for our case studies. Figure 6a shows speed estimation results for the Glades Road video. The Bluegrass video was recorded from a moving platform; we obtained a 100% accuracy for stopped vehicles and 95.83% accuracy for moving vehicles. Figure 6b shows speed estimation results for the Bluegrass video. At Burt Aaronson, we achieved 100% accuracy for stopped vehicles from a moving UAV and 96% accuracy for moving vehicles.

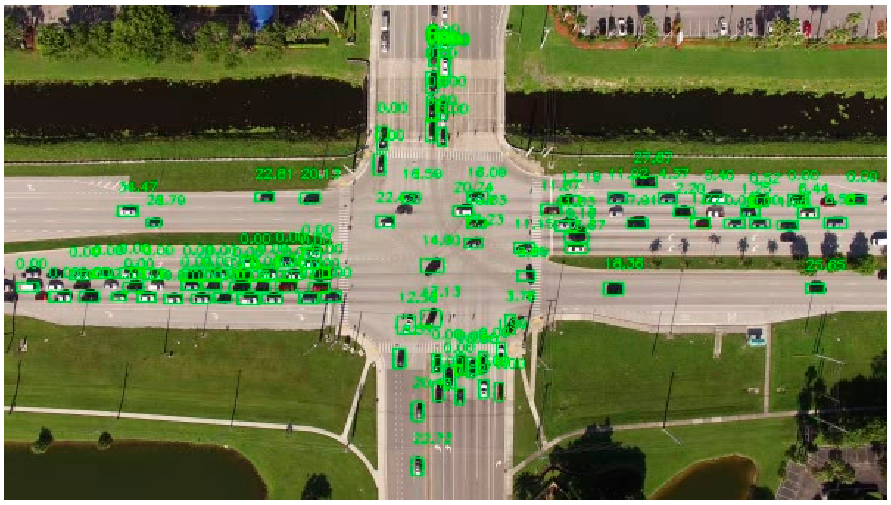
Table 1. Vehicle detection results.

	Manual Count		**Faster R-CNN Count**		**Accuracy**	
	Small Vehicle	**Large Vehicle**	**Small Vehicle**	**Large Vehicle**	**Small Vehicle**	**Large Vehicle**
Glades Road/441 Intersection	506	2	437	2	86.37%	100%
Bluegrass	38	13	35	12	92.10%	92.30%
Burt Aaronson (Static)	19	0	19	0	100%	-
Burt Aaronson (moving)	39	0	33	0	84.61%	-
Total average					90.77%	96.15%

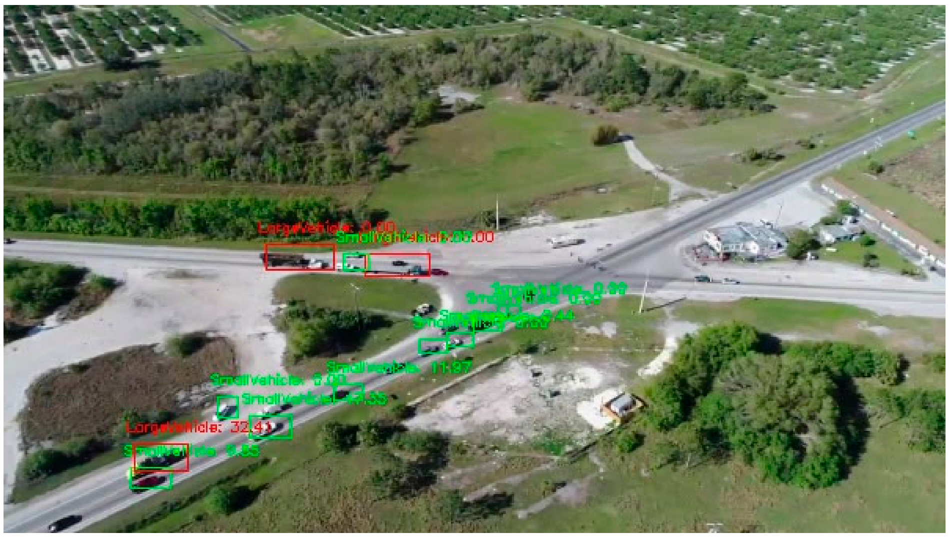
To measure the accuracy at the Glades Road and Bluegrass locations, we picked out two spots on the map and measured the distance using Google Maps. Then we counted the time each car took to cover the distance to calculate the speed of the cars. At Burt Aaronson Park, we drove two cars and recorded the speedometer readings. Later in the lab, we matched the speedometer recording video with UAV recording and compared the speedometer reading to our framework outputs. In Figure 6c, we can see that the right-side speedometer reads 21 mph, and the framework also shows 21 mph. The left-side speedometer displays 23 mph while the framework shows 24 mph.

Table 2. Speed detection results.

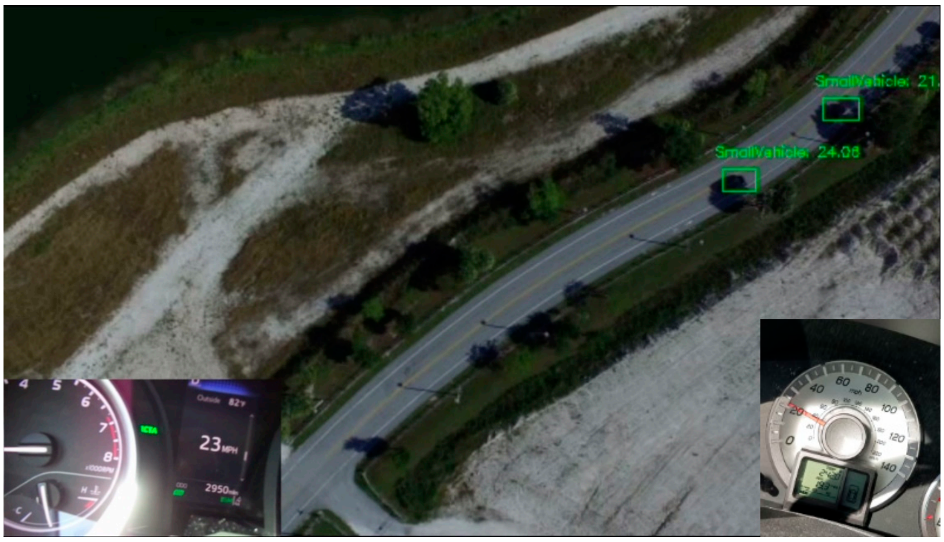
	Number of Cars Observed	**Average Speed Manually Observed (mph)**	**Average Speed from the Framework (mph)**	**Accuracy %**	**RMSE (mph)**
Glades Road/441 Intersection (Static drone)	43	0	0	100%	0
Glades Road/441 Intersection (Static drone)	10	30	30.5	98.33%	1.194
Glades Road/441 Intersection (Static drone)	15	25	26	96%	1.111
Bluegrass (Moving drone)	15	0	0	100%	0
Bluegrass (Moving drone)	5	12	11.5	95.83%	0.604
Burt Aaronson (Moving drone)	19	0	100%	100%	0
Burt Aaronson (Moving drone)	2	25	26	96%	1.044
Average				96.80%	0.564



(**a**) Speed detection at Glades and 441 intersection



(**b**) Speed detection at Bluegrass Yeehaw junction



(**c**) Burt Aaronson South County Regional Park

Figure 6. Speed estimation results.

6. Conclusions and Future Work

This study is a stepping stone to solve a long-existing problem. In this study, we achieved 100% speed accuracy while vehicles are non-moving. In addition, we achieved over 96% speed accuracy from a static platform (Table 2). The most challenging problem to solve was to estimate the speed when both the platform and target were moving. We accomplished a speed accuracy of over 95% in this scenario. We observed that when the number of vehicles is lowered, the frame per second rate (fps) of processing is higher, but as vehicles increase, the fps of processing reduces. The tracking algorithm takes more resources for more vehicles in the frames. We did not perform any comparative study to check how fps performs with respect to the number of vehicles.

Since UAV technologies, computer vision, and photogrammetry algorithms are under active development, we can skip the image alignment steps (FBIA) if the geolocation of frames from the UAV videos can be accurately determined. We would not need to perform FBIA to estimate the speed. Furthermore, we could handle the UAV elevation and angle fluctuation better with georeferenced videos, as image alignment alone cannot handle elevation or angle fluctuation well. Implementing the framework under different weather conditions and different light conditions is a future direction to pursue as well.

Author Contributions: Debojit Biswas and Hongbo Su conceived and designed the experiments; Debojit Biswas performed the experiments and analyzed the data; Debojit Biswas and Hongbo Su wrote the paper; Aleksandar Stevanovic and Chengyi Wang revised the paper.

Funding: This research was funded by Guangxi innovation driven development special projects on "Product development and applications of nearshore dual frequency LiDAR detector" and a Florida Space Research Grant (No. NNX15_002). The APC was funded by Guangxi innovation driven development special projects on "Product development and applications of nearshore dual frequency LiDAR detector".

Acknowledgments: The authors are thankful to FDOT staff for their help with acquiring the data under the project BDV27 TWO 977-12—"Development of a Traffic Map Evaluation Tool for Traffic Management Center Applications". It should be noted that the opinions, findings, and conclusions expressed in this publication are those of the authors and not necessarily those of the Florida Department of Transportation or the U.S. Department of Transportation.

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

References

1. Biswas, D.; Su, H.; Wang, C.; Blankenship, J.; Stevanovic, A. An automatic car counting system using OverFeat framework. *Sensors* **2017**, *17*, 1535. [CrossRef] [PubMed]
2. Alpatov, B.A.; Babayan, P.V.; Ershov, M.D. Vehicle detection and counting system for real-time traffic surveillance. In Proceedings of the 7th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 10–14 June 2018. [CrossRef]
3. Biswas, D.; Su, H.; Wang, C.; Stevanovic, A.; Wang, W. An automatic traffic density estimation using Single Shot Detection (SSD) and MobileNet-SSD. *Phys. Chem. Earth Parts A/B/C* **2019**, *110*, 176–184. [CrossRef]
4. Eamthanakul, B.; Ketcham, M.; Chumuang, N. The Traffic Congestion Investigating System by Image Processing from CCTV Camera. In Proceedings of the International Conference on Digital Arts, Media and Technology (ICDAMT), Chiangmai, Thailand, 1–4 May 2017. [CrossRef]
5. Wang, L.; Chen, F.; Yin, H. Detecting and tracking vehicles in traffic by unmanned aerial vehicles. *Autom. Constr.* **2016**, *72*, 294–308. [CrossRef]
6. Toledo, T. Driving behavior: Models and challenges. *Transp. Rev.* **2007**, *27*, 65–84. [CrossRef]
7. 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 (ICUAS), Atlanta, GA, USA, 28–31 May 2013; pp. 221–234.
8. Rodriguez-Canosa, G.R.; Thomas, S.; del Cerro, J.; Barrientos, A.; MacDonald, B. A realtime method to detect and track moving objects (DATMO) from unmanned aerial vehicles (UAVs) using a single camera. *Remote Sens.* **2012**, *4*, 1090–1111. [CrossRef]

9. Seenouvong, N.; Watchareeruetai, U.; Nuthong, C.; Khongsomboon, K.; Ohnishi, N. A computer vision based vehicle detection and counting system. In Proceedings of the 8th International Conference on Knowledge and Smart Technology (KST), Chiangmai, Thailand, 3–6 February 2016. [CrossRef]
10. Qin, H.; Zhen, Z.; Ma, K. Moving object detection based on optical flow and neural network fusion. *Int. J. Intell. Comput. Cybern.* **2016**, *9*, 325–335. [CrossRef]
11. Xu, J.; Wang, G.; Sun, F. A novel method for detecting and tracking vehicles in traffic image sequence. In Proceedings of the Volume SPIE 8878, Fifth International Conference on Digital Image Processing, Beijing, China, 19 July 2013; p. 88782P.
12. Elloumi, M.; Dhaou, R.; Escrig, B.; Idoudi, H.; Saidane, L.A. Monitoring Road Traffic with a UAV-based System. In Proceedings of the IEEE Wireless Communications and Networking Conference, Barcelona, Spain, 15–18 April 2018.
13. Gleason, J.; Nefian, A.V.; Bouyssounousse, X.; Fong, T.; Bebis, G. Vehicle Detection from Aerial Imagery. In Proceedings of the IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 2065–2070.
14. Leitloff, J.; Hinz, S.; Stilla, U. Vehicle detection in very high-resolution satellite images of city areas. *IEEE Trans. Geosci. Remote Sens.* **2010**, *48*, 2795–2806. [CrossRef]
15. Tuermer, S.; Kurz, F.; Reinartz, P.; Stilla, U. Airborne vehicle detection in dense urban areas using HoG features and disparity maps. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2013**, *6*, 2327–2337. [CrossRef]
16. Leitloff, J.; Rosenbaum, D.; Kurz, F.; Meynberg, O.; Reinartz, P. An operational system for estimating road traffic information from aerial images. *Remote Sens.* **2014**, *6*, 11315–11341. [CrossRef]
17. Hardjono, B.; Tjahyadi, H.; Widjaja, E.A.; Rhizma, M.G.A. Vehicle travel distance and time prediction using virtual detection zone and CCTV data. In Proceedings of the IEEE 17th International Conference on Communication Technology (ICCT), Chengdu, China, 27–30 October 2017. [CrossRef]
18. Cao, X.; Lan, J.; Yan, P.; Li, X. Vehicle detection and tracking in airborne videos by multi-motion layer analysis. *Mach. Vis. Appl.* **2011**, *23*, 921–935. [CrossRef]
19. Lingua, A.; Marenchino, D.; Nex, F. Performance analysis of the SIFT operator for automatic feature extraction and matching in photogrammetric applications. *Sensors* **2009**, *9*, 3745–3766. [CrossRef] [PubMed]
20. Ibrahim, O.; ElGendy, H.; ElShafee, A.M. Speed Detection Camera System using Image Processing Techniques on Video Streams. *Int. J. Comput. Electr. Eng.* **2011**, *3*, 6. [CrossRef]
21. 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.
22. Rad, A.G.; Dehghani, A.; Karim, M.R. Vehicle speed detection in video image sequences using CVS method. *Int. J. Phys. Sci.* **2010**, *5*, 2555–2563.
23. Ranjit, S.S.S.; Anas, S.A.; Subramaniam, S.K.; Lim, K.C.; Fayeez, A.F.I.; Amirah, A.R. Real-Time Vehicle Speed Detection Algorithm usingMotion Vector Technique. In Proceedings of the International Conference on Advances in Electrical & Electronics, NCR, India, 28–29 December 2012.
24. Wang, J.X. Research of vehicle speed detection algorithm in video surveillance. In Proceedings of the International Conference on Audio, Language and Image Processing (ICALIP), Shanghai, China, 11–12 July 2016; pp. 349–352.
25. Hua, S.; Kapoor, M.; Anastasiu, D.C. Vehicle Tracking and Speed Estimation from Traffic Videos. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, USA, 18–22 June 2018.
26. Zitová, B.; Flusser, J. Image registration methods: A survey. *Image Vis. Comput.* **2003**, *21*, 977–1000. [CrossRef]
27. Aicardi, I.; Nex, F.; Gerke, M.; Lingua, A.M. An Image-Based Approach for the Co-Registration of Multi-Temporal UAV Image Datasets. *Remote Sens.* **2016**, *8*, 779. [CrossRef]
28. Sheng, Y.; Shah, C.A.; Smith, L.C. Automated image registration for hydrologic change detection in the lake-rich arctic. *IEEE Geosci. Remote Sens. Lett.* **2008**, *5*, 414–418. [CrossRef]
29. Behling, R.; Roessner, S.; Segl, K.; Kleinschmit, B.; Kaufmann, H. Robust automated image co-registration of optical multi-sensor time series data: Database generation for multi-temporal landslide detection. *Remote Sens.* **2014**, *6*, 2572–2600. [CrossRef]

30. Aicardi, I.; Chiabrando, F.; Grasso, N.; Lingua, A.M.; Noardo, F.; Spanò, A. UAV Photogrammetry with Oblique Images: First Analysis on Data Acquisition and Processing. In Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLI-B1, 2016 XXIII ISPRS Congress, Prague, Czech Republic, 12–19 July 2016.
31. Jiang, S.; Jiang, W.; Huang, W.; Yang, L. UAV-Based Oblique Photogrammetry for Outdoor Data Acquisition and Offsite Visual Inspection of Transmission Line. *Remote Sens.* **2017**, *9*, 278. [CrossRef]
32. Vacca, G.; Dessì, A.; Sacco, A. The Use of Nadir and Oblique UAV Images for Building Knowledge. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 393. [CrossRef]
33. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *NIPS* **2015**, arXiv:1506.01497. [CrossRef]
34. LabelImg. Available online: https://github.com/tzutalin/labelImg (accessed on 7 January 2019).
35. Lukezic, A.; Vojir, T.; Zajc, L.C.; Matas, J.; Kristan, M. Discriminative Correlation Filter Tracker with Channel and Spatial Reliability. *Int. J. Comput. Vis.* **2018**, *126*, 671–688. [CrossRef]
36. Creditdonkey. Available online: https://www.creditdonkey.com/average-weight-car.html (accessed on 7 January 2019).
37. Truckersreport. Available online: https://truckersreport.wordpress.com/2013/09/09/20-insane-but-true-things-about-18-wheelers/ (accessed on 7 January 2019).