A Novel Framework for Real-time Traffic Flow Parameter Estimation from Aerial Videos

Ruimin Ke

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Civil Engineering

University of Washington
2016

Committee:
Yinhai Wang
Don MacKenzie
Edward McCormack

Program Authorized to Offer Degree:
Department of Civil and Environmental Engineering

University of Washington

Abstract

A Novel Framework for Real-time Traffic Flow Parameter Estimation from Aerial Videos

Ruimin Ke

Chair of Supervisory Committee:
Dr. Yinhai Wang
Department of Civil and Environmental Engineering

Unmanned aerial vehicles (UAVs) are gaining popularity in traffic monitoring due to their low cost, high flexibility, and wide view range. Traffic flow parameters such as speed, density, and volume extracted from UAV-based traffic video are critical for traffic state estimation and traffic control, and has recently received more and more attention from researchers. However, different from stationary surveillance videos, the camera platforms move with UAVs and the motion in aerial videos makes it very challenging to process for data extraction.

To address this problem, a novel framework composed of two complementary approaches for real-time traffic flow parameter estimation from aerial videos is proposed. The first approach is a motion-based approach, which identifies traffic streams and video background based on their motions using Kanade-Lucas-Tomasi (KLT) tracker and k-means clustering algorithm, and then extracts traffic flow

parameters (speed, density, and volume) using connected graph and traffic flow theory. The second approach is a detection-based approach, which requires a vehicle detector training process. In this approach, vehicles from a top-view perspective are detected by the vehicle detector first and then the vehicle motion is estimated using KLT tracker as well as the background motion. Specifically, the vehicle detector is a combined cascaded classifier composed of Haar-like features and neural networks, making use of the fast processing speed of cascaded Haar classifier and the high detection rate of neural network. These two complementary approaches have their own advantages and together form the proposed framework for aerial video-based traffic flow parameter estimation.

The system was tested on multiple aerial videos taken by UAVs operated in various scenarios including uncongested traffic condition, uncongested traffic condition, daytime, nighttime, UAV moving and UAV hovering. The experimental results show that the system is able to extract traffic flow speed, density and volume, and also achieves high performance in both traffic speed and vehicle count estimation in various challenging scenarios. The proposed system achieves a fast processing speed that enables real-time traffic information estimation.

Table of Contents

List of Figures

List of Tables

ACKNOWLEDGEMENTS

I owe a great many thanks to a great many people who helped and supported me during the writing of this master thesis. I would never have the thesis completed without the love, support and help from these people. At this moment, towards the end of my master's journey, I would like to use the opportunity the express my sincere acknowledgement to all those who have helped me in both my work and life.

First of all, a deep sense of gratitude to my graduate adviser, Professor Yinhai Wang for his profound guidance and advice on both my research and life. I really appreciate the great research opportunities he provided to me, which were cutting-edge and challenging. They have made me very excited and encouraged in exploring the world of intelligent transportation. In addition, Professor Wang himself is a great example to me that every word and action of him tells me the way to be an excellent researcher, professor as well as a successful person in general.

I am sincerely grateful to Professor Don MacKenzie and Professor Edward McCormack for serving on my thesis committee. They gave me insightful and valuable advice on improving my thesis work. Their assistance and suggestions mean a lot to me. Additionally, I greatly appreciate the partial financial support from National Natural Science Foundation of China (Project 51138003, 51329801), and also would like to thank Beihang University, China for providing the aerial video datasets.

I would like to express my gratitude to my fellows. I would like to thank Dr. Zhibin Li for giving me valuable advice on this research as well as how to write a quality thesis. I would like to say "thanks" to Kristian Henrickson and John Ash for their assistance in both my research work and English writing skills. I would also like to thank Sung Kim and Professor Ali Farhadi for their help in developing the

preliminary framework of this research in the computer vision class (EE 576) I took. I would like to thank Zhiyong Cui and Xinqiang Chen for assisting in the experimental work of the research. A special thanks to Dr. Yanping Liang, whose current research direction is also about aerial video processing, for all the constructive discussions we have had. I would also like to express my warm thanks to Dr. Yingying Zhang, Dr. Xiaolei Ma, Dr. Cathy Liu, Dr. Yajie Zou, Dr. Ziqiang Zeng, Dr. Wenhui Zhang, Jinjun Tang, Han Jiang, Wenbo Zhu, Ziyuan Pu, Matt Dunlap, Tao Zhu, Xianzhe Chen, Meng Xia, Sishuang Wang, Yifan Zhuang for all the support and help and best wishes to all of you on your future career.

I would also like to express my hearty gratitude to my family and all my friends for their support, trust, and understanding. A particular thanks to my girlfriend Xuan Zhou, for all her love and support. With her company, my two-year master study has been full of surprise, happiness, and encouragement. She is pursuing her Ph.D. degree in Tsinghua University and currently a visiting student in Duke University. She is so smart and considerate that she supports me constantly not only in work and life but also in spirit. My communication with her cover a wide range of areas including but not limited to friends, cooking, sports, and cutting-edge research, which are always enjoyable and inspiring. Finally, but most importantly, my deepest thanks to my parents, who bring me to the fascinating world and are always there supporting me. Thank you both for your endless love and giving me strength to chase my dreams.

I dedicate this thesis to all the people who I love and love me. It is a new beginning of my journey.

Chapter 1 Introduction

1.1 General Background

Unmanned aerial vehicles (UAVs) have been considered a novel traffic monitoring technology used to collect information about traffic conditions on roads (see different UAVs in Figure 1-1). Compared to traditional traffic monitoring methods and devices, roadway monitoring with unmanned aerial vehicles has several advantages. First, traditional monitoring devices such as loop detectors and surveillance video cameras are usually placed at fixed locations to achieve a fixed surveillance coverage range; this may not be cost effective because a large number of these devices are needed for a single road segment [1-2]. Additionally, the maintenance of any of the fixed detectors leads to additional fees and would inevitably interrupt the normal traffic. In contrast, the UAV is a cost-effective platform that can both monitor a large continuous stretch of roadway and focus on a specific road segment. Also, maintenance on UAVs can be conducted off site and would this not lead to congestion on the roadway. The UAV also has another advantage in being able to provide rapid assessment and reconnaissance of an incident site for emergency response where no traditional sensors are installed and even in locations that humans may have difficulty accessing [3-4]. Further, by achieving a top-view perspective, the aerial videos have the potential to be used to provide fast and accurate estimations of traffic information in multiple travel directions at the same time. For the aforementioned reasons, a UAV equipped with a camera is considered to be a low-cost and flexible platform that can provide for efficient data acquisition [4-6]. Although the privacy issue and short battery life appear to be the two main concerns limiting practical use currently, it is widely believed that the UAV will achieve broader use in the near future once these concerns are addressed.

Due to the advantages of UAVs, aerial video-based traffic surveillance has become an active study topic in the transportation engineering field. Relevant research was initially conducted by State Departments of Transportation in Ohio, Florida, Georgia, and California [6]. Several papers were published by researchers participating in those projects that laid the foundation for UAV studies [4, 7, 8]. Although not placing tremendous weight on technical details, these papers present the "big picture" issues associated with UAVs and provide researchers with guiding information for future research.

Figure 1-1 Examples of different types of unmanned aerial vehicles

1.2 Research Objectives

Inspired by the increasing needs in exploring new traffic monitoring means and all the advantages of UAV, this paper aims at extracting traffic flow parameters using UAV-mounted video cameras. Specifically, our research objectives are stated as follows:

- Identify and distinguish multi-directional traffic streams on roadway
- Extract instantaneous speed and vehicle count for each traffic stream in each frame
- Convert instantaneous speed and vehicle count into aggregated speed, density and volume for a given aerial video
- Build an efficient and accurate top-view vehicle detector

- Develop a framework for aerial video-based traffic flow parameter estimation working in different traffic scenarios
- Achieve a real-time processing speed in traffic flow parameter estimation

1.3 Problem Statement

In order to extract the basic traffic flow parameters (i.e., speed, density and volume) in aerial videos, besides traditional challenges in surveillance video based detection such as the challenges associated with occlusion, shadows, and reflections, the most challenging issue is that both video background and foreground are moving due to the motion of the camera. Thus, several traditional image-based traffic information extraction technologies for fixed camera videos such as background subtraction and blob detection do not work well for UAV-based videos. Over the past two decades, some studies have focused on extracting traffic information from aerial videos; most of this work focused on applying traditional image processing techniques such as image registration to detect and track vehicles in aerial videos [7-11]. Once vehicles are able to be properly detected and tracked, specific traffic information can then be extracted from video. Such methods require large computing workloads resulting in slow processing speeds. Moreover, consistent UAV motion makes it impossible to conduct camera calibration since the external parameters of the camera is keep changing. Thus, how to handle UAV motion properly and even make use of UAV motion in the parameter estimation process becomes the main problem in this research.

1.4 Idea Generation

To address the motion-related problems, there are basically two approaches. The first is making use of the motion instead of converting the moving background into static background like applying image registration in most previous studies. Given the intuition that no matter how a UAV moves, the motions of vehicles in the traffic streams are different from

the motion of the background, thus, extracting low-cost features that can represent the motions of the video background and moving traffic comes to mind. The second approach is ignoring the moving background on the first stage, instead, just detecting the vehicles in each frame. With the vehicles detected, you will get all the detected vehicles (i.e., the rectangles that represent the detected vehicles) as the region of interest (ROI), and then all the areas outside the ROI can be seen as background. Thus, the vehicle motion can be calculated from the average ROI motion between consecutive frames; the average motion outside the ROI can be seen as the background motion.

Feature-based traffic detection and tracking methods have been frequently used in previous studies, however, elegantly selecting "good" features for further processing is crucial to the overall effectiveness [35, 36]. In general, there are three groups of features commonly used in video-based traffic detection: region-based features, point-and-patch-based features, and contour-based features [36, 37]. Region-based features are mainly the foreground silhouettes extracted by foreground extraction methods, which as aforementioned, is not an easy task due to the irregular background movement [36]. Contour-based feature extraction is based on edge detection, however, edge detection is not appropriate for UAV-based vehicle detection in most cases since vehicles are small in aerial videos [36, 37]. In point-and-patch-based feature detection, patch-based features such as Scale-invariant feature transform (SIFT) features [39] are computationally expensive to compute, thus real-time performance is less likely to be achieved. In contrast, point-based features are generally less computationally expensive [37]. Since Kanade-Lucas-Tomasi (KLT) optical flow tracker is very powerful in motion analysis, Shi-Tomasi interest points were a natural selection as the ultimate features to be used in this work. Specifically, in both of our motion-based approach and detection-based approach, the KLT tracker plays an important role, supporting effectiveness in flow parameters estimation and real-time performance.

1.5 Scope of Study

In this work, we proposed a new framework that composed of a motion-based approach and a detection-based approach to achieve a fast and accurate detection of traffic flow parameter, i.e., speed, density, and volume. Specifically, the motion-based approach includes four components, which are interest point tracking, motion-vector clustering, connected graph-based vehicle detection and counting, and traffic flow parameter estimation. In the detection-based approach, there are two main components. The first is a detection process based on combined cascaded classifiers, which is composed of a cascaded Haar classifier and a multi-layer perceptron (MLP) neural network. The second component is KLT-based tracking and speed estimation of the vehicles. To the best of our knowledge, among previous studies, no method successfully extracts traffic flow information from UAV videos in real-time. Hence, our work is among the first efforts. The key contributions of this paper are summarized as follows: (1) a framework for estimating multi-directional traffic flow parameters from aerial videos is proposed; (2) a novel method combining the KLT tracker, k-means clustering, and connected graphs for vehicle detection and counting is built; (3) introducing a detection method to UAV-based vehicle detection, which combines cascaded Haar classifiers with neural network; (4) proposing a general framework containing a motion-based approach and detection-based approach for traffic flow parameter estimation in aerial videos; (5) the system operates in a real-time manner; and (6) our system is robust to various challenging scenarios: it works well in both daytime and nighttime settings, both congested and uncongested traffic conditions, and is not sensitive to UAV movements (i.e., regular movement, vibration, drifting, changes in speed, and hovering).

The rest of the thesis is organized as follows: Chapter 2 is a literature review section that gives an overview of the state-of-art in UAV-based traffic monitoring as well as vehicle detection and tracking technologies. Chapter 3 introduces the overall framework and

methodologies for the motion-based approach in detail. Chapter 4 introduces the overall framework and methodologies for the detection-based approach in detail. Chapter 5 describes the platform and parameter settings. Chapter 6 and Chapter 7 present the experimental results using multiple representative aerial videos and evaluate the performance of the proposed motion-based and detection-based approaches. Chapter 8 draws the conclusion and discusses the future work.

Chapter 2 State of the Art

2.1 Detectors Used in UAV-based Traffic Monitoring

Many systems have already been developed to meet the demand for transportation surveillance. Among such systems, some are equipped with inertial measurement units, infrared detectors, or high-precision position and orientation systems, which can provide additional information besides simply collecting aerial video [8, 13-16]. However, the costs of these systems are relatively high and thus limit their use in practical applications [17]. Other systems are only equipped with cameras and may be more cost-effective, but require more sophisticated computer vision techniques to achieve similar performance to the more advanced systems.

2.2 Literature Review on Research Objective

In term of research objectives, previous studies in the area can be roughly divided into three categories. The first category is road detection [6, 18-20]. UAV-based road detection is important because these approaches can be applied to vision-based navigation of UAVs [19]. Moreover, road detection can help automatically determine the region of interest (ROI) in a given traffic monitoring scenario. For example, Kim et al. [19] presented a unique real-time approach to detect various types of roads and other corridors. Their method learns a road structure from a single image and can then be applied for detecting and localizing the road in successive frames of a video. Zhou et al. [6] proposed an efficient road detection and tracking method for UAVs (see Figure 2-1). This was the first work to introduce a tracking technique to speed up the localization of the road in a UAV video.

The second category of relevant research is vehicle detection and tracking. Numerous previous works focused on the methodology part of vehicle detection and tracking in aerial videos [21-30] (see Figure 2-2 as an example). These studies have made contributions in

improving the detection and tracking performance, i.e., increasing detection and tracking rate, reducing false-positive rate, and speeding up computing time by making use of cutting-edge computer vision techniques or even developing novel algorithms. These works also demonstrate tremendous possibility of supporting UAV-based transportation surveillance and traffic parameter estimation. For instance, Yu et al. [20] proposed a tensor voting computational framework to detect and segment motion patterns in a 4D space. The results show that some difficult problems that challenge the existing UAV systems can be addressed, but a long sequence is needed to detect motion patterns in their system, and it thus cannot meet the real-time requirement. Cao et al. [29] proposed a novel framework for UAV-based vehicle tracking using KLT features and a particle filter. Their method achieves very good tracking performance, but automatic detection of vehicles is not incorporated.

The third category focuses on traffic parameter estimation such as extracting speed, density, annual average daily traffic (AADT), travel time, and delay from aerial videos [4, 7-11, 13, 38]. When methods for detection and tracking developed in the aforementioned studies are combined with concepts and models in transportation engineering, useful traffic information can be extracted from the videos. Hence, automatic traffic monitoring can be partially achieved in practice. For example, Angel et al. outlined methods to estimate speeds, travel times, densities, and queueing delays from aerial imagery [8]. Their work is one of the milestones in estimating multiple key traffic parameters and works reasonably well. However, multiple data sources including a global positioning system (GPS) unit and an inertial measurement unit (IMU) are used in their system which increases the cost and data processing time. McCord et al. proposed a method to estimate AADT from satellite imagery and aerial photos using density information as algorithm inputs [7]. Their results showed the estimation accuracy is very high. However, their work focused on the modeling part rather than the automatic detection part. Shastry et al. [11] successfully incorporated KLT trackers

in their framework to estimate traffic flow parameters, but their KLT trackers were applied to image registration which decreases computing speed (see Figure 2-3). Their system is hence incapable of achieving real-time performance. Ke et al. [38] used motion -vectors to successfully estimate traffic flow speed in aerial videos. Their method is very efficient, but cannot extract other parameters besides traffic speed.

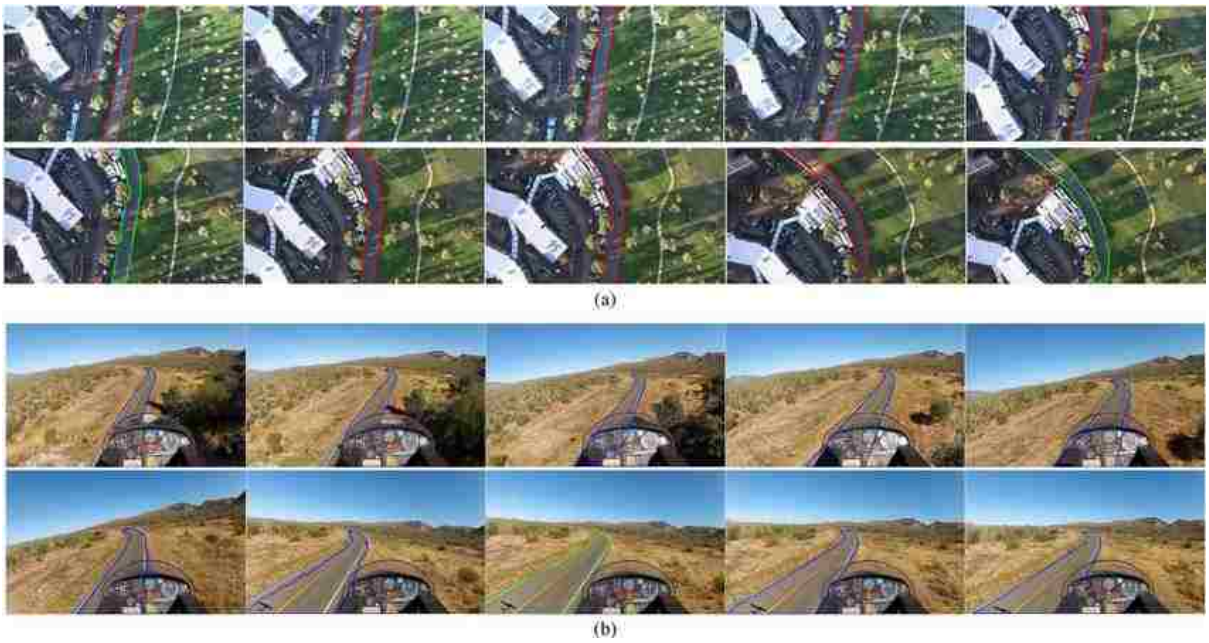


Figure 2-1 Road detection in aerial videos [6]

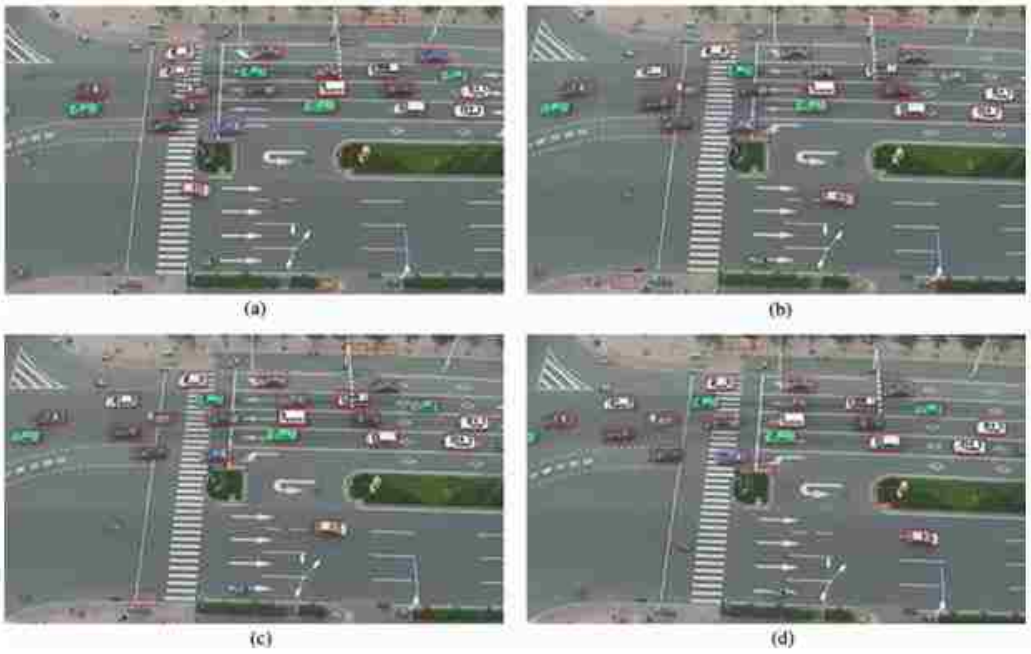


Figure 2-2 Vehicle detection and tracking in aerial videos [17]

Figure 2-3 Traffic flow parameter extraction from aerial videos [11]

2.3 Literature Review on Methodological Logic

Based on the methodology logic, previous work in aerial video-based vehicle detection or tracking can be roughly divided into two categories. The first category focuses on applying non-learning image processing methods to aerial videos. The most common two kinds of methods are image registration [8, 9, 11, 16, 30] and motion analysis [16, 28-30]. Image registration methods stitch image frames using matched feature points pairs from backgrounds. Stitched background can be seen as fixed background, thereby traditional methods dealing with fixed cameras can be applied. Image registration is more intuitive. Nevertheless, most image registration processes are relatively computationally expensive. Recently, optical flow-based motion analysis has become more and more popular for its high efficiency. Yu & Medioni [16] propose a Tensor Voting computational framework to detect and segment motion patterns in a 4D space. The results show that some difficult problems that challenge the existing UAV systems can be addressed, but a long sequence is needed to detect motion patterns in their system, thus cannot meet the real-time requirement. Cao et al. [28] propose a robust vehicle detection and tracking system by multi-motion layer analysis. However, in these optical flow-based work, if not a large number of feature points could be extracted from the background in the aerial video, the detection would not be properly done. The motion-based approach has the kind of issue, but in our detection-based approach, vehicle detector has nothing to do with the background. There are also some other image processing methods that have been used in vehicle detection from UAV's such as graph cut [40] and edge detection [41]. In general, image processing methods are usually limited to certain types of traffic scenes and conditions since they make use of the properties of the background and traffic.

The second category uses machine learning based detection approaches [17, 22, 24-26]. For example, Cao et al. [17] build a learning-based vehicle detection and tracking

framework using bLPS-HOG features, SVM classifier and motion analysis. Their method achieves good detection accuracy and near real-time performance, however, computing HOG features are usually time consuming. Cheng et al. [22] present an automatic vehicle detection system for aerial surveillance using dynamic Bayesian networks, which is partly based on learning of vehicles' colors. The results demonstrate flexibility and good generalization abilities of the proposed method. Cascaded Haar classifiers have already been used in vehicle detection from UAV's. Breckon et al. [25], propose a two-stage approach to vehicle detection: 1) primary detection using cascaded Haar classifiers and 2) secondary verification based on UAV altitude driven vehicle size constraints. However, the secondary verification may not be accurate since even for UAV's from the same height, different cameras may have different resolutions thus generate different constraints. In summary, compared to non-learning image processing technologies, machine learning-based approaches learns the patterns of vehicle appearances and thereby not as sensitive to certain scenes or traffic conditions. For example, optical flow-based motion analysis that can be applied to uncongested traffic may not be applied to congested traffic; methods handling intersections and freeways may also vary. However, learning-based methods only concern vehicles themselves.

Chapter 3 The Motion-based Approach

3.1 Overview

The motion-based approach for the proposed traffic flow parameter estimation can be segmented into four consecutive steps, which are described in details in the following sub-sections. In the first step, interest points are identified in a pair of consecutive frames; we used interest points where the eigenvalue of the second-moment matrix are large, i.e., Shi-Tomasi features. The Kanade-Lucas optical flow algorithm is then used to track interest points between consecutive frame pairs [12].

In the second step, the speed and direction of interest points on both vehicles and the background are used as inputs to a clustering algorithm. We found that the background of a frame is the cluster with the most points identified; thus the background cluster can be identified [28].

In the third step, in order to get the count of vehicles in each direction of travel, a connected graph method is applied to further determine the membership of interest points in each traffic stream cluster. The connectivity of two points is determined by the rule stating that interest points from one vehicle should have similar positions and velocities. Vehicle counts for each direction of traffic flow are thus the number of connected graphs per direction. Then, the speed of a vehicle can be calculated as the average speed of all interest points on that vehicle, but such a speed is in pixels per frame rather than a more sensible/intuitive speed unit (such as miles per hour, mph).

Finally, the actual traffic speed, density, and volume are estimated using reference markings on the roadway and the relationship among these three parameters. Figure 3-1 shows the workflow of the proposed motion-based approach.

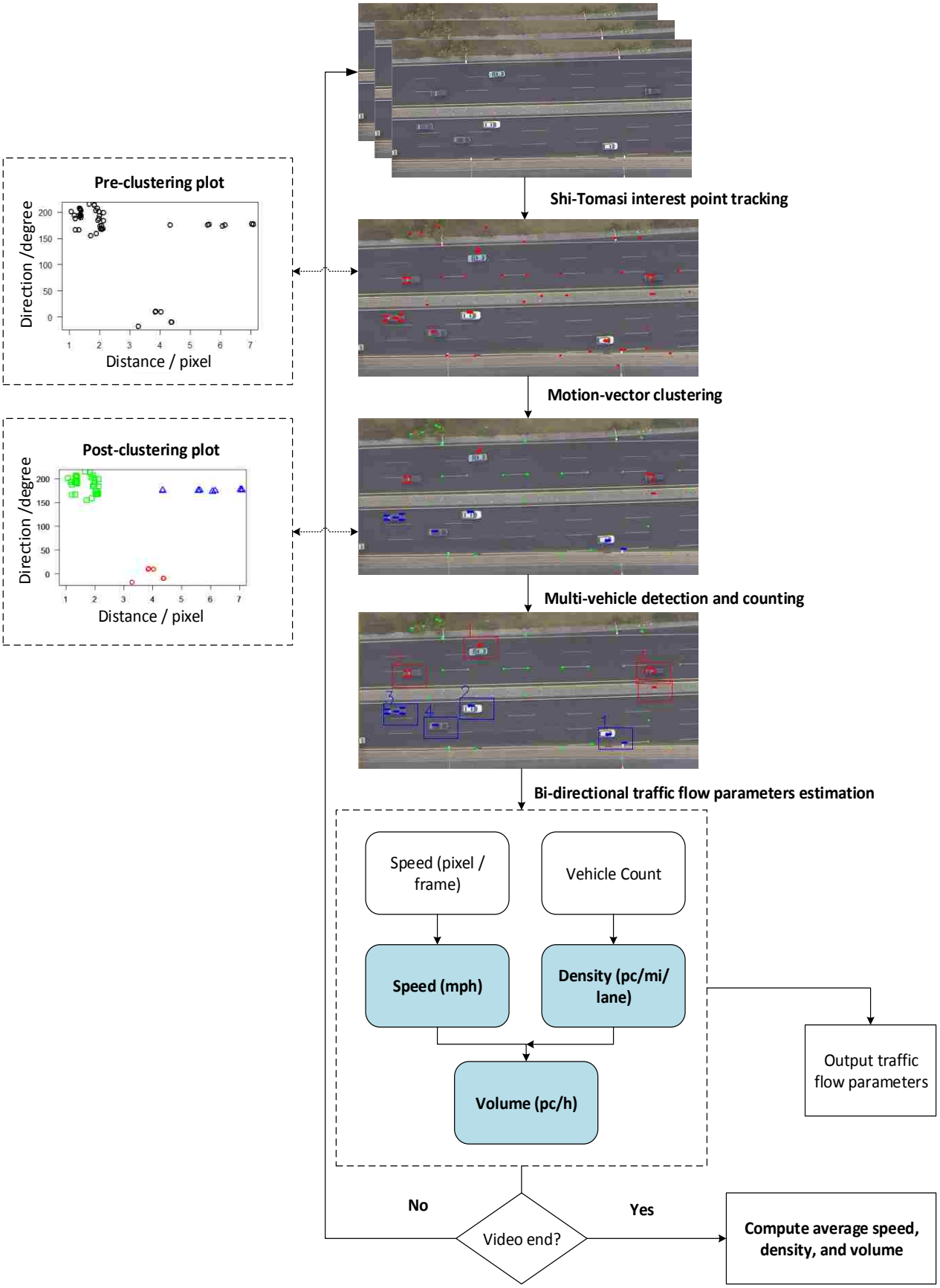


Figure 3-1 The workflow chart of the proposed framework.

3.2 Interest Point Tracking

Optical flow-based interest point tracking is a feature-based tracking process. One of the advantages of the optical flow approach is that it makes use of the spatial intensity gradient of the image to guide the correspondence search. Thus, it can achieve a faster image processing speed and interest point matching accuracy [31]. Selecting "good" features (i.e., those that can minimize an error criterion) is critical for tracking features robustly across image frames. The Harris corner detector is the most well-known feature detector [32]. However, Shi and Tomasi proposed another corner detector and proved it could outperform the Harris corner detector.

To select "good" Shi-Tomasi features, let the matrix

$$\mathrm{G} = \sum_{p_x-w_x}^{p_x+w_x} \sum_{p_y-w_y}^{p_y+w_y} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \tag{1}$$

be the second-moment matrix of image I about point u $= (p_x, p_y)$ in the window ω of size $(2\omega_x + 1) \times (2\omega_y + 1)$. Then, the interest points in *I* are located at the points u_i where G is non-singular, and the minimum eigenvalue $\lambda_{min} = \min(\lambda_1, \lambda_2)$ of G is above a specific threshold, λ_{th}. To provide non-maximal suppression, any interest point u_i is not considered if there is another interest point u_i' in a 3×3 neighborhood about u_i with a larger λ_{min}. Finally, any interest points after the first n, sorted in order of decreasing λ_{min}, are not considered.

After interest points in frames $I(x, y, t)$ and $I(x, y, t + 1)$ have been identified, an interest point u_i can be tracked from time *t* to *t+1* with the Kanade-Lucas algorithm for optical flow. In order to track points across distances on the order of several pixels, we used an iterative implementation with image pyramids [33]; the method is summarized in the following.

Let I^L be the pyramidal image of I at pyramid level L. Then I^L is related to the original image I by the relation

$$u^L = \frac{u}{2^L}, \tag{2}$$

where u is any point in I. The objective is to find the optimal displacement s^* such that the error function $\epsilon(s_x, s_y)$ is minimized, that is,

$$s^* = argmin\left[\epsilon\left(s_x, s_y\right)\right], \tag{3}$$

where the error function

$$\epsilon\left(s_x, s_y\right) = \sum_{p_x-w_x}^{p_x+w_x}\sum_{p_y-w_y}^{p_y+w_y}(I^L\left(p_x, p_y\right) - J^L(p_x + s_x, p_y + s_y))^2 \tag{4}$$

is the windowed sum-of-squared differences between images I^L and $J^L = I^L(\mathrm{t}+1)$.

If $s^{L+1} = g^{L+1} + v^K$ is an approximation for s^* at layer L+1, then the initial guess for the displacement for s^L is $\bar{s}^0 = g^L + v^0$, where

$$v^0 = [0\ 0]^T,$$
$$g^L = 2s^{L+1} = 2(g^{L+1} + v^K). \tag{5}$$

The update rule for v is based on the first-order Taylor approximation for the partial derivative of $\epsilon\left(s_x, s_y\right)$ *(33)*. For iteration *k*,

$$\mathrm{v}^k = v^{k-1} + \eta^k, \tag{6}$$

where

$$\eta^k = G^{-1}\bar{b}^k \tag{7}$$

$$\bar{b}^k = \sum_{p_x-w_x}^{p_x+w_x}\sum_{p_y-w_y}^{p_y+w_y}\begin{bmatrix}\delta^k(p_x,p_y)I_x^L(p_x,p_y)\\ \delta^k(p_x,p_y)I_y^L(p_x,p_y)\end{bmatrix}. \tag{8}$$

Here δ^k is the difference between the images I and J for the displacement $\bar{s}^{k-1}$, given by

$$\delta^k(p_x,p_y) = I^L(p_x,p_y) - J^L\left(p_x+\bar{s}_x^{k-1}, p_y+\bar{s}_y^{k-1}\right) = I^L(p_x,p_y) - J^L\left(p_x+g_x^L+v_x^{k-1}, p_y+g_y^L+v_y^{k-1}\right). \tag{9}$$

The iteration terminates either when $k > K-1$, or $\|\eta^k\|$ is less than a specified threshold. The aforementioned iterative process described is executed for each layer $L \in [0, L_m]$, starting from the image-layer L_m with the guess

$$g^{L_m} = [0\ 0]^T$$

Hence, for the layer $\mathrm{L} = 0$ corresponding to the original image I, an interest point u can be found at the point $u + s^0$ in J. Figure 3-2 shows an example of the interest points extraction and tracking process.

Figure 3-2 Optical flow-based Shi-Tomasi interest points extraction and tracking

3.3 Motion-vector Clustering

With the motion-vectors got from the first step, the two traffic streams can be identified and separated from the background using their motion characteristics. To ensure that interest points from traffic streams and the video background can be correctly separated, both the background and vehicles in a given traffic stream should satisfy a "similar motion" criterion. That is, interest points from the background should have small variation in l and θ between interest points (l denotes the moving distance, and θ denotes the moving direction.). Likewise, for traffic clusters the variation in motion for vehicles within a given traffic stream should not be too big. Intuitively, if the motion criterion is violated, the clusters corresponding to different traffic streams, or between a traffic stream and the background,

may become mixed. In practice, the motion criterion could be violated by conditions such as heavy congestion.

The result of the optical flow described in the previous section is a set of vectors *V* with elements $v_i = (l_i, \theta_i)$, where l and θ are given by

$$\begin{aligned} l &= \sqrt{s_x^2 + s_y^2} \\ \theta &= \arctan(s_y, s_x) \\ s^0 &= (s_x, s_y). \end{aligned} \tag{10}$$

Each element in V corresponds to an interest point u_i tracked from *t* to *t+1*, where s^0 is the displacement for u_i calculated from the optical flow. Given *V*, its elements can be clustered with respect to *l* and θ in a 2D velocity space using a standard k-means algorithm. For *k* traffic streams, the number of clusters should be set to *k+1* to account for interest points in the background. Specifically, for bi-directional traffic, *k* should be set to three.

Motion-vectors extracted from step one is plotted in the 2D velocity space and ready as the input to our clustering algorithm (see Figure 3-3). From Figure 3-3, three clusters can be pretty much identified even with human eyes. After applying k-means clustering algorithm, we show the three clusters using different colors (See Figure 3-4). In order to see where these points exactly come from (i.e., from traffic streams or background), we use the same color as in the post-clustering plot to show the memberships of the motion-vectors in the real frame (See Figure 3-5). From the colored points in the frame, it can be seen the clustering results match the intuition, which successfully identifies the bi-directional traffic streams as well as the background.

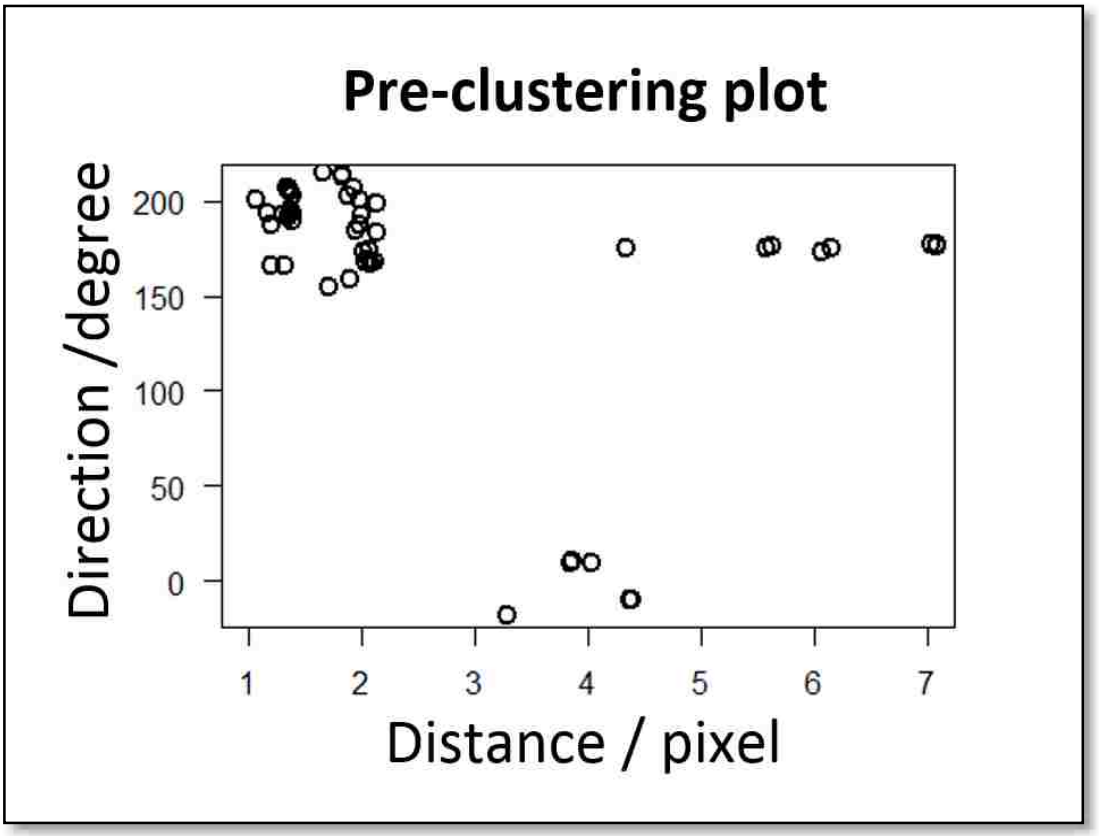


Figure 3-3 Pre-clustering plot showing the distribution of the motion-vectors

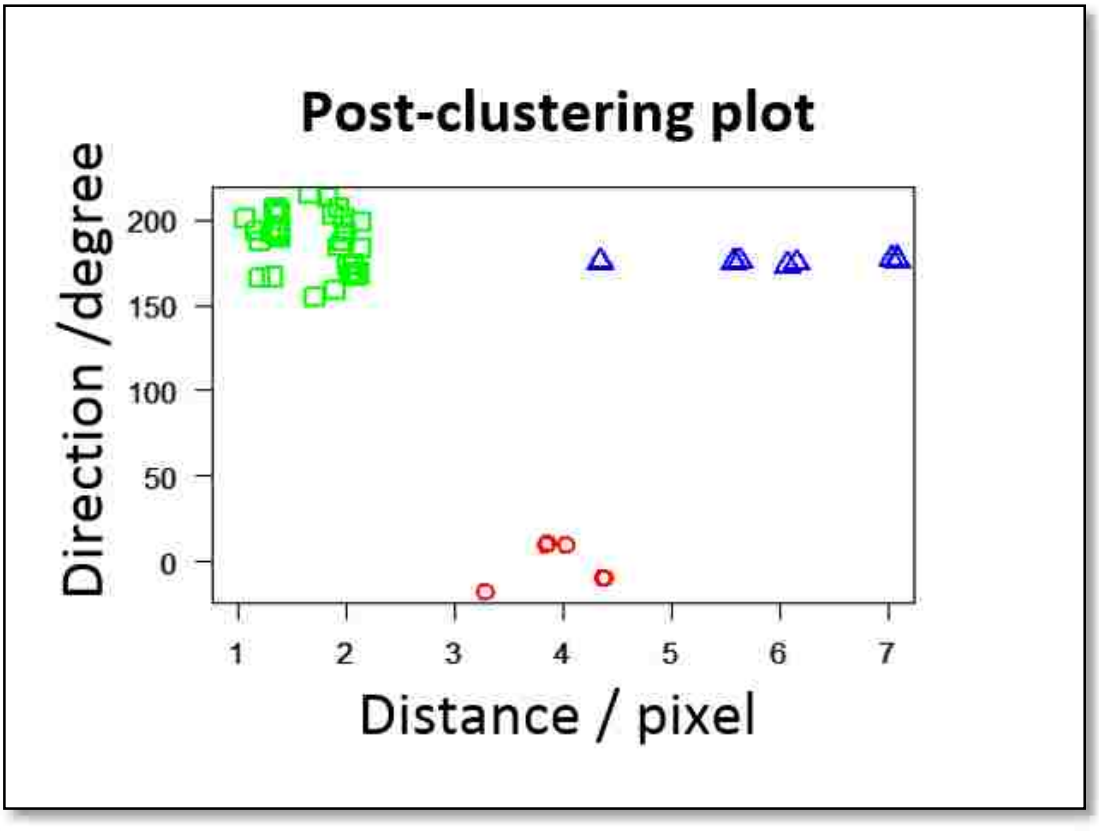


Figure 3-4 Post-clustering plot showing the membership of the motion-vectors

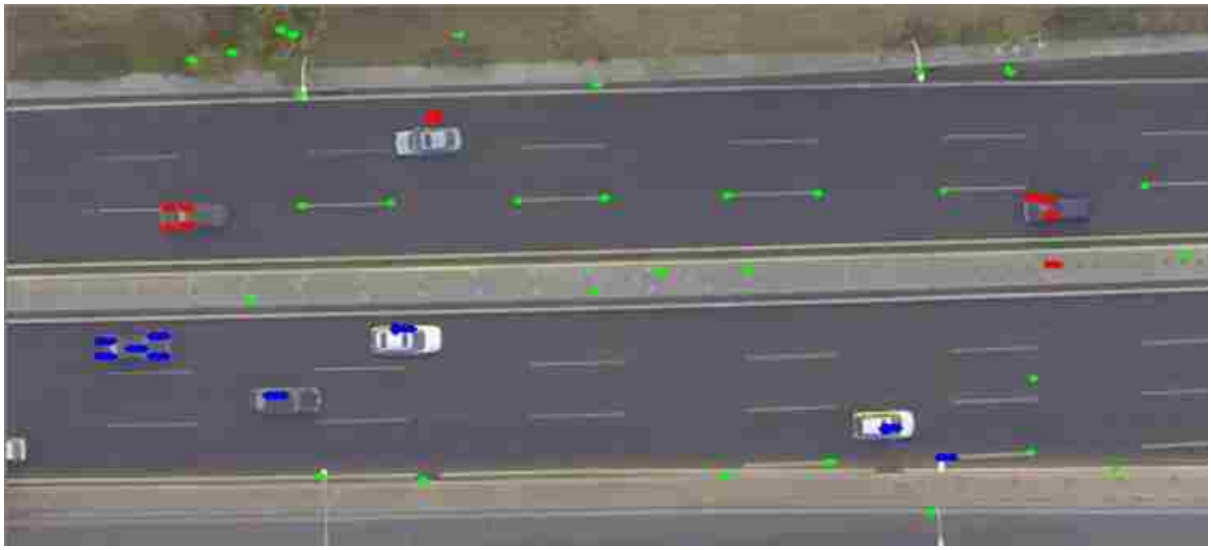


Figure 3-5 Clustering results shown in the original image frame using the same colors as in the post-clustering results

3.4 Connected Graph Based Vehicle Counting

So far, interest points have been grouped into clusters representing the background and different traffic streams. In each traffic stream cluster, interest points could come from different vehicles or the same vehicle. Determining the memberships of those interest points is crucial for vehicle counting and traffic stream speed estimation. Considering that vehicles are rigid objects, interest points on the same vehicle should share the same motion. Moreover, for an aerial video, interest points from one vehicle should be in close proximity in the 2D image space. Hence, to determine cluster memberships, a connected graph-based method is proposed. Assume an interest point is a vertex $\mathrm{P}_i(\mathrm{x}_i, y_i, l_i, \theta_i)$ in a 4D space, where x_i and y_i denote the x and y coordinates in the current image frame, l_i and θ_i denote the displacement and direction of P_i from the previous frame to current frame, respectively. The group constraints are defined as follows:

$$|x_i - x_j| < \alpha$$

$$|y_i - y_j| < \beta$$

$$|l_i - l_j| < \gamma$$

$$|\theta_i - \theta_j| < \delta,$$

where $P_i(x_i, y_i, l_i, \theta_i)$ and $P_j(x_j, y_j, l_j, \theta_j)$ are any two interest points in a specific traffic flow cluster. α, β, γ, and δ are the thresholds which determine the maximum difference of pixel numbers in the x and y directions and displacement. If all four constraints are satisfied, P_i and P_j are determined to be connected. According to this connectivity criterion, interest points can be grouped. In the grouping process, for each vertex that has not been assigned to a group, this vertex will be compared to existing vertex groups. The vertex will then be added to the group which satisfies the aforementioned criterion. If none of the groups satisfies the criterion, the vertex will be added to a new group and it will be the first element in this group. Ideally, one group represents one vehicle, thereby the motion of that vehicle can be estimated as the average motion of those interest points in the group. Likewise, the location of the vehicle in the current frame can be estimated as the centroid of the interest points. Figure 3-6 shows the further clustering results based on connected graph, and vehicle counting for each of the two traffic streams can be done.

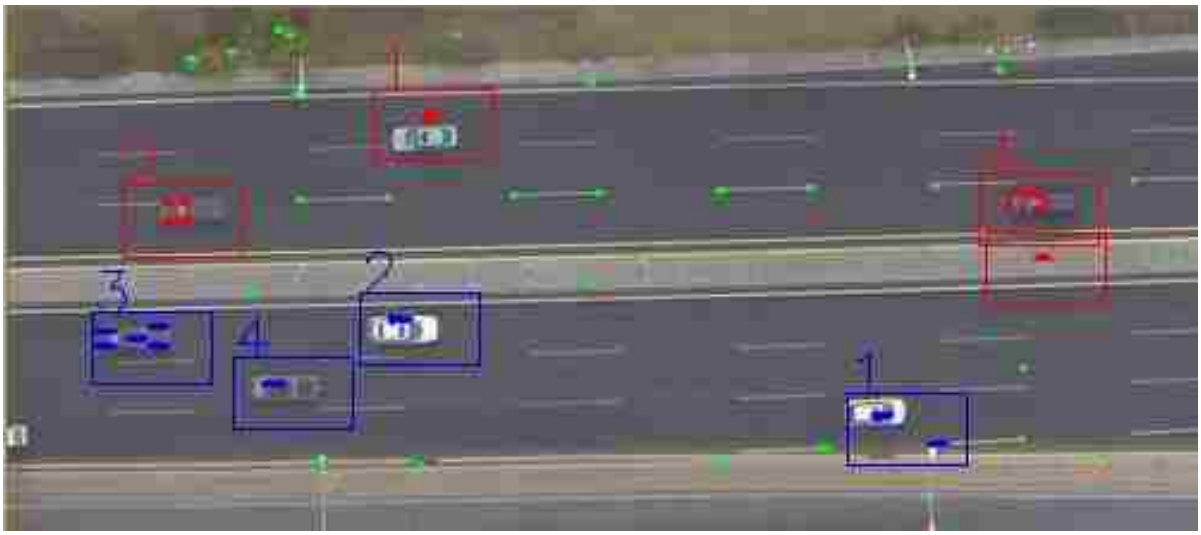


Figure 3-6 Example showing the connected graph based vehicle counting results

3.5 Traffic Flow Parameter Estimation

Speed, volume, and density are the three most important parameters of traffic flow. Our method can estimate these parameters in free-flow and moderately congested traffic conditions. For speed estimation, suppose there are *k* clusters (*k*=2 for bi-directional traffic) corresponding to traffic streams and a single background cluster such that the cluster center of the background is $v_{bg}(l_{bg}, \theta_{bg})$. For traffic stream $i, i \in [1, k]$, $v_{i,j}(l_{i,j}, \theta_{i,j})$ and $p_{i,j}(x_{i,j}, y_{i,j})$ denotes the velocity and position of vehicle j in stream i, respectively. Suppose the average velocity of traffic stream i relative to the background is $v_{i,avg}(l_{i,avg}, \theta_{i,avg})$ where

$$l_{i,avg} = \sqrt{d_{i,x}^2 + d_{i,y}^2}$$

$$\theta_{i,avg} = \arctan(d_y, d_x). \tag{11}$$

$$d_{i,x} = \frac{\sum_j l_{i,j}}{\sum_j 1} \times \cos\left(\frac{\sum_j \theta_{i,j}}{\sum_j 1}\right) - l_{bg} \times \cos(\theta_{bg})$$

$$d_{i,y} = \frac{\sum_j l_{i,j}}{\sum_j 1} \times \sin\left(\frac{\sum_j \theta_{i,j}}{\sum_j 1}\right) - l_{bg} \times \sin(\theta_{bg})$$

To covert distance in pixels to speed in miles per hour, we used the video frame rate and reference markings from video frames. For a video with frame rate f, where the pixel length and actual length of a reference marking are l_p and l_a, respectively, the actual speed s, of a vehicle that moves d_p in one frame pair is determined by

$$\mathrm{s} = \left(\frac{l_a}{l_p}\right) \times \frac{d_p}{f}. \tag{12}$$

To estimate density, suppose the vehicle count for traffic stream i is n_i and the length of the road segment in the ROI is l_{seg} for traffic in both directions; suppose the number of traffic lanes per direction is m_i. Further we suppose the UAV flies at a constant height. Then l_{seg} can also be determined by the ratio of pixel and actual lengths using reference markings. For bi-directional traffic, a UAV flies along a stretch of a roadway thereby l_{seg} stays relatively constant. Hence, the density of traffic stream i is given by

$$k_i = n_i/(l_{seg} \times m_i). \tag{13}$$

Then, to determine volume, since our method aims to solve parameter estimation of uncongested traffic flow conditions, the volume v_i for traffic stream i, is given by

$$v_i = s_i \times k_i \times m_i, \tag{14}$$

where s_i and k_i denote estimated speed and density of traffic stream i.

Chapter 4 The Detection-based Approach

4.1 Overview

The detection-based approach contains three main stages. In the first stage, the combined cascaded classifiers are built. In this work, Haar cascaded classifiers are trained using randomly generated Haar-like features and AdaBoost algorithm [42]. To train the multi-layer perceptron neural network as the strong classifier, each positive/negative image sample is treated as a pixel array storing pixels' grayscale intensities in sequence, then the optimal topology of neural network is found out with a topology search approach. Check Figure 4-1 for some manually collected training samples.

In the second stage, with the well-trained single classifiers, the detection framework is developed and shown in Figure 4-2. In the framework, first of all, a cascade of week Haar classifiers are used to reduce image search space, i.e., candidate windows. Although Haar cascades have a high false detection rate, they retain good recall, and they are very fast to compute. Then, MLP neural network detectors act as the final classifier on remaining candidate windows. MLP is capable of classifying complex objects with high accuracy, but it is slow to compute. This proposed detection method mingles both weak and sophisticated classifiers and has been proved to be both efficient and accurate.

Two sets of classifiers for detecting light-color vehicles and dark-color vehicles operate separately and their detection results are combined in the end. It is worth mentioning that normally a well-trained neural net should already do the classification without having to explicitly tell it what the most salient criteria are. However, making a distinction between light-color and dark-color vehicles in training makes higher detection rates and less training time with the same number of samples. This has improved our detectors' performance a lot given the fact that vehicle samples from a top-view perspective is still rare.

In the third stage, we apply the KLT tracker again in order to track the motion of both detected vehicles and background. The output of the detection process is some rectangles representing the detected vehicles. Thus, we can extract the motion-vectors inside these rectangles and average them to be the average motion of vehicles. Similarly, the motion-vectors outside these detected rectangles can be averaged as the background motion. In this way, with two more steps, which are the motion subtraction and motion-vector length converting, the average traffic speed can also be obtained.

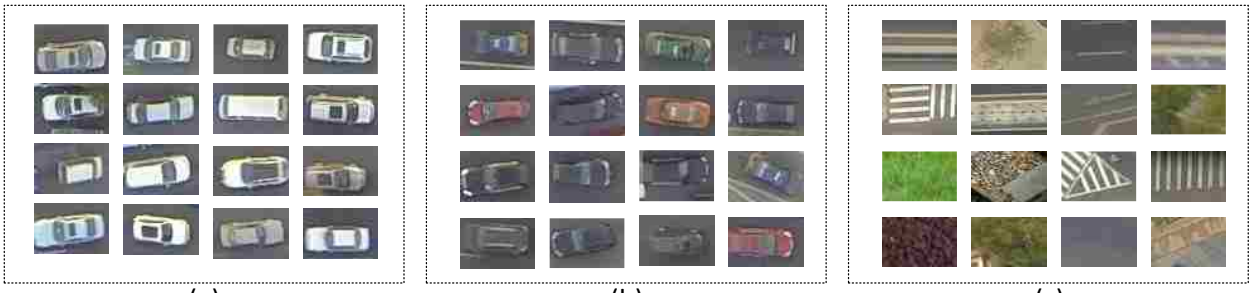


Figure 4-1 Examples of training samples: (a) positive samples of light-color vehicles; (b) positive samples of dark-color vehicles; (c) negative samples

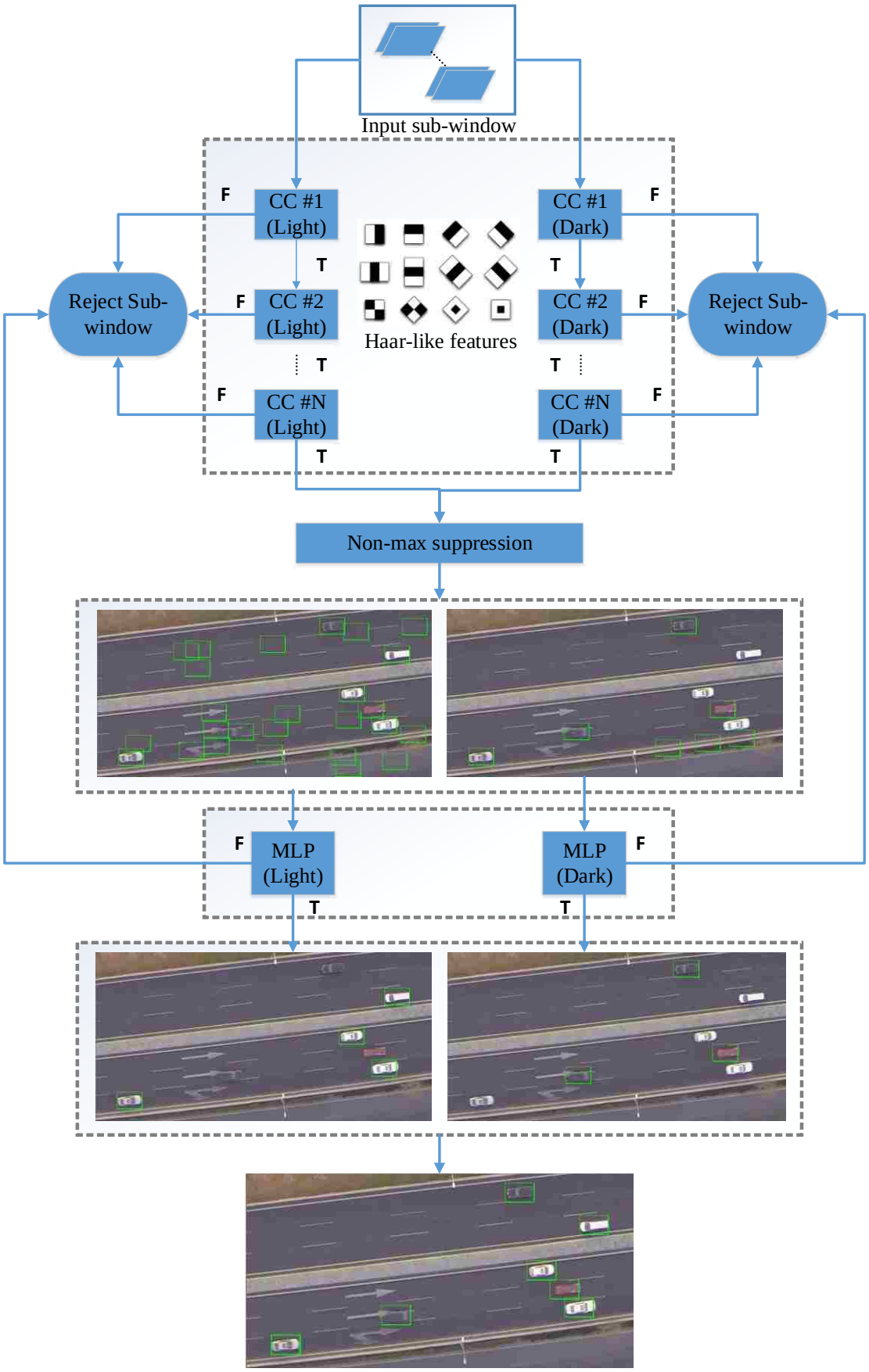


Figure 4-2 Workflow chart of the proposed detection processing framework

4.2 Training and Detection Framework

4.2.1 Cascaded Haar Classifiers

Originally, for the task of face detection, Viola and Jones employed a statistical approach to handle the large variety of human faces [43]. In this approach, Haar-like features and AdaBoost learning algorithm are the two major components. Instead of handling histogram of gradients-based features or pixel-level features, Haar-like features make use of a concept named "integral image", which is an intermediate representation for the original image. In integral image, rectangle features can be computed very rapidly. Compared with other approaches, which have to operate on multiple image scales, the integral image can achieve true scale invariance by eliminating the need to compute a multi-scale image pyramid, and significantly reduces the image processing time [44]. Using integral image, any rectangular sum in the original image can be computed as $sum(C) + sum(A) - sum(B) - sum(D)$ (see Figure 4-3 (a)), where $sum(x)$ represents the sum of all pixels located on the up-left region of x in the original image. With this concept, Haar-like features is defined as the difference of the sum of pixels of white and black areas inside a rectangle within the original image (Figure 4-3 (b)). A large number of Haar-like features in different scales and positions within an image are generated in the preliminary training process.

A single Haar-like feature is unable to achieve high classification accuracy, but a cascade of specifically selected classifiers can achieve high detection rates. The feature selection step, called *boosting*, is implemented with AdaBoost. The basic idea of the AdaBoost learning algorithm is boosting the classification performance of single weak classifiers by linearly combining them together, assigning different weight to each weak classifier inversely proportional to its training error rate. In each round of learning, one of the Haar-like features is selected from all the potential features and the weight of each sample is

updated based on the classification results. Basically, if a sample is not correctly classified, its weight for error computation in next round would increase. For each Haar-like feature f_j, a correspondent classifier $h_j(x)$ is defined by:

$$h_j(x) = \begin{cases} 1, & if \ \ p_j f_j(x) < p_j \theta_j \\ 0, & otherwise \end{cases} \tag{14}$$

where x denotes a tested sub-window, p_j denotes polarity indicating the direction of the inequality sign and θ_j denotes a threshold. The final weak classifiers are selected with least errors, i.e., they best separate the positive and negative samples. If N weak classifiers are selected at last, the final classifier $H(x)$ yields:

$$H(x) = \sum_{i=1}^{N} \alpha_i h_i(x) \tag{15}$$

where α_i denotes the normalized weight of the i th weak classifier $h_i(x)$.

Stage is a concept in cascaded classifier, which is constructed by training classifiers using AdaBoost. A sub-window that passes the classification tests of all stages is then able to be further processed, i.e., if any stage does not classify the sub-window as an object of interest, this sub-window would be immediately rejected (Figure 4-4). Hence, an excellent stage classifier should reduce false positives and not generate any false negatives at the same time. Otherwise, for instance, a first stage classifier only have a 80% detection rate, then not matter how well the following stages perform, this detector would not achieve a detection rate over 80%. Therefore, in the training process, the minimum hit rate is normally set close to 1.

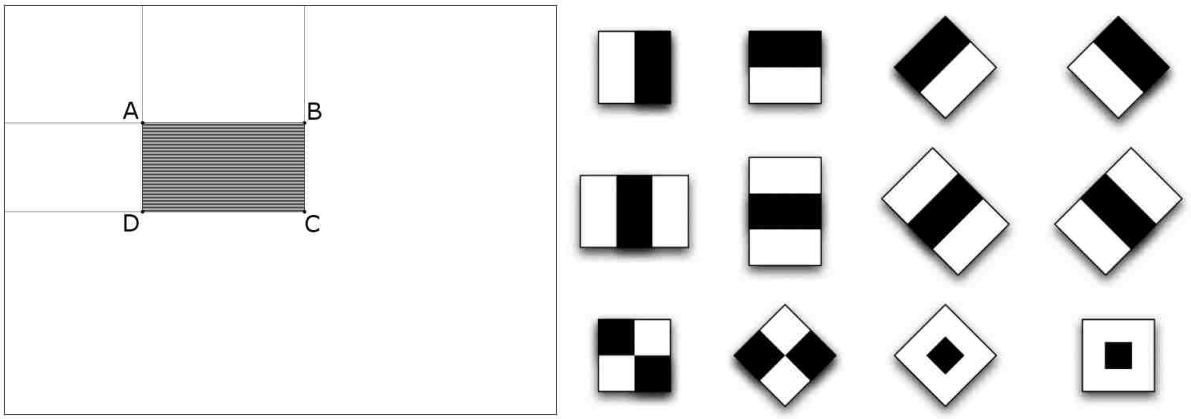


Figure 4-3 Basic concepts of Haar-like features: (a) the concept of "integral image"; (b) examples of Haar-like features

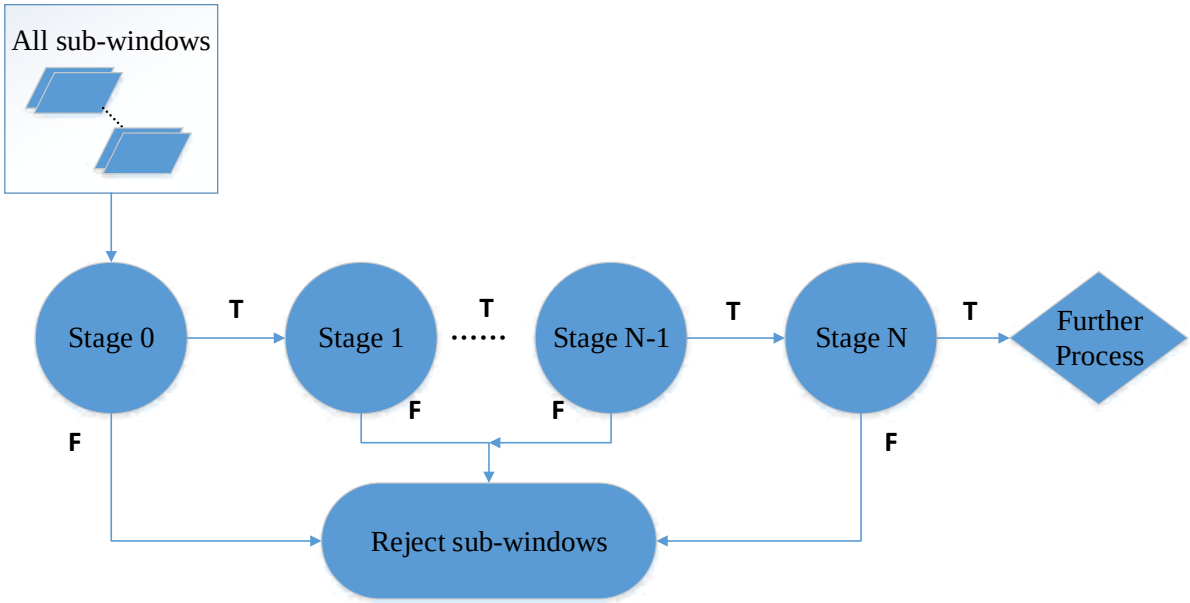


Figure 4-4 Schematic depiction of the detection cascade

4.2.2 Non-max Suppression

In the context of object detection, non-max suppression (NMS) is normally a necessary step, which has a large positive impact on performance measures [45]. As for every correct detection window, methods based on sliding windows over the whole image often produce

multiple windows near the correct location in the image. The result is a denser output than it should be, which is generally not satisfying for understanding the content of an image [45]. NMS is thus for handling the overlap of those detection windows in a defined radius. Usually, each detection window is assigned with a score, where higher score means the detection is more likely to be the correct object. Within the defined radius, only the detected object with the highest score will be retained.

In this paper, NMS is the second stage in the proposed framework. Generally, NMS could be implemented at two places: right after CC's or after MLP's. The main difference of these two options is the processing speed. If it is implemented after MLP's, much more windows remain to be processed by MLP's, and this will largely reduce the effectiveness of the idea of using combined classifiers. Hence, NMS is applied as the second stage right after CC's instead of the final stage in our framework.

4.2.3 Multi-layer Perceptron Classifiers

The outputs of the cascaded classifiers after processed by non-max suppression should have high detection rate but still some false positives. All of the remaining sub-windows are further processed by MLP's detectors. With the much smaller search-space, MLP's can achieve pretty fast detection speed and good detection rate to finish the detection process. MLP is the most commonly used type of artificial neural networks. They are universal since they can almost approximate any continuous non-linear function well. MLP consists of at least three layers, i.e., an input layer, one or more hidden layers and an output layer. Each layer contains one or more neurons linked with the neurons from the neighboring layers. Each neuron in MLP has input links and output links. For each neuron, the values got from the input links are summed up with certain weights, then the sum is transformed using a

certain activation function. In our method, the activation function for all neurons is the symmetrical sigmoid function f(x), which is given by:

$$f(x) = \beta \times \frac{1-e^{-\alpha x}}{1+e^{-\alpha x}} \tag{16}$$

This is the activation function used in our method; it is also the most common activation function used in MLP, called standard sigmoid, with $\alpha = \beta = 1$.
Therefore, to train the MLP, we need to know the number of hidden layers, the number of neurons in each layer and all the weights. The first two properties can be determined by a topology search procedure, which returns the topology with the least test classification error. The MLP's are then trained with a batched backpropagation procedure. Backpropagation, an abbreviation for "backward propagation of errors", aims at minimizing the loss function by updating the weights in MLP.

4.3 Estimation of Traffic Flow Parameters Using KLT Tracker

Similar to the previous motion-based approach, traffic speed and vehicle count are the two main parameters we extract. Then density and volume can be estimated from these two parameters. With the built vehicle detector, vehicle count can be easily obtained, which is the number of detected rectangles. However, traffic speed cannot be obtained straightforward. It requires multiple vehicle tracking and background motion estimation, which are both challenging tasks. In this detection-based approach, KLT tracker is still the key solution to motion estimation. The difference from the motion-based approach is that the KLT tracker does not apply to the whole image directly, instead, it applies to the region inside the detected windows as well as the region outside those detected windows separately. In this way, the motion-vectors inside the detected windows thereby represent the traffic motion, and those

outside the windows represent the video background motion. By adopting vector subtraction, the traffic speed in pixel/frame can be computed. Similar to the motion-based approach, reference markings such as school bus or lane marking with known length is used to convert the speed to mph. Figure 4-5 is a sample frame showing the detection results, as well as the motion-vectors inside (red) and outside (green) the detected windows. Finally, with the estimated traffic speed, vehicle count and reference markings, traffic flow parameters can be obtained.

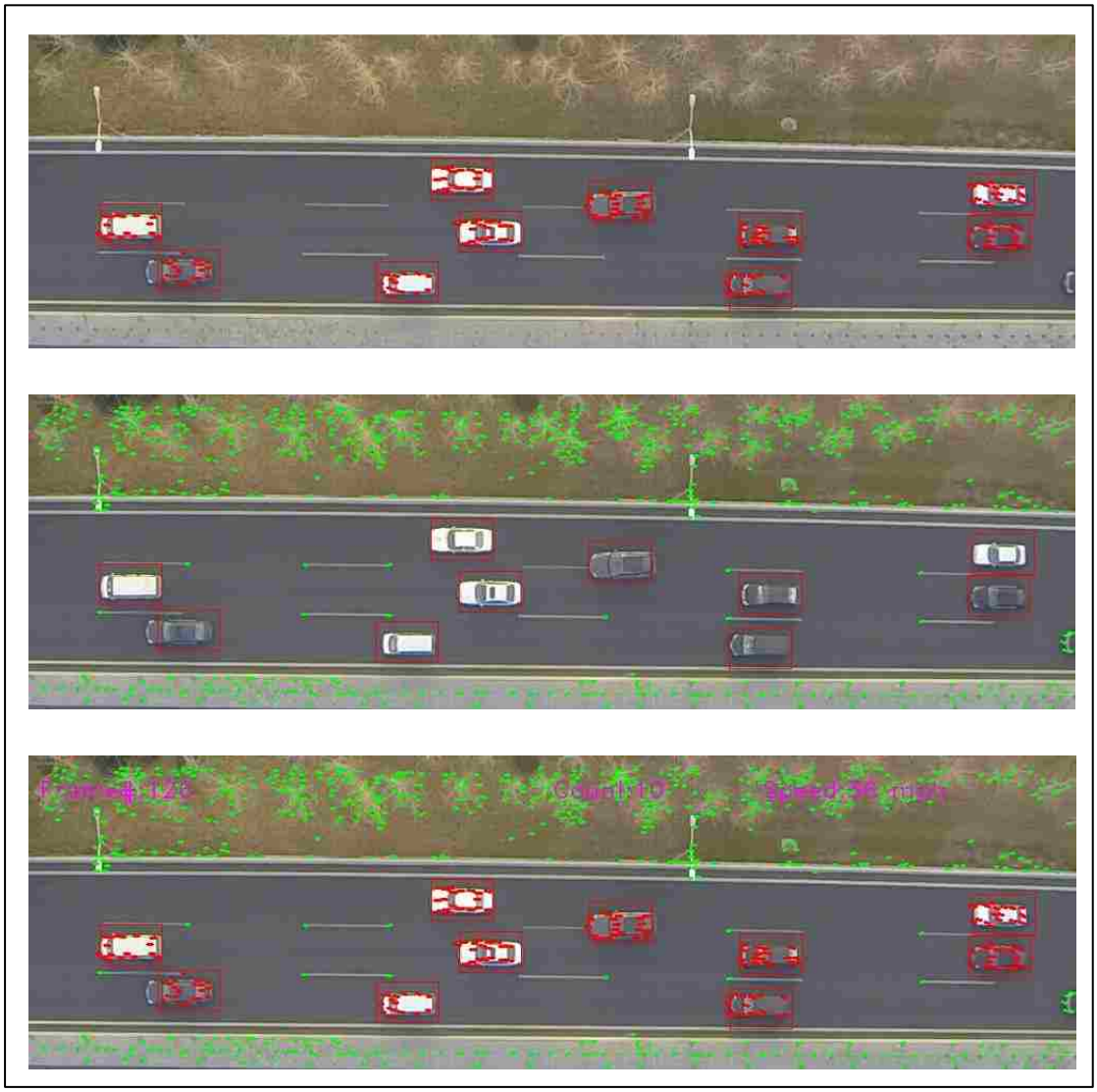


Figure 4-5 A sample frame showing the detection result, as well as the motion-vectors estimated by KLT tracker inside and outside the detection windows

Chapter 5 Preliminary System Settings

5.1 Settings for Motion-based Approach

The method was implemented with C++ and OpenCV 2.4.11 [34]. The test dataset, which will be discussed in detail later, consisted of a 280-frame video at a 960 × 540 resolution, taken at 24 frames-per-second by a UAV traveling above a freeway segment. The ROI was selected to include six lanes of traffic moving in two directions (i.e., three lanes per direction), denoted Direction A (with traffic moving towards the left) and Direction B (with traffic moving towards the right), resulting in a window of 500 × 220 pixels (see Figure 5-1). As aforementioned, reference markings were used to compute the ratio of pixel to actual (i.e., real-world) length. Different reference markings can be used, such as the length of a school bus [11]. Here, lane markings were used with a measured pixel length and actual length of 36 pixels and 6 meters, respectively.

As for parameters in the method, five need to be carefully adjusted, the first of which is the number of tracked interest points. The interest points are ranked by their matching errors. If N points are tracked, the first N points with the smallest errors would be tracked. Thus, if N is set too small, some vehicles would not be detected because there would be no interest points on them. On the other hand, if N is set too large, the motions of some interest points would be incorrectly estimated leading to some false-positives (non-vehicles detected as vehicles) in vehicle detection. We also observed that N influences the processing speed substantially. The other four parameters are the four thresholds in the connected graph based vehicle detection and counting step, i.e., $\alpha,\beta,\gamma,\delta$. If either α or β is too small, interest points from the same vehicle may be classified into more than one group. Thus, one vehicle can be erroneously detected as two or more vehicles. In contrast, if α or β is too large, two vehicles that are close to each other can be incorrectly classified as one vehicle. Likewise, if γ and δ

are too small, due to the estimation errors in tracking interest points, interest points coming from the same vehicle can sometimes have small differences in speed or direction values. These differences can also result in erroneously counting one vehicle as two or more. If γ or δ is too large, in some cases the 4D space for the connected graph based grouping is reduced to 2D or 3D, resulting in less effective classification.

Figure 5-1 The original frame with the ROI

Since the influence of α and β is the most intuitive, an example showing the influence of these parameters is presented in Figure 5-2. Four combinations of α and β were tested, showing the influence of thresholds on vehicle counting. Comparing the upper-left snapshot with the lower-left one, in the 80^{th} frame, larger thresholds resulted in detecting the two white vehicles in Direction B as one vehicle. However, in the 280^{th} frame, where there is a bus, the larger thresholds correctly classified the interest points on the bus as one connected group. The smaller threshold combination, however, failed to count it as one vehicle, and instead three buses were detected. Obviously, the errors in classification affect traffic parameter

estimation. Hence, setting proper parameter values is important to both accuracy and computing speed. Different parameter combinations were tested in this case, and the experiment presented here is with $N = 100$, $\alpha = 40$, $\beta = 14$, $\gamma = 0.5$, and $\delta = 10$ considering the tradeoff between accuracy and computing speed.

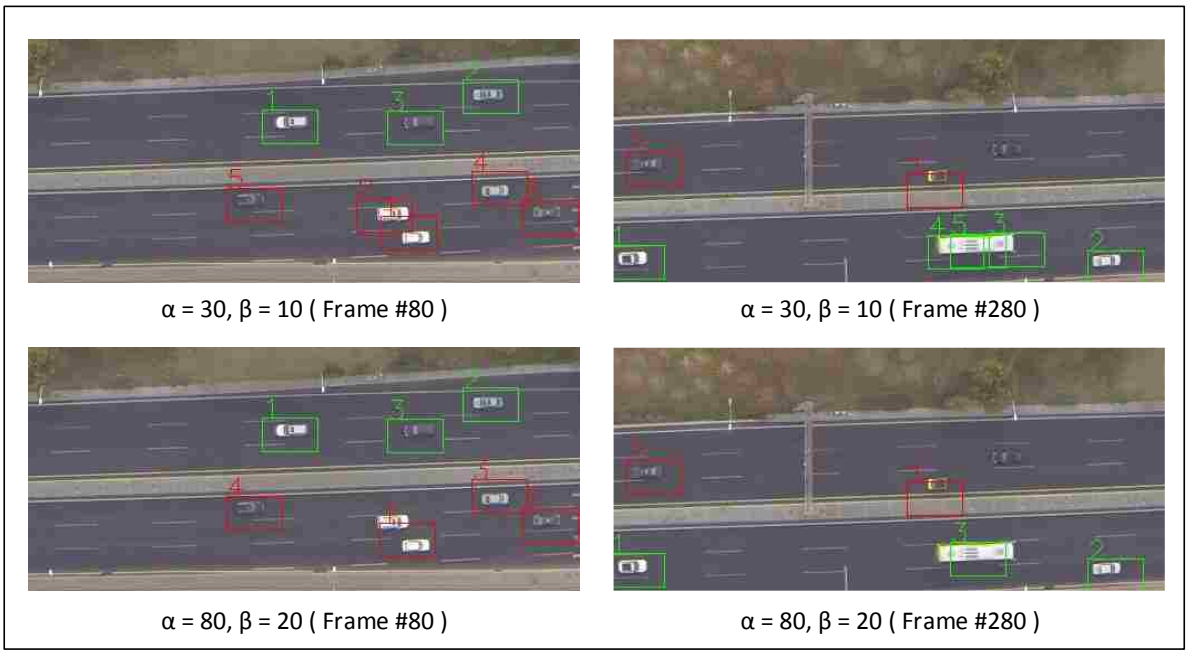


Figure 5-2 Examples showing the influence of parameter settings

5.2 Vehicle Detector Training for Detection-based Approach

Since the performance of the detection-based approach largely relies on the vehicle detector, besides the number of tracked interest points inside and outside those detected rectangles, the vehicle detector training is a very important preliminary process in this approach.

In the case study, several aerial video clips are specifically used for training data collection, which were taken from UAVs flying over different roadway segments. We have collected over 14000 training samples for testing our approach. As we are getting more UAV video data, we are keep training vehicle detectors with more samples. The latest trained top-view vehicle detector acts as an example in the study to show the effectiveness of the main idea of the proposed detection-based method. Both the positive and negative samples were

manually cropped from the training videos in the original size of 60 × 40 (width × height) (See Figure 4-1). Generally, vehicles in light colors (white, gray, light blue, etc.) are labeled as light-color vehicles; vehicles in dark colors (black, dark blue, red, etc.) are labeled as dark-color vehicles. There could be slight difference when different people do the labeling; however, even if a car is detected by both light-color vehicle detector and dark-color vehicle detector, the non-max suppression process will keep only one detection window, thus to ensure the vehicle count estimation accuracy. In the cascaded Haar classifiers training, the sizes of the samples are kept 60 × 40. However, in MLP training, the samples are scaled to the size of 20 × 13. The reason for downsampling is that it reduces the dimensionality of the input feature vector from 2400 to 260, thus largely speeding up both the training and detection processes. All the algorithms were implemented with C++ and OpenCV 2.4.11 as well.

For the CC's, we train each stage of the cascade with 400 and 800 randomly selected positive and negative samples, respectively. 7 stages of CC for light vehicles and 6 stages of CC for dark vehicles are finally got. Not too many stages are trained here for CC's because our purpose of using CC's is to prune the search-space rather than get the final detection results, so that too many stages may filter out true detections.

For the MLP neural network, we select an optimal topology by performing a brute-force search of all networks with number of hidden layers $h \in [1, 6]$ and nodes per layer $n \in \{32, 64, 128, 256, 512, 1024\}$, then 55986 topologies in total are tested and compared with one another. The MLP's are trained with the batched back-propagation procedure, with batches of 3000 and 1000 randomly selected negative and positive samples, respectively. The performance of a given network topology is rated by its error on a separate validation set that includes 3000 and 1000 randomly selected negative and positive samples.

Table 5-1 summarizes the results for the best CC's and MLP's that we were able to train with the current collection of samples. In the table, FP is an abbreviation for "False Positive", and MSE is short for "Mean Squared Error".

Table 5-1 Training results for the CC's and MLP's.

CC			**MLP**		
	Dark vehicle	Light vehicle		Dark vehicle	Light vehicle
# of stages	7	6	Input nodes	256	256
Min hit-rate	0.999	0.999	Hidden layers	1	1
Max FP rate/stage	0.5	0.5	Hidden nodes	512	64
			Output nodes	1	1
			MSE (validation)	10.32%	1.30%

Chapter 6 Experimental Results of the Motion-based Approach

6.1 Traffic Flow Parameter Estimation Results

For each pair of the consecutive frames in the aerial videos, the traffic parameters were calculated for the two travel directions. The results are shown in Figure 6-1. In both directions, the average speed of the traffic streams was relatively stable over time while the density and volume varied more obviously. The results are considered reasonable because in the free-flow and moderately congested traffic condition (which is the case in our example), vehicles travel at a relatively constant speed determined by the traffic state. However, the estimated density was very sensitive to the distribution of vehicles in different frames (i.e., there could be multiple vehicles within some frames but fewer vehicles in other frames). As volume is calculated by speed and density, the variation of volume over different frames was expected.

In practice, reporting frame-by-frame traffic information is unnecessary and tedious. Instead, aggregated traffic flow parameters similar to those reported from inductive loop detectors are more useful to people, such as traffic engineers, than instantaneous results. In our study, we further calculated aggregated traffic flow parameters from the frame-based information. The aggregated speed, density, and volume for Direction A were 54.6 mph, 25.4 pc/mi/lane, and 4177.6 pc/h; for Direction B, they were 42.5 mph, 41.4 pc/mi/lane, and 5222.9 pc/h (See Table 6-1). We see that the traffic stream in Direction A had a higher speed and lower density, a result which is quite intuitive (see Figure 6-1).

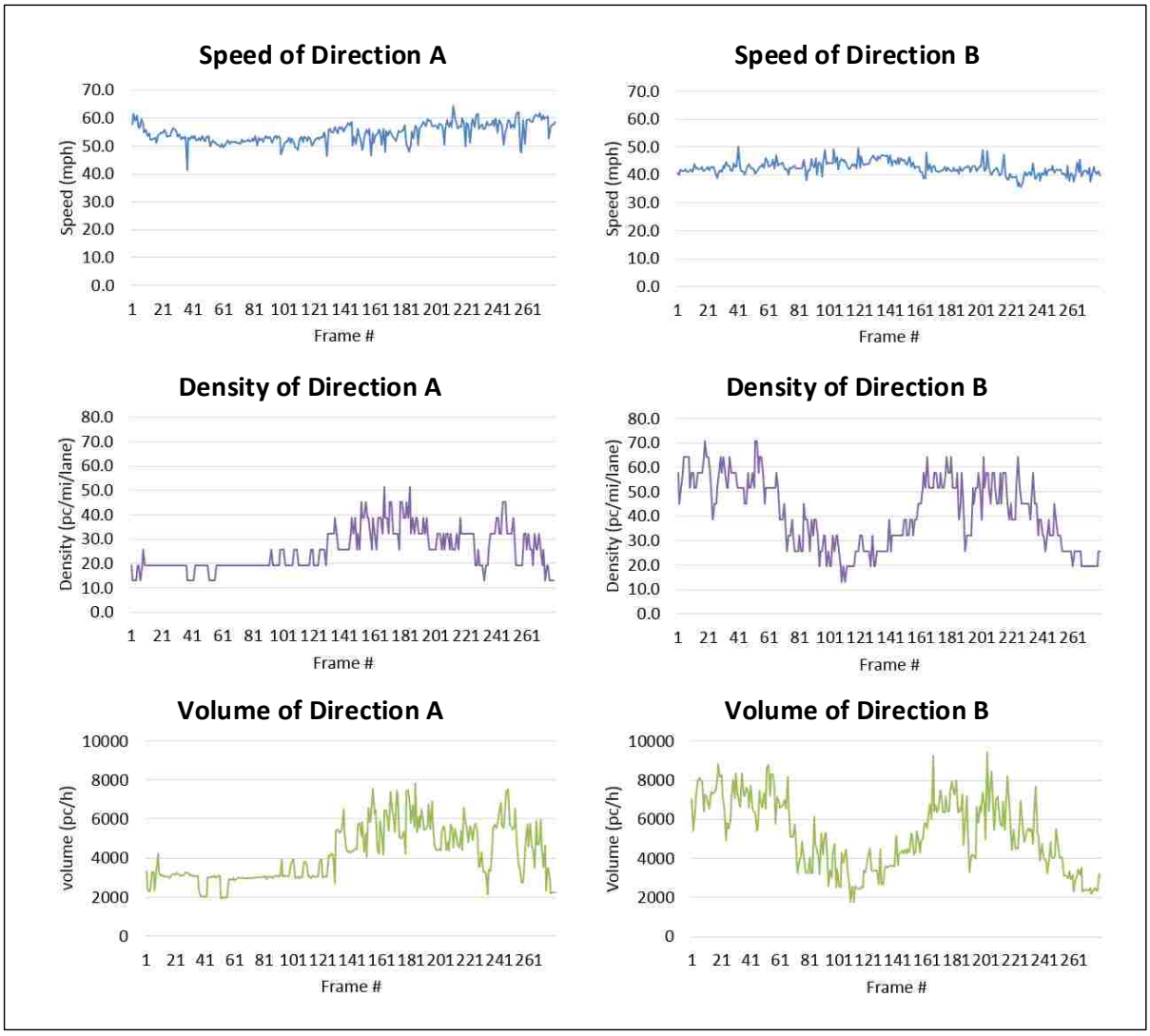


Figure 6-1 Estimated instantaneous bi-directional traffic parameters for the freeway segment in the test aerial video

Table 6-1 Aggregated flow parameters for the bi-directional traffic streams

	Number of lanes	Speed (mph)	Density (pc/mi/lane)	Volume (pc/h)
Direction A	3	54.6	25.4	4177.6
Direction B	3	42.5	41.4	5222.9

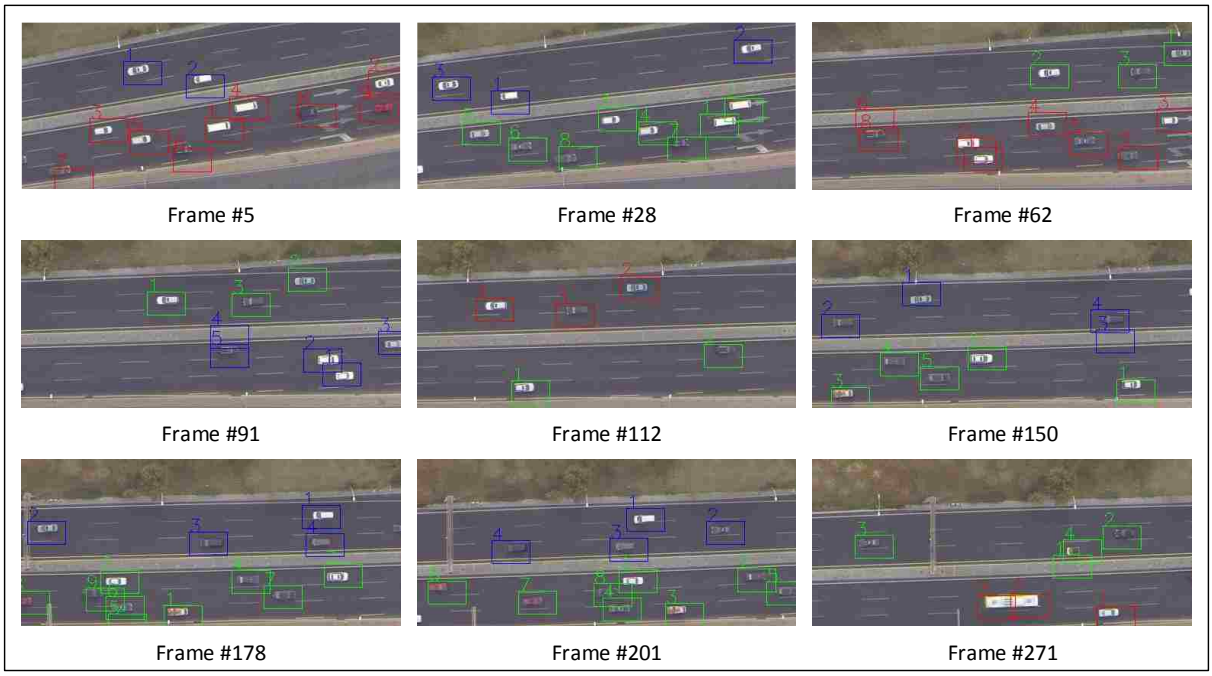


Figure 6-2 Selected frames showing the bi-directional vehicle counting results

6.2 Speed and Count Estimation Accuracy

To validate the traffic parameter estimation accuracy of our proposed methods in case study, ground truth data on the average speed per pixel by frame and vehicle count for traffic flow in each travel direction were measured from the UAV videos. An on-screen pixel measurement tool was used to manually collect ground truth speed data. The speed of individual vehicles was measured over intervals of five consecutive frame pairs. There were two factors we considered for choosing our measurement interval: 1) generally, the smaller the interval is, the larger the measurement error would be; and 2) if the interval is too large, resolution of the speed data would deteriorate. Normally, in five frames an individual vehicle moved over 20 pixels in the test video, and the actual time interval of five frames was less than 0.2 seconds; in such a period, the speed of a vehicle can be viewed as constant. For the aforementioned reasons, the decision to use five frame pairs as the measurement interval was made.

The measured ground truth speed, estimated speed by our method, and the error rate for each travel direction are presented in Figure 6-3. Let ε denotes error rate, and $y_{estimated}$ and y_{truth} denote estimated value and ground truth, respectively. Then,

$$\varepsilon = {|y_{estimated} - y_{truth}| \times 100\%}/{y_{truth}} \tag{17}$$

$$accuracy = 100\% - {|y_{estimated} - y_{truth}| \times 100\%}/{y_{truth}}. \tag{18}$$

Clearly, the trend of ground truth speeds and estimated speeds are both rather similar for Direction A and B. That being said, our method achieved promising and accurate results in speed estimation. The average error rates for Direction A and B were 2.366% and 2.634%, respectively (see Table 6-2); the maximum of error rates was less than 10%. We also found that the variances of the estimated speeds were both slightly larger than the ground truth data. This is because there were some false-positives in vehicle detection for each travel direction. Such detections were clustered as vehicles and their motions were added to the real motion. Since their motions were random and the traffic flow motion was relatively constant, they led to increased variance in parameter estimates. Such an issue could happen when an interest point from the background is mistakenly matched with another nearby interest point which shares high similarity in terms of intensity and gradient changes in the KLT tracking process.

Compared to speed, error rates associated with vehicle count estimation had higher averages and larger standard deviations. The average error rates were 17.393% and 17.055% for Directions A and B, respectively (see Table 6-2). From Figure 6-4 it can be seen that in most frames, the estimated vehicle counts were larger than the ground truth. This was due to the parameter settings of the number of tracked interest points. On one hand, the tracked interest points were ranked by their matching errors. Thus, increasing the associated parameter a small amount resulted in tracking all vehicles, but also increased the probability

of the occurrence of false-positives. During the experiments, we observed that if this parameter was set too small, the vehicle count would decrease and some vehicles were not detected. Ultimately, we set the thresholds of the parameter values associated with the connected graph process each to be relatively small. As we discussed before, use of smaller thresholds generates more vehicles in the estimation process.

It is important to note that the maximum error rates for vehicle count estimation reached 100% for Direction A and 75% for Direction B. These error rates may be quite surprising and discouraging at first glance. As such, our research team examined the data for the frames with high error rates. We found that large errors occurred most often when the image frames included very few vehicles. For example, when a 100% error rate appeared, there was only one vehicle within the detection zone, but our algorithm recognized it as two objects resulting in the extremely high calculated error rate. When there are some more vehicles within the detection zone, the error rate for vehicle count tends to be much lower. That said, even if we include these abnormal frames in the calculations of the average count estimates, we still got relatively accurate detection results, demonstrating the robustness and reliability of our method.

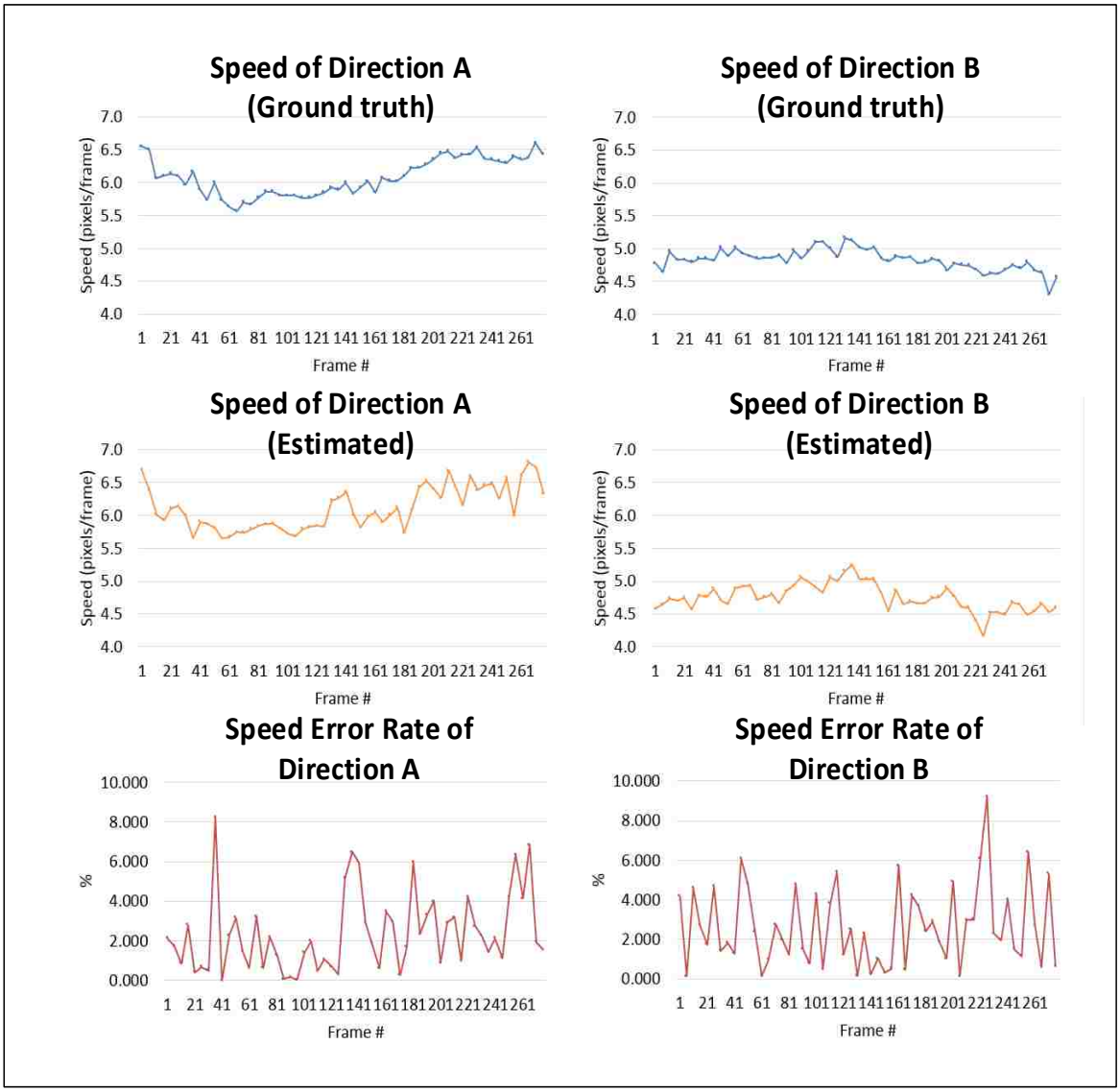


Figure 6-3 Plots showing the speed estimation accuracy

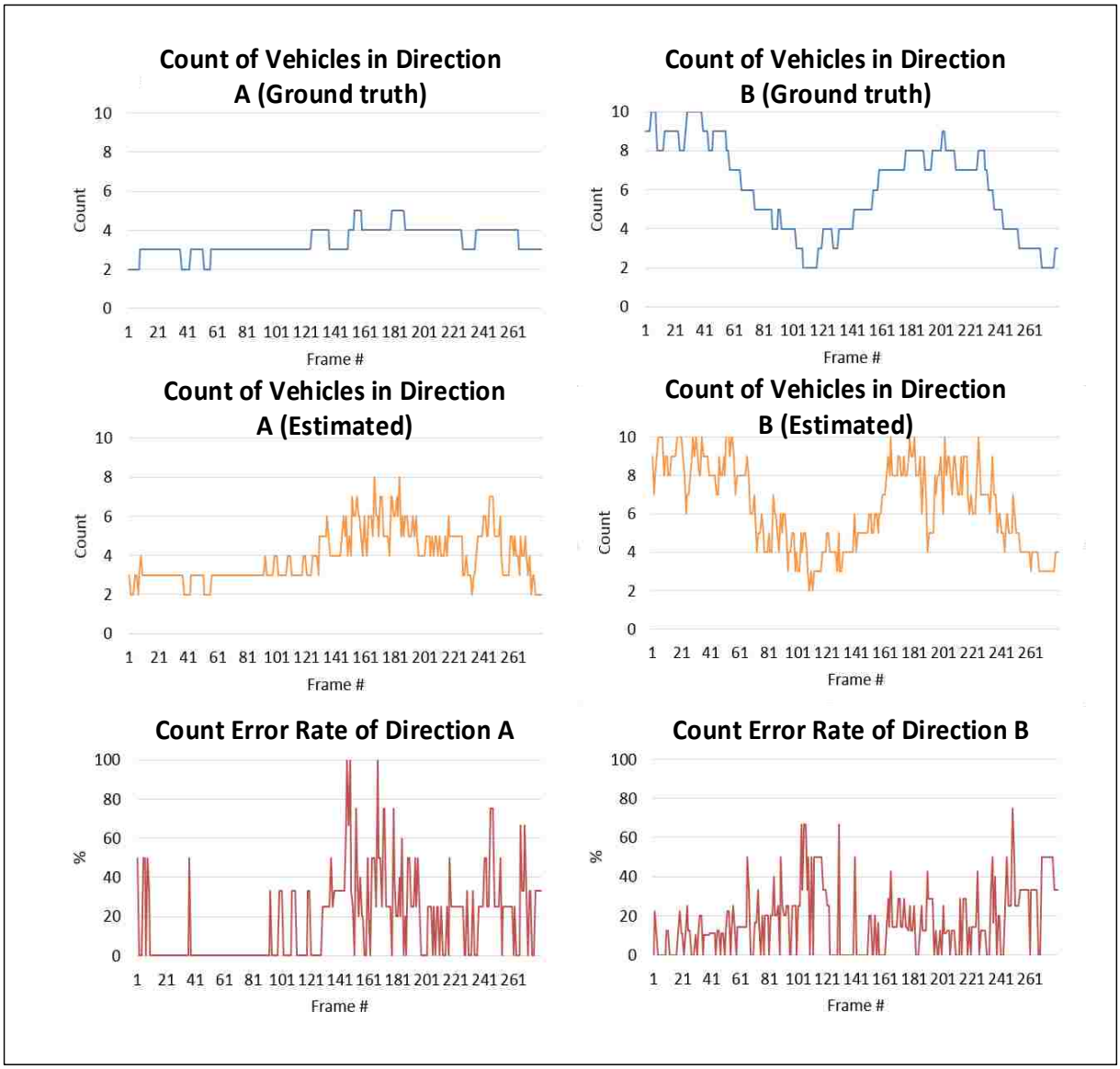


Figure 6-4 Plots showing the vehicle count estimation accuracy

Table 6-2 Statistics of estimation accuracy analysis on the 280-frame test video

		Speed (pixels/frame)				Vehicle Count			
		Avg.	Std.	Max.	Min.	Avg.	Std.	Max.	Min.
Direction A	Ground Truth	6.081	0.279	6.6	5.6	3.404	0.69	5	2
	Estimation	6.104	0.323	6.8	5.6	3.946	1.307	8	2
	Error Rate (%)	2.366	1.941	8.3	0	17.393	21.955	100	0
Direction B	Ground Truth	4.832	0.155	5.2	4.3	6.025	2.328	10	2
	Estimation	4.752	0.199	5.2	4.2	6.371	2.224	11	2
	Error Rate (%)	2.634	2.001	9.2	0.1	17.055	16.853	75	0

6.3 Analysis of System Performance in Different Scenarios

To further validate the performance of our proposed method, four more aerial videos taken in different scenarios were examined. The total monitoring time length was about forty minutes and very good detection results were observed. In order to carefully evaluate the performance, 1690 frames were manually examined frame-by-frame to obtain the ground truth values of speed and traffic count/volume. The traffic condition, time of day and the movement of the UAV were also different in each video; however, the performance of our system was fairly stable, staying at a high-accuracy level. The summary of the estimation results and performance are presented in Table 6-3. In the second test video, for the two traffic streams, speed estimation accuracy reached 97.300 % and 94.220 %; count estimation accuracy reached 86.612 % and 82.731 %. In the third test video, speed estimation accuracy was 94.995 % and 93.368 %; count estimation accuracy was 81.449 % and 93.335 %.

Other challenging scenarios, different from those previously tested, were tested in videos #4 and #5. One was an aerial video taken at night, in which the UAV experienced drift during its regular movement along the freeway. Normally, vehicle detection is much more difficult at night than during the day. However, we noticed that the count estimation in our test video #4, i.e., the video taken at night, obtained the highest accuracy among all test videos, almost reaching 90% (see Table 6-3). Figure 6-5 shows the vehicle detection results a selection of randomly selected frames. It can be seen there are very few false negatives and false positives. We notice that as far as long as vehicles are driving with their lights on at night, they will be detected with high probability and thus few false negatives will be generated. This observation is due to the fact that interest points most often come from locations in a frame where there is light when the frame was filmed at night time. Also, the number of false positives generated from this observation of this video was also quite fewer; this can be explained using Figure 6-6. False positives occur when non-vehicles are detected

as vehicles. In other words, they usually result when points in the image background are mistakenly recognized as vehicles. In our framework, false positives are primarily caused by inaccurate motion estimation of interest points from the background. In 6-6, it can be seen that during the day there were many more interest points in the frame background than at night, hence increasing the possibility of inaccurate motion estimation. Therefore, it is reasonable that our system performed even better at night than for some cases in which the video was taken during the day.

In order to compare the performance of our system on moving-background videos and fixed-background videos, another test video was taken when the UAV was hovering over a freeway segment, with no vibration or drifting, i.e., the video background was not moving at all. Figure 6-7 and Table 6-3 present the detection and estimation results of video #5. The results prove that the proposed system works well on both moving-background aerial video and fixed-background aerial video, thus further demonstrating the system's ability to adapt to different monitoring conditions.

By considering the results of the analyses collectively, we can conclude that the performance of our method is not sensitive to the dimensions of the ROI, UAV movements, or light conditions. The slight differences in detection accuracy that were observed could be caused by the differences in heavy vehicle ratios between videos or system parameter settings.

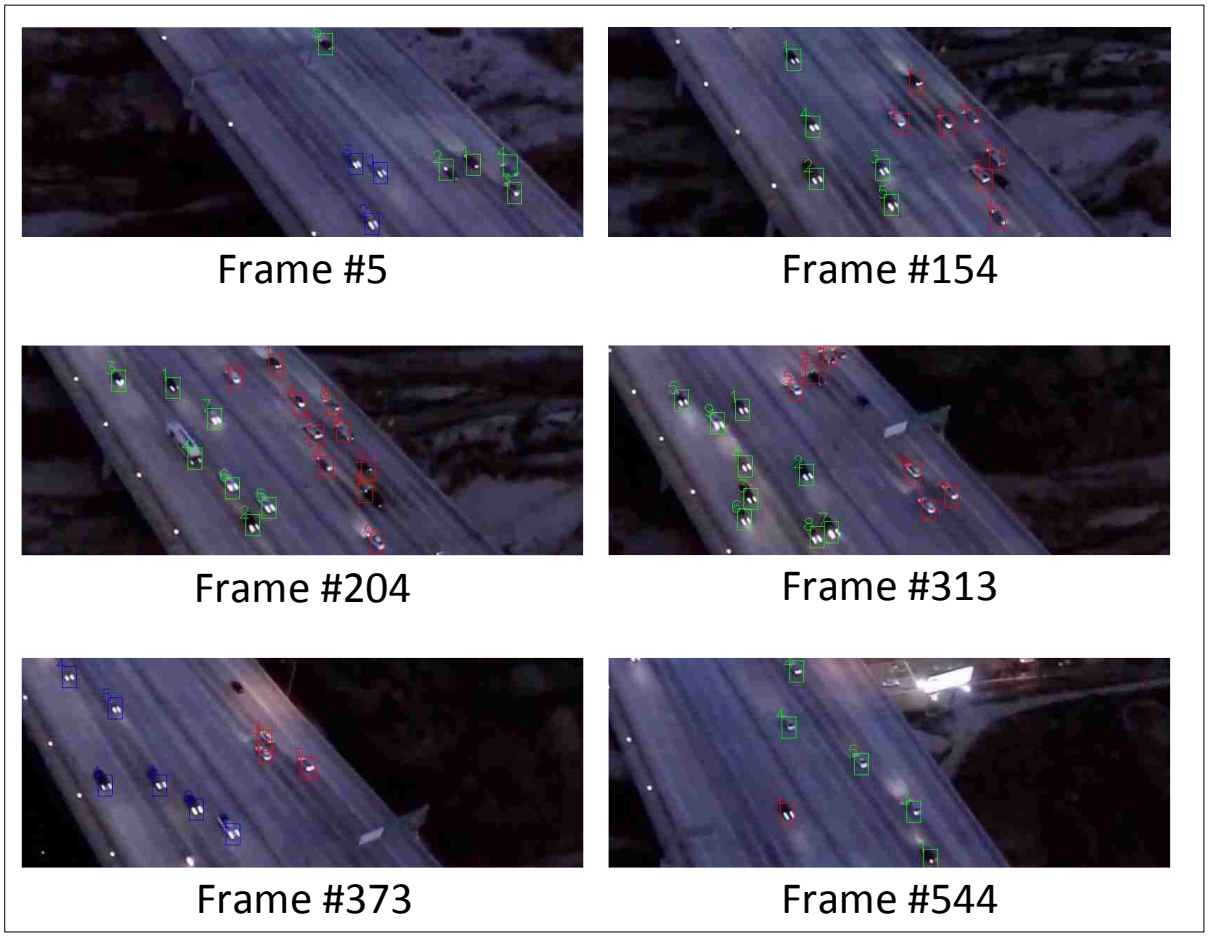


Figure 6-5 Selected frames showing the vehicle counting results on test aerial video #4, which was takin by a UAV flying along a freeway at night

Figure 6-6 Two snapshots showing the interest points extracted in daytime and nighttime

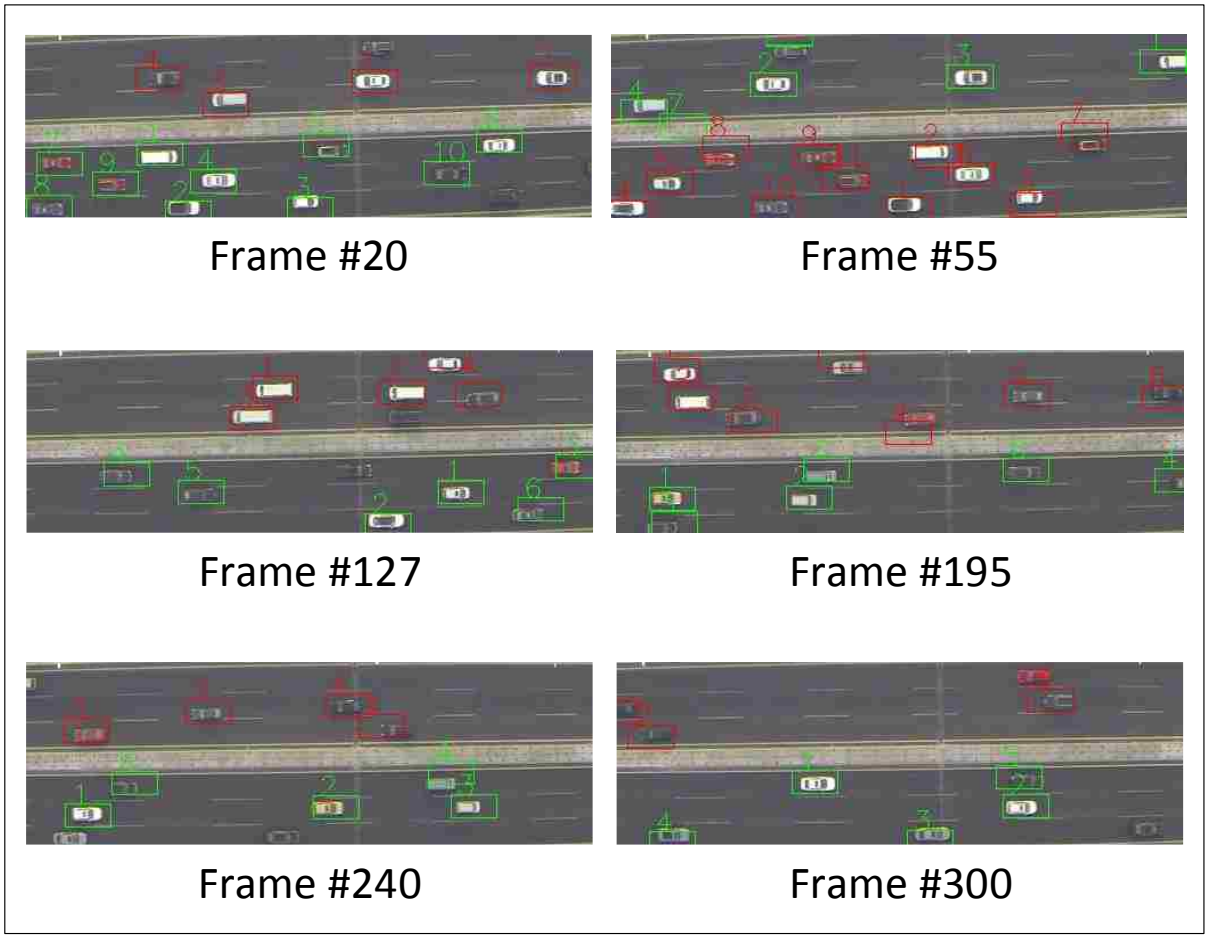


Figure 6-7 Selected frames showing the vehicle detection results on test video #5

Table 6-3 Estimated traffic flow parameters and performance evaluation of the motion-based approach on five test aerial videos

		Test Video #1	Test Video #2	Test Video #3	Test Video #4	Test Video #5
Total Frames		280	180	380	550	300
Dimension of ROI		500 × 220	500 × 175	500 × 170	800 × 300	500 × 160
UAV Movement		Move left with constant speed and vibration	Move right with constant speed and vibration	Move with changing speed and vibration	Move with drifting	Hover over the freeway segment
Video Background		Moving	Moving	Moving	Moving	Fixed
Time		Daytime	Daytime	Daytime	Nighttime	Daytime
Estimated Speed (mph)	Dir. A	54.6	46.4	48	56.7	40.8
	Dir. B	42.5	42.4	46.2	45.8	39.4
Estimated Density (pc/mi/lane)	Dir. A	25.4	31.1	25.8	30.1	35.5
	Dir. B	41.1	17	29.7	41.6	50.1
Estimated Volume (pc/h)	Dir. A	4177.6	4329.1	3715.2	6833.3	4344.1
	Dir. B	5222.9	2156.3	4114.4	7614.4	5914.8
Speed Estimation Accuracy (%)	Dir. A	97.634	97.300	94.995	95.475	96.525
	Dir. B	97.366	94.220	93.368	95.801	97.014
Count Estimation Accuracy (%)	Dir. A	82.607	86.612	81.449	90.234	85.470
	Dir. B	82.945	82.731	93.335	88.328	92.836
Processing Speed (fps)	Bi-direction	27.294	26.360	27.164	32.168	28.529

6.4 How Image Noises Affect System Performance

The overall quality of our test videos are pretty good, however, image noises can always have significant influences on video processing performances. Figure 6-8 shows the sample video frames with different levels of image noises added as well as the sample original video frame. Specifically, equidistributed noises with different intensities were added to the original frames. Here, the intensity of the noise added is measured by the YCbCr image space, which can be derived from RGB color space with the following equations:

$$\mathrm{Y} = 16 + (65.481 \times \mathrm{R} + 128.553 \times \mathrm{G} + 24.966 \times \mathrm{B}) \quad (19)$$

$$\mathrm{Cb} = 128 + (-37.797 \times \mathrm{R} - 74.203 \times \mathrm{G} + 112.0 \times \mathrm{B}) \quad (20)$$

$$\mathrm{Cr} = 128 + (112.0 \times \mathrm{R} - 93.786 \times \mathrm{G} - 18.214 \times \mathrm{B}). \quad (21)$$

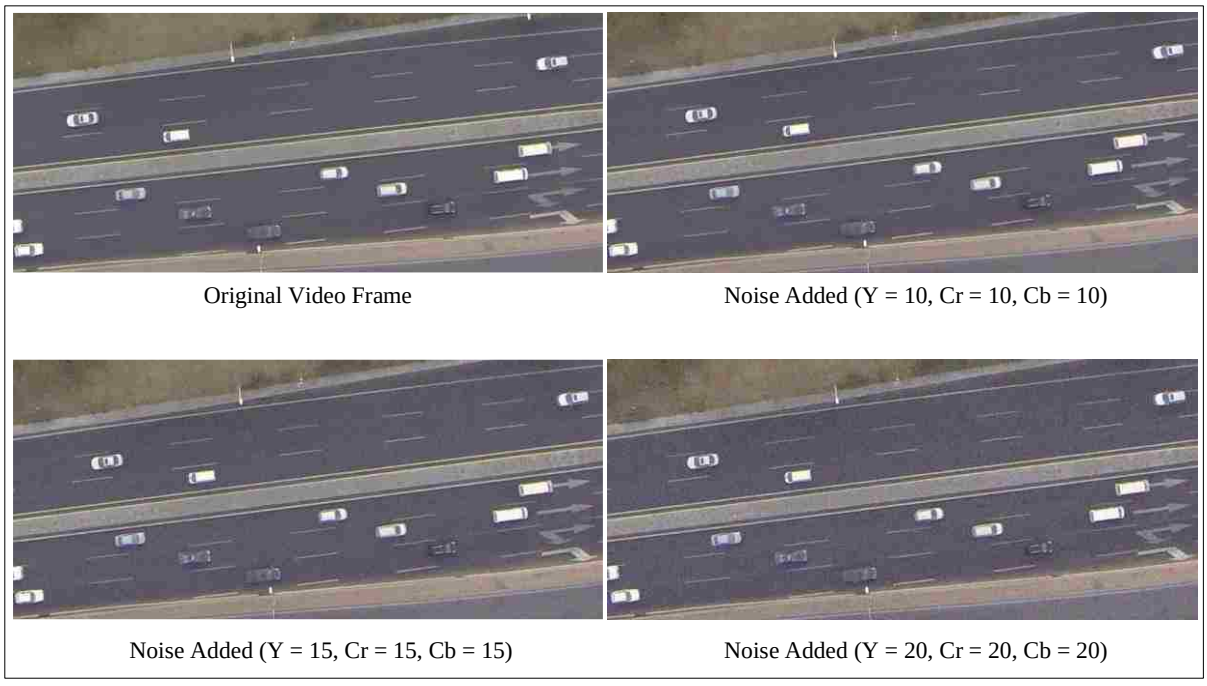


Figure 6-8 Different levels of image noises added to test video #1

After the image noises added, in order to quantitatively measure the signal-to-noise ratio, which can better show the quality of the video frames, a common method for image

signal-to-noise ratio estimation was implemented. In this signal-to-noise ratio estimation method, first of all, image should be converted to grayscale image. Then, small square windows are used to get the maximum local variance and minimum local variance of the pixel intensity values within the sliding window. If *S* denotes the largest local variance, which is regarded as the signal intensity, and *N* denotes the minimum local variance, which is regarded as the noise intensity, then the signal-to-noise ratio (SNR) can be estimated using the following equation:

$$\mathrm{SNR} = 10\log_{10}\left(\frac{S}{N}\right). \tag{22}$$

The SNR was calculated in randomly selected frames and then averaged. Results are shown in Table 6-4.

Table 6-4 Noise level and corresponding estimation accuracy

	Original	Video_N1	Video_N2	Video_N3
(Y,Cr,Cb)	NA	(10,10,10)	(15,15,15)	(20,20,20)
SNR	Very large	25 dB	19 dB	12 dB
Speed Accuracy in Dir. A (%)	97.634	96.937	97.488	96.829
Speed Accuracy in Dir. B (%)	97.366	96.7827	95.627	93.191
Count Accuracy in Dir. A (%)	82.607	80.952	80.018	73.381
Count Accuracy in Dir. B (%)	82.945	83.508	79.575	76.811

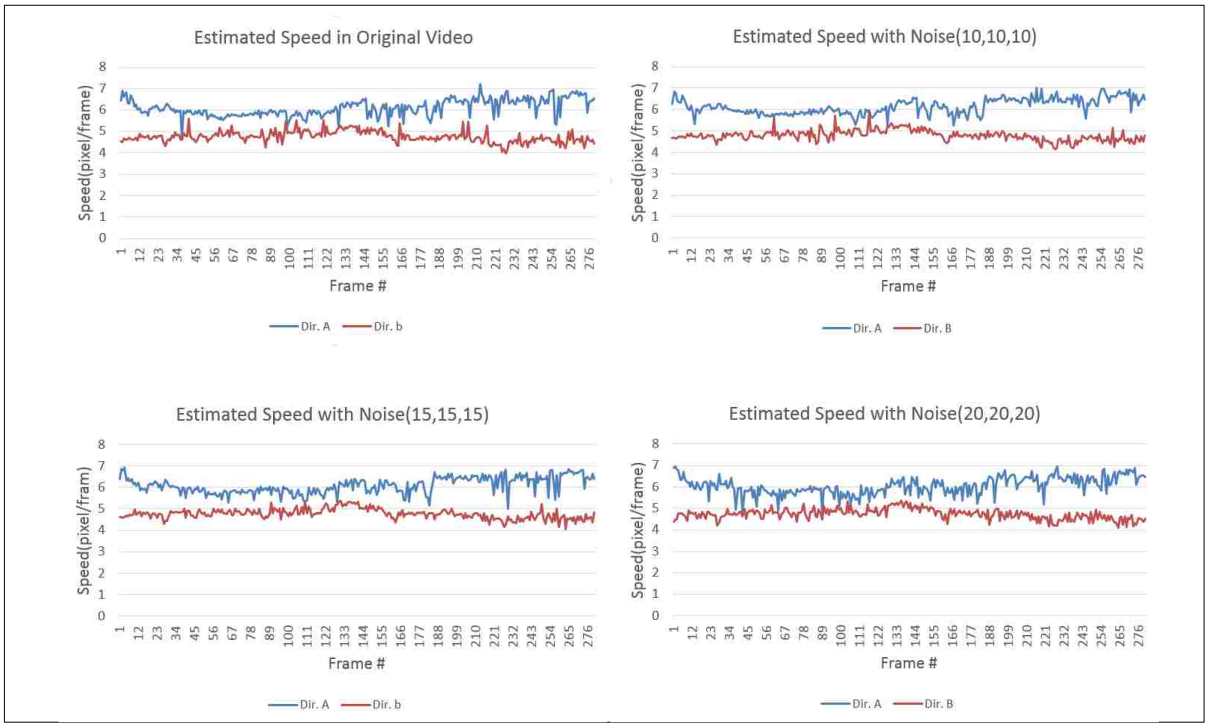


Figure 6-9 Comparison of bi-directional estimated speed with different level of noises

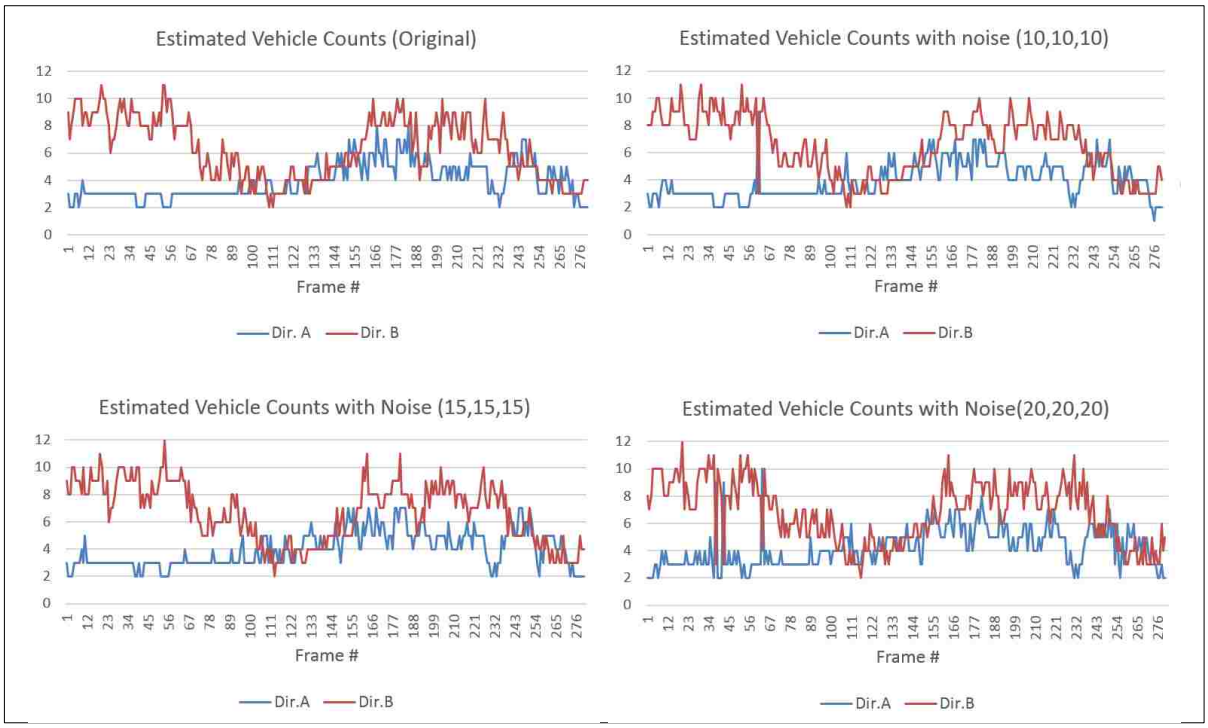


Figure 6-10 Comparison of bi-directional estimated vehicle counts with different level of noises

From Table 6-4 it can be seen, the estimation accuracy does not vary too much in the first three videos, i.e., the original video, Video_N1, and Video_N2. In Video_N3, which has

the highest noise level. We can see a sudden drop in count estimation accuracy as well as the speed estimation accuracy for Direction B. The fact is within our expectation that with higher level of noise added, the estimation accuracy decreases. However, the results already demonstrate the robustness of the proposed method. With moderate noises added, the estimation results pretty much stay at the same accuracy level as that of the original video. Moreover, Figure 6-9 and Figure 6-10 show the results of estimated bi-directional speeds and counts. From these two figures, we can see that the trends of all the corresponding curves are very similar, but as the noise increases, more outliers appear, resulting in the decrease in estimation accuracy.

Theoretically, the noises/disturbances would influence the results of the motion-based approach only if those noisy pixels were detected as Shi-Tomasi interest points and then tracked in next frame, which means the noisy pixels should have two properties: 1) they should be similar to corner points in terms of their gradients, and 2) they should keep appearing in the next frame with similar intensities and positions. However, in practice, these two properties are not easy to be satisfied at the same time, particularly in high-quality videos (i.e., videos with no noise or moderate noises).

6.5 Discussion on Real-time Processing Speed

As real-time traffic information is so important for traffic control or route guidance, the processing speed of our method is evaluated in this section. The experiments were conducted on a computer with an Intel i5-2310 CPU @ 2.9GHz processor and 6G of memory. Under current parameter settings, our system operates in real-time; the average processing speeds for the five videos were 27.294 fps, 26.360 fps, 27.164 fps, 32.168 fps, and 28.529 fps,

respectively (see Table 6-3). Considering our videos have a frame rate of 24 frames-per-second, real-time traffic parameter extraction is achieved.

The high processing speed of our motion-based approach results from key differences between our processing logic and framework when compared to previous ones discussed in the literature review. Most existing work has focused on trying to turn the problem into an image processing problem with a fixed background. Then, image registration is used to match features in two frames, but the overall process is quite time consuming and complicated. Each of the three algorithms that played an integral role in our methodology, i.e., optical flow, k-means clustering, and connected graph, have low computational complexities. We also observed that the processing speed of our method was mainly influenced by the number of interest points (i.e., the parameter *N*) and the accuracy level of the interest point extraction. The parameter *N* determines the elements in the array storing interest points and affects the processing speeds of each of the optical flow, k-means clustering and connected graph algorithms. Ultimately, one should attempt to make a balance between the processing speed and estimating accuracy. Our test results show that when *N* is set to 50, the average processing speed can reach up to 40 fps. When *N* is set to 200, the average processing speed decreased to approximately 15 fps on our machine. Clearly processing speed can be improved by using higher-performance computers, thus better enabling real-time traffic parameter extraction from aerial videos.

Chapter 7 Experimental Results of the Detection-based Approach

7.1 Detector Performance Evaluation

The parameters of the built vehicle detector include scale factor, the number of neighbors each candidate rectangle should have to retain, minimum possible vehicle size, maximum possible vehicle size, etc. In total 1200 different combinations of parameters were tested on a 240-frame aerial video with 800 × 400 pixels and 24 fps (frame-per-second) frame rate. Precision, recall, F-measure, and area-under-curve (AUC) of the precision-recall curve, are commonly used in evaluating detectors and classifiers. Precision, recall, and F-measure are defined as in the following equations,

$$precision = \frac{TP}{TP+FP} \tag{23}$$

$$recall = \frac{TP}{TP+FN} \tag{24}$$

$$F\text{-}measure = \frac{2\times precision\times recall}{precision+recall} \tag{25}$$

where TP is short for “true positive”, FP is for “false positive”, and FN is for “false negative”; F-measure denotes the average of the precision and recall; AUC is calculated using integration. Precision, recall, F-measure and AUC all reach their best value at 1 and worst at 0.

As supporting real-time performance is one of the most major objectives of our system, we did not test the performance of a standalone MLP neural network detector since it is far from a real-time detector. However, standalone CC detector and CC+MLP detector are compared in our experiment in order to show the improvement in reducing false-positives with the combination.

Figure 7-1 and 7-2 show the detected windows of CC detector and CC+MLP detector, respectively. From these two figures we can see the effectiveness of the combination of CC and MLP very clearly. As aforementioned, CC keeps almost all true-positives (vehicles) in those frames, at the same time it largely reduces the search-space for MLP from sliding the whole frame to only checking those remaining candidate rectangles. In Figure 7-2, it can be seen our MLP detector eliminates almost all remaining false-positives while keeping true-positives as the final strong classifier.

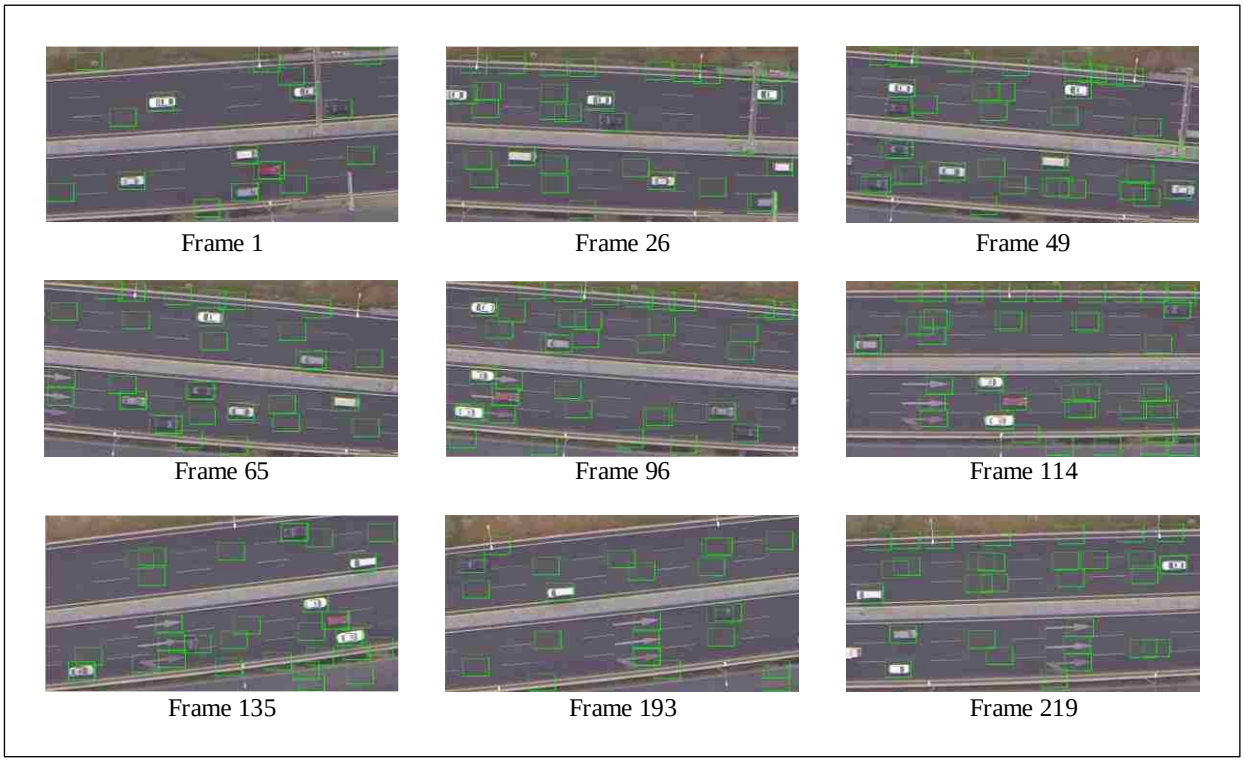


Figure 7-1 Detection of pre-MLP candidates

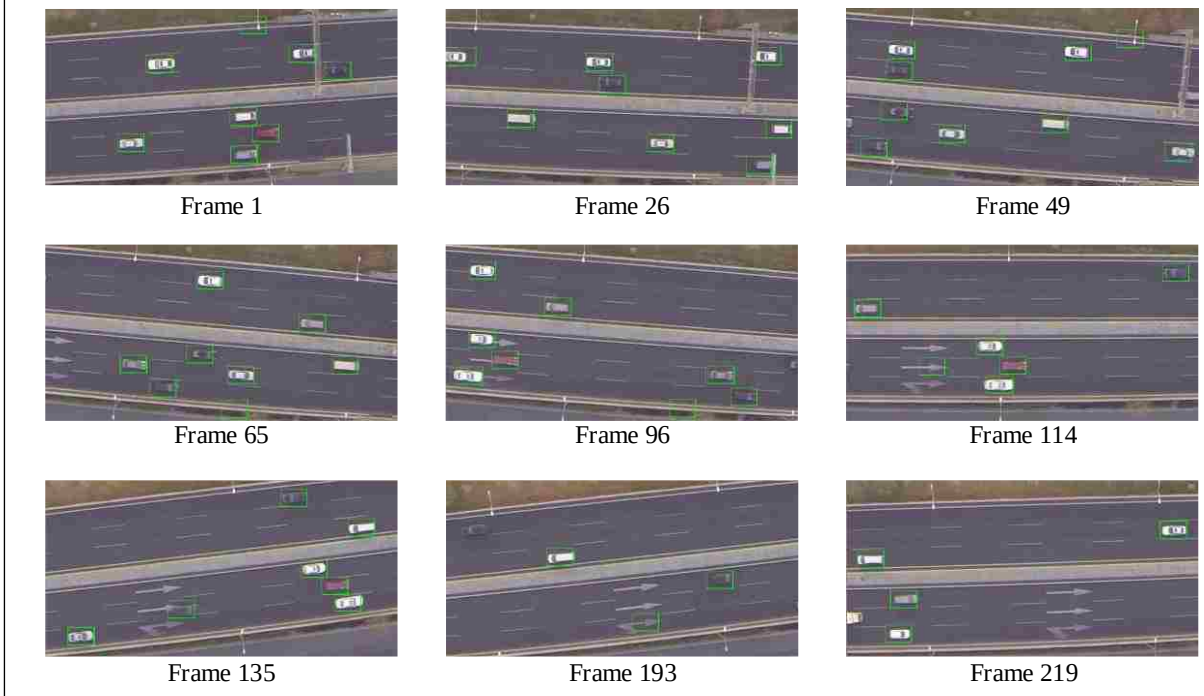


Figure 7-2 Detection of post-MLP results

As aforementioned, 1200 different combinations of parameters were tested. Figure 7-3 shows the precision-recall curve of both CC and CC+MLP detectors with these 1200 parameter combinations. Each point on the curve corresponds to a precision value and recall value of a specific combination. Intuitively, the curve of CC+MLP detector is above and on the right of CC, which means in general the CC+MLP detector has both higher precision values and recall values than standalone CC. This also verifies the detection results we observed from Figure 7-1 and 7-2.

Detailed statistical results are presented in Table 7-1. It can be seen the average precision, recall and F-measure of our final detector are all around or above 0.7. With parameters properly set, the precision and recall can reach 0.992 and 0.960, and F-measure can also reach 0.821. The AUC of CC+MLP's precision-recall curve is 0.871. All the statistics show that our combined vehicle detector performs very well. We believe that with more training samples collected and incorporated in the training process in the future, our

detector will become more and more robust and accurate in UAV-based vehicle detection. The performance statistics of standalone CC are also generated and displayed in Table 7-1. Interestingly, even if the precision and F-measure of standalone CC are much smaller than those of the combined detector, the recall is a little bit larger. This is reasonable and exactly what we expected: as the first-stage weak classifier, Haar cascaded classifier detects most true-positives though still detects some false-positives. The MLP acts as the final stage in the combined detector, so the candidate input windows to it are all from the output of first-stage CC's. Thus, the recall of CC+MLP must be smaller than that of CC. However, a good final detector should turn out to keep the recall still very close to that of the standalone CC, which would mean that it keeps most of the correctly detected vehicles. From the Table, we see actually our final detector satisfies this requirement, i.e., it does keep the recall very close to that of CC.

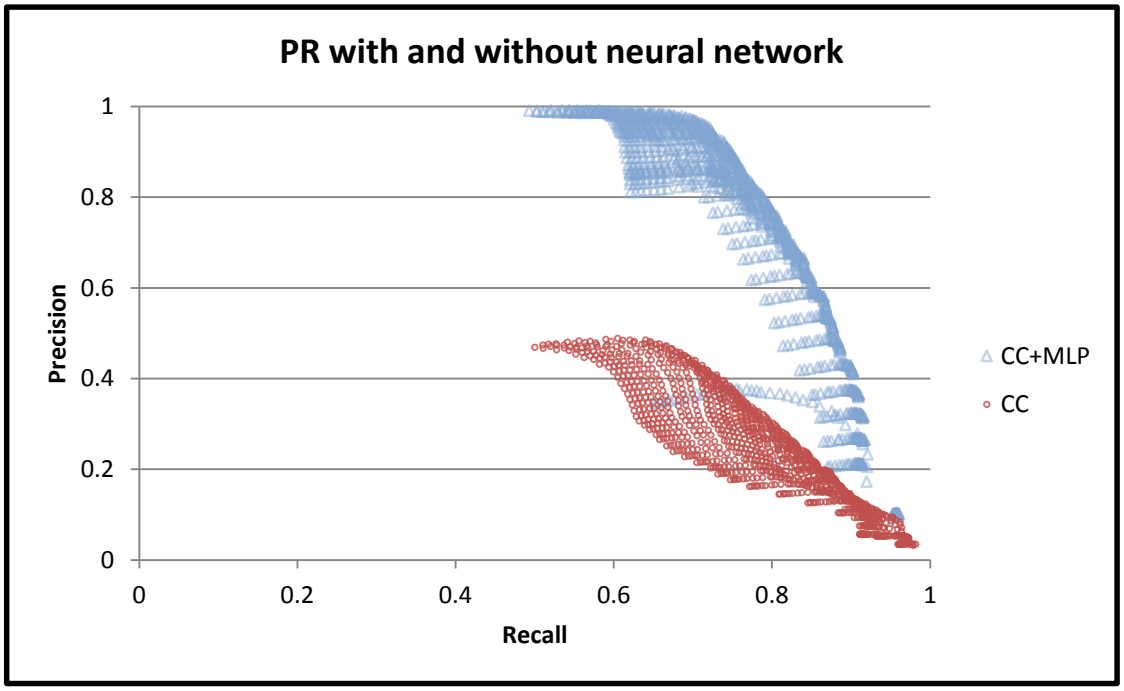


Figure 7-3 Plots showing the detection results of CC detector and CC+MLP detector with 1200 different combinations of parameters

Table 7-1 Performance Evaluation Results of CC+MLP and Standalone CC

Classifier	CC+MLP			CC		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Avg.	0.723	0.758	0.692	0.241	0.794	0.342
Std.	0.255	0.106	0.148	0.126	0.109	0.136
Max.	0.992	0.960	0.821	0.489	0.981	0.553
Min.	0.095	0.493	0.173	0.032	0.500	0.062
AUC		0.871			0.393	

7.2 Traffic Flow Parameter Estimation Analysis

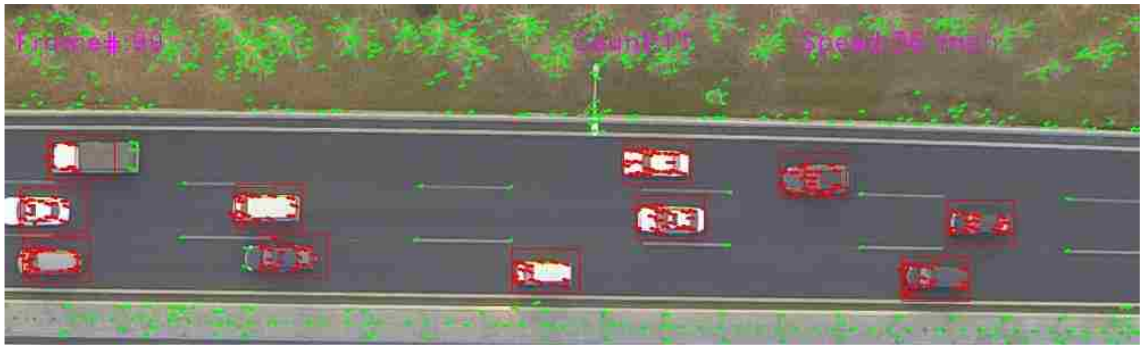
As described in Section 4.3, traffic flow parameter estimation using our detection-based approach also makes use of KLT tracker to estimate vehicles' motion but in a different way and order from the motion-based approach. The detection-based approach obtains the average motion-vectors inside and outside the detected rectangles separately. Thus, with motion-vector subtraction and reference markings with known lengths such as lane markings, both vehicle speed and vehicle count within a road segment can be estimated. In this way, the three basic traffic flow parameters, i.e., speed, density, and volume can be extracted from aerial videos.

In our experiment, in total about thirty minutes' video clips were tested using the combined detector and this detection-based approach. Very good estimation results have been observed. Two representative 400-frame video footages were manually examined in detail to evaluate the performance of the approach. Video #7 is a video clip taken by a UAV moving over a freeway segment, monitoring a three-lane freeway with smooth traffic flow. Video #8 is another video clip taken with UAV vibration over an urban arterial where the traffic is really heavy. With video #8, we are going to show the ability of the detection-based approach to estimate traffic flow parameters in congested condition.

Figure 7-4 shows some randomly selected sample frames in video #7 with detection and estimation results. Multiple-vehicle detection and tracking are challenging tasks, especially when the video background is not stable. However, in our framework, we not only detect and track multiple vehicles at the same time, but also extract useful traffic flow parameters. Figure 7-5 displays the instantaneous traffic flow parameters extracted from video #7. In addition, the aggregated values are displayed in Table 7-2, which is more meaningful considering that the instantaneous results in one frame can be either not accurate or not representative. The estimated average speed in video #7 is 32.7 mph, the estimated density is 59.4 pc/mi/lane, and the estimated volume is 5855.9 pc/hour for all three lanes.

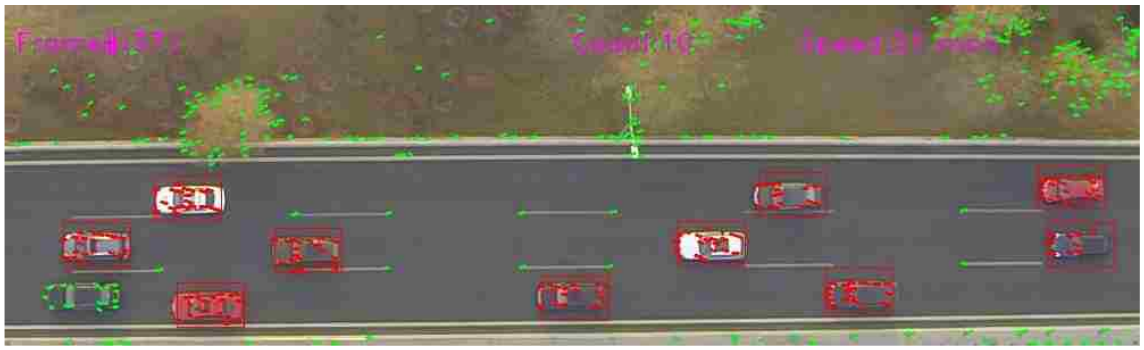
Figure 7-6 shows the randomly selected frames in video #8 with detection and estimation results. This case study demonstrates the power of detection-based approach: in congested traffic conditions where vehicles are in very low speed or even stationary, it is very difficult for traditional methods as well as the aforementioned motion-based method to detect and track all the vehicles, let alone estimate their motion at the same time. However, the proposed detection-based approach does not use any of the motion or video background information to detect/track vehicles. Instead, it just detects the vehicles based on their appearance patterns and then tracks them based on detection results. Therefore, the vehicle motion has little influence on the detection-based approach. This makes it possible for the approach to extract traffic flow parameters in congested condition. From Figure 7-6, 7-7 and Table 7-2 it can be observed that very good results were obtained. The average estimated speed in video #8 is only 2.1 mph; the density is high, reaching 113.7 pc/mi/lane; and the volume is low, which is 111.6 pc/hour.

Frame#: 6 Count: 10 Speed: 32 mph



Frame#: 99 Count: 11 Speed: 36 mph

Frame#: 255 Count: 10 Speed: 34 mph



Frame#: 371 Count: 10 Speed: 31 mph

Figure 7-4 Selected sample frames showing the vehicle detection and parameter estimation results in uncongested traffic condition

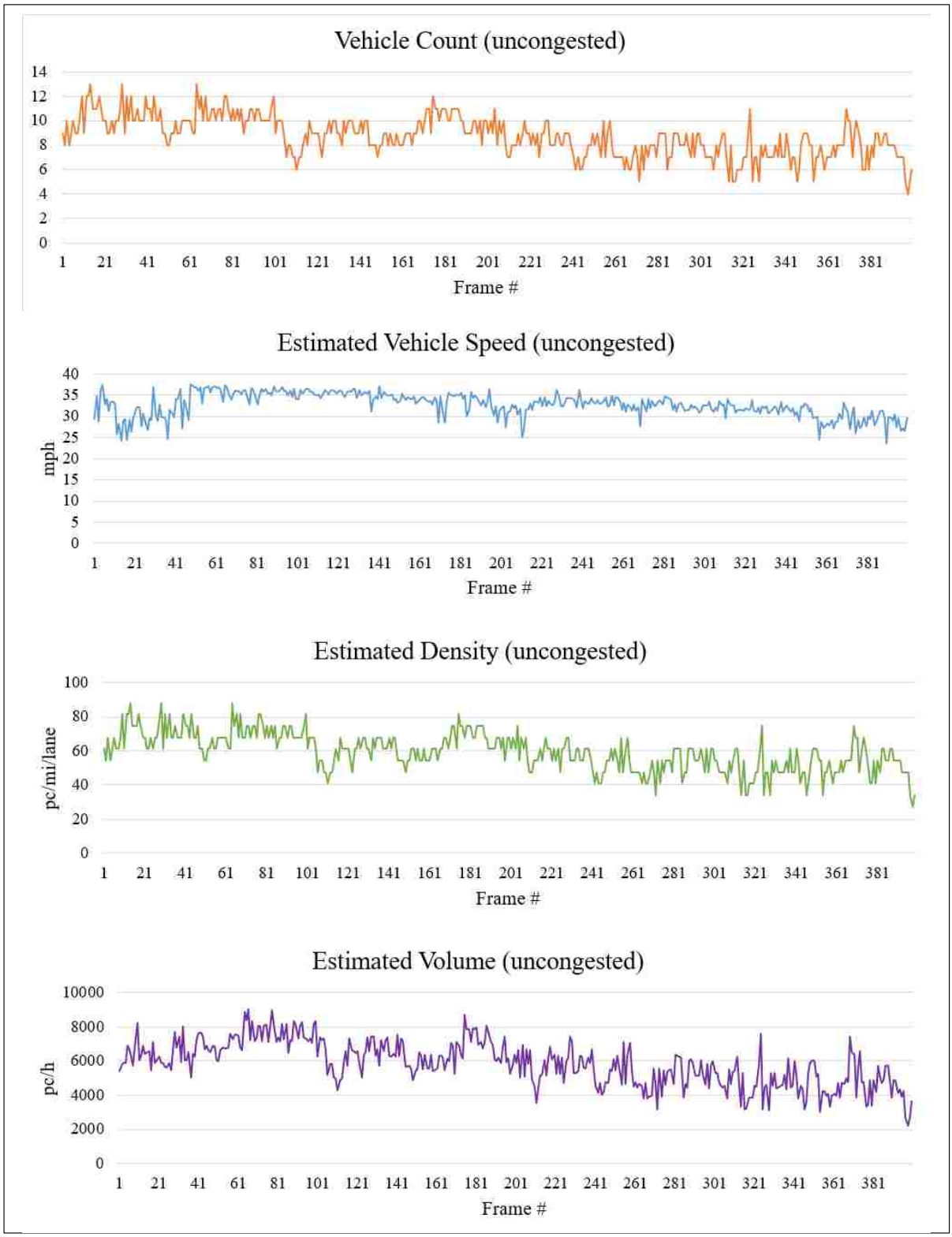


Figure 7-5 Estimated instantaneous traffic flow parameters for test video #7 with uncongested traffic

Frame#: 5 Count: 15 Speed: 4 mph

Frame#: 223 Count: 33 Speed: 1 mph

Frame#: 325 Count: 44 Speed: 1 mph

Frame#: 398 Count: 43 Speed: 0 mph

Figure 7-6 Selected sample frames showing the vehicle detection and parameter estimation results in congested condition

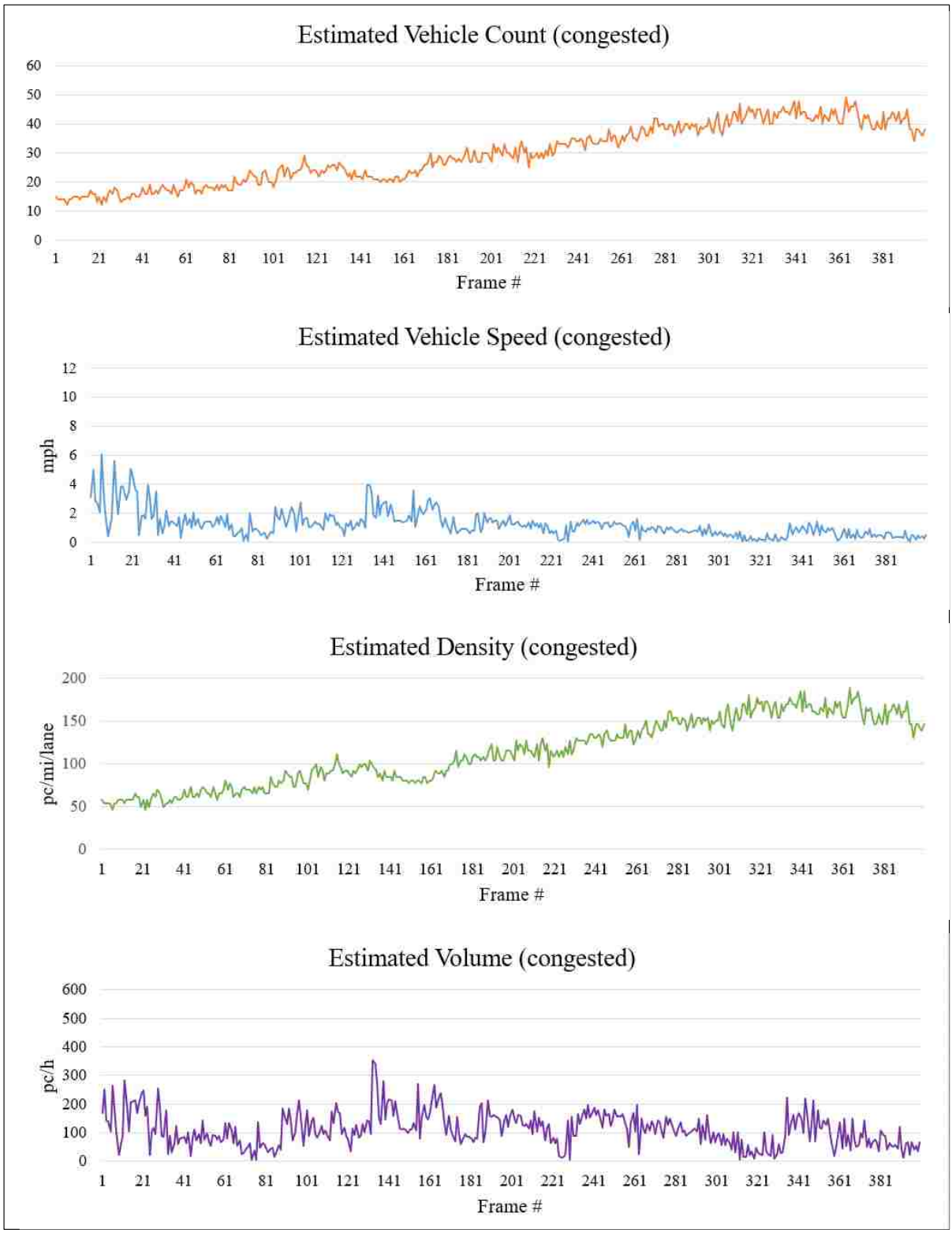


Figure 7-7 Estimated instantaneous traffic flow parameters for test video #8 with congested traffic

Table 7-2 Estimated Traffic flow Parameters and Performance Evaluation of the Detection-based Approach on Two test Aerial Videos

	Test Video #7	Test Video #8
Total Frames	400	400
Dimension of ROI	1000 × 300	1220 × 348
UAV Movement	Move along a freeway segment with little vibration	Move along an arterial segment with noticeable vibration
Estimated Speed (mph)	32.7	2.1
Estimated Count (per frame)	8.7	29.5
Estimated Density (pc/mi/lane)	59.4	113.7
Estimated Volume (pc/h)	5855.9	111.6
Speed Estimation Accuracy (%)	91.079	90.067
Count Estimation Accuracy (%)	84.193	88.276
Processing Speed (fps)	31.082	31.265

As aforementioned, vehicle speed and vehicle count are the two straightforward metrics used to evaluate the estimation accuracy. To get the ground truth of vehicle speed, we use an on-screen pixel measurement tool to manually measure the distances vehicles move between consecutive frames; to get the ground truth of vehicle count, we manually go through the video frame by frame and count the vehicles in each frame.

The speed estimation accuracy in video #7 and video #8 are 91.079 % and 90.067 %; the count estimation accuracy in the two videos are 84.193 % and 88.276 %. In the detection-based approach, the vehicle count estimation is directly determined by the detector performance, while the speed estimation is influenced by both the detector performance and the motion estimation performance. From Table 7-2 we see that the speed estimation accuracy is generally pretty high, but is lower than the motion-based approach. This can be explained as follows: in the motion-based approach, there is an assumption that vehicles in

one group share similar motion, so even though some non-vehicles are classified as vehicles, they still have the similar motion with real vehicles, and in this way the non-vehicles influence little on the aggregated speed estimation. However, in the detection-based approach, non-vehicles do not necessarily have similar motion with vehicles. More likely, they are from background and have very different motions. Hence, when they are counted as vehicles, their influence on speed estimation is much bigger than that in the motion-based approach, which leads to a relatively lower speed estimation accuracy.

As for the vehicle count, an average of over 86 % estimation accuracy was achieved. The estimation in aerial video #8 is more accurate than that in video #7 with the same vehicle detector we built. This can be interpreted as the influence by the increase in total number of vehicles. In video #8, the traffic is congested while in video #7 it is uncongested. Also, from the table we can see the average number of vehicles in video #8 is 29.5 while that in video #7 is only 8.7. Since it is the same detector that is used in the estimation from these two videos, when the detector generates the same number of false-positives or false-negatives, the error rate in video #7 is more likely to be larger than that in video #8.

7.3 How Image Noises Affect System Performance

Similar to the experiment on testing the motion-based approach, we are testing how image noises could affect the system performance of the detection-based approach. Figure 7-8 shows the sample video frames with different levels of image noises added as well as the sample frame of the original video. Specifically, equidistributed noises with different intensities were added to the original frame. Here, the intensity of the noise added is measured by the YCbCr image space, which can be derived from RGB color space with the Eq. (19) – (21).

Original Video Frame

Noise Added (Y = 20, Cr = 20, Cb = 20)

Noise Added (Y = 20, Cr = 20, Cb = 20)

Noise Added (Y = 20, Cr = 20, Cb = 20)

Figure 7-8 Different levels of image noises added to test video #7

After the image noises added, in order to measure the signal-to-noise ratio, which quantitatively shows the quality of the video frames, a common method for image signal-to-noise ratio estimation was implemented. In this signal-to-noise ratio estimation method, first of all, image is first converted to grayscale image. Then, small square windows are used to get the maximum local variance and minimum local variance of the pixel intensity values within the sliding window. If *S* denotes the largest local variance, which is regarded as the signal intensity, and *N* denotes the minimum local variance, which is regarded as the noise intensity, then the signal-to-noise ratio (SNR) can be estimated using Eq. (22). The SNR was calculated in randomly selected frames and then averaged. Results are shown in Table 7-3.

Table 7-3 Noise level and corresponding estimation accuracy

	Original	Video_N1*	Video_N2*	Video_N3*
(Y,Cr,Cb)	NA	(10,10,10)	(15,15,15)	(20,20,20)
SNR	Very large	25 dB	19 dB	12 dB
Speed Accuracy(%)	91.079	86.725	84.891	83.686
Count Accuracy(%)	84.165	78.046	74.048	68.796

From Table 7-3, it can be seen both speed estimation accuracy and count estimation accuracy decrease as the video noise increases. Generally, compared to the motion-based approach, the accuracy reduction here is faster such as that the count estimation accuracy decreases from 84 % to 68 % from original video to Video_N3*. This is considered reasonable: for speed estimation, in the detection-based approach, when there are more false-positives, background motion in some part of the image will be aggregated into traffic motion; likewise, when there are more false-negatives, some vehicles' motion will be aggregated into background motion. In either way, the speed estimation accuracy will be

influenced more than the motion-based approach. Count estimation accuracy mainly depends on the vehicle detector performance. We can further improve the robustness to noise by training a stronger detector with more samples collected.

7.4 Discussion on Real-time Processing Speed

As real-time traffic information is so important for traffic control or route guidance, the processing speed of our method is evaluated in this section. The experiments were conducted on a computer with an Intel i5-2310 CPU @ 2.9GHz processor and 6G of memory. Under current parameter settings, our system operates in real-time; the average processing speeds for the five videos were 31.082 fps, 31.265 fps, respectively (see Table 7-2). Considering our videos have a frame rate of 24 frames-per-second, real-time traffic parameter extraction is achieved.

The high processing speed of our detection-based approach results from key differences between our processing logic and framework when compared to previous ones discussed in the literature review. Most existing work has focused on trying to turn the problem into an image processing problem with a fixed background. Then, image registration is used to match features in two frames, but the overall process is quite time consuming and complicated. In addition, most existing aerial video-based vehicle detector are more computationally expensive than ours. In contrast, the two components of our proposed detection-based approach, i.e., combined cascaded classifier and KLT tracking method, are very efficient in both detection and tracking. The combined cascaded classifier takes advantage of the fast processing speed of cascaded classifiers and the high detection rate of neural network, thereby resulting in fast vehicle detection; KLT method tracks low-cost point features, making the following traffic speed estimation process very efficient as well. In the detection-based approach, the number of interest points does not influence processing speed

significantly since the only thing we do with the number of interest points is when we average the motion of vehicles and background. It is not like in the motion-based approach, in which every key algorithm takes the interest points as input. That is to say, based on this framework, the vehicle detector plays the key role in fast processing speed though the KLT method also influences the efficiency.

Chapter 8 Conclusion and Future Work

8.1 Conclusion

In this research, we proposed a novel framework that is composed of two complementary approaches for estimating traffic flow parameters, i.e., speed, density, and volume, from aerial videos. These two approaches both make use of optical flow information, machine learning methods, traffic flow theory, and reference markings to extract traffic flow parameters from aerial videos with moving background. The proposed framework works well in a variety of challenging scenarios. The system is very robust and has been proved to be not sensitive to UAV movements, traffic conditions (uncongested or congested), and time of day (daytime or nighttime).

The motion-based approach identifies traffic streams and extracts bi-directional traffic flow parameters. This approach includes four consecutive steps. The first step is Shi-Tomasi interest point detection and Kanade-Lucas optical flow-based tracking across frames. The second step clusters optical flow vectors via the k-means algorithm based on their speed and direction. The third step groups interest points in each travel direction and calculates the motion of each individual vehicle. The final step calculates speed, densities, and volumes for the two travel directions. Experiments on five datasets from aerial videos show that the proposed method yields about 96% and 87% accuracy in estimating average speed and vehicle count of the two traffic streams considered, respectively.

The detection-based approach is based on a well-trained vehicle detector and optical flow information from top-view perspective. In the first stage of training the detector, Haar cascaded classifiers are trained using Haar-like features, which are used to reduce the image search-space from the whole image to a limited number of sub-windows. In the second stage, multi-layer perceptron neural network classifier is trained as the final stage of the combined

cascaded classifier, examining the remaining candidate windows to determine if they are vehicles or not. Two Haar cascaded classifiers and two multi-layer perceptron neural networks are trained in the framework to detect light-color and dark-color vehicles separately to achieve a higher detection rate with the same size of training samples. The combined cascaded classifier takes advantage of the fast processing speed of cascaded classifier and the high detection rate of multi-layer perceptron. With vehicles detected in each frame, KLT method is then applied to estimate the vehicle motion as well as the background motion. Hence, average vehicle speed can be obtained by motion-vector subtraction. With an appropriate reference marking, actual traffic speed, density and volume can then be calculated. In our experiment, the speed and count estimation accuracy reaches over 90% and 86%, respectively.

The motion-based approach and detection-based approach demonstrates two complementary ideas in aerial video-based traffic flow parameter estimation. Both of them actually use motion information to handle the moving background problem in UAV video processing. However, the way to detect vehicles as well as get vehicle counts are different. The motion-based approach first extracts the motion information all over the current frame, then the vehicle detection and counting is conducted based on unsupervised learning methods, i.e., k-means clustering and connected graph. In contrast, detection-based approach first detect vehicles using supervised learning method, i.e., combined cascaded classifier, and then extract and aggregate motion information based on detection results. Therefore, the motion-based approach does not really deal with the pattern of the vehicle. The processing is all based on traffic and background motion, so it works well for both daytime and nighttime, but does not work in heavy congested condition. The detection-based approach detect vehicles based on pattern thus it works well in both uncongested and congested conditions, but it does not work at night since the positive samples were all collected in daytime.

The proposed framework is designed to enable real-time operation for transportation management and traffic monitoring. The two main approaches both runs in a real-time manner even on a normal desktop computer. With higher configuration, it is reasonable to say our system will be even more efficient and is able to support more complicated tasks.

8.2 Future Work

Future work could be focusing on the follows issues. In the first place, for the motion-based approach, our current method works well for free-flow and moderately congested traffic flow conditions because the motion criteria depend on similar movement of both background and traffic interest points. Similarly, our method works well for traffic on straight road segments. Testing our method on heavily congested traffic conditions and curved road segments, and adjusting it to improve performance would also be insightful. Then, our algorithm sometimes recognizes trucks, buses, and other large/heavy vehicles as multiple passenger cars. Future work may investigate to improve accuracy for estimation of large vehicles and to improve the overall performances of our method.

In the second place, for the detection-based approach, the number of training samples is still relatively small. At this moment we consider it reasonable since the positive samples of top-view vehicles from UAV perspective is still rare, but in the future as more aerial videos being collected, we are going to train more rigorous combined cascaded classifiers using our model. We can even try training deep neural networks rather than the regular neural network built in current research, especially given the fact that the deep learning methods have made great progresses recently. Actually our detection approach demonstrates an idea that with a well-trained vehicle detector and the KLT method, traffic flow parameters can be obtained. Thus, we are going to build more vehicle detectors using different methods, then test and compare them to see how each of them perform in extracting traffic flow parameters.

Last but not least, for our framework incorporating two approaches as a whole, even though we have concluded that each approach has its own advantages, e.g., the detection-based approach works better for congested conditions and motion-based approach works better for different lighting conditions, we still need to figure out a way for the system to automatically determine which approach should be used in a given situation. That is to say, when the system is really used in traffic monitoring and control tasks, the UAV itself does not know what the scenario is in advance, so effort is needed to develop a method for UAV to know which approach in the framework is more appropriate in a certain scenario.

BIBLIOGRAPHY

[1] G. Zhang, R. Avery, and Y. Wang, "Video-based vehicle detection and classification system for real-time traffic data collection using uncalibrated video cameras," Trans. Res. Rec., J. Transp. Res. Board, vol. 1993, pp.138 -147, 2007.

[2] Y. Wang and N. Nihan, "Freeway traffic speed estimation with single-loop outputs," Trans. Res. Rec., J. Transp. Res. Board, vol. 1727, pp. 120 -126, 2000.

[3] C. Kyoungah, I. Lee, J. Hong, T. Oh, and S. W. Shin, "Developing a UAV-based rapid mapping system for emergency response, " in Proc. SPIE Defense, Security, and Sensing, Orlando, FL, 2009, pp. 733209 -733209.

[4] B. Coifman , M. McCord , R. Mishalani, and K. Redmill, "Surface transportation surveillance from unmanned aerial vehicles," in Proc. 83rd Annu. Meeting Transp. Res. Board, Washington, D.C., 2004.

[5] S. Srinivasan , H. Latchman , J. Shea , T. Wong, and J. McNair, "Airborne traffic surveillance systems: Video surveillance of highway traffic," in Proc. ACM 2nd Int. Workshop Video Surveillance Sens. Netw., New York, NY, 2004, pp.131 -135.

[6] H. Zhou, H. Kong, L. Wei, D. Creighton, and S. Nahavandi, "Efficient road detection and tracking for unmanned aerial vehicle," IEEE Trans. Intell. Transp. Syst., vol. 16, no. 1, pp. 297 – 309, Feb. 2015.

[7] M. McCord, Y. Yang, Z. Jiang, B. Coifman, and P. Goel, "Estimating Annual Average Daily Traffic from Satellite Imagery and Air Photos: Empirical Results," Trans. Res. Rec., J. Transp. Res. Board, vol. 1855, pp. 136 -142, 2003.

[8] A. Angel, M. Hickman, P. Mirchandani, and D. Chandnani, "Methods of analyzing traffic imagery collected from aerial platforms," IEEE Trans. on Intell. Transp. Syst., vol. 4, no. 2, pp. 99 -107, 2003.

[9] F. Yamazaki , L. Wen, and T. Vu, "Vehicle extraction and speed detection from digital aerial images," in Proc. IEEE Int. Conf. Geosci. Remote. Sensing. Symp., Boston, MA, 2008, pp. III - 1334 -III – 1337.

[10] C. K. Toth and D. Grejner-Brzezinska, "Extracting dynamic spatial data from airborne imaging sensors to support traffic flow estimation," ISPRS J. Photogramm. Remote Sens., vol. 61, no. 3/4, pp.137 -148, Dec. 2006.

[11] A. C. Shastry and R. A. Schowengerdt, "Airborne video registration and traffic-flow parameter estimation," IEEE Trans. Intell. Transp. Syst., vol. 6, no. 4, pp. 391 -405, Dec. 2005.

[12] J. Shi and C. Tomasi, "Good features to track," in Proc. IEEE Conf. Comput. Vision Patt. Recog., Seattle, WA, 1994, pp. 593 -600.

[13] A. Puri, K. Valavanis, and M. Kontitsis, "Generating traffic statistical profiles using unmanned helicopter-based video," in IEEE Int. Conf. Robot. Auto., Roma, 2007, pp. 870 -876.

[14] W. Yao , S. Hinz, and U. Stilla, "Automatic vehicle extraction from airborne LiDAR data of urban areas aided by geodesic morphology," Patt. Recog. Lett., vol. 31, no. 10, pp.1100 -1108, July 2010.

[15] L. Eikvil , L. Aurdal, and H. Koren, "Classification-based vehicle detection in high-resolution satellite images," ISPRS J. Photogramm. Remote Sens., vol. 64, no. 1, pp.65 - 72, Jan. 2009.

[16] Q. Yu and G. Medioni, "Motion pattern interpretation and detection for tracking moving vehicles in airborne video," in Proc. IEEE Conf. Comput. Vision Patt. Recog., Miami, FL, 2009, pp.2671 -2678.

[17] X. Cao , C. Wu , J. Lan , P. Yan, and X. Li, "Vehicle detection and motion analysis in low-altitude airborne video under urban environment," IEEE Trans. Circuits Syst. Video Technol., vol. 21, no. 10, pp.1522 -1533, July 2011.

[18] Y. Lin and S. Saripalli, "Road detection from aerial imagery," in Proc. IEEE Int. Conf. Robot. Autom., Saint Paul, MN, 2012, pp. 3588 -3593.

[19] Z. Kim, "Realtime road detection by learning from one example," in Proc. IEEE Workshop Appl. Comput. Vision, 2005, pp. 455-460.

[20] R. Pless and D. Jurgens, "Road extraction from motion cues in aerial video," in Proc. ACM Conf. Geo. Info. Syst., 2004, pp. 31 -38.

[21] S. Hinz , J. Leitloff, and U. Stilla, "Context-supported vehicle detection in optical satellite images of urban areas," in Proc. IEEE Int. Conf. Geosci. Remote. Sensing. Symp, 2005, pp. 2937 -2941.

[22] H.-Y. Cheng , C.-C. Weng, and Y. Chen, "Vehicle detection in aerial surveillance using dynamic bayesian networks," IEEE Trans. Image Process., vol. 21, no. 4, pp. 2152 - 2159, Mar. 2012.

[23] X. Cao , C. Wu , P. Yan, and X. Li, "Linear SVM classification using boosting HOG features for vehicle detection in low-altitude airborne videos," IEEE Int. Conf. Image Process., Brussels, 2011, pp. 2421 -2424.

[24] T. Zhao and R. Nevatia, "Car detection in low resolution aerial images," Image Vis. Comput., vol. 21, no. 8, pp. 693 -703, Aug., 2003.

[25] T. P. Breckon , S. E. Barnes , M. L. Eichner, and K. Wahren, "Autonomous real-time vehicle detection from a medium-level UAV," in Proc. 24th Int. Unmanned Air Vehicle Syst. Conf., 2009, pp. 29 -37.

[26] A. Gaszczak, T. P. Breckon, and J. Han, "Real-time people and vehicle detection from UAV imagery," in Proc.SPIE Elec. Imag. Int. Soc. Opt. Photo., 2011, pp. 78780B - 78780B.

[27] K. Khaled , C. Benjamin , P. Claude, and V. Pascal, "A vision algorithm for dynamic detection of moving vehicles with a UAV," in Proc. IEEE Int. Conf. Robot. Automat., 2005 pp.1878 -1883.

[28] X. Cao, J. Lan, P. Yan, and X. Li, "Vehicle detection and tracking in airborne videos by multi-motion layer analysis," Mach. Vision Appl., vol. 23, no. 5, pp.921 -935, Sept., 2012.

[29] X. Cao, C. Gao, J. Lan, Y. Yuan, and P. Van, "Ego motion guided particle filter for vehicle tracking in airborne videos," Neurocomputing, vol. 124, pp. 168-177, Jan., 2014.

[30] H. Yalcin , M. Herbert , R. Collins, and M. J. Black, "A flow-based approach to vehicle detection and background mosaicking in airborne video," in Proc. IEEE Conf. Comput. Vision Patt. Recog., 2005, pp. 1202 –vol.

[31] B.D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” In IJCAI, 1981, vol. 81, pp. 674-679.

[32] C. Harris and M.J. Stephens, “A Combined Corner and Edge Detector,” in Proc. Fourth Alvey Vision Conf., 1988, pp. 147-151.

[33] J. Bouguet, “Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm,” Intel Corpor.5, pp. 1-10, 2001.

[34] G. Bradski, “The OpenCV Library,” Dr. Dobb’s Journal of Software Tools, 2000.

[35] R. Xu and D. Wunsch, “Survey of clustering algorithms,” IEEE Trans. Neural Networks, vol 16, no. 3, pp. 645 -678, May, 2005.

[36] N. Buch, S. A. Velastin, and J. Orwell, “A review of computer vision techniques for the analysis of urban traffic,” IEEE Trans. Intell. Transp. Syst., vol. 12, no. 3, pp. 920-939, Aug, 2011.

[37] R. Szeliski, Computer Vision: Algorithms and Applications. Springer, 2010.

[38] R. Ke, S. Kim, Z. Li, and Y. Wang, “Motion-vector clustering for traffic speed detection from UAV video,” in Proc. IEEE Conf. Smart Cities, Guadalajara, Mexico, 2015.

[39] D. G. Lowe, “Object recognition from local scale-invariant features,” in Proc. IEEE Conf. Comput. Vision Patt. Recog., 1999, vol.2, pp. 1150-1157.

[40] K. Kaaniche, B. Champion, C. Pegard, and P. Vasseur, “A vision algorithm for dynamic detection of moving vehicles with a UAV,” in Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conf. on, 2005, pp. 1878-1883.

[41] H. Moon, R. Chellappa, and A. Rosenfeld, “Performance analysis of a simple vehicle detection algorithm,” Image and Vision Computing, vol. 20, no. 1, pp. 1-13, 2002.

[42] Y. Freund, and R. E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting." Journal of computer and system sciences, vol. 55, no. 1, pp. 119-139, 1997.

[43] P. Viola, and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in Computer Vision and Pattern Recognition, 2001. Proceedings of the 2001 IEEE Computer Society Conference on, vol. 1, pp. I-511.

[44] Q. Chen, N. D. Georganas, and E. M. Petriu, “Real-time vision-based hand gesture recognition using haar-like features,” in Instrumentation and Measurement Technology Conference Proceedings, 2007.pp. 1-6.

[45] R. Rothe, M. Guillaumin, and L. Van Gool, “Non-maximum suppression for object detection by passing messages between windows”, Springer International Publishing, pp. 290-306, 2015.