Image-based Traffic Monitoring System

LAU Wai Hung

A Thesis Submitted in Partial Fulfillment
Of the Requirements for the Degree of
Master of Philosophy
In
Systems Engineering and Engineering Management


1 2 OCT 2007
CHINESE UNIVERSITY OF HONG KONG

Thesis/Assessment Committee

Professor Chun Hung Cheng (Chair)
Professor Janny Leung (Thesis Supervisor)
Professor Wai Lam (Committee Member)
Professor Kokin Lam(External Examiner)

Abstract

Traffic monitoring has been a major component of Intelligent Transport Systems (ITS). In many ITS around the world, cameras have been installed on the road to monitor the traffic condition. In Hong Kong, differing from most of the current systems, a sequence of discrete pictures instead of video is captured. In this thesis a static image-based traffic monitoring system is developed. Several methods based on edge detection are proposed to estimate the number of vehicles on the road. It is less computational demanding than video processing and yields reasonable results on tracking changes of traffic condition by analyzing a time series of these static images.

摘要

基於靜態圖像的交通監察系統

交通監視是智能運輸系統一個主要部分，世界各地許多智能運輸系統都利用安裝在路邊的攝錄機，拍攝影片以評估交通情況。而香港有別於大多數當前的系統，以攝取間斷圖片序列代替影片。在這論文中，我們開發一個基於靜態圖像的交通監察系統，利用幾種以邊緣檢測技術爲基礎的方法，估計路上的車輛數量。比較起影片圖像處理，靜態圖像處理的計算需求不高，並能透過分析一系時間序列的圖片，合理地評估隨著時間變動的交通狀況。

Acknowledgement

I work on this thesis with a grateful heart. Without the help and support from the professors, my family and friends, I would not have finished this thesis.

I would like to express my gratitude to my thesis supervisor, Professor Janny Leung, for her patient guidance on my research and thesis writing. During these two years of studies I have learnt the importance of rational analytical skills in conducting research as well as the cautious attitude of working. Her comments have benefited me a lot throughout the research.

I would like to thank Professor Cheng Chun Hung for his inspiring ideas to my research project, as well as his encouragement and support.

My research life in CUHK has been lightened by my friends in the department — Walter To, Chunli Liu, Keith Wong, Tony Woo, Gatien Lam, Gabriel Fung, Cathy Wong. All of them have shared my worries and thoughts, and now my happiness and this fruitful moment. I must thank also my dearest friends — Calvin Chan, Chilli Chiu, Thomas Hau, Jonah Wan and all my church mates for all their company and sharing.

My greatest gratitude goes to my family and Zia Lam who accompanied me and gave me courage to face my hardest time during the research. Thank you very much.

Finally, thank God for his blessing, leading and teaching.

Table of Contents

List of Figures

Chapter 1 Introduction

Traffic surveillance technologies have been one of the major components in road traffic management systems and traveler information systems all over the world. Traffic congestion can bring about significant economic loss to travelers due to time wasted in waiting. The problem is of growing concern to international cities with their growing population and increased motorization. Such cities rely on responsive road traffic surveillance systems in order to manage heavy traffic, detect traffic incidents and provide traffic guidance. Advances in technologies provide the means to collect latest traffic data in a large transport network that can support the building and solving of complicated traffic models as well as collecting latest traffic data in a large transport network.

Many Intelligent Transportation Systems (ITS) provide road traffic information on the Internet as a part of its traveler information system. There is a trend towards the collection of real-time traffic data by video image systems with CCTV installed along roads. In spite of the increasing popularity of using CCTV for road traffic surveillance in different aspects such as travel time estimation and traffic safety enforcement; such a scheme is computational demanding as it requires continuous robust processing of tens of frames per second. Yet systems often aim at providing up-to-date traffic condition for the public before traveling rather than tracking drivers who exceed the speed limits. In such cases, analyzing a series of static pictures for monitoring traffic condition taken at less frequent intervals is a cost-effective alternative. Provision of less-frequently

updated static pictures relieves the usage of bandwidth in the internet, and reduces computational effort for analyzing images as well. Some cities like London[1], Leicester[2] in the UK, and New York City[3] in the US have installed still picture cameras on the road for traveler's information through the internet.

Taking Hong Kong as an example, the need for the development of Intelligent Transportation System was well recognized. Hong Kong's roads are among the most heavily used in the world, with over 520 000 vehicles on 1 928 kilometers of roads. There are about 270 licensed vehicles for every kilometers of road. With growing population and continuous development, road traffic management has become a great challenge. The currently implemented ITS in Hong Kong includes several management systems such as Area Traffic Control Systems, Traffic Control and Surveillance Systems and Automatic Toll Collection System etc. In the website of the Transport Department Hong Kong[4], CCTV video clips are provided for several major roads in urban areas serving as traffic monitors. On the other hand, static pictures are also provided from more than 100 cameras installed on major roads all over the city.

Nonetheless, current static image-based traffic monitoring in Hong Kong and other cities mentioned provides merely one snapshot of the road. There is room for improvement upon still-imaged-based traffic monitoring. For instance, traffic condition can be identified from the image automatically with machine-learning techniques. In this way traffic condition can be tracked using a time series of such images not only for traveler's information but understanding road traffic dynamics. The trend of traffic

---

[1] http://www.bbc.co.uk/london/travel/jamcams/camloco/camlist.shtml
[2] http://www.northlight-images.co.uk/cameras.html
[3] http://nyctmc.org/
[4] The website of Road Traffic Infromation Services for Hong Kong: http://traffic.td.gov.hk/SwitchCenter.do

conditions over a time interval will definitely be more informative than only a single value at one single time-point about the traveling speed or travel time.

In this thesis, the potential of using periodically captured still images for automatic road traffic condition monitoring is investigated. The proposed system is able to learn the image patterns from collected images and identify the traffic condition in terms of number of vehicles on the road, and hence traffic density. Training data is required which are collected photos with manually-counted number of vehicles on the road. It will be shown that the changes in traffic condition can be tracked by analyzing a sequence of pictures over time.

The thesis is organized as follows. Chapter 2 introduces the studies on real time traffic data estimation, in particular techniques related to image-based road traffic surveillance, either by continuous image sequences from video or discrete static images. Chapter 3 introduces the framework of the proposed system and details about the learning methods. Chapter 4 shows experimental results for different learning methods and analyzes their performances. The strengths, weaknesses and potential of this system will be discussed in Chapter 5.

Chapter 2 Literature Review

The following review discusses traffic data collection methods, especially vision-based traffic monitoring. These image processing techniques provide the referential framework for our proposed systems.

2.1 *Traffic data collection methods*

Traffic data collection is one of the core components in every ITS because all strategic planning and operations management of traffic are based on traffic information. Real-time traffic measurements such as speed, volume, occupancy are the target values to be estimated using different detection techniques. Inductive-Loop detectors are one of the most common devices. When a vehicles passes over the detector, signals are induced inside; therefore, the flow (the number of vehicles passing per unit time) and occupancy (the percentage of time that a unit length of road is occupied by vehicles) data can be directly estimated. By determining or assuming a mean vehicle length, the traveling speed and thus travel time can be estimated (see [1], [2] and [3]). Using only one single-loop detector on a road and assuming a fixed mean vehicle length have some limitation, and can be problematic when estimating speed as discovered by [1]. In recent years, installing two detectors at each end of a road segment has also been tried, since individual speed can be directly estimated from the time delay between sensing by the two detectors. Effort has been paid on vehicle reidentification techniques, referring to techniques that identify the same vehicle from signals at two detectors.

Advanced sensing technologies such as Automatic Vehicle Identification (AVI) are possible which requires installation of additional hardware. Indeed RFID technologies have been used by companies like TransCore[5] as an alternative to loop detectors for vehicle reidentification. RFID readers are installed on the roads instead of loop detectors and tags are attached to vehicles so as to detect the vehicles. Besides, [4] makes use of dual-loop detectors, a kind of more advanced detector that explicitly tells the arrival time and leaving time of a vehicle on that detector. It matches vehicles by rough estimation of vehicle length. Some try using only the traditional single loop detector (see [5], [6] and [7]). [5] does not re-identify vehicles but estimates traveling speed from a vehicle's signal waveform. [6] uses cross-correlation of the flow at the upstream and downstream detectors, placed 0.5 miles apart, to estimate the travel time between the two. [7] assumes the travel time between two detectors can be regarded as drawn from the same probability distribution, and estimate the distribution from the cumulative upstream and downstream counts. Recent new technological developments include vehicle probes ([8]) and laser-based detection ([9]), and machine vision technologies, which has emerged with various advantages.

The advantage of vision-based traffic monitoring systems lays in its wide area detection capabilities. Many systems analyze traffic data collected with Closed Circuit Television (CCTV) cameras, usually mounted on roadside poles at heights from 5 to 15 meters. Once calibrated, the coordinates of the image can be mapped to the corresponding locations in actuality. This allows the efficient detection of spatial traffic

[5] TransCore is a company that provides advanced traffic management systems (ATMS). It implements automatic vehicle identification(AVI) by RFID technologies. Its homepage is on http://www.transcore.com/

parameters such as density, queue lengths, and speed variation at different positions on the road, which cannot be easily obtained by conventional loop devices. In addition, vehicle recognition with video provides ancillary information such as lane changing behavior, vehicle trajectories and car following behaviors which facilitate traffic flow analysis, speed enforcements and road safety monitoring [10]. There exist some variations of vision-based traffic-monitoring systems from literature in which CCTV are not installed at fixed position on road side. [11] proposes an aerial video monitoring method, meaning video taken from an airborne platform, though such methods are not as popular.

Existing comercial solutions for traffic monitoring systems includes Traficon, which measures traffic data as well as detects incidents by video image processing. On the other hand ILog provides traffic management solutions that consolidates different sources like cameras, sensors and database for traffic analysis. Its graphical user interfaces enables real-time traffic monitoring and on-screen interaction with data sources and signaling equipments. These solutions show that traffic monitoring is currently combining different techniques for accurate estimation. Static image-based traffic monitoring can be one of the new techniques integrated to these latest systems.

2.2 *Vision-based traffic monitoring techniques*

Most vision-based traffic-monitoring techniques depend heavily on the continuity of video images. Although it is possible for the human eye to roughly judge road traffic conditions from a single photo, the same task for a machine becomes too hard to be

accomplished effectively as it lacks the capability to "understand" road features and recognize vehicles without detecting moving objects. Nevertheless, a few studies have proposed methods that can avoid directly recognizing moving objects. For example, [12] analyzes colors of an imaginary line across the lane of the road, and determines whether some objects pass through that line, using a neural network methods based on the history of the pixel colors. For this step, continuous video is not needed. Unfortunately, background differencing techniques (to be introduced in Section 2.2.2) are needed, which in turn depends on continuous video again. In short it is difficult to estimate the traffic condition without continuous video frames. Yet these studies shed some light to tackling the current problem. The following review are presented from two aspects, vehicle tracking approaches and image processing techniques related to vision-based road traffic monitoring.

2.2.1 Vehicle tracking approaches

In road traffic monitoring, the video acquisition cameras are usually installed at fixed position above the ground to obtain a view on the passing vehicles on the road. With a stationary camera view, the position of roads and lanes in the image can be well defined by on-site surveying and calibration during the installation of the camera. Spatial differences (of colors) and temporal changes of features are considered in object recognition.

Region-based tracking

This is an explicit way of feature extraction directly recognizes the whole moving vehicle. The method identifies a 'blob', a region in the picture, associated with each vehicle and tracks its movement along the time series. These blobs are extracted by background differencing techniques. A background image is synthesized and updated continuously with image sequences. It is a background in the sense that there are no vehicles in the image. Any differences between the current frame and the background can only be attributed to vehicles. Therefore absolute differencing the two yields a "difference image" in which only blobs of vehicles remain. [13] reports that 90% of the vehicles were correctly detected and tracked. Simple classification of vehicles into two classes by blob dimensions was achieved also. However occluded vehicles may be grouped together as a large blob, mistaken as one large vehicle.

Active Contour tracking

This is similar to region based tracking but the boundaries of a vehicle, known as active contours, rather than the whole blob are considered. In [14] the contours are extracted with blobs from background differencing and frame differencing (to be introduced in Section 2.2.2). Similarly, the problem of vehicle occlusion may occur such that a contour includes more than one vehicle inside.

Model-based tracking

Studies like [15] classify vehicles by matching between the straight edges in the moving blob and 2D projection of predefined 3D model edge segments. It requires all

3D models of different vehicles to detect all vehicles. Clearly, this is not usually a feasible prerequisite due to extensive efforts in making 3D model for ALL possible vehicles. Again vehicle occlusion can ruin this approach.

Feature-based Tracking

Instead of recognition of the whole vehicle, some vehicle features are tracked. [16] detects different vehicle features in daylight and night-time. In daylight moving objects are recognized by frame-differencing algorithm. It aims at finding pixels with their color varying among 3 consecutive frames. When a vehicle is moving, the color of pixels at the peripheral of the vehicle varies along the frames, forming a hollow shape in differencing consecutive frames. These pixels with varying color are grouped or clustered into vehicles. Similar features are adopted in [17] too. At night, pairs of headlight moving in parallel direction are extracted as representation of vehicles. In [18], 'corner' features, which are defined as regions in the gray level intensity image where brightness varies in more than one direction, are tracked along the time frame as features of vehicles. Corners, which were tracked offset by the same spatial translation in every frame towards the exit region of the image, are grouped into a vehicle. The paper reports good accuracies in velocity estimation, but exhibits relatively larger error in flow rate and density estimation due to the difficulties in grouping 'corners'.

Feature-based tracking is capable of withstanding vehicle occlusion because partially occluded vehicles may still show some of their features unblocked for tracking. Nevertheless grouping feature units to vehicles are the major difficulties to deal with.

Applying this approach to our problem is feasible only if the movement of feature units can be traced with frequent enough image updating, say, 1 picture per second.

Vehicle passing detection

An alternative to object tracking by extracting object features is to select some reference points on the road in the image, and determine whether a vehicle is passing through the reference points in the current frame. [12] selects an imaginary line across a lane on the road as the reference points. It applies background differencing, and then builds a fuzzy neural network to analyze the pixel values on the line. By determining whether the line is covered by vehicles in each frame, number of vehicles passing through the line is counted and covering time is measured. Thus flow and occupancies can be measured. The estimation of speed then requires additional information about average vehicle length.

This method seems not to rely heavily on continuous image sequences. However, the core of this method lies in background update and background subtraction, which requires very frequent collection of background colors due to varying illumination condition and changing weather. In addition, the number of vehicles passing through the line cannot be counted without video.

2.2.2 Image processing techniques

The following two techniques are the most popular ones in effective tracing of moving objects from videos. They are usually adopted for vehicle tracking.

Frame differencing techniques

Frame differencing refers to pixel-by-pixel subtraction of pixel colors between two consecutive image frames. Consider a moving object that displaces a little between two frames, differences of colors occur at the peripheral of the vehicle. They appear as edges in the difference image. This is why the techniques are sometimes referred to as "moving edge" detection. [16] and [17] employ this technique. This technique suffers from slow moving traffic because no edge may form. Moreover colors of vehicle may affect the clarity of moving edges formed. Partial vehicle occlusions may lead to confusing edge patterns.

Background differencing techniques

Background differencing techniques has gained popularity in moving object tracking. Besides vehicle identification, other video-based monitoring uses the same techniques such as tracing the ball in soccer match monitoring, tracing the body movement in video conferencing or broadcasting. It performs well especially when the view of the video clips stay fixed for long period of time, fitting the situation of many traffic monitoring video cameras. Further studies such as [19] and [20] show that even vehicle shadows can be separated from vehicle bodies.

Background differencing techniques consists mainly of image segmentation, and background modeling. Image segmentation is the process to distinguish foreground object, ie. moving objects, from background. The current frame is subtracted (absolute difference) by the background image pixel by pixel, resulting in a "difference image". The operation can be described by equation (1) where $p_i(x,y)$ is the pixel value at

coordinate *(x,y)* of image *i*, and $p_{BG}(x,y)$ is that of background image BG. Moving objects cause color pixel differences between current frame and background image, leading to a high pixel value in the difference image ($p_{difference}(x,y)$). Background objects, in contrast, remains similar color in current frame and background image, so the difference value is small. Therefore thresholding is usually applied on the difference image. It segments the image into patches of foreground and background objects by considering the magnitudes of $p_{difference}(x,y)$. A binary image (black and white image) is obtained after thresholding.

$$p_{difference}(x,y) = \left| p_i(x,y) - p_{BG}(x,y) \right| \qquad (1)$$

$$p_{binary}(x,y) = \begin{cases} 1 \text{ (white pixel)} & \text{if } p_{difference}(x,y) > T \\ 0 \text{ (black pixel)} & \text{otherwise} \end{cases} \qquad (2)$$

The simplest way of thresholding is to decide a threshold value T such that pixel values larger than T in the difference image are regarded as foreground. Its mathematical representation is shown in equation (2). Note that higher T weakens the capability of object detection but lower T may falsely regard camera noise as foreground object. [13] determines a threshold value by considering the peak of the image histogram but in a rough manner. Other examples such as [21] and [22] use image histogram shapes or distribution to determine T. [23] estimates the standard deviation of camera noise and obtains T (denoted as TH_{BD} in that paper) by the standard deviation multiplied by 2.5.

Considering that foreground objects occur as blobs in difference image, [24] sets a smaller T for pixels adjacent to foreground pixels because they are more likely to be foreground pixels but a larger T for pixels adjacent to background pixels. This shares the same concept with hysteresis thresholding which is sometimes used in edge

detection. In hysteresis thresholding, two threshold T_1 and T_2 are used ($T_1 > T_2$) such that each foreground object must contains at least one pixel with value larger than T_1 and all other pixels with value larger than T_2. The popular Canny Edge Detector ([25]) has included this thresholding step. Obviously considering neighboring pixels is more computationally expensive. An exhaustive survey on thresholding techniques can be found from [26].

The background modeling defines how to estimate and update the background image. Frequent update of background is needed since illumination condition on the road changes frequently due to weather and sunlight, causing changes in background colors. Common approaches in background modeling are first segmenting the image into foreground and background, and then use the pixels classified as background to update the background image. In background estimation, [14] uses Kalman filters to track the evolution of pixel colors along a time series. If historical values of pixels can be stored, background can be estimated as median value of each pixel location of all the frames in the buffer ([27]), assuming that for each pixel the background color occupies more than half of all frames in the sequence. [20] and [28] assume that the pixel value of foreground (vehicles), background (static objects) or even shadows of vehicles (as distinguished from vehicle bodies) each follows a distribution. Mean and standard deviation of each distribution are then estimated and updated.

For updating the background image, a simple model like exponential smoothing can be used ([13], [14]). Some techniques consider the history of colors of a pixel. For instance, [23] updates a pixel as background only if the frame difference value is

persistently lower than a threshold TH_{FD} for consecutive several frames. [29] investigates the performance of different background estimation techniques.

The accuracy of vehicle tracking with background differencing techniques are usually less promising in traffic jams. Consider that image segmentation requires an accurate background image so that background and foreground objects can be distinguished by $p_{difference}(x,y)$ using (2). Under heavy traffic condition, the road is constantly covered by vehicles and motions are slow, thus no up-to-date background color is available. This leads to either contamination of the background image with vehicle colors or unchanged background since congestion, which is outdated with respect to the latest light condition. Inaccurate background image causes large values of $p_{difference}(x,y)$ for all pixels, making foreground and background hardly distinguishable. The problem is serious if the background update is not frequent enough relative to changes of background colors. For example a pixel is covered by moving objects for only next 2 frames, where frame update period is 2 minutes. Then the background color of the pixel becomes as outdated as 4 minutes before. Together with the problem of vehicle occlusion in traffic jams, high traffic density is unfavorable to background differencing techniques.

Chapter 3 Methodology

In this chapter, the framework of our system is introduced and the methods of estimating the number of vehicles from each image are discussed in detail. In order to avoid confusion, some terms are defined and explained first.

Terms:

Pixel: A point in the image. Its location is denoted by coordinates (x,y). The origin (0,0) of the image coordinates is at the top-left corner of the image.

Pixel value: The value representing the color of the pixel. Each pixel has a pixel value. In a grayscale image, pixel value 0 means black color. Higher pixel value refers to brighter gray color of the pixel. In a binary image, or usually called black & white image, there are only two possible pixel values: 0 for black and 1 for white.

Grayscale image: An image with gray colors ranging from black to white. In this thesis, grayscale image are all implied to be 8-bit grayscale image, meaning that there are 2^8=256 possible pixel values: 0 to 255 from black to white.

Sobel image: The resultant image after applying Sobel Edge Detector[6] to a grayscale image. In the input image, difference in color of a pixel from its neighboring pixel causes outline of objects to be seen. Therefore these

[6] Sobel Edge Detector is one of the popular edge detection operators in image processing. The details of its operation can be found in Appendix A.

pixels, colored differently from it neighbours, are regarded as edges. In a Sobel image (which is a grayscale image also), edges exist as bright lines in contrast to the dark background. Larger inter-pixel color difference in the input image causes brighter edges and thus with higher pixel values in the Sobel image.

Edge image: The image after processing with the Edge Detection module. It is an 8-bit grayscale image since edges in the edge image are of various brightness.

Edges: Bright pixels in the Sobel image and edge image, with relatively high pixel values. Each pixel is not clearly distinguished as belonging to edge or non-edge until a process called thresholding is applied.

Binary image: An image with only two pixel values: 0 for black and 1 for white. It is commonly called "black and white image".

Thresholding: A process that distinguishes each pixel in an image as either black or white based on its pixel value. The resultant image is a binary image. With respect to an edge image, it distinguishes each pixel as either an "edge pixel" (white) or a "non-edge pixel" (black).

3.1 *Solution Concept*

Our static image-based traffic-monitoring system deviates from many studies of the image based vehicle recognition in that static photos are collected as input instead of continuous video data. In order to apply our methods, we assume that the camera is

installed at a fixed position above the ground and is aimed along the road in the direction of travel. In this perspective, vehicles far from the camera can be occluded by vehicles closer to the camera. It is also assumed that the road is straight. In our system, consecutive images are collected every 2 minutes, forming a sequence of images. The relative large time interval between two consecutive pictures makes the tracking of moving object impossible. Therefore we cannot directly estimate the traveling speed and flow of the road by tracking a particular vehicle. Instead the number of vehicles in the image is estimated from each image, which is used as an indicator of traffic density.

The question we have to answer is "what features can be found from the image to in order to identify vehicles?" Although there is no video clips available, a sequence of pictures are provided which share the same background. Therefore by some image processing, it is possible to distinguish between background features and moving vehicles. The main concept of our method is identifying vehicles on the road by finding edges of vehicles. When a vehicle is on the road, differences in colors between the peripheral of vehicles and the ground, as well as difference in colors within the vehicle image, can be detected as edges. Similarly edges are found for the outlines of roads and trees too, but in contrast to vehicle edges, they occur at the same position in almost all the images. Therefore we can identify edges which exist at the same position in most images as background features and those exist in single images as moving vehicle features. In this sense we can use background differencing techniques to identify edges corresponding to vehicles.

In this thesis, 3 methods to process the edges are proposed in order to estimate the number of vehicles in each image. Combining these three methods is also experimented in our study.

3.2 *System Framework*

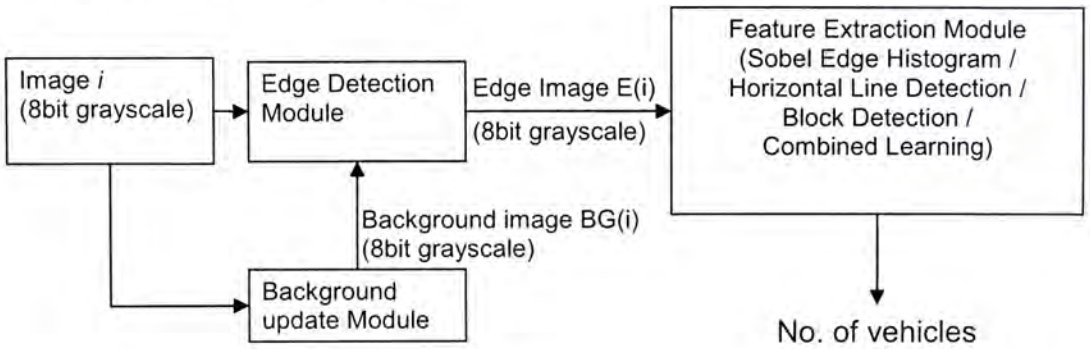


Image i: The i^{th} image in the image sequence
BG(i): Background image after updating with image i
E(i): Edge image after processing image i with edge detection module

Fig. 3.1 Framework of the proposed method

The system framework is shown in Fig.3.1. The edge detection module and background update module constitute the modified background differencing technique. In the Edge Detection Module, edge detection is applied to the acquired image. Edges of moving vehicles and also edges of static features like roads and streetlamps are detected. On the other hand the background image is constructed in Background Update Module so that it contains edges of only static features but not those of vehicles. After that, edges of static features are cancelled out by subtracting the image after edge detection with the background, but those of vehicles remain. The resultant image is then sent to Feature

Extraction module. For convenience of following discussion, this resultant image after Edge Detection Module is called an "edge image".

There are two reasons for using modified background differencing techniques. Firstly, edges of moving vehicles and of static features can only be separated with background differencing techniques. Secondly, as explained in Section 2.2.2, an accurate background image cannot be estimated if the image update is not frequent enough. The major problem is that the color of the road changes quickly under different illumination conditions. Therefore we resort to update the edge features. Edges of the road remain relatively stable for long time. So less frequent update of background image is required if edges is updated instead of colors.

After extracting the edges, three different ways of abstractions of the edge features is experimented to estimate the number of vehicles on the road, namely Sobel Edge Histogram, Horizontal Line detection and Block detection. From the former to the latter higher-level details are recognized and more computation are required. We will see if higher-level details extraction leads to more accurate estimation of number of vehicles. Sobel Edge Histogram simply considers the frequencies of pixels of different pixel values in the edge image. Horizontal Line Detection counts number of horizontal straight edges on the road and find the linear relationship between number of horizontal edges and number of vehicles. Block detection explicitly determines the location of each vehicle by considering the spatial distribution of edges on the road.

The process of each component in the system is explained as follows.

3.2.1 Edge Detection Module

We now describe the details of the Edge Detection Module.

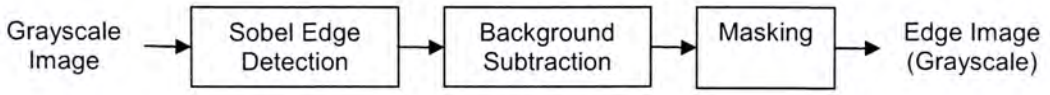


Fig. 3.2 Procedures of Edge Detection module

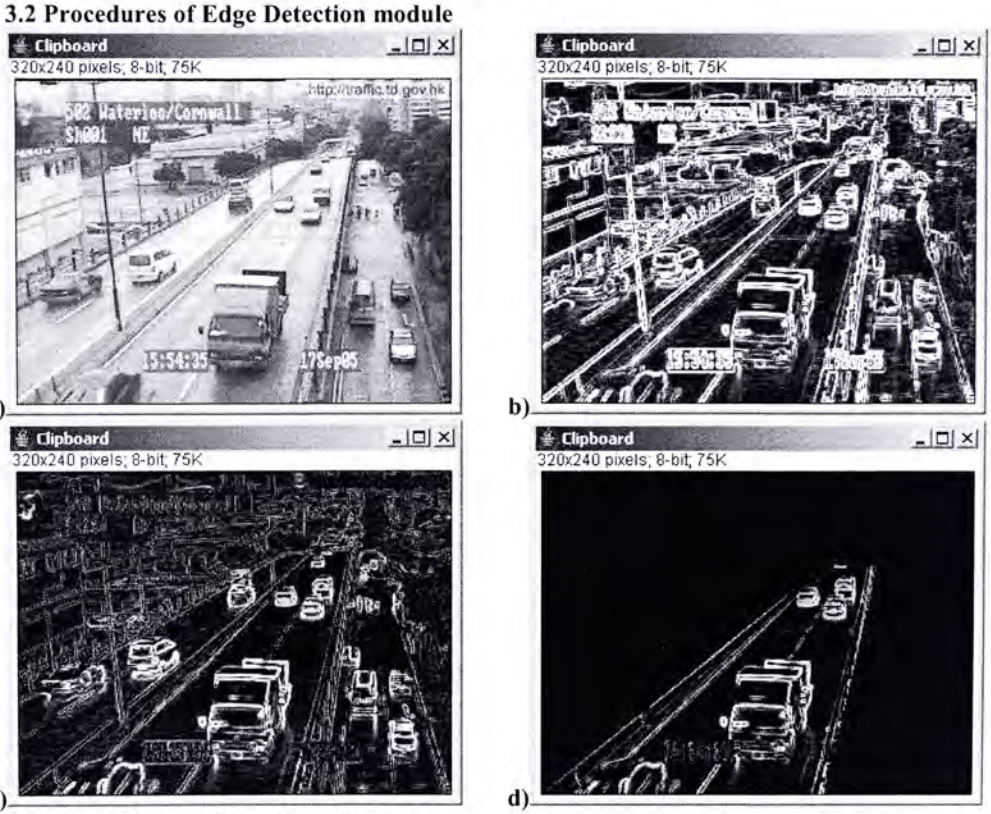


Fig. 3.3 Image through Edge Detection Module.
(a) Acquired Grayscale Image *i*. (b)Image after Sobel Edge detection: *Sobel(i)*. (c) Edge Image after background subtraction: *Sub(i)*. (d) Final Edge image after masking: *E(i)*.

When an up-to-date image is acquired, Sobel Edge Detection is first applied to find all the edges in the image. The Sobel image is obtained, which is then subtracted by the background edge image pixel by pixel. The background edge image, looking like the pictures in Fig.3.5, is a grayscale image that contains only edges of background features. Let us denote image *i* as the i^{th} collected image in the image sequence. The image

subtraction is represented by equation (3) where BG(i) is the current background image, Sobel(i) is the current Sobel Image and Sub(i) is the resultant edge image. Absolute sign is not taken in subtraction. This is because our background subtraction is operated based on edge detection. For edges of moving objects, they exist only in single image and the corresponding position in the background image should have no such edges. Therefore edges of moving objects must be of higher pixel value than the corresponding position in the background image.

$p_I(x,y)$: pixel value of pixel at coordinates (x,y) in image I.

$$p_{Sub(i)}(x,y)=max\{p_{Sobel(i)}(x,y)-p_{BG(i)}(x,y)\ ,\ 0\} \qquad (3)$$

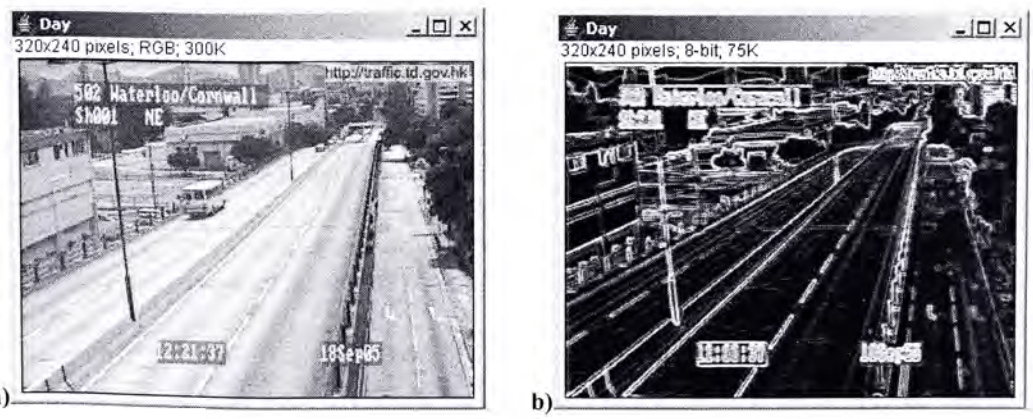
Subtraction cancels out edges of static features because they occur in the same position of both current frame and background image. In contrast edges of vehicles exist only in the current frame and thus remain after subtraction. In the next step the area outside of the selected road in the image is masked out (Fig. 3.3d). The mask is drawn manually. Fortunately the camera view is fixed, so that the mask can be established once and for all for each camera. The result is an edge image *E(i)*.

In our study, Sobel Edge detector is chosen among different common edge detectors. Generally, Canny edge detector ([25]) is recognized as the optimal edge detector according to several criteria, and it achieves especially good performance in localization to accurately mark edges, and minimizing the number of responses to a single edge, ie. only one-pixel thick for one edge. However Sobel edge detector shows stronger response to edges to give thicker edges. Thicker edges enable us to identify straight edges and regions densely distributed with edge pixels more easily. This favors

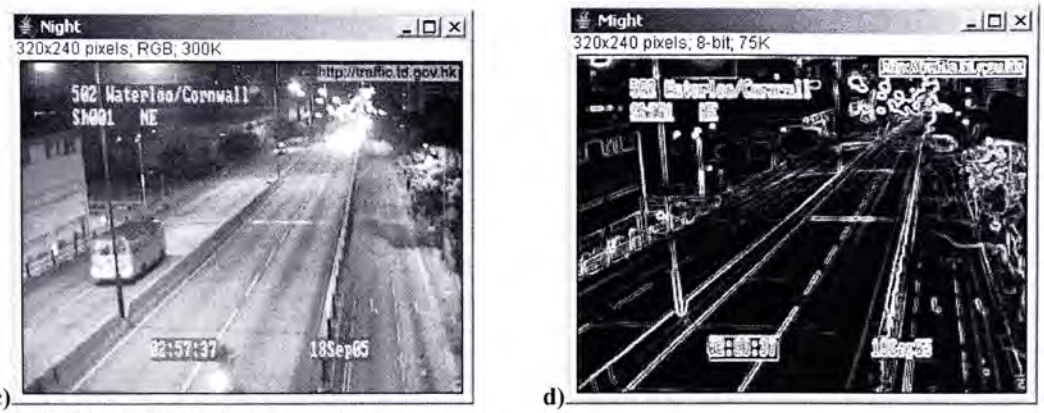
the processing in Horizontal Line Detection method and Block Detection method. Other examples using Sobel edge detector in vehicle detection includes [17], [30]and [31]. The description of the operation of Sobel Edge detection is provided in Appendix A.

3.2.2 Background Update Module

The Background Update Module constructs a background image which contains only static features like road markings, streetlamps and building structures. The edges in background image remain relative stable with respect to the time in terms of location and brightness, but they do changes slowly with different illumination. A clear example is shown in Fig.3.4, where a horizontal line on the road is obvious at night due to reflection of streetlamp lights but is unclear in daylight. It is therefore essential to update the background periodically.



a) b)



c) d)

Fig. 3.4 Example background edge images.
(a) An example image at noon. (b) An example background image at noon. (c) An example image at night. (d) An example background image at night.

When a new image is acquired, the background image is updated by exponential smoothing. Let BG(i-1) be the background image, and C be the current estimate of background, then the updated background image BG(i) is estimated as follows:

$$P_C(x,y)=\text{Min}\ \{P_{\text{Sobel}(i)}(x,y),\ P_{\text{Sobel}(i-1)}(x,y)\} \qquad (4)$$

$$P_{BG(i)}(x,y)= (1-\alpha)P_{BG(i-1)}(x,y)+ \alpha P_C(x,y) \qquad (5)$$

Let the i^{th} image be the latest acquired image. $P_i(x,y)$ refers to pixel value at coordinates (x,y) of the latest image in the sequence and α is a learning rate parameter between 0 and 1. The equation (4) estimates the current background. The aim of using the minimum function is to remove edges that exist in only in one single image. These edges with high pixel values that exist only in one image are probably due to moving objects. Fig. 3.4a illustrates that vehicle edges on one Sobel image are absent in the corresponding location of another Sobel image, so a minimum function picks the darker color(lower pixel value) as output, canceling out the vehicles edges in such a way.

Since it is possible that a pixel is occupied by vehicle edges for two consecutive pictures, false edges may be extracted in using equation (4). Therefore C is not directly adopted as the background. Instead it is used to update the background image by exponential smoothing, as described by equation (5). An initial background is needed for exponential smoothing. To obtain the initial background we can directly use image C, but start running the background update module for tens of pictures before the first picture to be studied in the sequence. Fig. 3.5 demonstrates how images are processed in the Background Update Module.

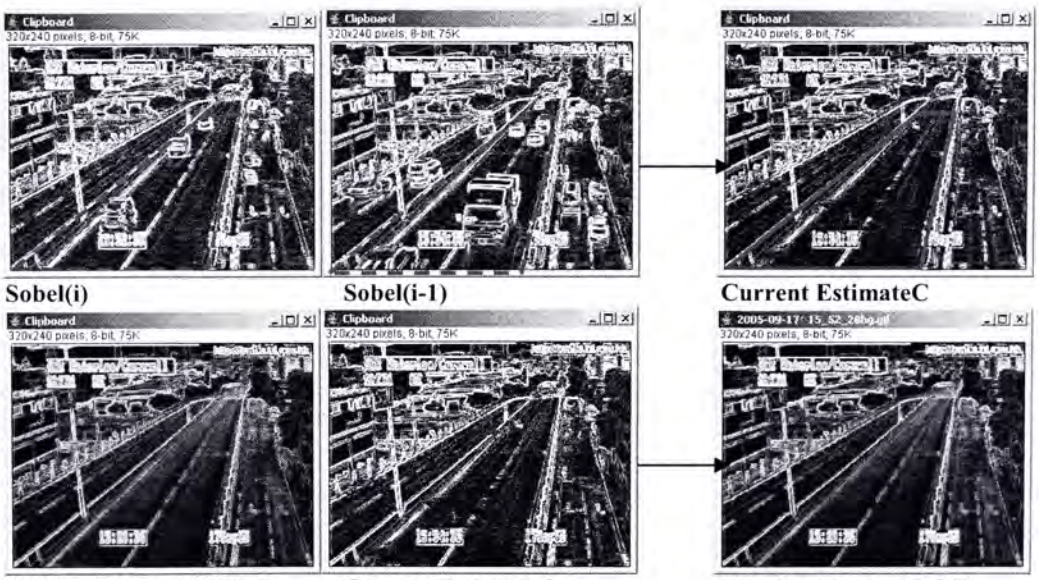


Fig. 3.5 Image through Background Update Module.
(a) Illustration of Equation 4. (b) Illustration of Equation 5.

In equation (5), a higher α implies a more responsive update to the current estimate of background while a lower α tends to keep the background stable. Yet consider that smaller learning rate can reduces the effect of false edge formation in equation (4), but larger learning rate may contaminate the background. Therefore a low

value of α is preferred. In order to choose the value of α, an experiment was conducted to compare the performance of Sobel Edge Histogram Method (to be introduced in Section 3.2.3.1) with several values of α. Sobel Edge Histogram Method is chosen among other methods because it utilizes the edge image, after background subtraction, the most directly and should be the most sensitive to the effectiveness of the background update module. As a result α=0.05 is found to have the lowest mean absolute error (MAE). Nevertheless the differences of MAE using various α values are less than 0.2 from the lowest MAE, showing that fine tuning of α is not critical to the performance of our system. Therefore extensive testing of other α value is omitted. Details of the experiment can be found from Table 6 of Appendix C.

3.2.3 Feature Extraction Modules

The Feature Extraction Modules receives the edge image from the Edge Detection Module and estimate the number of vehicles in the image. In this section, we describe three approaches of extracting edge features for the estimation. Data mining techniques are used and thus training data is needed. This is a set of images with known number of vehicles, which is counted manually. Combining the 3 approaches is also experimented.

3.2.3.1 Sobel Edge Histogram

The image histogram of an edge image can be used to estimate the number of vehicles on the road. When the number of vehicles on the road increases, the number of edges increases. And the image histogram changes too. The image histogram of an image

gives the tally of the number of pixels of different colors in the image. In an 8-bit grayscale image, there are 256 pixel values representing 256 gray colors. The frequency $h_E(v)$ of each pixel value v in an edge image E is defined by equation (6). When there are more vehicles on the road, there are more edges. Thus more pixels are brighter. Then the distribution of pixel values in the histogram shifts to higher values (brighter pixels have higher pixel value). An illustration can be found in Fig.3.5a and b.

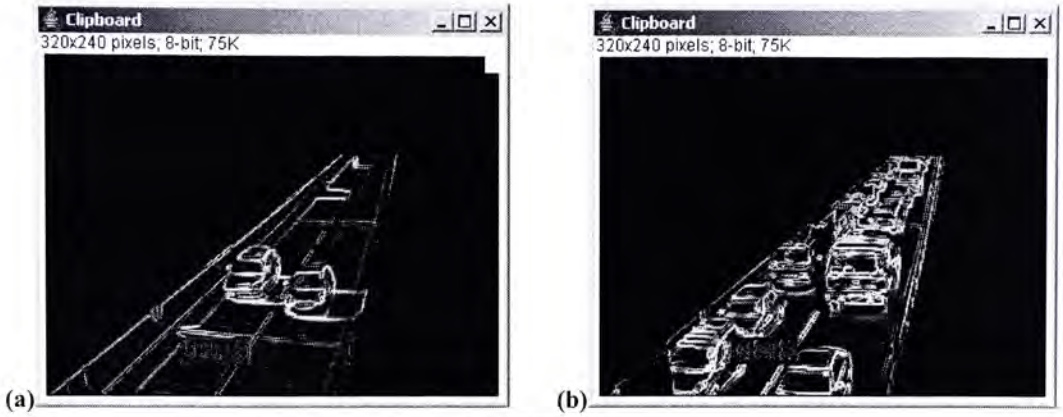
$$h_E(v) = \left|\{(x,y) \mid p_E(x,y) = v\}\right|,$$

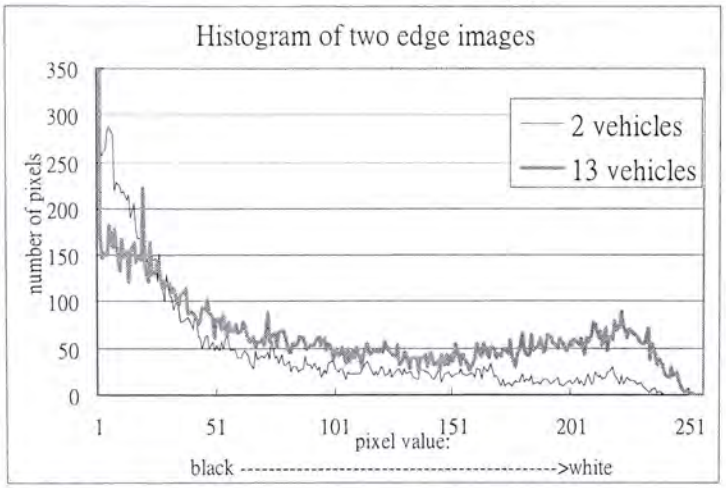
or equivalently

$$h_E(v) = \sum_{\{(x,y)|p_E(x,y)=v\}} 1, \qquad \text{where v=0, 1, 2, ..., 255.} \qquad (6)$$

In order to find the mapping between $h_E(v)$ and the number of vehicles, the Sobel Edge Histogram method builds a 3-layer feed-forward neural network(FNN)[7] for predicting the number of vehicles. In the input layer, there are 256 input nodes. Each node represents a $h_E(v)$ value in the Sobel Edge histogram constructed from the edge image. The hidden layer processes the input values with nonlinear functions (sigmoid function in our case) to give the output value. The output layer contains only one node representing the number of vehicles in the image. Because this is a neural network approach, training samples are required. These are a set of photos with the number of vehicles in the image counted manually. Detailed setting of the neural network is provided in Appendix B.

[7] A "feed-forward" neural network means a network that units in one layer are only connected to those in the next layer. In the 3-layer neural network, the 3 layers are connected from input layer, hidden layer, to output layer. Details about the neural network are provided in Appendix B.





(b)

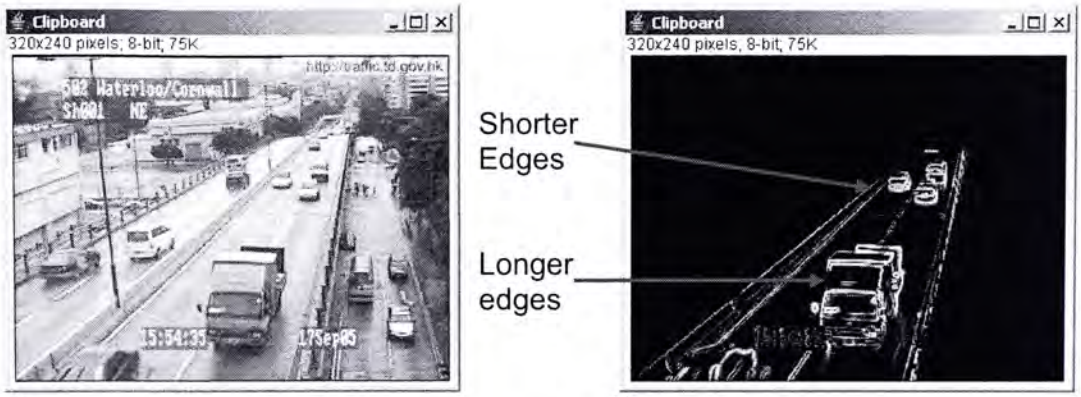
Fig. 3.6 Relationship between number of vehicles and histogram.
(a) Edge image of picture with 2 vehicles (b) Edge image of picture with 13 vehicles (c) Corresponding image histogram of Edge Images in (a) and (b)

It is easily noticed as illustrated by Fig. 3.7 that vehicles in front of the camera are larger, thus their edges are longer. This makes the histogram based prediction easily fooled by position of vehicles from the camera. In order to balance the effect of vehicles at different distances from the camera, the frequencies $h_E(v)$ have to be adjusted. We assume that the lengths of the edges are proportional to the width of the road at different distances from the camera. Hence, the tally given by one pixel is 1 divided by width of the road, $w(y)$, instead of being 1. The equation is given as follows:

$w(y)$: width of the road on road y, measured in pixels

$$h_E(v) = \sum_{\{(x,y)|p_E(x,y)=v\}} \frac{1}{w(y)} \quad \text{where v=0,1,2,...,255} \quad (7)$$

To distinguish the 2 ways of calculating $h_E(v)$, we call this modified method, which calculates $h_E(v)$ with equation (7), the Proportionate Sobel Edge Histogram. The previous method, which calculates $h_E(v)$ with equation (6), is called the Simple Sobel Edge Histogram.



a) b)

Fig. 3.7 Edge image with different lengths of edges due to location of vehicles (a)Grayscale Image. (b)Corresponding edge image

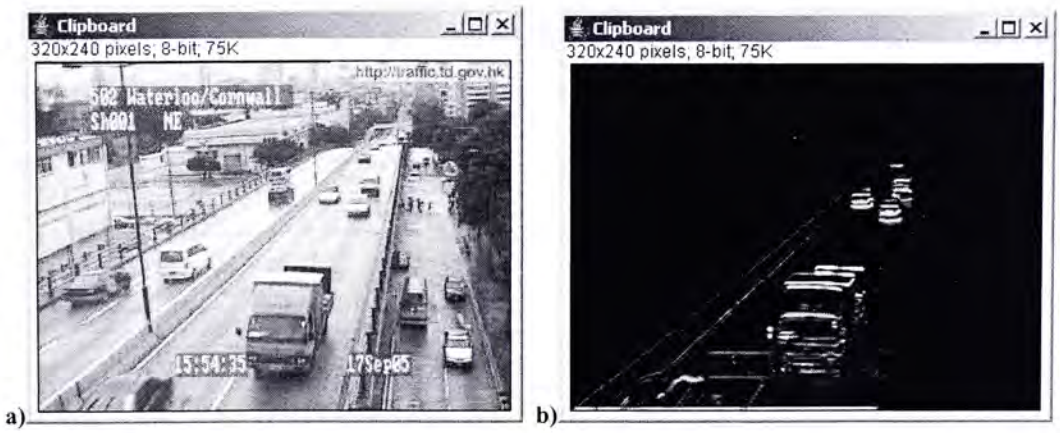
3.2.3.2 Horizontal Line Detection

Horizontal Line Detection aims at estimating the number of vehicles by counting the number of horizontal edges given by vehicles in the edge image. It is assumed that the number of vehicles is linearly related to the number of horizontal lines found on the road as suggested by the plots in Fig. 3.8e. The exact form of this linear relationship is determined by regression on a set of "training images".

Implicitly it is assumed that the direction of traveling is towards (or away from) the camera. Therefore these horizontal lines on the road are actually vehicle edges, not road markings or any other objects. In case the direction of camera's view makes an angle with the direction of traveling, then vehicle edges are not necessarily "horizontal" with respect to the image. Yet we can rotate the image so that the vehicle edges are horizontal again. Before going into details, the procedure is summarized and illustrated in Fig. 3.8.

Procedure:

1. *Use thresholding to distinguish each pixel in the edge image as either edge or non-edge*
2. *For each row in the image with least consecutive n edge pixels, identify a horizontal line (n=1/3 of width of the lane on the row y in the image)*
3. *Count the number of horizontal lines. (If two horizontal lines are found on adjacent rows and one line is merely one pixel above or below the other, then they are counted as one line only)*
4. *Estimate the number of vehicles from number of horizontal lines with trained linear regression model*



a) b)

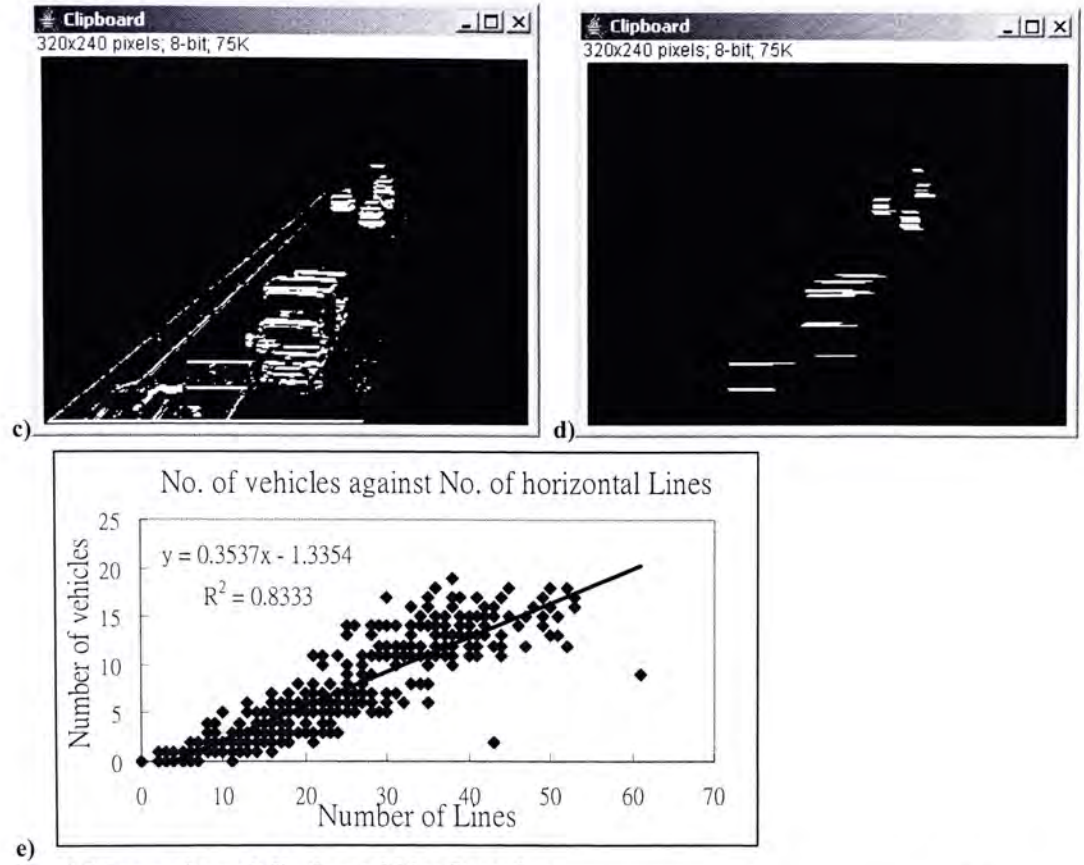


Fig. 3.8 Image through Horizontal Line Detection.
(a) Grayscale image. (b) Taylor-made edge image. (c) Edge image after thresholding. (d) Edge image with only horizontal lines detected. (e) Linear regression built from one sample of 600 images

The following explains the details of this method. Horizontal Line Detection starts with an edge image, but only horizontal edges are considered. Therefore there is a little modification to the Sobel Edge detection in the Edge Detection Module. Originally the pixel value for each pixel in the Sobel Image is calculated using equation (9), where Gx and Gy represent the magnitude of differences in color across the horizontal and vertical direction respectively. In this method, however, the pixel value is calculated with equation (8), merely considering color differences across the vertical direction.

$$|G| = |Gy| \tag{8}$$

$$|G| = \sqrt{Gx^2 + Gy^2} \tag{9}$$

As a result, the tailor-made edge image is obtained, showing mainly horizontal edges. The next step is applying thresholding to the edge image. Unimodal thresholding of [22] is chosen among other thresholding techniques with the following consideration. In the edge image, a large amount of dark pixels are present while there are few bright pixels (with high pixel value). As a consequence, there is only one significant peak in the histogram (shown in Fig.3.6b). This is referred to as an unimodal shape of the histogram, and is exactly the assumption made in [22]. A binary image is the result with edges pixels in white and non-edges pixels in black.

Then horizontal lines are identified from the binary image. There is a requirement on the lengths of the lines: let n be the minimum length of an identified horizontal line measured in pixels, n must be approximate to the width of a vehicle because we aim at vehicle edges. By inspection of the edge images in our study, n is set to one third of width of a lane (measured in pixels) as an approximation. As the width of a lane closer to the camera appears wider in the image but narrower for farther position, we express n in (11) for each row y with the help of w(y), the width of the road at y-coordinate value y. In short, we examine each pixel row y in the image, and identify a horizontal line with at least n consecutive pixels along the row.

$w(y)$: width of the road on row y, measured in pixels
r : number of lanes on the road

$$\text{width of one lane (measured in pixels)} = \frac{w(y)}{r} \qquad (10)$$

$$n = \frac{width\ of\ one\ lane}{3}$$

$$= \frac{\mathrm{w(y)}}{3\mathrm{r}} \qquad (11)$$

Finally the number of lines is counted. Sometimes two identified lines are on adjacent row and one line is directly below (one pixel below) the other, then they are regarded as one line only. In training the module, the number of horizontal lines is used to build a linear regression model with the number of vehicles as output variable (shown in Fig 3.8d). In operation, the regression equation is used directly to estimate the number of vehicles from the number of horizontal lines.

3.2.3.3 Block Detection

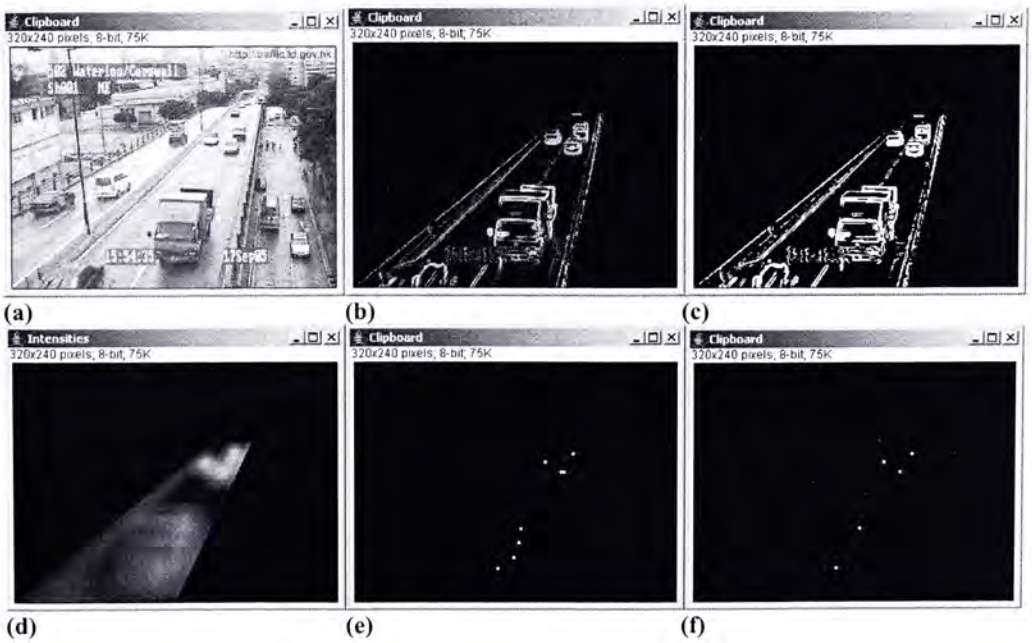
In the edge image, not only are there edges on the peripheral of a vehicle, but also edges inside the area occupied by it due to edges from window frames or logos. In contrast there are no edges in the uncovered road area in the image (except edges due to road markings, which are mostly cancelled out after background subtraction in the Edge Detection Module). Therefore vehicles are located in regions densely distributed with edges in the image.

Block Detection tries to estimate the location of a vehicle by finding a region more densely distributed with edge pixels than neighboring region. It considers all rectangular blocks anchored by every pixel in the image, and determines if the block represents a vehicle. The size of the block considered is determined by the position of

its anchoring pixel and is pre-set to be the size of an average vehicle. For each block, we compute its edge density. The block is determined to be representing a vehicle if its edge density is higher than that of neighboring blocks, or in other words, the block location is at a local maximum of edge density. However if two possible blocks are as close together, and as within one vehicle length of each other, they are considered the same vehicle. The procedures is summarized as follows and illustrated in Fig. 3.9.

Procedure:

1. *Threshold the edge image to binary image*
2. *Estimate the "edge density" of block B(x,y) on every location (x,y)*
3. *Find local maxima of edge density (blocks with higher edge density then neighboring blocks) and identify it as a vehicle*
4. *Group local maxima which are within one vehicle length and width from each other, as they are recognized as the same vehicle.*
5. *Count number of vehicles detected*



(g)

Fig. 3.9 Image through Block Detection.
(a) Grayscale image (b) Edge image (c) Binary image after thresholding (d) Visualization of density(x,y) (e) Local maxima of density(x,y) (f) Local maxima of density(x,y) after grouping (g) visualization of located vehicles (blocks in white and vehicle edges in gray)

The steps of Block Detection are more complicated than other methods. It first distinguishes each pixel in the edge image as either "edge" or "non-edge" by thresholding of the edge image. Same as Horizontal Line detection, unimodal thresholding is used. In the second step, for every possible location of a vehicle on the road, a block is constructed in the size of an average vehicle and its edge density is calculated. Let (x,y) be the location of bottom left corner of a vehicle, a block B(x,y) is the region in the image covered by the vehicle. As shown in Fig. 3.10, the B(x,y) is the region to the top right of pixel (x,y). Parameters *h* and *k* are defined as follows to express the dimension of B(x,y), which is equivalent to *vehicle width* by *vehicle height*. On the other hand, the edge density of a block B(x,y), denoted *density(x,y)*, is defined as the proportion of edge pixels to total number of pixels in the block. Equation (14) shows the calculation of *density(x,y)*.

h : average vehicle width, in fraction of width of a lane on the road
k : average vehicle height, in fraction of width of a lane on the road

average vehicle width (in pixels)$=h\times(width\ of\ one\ lane)$

$$=h(\frac{w(y)}{r}) \qquad (12)$$

average vehicle height (in pixels)$=k(\frac{w(y)}{r})$ (13)

$$density(x,y)=\frac{\text{no. of edge pixels in block B(x,y)}}{\text{total no. of pixels in block B(x,y)}}$$

$$=\frac{\sum_{i=x}^{x+hw(y)/r}\sum_{j=y-kw(y)/r}^{y}p(i,j)}{[hw(y)/r][kw(y)/r]} \qquad \text{where } p(i,j)=\begin{cases}1 & \text{if pixel (i,j) is edge pixel}\\ 0 & \text{otherwise}\end{cases}$$

$$=\frac{r^2\sum_{i=x}^{x+hw(y)/r}\sum_{j=y-kw(y)/r}^{y}p(i,j)}{hk[w(y)]^2} \qquad (14)$$

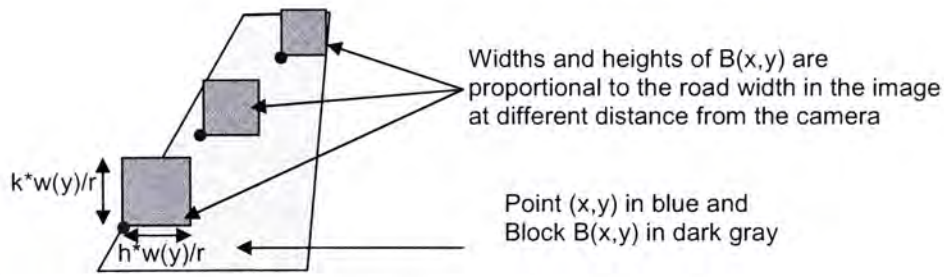


Fig. 3.10 Rectangular block construction for each pixel on the road

As a result, a 2D array of *density(x,y)* is formed with index x and y. Visualization of the density array is shown in Fig. 3.9c. In the third step, we determine block B(x,y) to be representing a vehicle if its edge density is the highest among its neighboring blocks. In other words the locations of vehicles are local maxima of edge density. This is based on consideration that a block covering the whole vehicle has higher edge density than

neighboring blocks covering only part of the vehicle. Nevertheless local maxima do exist even there are no vehicles on the road, though with low density. Hence, there must be a threshold value t such that only if $density(x,y)>t$, it is regarded as a vehicle.

In addition, one vehicle may cause more than one local maximum in the density array, thus local maxima are grouped in the fourth step. Consider a vehicle found in block B(x,y). There cannot be another vehicle in the region within one vehicle length and width from the block's location, otherwise two vehicles would be too close and clash. We denote this region as A(x,y). Then any local maximum of density(x,y) in region A(x,y) are regarded as from the same vehicle. Therefore in this step, we scan from the bottom of the image to the top, correspondingly from near side of the road to the far side. For each local maximum density(x,y) larger than t, any other local maximum in A(x,y) is erased. At last the number of vehicles can be counted.

The dimension of A(x,y) is defined below using equations (15)-(17), with the help of Fig. 3.11. Let (x_1,y_1) be the left-bottom corner of an identified vehicle. An imaginary line (dotted line in Fig.11) is drawn as the left side of the vehicle with x-coordinates $x_c(y)$. (15) requires the imaginary line to be parallel to the traveling direction on the road surface. (16) finds y_2 such that interval $[y_2,y_1]$ represents the average vehicle length l on the road. (17) confines y-coordinates of A(x,y) to be one vehicle length from y_1, and x-coordinates to be one vehicle width left and right from x_c.

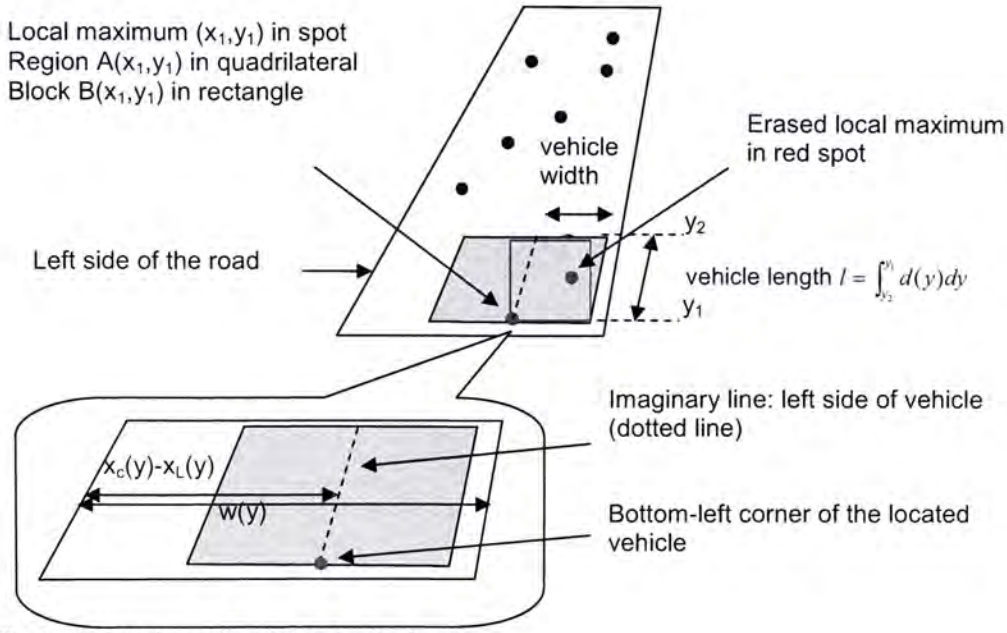


Fig. 3.11 Notations of variables in the image

$x_c(y)$: the x-coordinates of the left side of located vehicle
$x_L(y)$: the x-coordinates of left side of the road on row y in image coordinates
y_2 : the y-coordinates of the rear end of vehicle
$d(y)$: the actual distance along the traveling direction represented by a pixel on row y in image coordinates, in meters per pixel
l : the average vehicle length, in meters

$$\frac{x_c(y)-x_L(y)}{w(y)}=\frac{x_1-x_L(y_1)}{w(y_1)} \tag{15}$$

$$l=\int_{y_2}^{y_1} d(y)dy \tag{16}$$

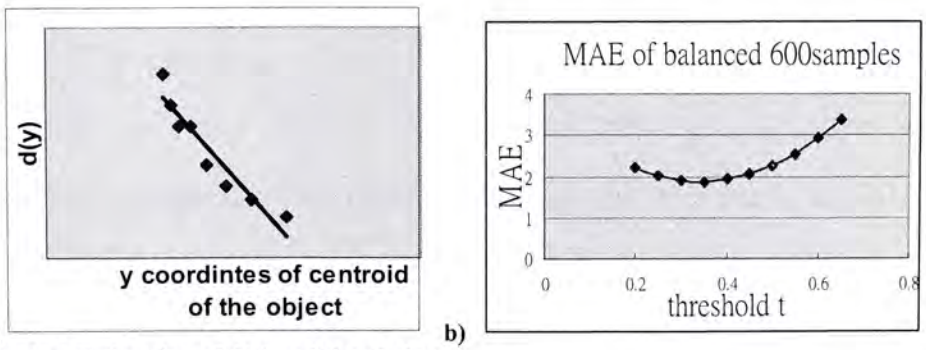
$$A(x_1,y_1)=\left\{(x,y)\middle|\left(y_2\le y\le y_1\right)\wedge\left(\left|x-x_c(y)\right|\le\frac{hw(y)}{r}\right)\right\} \tag{17}$$

In Block Detection method, some parameters are needed and obtained as follows. $w(y)$ is measured in pixels and thus is observed from the image. h and k can be determined by collecting average actual widths and heights of vehicles and width of the road. In our case h=k=½. l, average vehicle length, is usually assumed in most research literature but can be possibly found from the Transport Department. However, y_2 and $x_c(y)$ are

calculated based on d(y), which requires a calibration. The method proposed by [17] is used, which approximate d(y) as a linear function of y-coordinates of the picture. The length (in pixel) of a moving vehicle is measured in each frame to estimate $\hat{d}(y)$ by (18) (Fig. 3.12). In our case, moving vehicles are not available. But the lane dividing lines, whose length of each segment remains constant along the road, are used instead of vehicles to calibrate d(y).

$$\hat{d}(y) = \frac{\text{length of vehicle(in feet)}}{\text{length of vehicle(in pixels)}} \qquad (18)$$

The threshold t in the third step is determined by experiment. Indeed if t is too high, vehicles may be ignored and this leads to underestimation of number of vehicles. If t is too low, even little road markings may be identified as vehicles and overestimation occurs. Both lead to large error of estimation. Therefore we can experiment different values of t to "training" images and find one which minimizes the mean absolute errors.



a) b)

Fig. 3.12 Estimation of d(y) and threshold t
(a) Estimation of d(y) by linear regression. (b) Finding optimal t

3.2.3.4 Combined Learning

The above learning methods are all based on edge detection but treat the spatial distribution of edges in different ways. While one method may prevail over another in some situation, each method may have its own strength. In order to gain advantage of different methods, combining all methods is experimented. The above learning models are trained separately at first, and give their individual prediction values. Then another learning model is built on top of these predictions. The input parameters are predictions from individual models and the output feature is the number of vehicles.

A piecewise linear regression model is chosen as the learning model based on the following consideration. It is conjectured that the accuracy of each method varies with different traffic conditions. For example, the block detection method probably demonstrates the same weakness as region based vehicle tracking based on background differencing techniques. During heavy traffic, vehicle occlusion causes difficulties in identifying vehicles separately. Therefore it is possible to divide the sample into several pieces. For each piece prediction is made as a linear combination of individual predicted values. The coefficient in each piece of linear curve reflects the relative accuracies of prediction for different traffic condition. Therefore a tree-based piecewise linear model is built.

An example of the tree-based piecewise linear model is shown in Fig. 3.13. The WEKA package provides the implementation of this model. The model is named M5 Tree Model. It firstly builds a decision tree to minimize the intra-subset variation of the class value (the number of vehicles in our method) down the branch. Then linear

regressions is constructed on each leaf of the tree. The details of M5 Tree Model can be found from [32] and [33].

$Cars$: predicted values of combined learning
p_{Sobel} : predicted values from Proportionate Sobel Edge Histogram method
$p_{H\text{-}Line}$: predicted values from Horizontal Line Detection method
p_{Block} : predicted values from Block Detection

If ($p_{Sobel} <= 8.413$)
$Cars = 0.4695 * p_{Sobel} + 0.2442 * p_{H\text{-}Line} + 0.2919 * p_{Block} - 0.1528$
else
$Cars = 0.715 * p_{Sobel} + 0.1419 * p_{H\text{-}Line} + 0.0128 * p_{Block} + 1.997$

Fig. 3.13 An example M5 Tree built in training combined model

Chapter 4 Experiments and Evaluation

In this chapter experiments are conducted to evaluate the performance of the static image-based traffic monitoring system. Unfortunately no similar studies are available for comparisons because our measurement, the number of vehicles, is different from others.

4.1 *Setup and Data Collection*

Images are collected from the website of the Traffic Department of Hong Kong[8]. Two lanes on the bridge over Waterloo Road towards the south are the target for traffic density estimation. To prepare the pool of training data, sequences of photos were collected at different time periods between 17 Sept 05 and 13 Feb 06 by downloading one photo every two minutes. Considering that headlights of vehicles in night condition may cause different image patterns of edges, sampling, model training and testing of all methods are conducted separately for daylight condition and night condition. Based on the consideration of headlights, a picture is classified as night photo if all vehicles have switched on headlights in the image. Separation is done manually only in this stage because the aim is merely comparing the performance under two conditions. In spite of lacking automation, according to the mentioned rule, it is usually easy to find one picture among a sequence of pictures as the dividing case between day and night. Picture by picture inspection is not necessary.

[8] http://traffic.td.gov.hk/selection_c.htm

Since the traffic condition is dominated by free traffic everyday, random sampling may reduces prediction power for heavy traffic. Yet accurate information in heavy traffic is probably more important to drivers than free traffic. Therefore balanced sampling is used, trying to improve predicting accuracies. It is also compared with random sampling to investigate the effect of sampling method on the prediction accuracy for different traffic condition. To obtain a balanced sample, the photos are divided into three groups: 0-4 vehicles, 5-10 vehicles, 11-20 vehicles. Each sample contains the same proportion of pictures belonging to each group. In random sampling photos are randomly drawn from the pool. Image processing and implementation of learning methods are implemented with Java including ImageJ 1.36 package[9] and Weka 3.3 package. The program runs on Windows platform of a PC with CPU at speed 2.8GHz and memory of 1GB.

4.2 *Evaluation Criteria*

The objective of this system is to estimate the traffic condition in terms of number of vehicles on the road, which is an indicator of traffic density. The true value of the number of vehicles in each image is obtained by direct manual counting. Four methods, namely, Simple Sobel Edge Histogram, Proportionate Sobel Edge Histogram, Horizontal Line Detection and Block Detection all require training with labeled samples. 10-fold cross validation is used for evaluating the learning methods as the sample size is relatively small compared with the number of distinct values in

[9] ImageJ is an open source package for JAVA programming which provides tools for image processing. It can be found from the website : http://rsb.info.nih.gov/ij/

prediction (0 to 20 vehicles). Mean absolute errors (MAE) and root mean square errors (RMSE) are used as common measurements of error of numerical predictions. The formulae are shown in equation (19) and (20) where p_i and a_i represents the predicted number of vehicles and actual number of vehicles (by manual counting) for each instance i respectively.

$$MAE = \frac{\sum_{\text{each instance i}} |p_i - a_i|}{\text{no. of instances}} \qquad (19)$$

$$RMSE = \sqrt{\frac{\sum_{\text{each instance i}} (p_i - a_i)^2}{\text{no. of instances}}} \qquad (20)$$

To obtain a percentage measurement, mean absolute percentage error (MAPE) is commonly used. However zero values exist in a_i, actual number of vehicles, which is a denominator in calculation of MAPE in equation (22). Therefore MAPE is not a valid in our evaluation. A simpler measure is used instead, percentage mean absolute error (PMAE). We define PMAE in equation (21) as MAE divided by average number of vehicles per instances of the sample. Zero PMAE refers to absolutely accurate prediction and 100% PMAE means that MAE is as large as the average number of vehicles in the sample. Moreover the accuracies of prediction may vary with traffic condition, so error measurements will be also calculated separately for each of 3 groups of photos mentioned before (0-4, 5-10 and 11-20vehicles).

$$PMAE = \frac{MAE}{\bar{a}} \qquad (21)$$

$$MAPE = \frac{\sum_{\text{each instance i}} \left| \frac{p_i - a_i}{a_i} \right|}{\text{no. of instances}} \qquad (22)$$

4.3 *Experimental Results*

Observation of the experiments results are explained in this section, while the numerical data from our experiments are provided in Appendix C for convenience.

4.3.1 Comparing overall accuracies

The overall accuracies of prediction by different methods are measured at first. Fig. 4.1 shows the MAE at different sample sizes (150, 300, and 600 instances) by each learning method using balanced sampling. In overall, Proportional Sobel Edge Histogram performs the best among the individual learning methods. Its mean absolute error is 1.23 at sample size=600. Horizontal line detection makes slightly larger error. For Simple Sobel Edge Histogram, its error at sample size=150 is 1.85, larger than any other individual methods. But with larger sample size its error is close to that of Horizontal Line Detection and Proportionate Sobel Edge Histogram. Surprisingly Block Detection does not perform stably. Its error is 2.16 at sample size 300 in spite of scoring 1.44 at sample size 150, and the overall error is the highest except sample size=150. The next section(4.3.2) will help us understand its strange performance. On the other hand, we can see the advantage of combining methods, scoring 0.95 when sample size is 600, about 0.28 lower than Proportionate Sobel Edge Histogram.

It is observed that the decrease of error from sample size 300 to 600 is generally less than the reduction from 150 to 300, except Block detection that behaves strangely. For example the MAE of combined learning scores 1.91, 0.96 and 0.95 for sample size 150, 300 and 600 respectively. This suggests that further increase in sample size will hardly reduce the error significantly.

The patterns of RMSE and PMAE in Fig. 4.2 and 4.3 respectively are similar to those of MAE. The PMAE shows a slight increase for sample size 300 to 600. This is probably due to smaller average number of vehicles in the sample ($\bar{a}$=6.83 and 5.97 for sample size 300 and 600 respectively) only.

The lowest percentage error at sample size 600 among the individual methods is 20.6% of the Proportionate Sobel Edge Histogram. Combining methods reduces the error further to 16.0%. For convenience of the comparisons in later parts, MAE will be used as the only measurement.

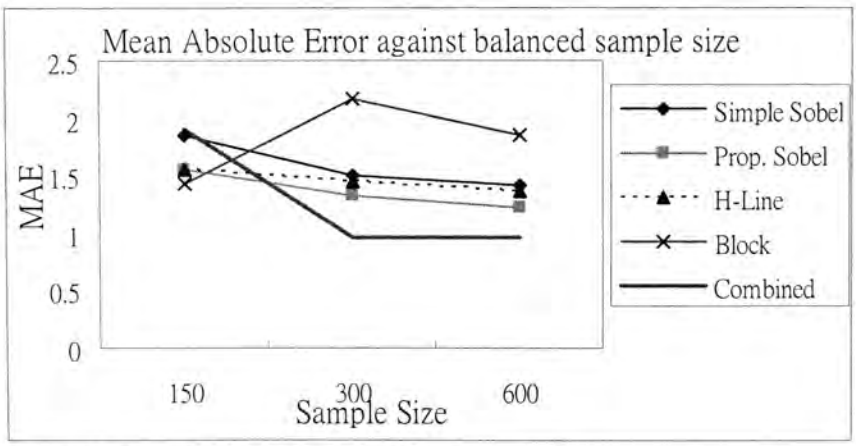


Fig. 4.1 MAE against sample size (balanced sample)

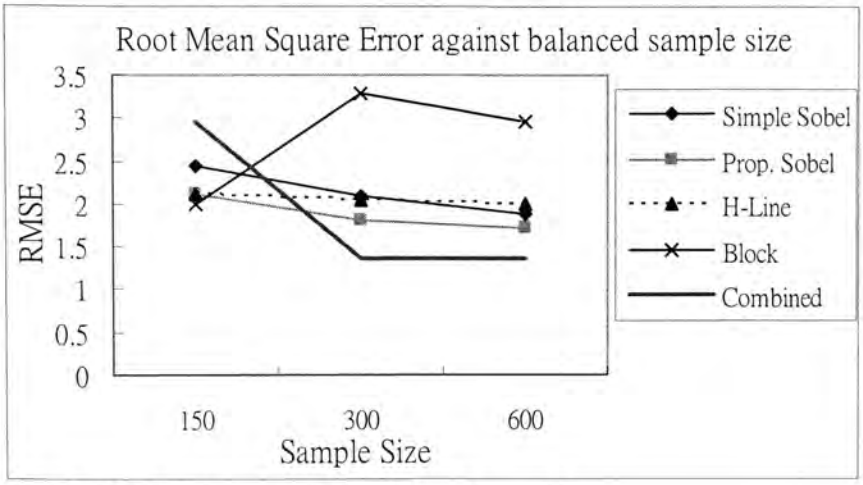


Fig. 4.2 RMSE against sample size (balanced sample)

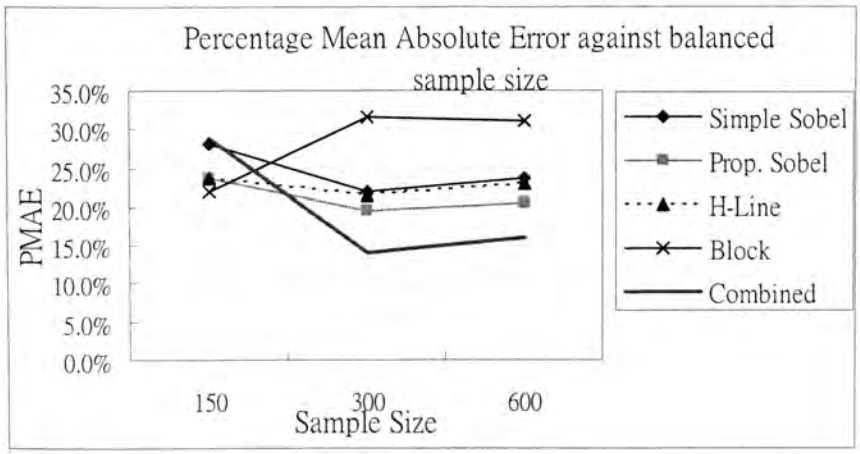


Fig. 4.3 PMAE against sample size (balanced sample)

4.3.2 Accuracies for different traffic conditions

In this section, errors are broken down by photo groups according to number of vehicles on the road. This reveals the variation in performance in different traffic conditions which cannot be observed from overall MAE and RMSE in Fig. 4.1 and 4.2.

Fig.4.4 shows the MAE of each photo group by different learning methods using a sample of size 600. We can see the MAE increases with increasing number of

vehicles. This trend can be explained by vehicle occlusion. When a vehicle is occluded by another, only part of its edges is visible in the image. The farther away a vehicle is from the camera, the larger a part is occluded. Therefore when there is heavy traffic, we can hardly count the vehicles far from the camera by finding their edges, thus all methods using edge image fail to estimate the number of vehicles accurately. An interesting point is that this trend is magnified in block detection, which depends heavily on the spatial distribution of edge pixels in the image. When vehicle occlusion occurs, the area between two vehicles may be more densely distributed with edges than areas of either vehicle. It is possible to identify one vehicle rather than two in this case. We can imagine an extreme case that every falsely identified vehicle corresponds to two or even three actual vehicles on the road. Indeed it was found that the maximum predicted value by block detection is 12, far lower than the maximum number of vehicles (19) in the sample.

Nevertheless, the error of block detection at the free traffic group (0-4 vehicles) is almost half of other individual methods. This verifies that finding local maxima of spatial density of edge pixels is a good way to directly identify vehicles when vehicles are clearly separated in the image. In brief, block detection is strong in free traffic but weak in dense traffic. Unfortunately the magnitude of error in dense traffic is too large to be overcome, causing an unstable overall performance as noticed in Fig.4.1.

Since the error of Simple Sobel Edge Histogram is consistently larger than the Proportionate version, it is omitted in later comparisons.

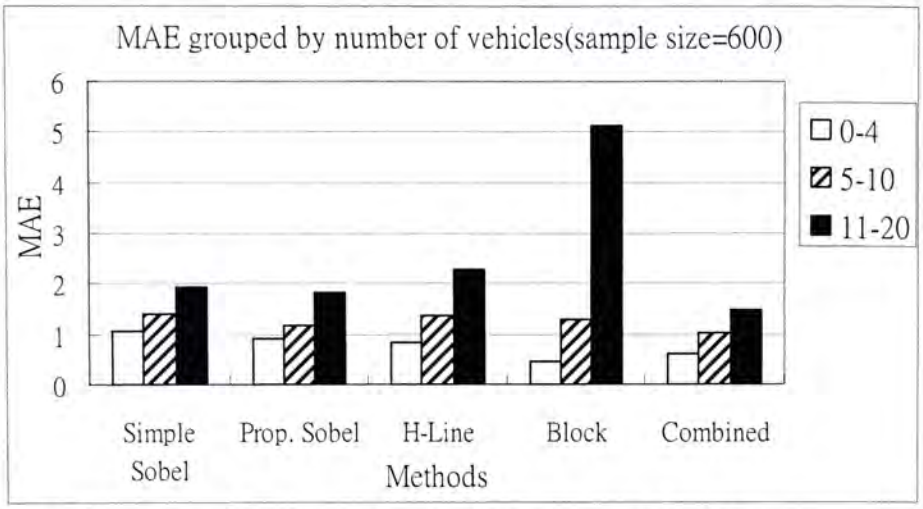


Fig. 4.4 MAE grouped by the number of vehicles for sample size 600

4.3.3 Comparing balanced sampling and random sampling

Balanced sampling requires some additional effort in preprocessing compared with random sampling. It is worthwhile to investigate whether balanced sampling is justified than random sampling. Therefore we compare the differences in prediction accuracy by two methods. The balanced sample of size 600 is compared with two random samples of size 600 and 14000 respectively. From Fig. 4.5, the error for free traffic is lower with random sampling, but for dense traffic the error of random sampling is larger. The differences in accuracy are reflected by the fact that a road is occupied by free traffic for most of the time. Instances of free traffic dominates the random sample and thus random sampling fails to provide enough proportion of dense traffic training data. Our random sample of 600 photos contains only 20 pictures that belong to the group 11-20. In addition, a larger data set of size 14000 was considered (Fig. 4.6). In this sample, 172 pictures belong to the group 11-20. The characteristics of results remain the same. This shows that the small **proportion** of heavy traffic data, rather than small **number**

of heavy traffic instances, causes the low accuracies in prediction for photos that contain a large number of vehicles.

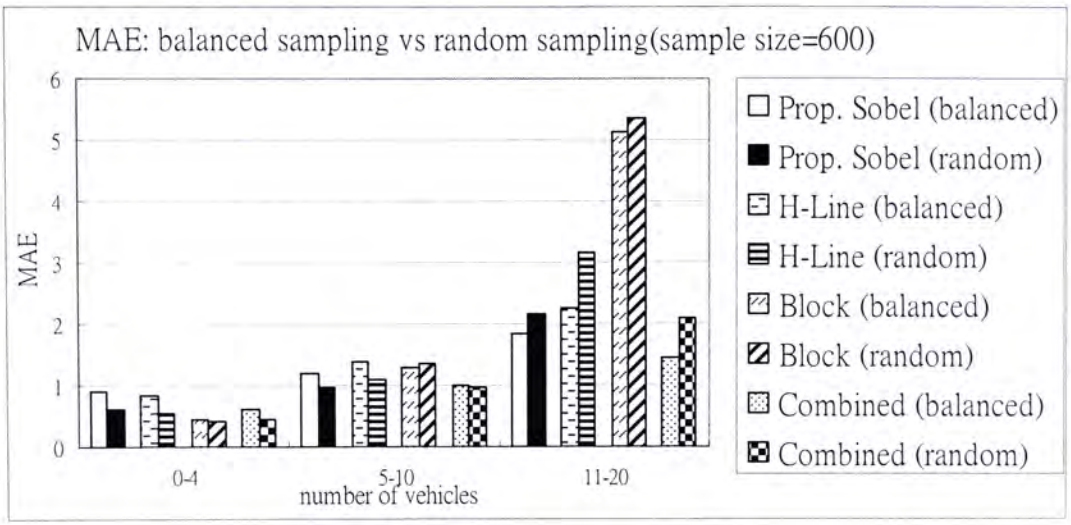


Fig. 4.5 MAE of random sample and balanced sample (sample size=600)

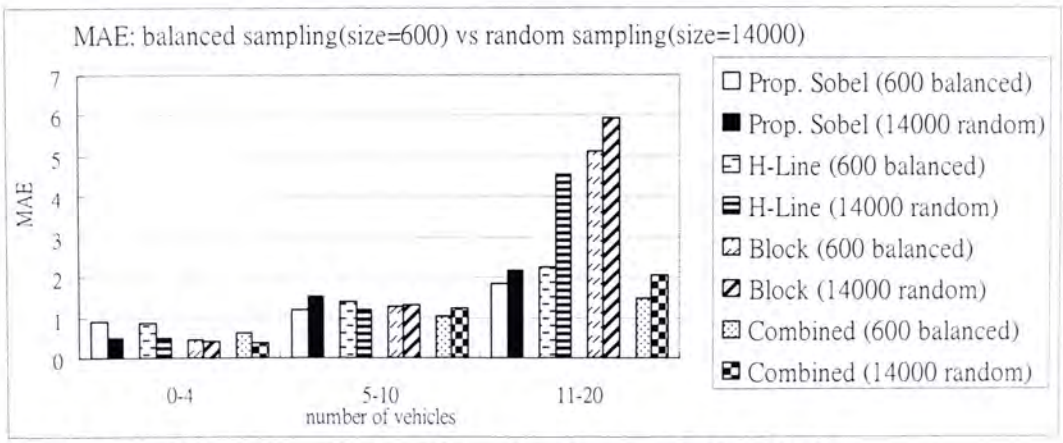


Fig. 4.6 MAE of balanced sample (size=600) and random sample (size=14000)

On the other hand the MAE for combined learning is observed in Fig. 4.5. In group 0-4, balanced sampling results in slightly higher MAE of about 0.25 (0.61 and 0.37 for balanced sampling and random sampling respectively), but in group 11-20,

balanced sampling leads to more significantly lower MAE of about 0.6 (1.46 and 2.06 for balanced sampling and random sampling respectively). This shows that using balanced sampling, the improvement in accuracy for heavy traffic is greater than the reduction in accuracy for free traffic. Thus balanced sampling is preferred over random sampling.

To conclude from above observations, balanced sampling is essential to the system to reduce the errors of prediction. Otherwise the predictive power for dense traffic can diminish significantly.

4.3.4 Comparing day and night conditions

This part tests whether the accuracy of prediction are affected by night conditions. A day sample and a night sample of equal sample size (600) are used for comparison. From Fig. 4.7, the MAEs for night conditions are generally higher than day condition. The source of this difference can be found by examining the edge images. At night, vehicles switch on headlights. Circular edges of headlights can be found in the edge image. However headlights show large contrast to the environment. This makes the outline of vehicles, being relatively in low contrast, unclear in the edge image. Moreover, reflection of headlights on the road show up more at night, which may give rise to additional edges in the edge image.

Proportional Sobel Edge Histogram simply learns the relationship between edges patterns and traffic condition, thus it is comparatively less affected by differences in the edge pattern. But horizontal line detection demonstrates significant difference for night condition. MAE increases from 0.84 to 1.86 for group 0-4, and from 2.27 to 3.80

for group 11-20. The distortion of edges causes horizontal lines to be blurred as distinguishing features of vehicles at night (Fig. 4.8). This suggests that headlights may be better vehicle features than horizontal lines in identifying vehicles at night. For Block Detection, local maxima of edge densities are still locations of vehicles so that the MAE for group 0-4 is similar for both day and night condition. But under heavy traffic, it is not working at all. Combining methods again improves the performance.

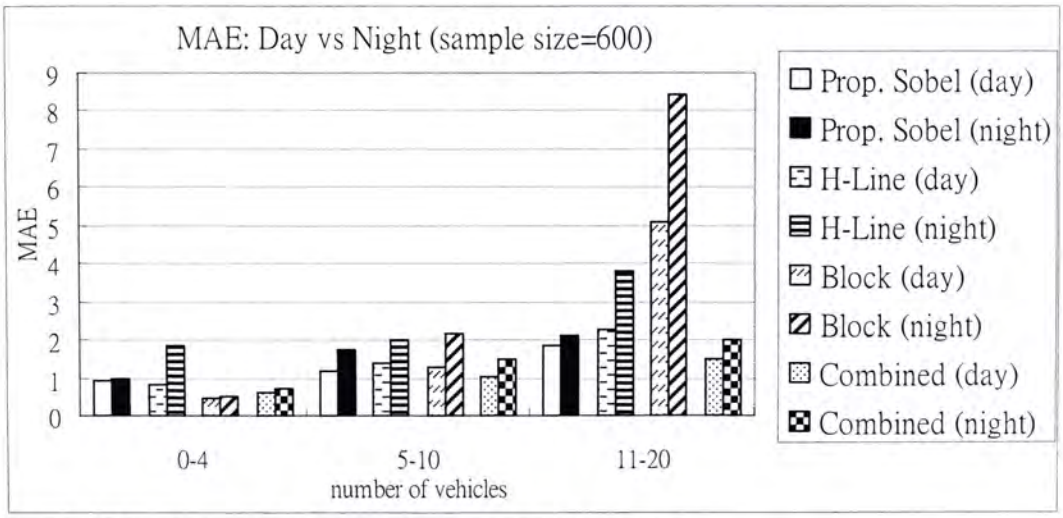


Fig. 4.7 MAE of balanced sample under day and night condition

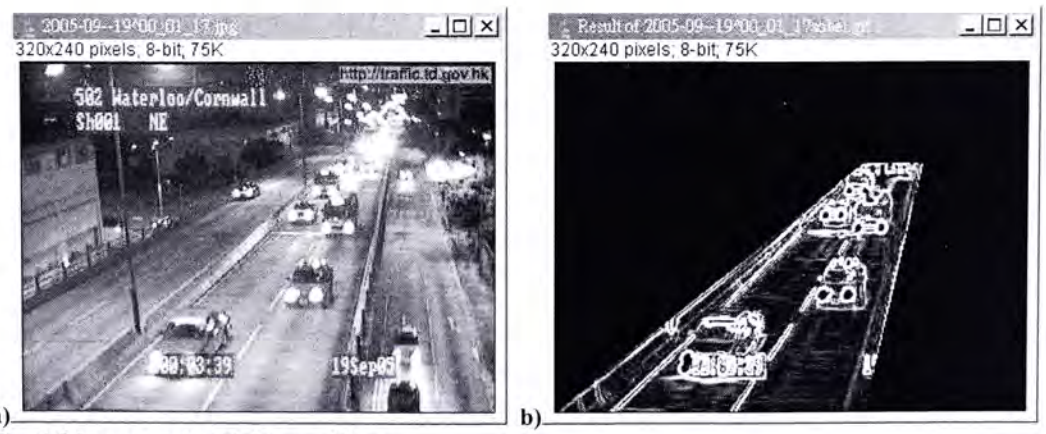


Fig. 4.8 Image under night condition
(a) Grayscale image (b) Edge image with circular edges rather than horizontal edges

4.3.5 Testing on time-series of images

In the following experiment, it is tested whether the system can track the change in traffic condition over time. Two time-series of images are selected from the collected data. Each contains a sequence of more than 60 images, equivalent to 120 minutes, including free-traffic and congested-traffic conditions. The number of vehicles in an image is predicted by combined learning built with a balanced sample of size 600, and plotted in Fig. 4.9 and Fig. 4.11. The 5-period moving average for both predicted and actual values are also plotted in thick trend lines in Fig. 4.10 and Fig. 4.12.

As seen from the figures, the true value of the number of vehicles fluctuates. This shows the deficiency of merely showing one picture for traveler's information as provided by current system in Hong Kong. One single picture can mislead travelers due to large deviation of number of vehicles from the trend. A trend line is probably more suitable for traveler's information. Moreover, it is observed that predicted values and true values follow the same trend. Predictions successfully track the changes in traffic condition for both day and night conditions.

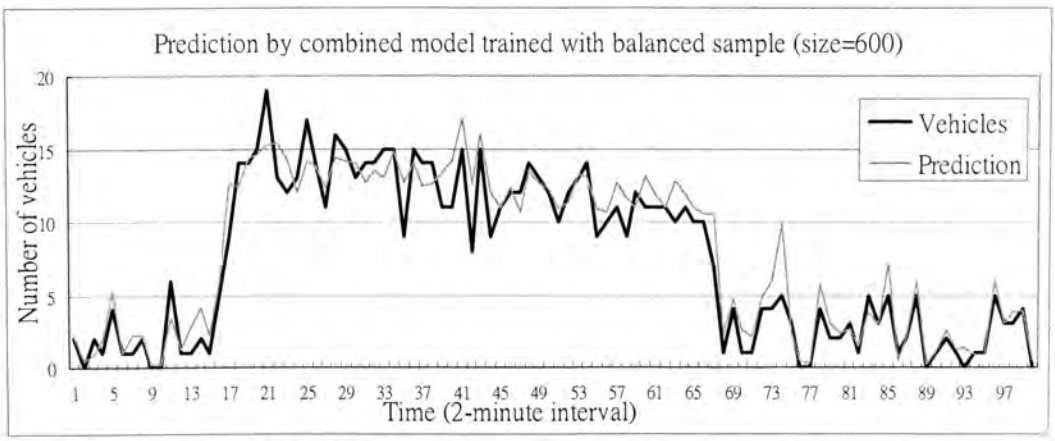


Fig. 4.9 Time series of day condition

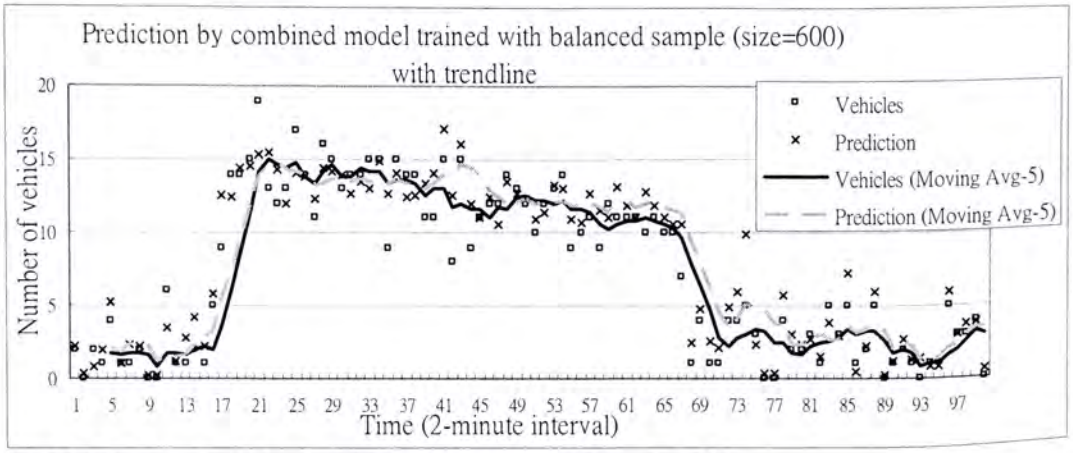


Fig. 4.10 Time series of day condition with moving average trend lines

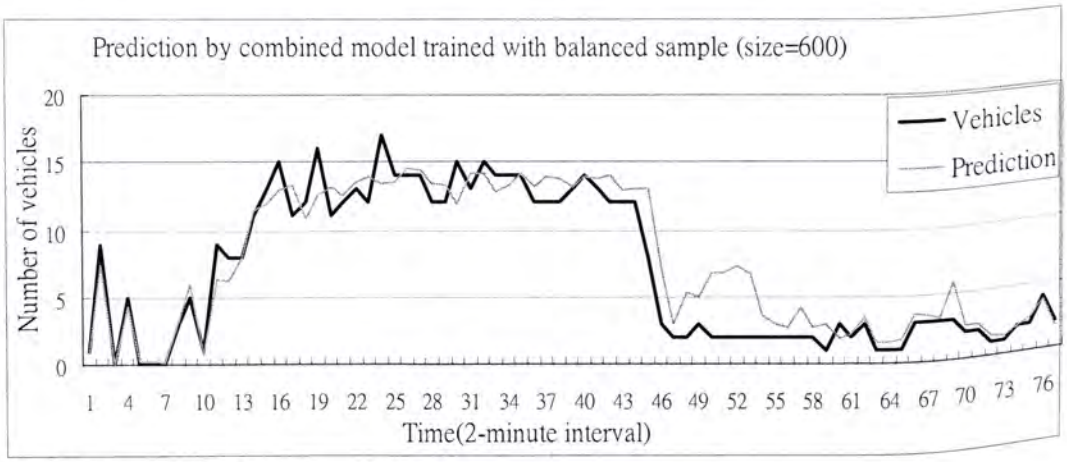


Fig. 4.11 Time series of night condition

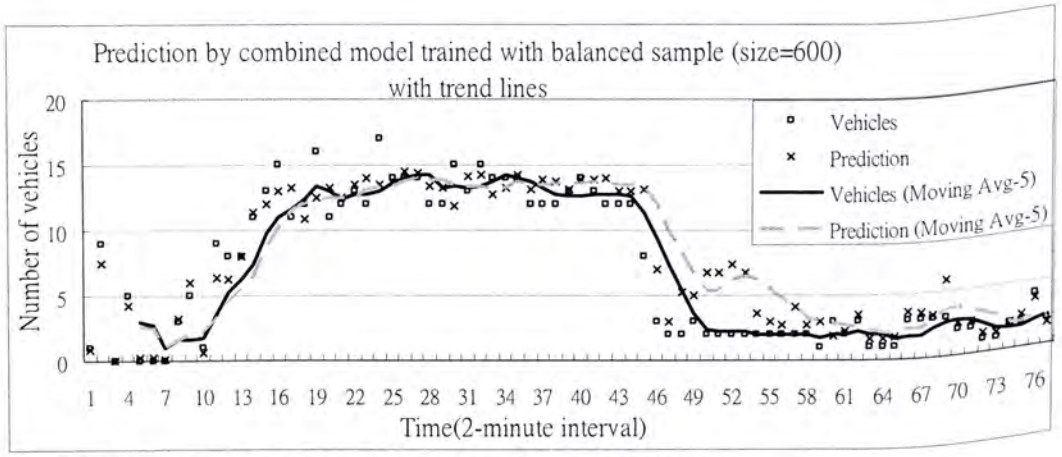


Fig. 4.12 Time series of night condition with moving average trend line

Chapter 5 Analysis

In this chapter the strengths and weaknesses of the several methods introduced in this thesis are discussed and analyzed, together with sample edge images. This helps us to understand the potential and pitfalls of these methods.

5.1 *Strengths and Weaknesses*

5.1.1 Sobel Edge Histogram

This is the simplest method in terms of computation and level of image details. It does not explicitly identify any describable features of the vehicles. Surprisingly, the Proportionate Sobel Edge Histogram is the most adaptive to different edge patterns. Under night conditions, it is still the most accurate prediction method. From section 4.3.1, it is seen that, even with as small a sample size as 150 instances, it is almost the most accurate one. The better result of Proportionate Sobel Edge Histogram over its simple version (Simple Sobel Edge Histogram) proves that weighting the frequencies of histogram can help reduce errors due to different apparent sizes of vehicles in the image.

The drawback of this method lies in its lack of understanding of edges. Vehicles in front of the camera appear larger in the image. This does not only lengthen the edges of vehicles, but also leads to additional edges of other details of vehicles such as logos

or shapes on the vehicle. Without further processing of edges, these additional edges lead to over-estimation of vehicles. Fig. 5.1b shows an example.

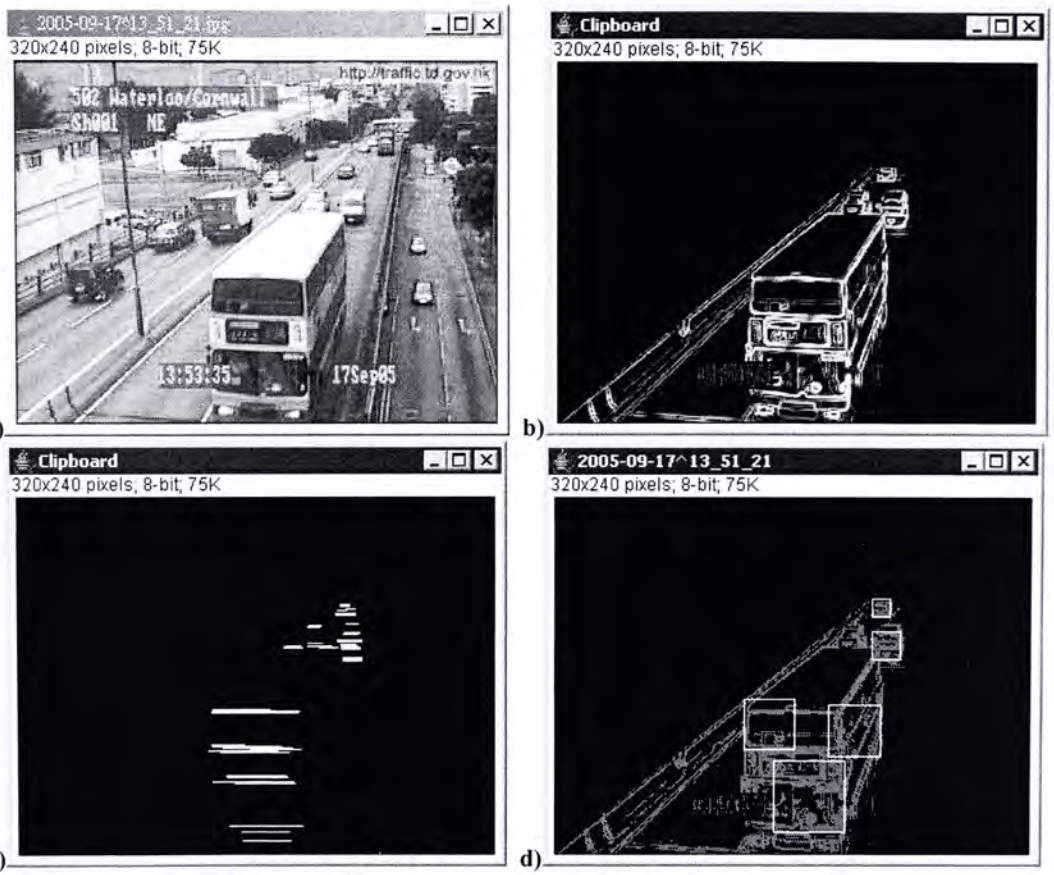


Fig. 5.1 An example of larger vehicle with many edges in front of camera
(a)Grascale image (b) Edge image (c) Horizontal lines detected (d) Blocks detected

5.1.2 Horizontal Line Detection

This model indeed starts from the idea of identifying features from vehicles occluded by others. Straight lines should be the features that every vehicle has. Different shapes and logos will be eliminated in counting straight edges. Learnt with linear regression, a small balanced sample can be good enough to achieve reasonable accuracy. However, there is little room for improvement with increasing sample size. From Fig.5.2, the

number of lines per vehicle shows large variance. Fig. 5.1c shows 8 lines detected from a bus in front of the camera but only 3 lines from the farthest vehicle. This variance is not controlled by sample size. So the ultimate accuracy of this method is limited by the variance of number of lines per vehicle. At night the situation becomes worse because straight lines become unclear in the picture due to headlights.

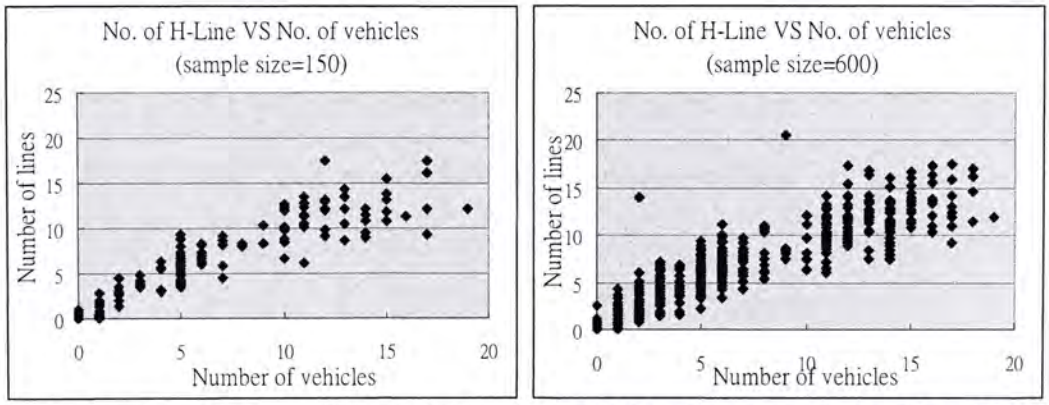


Fig. 5.2 Number of horizontal lines against number of vehicles for two sample size (150 and 600)

5.1.3 Block Detection

This model is similar to many region based vehicle tracking techniques (Section 2.2.1). The only difference is that edges are used instead of a whole blob because our modified background differencing method works on edge features. Extensive computation is unavoidable because calculating edge density for each pixel is required. Its accuracy is superior compared with other methods in free traffic.

However we can see the problem that a local maximum of edge density in the image does not necessarily represent the correct location of a vehicle, especially in heavy traffic. For example, when a vehicle is occluded by another, the edge density of

the area between two vehicles is probably the highest. Block detection then falsely identifies a vehicle somewhere between two vehicles instead of identifying two vehicles at their correct locations. This false location of vehicle could cause incorrect grouping of local maxima in the next step. The problem is magnified for closely packed vehicles, as misplacement of one vehicle can cause a miss or misplacement of the next. As a consequence, serious problem of under-estimation appears in heavy traffic. On the other hand over-estimation may originate from identifying more than one block from a large vehicle. Since an average vehicle length is assumed in grouping local maxima, a large vehicle may be long enough to have two blocks identified as shown in Fig.5.1d. This is particular troublesome for roads with mixed usage, that is , used by both private cars and trucks.

5.1.4 Combined Learning

Methods that combined models incur high computational cost because processing of all the individual models is required. Of course the errors are the smallest among all learning methods. Section 4.3.2 shows that for each group of vehicles, the MAE for the considered method is almost the lowest among other methods. This reveals that a combination model is able to take advantage of block detection for free traffic and that of Proportionate Sobel Edge Histogram for heavy traffic.

5.1.5 Overall Framework

All models show a common characteristic in prediction, which is giving larger errors in prediction for images with a larger number of vehicles. This reveals the limitation of using edge detection for traffic estimation, that is, vehicles occlusion. Edges of some vehicles are blocked by other. Therefore edges increases relatively slowly with the number of vehicles in busy traffic than in free traffic. Fortunately Proportional Edge Histogram method makes use of neural network to find the non-linear relationship between edges and number of vehicles that reduces the estimation error to the minimum. In fact most of other video based systems using background subtraction techniques should be weak in tackling this problem, but some may not need to face this because tracking moving objects from video can choose to deal with only vehicles in front of the camera. Of course they fail also for congested traffic where all vehicles are closely packed together and moving slowly.

Fig. 5.1 demonstrates another problem of the system, variation of vehicle type and size, which is not addressed in our approach. As shown in the pictures, larger vehicles like trucks and buses display more edges and lead to overestimation of number of vehicles. Unfortunately it is far more difficult to identify vehicle types in static image than in video. Yet consider that the "vehicles per km" measurement of density fails to express the effect of different vehicle size to actual traffic density. Therefore some studies calculate density as "passenger car units(pcu) per km", where pcu increases with vehicle size. This means larger vehicles lead to higher traffic density. From this point of view, overestimation of number of vehicles due to larger vehicle size

can possibly give us more accurate density estimation, though this needs further experiments to justify.

Last but not least, the system has lower computation requirement than video based traffic monitoring systems. Only off-line training of models in our system is computationally demanding. On-line operation processes a picture every two minutes, and processing of each picture takes as short time as less than a second in our testing platform. Yet estimated trend always lags actual change of traffic condition because estimation relies on collected data. If more frequent collecting of images is possible, says, one picture per several seconds, the time lag can be reduced. This enables real-time monitoring or even management of traffic. For example, if busy traffic is detected in a region, traffic signals can be coordinated to prevent drivers from encountering a long string of green lights towards the region. This practice discourages high volumes of traffic by inducing delay yet preventing congestion.

5.2 *Future Research*

5.2.1 Static image based monitoring combined with other traffic monitoring approaches

As analyzed in Section 2.2.1, moving object tracking may still be subject to failure under congested traffic. Background update in video-based monitoring may fail as the road is continuously covered by vehicles, preventing updated background colors to be captured. In addition vehicle occlusion causes to error in identifying vehicles. Our system which is based on static picture is able to function properly even when motion

tracking fails. The system can still estimate the traffic density even the traffic is busy. Moreover, by combining different sources of traffic monitoring, the mapping between the number of vehicles from our system and traveling speed and travel time from video or loop detectors can be found. This may further allows estimation of traffic speed, flow and density from only static images. For packages like ILog traffic management tools, one more source of traffic estimation means more reliable traffic analysis.

5.2.2 Horizontal Line Detection as tracked features of vehicles

From related literature, it is found that features of vehicles can be basic units of moving object tracking. Section 2.2.1 introduces some features such as corner points or headlights of vehicles. Horizontal Line can be vehicle features as well, sharing the advantage that a partly occluded vehicle can be recognized as some of its features can be still found. Furthermore, Horizontal Line could be better features than others in daylight conditions. For example, reflection on the roads at night may cause false recognition by additional corner points and light spots. But using horizontal line detection reduces this probability. This is because in this method the direction of straight edges detected are perpendicular to the direction of the road. Non-vehicle objects may lead to different edges on the road but seldom exhibit such straight edges.

5.2.3 Application in aerial image-based system

As the limitation of this system is vehicle occlusion, a lower error rate can be expected if this problem can be avoided. The aerial image-based system suits this case because images are taken almost 90 degrees down from the sky to the road. Satellite images like

those provided by Google Earth are similarly applicable. Using satellite images, one snapshot may be used to monitor several roads in a region. Moreover, they can be applied to highways in remote areas where installation of cameras and connection to a data network is difficult.

Chapter 6 Conclusion

In this thesis, a static image-based traffic-monitoring system is proposed. The system is capable of analyzing the traffic condition from static pictures collected at a fixed view. Not only can the number of vehicles be identified, which is an indicator of traffic density, but also the trend of traffic condition can be estimated by studying a time series of images. Edges in the images are extracted as the key features to identify the existence of vehicles. Several learning methods are experimented to investigate the relationship between edges and traffic condition. Using a combined learning model, the mean absolute error errors with balanced sample size 600 are 0.95 vehicles for day condition and 1.4 vehicles for night condition. The accuracy is mainly limited by the occlusion of vehicles which is also faced by video-based monitoring system. Yet our framework can be applied to systems without such problem such as aerial image-based traffic monitoring; in which case, better performance can be expected. Moreover this system may be integrated with other systems such as video-based monitoring and loop detector-based monitoring for more accurate estimation of traffic parameters. If more frequent update of images is allowed, our system can even facilitate real-time traffic control by estimating the current traffic condition..

Bibliography

[1] Hall, F. and Persaud, B. (1989) *Evaluation of Speed Estimates Made with Single-Detector Data from Freeway Traffic Management Systems.* Transportation Research Record 1232, pp. 9-16.

[2] Pushkar, A., Hall, F.L., and Acha-Daza, J.A. (1994) *Estimation of Speeds from Single-loop Freeway Flow and Occupancy Data Using Cusp Catastrophe Theory Model.* Transportation Research Record 1457, pp. 149-157.

[3] Petty, K., Bickel, P., Jiang, J., Ostland, M., Rice, J., Ritov, Y., and Schoenberg, F. (1998) *Accurate Estimation of Travel Times from Single-Loop Detectors.* Transportation Research 32A (1), pp. 1-17.

[4] Coifman, B. and Cassidy, M. (2002) *Vehicle Reidentification and Travel Time Measurement on Congested Freeways.* Transportation Research 36A (10), pp. 899-917.

[5] Sun, C., and Ritchie, S.G. (1999) *Individual Vehicle Speed Estimation Using Single Loop Inductive Waveforms.* Journal of Transportation Engineering, American Society of Civil Engineers, vol 125, no 6, pp. 531-538.

[6] Dailey, D. J. (1993) *Travel-time estimation using cross-correlation techniques.* Transportation Research 27B(2), pp. 97-107.

[7] Petty, K., Bickel, P., Jiang, J., Ostland, M., Rice J., Ritov, Y., and Schoenberg, F. (1998). *Accurate estimation of travel times from single-loop detectors.* Transportation Research A 32(1), pp.1-17.

[8] Rakha, H., Aerde, M.V. (1995) *Accuracy of Vehicle Probe Estimates of Link-Travel Time and Instantaneous Speed,* ITS America Conference, Ann Arbor MI, March 1995.

[9] Larson, J.E., Cheng, H.H., Shaw, B.D., Palen, J., Hu, X., Katwyk K.V. (1998) *A real-time laser-based prototype detection system for measurement of delineation of moving vehicles.* MOU 263 Working paper to California PATH, University of California, Davis.

[10] Foresti, G.L., Micheloni, C., Snidaro L. (2003) *Advanced visual-based traffic monitoring systems for increasing safety in road transportation.* Advances in Transportation Studies, Section A 1

[11] Angel, A., Hickman, M., Mirchandani, P., and Chandnani, D. (2002) *Methods of traffic data collection using aerial video.* IEEE 5th International Conference on Intelligent Transportation Systems, Sept. 2002, Singapore, pp.31-36.

[12] Lan, L.W., Kuo, A.Y., Huang, Y. (2003) *Color Image Vehicular Detection Systems with and without fuzzy neural network: a comparison.* Journal of the Chinese Institute of Engineers, vol. 26, no. 5, pp. 659-670.

[13] Gupte, S., Masoud, O., Martin, R. F. K., Papanikolopoulos, N.P. (2002) *Detection and Classification of Vehicles.* IEEE Transactions on Intelligent Transportation Systems, vol.3, no. 1.

[14] Koller, D., Weber, J., Malik, J. (1993) *Robust Multiple Car Tracking with Occlusion Reasoning.* Technical Report UCB/CSD 93/780, Computer Science Division (EECS), University of California, Berkeley.

[15] Koller, D., Daniilidis, K., Nagel, H.-H. (1993) *Model-based Object Tracking in Monocular Image Sequences of Road Traffic Scenes.* International Journal of Computer Vision, vol. 10, pp.257-281.

[16] Cucchiara, R., Piccardi, M. (2000) *Image Analysis and Rule-Based Reasoning for a Traffic Monitoring System.* IEEE Transactions on Intelligent Transportation systems, vol. 1, No. 2.

[17] Dailey, D.J. (2004) *An Algorithm to Estimate Mean Traffic Speed Using Uncalibrated Cameras.* IEEE Transactions on Intelligent Transportation Systems, vol. 1, no. 2.

[18] Coifman, B., Beymer, D., McLauchlan, P., Malik, J. (1998) *A real-time computer vision system for vehicle tracking and traffic surveillance.* Transportation Research 6C, pp.271-288.

[19] Kilger., M. (1992) *A shadow handler in a video-based real-time traffic monitoring system.* IEEE Workshop on Applications of Computer Vision, Palm Springs, CA, pp.1060-1066.

[20] Friedman, N., Russel, S. (1997) *Image segmentation in video sequences: A probabilistic approach.* Proceedings 13 Conference on Uncertainty in Artificial Intelligence, pp.175-181.

[21] Otsu, N. (1979) *A threshold selection method from gray-level histograms.* IEEE Trans. SMC, 9:62-66.

[22] Rosin, P.L. (2001) *Unimodal thresholding.* Pattern Recognition 2001: 34, pp.2083-296

[23] Huang, Y.W., Chien, S.Y., Hsieh, B.Y., Chen, L.G. (2002) *Automatic threshold decision of background registration technique for video segmentation.* Proceedings of Visual Communications and Image Processing 2002, pp. 552-563.

[24] Yoshida., T. (2004) *Background Differencing Technique for Image Segmentation Based on the Status of Reference Pixels*, Proceedings of the 2004 International Conference on Image Processing, pp. 3487-3490

[25] Canny J. (1986) *A Computational Approach to Edge Detection.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 8, no 6.

[26] Sezgin, M., Sankur., B. (2004) *Survey over image thresholding techniques and quantitative performance evaluation*. Journal of Electronic Imaging 13(1) pp.146-165.

[27] Zhou, Q., Aggarwal, J. (2001) *Tracking and classifying moving objects from videos*. Proceedings of IEEE Workshop on Perfomance Evaluation of Tracking and Surveillance 2001.

[28] Stauffer, C., Grimson, W. (2000) *Learning patterns of activity using real-time tracking*. Pattern Analysis and Machine Intelligence, IEEE Transactions, vol. 22, Issue 8, 2000, pp.747-757.

[29] Cheung, S., Kamath. C. (2004) *Robust techniques for background subtraction in urban traffic video*. Proceedings of the SPIE, vol. 5308, pp. 881-892.

[30] Yu, M., Jig, G., Yu, B.(2000) *An Integrative Method for Video Based Traffic Parameter Extraction in ITS*, The 2000 IEEE Asia-Pacific Conference on Circuits and Systems, pp. 136-139.

[31] X. Liu, X., Yao, D., Cao, L., Peng, L., Zhang., Z. (2001) *A feature-based real-time traffic tracking system using spatial filtering*. 2001 IEEE Intelligent Transportation Systems Conference Proceedings, pp. 514-518.

[32] Quinlan, J. R. (1992) *Learning with Continuous Classes*. Proceedings of the Australian Joint Conference on Artificial Intelligence, World Scientific, Singapore. pp.343–348.

[33] Wang, Y and Witten, I. H. (1997). Induction of model trees for predicting continuous classes. Proceedings of the poster papers of the European Conference on Machine Learning. University of Economics, Faculty of Informatics and Statistics, Prague.

Appendix A Sobel Edge Detection

The procedure for Sobel Edge Detection are shown as follows given a grayscale image.

1. The image undergoes convolution, a kind of image processing techniques using the following convolution kernels *Kx* and *Ky*, which are 3 by 3 matrices. The resultant values *Gx* and *Gy* represents the magnitude of color differences across the horizontal direction and vertical direction respectively. The mathematical representation of the convolution is given by equation (23a) and (23b).

-1	0	+1
-2	0	+2
-1	0	+1

Kernal *Kx* for calculating *Gx*

+1	+2	+1
0	0	0
-1	-2	-1

Kernal *Ky* for calculating *Gy*

p(i,j): pixel value at coordinate *(i,j)* of the input image.

$$Gx(i,j) = \sum_{k=1}^{3}\sum_{l=1}^{3} p(i+k-1, j+l-1)Kx(k,l) \qquad (23a)$$

$$Gy(i,j) = \sum_{k=1}^{3}\sum_{l=1}^{3} p(i+k-1, j+l-1)Ky(k,l) \qquad (23b)$$

2. In the Sobel image, the pixel value *G(i,j)* of each pixel *(i,j)* is calculated using the following equation:

$$G(i,j) = \sqrt{Gx(i,j)^2 + Gy(i,j)^2}$$

For pixels surrounded by differently colored pixels, their *G(i,j)* are large. As a result, edges have high values of *G(i,j)* in the Sobel image, appearing as bright lines in the image. In contrast non-edges occur as dark regions in the Sobel image.

Appendix B Neural Network Setup

The neural network for Sobel Edge Detection Module is the commonly used feed-forward multi-layer neural network, trained with backpropagation method. There are 3 layers in our neural network: input layer, hidden layer and output layers. In the input layer, there are 256 nodes, each with input values from Sobel Edge Histogram $h_E(v)$ for each pixel value v. In the middle layer, a sigmoid function, denoted $f(I_j)$ below, is used to implement non-linear mapping between input values and output values. In the output layer there is only one node with its output value representing the number of vehicles in the picture. Each node in input layer is connected to nodes in the hidden layer, which are in turn connected to the output layer node. Using 10-fold cross-validation evaluation, 10% of the instances belong to test set. Out of the remaining 90%, 80% are training set data and 20% are validation set. The training procedure is described as follows:

Ij : value input to node j
Oj : value output from node j
w_{ij} : weight of connection from node i to node j
θ_j : bias value of node j
y : true number of vehicles (normalized to be between -1 and 1)
Err_j : error of node j in hidden layer
l : learning rate between 0 to 1, parameter
n : stopping threshold of training

1. $h_E(v)$ is extracted from one instance in training set data. In the input layer 256 histogram values, each $h_E(v)$ are fed into O_j of each input node. Then by calculating O_j and I_j of each node (both hidden layer and output layer) with the following equations, O_j of the node in output layer is obtained as the predicted value:

$$O_j = h_E(j) \qquad \text{for input layer nodes}$$

$I_j = \sum_i w_{ij} O_i + \theta_j$ for hidden layer and output layer nodes, where I are nodes in the previous layer

$O_j = f(I_j) = \frac{1}{1+e^{-I_j}}$ for hidden layer and output layer nodes ($f(I_j)$ is known as sigmoid function)

2. Error of prediction is estimated for each node:

$Err_j = O_j(1-O_j)(y-Oj)$ for output layer node

$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$ for input and hidden layer nodes, where k are nodes in the next layer

3. The errors are used to update the weight and bias values:

$w_{ij} = w_{ij} + (l) Err_j O_i$

$\theta_j = \theta_j + (l) Err_j$

4. The above steps are repeated using instances from training set until there is no reduction in error predicting the validation set for consecutive c number of cycles.

Tuning of parameters

w_{ij} *and* θ_j are initialized to 0 for every training. Clementine, a well-known data mining package, is used to determine the number of nodes in the hidden layer by pruning method. The pruning method first constructs a network with large number of hidden layer nodes. Then it tries to reduce the number of nodes and finds one which yields the best result. After that c and l are tuned by trying different values. Values of l tested include 0.1 to 0.9 for every interval of 0.1. c is tested from 50 to 500 (cycles) for every interval of 50. In our case the number of nodes in hidden layer is 20, and l=0.3, c=250.

Appendix C Numerical Results

Table 1 MAE for different sample size (balanced sampling)

	Sample size (Day)			Sample size (Night)		
Sample size	150	300	600	150	300	600
Simple Sobel Edge Histogram	1. 84953	1. 49603	1. 41407	2. 03038	1. 88197	1. 66030
Proportionate Sobel Edge Histogram	1. 56093	1. 33089	1. 22844	2. 03003	1. 79354	1. 62276
Horizontal Line Detection	1. 56093	1. 45793	1. 36953	2. 54428	2. 63887	2. 54856
Block Detection	1. 43725	2. 16000	1. 86000	3. 68667	3. 69667	3. 70000
Combined Learning	1. 90667	0. 95771	0. 95455	1. 69821	1. 53708	1. 40072

Table 2 RMSE for different sample size (balanced sampling)

	Sample size (Day)			Sample size (Night)		
Sample size	150	300	600	150	300	600
Simple Sobel Edge Histogram	2. 43230	2. 08293	1. 8867	3. 04172	2. 44674	2. 27587
Proportionate Sobel Edge Histogram	2. 12535	1. 81092	1. 71161	3. 07356	2. 31124	2. 22656
Horizontal Line Detection	2. 12535	2. 04652	1. 99550	3. 29909	3. 39453	3. 29931
Block Detection	2. 00647	3. 28634	2. 96142	5. 46077	5. 30566	5. 38609
Combined Learning	2. 94845	1. 37284	1. 36881	2. 57495	2. 38007	2. 00191

Table 3 PMAE for different sample size

	Sample size (Day)			Sample size (Night)		
	150	300	600	150	300	600
Simple Sobel Edge Histogram	28. 08%	21. 90%	23. 69%	29. 28%	27. 24%	24. 08%
Proportionate Sobel Edge Histogram	23. 70%	19. 49%	20. 58%	29. 28%	25. 96%	23. 54%
Horizontal Line Detection	23. 70%	21. 35%	22. 94%	36. 70%	38. 19%	36. 96%
Block Detection	21. 82%	31. 63%	31. 16%	53. 17%	53. 50%	53. 66%
Combined Learning	28. 95%	14. 02%	15. 99%	24. 49%	17. 90%	20. 31%

Table 4 MAE grouped by number of vehicles (Day Samples, balanced sampling and random sampling)

		Simple Sobel Edge Histogram	Proportionate Sobel Edge Histogram	Horizontal Line Detection	Block Detection	Combined Learning
Balanced Sampling (size=600)						
Group	0-4	1. 05947	0. 91720	0. 84128	0. 44095	0. 61937
	5-10	1. 41409	1. 18748	1. 38499	1. 28500	1. 0066
	11-20	1. 92851	1. 82599	2. 26735	5. 11644	1. 46635
Random Sampling (size=600)						
Group	0-4	0. 68977	0. 61779	0. 55770	0. 40509	0. 45436
	5-10	1. 36722	0. 97804	1. 09237	1. 34722	0. 96722
	11-20	1. 89755	2. 15969	3. 14729	5. 35294	2. 10310
Random Sampling (size=14000)						
Group	0-4	0. 67713	0. 48973	0. 47409	0. 40201	0. 37164
	5-10	1. 91013	1. 49874	1. 18163	1. 30830	1. 20850
	11-20	2. 58293	2. 17209	4. 55090	5. 94521	2. 05765

Table 5 MAE grouped by number of vehicles (Day and Night)

		Simple Sobel Edge Histogram	Proportionate Sobel Edge Histogram	Horizontal Line Detection	Block Detection	Combined Learning
Day Sample (size=600)						
Group	0-4	1. 05947	0. 91720	0. 84129	0. 44095	0. 61937
	5-10	1. 41410	1. 18749	1. 38499	1. 28500	1. 00663
	11-20	1. 92852	1. 82600	2. 26735	5. 11644	1. 46635
Night Sample (size=600)						
Group	0-4	0. 94359	1. 00063	1. 85839	0. 5400	0. 70809
	5-10	1. 84871	1. 76750	1. 98912	2. 1400	1. 51207
	11-20	2. 18861	2. 10015	3. 79818	8. 4200	1. 98200

Table 6 MAE with different background update parameter α

In this test different values of α are used in background update module and the performance of Proportionate Sobel Edge Histogram is evaluated using a balanced sample of 600 pictures. "α=0" refers to no update at all after constructing the first background image.

Value of α	*no background subtraction	0	0. 02	0. 05	0. 1
MAE	1. 522	1. 452	1. 434	1. 278	1. 384

*In "no background subtraction", no background image is constructed and background subtraction is not applied in the Background Detection Module.