國立交通大學

生醫工程研究所

碩 士 論 文

結合局部與全域特徵之新型車輛偵測系統

A Novel Vehicle Detection System Using Local and Global Features

研 究 生：李佳芳

指導教授：林進燈　教授

中 華 民 國 九 十 九 年 七 月

結合局部與全域特徵之新型車輛偵測系統
A Novel Vehicle Detection System Using Local and Global Features

研 究 生：李佳芳　　Student：Ja-Fan Lee

指導教授：林進燈　　Advisor：Chin-Teng Lin

國 立 交 通 大 學

生醫工程研究所

碩 士 論 文

A Thesis

Submitted to Institute of Biomedical Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

July 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年七月

結 合 局 部 與 全 域 特 徵 之 新 型 車 輛 偵 測 系 統

學生：李佳芳　　指導教授：林進燈 教授

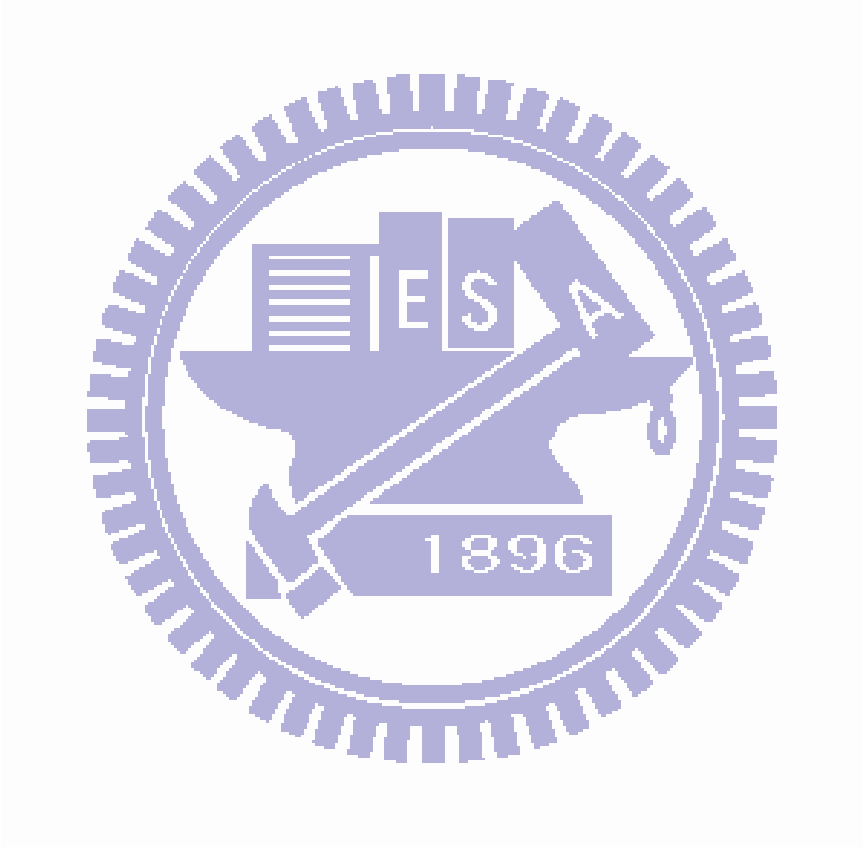
國立交通大學 生醫工程研究所碩士班

摘　　要

近年來，基於影像式的車輛偵測技術在智慧型運輸系統中受到廣泛的重視與研究。然而，在交通流量大的場景偵測車輛仍然是一個困難且具有挑戰性的問題。在本研究中，我們提出了一個創新與可靠的自動化偵測系統。本系統先以車輛的邊緣強度與對稱性統計方法來假設車輛的位置以減少運算成本。接下來，以 AdaBoost 與決策型類神經網路 (Probabilistic Decision-Based Neural Network)分類器分別對車輛的局部與全域特徵來進行確認車輛的位置，我們相信由兩個不同性質的分類器在特徵擷取有互補的效果，可以同時降低誤報率與達到高偵測率的效果，由我們蒐集的樣本資料庫測試結果，我們的系統可達到 96.12%的偵測率，同時只產生 0.0153%的誤報率；另外對公開的 MIT CBCL 資料庫測試，我們的系統可達到 96.3%的偵測率，同時只有 0.0013%的誤報率。

本研究的目標是在局部與全域上抽取出車輛的特徵，由此系統所得到的結果可以在車輛偵測的技術中，針對以背景為基礎的車輛之偵測系統所遭遇到的問題提供更好的解決方案。實驗的結果證實我們提出的系

統可以不依賴背景資訊，偵測出影像中的車輛，我們所實作的系統也能為後續處理，e.g. 車輛追蹤、計數、分類、識別等應用，提供有用的資訊。



A Novel Vehicle Detection System Using Local and Global Features

Student：Ja-Fan Lee　　Advisors：Prof. Chin-Teng Lin

Degree Program of Computer Science
National Chiao Tung University

ABSTRACT

Vehicle detection techniques in visual-based Intelligent Transportation System (ITS) have been studied for years. However, to detect vehicles in a scene with heavy traffic is still a challenging problem. In this study, we present a novel automatic vehicle detection system. It first hypothesize potential locations of vehicles to reduce the computational costs by statistic of edge intensity and symmetry, then verify the correctness of the hypotheses using AdaBoost and Probabilistic Decision-Based Neural Network (PDBNN) classifiers, which exploits local and global features of vehicles respectively. The combination of two classifiers can learn the complementary relationship among local and global features, and it gains the extremely low false positive rate while still keeps high detection rate. For proprietary database, a 96.12% detection rate leads to a false-positive rate of approximately 0.0153%. For the MIT CBCL database, a 96.3% detection rate leads to a false-positive rate of approximately 0.0013%.

The objective of this study is to extract the characteristic of vehicles in both local- and global-orientation, and model the implicit invariance of vehicles. This novel approach would provide a better solution to handle the problems encountered by conventional background-based detection systems. The experimental results proved the proposed system achieved a good performance of detecting vehicles without background information. The implemented system also extracted useful traffic information that can be used for further processing, like tracking, counting, classification and recognition.

致　謝

本論文的完成，首先要感謝指導教授林進燈博士這兩年來的悉心指導，讓我學習到許多寶貴的知識，在學業及研究方法上也受益良多。另外也要感謝口試委員們的建議與指教，使得本論文更為完整。

其次，感謝超視覺實驗室的大家長鶴章，還有剛維、建霆、肇廷、東霖與勝智學長，提供許多寶貴的意見與耐心的指導。健豪、嶸健、聖傑及哲男同學的相互砥礪，及所有學長、學弟們在研究過程中所給我的鼓勵與協助。尤其是建霆與勝智學長，在理論及實作技巧上給予我相當多的幫助與建議，讓我獲益良多。

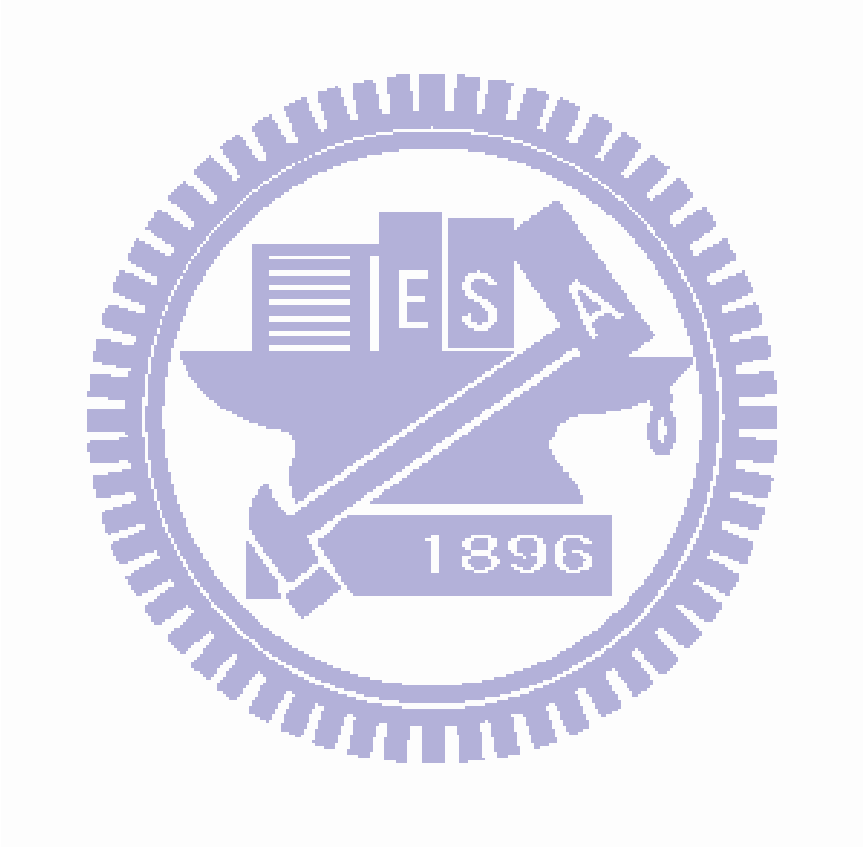
感謝我的家人對我的關心與鼓勵，我的奶奶、外婆、爸媽、姊姊與弟弟，在這段期間給予了我精神及物質上百分百的支持，讓我能夠專心致力於學業，也感謝朋友們的陪伴，更豐富這兩年的求學時光。

謹以本論文獻給我的家人及所有關心我的師長與朋友們。

Table of Contents

List of Figures

List of Tables



Chapter 1
Introduction

1.1 Motivation

In recent years, utilizing video processing to help for improving safety or human's life has attracted great attention in computer vision. For example, how to improve and control the traffic condition with advanced techniques is one of the most important missions among the developed countries. Traditional traffic surveillance systems often use sensors to detect passing of vehicles and gather simple information or use cameras and manually check the video when some events happened. Those methods are inefficient in information extraction and short of improving. Therefore, visual-based Intelligent Transportation System (ITS) has attracted great attention in computer vision recently. For example, video-based automatic surveillance, behavior analysis for accident prediction, traffic monitoring etc. are presented in many application system.

For most of these application systems, foreground object extraction is a very fundamental and important step before further processing, like tracking, classification and recognition. In conventional method, background subtraction and temporal difference are usually used for foreground segmentation. However there are some factors that may affect the result of foreground segmentation and make foreground detection very challenging. Background subtraction which relies on built background to extract foreground objects in a frame is not efficient enough in some circumstances. For example, swaying trees might be treated as foreground objects, but actually they are not the objects of interesting, moreover, the background construction relies on the frequent appearance of true information, in other words, the background is difficult to be built when there is too heavy traffic since the

frequency of foreground is higher than that of background. Though the background information can be built in advance, it is prone to be tainted by the still vehicles or the heavy traffic. Temporal difference which relies on the motion and texture of objects fails if the texture of objects is smooth. Furthermore, it malfunctions when the objects keep still periodically.

1.2 Objective

The objective of this study is to analyze the characteristic of vehicles in local and global features. We hope this novel approach would provide a better solution in the field of vehicle detection to handle the problems encountered by conventional background-based detection systems. Therefore, we propose to develop a vehicle detection system that has these characteristics:

1) Can detect vehicles in static images.

2) Can work without background information.

3) Can detect vehicle in a scene with heavy traffic.

1.3 Organization

This thesis is organized as follows: Chapter II gives an overview of related work about this research and introduces the research by different modules. Chapter III presents the main modules of proposed system, and details algorithm used in each module. Chapter IV shows experimental results and performance comparison. Finally, the conclusions of this study are stated in Chapter V.

Chapter 2
Related Works

A kind of solution for vehicle detection without background model is the exhaustive search at all positions in the image. This solution is not satisfactory for real-time applications. To attack this problem, most of the methods reported in the literature can be decomposed into two steps as follow.

(i) Hypothesis Generation (HG): this step provides potential positions of vehicles in a simple and rapid way resulting in a reduced area to search.

(ii) Hypothesis Verification (HV): residual candidate regions in HG step are verified by using some complex algorithms to validate the exact positions of vehicles.

Some researches related with these two steps are introduced.

2.1 Hypothesis Generation (HG) Methods

Various HG approaches have been proposed in the literature. The objective of the HG step is to find candidate vehicle locations in an image quickly so that it can reduce the computational requirements for further searching. It is generally based on simple, low-level algorithms which hypothesize potential locations of vehicles. The hypothesized locations from the HG step form the input to the HV step, where tests are performed to verify the correctness of the hypotheses. Clearly, the principle of HG is to filter out unqualified searching windows as many as possible while keeping overall detection rate as high as possible.

Due to the rear- or frontal-views of vehicles are in general symmetric in horizontal direction, A. Bensrhair et al. [1], A. Kuehnle [2] and T. Zielke et al. [3] used symmetry, as

one of the main features of artificial objects, to hint the existence of vehicles in their studies. S.D. Buluswar et al. [4] and D. Guo et al. [5] used color information in RGB and L*a*b space respectively to segment vehicles from background. Bertozzi et al. [6] proposed a corner-based method to hypothesize vehicle locations. Matthews et al. [7] used edge detection to find strong vertical edges. By computing the vertical profile of the edge image (i.e., by accumulating the pixels in each column) and smoothing using a triangular filter, the local maximum peaks of the vertical profile cues the left and right borders of a vehicle. Goerick et al. [8] proposed Local Orientation Coding (LOC) to extract edge information. An image obtained by this method consists of strings of binary code representing the directional gray-level variation in the pixel's neighborhood. These codes carrying essentially edge information are used as the hypothesis. The presence of vehicles in an image causes local intensity changes. This property can be used as a cue to narrow down the search area for vehicle detection. U. Handmann et al. [9] proposed that the intensity changes follow a certain texture pattern, and entropy was used as a measure for texture detection. For each image pixel, a small window was chosen around it, and the entropy of that window was considered as the entropy of the pixel. Only regions with high entropy were considered for further processing. Besides, optical flow can provide strong information for HG. C. Demonceaux et al. [10] and A. Giachetti et al. [11] use optical flow to distinguish motion of any objects including preceding vehicles from the road motion and segment the objects.

Combining multiple cues should also be explored more actively as a viable means to develop more reliable and robust systems. The main motivation is that the use of a single cue suitable for all conceivable scenarios seems to be impossible. Combining different cues has produced promising results (e.g., combining LOC, entropy, and shadow [9], shape, symmetry, and shadow [12], color and shape [13], and motion with appearance [14]). Effective fusion mechanisms as well as cues that are fast and easy to compute are important research issues.

2.2 Hypothesis Verification (HV) Methods

The input to the HV step is the set of hypothesized locations from the HG step. During HV, tests are performed to verify the correctness of a hypothesis. A. Khammari et al. [15] classified HV methods into two categories: 1) template-based and 2) appearance-based.

Template-based methods use predefined patterns of the vehicle class and perform a correlation between an input image and the template. M. Betke et al. [16] proposed a multiple vehicle detection approach using deformable gray-scale template matching. J. Ferryman et al. [17] proposed a deformable model formed from manually sampled data using Principal Component Analysis (PCA). Both the structure and pose of a vehicle can be recovered by fitting the PCA model to the image.

Appearance-based methods acquire the characteristics of the vehicle class from a set of training images which capture the variability in vehicle appearance. Usually, the variability of the non-vehicle class is also modeled to improve performance. First, each training sample is represented by a set of local or global features. Then, the decision boundary between the vehicle and non-vehicle classes are learned either by training a classifier (e.g., Support Vector Machine (SVM) [18], Neural Network (NN) [19, 20], etc.) or by modeling the probability distribution of the features in each class (e.g., using the Bayes rule assuming Gaussian distributions [21]). In N. Matthews et al. [7], feature extraction is based on PCA. C. Goerick et al. [8] used a method called Local Orientation Coding (LOC) to extract edge information. The histogram of LOC within the area of interest was then fed to a NN for classification. In [22], wavelet transform was used for feature extraction and Support Vector Machines (SVMs) was used for classification.

Most research efforts have focused on feature extraction and classification based on learning and statistical models. Efforts in this direction should continue while capitalizing on recent advances in the statistical and machine learning areas. Basically, the most important

issue in object detection literature is selecting a good set of features. In most cases, a large number of features are employed to compensate for the fact that relevant features are unknown a priori. However, without employing some kind of feature selection strategy, many of them would be either redundant or even irrelevant which could affect classification accuracy and efficiency seriously. In general, it is highly desirable to use only those features that have great separability power while ignoring or paying less attention to the rest. For instance, to allow a vehicle detector to generalize nicely, it would be nice to exclude features encoding fine details which might be present in some vehicles only. Finding out what feature to use for classification/recognition is referred to as feature selection.

R. Wang et al. [23] proposed a vehicle detection system based on local features that are located within three significant subregions of vehicles. By combining PCA and Independent Component Analysis (ICA), each examined subregion is projected onto its associated eigenspace and independent basis space to generate a PCA weight vector and an ICA coefficient vector respectively. A likelihood evaluation process is then performed based on the estimated joint probability of the projection weight vectors and the coefficient vectors of the subregions with position information. The use of subregion position information minimizes the risk of false acceptances, whereas the use of PCA to model the low-frequency components of the eigenspace and ICA to model the high-frequency components of the residual space improves the tolerance of the detection process toward variations in the illumination conditions and vehicle pose.

P. Viola and M. J. Jones [24] proposed an original feature selection scheme for object detection. The approach consists in a cascade of boosted classifiers with increasing complexity: each layer in the cascade reduces the search zone while rejecting regions that do not contain interested object. This method uses Haar-like features, also called rectangular filters (experimented by Papageorgiou et al. [25]), and AdaBoost learning [26], the latter

permits to select a limited number of features in each layer. The use of integral images to calculate Haar-like features and the cascade approach results in a real-time face-detection application. This approach has inspired a lot of recent works in vehicle detection.

There are some improvements about the used features, P. Negri et al. [27] combined the rectangular filters (Haar-like features) and the histograms of oriented gradient (HoG) with AdaBoost algorithm, The fusion combines the advantages of the other two detectors: generative classifiers composed of Haar-like features eliminate "easily" negative examples in the early layers of the cascade, while in the later layers, the discriminative classifiers composed of HoG features generate a fine decision boundary removing the negative examples near the vehicle model, so that the fusion achieved better performances than either feature.

In this study, hypothesis generation methods based on symmetry and statistic of edge intensity are used as candidate region filters. Following cascaded Haar feature classifier refined by AdaBoost algorithm is used as first hypothesis verification method, and plays the role of vehicle detector. Then, a probabilistic variant of decision-based neural network [19, 20] is used as second hypothesis verification method and a target verifier, which reduces a great deal of false positive. It can be observed that the nature and decision boundary of former differ from those of latter: the former is local-feature oriented, whereas the latter is global-feature oriented. We think that these two kinds of classifiers can be complementary when dealing with the problem.

Chapter 3
Vehicle Detection System

In this chapter, the proposed system structure is defined. The system structure is composed of five sub-systems: pre-processing, candidate region filtering, vehicle detection, target validation and post-processing. Section 3.1 demonstrates the diagram of global system, which shows five sub-systems and their key modules. Section 3.2 illustrates the candidate region filters and quantitative performance of the sub-system. Section 3.3 describes the origin of using AdaBoost classifiers and details AdaBoost classifier. Section 3.4 introduces the concept of combining two classifiers, and details Probabilistic Decision-Based Neural Network (PDBNN) classifier. Section 3.5 illustrates the two algorithms used for adapting the size of candidate windows and removing redundant detected windows respectively.

3.1 System Overview

At first, pre-processing module directly uses the raw data of surveillance video as inputs. This sub-system transforms the raw color image into a gray level image, for acceleration, the gray level image is downsized by a simple interpolation algorithm. Sobel edge operation is applied on raw-size gray level image to preserve detailed edge intensity of the image, then the edge intensity image is resized by the same interpolation algorithm to generate downsized edge intensity image as the input of next sub-system: candidate region (CR) filtering. After filtering out unqualified regions by CR filters, the residual positions of downsized gray level image that might contain interested objects are examined and located by AdaBoost classifier. Next, these positions are verified by PDBNN classifier based on probability estimation to confirm the existence of interested objects. Finally, the post-processing deals with redundant

detected overlapping windows belong to the same object and results in exact locations of the objects. The diagram of global system is shown in Figure 3-1.

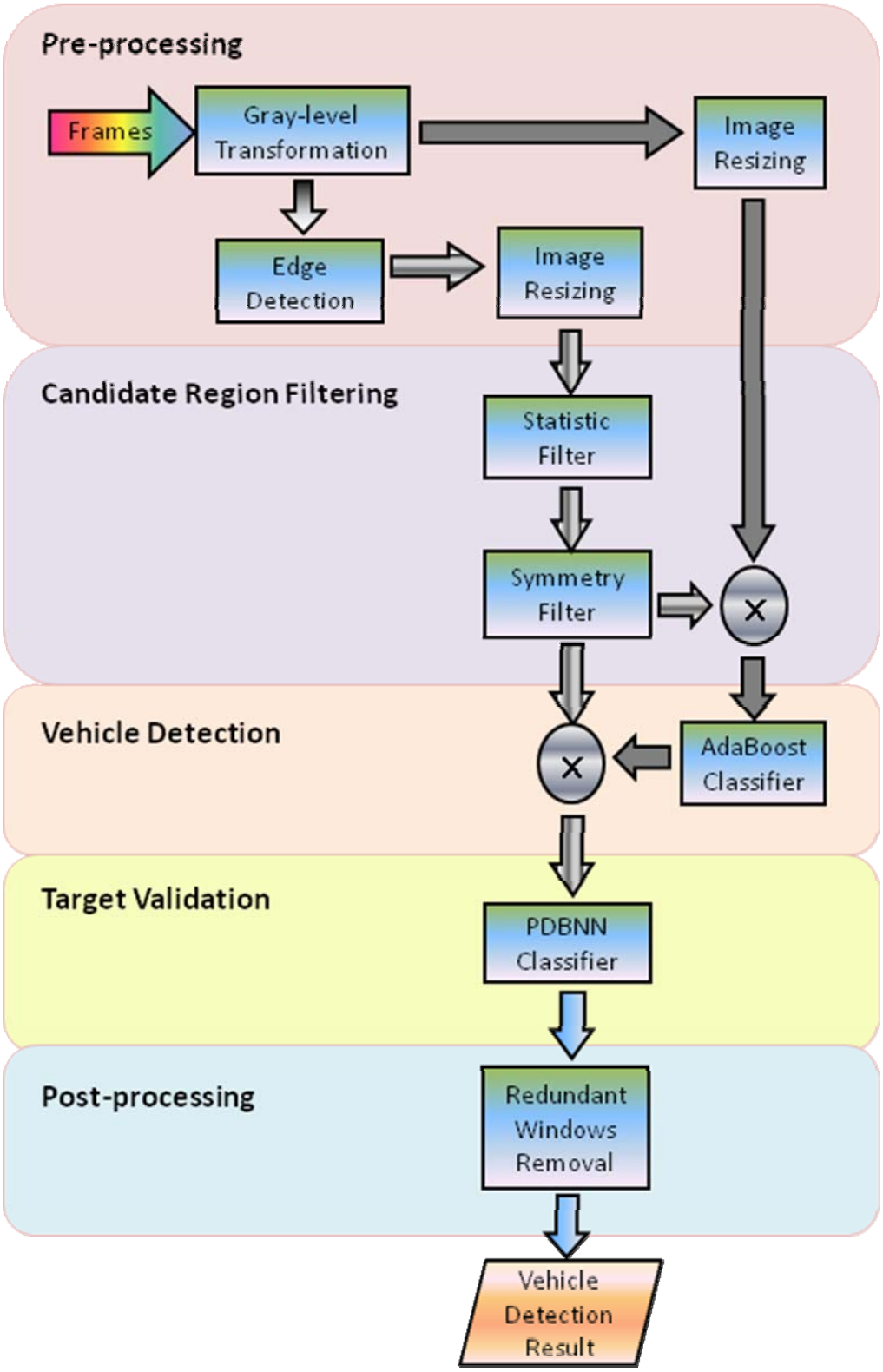


Fig. 3-1 System diagram

3.2 Candidate Region (CR) Filtering

In this research, Candidate Region (CR) filters are the alias of hypothesis generation (HG) methods. Based on simple, fast and low-level algorithms, the purpose of CR filtering is to eliminate regions which are affirmed to be no object present, so the potential locations of vehicles are hypothesized, as a result, the follow-up hypothesis verification (HV) steps, (i.e. vehicle detection and target validation in this research) can focus on those regions which are likely to contain interested objects. This step reduces a great deal of computation, and can lower the false positive rate simultaneously. As mentioned in section 2.1, the principle of candidate region filtering is to filter out unqualified searching windows as many as possible while keeping overall detection rate as high as possible, so the rule of determining thresholds tends to be conservative.

3.2.1 Statistic Filter

By a simple statistical method, the rational edge intensity distributions of interested objects can be estimated, therefore, candidate regions whose texture is too sparse or too dense can be filtered out rapidly. The distribution of the edge intensity statistic and the threshold for the scope are shown in Figure 3-2.

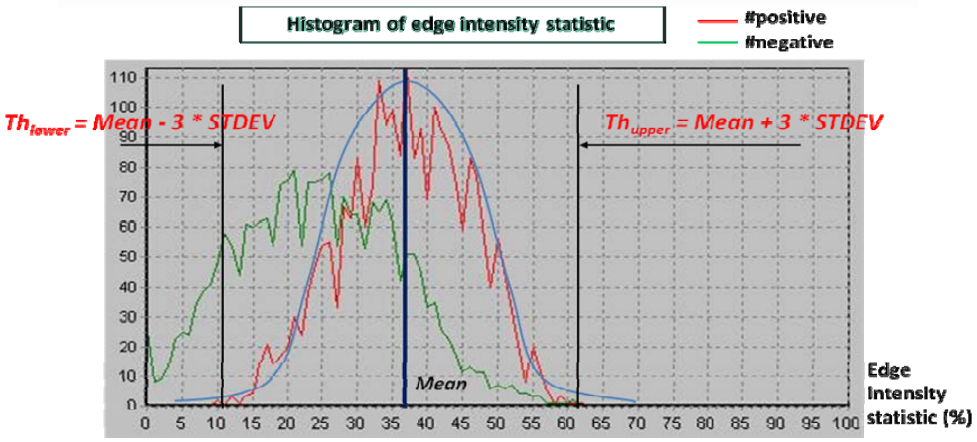


Fig. 3-2 The scope of edge intensity statistic

As shown in Figure 3-2, the edge intensity distribution of positive samples (vehicle) is approximate a Gaussian distribution. So the lower and upper thresholds can be obtained by mean and standard deviation of this Gaussian distribution with Equation 3-1 and 3-2.

$$Threshold_{lower} = Mean - 3 * STDEV \quad (3\text{-}1)$$

$$Threshold_{upper} = Mean + 3 * STDEV \quad (3\text{-}2)$$

where Mean and STDEV are mean and standard deviation of the Gaussian distribution respectively. It has the nice property that a known percentage of all possible values of data lie within a certain number of standard deviations from the mean. Theoretically, 99.7% of the rates will fall within plus or minus 3 standard deviations from the mean.

3.2.2 Symmetry Filter

Lower parts of vehicles are visually symmetric and textured, see Figure 3-3, so the symmetry degree of candidate region can be used to hypothesize whether the candidate region should contain a vehicle or not. By accumulating the histogram of intensity (HoI) vertically and horizontally, the rough degree of symmetry can be obtained with the formula in Equation 3-3.

$$Degree(A = B) = \frac{A \cap B}{A \cup B} \quad (3\text{-}3)$$

where A is the measure of left half HoI, and B is the measure of right half HoI. By a statistical method, the threshold of minimum symmetry degree which a vehicle should hold can be obtained. As shown in Figure 3-4, any region of interest with symmetry degree below the threshold is kicked out. Figure 3-5 sketches the eliminated searching scope (marked with red areas) of CR filtering in scenes with (a) light traffic and .(b) heavy traffic.respectively.

The quantative performance of these two candidate region filters, estimated in cluttered

and uncluttered conditions, is shown in Table 3-1. Though the utilized CR filters tend to be conservative, the CR filters can narrow down 60% of search area in best-case scenarios.

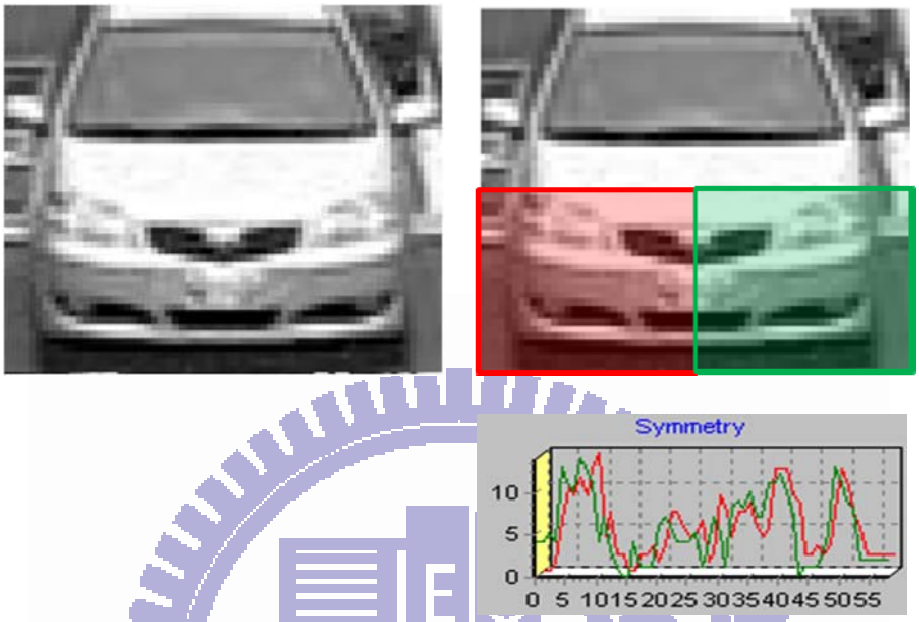


Fig. 3-3 Symmetry of vehicles

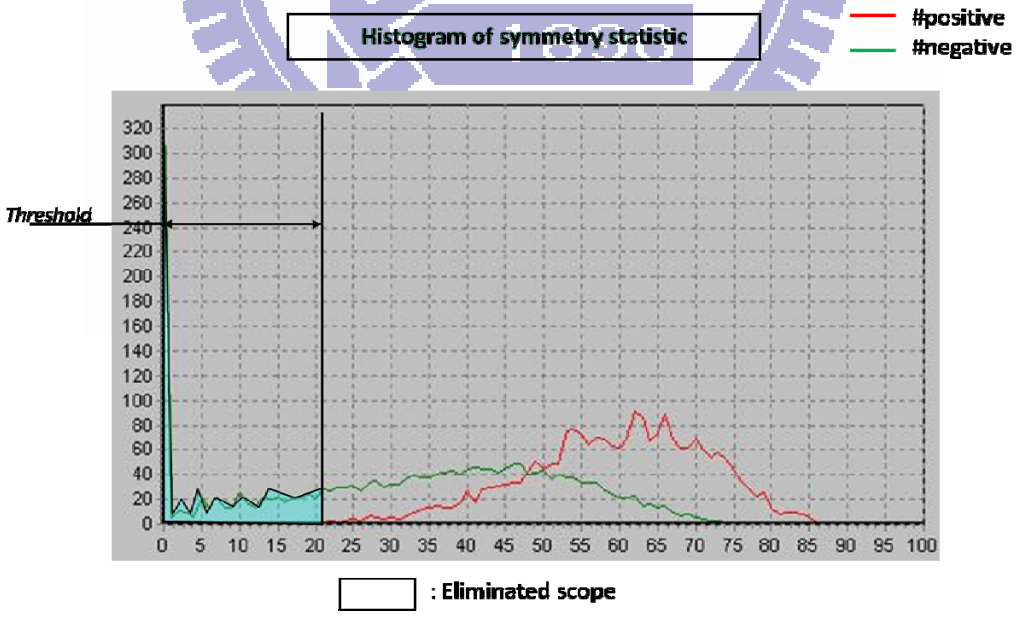
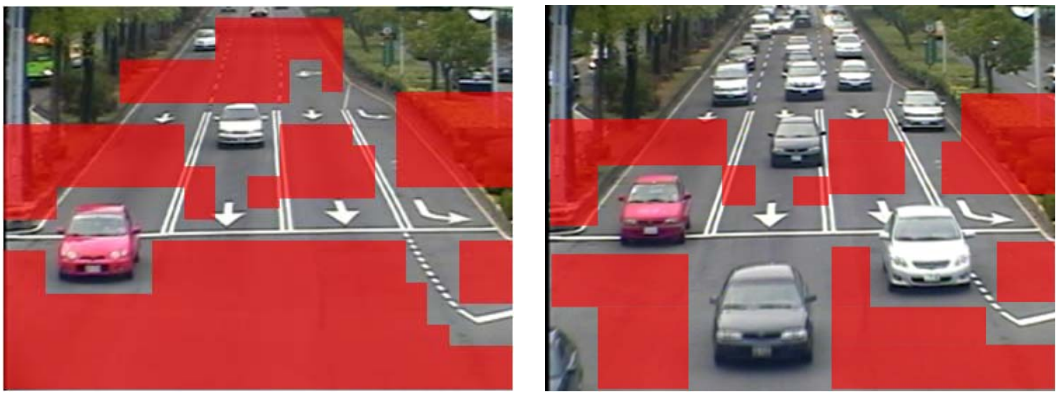


Fig. 3-4 The statistic scope of symmetry



(a) (b)

Fig. 3-5 The eliminated search scope of CR filtering

Tab. 3-1 The performance of CR filters

Type of Candidate Region Filter	Rejection Rate of Uncluttered Negative Samples	Rejection Rate of Cluttered Negative Samples
Statistic filter	61.15%	18.55%
Symmetry filter	65.55%	21.84%

3.3 Vehicle Detection using AdaBoost

This sub-system is the main process of entire system, as mentioned in 2.2, most research efforts have focused on feature extraction and classification by learning and statistical models, the most important issue in object detection literature is selecting a suitable set of features which can be the representation and soundly express the implicit invariant of interested objects. An intuitive method of feature extraction is to focus on the common components of interested objects. Any perceptual characteristic such as color, edge, texture, entropy, symmetry, ... etc. can be used as a feature solely or cooperatively. Though this method is easy to imagine and implement, it has its own limits. One of the limitations is that the manually chosen features are inferior in both value and quantity. Besides, the manually chosen features tend to apparent, moreover, the physical nature of human's perception is usually not steady enough to stand for the interested objects. So, it can be explained that the efforts of feature

extraction have turned to the statistical and machine learning areas.

The famous AdaBoost algorithm proposed by Y. Freund et al. [26] is one of machine learning techniques and has been progressed for pattern recognition rapidly and widely. AdaBoost combines weak classifiers into a weighted voting machine, and it shows high performance in various fields. P. Viola et al [29] built an efficient moving person detector, using AdaBoost to train a chain of progressively more complex region rejection rules based on Haar-like wavelets and space-time differences. In P. Viola, M. J. Jones [24], AdaBoost is applied to face detection and yields best performance comparable to the previous systems. A. Khammari et al. [15] and P. Negri [27] proposed the applications of AdaBoost in vehicle detection, both researches demonstrate the results through few proprietary examples, the former does not report any quantitative performance, the latter provides quantitative performances of proprietary databases.

Originally, the proposed system imitated the idea of face detection. Windshield, lamps and back mirrors of a vehicle can be regarded as forehead, eyes and ears of a human face. Due to this perceptual imagination, the proposed system intended to model vehicles out of a mixture of Gaussian distributions trained by Probabilistic Decision-Based Neural Network (PDBNN), which will be introduced in 3.5.1. The model preserves the characteristics of a vehicle, including shape of entire body, windshield and hood. It should be mentioned that the significance of vertical edges is overwhelmed by that of horizontal edges, as a result, the built model is very sensitive to regions with horizontal texture. This is induced by the property of neutralization, It is observed that the features trained by PDBNN tend to ignore the local structure, the detail of geometrical patterns. The property of neutralized features can be defined as global-orientation. Besides, compare with the geometrical structure of a face (PDBNN was applied to face detection and recognition in S.H. Lin et al. [20]), the structure of a frontal-view vehicle is more complicated. The main characteristics of vehicles vary on color,

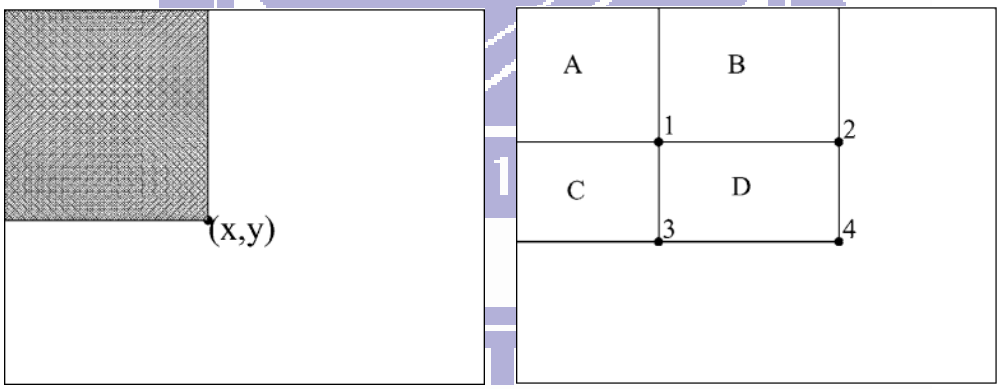
shape of whole body, lamps or windshield, such that the features are neutralized in built Gaussian models, and the influence on vehicle models is more notable than that on face models. To give considerations to the local structure of vehicles, any classifier with local-oriented nature is preferred, AdaBoost classifier (will be introduced in 3.4.1~3.4.4) has exhibited its robustness in object detection and definitely aiming at local structure, so it is chosen to make up the flaw of PDBNN classifier in this study.

3.3.1 Haar Features (Rectangle Features)

Haar basis functions (Haar features) used by Papageorgiou et al. [28] provide information about the grey-level distribution of two or more adjacent regions in an image. Figure 3-6 shows a set of simple Haar features. These features consist of two to four rectangles. To compute the output of a Haar basis function on a certain region of image, the sum of all pixels intensity in the black region is subtracted from the sum of all pixels intensity in the white one and normalized by a coefficient in case of a filter whose square measures of white and black regions are different. To reduce computation time for the filters, P. Viola et al. [24] introduced the integral image which is an intermediate representation for an input image. The concept of integral image is illustrated in Figure 3-7(a), the value of the integral image at point (x, y) is the sum of all the pixels above and to the left. In Figure 3-7(b), the sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is A + B, at location 3 is A + C, and at location 4 is A + B + C + D. The sum within D can be computed as 4 + 1 − (2 + 3). Utilizing integral images, sum of a rectangular region can be calculated by using only four references in the integral image. As a result, the difference of two adjacent rectangular regions can be computed by using only six references in the integral image, eight in the case of the three-rectangle filters, and nine for four-rectangle filters.

Every feature j is defined as fj(r_j, w_j, h_j, x_j, y_j), where r_j is the type of Haar feature, w_j and h_j are width and height of the Haar feature, and (x_j, y_j) is its position in the window. $value_{subtracted} = f(r, w, h, x, y)$, is the weighted sum of the pixels in white rectangles subtracted from those of dark rectangles.

Fig. 3-6 The common used Haar features



(a) (b)

Fig. 3-7 Integral image

3.3.2 Weak Classifier

A weak classifier h consists of a rectangle feature f (defined in 3.3.1), a threshold (θ) and a polarity (p) indicating the direction of the inequality in Equation 3-4.

$$h = \begin{cases} 1 & if \quad p_j f_j < p_j \theta_j \\ 0 & otherwise \end{cases} \tag{3-4}$$

where f_j is the absolute value ($value_{subtracted}$) of the feature j, θ_j is the threshold, and p_j is the parity. For each feature j, AdaBoost algorithm (will be introduced in 3.3.3) is used to determine an optimal threshold θ_j for which the classification error on training database (with positive and negative samples) is minimized. The brute force threshold selection for weak classifiers is illustrated in Figure 3-8. By selecting the threshold, the blue points (positive samples) and red points (negative samples) can be separated with a lowest classification error.

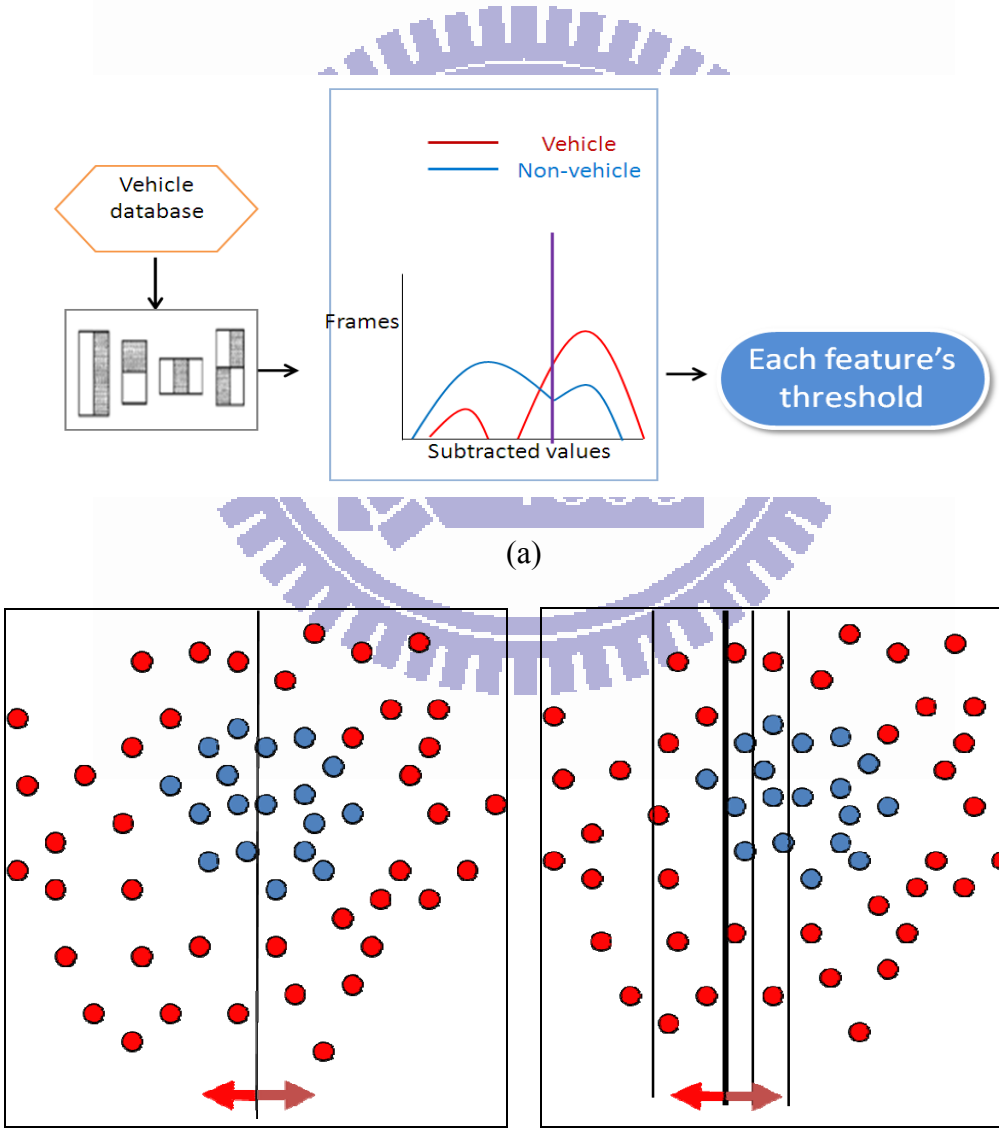


Fig. 3-8 Select thresholds for weak classifiers

3.3.3 AdaBoost Algorithm

The size of feature set originated from the permutation of Haar feature types, scales and positions, is many times greater than the number of pixels in the input image. Even through each rectangle feature can be computed very efficiently, computing the complete set is prohibitively expensive. There are some features from this set do not contain useful information. In literature, different methods have been used for the selection of useful features, e.g. Principal Component Analysis (PCA) used in [17, 22, 23], Independent Component Analysis (ICA) used in [23], and so forth. Among these methods, AdaBoost algorithm has shown its capability to improve the performance of various classification and detection systems. It finds precise hypotheses by combining several weak classifiers which, in general, have moderate precision. This iterative algorithm finds some weak but discriminative classifiers and combines them in a strong classifier which is defined in Equation 3-5.

$$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & otherwise \end{cases} \quad (3\text{-}5)$$

where h and C are the weak and strong classifiers, respectively, and α is a weight coefficient for each h.

Consider a 2D feature space with positive and negative training samples. Each weak classifier splits the training examples with at least 50% accuracy. As shown in Figure 3-9(a) , samples misclassified by a previous weak learner are given more emphasis at future rounds. The process of selecting a weak classifier is shown in Figure 3-9(b)(c)(d)(e)(f)(g), as shown in Figure 3-9(b), the misclassified blue points (positive samples) in left side of the black line and red points (negative samples) in right side of the black line are emphasized in the next round shown in Figure 3-9(c), similarly, the misclassified blue points in right side of the black line and red points in left side of the black line are emphasized in the next round, as shown in Figure 3-9(d) and Figure 3-9(e). Finally, the strong classifier is form with a linear

combination of weak classifiers shown in Figure 3-9(g), and the boosting algorithm for selecting a set of weak classifiers to compose a strong classifier is shown in Table 3-2.

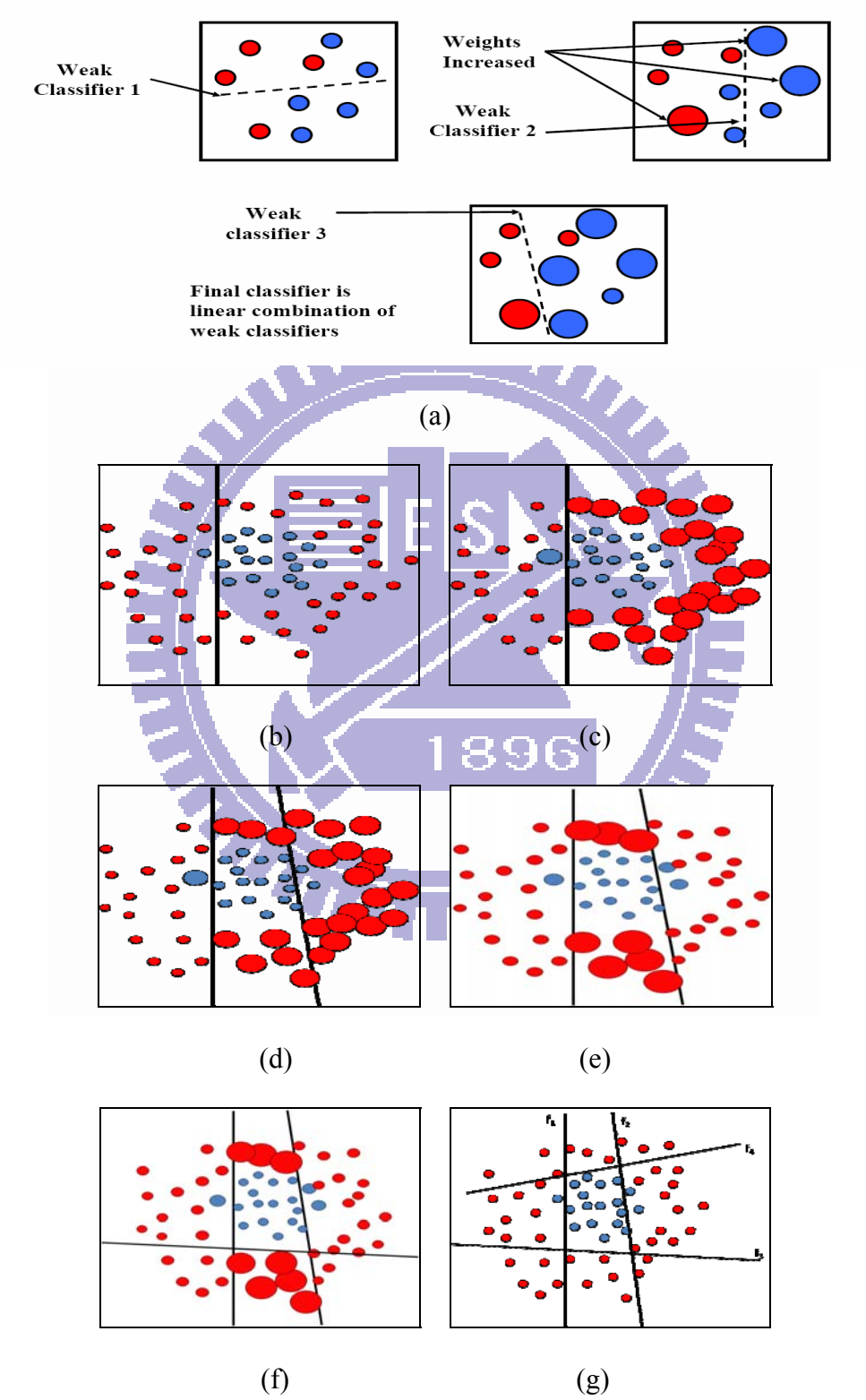


Fig. 3-9 The process of selecting a weak classifier

Tab. 3-2 The boosting algorithm for learning a query online.

T hypotheses are constructed each using a single feature. The final hypothesis is a weighted linear combination of the T hypotheses where the weights are inversely proportional to the training errors.

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  - Normalize the weights,
    $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
  - Select the best weak classifier with respect to the weighted error
    $$\varepsilon_t = \min_{f,p,\theta} \sum_i w_i \left| h(x_i, f, p, \theta) - y_i \right|$$
  - Define $h_t(x) = h(x, f_t, p_t, \theta_t)$ where f_t ,p_t ,and θ_t are the minimizers of ε_t.
  - Update the weights:
    $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$
    where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise,
    and $\beta_t = \frac{\varepsilon_t}{1 - \varepsilon_t}$
- The final strong classifier is:
  $$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & otherwise \end{cases}$$
  where $\alpha_t = \log \frac{1}{\beta_t}$

3.3.4 Strong Classifier

A strong classifier is a stage composed of at least one weak classifier. The cascade of these stages is also constructed by AdaBoost algorithm, described in Table 3-3. In each stage, if an object extracted by the searching window is classified as vehicle, it is allowed to enter the next stage; otherwise, the object is rejected immediately. In brief, an object labeled as vehicle should pass through a series of stages; an object is rejected by particular stage even if

it enters the last stage. Figure 3-10 demonstrates the schema of a detection cascade.

Tab. 3-3 The training algorithm for building a cascaded detector.

- User selects values for f, the maximum acceptable false positive rate per layer and d, the minimum acceptable detection rate per layer.
- User selects target overall false positive rate, F_{target}.
- P = set of positive examples
- N = set of negative examples
- $F_0 = 1.0$; $D_0 = 1.0$
- i = 0
- while $F_i > F_{target}$
  - $i \leftarrow i + 1$
  - $n_i = 0$; $F_i = F_{i-1}$
  - while $F_i > f \times F_{i-1}$
    - $n_i \leftarrow n_i + 1$
    - Use P and N to train a classifier with n_i features using AdaBoost
    - Evaluate current cascaded classifier on validation set to determine F_i and D_i.
    - Decrease threshold for the ith classifier until the current cascaded classifier has a detection rate of at least $d \times D_{i-1}$ (this also affects F_i)
  - $N \leftarrow 0$
  - If $F_i > F_{target}$ then evaluate the current cascaded detector on the set of non-face images and put any false detections into the set N

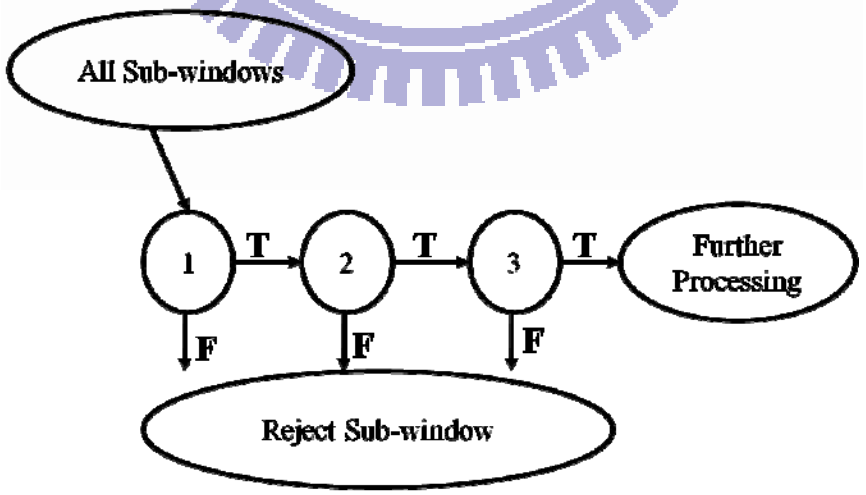
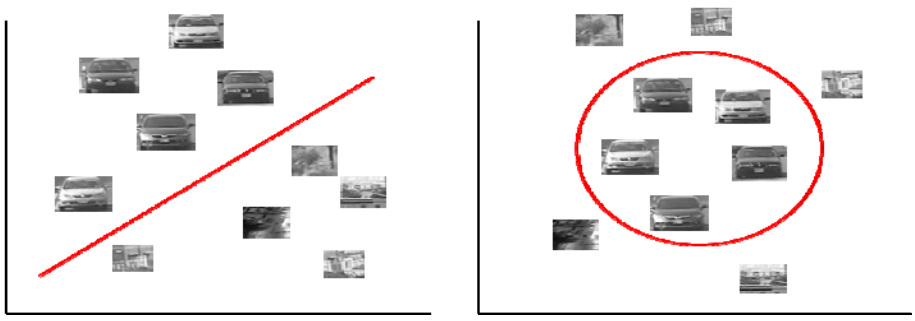


Fig. 3-10 A detection cascade.

3.4 Target Validation using PDBNN

Neural networks learn the classification rules from the given collection of representative examples. This ability to automatically learn from examples makes neural network approaches attractive and exciting. Moreover, it is well known that neural networks are very robust and adaptive. By estimating the maximum likelihood on the features of images (edge intensity is used as the main feature of vehicles in this study), the likelihood density of the object class can be obtained. PDBNN is applied to face detection and recognition in S.H. Lin [20], which is a probabilistic variant of its predecessor, decision-based neural network (DBNN) [19]. DBNN is an efficient classification neural network. It has a modular network structure. One subnet is designated to represent one object class.

As mentioned in 3.3, AdaBoost classifiers are good at recognizing local features while PDBNN classifiers are good at global features. Previous research has revealed that the decision boundary of an AdaBoost classifier is a hyper plane, which is illustrated in P. Negri et al. [27], testing samples are categorized by their position in the feature space with respect to this hyper plane, as shown in Fig 3-11(a). Moreover, [20] proved that the decision boundary of a PDBNN classifier exhibits the similar appearance as shown in Figure 3-11(b). Testing samples are compared to a PDBNN classifier and categorized by their similarity, the idea of fusing two classifiers with different characteristics is consequently inspired.



(a) (b)

Fig. 3-11 Decision boundaries of AdaBoost and PDBNN classifiers

3.4.1 Conception of PDBNN

PDBNN assumes that the class likelihood function for the interested object class (i.e. vehicle class) can be represented by a mixture of Gaussian distributions. The training scheme of these Gaussian distributions includes two phases: 1) locally unsupervised (LU) phase and 2) globally supervised (GS) phase.

The values of the parameters so-called weight vector in the network are initialized in LU learning phase. An unsupervised clustering algorithm: k-means, is applied to segment the training samples of object class into several clusters and determine initial positions of the cluster centroids, then the expectation-maximization (EM) algorithm [30] is used to learn the cluster parameters of object class, obtain maximum likelihood estimation (MLE) of the data distribution, and determine the parameters of Gaussian distributions, demonstrated in Figure 3-12(a). In the global supervised (GS) training phase, two sets of patterns (i.e. both positive and negative samples) are involved, as shown in Figure 3-12(b), teacher information is utilized to fine-tune the decision boundary which is determined by a threshold. When a training pattern is misclassified, the reinforced or antireinforced learning technique [19] is applied. If the misclassified training pattern is from the positive training (i.e. vehicle) set (the training data set in the LU phase), reinforced learning will be applied to adjust the subnet parameters and the boundary by updating the weight vector in the direction of the gradient of the discriminant function. If the training pattern belongs to the so-called negative training (i.e. non-vehicle) set, then the anti-reinforced learning rule will be executed to adjust only the boundary in the opposite direction of the gradient of the discriminant function, illustrated in Figure 3-12(c)(d).

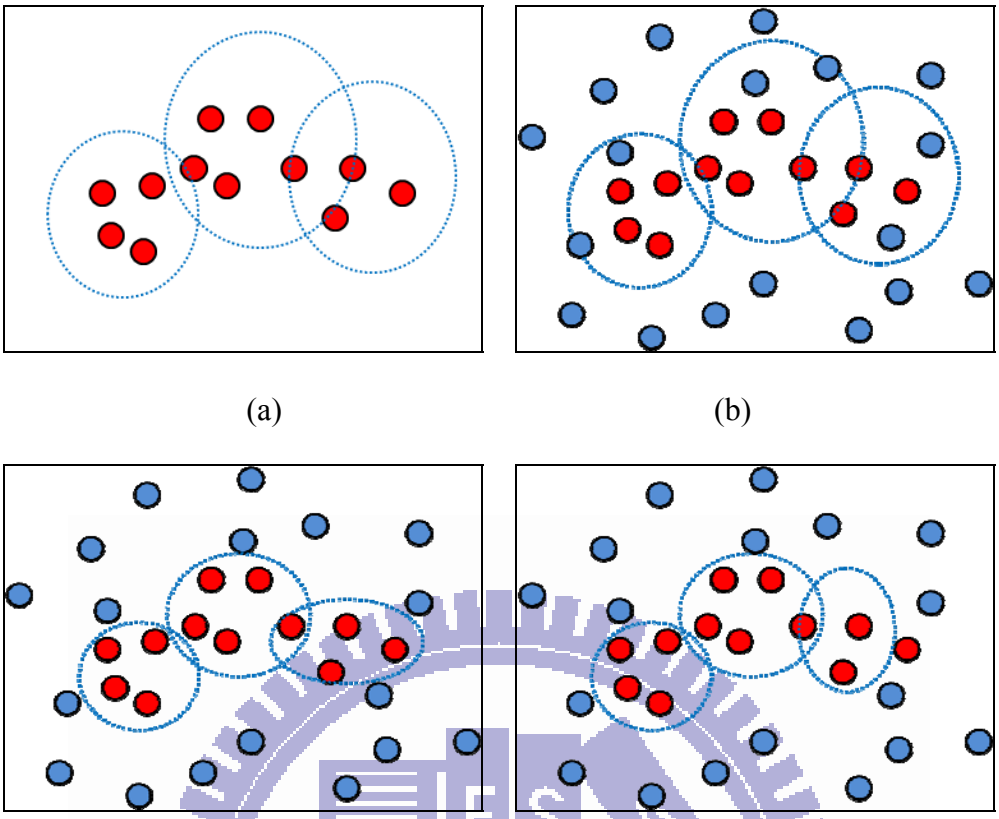


Fig. 3-12 The diagram of PDBNN learning

3.4.2 Discriminant Functions of PDBNN

The subnet discriminant functions of PDBNN are designed to model the log-likelihood functions. The reinforced and antireinforced learning is applied to all the clusters of the global winner and the supposed (i.e., the correct) winner, with a weighting distribution proportional to the degree of possible involvement (measured by the likelihood) by each cluster.

Given a set of identical independent distributed patterns $X^+ = \{x(t), t = 1, 2, \ldots, N\}$. Assume the class likelihood function $p(x(t)|\omega)$ for class ω (i.e. vehicle class) is a mixture of Gaussian distributions. Define $p(x(t)|\omega,\Theta_r)$ to be one of the Gaussian distributions $p(x(t)|\omega,\Theta_r) \equiv N(\mu_r, \Sigma_r)$

$$p(x(t) \mid \omega) = \sum_r P(\Theta_r \mid \omega) p(x(t) \mid \omega, \Theta_r) \qquad (3\text{-}6)$$

where Θ_r represents the rth cluster in the subnet. $P(\Theta_r|\omega)$ denotes the prior probability of cluster r. By definition $\Sigma_r P(\Theta_r|\omega) = 1$.

The discriminant function of one-subnet PDBNN models the log-likelihood function

$$\phi(x(t), w) = \log p(x(t) \mid \omega) = \log[\sum_r P(\Theta_r \mid \omega) p(x(t) \mid \omega, \Theta_r)] \quad (3\text{-}7)$$

where

$$w \equiv \{\mu_r, \Sigma_r, P(\Theta_r \mid \omega), T\} . \quad (3\text{-}8)$$

T is the threshold of the subnet. It will be used in the GS learning phase. The overall diagram of such discriminant function is depicted in Figure 3-13.

In most general formulation, the basis function of a cluster should be able to approximate the Gaussian distribution with full-rank covariance matrix. A hyper-basis function (Hyper BF) is meant for this [31]. However, for those applications which deal with high-dimensional data but finite number of training patterns, the training performance and storage space discourage such matrix modeling. A natural simplifying assumption is to assume uncorrelated features of unequal importance. That is, suppose that $p(x|\omega,\Theta_r)$ is a D-dimensional Gaussian distribution with uncorrelated features

$$p(x(t) \mid \omega, \theta_r) = \frac{1}{(2\pi)^{D/2} \prod_d^D \sigma_{rd}} \cdot \exp(-\frac{1}{2} \sum_{d=1}^{D} \frac{(x_d(t) - w_{rd})^2}{\sigma_{rd}^2}) \sim N(\mu_r, \Sigma_r) \quad (3\text{-}9)$$

where $x(t) = [x_1(t), x_2(t), ..., x_D(t)]^T$ is the input pattern, $\mu_r = [w_{r1}, w_{r2}, ..., w_{rD}]^T$ is the mean vector, and diagonal matrix $\Sigma_r = diag[\sigma_{r1}^2, \sigma_{r2}^2, ..., \sigma_{rD}^2]$ is the covariance matrix.

To approximate the density function in Equation 3-9, the elliptic basis functions (EBF's) is to serve as the basis function for each cluster

$$\psi(x(t), \omega, \Theta_r) = -\frac{1}{2} \sum_{d=1}^{D} \beta_{rd} (x_d(t) - w_{rd})^2 + \theta_r \quad (3\text{-}10)$$

where

$$\theta_r = -\frac{D}{2}\ln 2\pi - \sum_{d=1}^{D}\ln \sigma_{rd} \qquad (3\text{-}11)$$

After passing an exponential activation function, $\exp\{\phi(x(t), \omega, \Theta_r)\}$ can be viewed the same Gaussian distribution as described in Equation 3-9, where $1/\beta_{rd} = \sigma_{rd}^2$

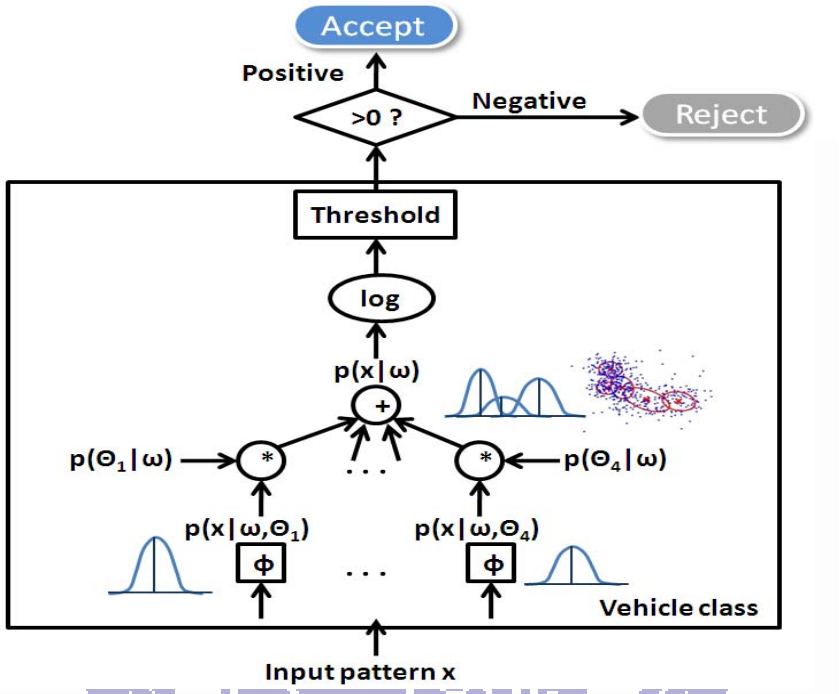


Fig. 3-13 The diagram of a PDBNN classifier

3.4.3 Learning Rules for PDBNN

As mentioned in 3.4.1, the training scheme for PDBNN contains LU and GS phases. The locally unsupervised (LU) phase for the PDBNN can adopt one of the unsupervised learning schemes (e.g., VQ, k-means, EM, ...etc.). The network enters the globally supervised (GS) phase after the LU training is converged. For the GS learning, the decision-based learning rule is adopted. Both training phases need several epochs to converge. The training phases are introduced as follows:

3.4.3.1. Unsupervised Training for LU Learning

The values of the parameters in the network are initialized in the LU learning phase. The well-known unsupervised clustering algorithm, k-means, and the expectation-maximization (EM) algorithm are used to learn the cluster parameters of object class, obtain maximum likelihood estimation (MLE) of the data distribution, and initialize the parameters of clusters modeled by several Gaussian distributions.

The k-means method adjusts the center of a cluster based on the distance $||x - \mu_r||^2$ of its neighboring pattern x. Basically k-means algorithm assumes that the covariance matrix of a cluster i is $\sigma_i^2 I$. An iteration of the k-means algorithm in the LU phase contains two steps: first, all the input patterns are examined to find out their closest cluster centers. Then, each cluster center are moved to the mean of its neighboring patterns.

The EM algorithm is a special kind of quasi-Newton algorithm with a searching direction having a positive projection on the gradient of the log likelihood. In each EM iteration, there are two steps: Estimation (E) step and Maximization (M) step. The M step maximizes a likelihood function which is further refined in each iteration by the E step. The EM algorithm in the LU phase is as follows [32]. Use the data set $X^+ = \{x(t); x(t) \in \omega, t = 1,2,..., N\}$. The goal of the EM learning is to maximize the log likelihood of data set X^+

$$l(w; X^+) = \sum_{t=1}^{N} \log p(x(t) \mid \omega) = \sum_{t=1}^{N} \log[\sum_{r} P(\Theta_r \mid \omega) p(x(t) \mid \Theta_r, \omega)] \qquad (3\text{-}12)$$

The EM algorithm begins with the observation that the optimization of the likelihood function $l(w; X^+)$ would be simplified if only a set of additional variables, called “missing” or “hidden” variables, were known. In order to include the missing information into the optimization formula, here the indicator variables $z_r(t)$ are defined to specify which cluster generate the pattern

$$z_r(t) = \begin{cases} 1, & \textit{if pattern } x(t) \textit{ is from cluster } r \\ 0, & \textit{otherwise.} \end{cases} \tag{3-13}$$

The observable data X^+ is defined as "the incomplete data" and posit a "complete data" set Y that includes the missing variables Z. The probability model p(y|x, ω) is used to link the missing variables to the actual data. The logarithm of this density defines the "complete-data likelihood"

$$\begin{aligned} l_c(w; X^+) &= \sum_{t=1}^{N} \sum_r z_r(t) \log[P(\Theta_r \mid \omega) p(x(t) \mid \Theta_r, \omega)] \\ &= \sum_{t=1}^{N} \sum_r z_r(t) [\log P(\Theta_r \mid \omega) + \log(p(x(t) \mid \Theta_r, \omega)] \end{aligned} \tag{3-14}$$

Since $z_r(t)$ indicates which cluster the input pattern belongs to, the log operator and the second summation can be exchanged. The optimization has been decomposed into several subproblems. Notice that since the indicator variable $z_r(t)$ is actually unknown, in the E step of the jth iteration of the EM algorithm the expectation of the complete-data likelihood is taken.

$$Q(w, w^{(j)}) = E[l_c(w; X^+) \mid X^+, \omega, w^{(j)}] \tag{3-15}$$

For simplicity sake, the notation of the expectation is changed to the following:

$$\begin{aligned} Q(w, w^{(j)}) &= E^{(j)}[\sum_{t=1}^{N} \sum_r z_r(t) [\log P(\Theta_r \mid \omega) + \log(p(x(t) \mid \Theta_r, \omega)]] \\ &= \sum_{t=1}^{N} \sum_r h_r^{(j)}(t) [\log P(\Theta_r \mid \omega) + \log(p(x(t) \mid \Theta_r, \omega)] \end{aligned} \tag{3-16}$$

where $p^{(j)}(x(t) \mid \Theta_r, \omega) \equiv N(\mu_r^{(j)}, \sum_r^{(j)})$ and

$$\begin{aligned} h_r^{(j)}(t) &\equiv E^{(j)}[z_r(t)] \\ &= P^{(j)}(z_r(t) = 1 \mid x(t), \omega) \\ &= \frac{P^{(j)}(z_r(t) = 1 \mid \omega) p^{(j)}(x \mid \omega, z_r(t) = 1)}{P^{(j)}(x(t) \mid \omega)} \\ &= \frac{P^{(j)}(\Theta_r \mid \omega) p^{(j)}(x(t) \mid \omega, \Theta_r)}{\sum_k P^{(j)}(\Theta_k \mid \omega) p^{(j)}(x(t) \mid \omega, \Theta_r)} \end{aligned} \tag{3.17}$$

$w^{(j)}$ represents the parameter set in the PDBNN at epoch j. After the Expectation step, in the

M step of the EM algorithm, Q(w, $w^{(j)}$) is maximized with respect to weighting parameters.

There are two operations taken in an iteration of the EM algorithm. At iteration j. 1) E-step: computing the conditional posterior probabilities $h_r^{(j)}(t)$, $\forall r$

$$h_r^{(j)}(t) = \frac{P^{(j)}(\Theta_r \mid \omega) p^{(j)}(x(t) \mid \omega, \Theta_r)}{\sum_k P^{(j)}(\Theta_k \mid \omega) p^{(j)}(x(t) \mid \omega, \Theta_r)} \tag{3-18}$$

2) M-step: maximizing Q(w, $w^{(j)}$) with respect to w (Reference to Equation 3-8)

$$P^{(j+1)}(\Theta_r \mid \omega) = (1/N)\sum_{t=1}^{N} h_r^{(j)}(t)$$

$$\mu_r^{(j+1)} = (1/\sum_{t=1}^{N} h_r^{(j)}(t))\sum_{t=1}^{N} h_r^{(j)}(t)x(t)$$

$$\sum_r^{(j+1)} = (1/\sum_{t=1}^{N} h_r^{(j)}(t))\sum_{t=1}^{N} h_r^{(j)}(t)[x(t) - \mu_r^{(j)}] \cdot [x(t) - \mu_r^{(j)}]^T \tag{3-19}$$

Since the threshold will not affect the likelihood value, it is not updated here. When the EM iteration converges, it should ideally obtain maximum likelihood estimation (MLE) of the data distribution. EM has been reported to deliver excellent performance in several data clustering problems [32].

3.4.3.2. Supervised Training for GS Learning

In the global supervised (GS) training phase, teacher information is utilized to fine-tune decision boundaries. When a training pattern is misclassified, the reinforced or antireinforced learning technique [19] is applied.

Reinforced learning: $w^{(j+1)} = w^{(j)} + \eta \nabla \phi(x, w)$

Antireinforced learning: $w^{(j+1)} = w^{(j)} - \eta \nabla \phi(x, w)$ (3-20)

In this training phase, two sets of patterns are involved. If the misclassified training pattern is from the positive training (i.e., vehicle) set (the data set X^+ in the LU phase), reinforced learning will be applied. If the training pattern belongs to the so-called negative

training (i.e., non-vehicle) set, then only the anti-reinforced learning rule will be executed - since there is no correct class to be reinforced.

The gradient vectors in Equation 3-20 are computed as follows:

$$\left.\frac{\partial\phi(x(t),w)}{\partial w_{rd}}\right|_{w=w^{(j)}} = h_r^{(j)}(t)\cdot\beta_{rd}^{(j)}(x_d(t)-w_{rd}^{(j)}) \tag{3-21}$$

$$\left.\frac{\partial\phi(x(t),w)}{\partial \beta_{rd}}\right|_{w=w^{(j)}} = h_r^{(j)}(t)\cdot\frac{1}{2}\left(\frac{1}{\beta_{rd}^{(j)}}-(x_d(t)-w_{rd}^{(j)})^2\right) \tag{3-22}$$

where $h_r^{(j)}(t)$ is the conditional posterior probability as shown in Equation 3-18, $w_{rd}^{(j)}$ and $\beta_{rd}^{(j)}$ are defined in Equation 3-9 and Equation 3-10, respectively. As to the conditional prior probability $P(\Theta_r \mid \omega)$, since the EM algorithm can automatically satisfy the probabilistic constraints $\sum_r P(\Theta_r \mid \omega)=1$ and $P(\Theta_r \mid \omega)\geq 0$, it is applied to update the $P(\Theta_r \mid \omega)$ values in the GS phase so that the influence of different clusters are regulated: At the end of each epoch j

$$P^{(j+1)}(\Theta_r \mid \omega) = (1/N)\sum_{t=1}^{N} h_r^{(j)}(t) \tag{3-23}$$

3.4.3.3. Threshold Updating

The threshold value of PDBNN classifier can also be learned by the reinforced and antireinforced learning rules. Since the increment of the discriminant function $\phi(x(t),w)$ and the decrement of the threshold T have the same effect on the decision making process, the direction of the reinforced and anti-reinforced learning for the threshold is the opposite of the one for the discriminant function. For example, given an input x(t), if $x(t)\in\omega$ but

$\phi(x(t),w)$ < T, then T should reduce its value. On the other hand, if $x(t) \notin \omega$ but $\phi(x(t),w)$ > T, then T should increase. An adaptive learning rule to train the threshold T is illustrated in the following: Define $d(t) \equiv T - \phi(x(t),w)$ and a penalty function $l(d(t))$, $l(d(t))$ can be either a step function, a linear function, or a fuzzy-decision sigmoidal function. Once the network finishes the training, the threshold values can be trained as follows: Given a positive learning parameter η, at step j

$$T^{(j+1)} = \begin{cases} T^{(j)} - \eta l'(d(t)) & if \quad x(t) \in \omega \\ T^{(j)} + \eta l'(d(t)) & if \quad x(t) \notin \omega \end{cases} \tag{3-24}$$

Figure 3-12 illustrates the testing procedure of a PDBNN classifier. The input testing data is estimated the probability of its belonging to the mixture Gaussian model, and discriminated by the threshold which is also learned by PDBNN.

3.5 Adaptive Candidate Window and Redundant Detected Window Removal

The size of candidate regions is perspectively enlarged along the vertical direction, see Figure 3-14, so the respective size of candidate region can be represented as a linear equation. According to the initial window sizes of a distant and a near place, the perspective window size can be estimated by interpolation defined in Equation 3-25.

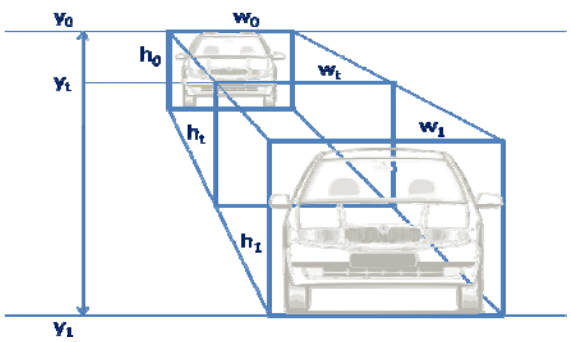


Fig. 3-14 Adaptive candidate window

$$\frac{y_t - y_0}{y_1 - y_0} = \frac{w_t - w_0}{w_1 - w_0} = \frac{h_t - h_0}{h_1 - h_0} \tag{3-25}$$

In general, each successful vehicle detection result usually leads to the creation of redundant detected candidate windows. To resolve this problem, the schemes presented in [33] combined all of the overlapping candidate windows together and still maintained a good detection performance. However, the non-overlapping window constraint may be too strict for closely spaced targets since they inevitably generate overlapping candidate windows. To solve this problem, the current study adopts a simple principle: among overlapping windows in a distance adapted to the quarter of window's diagonal, the candidate window with the highest likelihood probability ratio estimated by the PDBNN model is designated as the vehicle location, illustrated in Figure 3-15.

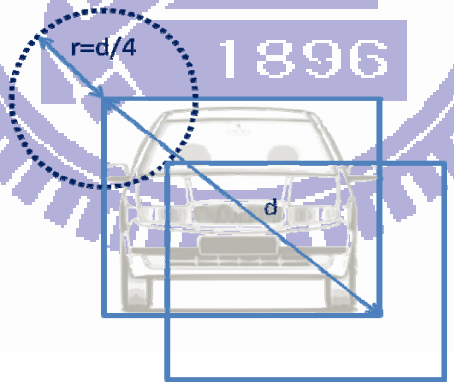


Fig. 3-15 Overlapping detected windows

Chapter 4
Experimental Results

The vehicle detection system is implemented on a PC system and the equipment of the PC is Intel(R) CPU @ 1.73G and 2GB RAM. The integrated development environment is Borland C++ Builder 6.0 on Windows XP OS. The inputs are video files (AVI uncompressed format) or image sequences (BMP or PPM format). These inputs were captured with a DV at traffic intersection or referred to testing samples which were used by other research.

Section 4.1 illustrates the process of training the two classifiers, including the principles of building proprietary vehicle database, and the criterion of collected samples. Section 4.2 illustrates the approach to test the built model. Section 4.3 demonstrates the experimental results of detecting vehicles in static images. The first testing database is proprietary car database and the second one is a public testing database - MIT CBCL car database 1999. The performance of AdaBoost classifier and the fusion (AdaBoost + PDBNN) is listed and compared. Section 4.3 demonstrates the experimental results of detecting vehicles in videos.

4.1 Model Training

The proposed vehicle detection system based on machine learning algorithms comprises training processes and testing processes. Sample collection and feature extraction are the keys of object detection and classification. To build the proposed system, first, the common components of vehicles must be determined (e.g., windshield, lamps, roof, wheels, ... etc.), the basis of collecting samples is retaining these common parts and keeping out unnecessary parts to reduce the interference of noise, as a result, the images are tightly cropped. Second, the manually collected samples are usually not uniform in size, point of view and illumination

condition, a normalization process is necessary, so that an identical criterion is set before the samples enter in training or testing step.

4.1.1 Training Dataset

The study considers the specific application of a vehicle counting system at the intersection, The proposed system presently focuses on frontal views of small vehicles such as sedans, sport-utility vehicles (SUVs), and minivans, which are the majority of traffic. The collected images of proprietary database were cut out from proprietary videos captured by a hand-held camera in daytime or early evening. The captured sources contain several scenes, most of these scenes are captured at the intersection, so stop-and-go traffic becomes a general case in this study. While collecting vehicle samples, the pose of captured cars is limited to frontal-views and the angle tolerance of pan-rotation range is about ±20°. Figure 4-1 demonstrates the diagram of extracting samples from video frames. Roof, windshield, lamps, back mirrors are the main characteristics chosen to recognize a vehicle in proposed system. So, all manually extracted vehicle samples should contain these components without occlusion. In crowded scenes, a vehicle is sometimes, even always occluded by other vehicles, so a following vehicle will not be a qualified sample till the preceding vehicle moves over. As a result, the sizes of extracted samples are inconsistent. The average width and height of collected 2344 vehicle samples are 61 x 53. For normalization, the samples can be resized in an equivalent ratio. Figure 4-2 demonstrates the collected vehicle and non-vehicle samples.

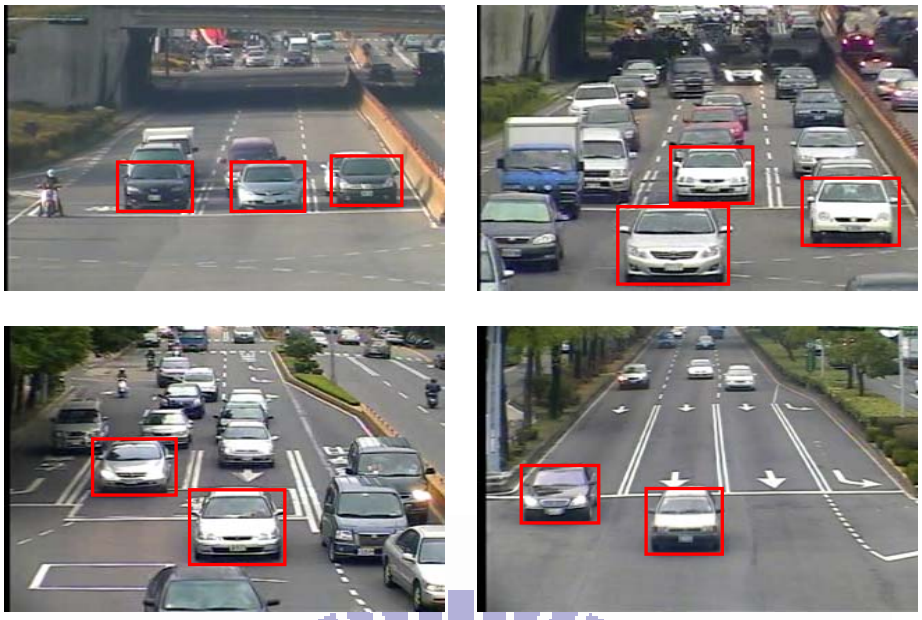


Fig. 4-1 Diagram of extracting vehicle samples

(a) Positive samples (vehicles)

(b) Negative samples (non-vehicles)

Fig. 4-2 Proprietary samples

4.1.2 AdaBoost Training

In this study, the raw input of AdaBoost training process is the gray level information of 2009 vehicle images and 11090 non-vehicle images from the proprietary database. The gray level information is normalized to a uniform size 21 x 18 without any further processing. The weak classifiers are the permutations of the type, positions and scale of 9 Haar features, as shown in Figure 3-6, unlike previous research, the scale of Haar features is increased in a brute force manner, i.e. the size of Haar features is from 2 x 2 to 21 x 18 progressively, as a result, the weak classifier learner yielded 289170 weak classifiers. The final classifier is a 3 layer cascade of classifiers which included a total of 940 weak classifiers. The first classifier in the cascade is constructed using 35 weak classifiers and rejects about 51% of non-vehicles while correctly detecting 100% of vehicles. The next classifier has 130 weak classifiers and rejects about 77% of non-vehicles while detecting 100% of faces. The last layer is a 765-feature classifier with a 100% detection rate and rejects about 87% of non-vehicles. The training process was terminated due to non-convergence, i.e. the boosting algorithm was unable to find anymore a weak classifier which is capable to obtain an error rate below 50%.

4.1.3 PDBNN Training

Due to the color or brightness of vehicles varies, edge intensity is the only feature used for PDBNN training, the raw input of PDBNN training process is the edge intensity information of 2009 vehicle images and 11090 non-vehicle images (same as those used in AdaBoost training) from the proprietary database. The edge intensity information is obtained from Sobel operator, and normalized to a uniform size 21 x 18 without any further processing. The training process finished while the false negative rate and false positive rate reached the expected thresholds, which are 98 %. The schema of training AdaBoost classifier and PDBNN classifier is shown in Figure 4-3.

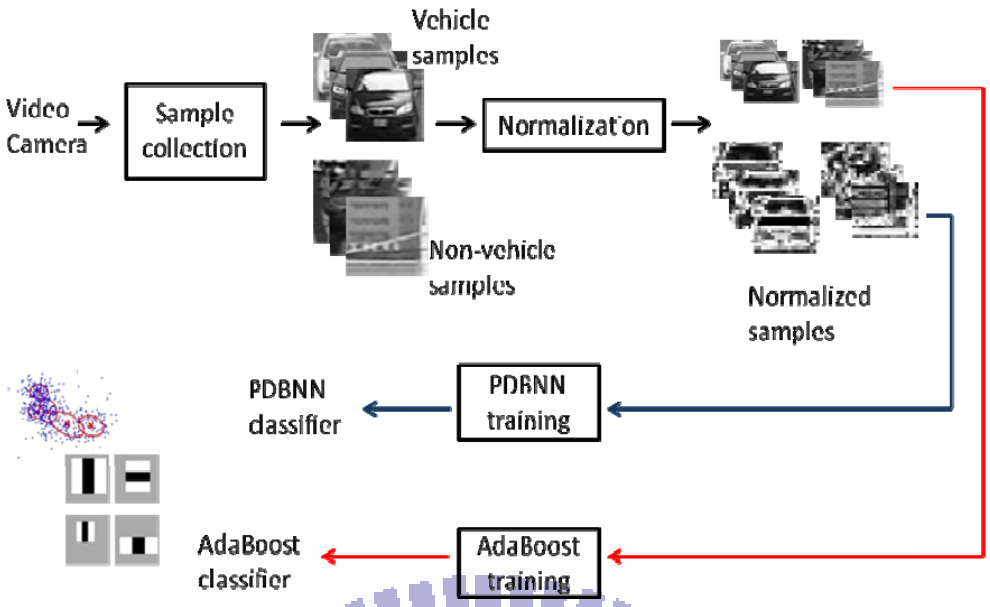


Fig. 4-3 The flow chart of model training

4.2 Model Testing

The model testing can be categorized to 1) static images and 2) videos. For the case of static images, two databases (proprietary and MIT CBCL 1999) are used for performance testing. The vehicle samples of proprietary database are tightly cropped to fit the size of vehicles, so the images are normalized to the same dimension and feature space (gray level or edge intensity) as the trained models', and compared with the models without redundant candidate regions. In MIT CBCL database, the size of vehicles are not fit to the size of images, so there are numerous candidate regions in a single image to be examined by the built classifiers, and these candidate regions are normalized to the same dimension and feature space before inputted to the classifiers. For the case of videos, every single frame is treated as a static image, and the size of video frames is 320 x 240, which is much larger than the size of collected samples, so the testing procedure of video is similar to that of MIT CBCL database. Figure 4-4 illustrates the sketch of testing the proposed system.

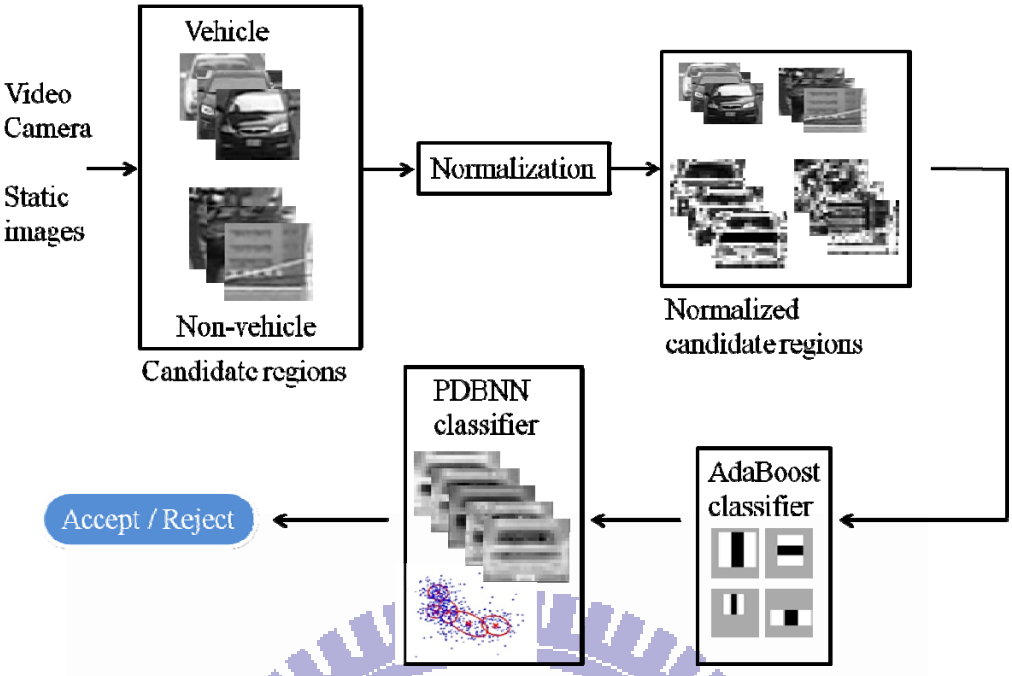


Fig. 4-4 The flow chart of model testing

4.3 Results of Detecting Vehicles in Static Images

4.3.1 Proprietary Car Database

There are 2344 positive samples (vehicle) and 30694 negative samples (non-vehicle) manually cut from proprietary video files. The positive samples comprise frontal-viewed images of typical cars, sport-utility vehicles (SUVs), and minivans.

The examples of positive and negative samples are shown in Figure 4-2. Both positive and negative samples are randomly divided into training and testing sets. The criteria of performance measurement are defined in Equation 4.1 and 4.2. The comparison of AdaBoost classifier, PDBNN classifier, and the fusion is shown in Table 4-1.

It is shown in the experimental result that the combination of two classifiers can learn the complementary relationship among local and global features, and it gains the extremely low false alarm rate and still keeps high detection rate.

Tab. 4-1 Performance comparison (proprietary car database)

	AdaBoost Classifier	PDBNN Classifier	AdaBoost Classifier + PDBNN Classifier
Detection Rate (Recall)	98.8%	96.12%	96.12%
False Positive Rate	1.2804%	0.8111%	0.0153%
Training time	7200 min	20 min	7200 min

$$Detection\ \ Rate = \mathrm{Re}call = \frac{Number\ \ of\ \ dectected\ \ vehicles}{Total\ \ number\ \ of\ \ vehicles\ \ in\ \ testing\ \ data\ \ set} \quad (4.1)$$

$$False - Positive\ \ Rate = \frac{Number\ \ of\ \ false\ \ alarms}{Total\ \ number\ \ of\ \ nonvehicles\ \ windows\ \ in\ \ testing\ \ data\ \ set} \quad (4.2)$$

4.3.2 MIT CBCL Car Database 1999

In the database, each image was extracted from raw data and was scaled to the size 128x128 and aligned so that the car was in the center of the image; the size of the cars is such that the front or rear bumper is approximately 64 pixels across. The data is presented without any normalization. Figure 4-5 shows the experimental results. There are few researches that provided the experimental result of public frontal-viewed car database. So far, R. Wang et al. [23] provided their experimental result of MIT car database, so, we compared the experimental result of [23] with that of the proposed system, and the comparison results are presented in Table 4-2. The proposed approach outperforms the method in [23] with average detection rates of 95% as well as the false positive rate of 0.002%.

Fig. 4-5 Experimental results of MIT CBCL car database

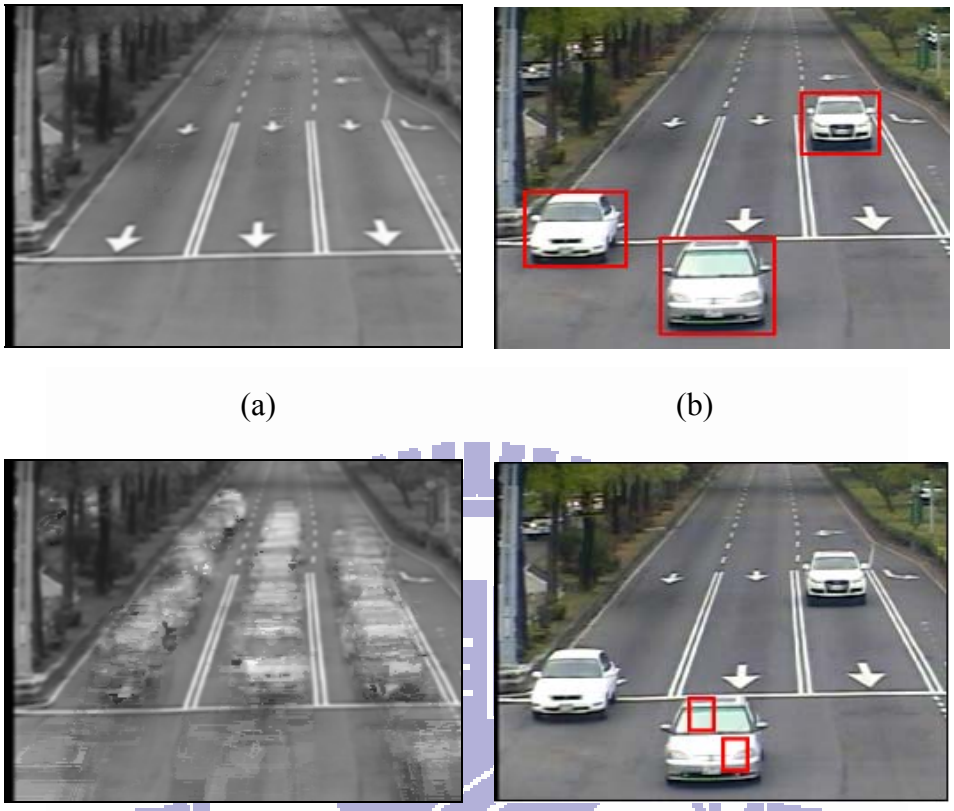
Tab. 4-2 Performance comparison (MIT CBCL car database)

	PCA + ICA (Local Features) [23]	AdaBoost Classifier	PDBNN Classifier	AdaBoost Classifier + PDBNN Classifier
Detection Rate (Recall)	95%	97.5%	96.6%	96.3%
False Positive Rate	0.002%	0.0018%	0.0078%	0.0013%

4.4 Results of Detecting Vehicles in Videos

In this section, the experimental results of detecting vehicles in videos are demonstrated. The experimental results of common used vehicle detection by background subtraction system are shown. Here, both the successful and unfortunate cases are demonstrated. Figure 4-6(a)(b) exhibits the clean background and the successful case of vehicle detection by background subtraction; Fig 4-6(c)(d) exhibits the tainted background and the unsuccessful case of vehicle

detection by background subtraction.



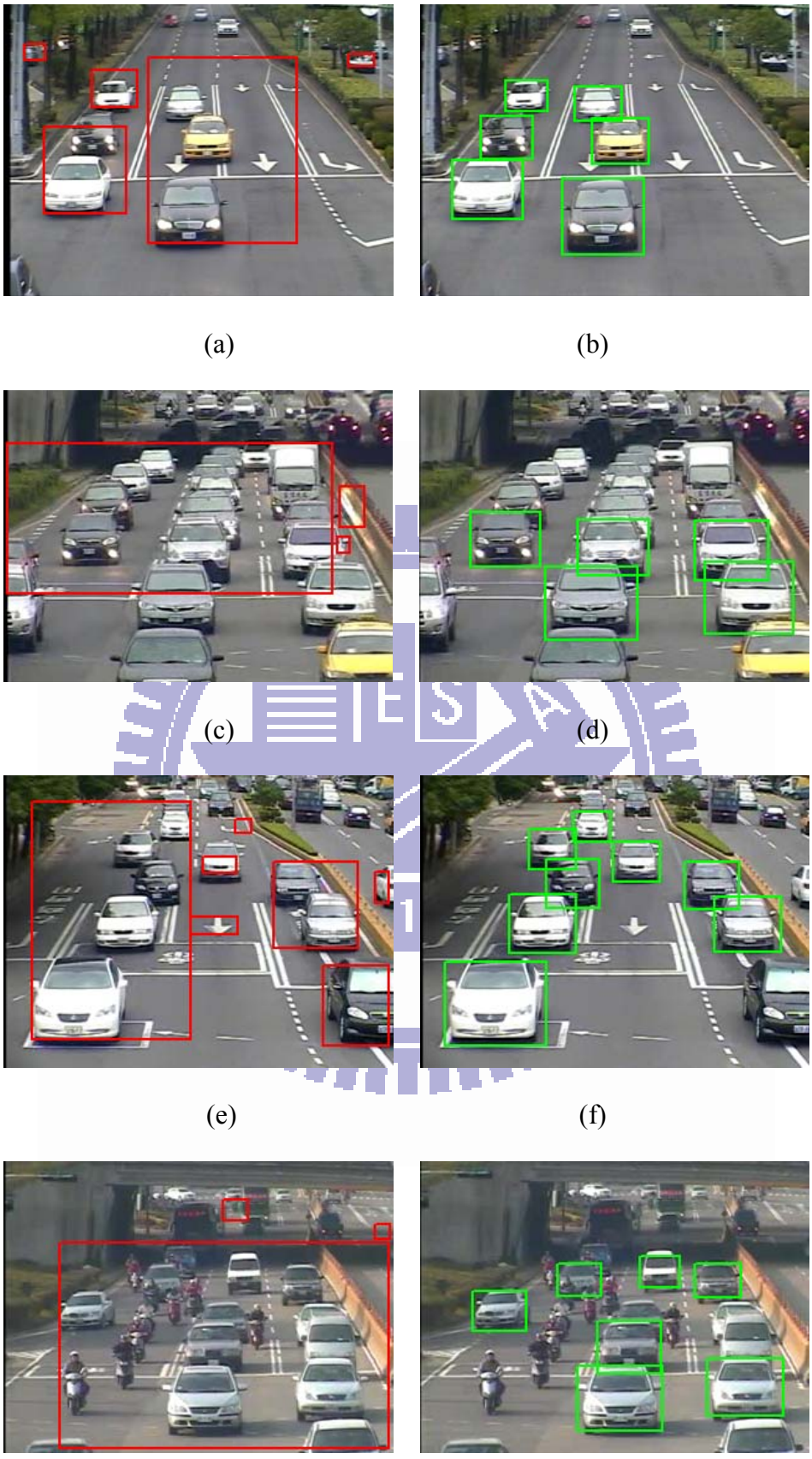
(a) (b)

(c) (d)

Fig. 4-6 Background subtraction

When the traffic is generally heavy in a scene, the background information is prone to overwhelmed by the foreground information, causes the unreliability of built background. As a result, the foreground subtracted from background is in a mess.

Figure 4-7(a)(c)(e)(g) illustrate the failed cases of vehicle detection using background subtraction in scenes with heavy traffic. Figure 4-7(b)(d)(f)(h) illustrate that the proposed system is capable of detecting vehicles in these conditions.



(a) (b)

(c) (d)

(e) (f)

(g) (h)

Fig. 4-7 The comparison of background subtraction and the proposed system

Chapter 5
Conclusions and Future Work

It is inefficient to detect the target using brute-force method. Hypothesis generation methods provide a better way to solve this problem; using local or global features solely is insufficient to extract the implicit invariance of vehicles; AdaBoost algorithm has considerable potential as a capable classifier of dense training data and provides robust generalization ability.

By combining the local features and global features of vehicles, the detection rate of proposed system is higher than others', while the false positive rate is significantly suppressed. Without relying on background information, the proposed system works well in both light and heavy traffic scenes and can be applied to both static images or video frames.

So far, the proposed vehicle detection system can operate well in variant conditions in the real environment. However, to further improve the performance of our system, some enhancements or trials can be made in the future. Firstly, vehicles cross through the intersection would meet our features and produce false positive in our testing samples due to the lack of auxiliary features which lead to the similarity in global geometric structure. Secondly, the proposed system is not yet efficient enough to be real-time. Therefore, if these problems can be solved, our system will be more applicable.

This paper demonstrates a robust system for vehicle detection, and it involves the local- and global-orientation feature extraction of vehicles and the fusion of classifications. Experimental results show the opportunity of tracking and counting systems and advanced applications.

References

[1] A. Bensrhair, M. Bertozzi, A. Broggi, P. Miche, S. Mousset, and G. Toulminet, "A cooperative approach to vision-based vehicle detection," in *Proceedings of the 4th IEEE Conference on Intelligent Transportation Systems (ITSC '01)*, pp. 207–212, Oakland, Calif, USA, August 2001.

[2] A. Kuehnle, "Symmetry-based recognition of vehicle rears," *Pattern Recognit. Lett.*, vol. 12, no. 4, pp. 249–258, Apr. 1991.

[3] T. Zielke, M. Brauckmann, and W. V. Seelen, "Intensity and edge-based symmetry detection with an application to car-following," *CVGIP, Image Underst.*, vol. 58, no. 2, pp. 177–190, Sep. 1993.

[4] S.D. Buluswar and B.A. Draper, "Color Machine Vision for Autonomous Vehicles," *Int'l J. Eng. Applications of Artificial Intelligence*, vol. 1, no. 2, pp. 245-256, 1998.

[5] D. Guo, T. Fraichard, M. Xie, and C. Laugier, "Color Modeling by Spherical Influence Field in Sensing Driving Environment," *Proc. IEEE Intelligent Vehicle Symp.*, pp. 249-254, 2000.

[6] M. Bertozzi, A. Broggi, and S. Castelluccio, "A real-time oriented system for vehicle detection," *Journal of Systems Architecture*, vol. 43, no. 1–5, pp. 317–325, 1997.

[7] N. Matthews, P. An, D. Charnley, and C. Harris "Vehicle Detection and Recognition in Greyscale Imagery," *Control Eng. Practice*, vol. 4, pp. 473-479, 1996.

[8] C. Goerick, N. Detlev, and M. Werner, "Artificial Neural Networks in Real-Time Car Detection and Tracking Applications," *Pattern Recognition Letters*, vol. 17, pp. 335-343, 1996.

[9] U. Handmann, T. Kalinke, C. Tzomakas, M. Werner, W.v. Seelen "An image processing system for driver assistance," *Image and Vision Computing, 2000 - Elsevier*, Volume 18, Issue 5, April 2000, Pages 367-376

[10] C. Demonceaux, A. Potelle, and D. Kachi-Akkouche, “Obstacle detection in a road scene based on motion analysis,” *IEEE Transactions on Vehicular Technology*, vol. 53, no. 6, pp. 1649–1656, 2004.

[11] A. Giachetti, M. Campani, and V. Torre, “The Use of Optical Flow for Road Navigation,” *IEEE Trans. Robotics and Automation*, vol. 14, no. 1, pp. 34-48, 1998.

[12] J. Collado, C. Hilario, A. de la Escalera, and J. Armingol, “Model Based Vehicle Detection for Intelligent Vehicles,” *Proc. IEEE Intelligent Vehicles Symp.*, 2004.

[13] K. She, G. Bebis, H. Gu, and R. Miller, “Vehicle Tracking Using On-Line Fusion of Color and Shape Features,” *Proc. IEEE Int'l Conf. Intelligent Transportation Systems*, 2004.

[14] J. Wang, G. Bebis, and R. Miller, “Overtaking Vehicle Detection Using Dynamic and Quasi-Static Background Modeling,” *Proc. IEEE Workshop Machine Vision for Intelligent Vehicles*, 2005.

[15] A. Khammari, E. Lacroix, F. Nashashibi, and C. Laurgeau “Vehicle Detection Combining Gradient Analysis and AdaBoost Classification,” *Proc. IEEE Conferences on Intelligent Transportation Systems*, 2005, pp. 66-71.

[16] M. Betke, E. Haritaglu and L. Davis, “Multiple Vehicle Detection and Tracking in Hard Real Time,” *IEEE Intelligent Vehicles Symposium*, pp. 351–356, 1996.

[17] J. Ferryman, A. Worrall, G. Sullivan, and K. Baker, “A Generic Deformable Model for Vehicle Recognition,” *Proceedings of British Machine Vision Conference*, pp. 127–136, 1995.

[18] Z. Sun, G. Bebis, and R. Miller, “On-Road Vehicle Detection Using Gabor Filters and Support Vector Machines,” *Proc. IEEE Int’l Conf’ Digital Signal Processing*, July 2002.

[19] S. Y. Kung and J. S. Taur, “Decision-based neural networks with signal/image classification applications,” *IEEE Trans. Neural Networks*, vol. 6, pp. 170-181, Jan 1995.

[20] Shan-Hung Lin, Sun-Yuan Kung and Long-Ji Lin “Face Recognition/Detection by Probabilistic Decision-Based Neural Network” *IEEE Transactions on Neural Networks*, Vol.

8, No. 1, January 1997.

[21] Ruck, D.W., S.K. Rogers, M. Kabrisky, M.E. Oxley and B.W. Suter, "The multilayer perceptron as an approximation to a Bayes optimal discriminant function," *IEEE Trans. Neural Networks*, 1990, 1(4), 296-298.

[22] Z. Sun, R. Miller, G. Bebis and D. Dimeo, "A Real-time Precrash Vehicle Detection System," *IEEE Intelligent Vehicles Symposium 2000*. Dearborn, MI, USA.

[23] Chi-Chen Raxle Wang and Jenn-Jier James Lien "Automatic Vehicle Detection Using Local Features – A Statistical Approach" *IEEE Transactions on Intelligent Transportation Systems*, Vol. 9, No. 1, March 2008.

[24] Paul Viola, Michael J. Jones "Robust Real-Time Face Detection" *International Journal of Computer Vision* 57(2), 137–154, 2004 Kluwer Academic Publishers.

[25] C. Papageorgiou and T. Poggio, "A trainable system for object detection," *International Journal of Computer Vision*, vol. 38, no. 1, pp. 15–33, 2000.

[26] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proceedings of the 13th International Conference on Machine Learning (ICML '96)*, pp. 148–156, Bari, Italy, July 1996.

[27] Pablo Negri, Xavier Clady, Shehzad Muhammad Hanif, and Lionel Prevost "A Cascade of Boosted Generative and Discriminative Classifiers for Vehicle Detection" *EURASIP Journal on Advances in Signal Processing* Volume 2008, Article ID 782432, 12 pages.

[28] C. Papageorgiou, M. Oren, and T. Poggio "A general framework for object detection," *International Conference on Computer Vision*, 1998.

[29] P. Viola, M. J. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance." *The 9th ICCV, Nice, France*, volume 1, pages 734–741, 2003.

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," in *J. Roy. Statist. Soc.*, B39, 1976, pp. 1-38.

[31] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, vol. 78, pp. 1481-1497, Sept. 1989.

[32] L. Xu, M. I. Jordan, and G. E. Hinton, "A modified gating network for the mixture of experts architecture," in *Proc. World Congr. Neural Networks*, San Diego, CA, 1994, pp. II405-II410.

[33] M. Betke, E. Haritaoglu, and L. S. Davis, "Multiple vehicle detection and tracking in hard real-time," in *Proc. IEEE Intell. Veh. Symp.*, 1996, pp. 351–356.