UNIVERSITY OF NEWCASTLE, AUSTRALIA

Feature Selection for Intelligent Transportation Systems

A Dissertation submitted in fulfilment of the degree of Doctor of Philosophy

Yu Peng

April, 2014

The UNIVERSITY of NEWCASTLE
AUSTRALIA

Contents

Acknowledgement

I would like to express the deepest appreciation to my supervisors for mentoring and supervising me without any reservation. This thesis would have never been completed without their guidance and help.

Thanks also go to the China Scholarship Council (CSC) and CSC-Newcastle Joint Scholarship, who support this work.

Finally, I would like to thank my parents, sister and wife for their love, care, trust and faithful support. I deeply appreciate this family I am so lucky to have.

List of Figures

List of Tables

List of Abbreviations

ANN----------Artificial Neural Network

ASIFT----------Affine Scale Invariant Feature Transform

BOF----------Bag of Features

CCMLF----------Crowd Counting using Multiple Local Features

CUDA----------Computer Unified Device Architecture

CVFE----------Correct Vehicle Front Extraction

FD----------False Detection

FP----------Foreground Pixel

GA----------Genetic Algorithm

GMM----------Gaussian Mixture Model

GPU----------Graphics Processing Unit

HF----------Hybrid Features

HLF----------High-Level Features

HaLF----------Haar-Like Features

HM----------Horizontal FP Mean

HOG----------Histogram of Oriented Gradient

HT----------Hough Transform

HV----------Horizontal FP Variance

ITS----------Intelligent Transportation System

KNN----------K Nearest Neighbour

LDA----------Linear Discriminant Analysis

LLF----------Low-Level Features

LP----------License Plate

LSF----------Line Segment Features

LUT----------Look Up Table

MGA ----------Multiple-eigenspcae with Genetic Algorithm

MLP----------Multi-Layer Perceptrons

OCR----------Optical Character Recognition

PCA----------Principal Component Analysis

PD----------Positive Detection

RANSAC----------RANdom SAmple Consensus

ROI----------Region of Interest

SGA---------- Single-eigenspcae with Genetic Algorithm

SGD----------Stochastic Gradient Descent

SIFT----------Scale Invariant Feature Transform

SIMD----------Single Instruction Multiple Data

SPM----------Spatial Pyramid Matching

SPR----------Statistical Pattern Recognition

SURF----------Speeded Up Robust Features

SVM----------Support Vector Machine

TES----------Top EigenSpace

TTC----------Time-to-Collision

UAV----------Unmanned Aerial Vehicle

VM----------Vertical FP Mean

VV---------- Vertical FP Variance

VWD----------Visual Word Dictionary

Abstract

This thesis addresses the problem of feature selection for developing vision-based applications for intelligent transportation systems. The aim is to establish an efficient and robust framework for feature selection to guide the development of vision-based applications for intelligent transportation systems.

Traditional traffic surveillance relies on monitoring by human operators. The developments of computer vision techniques and traffic cameras provide new solutions. Intelligent vision-based applications can work continuously without rest, which allows them to monitor and manage the traffic more efficiently than human beings. To understand the correlation and mechanism between these applications fully, we categorise all of the applications into four vision-based cognitive systems, specifically, vehicle, pedestrian, driver, and traffic infrastructure.

Similar to the human cognitive system, cognitive systems in intelligent transportation systems recognise objects and events based on features. Feature selection is extremely critical for achieving good performance. However, this task presents a considerable challenge because an intelligent transportation system usually contains a complicated environment, multiple objects, and an unstable background. Previous studies have indicated that feature selection is usually performed at random and without convincing reasons. To address this problem, we originally propose an efficient framework for feature selection to guide the development of cognitive systems in intelligent transportation systems. More specifically, our framework includes the scheme of maximum dependency and minimum redundancy. The first scheme guides us to select a feature candidate set such as low-level, high-level, or hybrid features; next, the cognitive system targets such as vehicles, pedestrians, drivers, or roads with a system performance requirement that is evaluated in terms of the accuracy, processing speed and robustness. Subsequently, the scheme of minimum redundancy helps us to select the optimal feature subset from the candidate set by analysing the correlations among the group of features.

In the remainder of this work, we focus on the integration of feature selection schemes into real-world cognitive systems for intelligent transportation. Vehicles are the primary road users. Vehicle surveillance is one of the most important components

of an intelligent transportation system. Using the proposed framework of feature selection, a vehicle classification system with roadside cameras is proposed. As another main road user, pedestrians have different attributes, such as irregular shapes and frequent occlusions. A pedestrian detection and counting system is developed with the guidance of the framework of feature selection. Additionally, intelligent vehicles are important cognitive systems in an intelligent transportation system. With a monocular camera installed on the front of a vehicle, the front vehicle on the road is detected, and its pose is estimated simultaneously. This system progresses toward being a better driver assistant in the future. Finally, we propose an inference bag-of-features model that selects the optimal feature subset. We demonstrate the advances of this proposed model through testing on an extremely challenging task: gender classification based on face recognition.

These aforementioned cognitive systems are the most important and essential components of intelligent transportation systems. Each system is a complex system rather than a single task. For example, in a vehicle type classification system, license plate detection and vehicle front extraction are considered together before classifying the types of vehicles. The proposed schemes for feature selection that are applied during the overall procedure for each system and the advantages of our feature selection schemes are shown next.

Throughout this work, focussed emphasis is placed on performing a thorough performance evaluation for both the methodology and the real-world datasets. Several datasets that are used in this thesis have been made publicly available for further research in this field. Our results indicate that a significant improvement in performance is achieved by using our feature selection methods. This thesis concludes with a critical analysis of the work and an outlook for future research opportunities.

Chapter 1 Introduction

In recent years, the ever-increasing need for mobility has overflowed in major cities in the form of vehicles and pedestrians, which has resulted in growing traffic congestion accompanied by reduced efficiency in the transportation infrastructure, increased travel time, air pollution, fuel consumption, unpredicted emergencies, and accidents. These facts reflect the dramatic inefficiencies of transportation systems. Hence, the development of systems for more efficient and safer mobility is urgently required.

To tackle this problem, there are three main countermeasures, worldwide: infrastructure improvement, policy adjustment and intelligent transportation systems (ITSs). Infrastructure improvement comprises grade separation, which uses bridges or tunnels to free movements from having to stop for other crossing movements, local-express lanes that provide through-lanes that bypass junction on-ramp and off-ramp zones, limited-access roads that limit the types and numbers of driveways along their lengths, and more techniques. Policy adjustment includes parking restrictions that make use of motor vehicles by increasing the cost of parking and road pricing that charges money for accessing a road/specific area at certain times or at certain congestion levels or for certain road users. Obviously, improving the infrastructure is necessary while developing a society or community. However, good traffic conditions that are created by improved infrastructure can exist only for a short period of time because the increase in the number of traffic users never stops. Nevertheless, infrastructure cannot be improved forever. At the same time, policy adjustment is only a makeshift approach to addressing the traffic problem over a short period of time with administrative means, and it cannot touch the heart of the matter. Nevertheless, the aim of the ITS is to provide innovative services that are related to different modes of transport, and traffic management could help us to efficiently address the traffic problem.

1.1 Computer vision in intelligent transportation system

The aim of an ITS is to develop more efficient and safer transportation. ITSs are composed of innovative and cost-effective mobile services and applications that incorporate electronic, computer, and communication technologies into vehicles and roads for monitoring traffic conditions, reducing congestion, detecting accidents, and

more tasks (Figueiredo, Jesus et al. 2001) (Wang, Herget et al. 2005). Transportation resonates as both a means of communication and a powerful metaphor for communication (Carey 2009). Other scholars have established the foundation for this type of analysis by calling attention to the increased information and automated regulation of spaces and practices (Hommels 2005) (Wood and Graham 2006). Recently, researchers have attempted to apply findings from the area of cognitive networks to the field of ITSs (Dimitrakopoulos and Demestichas 2010) (Ryan, Daniel et al. 2006). This application can be facilitated by exploiting cognitive networking principles. The "cognitive" aspect incorporates a spectrum of cognitive behaviours, from goal-based decisions to proactive adaption when applied to communication technologies. The term cognitive, as used in this thesis, follows in the footsteps of the definition used in (Dimitrakopoulos and Demestichas 2010). We associate cognitive systems with machine learning, which is a set of algorithms that can retain knowledge from past interactions with the environment, transform this knowledge into experience, and plan future actions accordingly. By introducing cognitive systems into an ITS, we obtain adaptability to new traffic contexts, which facilitates cooperation and also addresses complexity by developing transportation management mechanisms that have learning capabilities. These systems enable prior perceived potential and, accordingly, amend behaviour with respect to traffic.

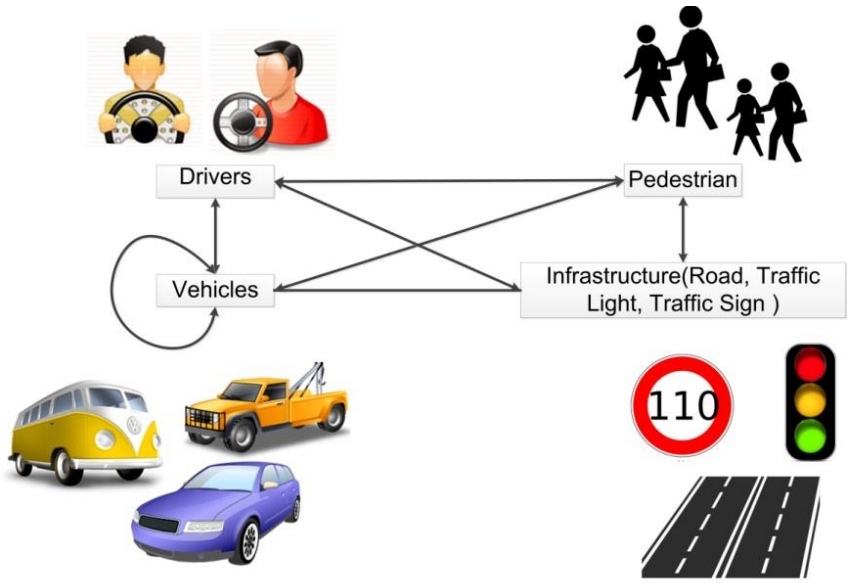


Fig. 1.1 Cognitive systems among vehicles, drivers, pedestrians, and infrastructure, which are the four main nodes in an intelligent transportation system.

Cognitive systems in an ITS offer functionality that is capable of exploiting the intelligence that is accumulated through the exchange of information among nodes, such as vehicles, drivers, pedestrians, and traffic control centres. From this viewpoint, we can recognise that the most distinguishable difference between an ITS and traditional transportation is communication and decision making transfers, transitioning from methods that require human intervention to an automatic system. Cognitive systems improve the performance and also the reliability of communication and decision making. As shown in Fig. 1.1, an ITS comprises four nodes: vehicles, drivers, pedestrians and infrastructure, including roads, traffic signs, traffic lights and traffic control centres. The whole ITS is composed of all of these cognitive systems, each of which works between every pair of nodes. As shown in Fig. 1.2, the operation of a cognitive system that is placed between two adjacent nodes can be reflected in a feedback loop. The system first retrieves context information such as vehicles, pedestrians, velocities and positions of neighbouring vehicles. Through the analysis of this information, while considering its own preferences, goals, and policies, the system decides on its actions, such as issuing a directive toward a traffic control centre to change the traffic light or sending an instruction to a driver to change the vehicle's direction. More importantly, the output of the system is stored in a knowledge database for future reference. The system keeps track of its actions, learns from its actions and facilitates future decisions. This scenario is repeated in a machine learning (Bishop 2006) process that leads to cognition. The aforementioned knowledge and experience is learned while the system runs. Moreover, the knowledge can be learned "offline" and/or can be improved after the system runs.

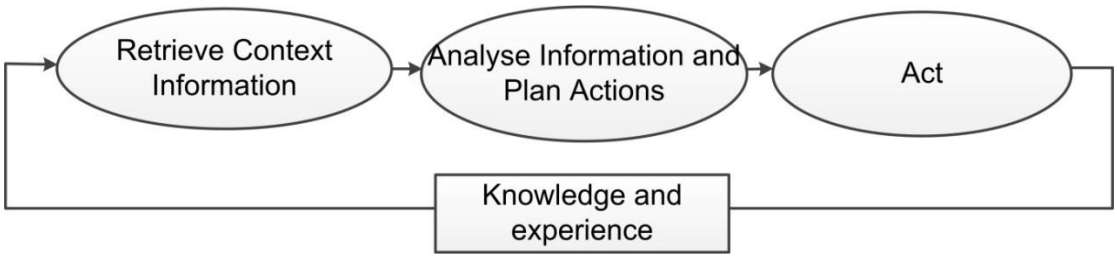


Fig. 1. 2 Operation procedure of each cognitive system.

Referring to Fig. 1.2, we require the help of sensors for information retrieval and knowledge learning in the cognitive system between two nodes. As shown in Fig. 1.3, the majority of sensors can be categorised into two groups: active sensors and passive sensors (Hebert 2000). Active sensors, such as radar-based (Park, Kim et al. 2003), laser-based (Wang, Thorpe et al. 2003) (Hancock, Hoffman et al. 1998), and acoustic-based (Chellappa, Gang et al. 2004) sensors are called active because they can detect the distance to an object by measuring the travel time of a signal that is emitted by a sensor and reflected by the object. The main advantage of active sensors against passive sensors is that they can measure a certain distance directly without requiring a powerful computing resource. However, with the rapid development of hardware and computational ability, it is not obvious that there is an advantage to the active sensors. Nevertheless, active sensors suffer from serious drawbacks that they are expensive to install and maintain. Visual sensors, such as cameras, are usually referred to as passive sensors because they acquire data in a nonintrusive way. Visual sensors offer a relatively low installation cost and cause little traffic disruption during maintenance. Furthermore, they capture much more information than active sensors. With the information that is acquired by the active sensors, we can implement multiple tasks, such as the analysis of traffic flows and turning movements, speed measurements, multiple-point vehicle counts, vehicle classification and road state assessments.

Fig. 1.3 Sensors used in an intelligent transportation system. According to the mechanism of receiving information, the sensors are divided into two groups: active sensors and passive sensors.

Fig. 1.4 The cognitive system helps deriver to have a better understanding of the environment. LightSpeed display shows information on the windscreen. © LightBlueOptics.

With respect to the 4 nodes in Fig. 1.1, we are going to demonstrate some applications to show the importance of computer vision in an ITS. In the cognitive system between the vehicle and traffic control centre, the images/videos of vehicles are captured by surveillance cameras; they are analysed automatically and then sent to the traffic control centre to aid in suitable decision making. License plate recognition (Jia, Zhang et al. 2007) (Peng, Xu et al. 2011) (Duan, Duc et al. 2004) and vehicle type classification (Petrovic and Cootes 2004) (Peng, Jin et al. 2012) (Peng, Jin et al. 2013) are typical applications. With computer vision techniques, vehicles as primary road users can perceive the infrastructure, with road lane detection (Paetzold and Franke 2000) (Wang, Chung et al. 2004) (Kluge 1994) and traffic sign recognition (Negri, Clady et al. 2008) (Soetedjo and Yamada 2005) (Hoferlin and Zimmermann 2009), using an on-board camera. As another road user, pedestrians can also use the help of visual sensors to understand the infrastructure. For example, with the help of location information and a smart mobile camera, a good solution for finding a perfect rendezvous is proposed in (Liu, Mei et al. 2012). Similarly, the traffic control centre usually detects pedestrians and even analyses the motion of pedestrians (Peng, Xu et al. 2012) (Pai, Tyan et al. 2004) (Besbes, Rogozan et al. 2010), using a surveillance camera. Drivers who sit in vehicles can recognise the environment by themselves. However, they still can use the help of visual sensors to enhance and accelerate the perceptions between them and other nodes in the ITS. To allow drivers to perceive the road environment better, Light Blue Optics proposed a technique, which is Light

Speed (Optics 2012), as shown in Fig. 1.4, to display vital information such as speed, road warnings and GPS information at an apparent distance of 2.5 meters from the driver's eyes. The display appears to be floating off the end of the vehicle's bonnet, reducing the need for drivers to shift their focus from the road ahead. On the other hand, computer vision is also useful in recognition between drivers and vehicles. To detect driver inattention, such as distractions and fatigue, computer vision measurements are widely applied (Yanchao, Zhencheng et al. 2011). Moreover, recognising pedestrians and other vehicles on the road using computer vision has become very attractive recently. In (Enzweiler and Gavrila 2008), an on-board camera observed the road ahead for possible collisions with pedestrians. In (Peng, Xu et al. 2012), the accurate 3D position of the front vehicle is estimated via an on-board camera.

1.2 Feature selection for an intelligent transportation system

From the aforementioned analysis, we note the importance of computer vision in all of the cognitive systems of the ITS. Computer vision stimulates or enhances the functions of the human vision system (Nixon and Aguado 2012). In human vision, the sensing element is the eye, from which images are transmitted via the optic nerve to the brain, for further processing. The optic nerve has insufficient bandwidth to conduct all of the information that is sensed by the eye. Accordingly, there must be some pre-processing to discard non-useful information while keeping important information before the image is transmitted to the optic nerve. Similarly, as shown in Fig. 1.2, it is not necessary to send all of the context information, namely, images/videos captured by cameras, to analyse and learn. We must select and extract from all of the features of the images or videos, to describe the characteristics of our objects. Feature selection removes irrelevant, redundant, or noisy data, and brings about immediate effects for applications: speeding up an algorithm or improving the application performance, such as the predictive accuracy and result comprehensibility (Huan and Lei 2005).

As shown in Fig. 1.5, feature selection is composed of two stages—feature candidate set selection and feature sub-set selection. In the first stage, we select a feature candidate set. Usually, we divide all of the features applied in an ITS into three groups—low-level features, high-level features, and hybrid features.

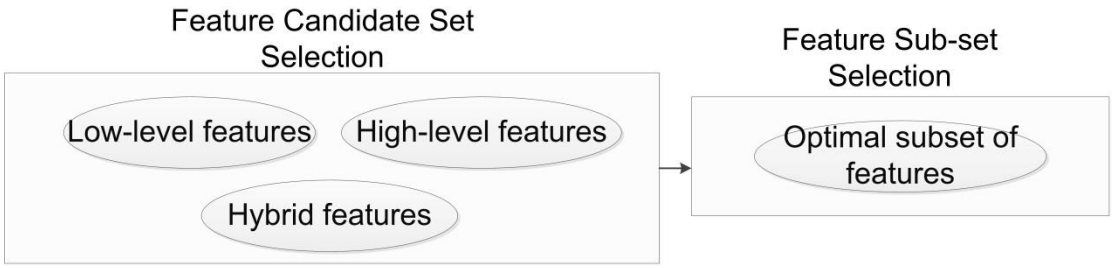


Fig. 1.5 Two steps of feature selection: feature type selection and feature subset selection. Through these two steps, the best representative features can be extracted for objects of interest.

According to the definition in (Nixon and Aguado 2012), low-level features are those basic features that can be extracted automatically from an image without any shape information (information about spatial relationships). First, low-level feature extraction, including edge detection, phase congruency, corner-based extraction (such as Moravec and Harris detectors), region/patch analysis (such as scale-invariant feature transforms (SIFT) (Lowe 2004)) and speeded-up robust features (SURF) (Bay, Ess et al. 2008). Second, high-level feature extraction finds out shapes and objects in images and videos. The methods are template matching, parts-based shape analysis, active contours and shape skeletonisation. Finally, we utilise hybrid features in some ITS applications, which are a combination of low-level features and high-level features. Obviously, no feature is universally suitable. Different types of features were chosen in different applications. For example, SIFT, a low-level feature, was chosen in (Peng, Luo et al. 2012) to classify gender with a Bag-of-Features (BoF) framework. (Peng, Xu et al. 2012) chose several statistical features, such as high-level features, to describe the spatial characteristics of pedestrian groups. (Peng, Xu et al. 2011) utilised a combination of line segment features and Haar-like features, as hybrid features, in a license plate localisation system.

The selected feature candidate set is not sufficient to achieve the best performance for ITS cognitive systems. To use the most discriminative features to describe targets, we must select an optimal subset of features. In fact, there are two types of methods for selecting the optimal subset—the filter model and the wrapper model. The filter model relies on general characteristics of the data to evaluate and select feature subsets with pre-determined criteria. For example, in the traditional Principal

Component Analysis (PCA) (Jolliffe 2005) method, a pre-defined number of eigenvectors were selected to build up the eigenspace. The wrapper model is usually a procedure of iterative evaluation. It requires some mining algorithms and uses their performance as the evaluation criteria. For example, RANdom SAmple Consensus (RANSAC) (Fischler and Bolles 1981) discarded outliers while searching for inliers and is better suited to the mining algorithm, which aims to improve the performance (such as the classification rate).

Identifying the optimal set of features is extremely important. Using most of the discriminative information that is contained in this optimal set, the most accurate model can be calculated to represent the target for classification, localisation, or recognition. It is worth noting that having the optimal amount of information (to attain the best accuracy) is important because either more or less information would deteriorate the performance.

1.3 Applications

Despite the inferiority of vision-based cognitive systems compared with human vision and the severe methodological challenges and performance demands, one of the key questions is to determine the performance that is deemed to be necessary to deploy a vision-based cognitive system in an ITS. Although there are limitations in terms of the recognition performance, artificial systems have an advantage over humans in that they do not fatigue, are always vigilant and can possible react in a small fraction of a second.

There are many applications and existing commercial systems, as shown in Fig. 1.6. We are specifically interested in the applications of feature selection in the cognitive systems that are in an ITS.

Applications include surveillance cognitive systems, in which a camera looks down a street. For example, feature selection methods are widely applied in vehicle detection and classification systems (WebSource 2012).

According to the deployment environment in which the surveillance cameras are installed and the camera angle, different feature selection methods are applied to avoid adverseness, such as the occlusion between densely spaced vehicles. License

Automotive night vision with pedestrian recognition. © Daimler AG

Traffic Surveillance System. ©ACTi

Intelligent car. ©Google

Fig. 1.6 Commercial products of visual based cognitive systems in an ITS.

plate recognition is another well-researched application that is based on feature selection. There is a series of companies, e.g., (WebSource 2012) and (WebSource 2012), which provide solutions for tolling, congestion charging, and vehicle identification. Because license plates are usually very small within the whole image, features should be carefully chosen, to obtain the best performance on both localisation and recognition. In addition to the surveillance cognitive system, another application area is intelligent vehicles, specifically, a vehicle cognitive system, where an on-board camera watches the road ahead for possible collisions with pedestrians and other vehicles (e.g., Mercedes-Benz E-Class 2009, BMW 7 series 2008 and Audi A8 2010)) or monitors the driver for inattention. Because of the different characteristics of different objects, such as walking pedestrians, vehicles and drivers, various features are selected.

All of the previously mentioned applications would significantly benefit from more advanced feature selection methods for cognitive systems, to improve their performance and to address a wider range of scenarios in ITSs.

Chapter 2 Methodology

2.1 Analyse an ITS in terms of its cognitive system

In the aforementioned analysis, computer vision and video analytics become increasingly important in ITSs, and there has been an increased scope for the automatic image/video analysis of urban traffic activity. There has been a large number of surveys of the studies in this area. However, most of these surveys focus on specific aspects of vision-based applications in an ITS, such as video surveillance or vision-based intelligent vehicles. As shown in Table 2.1, we organised representative surveys over the past decade.

All of these surveys focus on specific applications of an ITS rather than providing summaries from the methodology viewpoint that can guide future work in this field. For example, (Gelmose, Trivedi et al. 2012) provided a survey of the traffic sign detection literature, detailing systems of traffic sign recognition for driver assistance in an ITS, while (Yanchao, Zhencheng et al. 2011) focused on another part of the ITS—driver inattention monitoring, such as for distraction and fatigue.

Table 2. 1. Survey papers of computer vision in intelligent transportation systems over the past decade.

Year	Title	Focus	Cognitive systems in an ITS
2012	Vision-Based Traffic Sign Detection and Analysis for Intelligent Driver Assistance Systems Perspectives and Survey (Gelmose, Trivedi et al. 2012)	Traffic sign detection and analysis	Infrastructure to Vehicle
2011	A Review of Computer Vision Techniques for the Analysis of Urban Traffic (Buch, Velastin et al. 2011)	Vehicle detection, vehicle tracking and vehicle classification	Infrastructure to Vehicle
2011	Driver Inattention Monitoring System for Intelligent Vehicles: A Review (Yanchao, Zhencheng et al.	Driver inattention such as distraction and fatigue	Vehicle to Driver

	2011)	detection	
2009	Vehicular Communication Systems Enabling Technologies Applications and Future Outlook on Intelligent Transportation (Papadimitratos, La Fortelle et al. 2009)	Roadside and on-board equipment communication	Vehicle to Vehicle
2009	Monocular Pedestrian Detection Survey and Experiments (Enzweiler and Gavrila 2009)	Pedestrian detection using an on-board camera	Vehicle to Pedestrian
2008	Study of Robust and Intelligent Surveillance in Visible and MultiModal Framework (Kumar, Mittal et al. 2008)	Pedestrian detection using a distributed surveillance system	Infrastructure to Pedestrian
2008	Inter Vehicle Communication Systems a Survey (Sichitiu and Kihl 2008)	Communication systems between vehicles	Vehicle to Vehicle
2006	On Road Vehicle Detection: A Review (Sun, Bebis et al. 2006)	Vehicle detection using an on-board camera	Vehicle to Vehicle
2005	Intelligent Distributed Surveillance Systems: A Review (Valera and Velastin 2005)	Objects, especially pedestrian, detection and analysis using a distributed system	Infrastructure to Pedestrian
2003	Detecting Moving Shadows Algorithms and Evaluation (Prati, Mikic et al. 2003)	Detecting and removing shadows using a road-side surveillance camera	Infrastructure to Infrastructure
2003	A Survey of Video Processing Techniques for Traffic Application (Kastrinaki, Zervakis et al. 2003)	Detecting lanes and objects using a road-side surveillance camera or an on-board camera	Infrastructure to Infrastructure, Infrastructure to Vehicle

2002	The Development of Machine Vision for Road Vehicles in The Last Decade (Dickmanns 2002)	Intelligent vehicle	Vehicle to the other three nodes
2002	Building Safer Cars (Jones 2002)	Intelligent vehicle	Vehicle to the other three nodes
2002	Artificial Vision in Road Vehicles (Bertozzi, Broggi et al. 2002)	Intelligent vehicle	Vehicle to the other three nodes

As shown in Fig. 1.1, from the viewpoint of considering cognitive systems in an ITS, all of this review work fell into one of the cognitive systems among the four "nodes": infrastructure, vehicle, pedestrian and driver. The operation of a cognitive system placed between two nodes can be reflected in a feedback loop, as explained in Chapter 1.

The cognitive system between a vehicle and driver usually retrieves the information on the driver by the camera or other sensors that are installed inside a vehicle, to evaluate the state of the driver and, accordingly, make a decision. (Yanchao, Zhencheng et al. 2011) presented a comprehensive review of current technology for monitoring driver inattention, which includes distraction and fatigue. For example, by recognising a driver's face, Fan *et al.* (Fan, Sun et al. 2010) constructed a classifier for fatigue detection; by measuring the pressure distribution on the seat of male subjects using simulated long-term driving, Furugori *et al.* (Furugori, Yoshizawa et al. 2005) showed good results on fatigue detection. The authors of (Yanchao, Zhencheng et al. 2011) demonstrated that combining a driver's physical measurements with driving performance measures can intuitively increase the inattention detection confidence. In addition to drivers, vehicles can communicate with other nodes through the cognitive system as well. In the cognitive system between a vehicle and a pedestrian, an on-board camera watches the road ahead for possible collisions with pedestrians (Enzweiler and Gavrila 2009). With the camera fixed on the roadside, vehicles (Buch, Velastin et al. 2011) and pedestrians (Kumar, Mittal et al. 2008) were detected and analysed for traffic management. At the same time, transport infrastructure such as road lanes (Prati, Mikic et al. 2003) and traffic signs (Gelmose, Trivedi et al. 2012) were recognised by vehicles through the cognitive system

between them. Moreover, a cognitive system between vehicles is also popular in ITSs. Inter-vehicle communication systems (Papadimitratos, La Fortelle et al. 2009) demonstrated the potential for radically improving the safety, efficiency, and comfort of everyday road travel. Using a camera mounted on the vehicle, the cognitive system detected the front vehicle on the road and issued a directive to the drivers to make the correct decision (Sun, Bebis et al. 2006). Some concepts of intelligent vehicles were proposed in (Bertozzi, Broggi et al. 2002) (Dickmanns 2002) (Jones 2002). These intelligent vehicles communicated with pedestrian, driver, infrastructure and other vehicles in ITSs through cognitive systems.

2.2 Efficient feature selection scheme for an ITS

To make appropriate decisions, these aforementioned vision-based cognitive systems must receive information from the environment by using cameras and, then, analyse and recognise them. The objective in a complex environment could be located and analysed by a human vision system with little effort, whereas this task is extremely challenging for cognitive systems in ITSs. Noise is one of the primary hardships. In general, image noise means random variation in the brightness or colour information in images, which is usually an aspect of electronic noise. This type of noise is always produced by the sensor and circuitry of the digital camera. In this thesis, a broader concept of noise is preferred—any "unwanted information" is regarded as noise. The "wanted information" depends on specific applications. For example, the objective of a pedestrian counting system (Peng, Xu et al. 2012) was to count pedestrians only. Any other information, such as tree branches, traffic signs and vehicles, are noise. In contrast, vehicles rather than pedestrians constituted the "wanted information" for a vehicle type classification system (Peng, Jin et al. 2012). To enhance the accuracy of the analysis and recognition performance in the cognitive system, reducing noise while retaining the objective information is critical.

In all of the cognitive systems in an ITS, feature selection is a significant step toward identifying the most characteristic features of the captured data (images or videos). In this thesis, we propose efficient feature selection schemes for vision-based cognitive systems in an ITS via feature dependency and feature redundancy analysis. Given the input data D and the task performance variable c, the feature selection problem is to find X and a subspace of m features from the M-dimensional feature space

$X = \{x_i, i = 1, \dots, M\}$, in such a way as to optimally characterise c. The task performance could be determined by the classification accuracy (a), robustness (r) and processing speed (t).

$$Max\ c(X, m)$$

$$c = w_a \times a + w_r \times r + w_t \times t \tag{2.1}$$

To achieve the maximum c, we need the optimal X and m. This problem becomes which types of features and which subsets of features we must choose, therefore, to maximise the performance of the cognitive systems in the ITS. Here, c is determined by weighted performance requirements. Different ITS applications have different performance requirements. Obviously, a high accuracy is always desired. Some applications, such as intelligent vehicle systems (Peng, Xu et al. 2012), strictly require real-time processing. At the same time, other applications, such as a vehicle classification system (Peng, Jin et al. 2012), do not have a restrict real-time requirement. Therefore, we could compromise some speed for high-accuracy features in an ITS system. Attention should be paid to robustness in various environments for developing applications such as urban traffic systems, where changing weather and complex streets have a substantial effect on the performance. On the other hand, some applications, such as vision-based access control systems, are implemented in a relatively stable environment. In such cases, robustness is not the priority.

2.2.1 Maximum dependency

Existing feature selection methods mainly focus on finding an optimal subset of features (John, Kohavi et al. 1994) (Kohavi and John 1997) (Kwak and Choi 2002) (Peng, Long et al. 2005). The analysis of feature type selection is always ignored. In this thesis, we define feature dependency and feature redundancy. Based on the analysis of feature dependency and redundancy, a new framework of feature selection is introduced that decouples feature type selection and feature subset selection.

In the previous analysis, the optimal performance often means a weighted summation of the accuracy, processing time, and robustness. In an unsupervised situation in which the classifier is not specified, high accuracy and speed processing usually require the maximal statistical dependency of the target o on the data distribution.

This scheme has maximal dependency, which means that the best performance usually requires the maximal dependency of the suitable features on a specific target. In the pedestrian counting system (Peng, Xu et al. 2012), the authors utilised some shape-related features, such as the width, height, and foreground pixel distribution to characterise moving blobs and to indicate the number of pedestrians in each moving blob. The experiments demonstrated promising performance. However, these features cannot achieve good results in other cognitive systems in an ITS. For example, facial expression recognition in the aforementioned driver fatigue detection system needed other features, such as Gabor features (Fan, Sun et al. 2010), to describe the face, which was much more subtle than the techniques used in the pedestrian recognition problem.

Dependency is usually characterised in terms of mutual information, which is one of the widely used measures to define dependency among variables (Peng, Long et al. 2005). Given two random variables x and y, their mutual information is defined in terms of their probabilistic density functions $p(x), p(y)$, and $p(x, y)$:

$$I(x: y) = \int\int p(x, y) log \frac{p(x,y)}{p(x)\, p(y)} dxdy \tag{2.2}$$

In the scheme of maximum dependency, the selected features x are required, individually, to have the largest mutual information $I(x: o)$ with the target o, reflecting the largest dependency on the target o. Obviously, a set of features (M) rather than a single feature would be utilised. The dependency formula takes the form of:

$$I(X: o) = \int\int p(X, o) log \frac{p(X, o)}{p(X)\, p(o)} dXdo$$

$$= \int\int p(x_1, \dots, x_M, o) log \frac{p(x_1, \dots, x_M, o)}{p(x_1, \dots, x_M)\, p(o)} dx_1, \dots, dx_M do \tag{2.3}$$

In the analysis in (Peng, Long et al. 2005), it is very difficult to obtain an accurate estimation for the multivariate density $p(x_1, \dots, x_m)$ and $p(x_1, \dots, x_m, o)$ because the multivariate density estimation often involves computing the inverse of the high-dimensional covariance matrix, which is usually an ill-posed problem.

In an ITS, the types of targets and utilised features are usually limited. Target o could be the vehicle, pedestrian, road, and traffic sign. Feature X could be edge-based features, patch-based features, or shape-based features. Rather than calculating the mutual information between the feature and target directly, we establish the relationship between the features and target through the analysis of the feature characteristics. Via feature analysis, we categorise all of the features that are applied in the ITS into three groups: low-level features, high-level features and hybrid features. This thesis discusses these three groups of features and reviews their applications in cognitive systems in ITSs in Chapter 3. The analysis of the characteristics of the features guides us to select suitable feature types for applications. In the end, a candidate set of features is chosen via the evaluation in terms of the performance c. Therefore, the scheme of maximum dependency has the following form:

$$Max\ D(X, o, c), \{X, X \in LLF, HLF, HF\} \tag{2.4}$$

LLF, HLF, and HF stand for low-level features, high-level features, and hybrid features, respectively.

2.2.2 Minimum redundancy

After selecting the candidate set of features, feature subset selection is applied to reduce the number of features. In theory, more features could provide more discriminating power, but in practice, with a limited amount of training data, excessive features not only will significantly slow down the learning and testing process but will also cause the classifier to over-fit the training data because the redundant features could confound the learning process (Guyon and Elisseeff 2003) (Yang and Pedersen 1997) (Yu and Liu 2004) (Dash and Liu 1997). In other words, a vision-based system can work more efficiently and effectively with non-redundant features.

In (2.1), we have explained that the best performance is a function of the most suitable types of features and an optimal set of features, which includes the most informative features while excluding the redundant features. The scheme of maximum dependency guides us to select the most suitable types of features in the ITS. It has been recognised that the combinations of individually good features do not

necessarily lead to good classification performance. In other words, "the *m* best features are not the best *m* features" (Cover and Thomas 2006) (Webb 2003). As shown in Fig. 2.1, the maximum-dependent features are selected with the scheme of maximum dependency from low-level, high-level, or hybrid features. However, there are redundant features that exist in these dependent features. Our aim is to identify the optimal subset, which is the set of "max-dependent and min-redundant features", as shown in Fig. 2.1. To select the optimal subset of the features, we introduce the scheme of minimum redundancy.

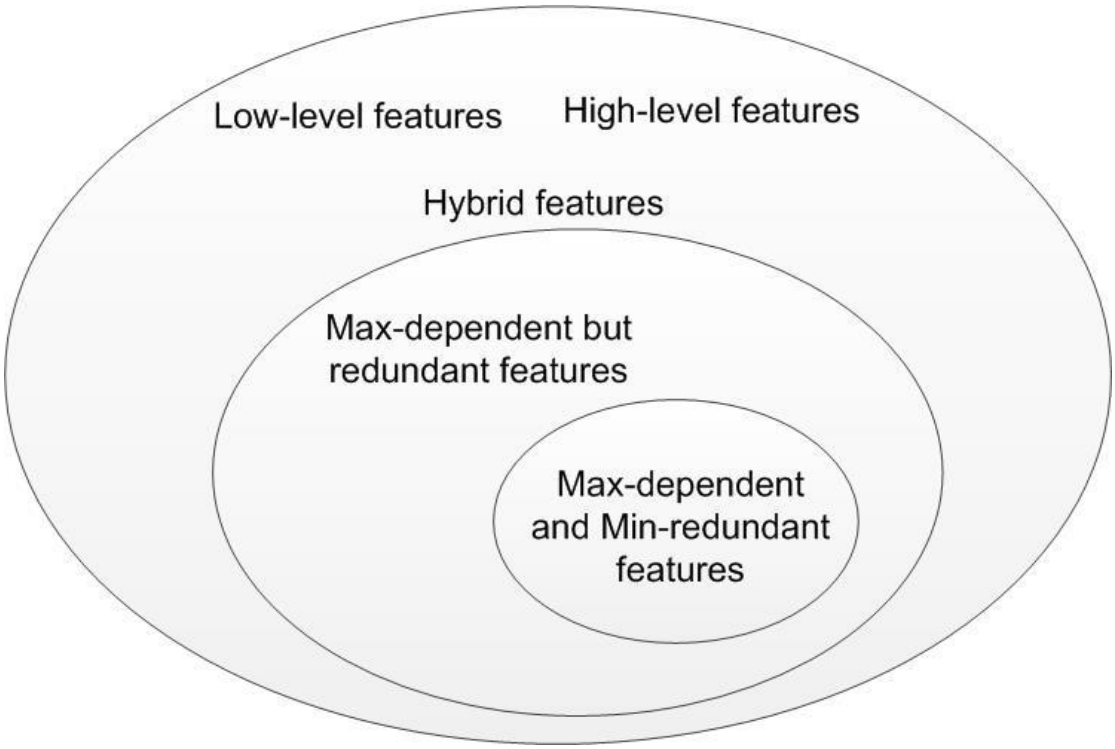


Fig. 2.1 The relationship between dependent features and redundant features. Our aim is to identify the optimal subset, which is the set of max-dependent and mini-redundant features.

In the scheme of maximum dependency, we select the candidate set of features through analysing the correlation among the features (low-level features, high-level features, and hybrid features), targets (vehicles, license plates, pedestrians, roads, and traffic signs), and performance requirements (accuracy, processing time, and robustness). In the scheme of minimum redundancy, we select the optimal subset of features via revealing the correlation between any pair of features. The basic

mechanism is that when two features depend highly on one another, the respective class-discriminative power would not change much if one of them was removed.

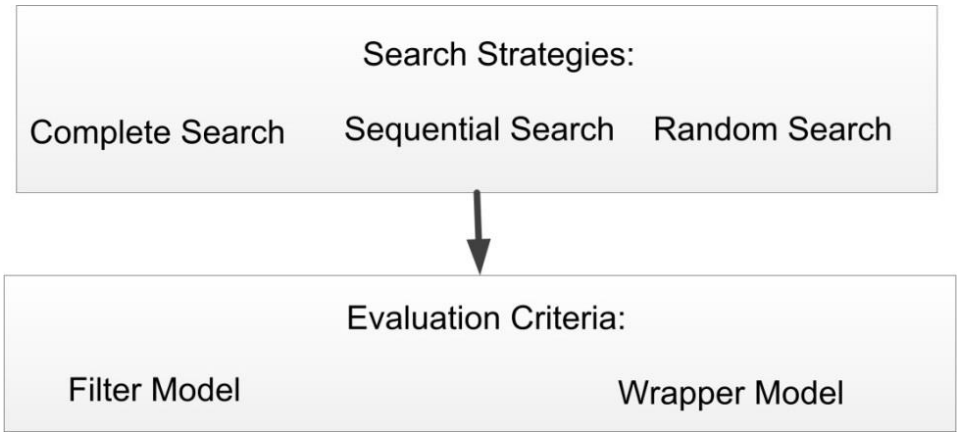


Fig. 2.2 Feature subset selection algorithms are combination of search strategies and evaluation criteria.

Referring to the mutual information formula (2.2), we propose the following minimal redundancy formula to select the optimal subset of m features :

$$minR(m), R = \sum_{x_i,x_j \in m} I(x_i; x_j), \{m, m \in M\} \quad (2.5)$$

General feature subset selection algorithms are a combination of search strategies and evaluation criteria, as shown in Fig. 2.2. Depending on the search starting point and the search strategies, the categories are complete search, sequential search, and random search. A complete search guarantees finding the optimal result according to the evaluation criterion through an exhaustive search (Narendra and Fukunaga 1977). Most sequential search methods are variations of the greedy hill-climbing approach, such as sequential forward selection, sequential backward elimination, and bidirectional selection (Liu and Motoda 1998). This strategy is faster than the previous strategy because it does not retain the goal of searching set completely. For the same reason, however, it risks losing the optimal subsets. Random search starts awith a randomly selected subset and generates the next subset in a completely random manner, such as in the Las Vegas algorithm (Cormen, Leiserson et al. 2001). At the same time, evaluation criteria broadly fall into two categories: the filter model (Dash, Choi et al. 2002) (Yu and Liu 2003) and the wrapper model (Das 2001) (Xing,

Jordan et al. 2001). The filter model evaluates the quality of a feature subset by exploiting the intrinsic characteristics of the training data without involving a mining algorithm. These characteristics include distance, information, dependency, and consistency. For example, in our previous work (Peng, Park et al. 2010), we utilised information entropy as measurement for selecting 9 features with large entropy values, to detect cervical nuclei clusters. The wrapper model requires a predetermined algorithm and uses its performance as the evaluation criterion. For example, in the task of clustering (Peng, Jin et al. 2012), the wrapper model evaluated the quality of a feature subset by the quality of the clusters that resulted from applying the clustering algorithm.

We present a hybrid scheme that combines search strategies and evaluation criteria for selecting optimal feature subsets efficiently. Our scheme solves the feature selection problem without including filter steps. Therefore, the scheme does not compromise either the accuracy or speed. The scheme of minimum redundancy organises the feature subset selection into three stages.

Stage 1: This stage guides the global search in a feature candidate space. Usually, search strategies are utilised through the feature candidate space. Every new subset is accepted at this stage to include as many features as possible.

Stage 2: This stage employs optimisation. The main purpose of this stage is to obtain possible optimal feature subsets that are optimal in terms of their performance. Through the evaluation of accuracy, processing time, and robustness, good solutions are converged on rapidly.

Stage 3: This stage applies a local search to the k-best solutions that are given in Stage 2. The algorithms perform local searches on the k-best solutions and select the best features for minimising the redundancy of the feature subset.

Suitable feature candidates X are determined by the scheme of maximum dependency. Depending of the feature candidates X and the performance requirements c for specific cognitive systems in an ITS, general feature subset selection methods are adapted and improved. In the next chapter, we will review popular methods for feature subset selection in ITS.

2.3 Implementing the feature selection scheme

The goal of our thesis is to design an efficient scheme to select the best features for a vision-based cognitive system in an ITS. In Section 2.2, we proposed an efficient feature selection scheme of maximum dependency and minimum redundancy. A remaining issue is how to implement this scheme in the development of ITS cognitive systems.

Under the guidance of a proposed feature selection scheme, we present a two-stage implementation framework. In the first stage, we find a suitable candidate feature set with the scheme of maximum dependency. To select the candidate feature set for a vision-based cognitive system in an ITS, we fully analyse the correlations among different features (low-level features, high-level features, and hybrid features), targets in cognitive systems (vehicles, license plates, pedestrians, and traffic signs), and the performance requirements of a cognitive system (accuracy, processing time, and robustness). In the second stage, we search for a compact feature subset from the candidate feature set via the scheme of minimum redundancy. With the idea of removing redundant features, many sophisticated methods that combine search strategies, and evaluation criteria can be used to acquire the compact feature subsets from the candidate set that was selected in the first stage. To achieve the best compactness, however, we must analyse the characteristics of specific features.

In the following chapters, the feature selection scheme and general two-stage implementation framework will guide us in developing several vision-based cognitive systems in ITSs. The significance and advantages of our scheme will be demonstrated in these real-world systems.

Chapter 3 Related work about feature selection

In Chapter 1, we introduced cognitive systems in ITSs and the significance of feature selection in the successful development of these cognitive systems. We explained the challenge of feature selection and briefly reviewed the current feature selection methods in Chapter 2. In addition, we proposed the feature selection scheme for visual cognitive systems in an ITS. In this Chapter, we will provide a comprehensive review of feature selection. First, we discuss feature selection in general terms. Subsequently, the previous work about feature type selection and feature subset selection, which pertains to our proposed feature selection scheme for ITSs, is reviewed comprehensively.

3.1 Feature selection in general terms

Feature selection has been an active and fruitful field of research and development for decades and is a multidisciplinary joint effort from data mining (Dash and Liu 1997) (Dash, Choi et al. 2002) (Kim, Street et al. 2000), machine learning (Hall 1999) (Sebastiani 2002) (Sun, Todorovic et al. 2010), and statistics (Saeys, Inza et al. 2007) (Vasconcelos and Vasconcelos 2009). A successful choice of feature selection has been proven in both theory and practice to effectively enhance the learning efficiency while increasing the predictive accuracy and reducing the complexity of the learned results. Feature selection widely applies to many fields, such as text categorisation, image retrieval, customer relationship management, intrusion detection, and genomic analysis. Feature selection champions turning mountains of data into nuggets as well as reducing irrelevant information.

Feature selection was considered to be a process that selects a subset of the original features; it reduces the number of features, removes irrelevant, redundant, or noisy data, and has immediate implications to applications: speedup of the data mining algorithms and improvements in the mining performances, such as predictive accuracy and result comprehensibility. There is a vast body of available feature selection methods. In (Jain and Zongker 1997), feature selection algorithms were categorised into two groups, statistical pattern recognition (SPR) techniques and those using artificial neural networks (ANN), as shown in Fig. 3.1. The techniques of the SPR category find an optimal set of features, which the classifiers are trained on later.

The SPR category was further split into Deterministic methods (Pudil, Novovičová et al. 1994) and Stochastic methods (Siedlecki and Sklansky 1989). Deterministic methods must decide the starting point, which in turn influences the feature search direction. There are two approaches: forward methods that start with the empty set and add features; and backward methods that start with the full set and delete features. Stochastic methods start with a randomly selected subset of features and follow a sequential search or generate the next subset in a completely random manner. On the other hand, ANN methods, such as Node Pruning (Kearns and Mansour 1998), simultaneously develop both the optimal feature set and the optimal classifier. The node-pruning-based feature selection methodology first trains a network and, then, removes the least salient node (input or hidden). The reduced network is trained again, followed by removal of the least salient node. This procedure is repeated until the desired trade-off between the classification error and the size of the network is achieved.

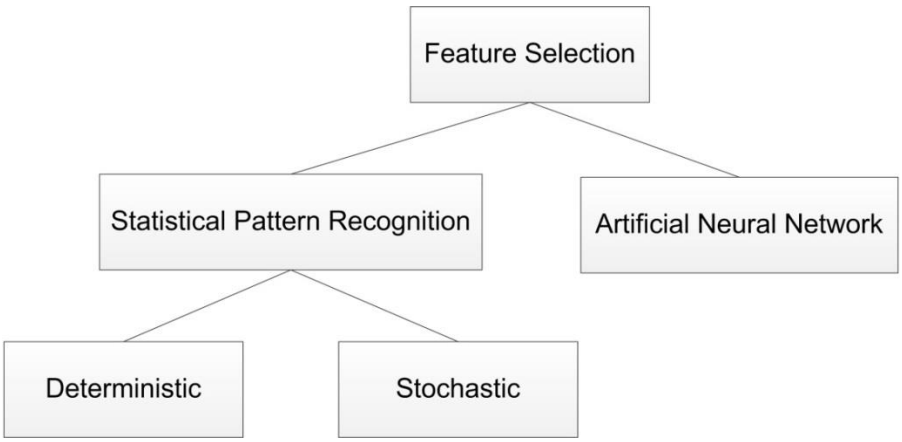


Fig. 3.1 A taxonomy of feature selection algorithms.

In (Liu and Yu 2005) (Liu, Dougherty et al. 2005), the general procedure of feature selection was described as having four key steps; feature subset generation, subset evaluation, stopping criteria, and result validation, as shown in Fig. 3.2.

The feature subset was generated by a process of heuristic search, with each state in the search space specifying a candidate subset for evaluation. The nature of this process was in the feature subset search strategies. For a data set with N features, there are 2^N candidate subsets. This search space was exponentially prohibitive for an exhaustive search with even a moderate N.

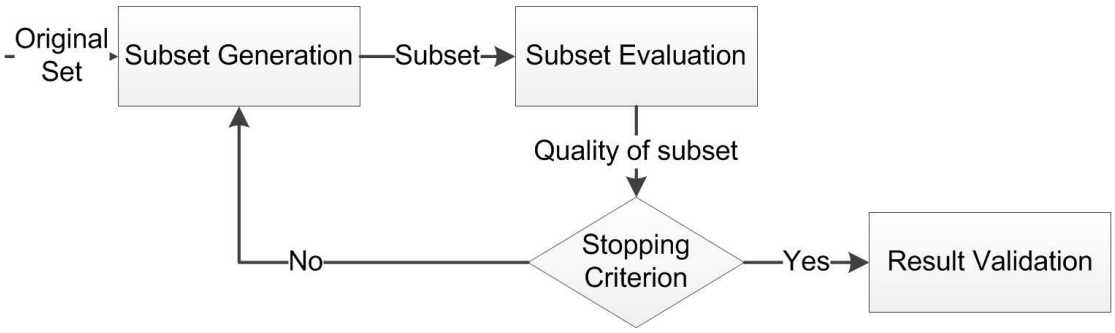


Fig. 3.2 Four key steps of the feature selection procedure.

There were three common search strategies: the complete search, the sequential search, and the random search. A complete search, such as branch and bound (Narendra and Fukunaga 1977) (Chen 2003) and beam search (Gupta, Doermann et al. 2002), could find the optimal subset of features much more quickly than an exhaustive search according to the evaluation criterion used. One drawback was that these procedures usually require the feature selection criterion function to be monotonous. A sequential search constituted a sequential forward search, a sequential backward search, and bidirectional selection. The sequential search methods were usually fast in performance but gave up completeness and thus risked losing optimal subsets. A random search generated subsets in a random manner. It helped to escape local optima in the search space, but the use of randomness risked missing the most optimal subset.

The second step was subset evaluation, in which the subset generated in the first step must be evaluated to assess optimality by an evaluation criterion. This step was the core of the general feature selection procedure. The generated subset in the first step was the input for this step, the consequences of which are displayed in the third and fourth step. The evaluation criteria included a filter model (Dash, Choi et al. 2002) (Hall 2000) (Yu and Liu 2003) and a wrapper model (Dy and Brodley 2000) (Kim, Street et al. 2000). The filter model relied on the general characteristics of the data to evaluate the selected feature subsets without involving any mining algorithm. Some popular independent criteria were the distance measure, which attempts to find the features that can separate the classes as much as possible, an information measure, which determines the optimal features according to the information gain of the features, a correlation measure, which measures the similarity between the features,

and a consistency measure, which attempts to find the minimum number of features that separate classes as consistently as the full set of features can. The wrapper model required a predetermined mining algorithm in the feature selection and determined the optimality by applying the mining algorithm on the selected features. It was argued that, compared to the filter model, the wrapper model was remarkably universal and simple. Filter models selected subsets of features as a pre-processing step of the whole task, independent of the chosen predictor. At the same time, wrapper models utilised the learning machine of interest as a black box to score subsets of features according to their predictive power.

Based on the criteria chosen in the second step, an evaluation was started. A stopping criterion in the third step determined when the generation and evaluation process should stop. Once the feature subset selection stopped, the selection was validated by monitoring the change in the task performance with the change in the features. These two steps were often heuristic and depended on specific tasks. For example, in clustering applications, we used a number of heuristic criteria, such as cluster compactness and scatter separability, for estimating the quality of the clustering results; in classification applications, we usually used the classification error rate as a performance indicator.

3.2 Related work about feature type selection

3.2.1 Feature type selection in general terms

Feature selections are usually utilised with machine learning techniques. Moreover, the central problem in machine learning is to identify a representative set of features from which to construct a recognition model for a specific task. Machine learning maps inputs to desired outputs. However, machine learning would not address the raw data directly because of the very large amount of information and its associated noise. Feature selection plays the role of providing a bridge between the raw data and the outputs. Pre-processing the data to obtain a smaller set of representative features while retaining the optimal salient characteristics of the data not only decreases the processing time but also leads to more compactness of the models learned and to better generalisation. When the outputs (which are also called labels) are unknown, unsupervised machine learning is appropriate. However, very large amounts of data

are required for unsupervised learning methods. In (Le, Ranzato et al. 2012), with a dataset of 10 million 200×100 pixel images downloaded from the Internet, the authors trained a face detector using model parallelism and asynchronous stochastic gradient descent (SGD) on a cluster with 1,000 machines for three days. The trained detector obtained 81.7% accuracy in recognising a test set that was composed of 37,000 image samples, which include more than 13,026 face images; this finding is a substantial improvement over the previous state-of-the-art methods. Nevertheless, with prior knowledge of the domain, we can achieve an even better result by labelling the training samples with a much lower computational capability and a much smaller training set.

In ITSs, the number of objects of interest is limited. As analysed in the first and second chapters, we usually focus on the roads, vehicles, license plates, drivers, traffic signs and pedestrians. Prior knowledge can help substantially in machine learning and can guide us in feature selection as well. We will focus on the feature selection and associated supervised machine learning.

As described in the general procedure of feature selection, most feature selection methods treat features as variables without considering the environment of the application and the domain knowledge. For example, both SIFT and HOG features are represented by vectors with multiple dimensions, to describe images. The performances would be significantly different even if we use the exactly identical feature selection procedure for different tasks. The art of data mining starts with the design of appropriate data representations. Building a feature representation is an opportunity to incorporate domain knowledge into the data and can be very application-specific (Guyon and Elisseeff 2003). Better performance is often achieved through using an appropriate feature type.

Therefore, it is necessary to integrate the original feature set generation into the whole procedure of feature selection. This thesis discusses and analyses the feature selection in ITSs with the consideration of an appropriate feature type. In view of our proposed feature selection scheme, feature types in ITSs are comprehensively reviewed in terms of low level, high level, and hybrid in the next section.

3.2.2 Feature type selection for an ITS

By considering the accuracy, robustness and processing time, different features are selected in the cognitive systems in an ITS. Via the analysis of feature characteristics, we group features that are applied in ITSs into three groups: low-level features, high-level features and hybrid features. Some cognitive systems, such as face identification and vehicle manufacture classification, must detect and recognise subtle details. Low-level features are always extracted from the objects, such as faces and vehicle fronts in these cases. However, the extraction and analysis of low-level features are usually much more time-consuming than for high-level features. Some cognitive systems strictly require an instant reaction. For this reason, high-level features are always applied, such as in vehicle counting and pedestrian counting systems, which cannot afford much processing time. Moreover, some studies have shown that a combination of low-level features and high-level features can take advantage of both of these types of features. In the following sections, we will present a comprehensive analysis of these three groups of features and review their applications in ITS cognitive systems.

3.2.2.1 Low-level features

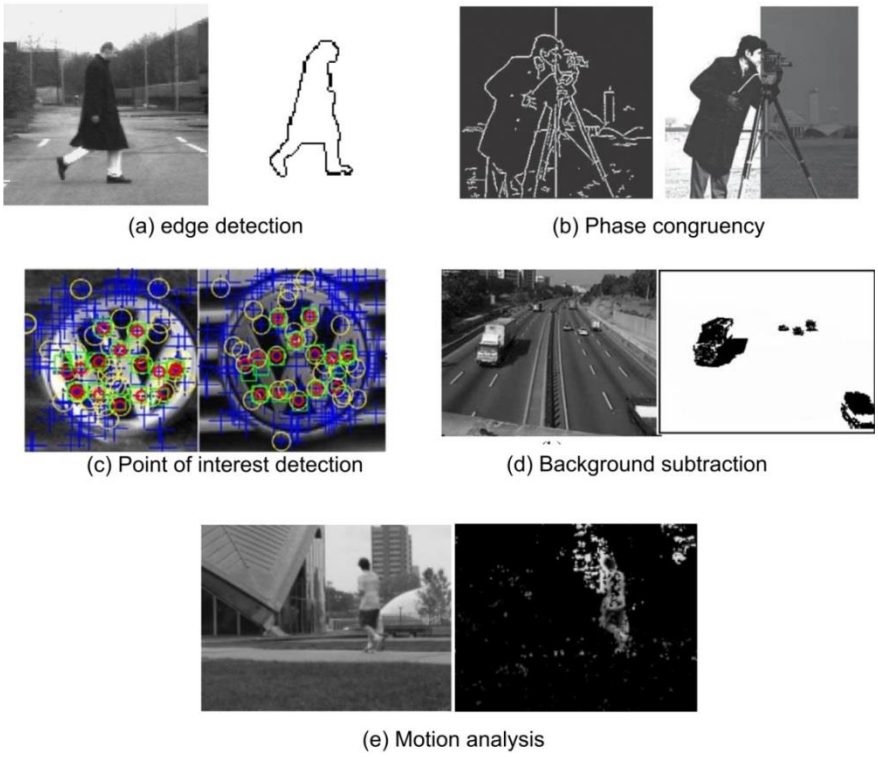


Fig. 3.3 Applications in intelligent transportation systems that use low-level features.

Low-level features were defined in (Nixon and Aguado 2012) to be those basic features that can be extracted from an image without any information about the spatial relationships. As such, thresholding was actually a form of low-level feature extraction, as a point operation. It was well known that we can recognise people from caricaturists' portraits, which is similar to edge detection in computer vision, as shown in Fig. 3.3 (a). There have been many edge detectors that are based on first-order or higher-level differentiation. An alternative form of edge detection is called phase congruency, which was an analysis in the frequency domain, as shown in Fig. 3.3 (b). Moreover, there is another set of low-level features called localised features, which extend from curvatures such as Harris corners to patches such as SIFT, as shown in Fig. 3.3 (c). All of these features were used on images or video frames. Moreover, dynamic analyses of objects in video were essential in ITS cognitive systems as well. Low-level feature analysis, such as Gaussian Mixture Models (GMMs) and optical flow estimation, were applied to describe the motion, as shown in Fig. 3.3 (d) and Fig. 3.3 (e). Usually, these features were not applied individually in ITS cognitive systems. A combination of these features was used for locating pedestrians, classifying vehicles, and detecting road lanes. Consider a vehicle moving on the road, for which the edges were the frame of the vehicle; the corners were the apices, and the flow was how the vehicle moves. All of these features can be collected together to detect and recognise the moving vehicle. The low-level features in an ITS are summarised in Table 3.1.

Table 3. 1 Summary of low-level features.

Low-level features	**Instances**
Edge	First-order edge (Sobel, Prewitt); Second-order edge (Laplacian, difference of Gaussian)
Phase Congruency	Frequency domain analysis
Localised feature	Harris corner, Scale Invariant Feature Transform(SIFT), Speeded Up Robust Features(SURF)
Motion analysis	Gaussian Mixture Model(GMM), optical flow

Pixel

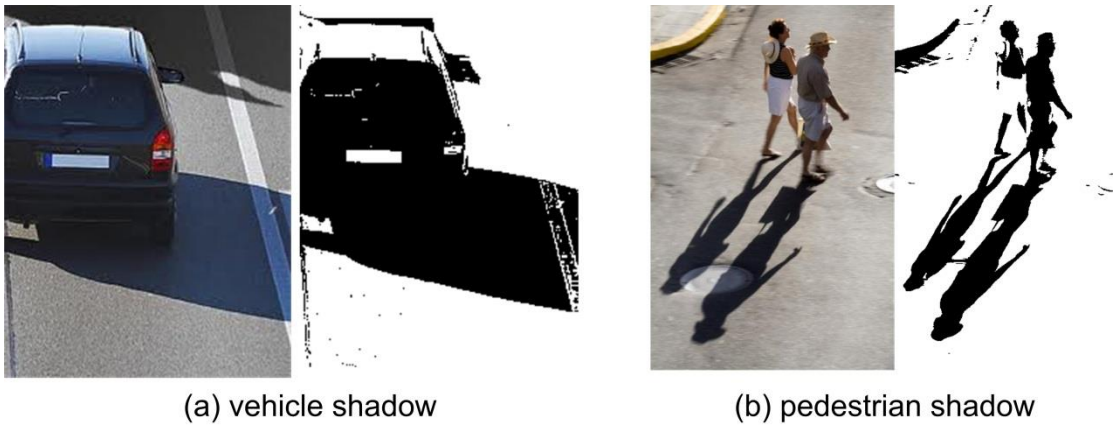


Fig. 3.4 Shadows of a vehicle and a pedestrian. Shadows always occur in intelligent transportation systems, where images are captured in outdoor environment.

Thresholding is selecting pixels that have a particular value or are within a specified range. It can be used to find objects within a picture if their brightness level (or range) is known. This implies that object's brightness must be known as well. There were two main forms: uniform and adaptive thresholding. Uniform thresholding clearly required knowledge of the intensity level, or the target features might not be selected in the thresholding process. As the environment was complex and lighting changes much in ITS, the intensities of objects in an image varied very often. Uniform thresholding was rarely applied in the cognitive systems in ITS. Adaptive thresholding, as a more advanced technique, sought a value for the threshold that separated an object from its background.

One successful application of adaptive thresholding in ITS was shadow removal. Shadow detection and removal was an important task in cognitive systems in ITS, where all images and videos were captured in an outdoor environment. Shadows cast by vehicles or pedestrians together with the objects formed distorted figures. Furthermore, separate objects can be connected through shadows, which always happened in pedestrian crowds or when vehicles are close to each other, as shown in Fig.3.4. Shadows confused cognitive systems in ITS a great deal. In (Wang, Chung et al. 2004), an adaptive thresholding-based method was presented for detecting and removing shadows from foreground vehicles. The authors assumed that foreground

vehicles had been extracted from the background. Adaptive thresholding was applied to firstly confirm the existence of shadows and remove them if confirmation was made. To decide whether there were shadows presenting in an image, two values, brightness and energy values, were calculated for each image pixel. These values represented the intensity relationships between image pixels around the pixel under consideration. With the assumption that objects were usually brighter than shadows in a scene, authors indicated a large brightness signified a high possibility that shadow existed. Having confirmed shadows exist in a figure, authors determined the shadow boundary and direction based on Otsu's method (Otsu 1975) and analysis of pixel intensity with some predefined conditions. In (Pai, Tyan et al. 2004), pedestrians' shadows were detected by checking the status of the hue, saturation, and intensity between the foreground image and background image with three thresholds. These thresholds were all determined by the condition of illumination. Adaptive thresholding was also applied in vehicle detection. In (Hsieh, Yu et al. 2006), adaptive thresholding was simply utilised to segment vehicle in a traffic surveillance system with static camera. Vehicles were extracted according to the difference images between video frame and background and a predefined threshold, which was chosen as the average of the difference image.

From the above analysis, we can see thresholding methods always required prior knowledge regarding the background image and were very sensitive to the illumination variance. This limited its application in ITS, where backgrounds were usually complex and light changed a lot. Therefore, the thresholding method was always applied in the pre-processing step, such as rough pedestrian or vehicle segmentation at the very beginning of cognitive systems. Besides, thresholding was often used to select the brightest points, following application of an edge-detection operator.

Edge-based feature

Unlike the thresholding method, edge feature is insensitive to change in the overall illumination level. Essentially, the boundary of an object is a step change in the intensity levels. The edge is at the position of the step change. Many edge detectors were proposed based on first-order or second-order differentiations. Based on a different convolution template, edge detectors, such as Prewitt, Sobel, Laplacian,

Marr-Hildreth, Canny and Spacek (Gonzalez and Richard 2002), had capabilities to deal with different problems. The selection of an edge operator for a particular application in ITS depends on the application itself.

The most intuitive application of edge detection in ITS was lane detection. There were two situations of lane detection-cognitive system between the surveillance camera and lane and cognitive system between vehicle and lane. Lane detection in the former cognitive system was always an aid for vehicle detection. In (Kluge 1994), road curvature were tangential to the camera. A simple one-dimensional operator was used to detect edges horizontally. Edge orientation was estimated by matching edge points across several adjacent rows and fitting a line to the points. The estimation was achieved through the use of global constraints on the individual lane-boundary shapes derived from an explicit model of road structure in the world. The lane detection and tracking in later cognitive systems played an important role in intelligent vehicles. In (Paetzold and Franke 2000), road topography was recognised for vision-based driver assistance in urban traffic. Through global edge detection, a polygonal edge image was obtained. Lane structures such as markings, curbs, crosswalks and stop lines were detected in this global edge image. For an update of the known road situation it was not required to scan the entire image for road structures. Fast tracking was achieved through a local boundary model with some geometric assumptions.

Besides lane detection, edge detectors were also utilised in traffic sign detection, which was mainly interesting for driver assistance systems. In (Soetedjo and Yamada 2005), a technique for detecting circular traffic signs used edge detection. The common methods usually extracted all edge points of objects. However, the authors detected edge points of the ellipse objects only. In their method, edge points of an ellipse were classified into five classes. Through the analysis of connection between edge points, only the points that belonged to these five classes were extracted. The noisy points or the points that did not lie on the edges of the ellipse were discarded or reduced. Regarding traffic sign detection, many other edge-based methods were proposed. Unlike the above one, most of them detected signs through revealing the geometric relationship of these edges, which combined into high-level features. We will review them in the next section.

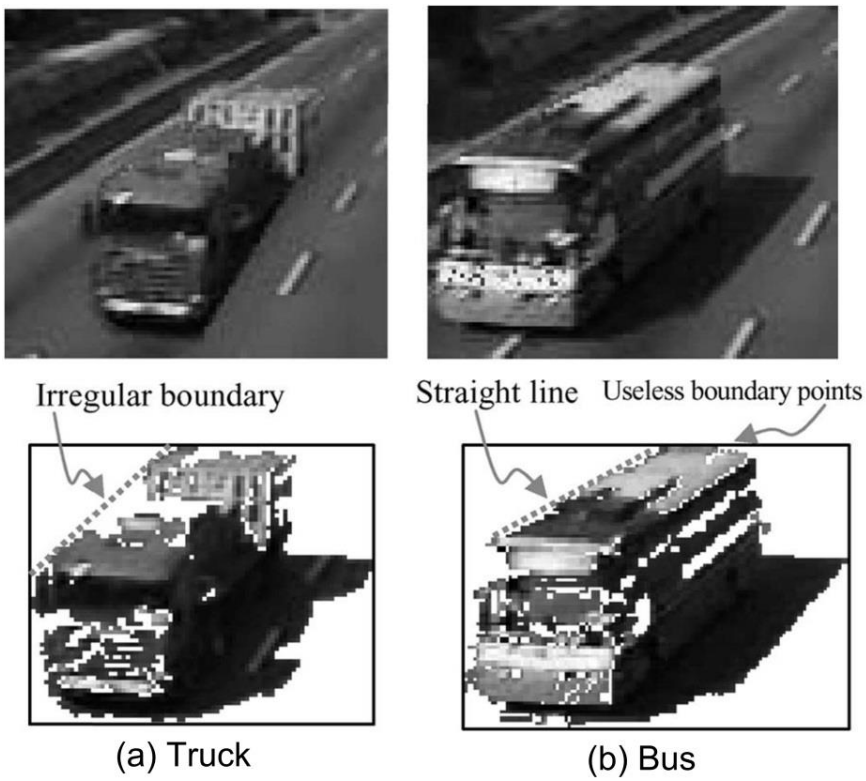


Fig. 3.5 Edges based features discriminate truck from bus, which have similar shape

Different views of a vehicle, especially rear/frontal views, contained many horizontal and vertical structures, such as vehicle top, rear-window, bumper, *etc.*. Using constellations of vehicles and horizontal edges has shown to be a strong cue for vehicle detection and recognition. In (Hsieh, Yu et al. 2006), an edge-based feature named by authors as a "linearity" feature was introduced to discriminate trucks from buses. As shown in the Fig.3.5, the truck and bus had similar sizes and speeds but different up-slanted edges, which was obtained by tracing all the edge pixels. When we scanned edge pixels, only edge pixels further from the vehicle base were recorded as candidates of "linearity" edges. After evaluating the regularity of candidature edges, some outliers were filtered out. In order to localise left and right positions of a vehicle, (Matthews, An et al. 1996) found strong vertical edges using an edge detector. They computed the vertical profile of the edge image by summing the pixels in each column followed by smoothing using a triangular filter. By finding the local maximum peaks of the vertical profile, they claimed that they could find the left and right position of a vehicle. (Betke, Haritaoglu et al. 2000) utilised edge information to detect distant cars by a coarse-to-fine search method looking for rectangular objects.

A refined search was activated only for small regions of the image, suggested by the coarse search. The coarse search looked through the whole edge maps for prominent edges, such as long uninterrupted edges. Whenever such edges were found, the refined search process was started in that region. Assuming the lanes had been detected, (Bucher, Curio et al. 2003) localised vehicles by scanning each lane starting from the bottom to a certain vertical position based on a predefined maximum distance in the real world. Potential candidates were obtained if a strong horizontal segment delimited by the lane borders had been found.

Edge detection was also utilised for detecting license plates, which contained many edge structures such as license plate borders and characters. License plate recognition was one of the most important cognitive systems in ITS. In order to recognise a license plate efficiently, however, the location of the license plate, in most cases, must be detected accurately in the first place. In (Jia, Zhang et al. 2007), candidate regions were firstly obtained by mean shift with colour information. Using mean shift, the input colour vehicle images were segmented into many regions, where the pixels in a region shared the same colour and different regions were represented by different colours. Considering that license plate regions generally had higher variance in their pixels' value because of the presence of characters, an important feature to describe license plate region was local variance. Edge density was then extracted to differentiate the license plate regions from others. With this consideration, the authors (Peng, Xu et al. 2011) went further. After applying a vertical edge detector to the input images, dense vertical strokes were always observed inside the license plate region even from different views. Line segment features were introduced in terms of density, directionality and regularity to characterise these strokes. License plate regions were always with high density of line segments. Rather than disorderly, line segments inside license plate region tended to be approximately vertical. Directionality was utilised to describe this pattern. Finally, regularity was introduced to measure the regular repetition of line segments inside license plate.

Many edge-based methods were also proposed for detecting pedestrians, which was another main road user besides vehicles. These methods mainly explored shape-related information of pedestrians using edge detection as a tool. The methods and applications will be discussed in the section of high-level features.

Phase congruency

(a) Modified license plate image (b) Edges detected by Canny operator (c) Edges detected by phase congruency

Fig. 3.6 Comparing Canny detector with phase congruency on modified license plate image. Phase congruency based edge detector shows better robustness than the Canny method.

As the aforementioned analysis suggests, thresholding methods generally followed edge detection to select the strongest edges, such as the Canny algorithm. However, the selection of a threshold was often inadequate for all the regions in an image since there were many changes in local illumination. As shown in Fig. 3.6 (a), the contrast changed in the two halves of the license plate image. The edges of characters were barely detected in the image of Canny edges, as shown in Fig 3.6 (b). The absence of substantial contrast change was due to the parameter settings used in the Canny operator. These can be changed, but if the contrast was to change again, then parameters would need to be re-optimised for the new arrangement. On the other hand, as shown in the Fig.3.6 (c), edges can be clearly detected by phase congruency. Phase congruency (Kovesi 2000) was derived by frequency domain considerations operating on the considerations of phase. Based on the Fourier transform analysis, any function was made up from the controlled addition of sine-waves of differing frequencies. We can determine edge points when changes happened at the same time. The advantage of congruency detection was that it was invariant with local contrast: the sine-waves still added up as the changes were still the same place, even if the magnitude of the step edge was much smaller. Phase congruency was also applied to detect the vehicle's logo in (Psyllos, Anagnostopoulos et al. 2011). Phase congruency was implemented to assess the existence of significant features. Values of phase congruency varied from a minimum of zero indicating no significance to 1 indicating a very significant feature. Through observing the phase congruency curve that corresponded to logos, radiator grille and headlights, the most important part of the "image signature" was the central region of the vehicle mask, where the manufacturer's logo usually appears.

Curvature

Edges are the low-level image features that are most obvious to human vision. They preserve significant features, so we can recognise what an image contains from its detected edges. However, we always need to recognise more subtle detail than the outline of an object. For example, we can hardly use the edge-based method to recognise drivers' face in the aforementioned driver monitoring system. We need to utilise more other features.

Intuitively, curvature was considered as the rate of change in edge direction. The rate of change characterised the points in a curve: points where the edge direction changed rapidly are corners, whereas points where there was little change in edge direction correspond to straight lines. The basic principle of curvature detection was to measure the angular change along the curve's path. Moravec and Harris detectors (Harris and Stephens 1988) were two famous corner detection methods. They measured curvature by considering changes along a particular direction in the image. The operators computed the average change in image intensity when a window was shifted in several directions. When the intensity changed greatly, the point under consideration was selected as a corner. In order to recognise the vehicle type, Harris Corners were extracted from sections of car front images with a simple set of features including Sobel edges and Spectrum Phase (Petrovic and Cootes 2004). With these extracted features, a normalised sample of the vehicle front structure of each class was defined. The structure was expressed in a feature vector of pre-defined length that was representative of the vehicle identity. Finally, simple nearest-neighbour classification was used to determine the vehicle type associated with each vector. Ego-motion estimation was an important component in intelligent vehicles. A motion estimation system with visual input alone was proposed in (Nistér, Naroditsky et al. 2006). Without prior knowledge of the scene or the motion, point features were matched between pairs of frames and linked into image trajectories at video rate. The authors chose Harris corners, which has been improved to give stable detection under small- to moderate-image distortions (Schmid, Mohr et al. 2000).

Patch features

There are some constraints on corner features we have reviewed so far such as sensitivity to object size and viewing angle. Patch features were introduced to relieve some of these constraints to some extent. The object can be characterised by a

Fig. 3.7 Scale Invariant Feature Transform (SIFT) detected on example images (Image of Lena Soderberg used in many image experiments. It comprises 512*512 pixels, and was originally cropped from centrefold of November 1972 issue of Playboy magazine (Rosenberg 2001)).

collection of patches. This allowed for the inclusion of scale and viewing change: an object can be recognised irrespective of its apparent size, even when part of the object was obscured; the object can be recognised from different viewing angle as a patch-collection, which still appeared in a similar arrangement.

SIFT (Lowe 2004) was one of most successful patch features to resolve the practical problems of object recognition. SIFT methods selected salient features in a manner invariant to image scale and rotation and with partial invariance to change in illumination. Further, the formulation reduced the probability of poor extraction due to occlusion clutter and noise. Fig. 3.7 shows SIFT features detected on the "Lena" image, which was a very famous testing image in the field of computer vision. Each patch feature extracted by SIFT was described by sampling the magnitudes and orientations of the image gradient of the patch. As shown in Fig 3.7(b), each arrow

indicated the direction of SIFT point while the length of arrow shows its magnitude. An array of histograms, each with orientation bins, captured the rough spatial structure of the patch. Based on SIFT, many methods were proposed in terms of improving processing speed and accuracy of SIFT. In order to compress the description vector of the patch, PCA-SIFT was raised in (Ke and Sukthankar 2004). Instead of using SIFT's smoothed weighted histograms, the authors applied PCA to the normalised gradient patch. In addition, in order to boost the processing speed, rather than using the difference of Gaussians to determine local features, SURF (Bay, Ess et al. 2008) employed approximations to second-order edge detection at different scales to extract features. Traditional SIFT methods were fully invariant with respect to only four parameters that represent zoom, rotation, and translation, but worked poorly under affine transform. Affine-SIFT (ASIFT) (Morel and Yu 2009) simulated all image views obtainable by varying the two camera axis orientation parameters, namely the latitude and the longitude angles, left over by the SIFT method. This method permitted identification of features that have undergone very large affine distortions, which always happened in reality. Fig.3.8 illustrates the comparison of two matching results using SIFT and Affine-SIFT, respectively. The line indicated the matched features between rotated vehicle front image and the whole image captured by traffic surveillance camera. We obtained many more matched features using Affine-SIFT more than that using SIFT. However, we can see there are two faults indicated by red lines in Fig.3.8 (b). Actually, it is a dilemma in feature extraction. On one hand, we desire more features when the types of these features are suitable for the problem. On the other hand, we have to avoid "bad" features, which would deteriorate the learning process. Therefore, the dependency and redundancy of features needs to be analysed. This problem is tackled in our proposed feature selection scheme.

Because of their good performance, SIFT-based methods were applied widely in all the cognitive systems in ITS.

Fig. 3.8 Comparison of matched results with SIFT and Affine-SIFT features. Affine-SIFT based method obtained more matched points, however, brought some false matching as well.

In (Psyllos, Anagnostopoulos et al. 2011), the vehicle logo was recognised based on SIFT features matching. To enhance the recognition process, vehicle model logos with a merged set of SIFT features were employed instead of using a single image. All SIFT features were detected from a set of images and one image was selected as a reference. The SIFT features of the remaining images were transformed to the coordinate system of the reference image by calculating their homographies using RANSAC (Fischler and Bolles 1981). At the end, a logo database with fused features was formed.

In order to estimate the real-time distance from the front vehicle, authors in (Peng, Xu et al. 2012) proposed a driver assistance system. With a monocular camera mounted on the front of the vehicle, the vehicle rear was first detected. Because the system strictly required high processing speed, SURF extracted from the vehicle rear was utilised to register the video frame. After complementing the registration, the homography between two consecutive frames were calculated.

Rather than applying an Optical Character Recognition (OCR) system, the authors in (Hoferlin and Zimmermann 2009) made use of the Linear Discriminant Analysis (LDA) with SIFT features to distinguish between the road signs. After detecting the traffic signs, SIFT features were extracted from the candidate regions. For the recognition, Multi-Layer Perceptrons (MLP) were utilised. The input layer of this neural net consisted of 128 neurons reflecting the dimensionality of the SIFT feature vector. The output layer represented each traffic sign class by a neuron.

In (Besbes, Rogozan et al. 2010), SURF features with Support Vector Machines (SVMs) were used for pedestrian detection. Fast features extraction was assured by using a hierarchical codebook of SURF features. The codebook was built by SURF features clustering. Starting from each SURF as a separate cluster, the two most similar clusters were merged as long as the Euclidean distance between SURF descriptors was below a threshold. Sometimes pedestrian identification such as gender classification was required in ITS. Using SIFT-based features, a Bayesian classifier was trained to detect, localise, and classify faces in terms of gender and age (Toews and Arbel 2009). The experiments showed good performance of the recognition system even from arbitrary viewpoints and in the presence of occlusion.

Motion analysis

We have reviewed the main low-level features that we can extract from a single image and their applications in ITS. However, the input of cognitive systems in ITS is consecutive video frames. We always need to analyse the motion of objects such as walking pedestrians and moving vehicles. If we have two images obtained at different times, the simplest way in which we can detect motion is by image differencing. However, simply subtracting the background image from video frames can hardly obtain accurate foreground objects. This is because the background image is never static and change exists constantly with lighting. The Gaussian Mixture Model (GMM) modelled each pixel as a mixture of Gaussians to determine whether or not a pixel was part of the background. The GMM stores M separated normal distributions for each pixel, where M is typically between 3 and 5 depending on the complexity of the scene. A foreign object appearing in the scene will be represented by some additional components with low weights and high variances. This leads to the conclusion that background-foreground segmentation can be achieved by selecting the mixture of the components of the highest weight-to-variance ratios as a background model, and the remainder as foreground. GMM was widely used in the cognitive systems in ITS, especially when the camera is fixed and objects are moving such as in a pedestrian counting system (Peng, Xu et al. 2012) and a vehicle detection system (Wang, Zou et al. 2009).

Background subtraction can perceive the global motion only. In order to describe the way the points in an image actually move, methods measuring how a pixels' position

changes in each image frame were proposed. The displacement corresponds to the projection of movement of the objects in the scene and it was referred to as the optical flow, which was a measurement of a pixel's velocity. Optical flow can be found by looking for corresponding features in images. In (Haag and Nagel 1999), using the combination of edge detection and optical flow, a comprehensive system was proposed to track vehicles based on a 3D model. The optical flow approach exploited the entire object image to estimate orientation, speed and angular speed, while the edge element approach concentrated only on edge elements in the vicinity of modelled edge segments. Correspondingly, the edge element approach is more sensitive to model errors. On the other hand, the optical flow approach cannot be employed while the object to be tracked either was still or moved very slowly.

3.2.2.2 High-level features

High-level feature extraction concerns finding the shapes of objects. In high-level feature extraction, we generally seek invariance properties in such a way that the extraction result does not vary according to specified conditions. In other words, techniques should find shapes reliably and robustly regardless of the value of any parameter that can control the appearance of a shape. Collections of low-level features described in the last section were usually used. For example, a group of low-level features such as wavelet features and Histogram of Oriented Gradient (HOG) features can provide appearance descriptions of objects. Intuitively, we can also investigate the use of shape: template matching, active contours, and shape skeletons. Template matching was a model-based approach in which the shape was extracted by searching for the best correlation between a known model and the pixels in an image. It is possible to use an active contour such as using a plastic bag to find a shape: the plastic bag is placed outside the object, and then, by taking air out of the bag, making it smaller, the shape of the object is found when the bag stops shrinking. In addition to using an object's perimeter or area, its shape can also be described by its skeleton. Table 3.2 shows a summary of high-level features. Using these high-level features, pedestrians, vehicles, license plates, and traffic signs were recognised in the ITS.

Table 3. 2 Summary of high-level features.

High-level features	Instances
Collection of low-level features	Haar-like features, Gabor features, Histogram of Oriented Gradients (HOG)
Fixed template matching	2-D or 3-D template matching, Hough Transform (HT)
Deformable template matching	Parts based template matching, Active contour

Collection of low-level features

Low-level features can be grouped to give structure, shape and appearance of objects. Using the form of Haar wavelets, the Viola-Jones approach originally detected objects of interest in images (Viola and Jones 2001) which was later extended to be one of the most popular techniques for detecting human faces (Viola and Jones 2004).

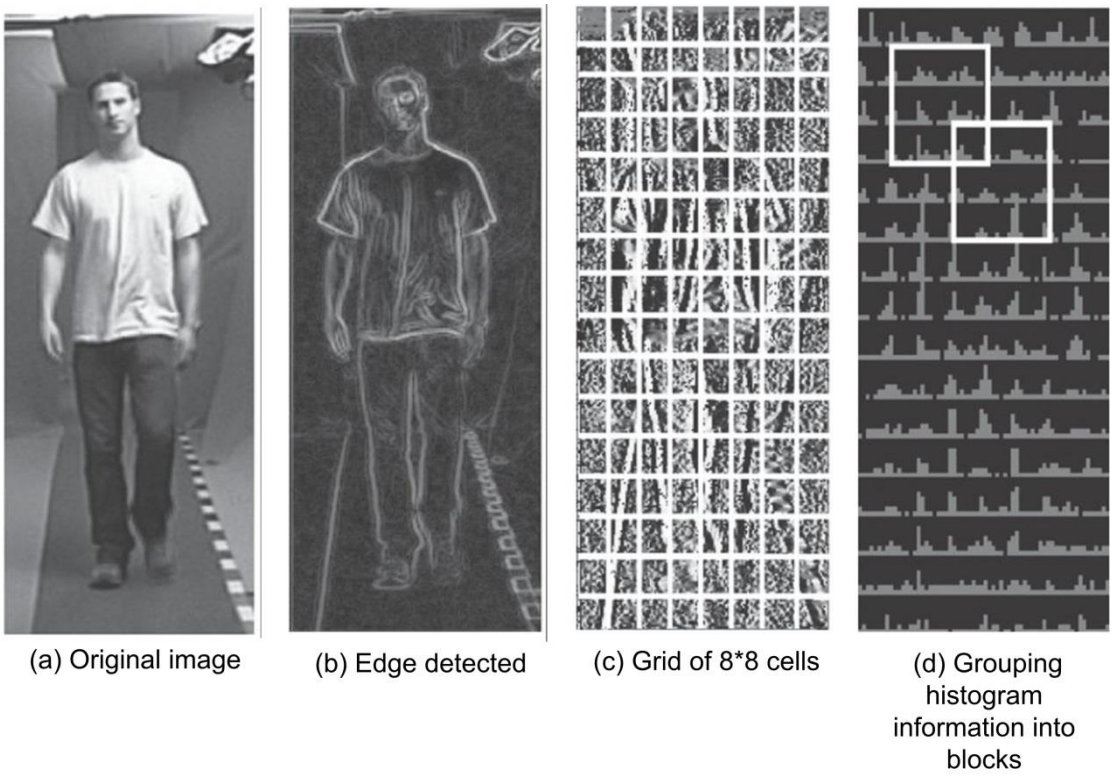


Fig. 3.9 Pedestrian detection using Histogram of Oriented Gradient (HOG).

Using the integral image approach, rectangles detected image features fast which can describe curved structures. For example, if we were to consider the face image, then the eyes were darker than the cheeks which were immediately below them, and the eyes were also darker than the bridge of the nose. A series of templates were matched to the image to encode differences in averaged intensities between different regions. The template was selected if it best matched the face at some position. In this way we can find the underlying shape of face. In (Monteiro, Peixoto et al. 2006) (Zhang and Zhang 2006), the authors presented a visual driver surveillance system to monitor the driver's head motion as well as the eye blink patterns. The driver's face was rapidly detected based on a boosted classifier of Haar-like features. The eye was then found using thresholding by assuming that the eyes are the darkest regions in a face. Similar frameworks were widely used in the detections of pedestrians (Cui, Liu et al. 2007) (Monteiro, Peixoto et al. 2006), vehicles (Haselhoff and Kummert 2009), traffic signs (Baró, Escalera et al. 2009) and license plates (Zhang, Jia et al. 2006).

As an alternative to Haar-like features, an edge-based method called Histogram of Oriented Gradients (HOG) (Dalal and Triggs 2005) showed its positive performance on pedestrian detection. The implementation of HOG was simply shown in Fig.3.9. The method first detected edges from the original image (Fig 3.9(a)) using an advanced first-order detector, as shown in Fig 3.9(b). A histogram was then created to store the vote decided from each pixel's edge magnitude and direction, as shown in Fig 3.9(c). The image was divided into small "cells", each cell accumulating a local 1-D histogram of gradient direction or edge orientations over the pixels of the cell. The combined histogram entries formed the shape of objects. In order to deal with the illumination variation, contrast normalisation is applied in blocks that have somewhat larger spatial regions than the "cell". The blocks were illustrated as white in Fig 3.9(d). The normalised descriptor was referred as HOG descriptor. HOG methods have been applied on objects other than pedestrians only in ITS. Authors in (Xie, Liu et al. 2009) utilized HOG features with Support Vector Machines (SVMs) to detect traffic signs. In the implementation of this paper, the description patch was 32×2 pixels size, the block was 32×32 pixels size, each cell was 8×8 pixels size and each cell contains 9 orientation bins. Each block consisting of 2×2 cells can obtain a 36-dimension feature vector. Then the whole patch which included 3×3 blocks could achieve a 324-dimension descriptor vector. The classifier was then trained by SVM with these

vectors as input. In (Negri, Clady et al. 2008), in order to detect vehicles by on-board camera, using a cascade of boosted classifiers, three features were compared: Haar-like features, HOGs, and their combination. The experiments showed the best performance was achieved by the combination of features. The authors claimed the reason was that HoG-based classifiers eliminated "easily" negative examples in the early layers of the cascade, while in the later layers, the Haar-like features based classifiers generated a fine decision boundary removing the negative examples near the vehicle model.

The Viola-Jones approach and HOG method described the entire objects such as faces and license plates by the collections of local texture features. There have been many approaches which applied other wavelets to detect objects by combinations of parts. Each part was a transform from a subset of wavelet coefficients to a discrete set of values. Such parts were designed to capture various combinations of locality in space, frequency, and orientation (Schneiderman and Kanade 2004). These approaches brought in great flexibility in the representation of the part. The method grouped images into sets, and each set is a part. For a vehicle, the parts included vehicle front/rear, vehicle sides, and vehicle top. In (Peng, Jin et al. 2013) (Peng, Jin et al. 2012), using a fixed traffic surveillance camera facing oncoming vehicles, the vehicle was localised and classified into either a truck, bus, sedan, or minivan. The aim was achieved by vehicle front detection and analysis based on eigenvalue analysis. For a human face, the parts included the eyes, nose, and mouth. In (Arbab-Zavar and Nixon 2011), a person's identity can be recognised by ear biometrics analysis. Based on an ear's centre points, a radial scan was taken using Gabor wavelets to capture the ear's features. The detail was preserved at the short wavelength and the larger structures were detected at longer wavelengths.

Fixed template

The most intuitive approach to investigate the use of shape was template-matching. To compute the best correlation between a known model and image, alternative ways from simple pixel comparison to maximum likelihood estimation were applied. The process of simple pixel comparison was similar to the process of template convolution. The template started from the origin of the image and matched with the image to record a count which indicated how well the template matched that region of the

image. After the template scanned through the whole image, the best matched part was obtained. However, this simple pixel comparison was impractical in reality, because the image was corrupted by noise and illumination changes. By considering each pixel in the image was corrupted by additive Gaussian noise, template-matching became a parameter estimation problem that maximised the probability that a point in the template matched the corresponding pixel in the image at the same position. The position where the template best matched the image was the estimated position of the template within the image. Due to the nature of the template matching methods, they have been applied in ITS for recognised rigid bodies such as vehicle and traffic signs only. In (Handmann, Kalinke et al. 2000), authors proposed a template based on the observation that rear/frontal view of a vehicle has a "U" shape. During matching, a vehicle was considered to be detected if the "U" shape was found. In (Bensrhair, Bertozzi et al. 2001), based on the assumption that a vehicle was generally symmetrical, characterised by a rectangular bounding box which satisfied specific aspect ratio constraints. First, the input image was checked for the presence of two corners representing the bottom of the bounding box using perspective and size constraints. Then the top part of the bounding box was detected, once again, by perspective and size constraints. Once the bounding box was detected successfully, authors claimed the vehicle was found in the image. In (Hsu and Huang 2001), a coarse-to-fine search was conducted for traffic sign detection. The coarse search used a rectangular block with fixed size, whereas the fine search used the triangular or circular template with variable size. Both of them used template-matching to find the road signs in the Region of Interest (ROI). The latter operated by counting the red image pixels inside the rectangular block or the template, and then selected the rectangular block with a red pixel ratio of at least 20% or the variable shape-template with a red pixel ratio of at least 75%.

Actually, these 2-D template-matching methods were very limited in practice as they were very sensitive to rotation and scale. In order to overcome these limitations, 3-D model matching methods were proposed. In (Hinz, Schlosser et al. 2003), an adaptive 3D-model was used to describe the vehicle. The prominent geometric features of cars were represented as wireframe in this model. The model further contained substructures like windshield, roof and hood. As a radiometric feature, colour constancy between hood colour and roof colour was included. Matching was carried

out by a top-down scheme and maximum likelihood estimation. However, constructing a comprehensive 3D-model of vehicle was hard. Due to these limitations of template matching, most papers in the literature did not report quantitative results but demonstrate performance through examples only.

(a) Circle traffic sign (b) Triangle traffic sign (c) Rectangle traffic sign

Fig. 3.10 Traffic signs are with basic shapes such as circle, triangle, and rectangle.

Clearly one drawback of the above template-matching method was that it required construction of the necessary models and parts if you are to detect other objects. This can be quite demanding. In fact, it can be less demanding to reformulate template-matching using basic shapes such as lines, circles, and ellipses. The Hough transform (HT) (Hough 1962) was a technique that can locate these basic shapes in images. The HT implementation defined a mapping from the image points into an accumulator space. The mapping required much less computational resources than template-matching. Because many objects of interest consisted of the aforementioned basic shapes in ITS, HT was often applied in ITS cognitive systems. In many countries, traffic signs were of basic shapes such as circles, rectangles, and triangles, as shown in Fig 3.10. HT were applied in (Damavandi and Mohammadi 2004), (Garc á-Garrido, Sotelo et al. 2005) to detect traffic signs quickly. License plates were another object of basic shape – a rectangle, in ITS. In (Duan, Duc et al. 2004), the authors optimised the speed and accuracy of license plate detection through applying the HT to contour images.

Deformable template

In the last section, the matching of methods with knowledge of a model of the target object in ITS were reviewed and discussed. The model was fixed in that it was flexible only in terms of the parameters that define the shape or the parameters that define a template's appearance. However, as per the limitations pointed out in the last

section, fixed template-matching methods (Handmann, Kalinke et al. 2000) (Bensrhair, Bertozzi et al. 2001) (Hsu and Huang 2001) were always very "loose" because it was usually impossible to model a shape with sufficient accuracy, or it provided a template of the target that was impossible to parameterise. In this case, deformable template-matching provided techniques that can evolve to the target shape or adapt their solution to the objects of interest. This section will review and discuss the technique that can be used to find flexible shapes in images and their applications in ITS.

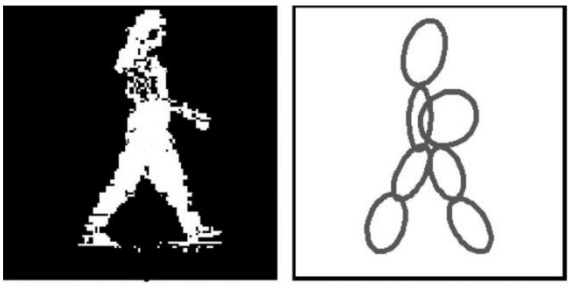


Fig. 3.11 Pedestrian detection using parts-based template matching. The body shape changes but the spatial relationship remains constant when the pedestrian moves.

Rather than matching an object with a fixed template, objects were represented as a collection of parts arranged in a deformable structure. We can imagine that the object was modelled as a network of parts which are connected by "springs", which controlled the spatial relationships between the parts and allowed these parts to move relative to each other. For example, it was hard to use a fixed template to characterise a human's face as there were too many different facial expressions. However, the parts of the face have fixed shapes and their spatial relationships were constrained, such as, the nose must be beneath and between the eyes. Deformable template matching was then a comparison between the match of part-template to the image and the interrelationships between the locations of the objects. As shown in Fig. 3.11, the shape of a body changed when a pedestrian moves. However, the spatial relationship among parts of the body did not change. As reviewed in the last section, fixed template-matching can be considered as a maximum-likelihood problem. Also, parts-based template-matching can be regarded as a parameter estimation problem. The parameters were estimated that can maximise the compromise between the positions of the parts and the deformation. This parameter estimation was proposed firstly in the

pictorial model introduced by Fischler and Elschlager (Fischler and Elschlager 1973). However, determining these parameters was computationally very challenging. In (Felzenszwalb and Huttenlocher 2005), a statistic framework was proposed to achieve the solution efficiently. Using a pictorial structure model, authors found faces and human bodies in an image. In (Huber, Kapuria et al. 2004), a parts-based method for classifying scenes of 3D vehicles into a set of pre-determined classes (pickup trucks, minivans, and sports cars). Parts were extracted from training vehicles and grouped into part-classes using a hierarchical clustering algorithm. In the training stage, each part-class was represented as a collection of semi-local shape features. A mapping from part-classes to vehicles classes was derived from the learned part-classes and known vehicles classes. In classifying, local shape features were computed. The object class was determined using the learned part-classes and the part-to-object mapping.

3.2.2.3 Hybrid features

From the above review of low-level and high-level features and their applications in an ITS, we can tell that most systems for object detection or classification utilised a combination of features rather than a single feature. Based on the objects of interest and the environment, various features were utilised. High-level features, such as fixed and deformable template matching, usually achieved fast processing speed and are understandable because they contain shape information on the objects. On the other hand, low-level features, such as edges, patch-based features, and motion analysis features, were computationally complex but deliver more subtle information.

To take advantage of both types of features, hybrid features that combine low-level features and high-level features were usually utilised in an ITS. There were two ways to apply hybrid features in an ITS. First, low-level or high-level features were employed on different steps in a complex system in the ITS. In (Peng, Jin et al. 2013), we used hybrid features to classify vehicles into different classes (bus, truck, sedan, and minibus). At the beginning, to extract the vehicle front, the spatial relationship between the license plate and vehicle front was utilised. Consequently, eigenvectors were extracted from each vehicle front for training the classifier. In addition to using hybrid features to achieve the goals of this step, hybrid features were also applied for a single recognition task. Haar-like features were a popular type of low-level feature

for object recognition. However, because they are very weak features, a very large number of them was usually required to obtain several strong classifiers. (Peng, Xu et al. 2011) proposed line segment features that were based on the appearance of patterns in a license plate. Through combining line segment features with Haar-like features, we dramatically reduced the demanded dimensions of the Haar-like features, and therefore, we saved much time in both the training and testing stages. Moreover, because most of the significant Haar-like features were retained while others were discarded, the recognition accuracy was boosted a substantial amount. Essentially, making suitable choices for the feature types is the critical step in building cognitive systems in an ITS. However, this step is not sufficient. A high dimensionality of features or "bad" features can consume processing time and bring down the accuracy of the cognitive system in an ITS. In fact, to retain the most significant features while discarding the "bad" ones, optimal feature subsets must be selected. Related work about feature subset selection and their application in ITSs will be reviewed in the next section.

3.3 Related work about feature subset selection

3.3.1 Feature subset selection in general terms

In a classic cognitive system, an object is typically described as an assignment of values to a set of features $X = (x_1, \dots, x_N)$ and one of l possible classes $c_1, \dots, c_N$ to the class label. The cognition task is to train a classifier that accurately predicts the labels of new incoming objects.

The learning of the classifier is inherently determined by the existing data sets. A data set contains a number of vectors, each of which corresponds to some occurrence of an event, and each vector is composed of a large number of features. In general, which features matter is not known. As a result, all of the types of information about events of interest are often gathered. Many of these features are redundant. Redundant features increase the size of the search space and make generalisation more difficult. The curse of redundant features has been a major obstacle in machine learning and data mining (Mitra, Murthy et al. 2002) (Robnik-Šikonja and Kononenko 2003) (Dash, Choi et al. 2002). Feature subset selection entails choosing the feature that maximally generates new knowledge about events and phenomena from existing data sets for the

classification or forecasting of future events. The feature subset selection approach was based on the principle of parsimony (Bell and Wang 2000). The model with the smallest possible number of parameters that adequately represents the data is favourable. However, the task of feature subset selection is difficult. Selecting the best feature subset has been proven to be an NP-complete problem (Hoos and Stützle 2004). In (Hua, Tembe et al. 2008), the authors summarised the challenges as follows: First, the features that appeared redundant singly could become highly redundant when taken with others. Second, there were many levels of multi-way redundancy in the feature space. Third, a high feature correlation did not imply an absence of feature complementarity. Fourth, high levels of multicollinearity increased the probability that a good predicator of the output signal will be found to be non-significant and be rejected from the model.

As discussed in 3.1.1, two broad categories of optimal feature subset selection have been proposed: the filter and the wrapper. In filter approaches, the features are scored and ranked based on certain statistical criteria, and the features with the highest ranking values are selected. Popular filter methods have included the t-test (Hua, Tembe et al. 2008), chi-square test (Jin, Xu et al. 2006), Whitney test (Liao, Li et al. 2007), mutual information (Peng, Long et al. 2005), Pearson correlation coefficients (Biesiada and Duch 2007) and Principal Component Analysis (PCA) (Rocchi, Chiari et al. 2004). Filter methods were fast but lack robustness against interactions among features. Moreover, it was not clear how to determine the cut-off point for ranking, to select only truly important features and exclude noise. For example, most PCA methods simply select the top N features with the largest eigenvalues. In the wrapper methods, the accuracy of the potential subsets in the predication is assessed. Wrapper methods are computationally more demanding than filter methods because they evaluated candidate feature subsets using a learning algorithm, and the learning algorithms usually involve iterative methods. Support Vector Machines (SVMs) (Mao 2004) were the most frequently used wrapper methods. However, SVMs have typically been criticised for their computational cost.

Recently, to take advantage of both the filter and wrapper methods, researchers have proposed hybrid methods (Tan, Fu et al. 2006) (Yan and Yuan 2004). Hybrid methods first applied filter methods to find a feature pool quickly, and then, applied wrapper methods to select the optimal subset from the selected feature pool.

3.3.2 Feature subset selection for an ITS

In the previous section, we reviewed feature subset selection in general. In this section, we will investigate the most popular feature subset selection methods for an ITS and our improvement under the guidance of the minimum redundancy scheme.

As shown in Fig. 2.2, most of the existing feature subset selection approaches are a combination of search strategies and evaluation criteria. K-means clustering (Hartigan and Wong 1979) and the Random Sample Consensus Algorithm (RANSAC) (Fischler and Bolles 1981) were two of the most classic methods. K-means partitioned the full set of features into k clusters, in which each observation belonged to the cluster that has the nearest mean. Given the candidate feature set $X = (x_1, \dots, x_N)$, k-means clustering aimed to partition the full set X into k sets $S = (s_1, \dots, s_k)$, to minimise the within-cluster sum of squares:

$$\min \sum_{i=1}^{k} \sum_{x_j \in S_i} \left\| x_j - \mu_i \right\|^2 \tag{3.1}$$

where μ_i is the mean of the features in S_i. In an ITS, K-means clustering was widely used in detection and classification problems, such as pedestrian detection (Zhang, Cai et al. 2007), license plate detection (Hsieh, Juan et al. 2005), and traffic abnormality detection (Münz, Li et al. 2007).

RANSAC was a resampling technique that generates candidate solutions by using the minimum number of features that are required to estimate the underlying model parameters. Unlike conventional sampling techniques that used as many of the features as possible to obtain a solution with the least squared error, RANSAC used the smallest set possible and proceeded to enlarge this set with consistent features. RANSAC has been widely used in various registration applications in ITSs, for example, in visual odometry for intelligent vehicles (Scaramuzza, Fraundorfer et al. 2009) (Nistér, Naroditsky et al. 2006) and in a vision system for pedestrian navigation (Jirawimut, Prakoonwit et al. 2003). RANSAC was very useful when addressing a large set of features. In chapter 7, in the development of the pose estimation system toward intelligent vehicles, we utilised RANSAC to remove redundant features to boost the processing speed as well as the registration accuracy.

PCA and BoF were two additional powerful methods for feature subset selection. Simply speaking, PCA sorted the features from the most important to the most trivial. PCA was mathematically defined as an orthogonal linear transformation that transformed the features to a new coordinate system in such a way that the greatest variance by any projection of the data lies in the first principal component, the second greatest variance in the second principal component, and so forth (Jolliffe 2005). However, PCA-based methods usually ignored redundancy analysis. In Chapter 5, through the implementation of a vehicle classification system, we will demonstrate the advance of our raised adaptive PCA method under the scheme of minimum redundancy in feature subset selection. BoF was the bag-of-words model applied in computer vision. In document classification, a bag of words was a sparse vector of occurrence counts of words. Similarly, "words" in images were the detected features. With the BoF model, after the feature subset selection, a feature representation set was generated that is called the "Codebook". However, the codebook can increase with more training data. A large codebook requires more processing time. In Chapter 8, we propose the Inference-BoF model to solve this problem. The related work about PCA and BoF in ITSs will be reviewed in detail in Chapters 5 and 8, respectively.

Chapter 4 Outline and Contributions

The aim of this thesis is to develop feature selection schemes for an ITS that can significantly improve the performance of cognitive systems in ITSs. We do not focus on the development of new feature primitives to be used in vision-based cognitive systems. Instead, we propose an efficient feature selection framework that is composed of a maximum dependency scheme and a minimum redundancy scheme.

With the scheme of maximum dependency, through fully considering the specific applications such as the task, performance, and environment of the implementation and the characteristics of different features including low-level, high-level, hybrid features, the most suitable types of features are selected. A comprehensive review of feature type selection and its applications in ITSs is conducted. This survey reveals the attributes of features of different types and their applicable scenarios. Once deciding on the suitable types of features, the optimal feature subset must be selected. The necessity of selecting an optimal feature subset is discussed. After reviewing the related work regarding feature subset selection, we propose the scheme of minimum redundancy to balance between the most compact feature subset and the best application performance.

Throughout this thesis, special emphasis is placed on having the performance improved by this feature selection scheme in terms of practical and challenging applications in ITSs. Based on the viewpoint of cognitive systems, an ITS is composed of a cognitive system that involves vehicles, pedestrians, infrastructure (for example, the surveillance camera and road), and drivers. Because ITSs is an applications-oriented field, it is meaningless to propose a methodical scheme without considering specific applications. In this thesis, we elaborate on the proposed feature selection scheme in three of the most popular and challenging cognitive systems in an ITS: vehicle surveillance, pedestrian surveillance, and intelligent vehicles. Finally, we propose the Inference BOF model under the guidance of feature selection schemes and demonstrate its advances through addressing an extremely challenging task, which is gender classification that is based on face recognition.

4.1 Feature selection for vehicle surveillance—vehicle classification using multiple eigenspaces

As the main road user, the vehicle is one of the most important components in ITS. To some extent, ITS is pushed by the increase of vehicles as ITS was introduced as a solution to the problem of traffic congestion in the towards the end of the 1990s. We first apply the proposed feature selection scheme in vehicle surveillance system in Chapter 5.

Vehicle type classification became an attractive topic due to its importance in ITS. Traditional vehicle type classifications were computationally expensive and have not been successful under conditions with variable illumination, occlusions, shadows and image rotations. In order to tackle these problems we propose a robust vehicle type classification method based on an adaptive PCA method. The function of PCA is to build an eigenspace, which consists of eigenvectors encoding important information regarding the object of interest. Specifically, traditional PCA methods selected a certain percentage of the top eigenvectors without considering the specific classification task. The most notable weakness of the traditional PCA method was ignoring the analysis of feature redundancy. In this chapter, we propose an advanced method that builds multiple eigenspaces with full consideration of different classes to deal with redundant and irrelevant eigenvectors. At first, in the training stage, the eigenspace is built for each class after selecting eigenvectors via a genetic algorithm. Then in the classifying stage, the front view of a vehicle is extracted automatically by examining the front width and the location of the license plate. Next, the vehicle is categorised into one of four classes (truck, bus, minivan and sedan) after projecting it into constructed multiple eigenspaces. Lastly, the comparison with current popular methods demonstrates the performance of our proposed method.

4.2 Feature selection for pedestrian surveillance—pedestrian counting using hybrid features

The pedestrian is another main road user. Unlike vehicles, pedestrians have different attributes such as irregular shapes and frequent occlusions. Reliable pedestrian detection and counting is an important and challenging topic in visual surveillance. In recent years, many impressive approaches were proposed in this field, but these

solutions have various restrictions such as unaffordable computational complexity, occlusions, complicated scenario and changing camera angles. Chapter 6 aims to propose a fast and stable method to detect and count pedestrians in both simple and complicated scenarios by taking advantage of our proposed feature selection scheme. First, post-processing steps are performed on foreground segmentation results to separate moving blobs, which includes pedestrians or other moving objects such as leaves and animals. Second, 9 features are extracted from each blob. Based on these extracted features, a classifier is trained using SVM. Via this method, the number of pedestrians in the scene is obtained as the sum of pedestrian numbers in all blobs. Finally, a novel analysis method with traditional tracking is proposed to optimise the estimation of pedestrian numbers. This method, which has been evaluated from 3 videos recorded from both simple and complicated scenarios, were implemented with parallel computing architecture for acceleration. Comparing these to the other two state-of-the-art methods, our method has been demonstrated to be effective. The speedup benefits from the parallel computing architecture implementations has also been shown.

4.3 Feature selection for intelligent vehicles—combining front vehicle tracking with 3D pose estimation

In Chapter 7, a feature selection scheme is applied in the development of intelligent vehicle system. In the vehicle surveillance system and pedestrian surveillance system, the camera was fixed. However, in an intelligent vehicle system, the background is not static as the camera is installed on the front of a moving vehicle. Moreover, processing time is strictly required in an intelligent vehicle system as system must react instantly when the vehicle is driving on the road.

Driver assistant systems enhance traffic safety and efficiency. The accurate 3D pose of a front vehicle can help a driver to make the right decision on the road. We propose a novel real-time system to estimate the 3D pose of the front vehicle. This system consists of two parallel threads: vehicle rear tracking and mapping. The vehicle rear is first identified in the video captured by an on-board camera, after license plate localisation and foreground extraction. The 3D pose estimation technique is then employed with respect to the extracted vehicle rear. Most current 3D pose estimation techniques need prior models or a stereo initialisation with user cooperation. It is

extremely difficult to obtain prior models due to the varying appearances of vehicles' rears. Moreover, it is unsafe to ask for drivers cooperation when a vehicle is running. In our system, two initial keyframes for stereo algorithms are automatically extracted by vehicle rear detection and tracking. Map points are defined as a collection of point features extracted from the vehicle's rear with their 3D information. These map points are inferences that relate the 2D features detected in following vehicles' rears with the 3D world. The map is extremely important because the relative 3D pose of the on-board camera to the front vehicle rear is then estimated through matching the map points with point features detected on the front vehicle rear. We will see the importance of our feature selection schemes in building and maintaining the map. We demonstrate the capabilities of our system by testing on real-time and synthesised videos. In order to make the experimental analysis visible, we demonstrate an estimated 3D pose through augmented reality, which needs accurate and real-time 3D pose estimation.

4.4 Feature selection with inference bag of features

In Chapter 8, we originally proposed an inference BoF method for selecting features efficiently. Current BoF methods construct a Visual Word Dictionary (VWD) from training images. More training data is desired for a higher classification rate. However, more training data increase the size of the VWD as well as the testing time. A fixed size of the VWD in the current methods guarantees the processing speed but would not address the available training data. Our method addresses this dilemma. We use three sets of images: training, inference and testing images. Using sparse coding, VWD is constructed from inference images, the amount of which is fixed. Posterior probabilities of visual words over classes are learned from training images in a Bayesian framework. In testing, the testing images are represented by visual words in VWD. The choices for representing the visual words determine the classification decision. We compare our method with two popular methods via addressing a challenging task, which is face recognition-based gender classification. The experiments show that our Inference BoF method can always achieve an optimal set of features.

4.5 Evaluation methodology

Performance evaluation of our feature selection scheme is a major aspect of this work, in terms of both the methodology and datasets that are used. Evaluation can be performed using a per-image measure (the detection context) or a per-window measure (the classification context). Per-image evaluation involves shifting a classifier through its location and scale across the whole test image. In per-window evaluation, the test data involves extraction and scaled bounding cropped from full test images.

Depending on the application context of the systems to be evaluated, we use both evaluation methods. Per-image evaluation is used to evaluate sliding-window classifiers, for example, the license plate detection that is utilised in both of the vehicle surveillance systems in Chapter 5 and in the intelligent vehicle in Chapter 7. On the other hand, for the evaluation of integrated systems that include hypothesis generation such as face recognition in Chapter 8, per-image evaluation is the only viable choice. Most real-world systems, however, integrate several modules that do not follow a brute-force sliding-window detection scheme but instead use a pre-processing step to determine the initial pedestrian location hypotheses for both the enhanced performance and the computational efficiency. For example, this task is performed by background subtraction in a pedestrian surveillance system in Chapter 6.

For the evaluation data, we chose public and real-world data, such as pedestrian counting data, face recognition data, and license plate recognition data. In Chapter 5, the applied cognitive system recognises a vehicle by using images of the vehicle from a front view. We could not find a suitable public database for this evaluation. Therefore, we recorded 4600 images of vehicle front views on a highway and made these images publicly available for benchmarking and to stimulate further research.

4.6 Publications

This thesis has led to a number of publications that are listed in Appendix Publications. Note that the corresponding publications have been included in the discussion of related work in Chapter 3.

Chapter 5 Feature Selection for Vehicle Type Classification

The content in this chapter has been published in (Peng, Jin et al. 2013) (Peng, Jin et al. 2012) (Peng, Xu et al. 2011).

The content in this chapter has been presented in the journal manuscript "Vehicle Type Classification Using Multiple Eigenspaces", which has been submitted to IEEE Transaction on Intelligent Transportation System.

5.1 Background

As an important cognitive system in ITS, vehicle type classification is widely used in vehicle authentication systems, parking optimisation, autonomous navigation, law enforcement, *etc*. However, at most places, vehicle types are still classified by humans due to the complexity of this work. Several disturbances on road such as occlusions, changing light conditions, shadows, rotations, *etc*. have affected the accuracy of vehicle type classification. Vehicle type classification is always a complicated system involving more than one step: foreground segmentation, vehicle detection, feature extraction and classification. The robustness and effectiveness of the whole system depends on the performance of each step as well as the compliance of these steps. In this chapter, based on the methodology of feature selection, a practical vehicle type classification system is proposed to address these problems.

Comparing this with the large amount of literature on automatic vehicle detection and tracking (Sivaraman and Trivedi 2010) (Wang, Cui et al. 2012) (Zhang, Wu et al. 2012) (Sun, Bebis et al. 2006), limited work has been reported in the field of vehicle classification. A summary of related work is shown in Fig.5.1. Most existing methods for classifying vehicle types are visual-based or non-visual sensor-based. Non-visual sensor-based methods employ buried inductive loops (Ki and Baik 2006) or radar (Urazghildiiev, Ragnarsson et al. 2007) to measure the sizes and lengths of vehicles. The main drawback of such systems is the complicated setup and high maintenance cost. On the other hand, large-scale deployment of traffic surveillance cameras and rapid development of image processing techniques have made visual-based vehicle type classification more and more popular.

Visual-based vehicle type classification can be grouped into two categories: methods

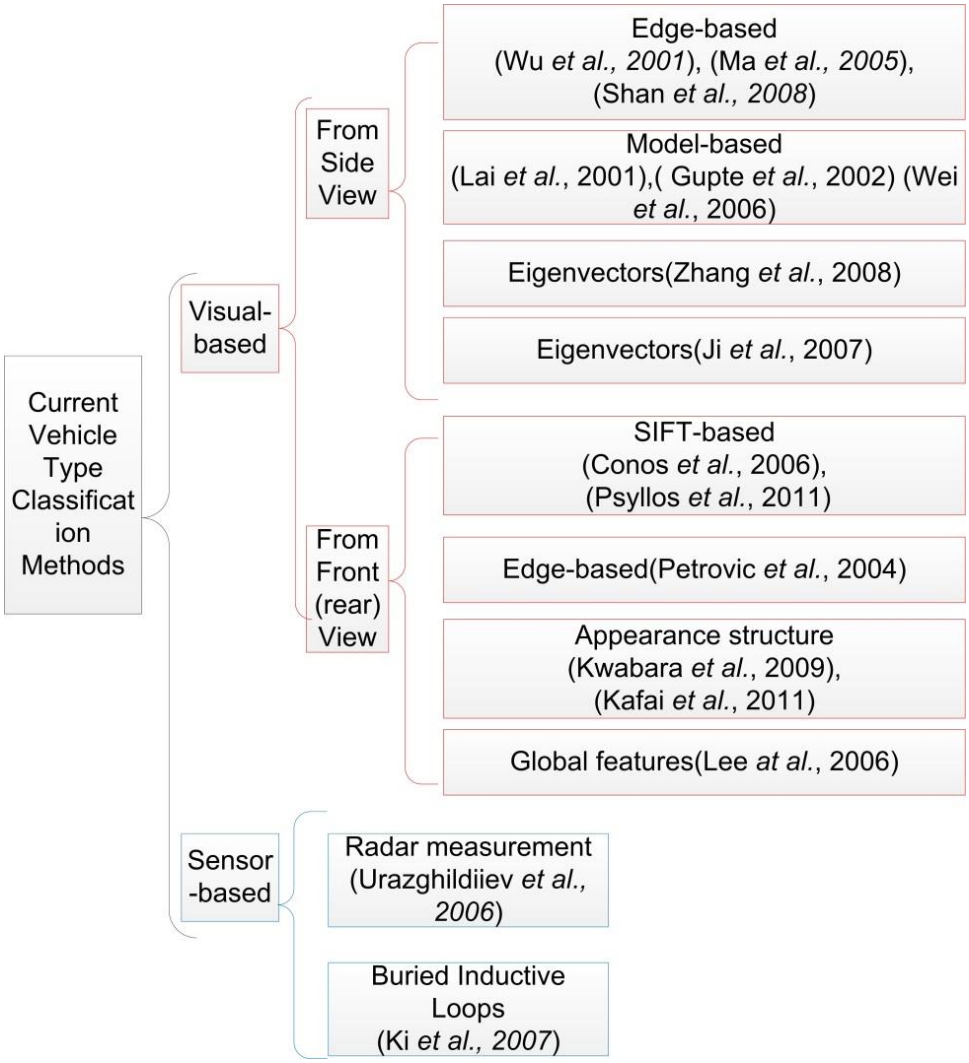


Fig. 5.1 Current vehicle type classification methods.

using the vehicle's side view and methods using the vehicle's front/rear view. For the side view examples, edge and model based methods have been widely used. Edge-based approaches include parameterised edge models (Wei, Zhang et al. 2001), edge point groups (Ma and Grimson 2005) and weighted edge matching (Shan, Sawhney et al. 2005). Wu *et al.* (Wei, Zhang et al. 2001) proposed a parameterised edge model to describe the topological structure of vehicle, and then fed models into a multi-layer perceptron networks-based classifier. Although the classification rate was satisfactory with high-quality images, the authors stated that their method was limited when performing on low quality images. Ma *et al.* (Ma and Grimson 2005) associated edge points with (Scale-Invariant Feature Transform) SIFT based descriptors to guarantee the repeatability and sufficient discriminability of their proposed feature. Though they

reported impressive results, the method was time-consuming and fell short when views changed. Shan *et al.* (Shan, Sawhney et al. 2005) presented an edge-based method to match vehicle images captured from two non-overlapping cameras. The vehicle matching problem was considered as a same-different classification problem. The similarity of two vehicles from two cameras was calculated. After automatically collecting representative samples from same-different classes by a weak classification algorithm, a more discriminative classifier based on Fisher's linear discriminates and Gibbs sampling was obtained. The main weakness of this method was that it required vehicle images captured from two different cameras. This camera arrangement was hardly used in current traffic surveillance systems.

Model-based approaches that used prior vehicle shape information had been investigated (Dubuisson Jolly, Lakshmanan et al. 1996) (Lai, Fung et al. 2001) (Gupte, Masoud et al. 2002) (Hsieh, Yu et al. 2006). Jolly *et al.* (Dubuisson Jolly, Lakshmanan et al. 1996) proposed five classes of deformable 2D models to segment a vehicle from background. Although this method too depended on camera view and image quality to practice with real world condition, it explored a new path for vehicle type classification. In order to generate 3D models of vehicles (Lai, Fung et al. 2001) (Gupte, Masoud et al. 2002) (Hsieh, Yu et al. 2006) , vehicle appearance parameters, such as length, width and height were recovered from 2D projections of vehicles under a calibrated camera model. In (Hsieh, Yu et al. 2006), besides traditional parameters recovery, a feature named "linearity" was proposed to discriminate trucks from buses, which had very similar appearances. However, the complexity of stereo algorithms made these methods time-consuming. Moreover, in order to recover 3D parameters of a vehicle, both the vehicle side view and frontal/rear views were needed. Therefore, a specific camera arrangement was required.

Inspired by the facial expression recognition, Zhang *et al.* (Zhang, Chen et al. 2006) extracted eigenvectors for type classification from side view images of vehicles using Principal Component Analysis (PCA). Ji *et al.* (Ji, Jin et al. 2007) classified vehicle types based on Gabor features bank. PCA and Gabor features bank were powerful methods to find a set of discriminative features that represent vehicles in a more compact and robust way. Traditional methods first built a feature space such as eigenspace in PCA method and Gabor features bank in Gabor feature method from training images. Then testing images were represented by the projection on the feature

space. The question of whether or not a uniform feature space can describe most discriminative features of images in different classes was usually ignored. Moreover, the method with the side view of vehicle easily failed for classification even when a slight camera view change occurs.

Some researchers' attention was attracted to the appearance structure of the vehicle's front/rear for each vehicle type. Based on the SIFT descriptor, vehicle make and model recognition from the frontal views was investigated in (Conos 2006) (Psyllos, Anagnostopoulos et al. 2011). (Conos 2006) fed SIFT descriptors extracted from frontal view image into a kNN classifier. (Psyllos, Anagnostopoulos et al. 2011) used a neural network classifier to recognise the logo, manufacturer and model of a vehicle. Such a method entirely depended on the logo detection; it would fail if the logo cannot be detected correctly. Unfortunately, logos were usually very small in the image.

Edge-based features sets were utilised in (Petrovic and Cootes 2004). From a vehicle frontal view, Petrovic *et al.* (Petrovic and Cootes 2004) extracted a set of features including Sobel edge, edge orientation, direct normalised gradients, locally normalised gradients, square mapped gradients, Harris corner, and spectrum phase. Based on these features, a simple nearest-neighbor classifier was applied. Although impressive recognition accuracy was achieved, calculating so many low-level features was time consuming.

Some researchers' attentions were attracted to the appearance structure of vehicle's front/rear for vehicle type. Kuwabara *et al.* (Kuwabara, Yano et al. 2009) proposed an appearance structure with Gaussian Mixture Model (GMM) to recognise a vehicle's type. The appearance model representing the rear view image of a vehicle includes the rear window, tail lights, and so on. However, the appearance model was established based on colour recognition, which made this method very sensitive to light conditions. Kafai *et al.* (Kafai and Bhanu 2012) described vehicle rear appearance using a feature vector representing a tail light, license plate, window, *etc.*, thereafter, processed this feature vector by a Hybrid Dynamic Bayesian Network to classify vehicle.

Moreover, global features were also used in vehicle type classification. Lee *et al.* (Lee 2006) extracted from frontal view of vehicles texture descriptors such as contrast,

homogeneity, entropy and momentum. Based on these features, a classifier was trained through a three-layered neural network.

In all aforementioned methods from frontal/rear view, accurate extraction of vehicle front/rear was a crucial step. Vehicle front/rear area was pre-defined according to the location of license plates (Lai, Fung et al. 2001) or logos (Gupte, Masoud et al. 2002). However, defining a vehicle's front/rear area in this way was not ideal. Pre-defined vehicle front/rear area would possible inaccurate. Incorrect vehicle/rear extraction lead to unsatisfactory classification rate. In our method, we addressed this problem and developed an automatic vehicle front extraction method by combining license plate detection, vehicle segmentation and shadow removal.

In this chapter, using the framework of feature selection, we proposed a more robust and more practical vehicle type classification system to address the aforementioned problems. At the beginning, we applied a quick and accurate AdaBoost method with the combination of Haar-like features and line segment features to localise the license plate (Peng, Xu et al. 2011). We used a background updating method to avoid the effect of lighting changes and obtained vehicle front width from an extracted binary mask. Thereafter, the vehicle frontal region was extracted by referring to the license plate's location and the vehicle's front width. After that, vehicle frontal images were projected into previously constructed multiple eigenspaces for type classification. To distinguish from building a uniform eigenspace by picking up a percentage of the top eigenvectors to represent all the target objects in traditional PCA method, we constructed multiple eigenspaces for different classes using an adaptive genetic algorithm (GA) to search the full set of eigenvectors with the goal of selecting a subset of eigenvectors encoding important information about each class image. This development enhances classification accuracy. In order to demonstrate the effectiveness of our method, our experiments evaluated the performance of each step as well as the compliance of these steps.

5.2 Related work about PCA

An object classification system using PCA involves two main steps:

1) extracting principal components from training images and representing both training and test images by extracted components, and

2) training a classifier based on the eigenvector-represented training images using different machine learning methods.

PCA intends to represent images in a lower dimensional space, which is usually called "eigenspace" consisting of eigenvectors. Eigenspace spans the principal components of the covariance matrix of the data. Eigenvectors is with a huge amount, which is total number of training images submitted one. We would like to use only eigenvectors that have a high separability power while ignoring or paying less attention to the rest. A typical strategy is picking a percentage of the top eigenvectors to represent the target object, independent of the classification task.

Several researchers (O'Toole, Abdi et al. 1993) (Abdi, Valentin et al. 1995) (Valentin and Abdi 1996) (Yambor, Draper et al. 2002) had found that different tasks made different demands in terms of the information that is needed to be processed, and that this information was not contained in the same range of eigenvectors. (Etemad and Chellappa 1997) found that the recognition information of eigenvectors did not decrease monotonically with their corresponding eigenvalues. Therefore, the common practice of choosing top eigenvectors was the not best method. The question of how to choose suitable eigenvectors became a concern. (Balci and Atalay 2002) selected eigenvectors that most contribute to classification using a neural network classifier. They also demonstrated that not all of the top eigenvectors contributed to correct classification and that some of them had been discarded by the network. In order to avoid the bias brought about by simply selecting top eigenvectors, some researchers utilised advanced methods to choose eigenvectors. (Sun, Bebis et al. 2004) showed that the Genetic Algorithm (GA) was powerful in selecting most important eigenvectors. The number and range of eigenvectors were decided by evaluation on a set of validation images. These strategies were still with limitations, because a uniform eigenspace containing the same set of eigenvectors cannot encode the specific characters of various classes, even though the set of eigenvectors were selected by some optimal methods. For example, the most discriminative eigenvectors for different class vehicles (truck, bus, minivan or sedan) should be different. Therefore, it is extremely necessary to build for each class a specific eigenspace, which can describe the corresponding class best.

In our method, multiple eigenspaces rather than a uniform eigenspace were built for different class images. For each class, our feature selection method discarded insignificant eigenvectors while keeping the ones encoding the most useful information for the current class. In the training stage, in order to train the classifier, each training image was projected into the corresponding eigenspace. In the classification stage, a testing image was projected into all eigenspaces. Fast KNN was then applied for classification determination.

5.3 Method overview

Our proposed method consisted of three sets of data: training images, validation images and test images; and two steps: training, and testing. As shown in Fig. 5.2(a), on training stage, image normalisation processing was first applied to the training images and validation images, both of which consisted of manually cropped vehicle front images with 4 class labels (truck, bus, minivan, and sedan) to compensate for light variations, noise, *etc.*. A comprehensive set of eigenvectors were generated from normalised training images. By evaluating 4 sets of normalised validation images, a genetic algorithm was carried out to build 4 eigenspaces for a truck, bus, minivan and sedan, respectively. Each set of eigenvectors encoded mostly important information about the corresponding vehicle type. Thereafter, each training image was represented by a vector of projection coefficients after projecting into its eigenspace.

The classification procedure was demonstrated in Fig. 5.2(b). In order to quickly and accurately localise the license plate, we proposed a license plate localisation method rested on Haar-like feature and line segment features. After implementing background subtraction and shadow removal to extract the vehicle body in images, the width of vehicle front was learned. According to obtained license plate location and vehicle front width, the vehicle front can be extracted correctly. By projecting the vehicle front into the 4 previously constructed eigenspaces, 4 vectors of projection coefficients were obtained to represent the vehicle front. The vehicle was classified into a specific type after KNN searching for its represented vector within vector representations of training images.

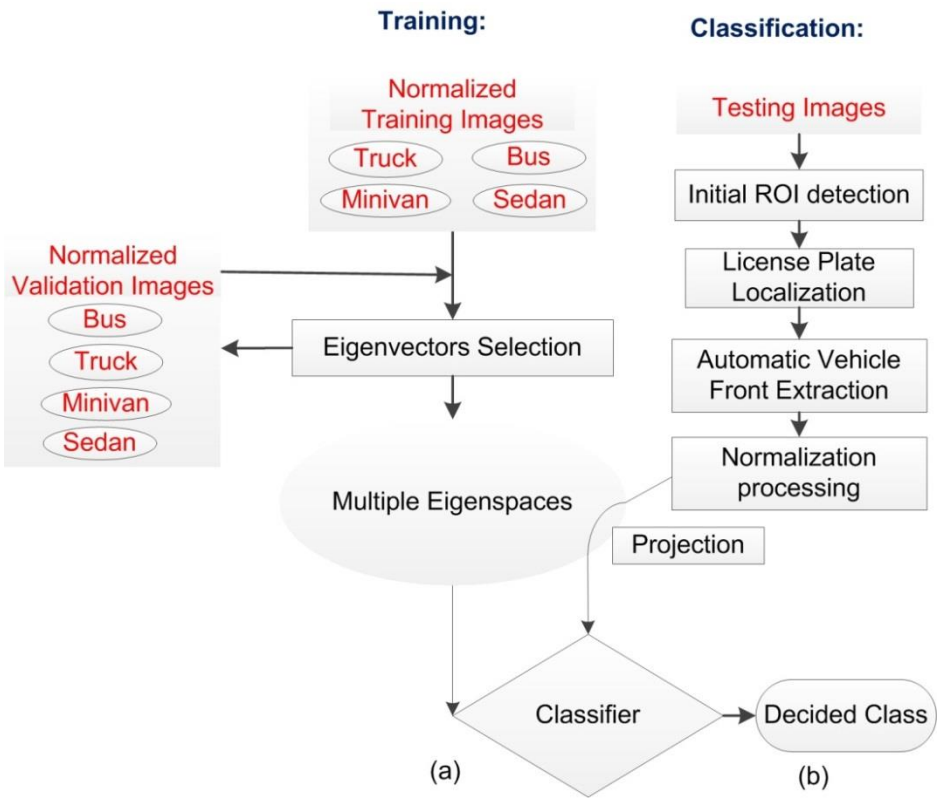


Fig. 5.2 The procedures of vehicle classification system: (a) Training stage: building multiple eigenspaces and training classifier, (b) Classification stage: extracting vehicle front, projecting on multiple eigenspaces and classifying.

5.4 Feature selection scheme for localising license plate and classifying vehicle type

As demonstrated in Fig. 5.2, the whole system of vehicle type classification is complex system consisting of multiple modules, among which license plate localisation and vehicle type classification are the two most core ones.

License plate localisation is the foundation of accurate extraction of vehicle front. The accuracy is the highest priority. According to the maximum dependency scheme, we propose Line Segment Feature (LSF) based on the special pattern of license plates. In order to speed up the process, we combine LSF with Haar-like features to generate a candidate set of features. Because Haar-like features are weak features, the generated candidate set contains huge amount of features. For achieving the optimal feature set, we utilised AdaBoost to produce strong features. This procedure becomes much faster than the traditional Haar-like—AdaBoost method with the introduction of LSF.

For vehicle type classification, we proposed an adaptive PCA method to select max-dependent and min-redundant features to present the vehicle front through building up multiple eigenspaces than the single eigenspace in traditional PCA method. We will detail the advantages of our method in Section 5.8.

5.5 License plate localisation

5.5.1 Combination of line segment features and Haar-like features based on maximum dependency

Fast and accurate license plate localisation was the crucial step in our system. We achieved this through a coarse-to-fine method. We firstly normalised the input image to reduce effect of various lighting condition and noise. We band-pass filtered the image and weighted the pixels using a Gaussian function centred on the image. The image was then normalised to have zero mean and unit standard deviation. The normalised image was shown as the top part of Fig. 5.3.

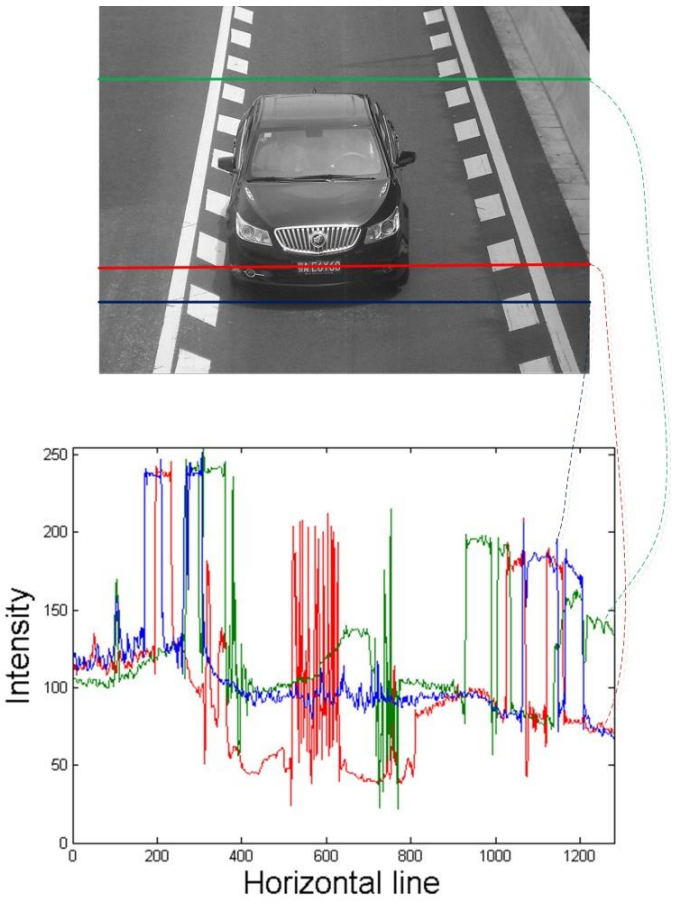


Fig. 5.3 ROI of license plate detection, roughly detect ROI based on histogram pattern of license plate.

As shown in Fig. 5.3, we found that the intensity histograms of potential license plate regions, as the red lines, fluctuate wildly when we showed every horizontal line of image as intensity histograms. The initial ROIs of license plates were detected roughly and quickly, according to this feature.

We further located license plates on ROIs based on the combination of line segment features and Haar-like features, which was proposed in our previous work (Peng, Xu et al. 2011). For LP detection, many previous researchers (Wang and Lee 2007) used AdaBoost in conjunction with Haar-like features. Compared to other low-level features, Haar-like features saved considerable computational cost since they were extracted from integral representations of an image rather than individual pixels. However, Haar-like features was very illumination-sensitive and the selected feature set was very large, which made the training process very time consuming and the classifying process unstable. In our method, we introduced line segment features. We first ran across ROIs with the Sobel operator to obtain the gradient of the image in a horizontal direction. The ROI of the license plate with detected vertical edges was shown in Fig. 5.4 (b). Binarisation was then deployed on ROI to remain significant edges, shown in Fig. 5.4(c). Finally, line segments were obtained by applying Hough Transform, as the red lines shown in Fig. 5.4 (d).

We constructed three properties of these line segments that discriminate license plate regions from their background. These three properties are density L_D , directionality L_{Di} and regularity L_R:

$$\begin{cases} L_D = n_{line}/N_{block} \\ L_{Di} = \frac{N_{\theta_v}}{N_{\theta_l}} (80^\circ \ll \theta_v \ll 100^\circ, 0^\circ \ll \theta_l \ll 179^\circ) \\ L_R = std\{\gamma\}/mean\{\gamma\} \end{cases} \tag{5.1}$$

As shown in Fig.5.4 €, n_{line} was the number of line segments in a block while N_{block} indicated the size of block by the number of pixels in this block. It tended to have a high density of line segments in a license plate region.

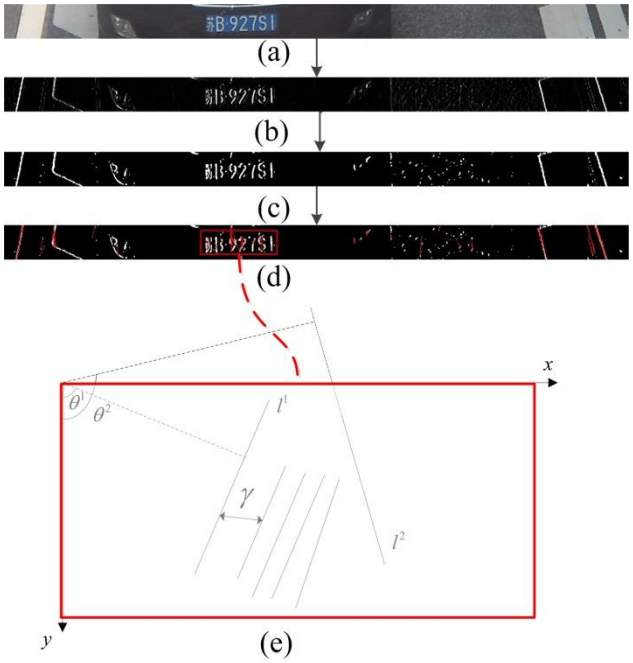


Fig. 5.4 License plate localisation based on line segments features including Density, Directionality and Regularity, (a) Original ROI, (b) ROI with detected vertical edges, (c) Binarised ROI, (d) ROI with detected line segment, (e) Line segment features.

Moreover, rather than being distributed in a disorderly fashion, line segments inside license plate tended to be approximately vertical. We utilised directionality L_{Di} to describe this pattern of line segments. Still referring to Fig.5.4, θ was the angle between the vertical axis (y-axis) and the perpendicular of the line segment. A line segment with θ within range [80, 100] was regarded as a vertical stroke. As shown in Fig. 5.4, both l^1 with $\theta^1$and l^2 with θ^2 were considered as vertical strokes. N_{θ_v} was the number of vertical line segments in a block while $N_{\theta_l}$was the number of all line segments in this block. The LP region should have high value of L_{Di}. Finally, regularity was introduced to measure the regular repetition of line segments inside LP. $\{\gamma\}$ was the inter-distances of vertical line segments in a block. The lower value of R indicates more regularity of the block. The localised license plate was shown as a red rectangle in Fig. 5.4 (d).

5.5.2 Feature subset selection by AdaBoost to achieve minimum redundancy

AdaBoost (Das 2001) is a popular and impressive method of selecting a subset of a large amount of Haar-like features and combine many weak classifiers into a strong

one. After constructing weak classifiers, one of which is based on a Haar-like feature, the training samples are re-weighted to emphasise ones that are incorrectly classified. Then the next weak classifier is trained with re-weighted samples. The predictions from those weak classifiers are combined through weighted votes to produce the prediction of a strong classifier. These weights are determined by classification error of each weak classifier.

After integrating line segment features into a traditional cascade of AdaBoost classifiers with Haar-like features, we dramatically drove down the amount of demanded Haar-like features, from 412 in traditional method to 180 features in our experiment, as well as the training time. With a rejection cascade consisting of 13 nodes, as illustrated in Fig. 5.5, a set of LP candidates were detected. We chose the fittest one after implementing non-maximal suppression.

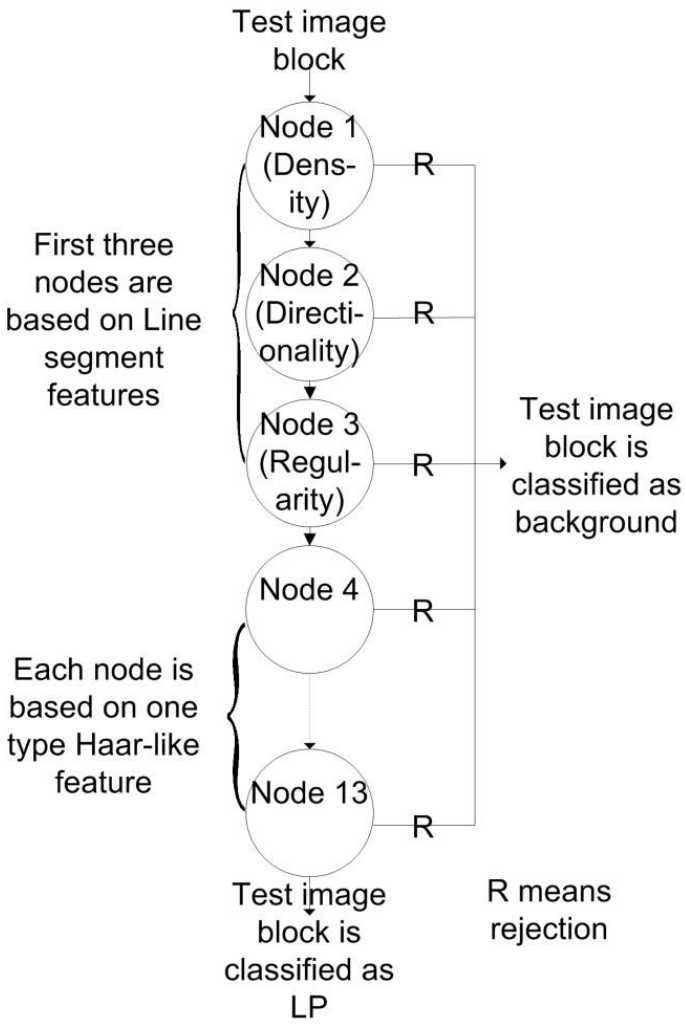


Fig. 5.5 Rejection cascade trained from line segment features and Haar-like features.

5.5.3 Vehicle type classification

As analysed in previous sections, we argued classifying vehicle type based on vehicle front is practical and robust. The vehicle front is usually the part of the vehicle body with the most details such as headlamps, blinkers, grille, logos, and license plates. It is very hard to use patch features such as SIFT and SURF to describe the vehicle front. However, same types of vehicle have fronts with similar patterns and designs. It is coincident that people can recognise the type of a vehicle depending on the view of vehicle front only. Eigenvectors are a powerful approach to describe the general pattern of objects. Eigenfaces are a set of eigenvectors used for human face recognition. It is considered as one of the first successful examples of facial recognition technology. Rather than projecting training samples of all classes in a single Eigenspace in traditional methods, we trained the classifier with multiple Eigenspaces.

5.5.3.1 Vehicle front extraction

Based on localised license plates, the vehicle front is extracted using background subtraction. We deal with this task using two methods for night-time and daylight, respectively.

Daylight case

For classification, each vehicle front should be detected and extracted. We implemented two steps:

1) vehicle front width was obtained by background subtraction; and

2) vehicle front was extracted based on vehicle width and the location of detected license plate.

License plate localisation has been explained in the last section. In this section, we dealt with vehicle front extraction, which were different for daylight cases and nightlight cases.

The situation was complicated in the daylight case. For daylight cases, we not only needed to subtract the background, but also remove shadows of the vehicle. To make our background subtraction algorithm robust, we captured several road images

containing no vehicle, under different lighting and weather conditions and then averaged them as a background. The subtraction operation was presented as follows:

$$D_k(x,y) = \begin{cases} 0, if\ |I_k(x,y) - B_k(x,y)| \leq T_d \\ 1, otherwise \end{cases} \quad (5.2)$$

where D_k was the difference image between input image and background image, I_k was the ROI region containing license plate, B_k was the correspondent ROI on averaged background image, and T_d was a predefined threshold which is chosen as the average of D_k . After subtraction, a simple morphological operation was applied to remove noise. However, due to the shadow, we cannot obtain the exact location of the vehicle after background subtraction.

In order to remove the shadow, we further implemented three steps proposed in (Wang, Chung et al. 2004) on the image processed by background subtraction:

1) Illumination assessment for the detected ROI to determine whether there were shadows present in this figure.

2) Through determining the direction of illumination and sampling shadow point, the attributes including average intensity and brightness & contrast of shadows were calculated.

3) The vehicle front was exactly extracted by three criteria: preserving bright pixels; preserving pixels with attributes different from attributes of shadow; preserving pixels nearby object edges.

Night-time case

When the input image was captured in nightlight, we easily obtained vehicle front width using above background subtraction technique, as there was no shadow that need to be removed.

5.5.3.2 Vehicle front extraction

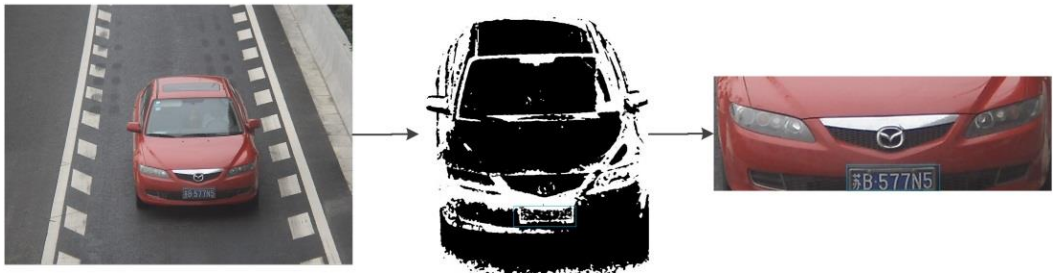


Fig. 5.6 Vehicle front extraction; the vehicle front is extracted correctly based on license plate localisation and foreground segmentation.

With vehicle front width and location of a license plate, we further extracted the vehicle front. As shown in Fig. 5.6, the height of the vehicle front was four times the height of the detected license plate. Moreover, the rectangle of the vehicle front shares it's bottom line with the license plate.

5.6 Eigenvectors extraction

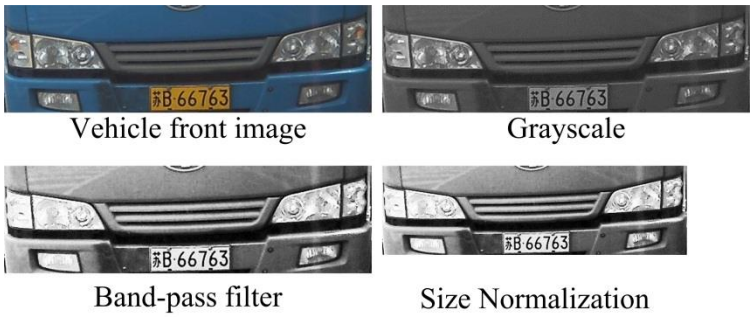


Fig. 5.7 Image pre-processing procedures including converting to greyscale, convoluting by band-pass filter and normalising the size of image.

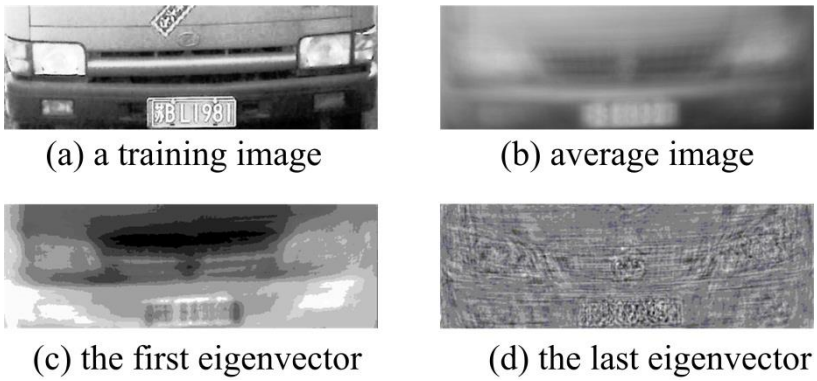


Fig. 5.8 Eigenvectors generation from training images; (a) an example of training image (b) average image (c) the first eigenvector (d) the last eigenvector.

In image classification, it was impossible to compute the difference between two images pixel-by-pixel when comparing these two images, because it was too time-consuming and the total noise contributed by every pixel will be very high. This noise could be anything that affects a pixel's intensity value. Therefore, we usually compared two images in a subspace with much lower dimensionality than the total pixels' number. PCA was a popular method to find the subspace, namely eigenspace. In our method, eigenvectors extracted from normalised vehicle front images were used for vehicle type classification. The eigenvectors described invariant characteristics of vehicle front images. Theoretically, the total number of eigenvectors we could find is the number of all images minus one. However, in practice, we only kept the ones with good separation capacity.

It was extremely important to apply image pre-processing techniques to standardise the images before eigenvector generation. Most image classification algorithms were extremely sensitive to many factors such as lighting conditions and image size. In order to reduce the adverse effect of various lighting conditions, we first converted all vehicle front images to greyscale and smooth them with a band-pass filter. Finally, we transformed all processed images into a fixed size. The image pre-processing procedure was explained by an example image in Fig. 5.7.

After image pre-processing, we obtained a set of regularised training vehicle front images. For training, we needed to label manually all images into four categories as above: truck, bus, minivan, and sedan. Every image can be represented as a m by n matrix that m and n being image height and width, respectively. In order to compute eigenvectors conveniently from all image matrices, we needed to store all images in one matrix. We first reshaped every image matrix to a $mn \times 1$ vector. With k being the number of training images, we stored all these images into a matrix of k columns $I = [I_1\ I_2 \cdots I_k]$. The length of I_i is $m \times n$. Then we can compute the average image of all training images and the difference images:

$$a = \frac{1}{k}\sum_{i=1}^{k} I_i\ , \sigma_i = I_i - a \tag{5.3}$$

where a was the average image as shown in Fig.5.8(b) while σ_i was a difference image. Both of them were represented by a $mn \times 1$vector. σ was a matrix storing all

difference images. The covariance matrix of I was:

$$C = \frac{1}{k}\sum_{i=1}^{k} \sigma_i {\sigma_i}^T = \sigma\sigma^T \tag{5.4}$$

We needed the eigenvectors of C. However, it was infeasible to calculate the eigenvector in (5.4) since $\sigma\sigma^T$ was a too large a matrix. In our experiment, the used vehicle front image was the size of 450×150. This made the size of $\sigma\sigma^T$ 67500×7500. This computation burden was avoided by the method in (Turk and Pentland 1991). Suppose μ_i was an eigenvector of $\sigma\sigma^T$ and λ_i was the associated eigenvalue, then:

$$\sigma\sigma^T\mu_i = \lambda_i\mu_i \implies \sigma\sigma^T\sigma\mu_i = \lambda_i\sigma\mu_i \tag{5.5}$$

where we can deduce $\sigma\mu_i$ is an eigenvector of $\sigma\sigma^T$. This method greatly reduced the computation complexity since the size of $\sigma^T\sigma$is only $k \times k$.

As explained above, $k-1$ numbers of eigenvectors would be generated from k training images. Fig. 5.8 (c) and Fig. 5.8 (d) showed the first and last eigenvectors in our experiment, respectively. As explained in Section 5.2, rather than selecting a set of eigenvectors to build a uniform eigenspace, we picked up the eigenvectors with biggest discriminability for each class to build the specific eigenspace. Each training image and test image was then represented as projection coefficients after being projected into the corresponding eigenspace.

5.7 Building multiple eigenspaces

We built multiple eigenspaces from different classes of training images using a Genetic Algorithm (GA). GA has been used to select eigenvectors before (Sun, Bebis et al. 2004), but for building a single eigenspace. GA was inspired from the biological mechanisms of reproduction.

GA operated iteratively on a population of eigenvectors, which were encoded as a string of binary symbols (selected eigenvectors are represented as 1, others are noted as 0). A randomly generated set of such strings forms an initial population from which the GA starts its search. As shown in Fig. 5.9, the genetic search process is iterative: evaluating, selecting, and recombining strings of the eigenvectors in iterations until reaching a termination condition. For each vehicle class, the eigenspace was built via this procedure with the corresponding class of validation images.

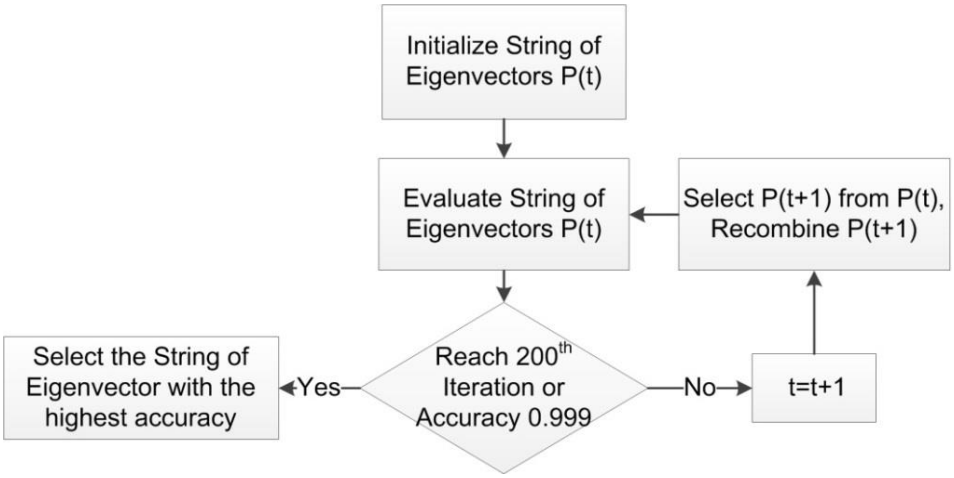


Fig. 5.9 The procedure of eigenvectors selection using genetic algorithms.

In our experiment, the termination condition depended on iteration number and accuracy range. The search process iterated until reaching accuracy 0.999 or maximum iteration number 200. Consequently, we chose the string of eigenvectors with highest accuracy. If there were more than two strings with highest accuracy, the string with less population of eigenvectors was preferred because the goal of feature selection was to use less features to achieve the same or better performance.

There were 3 basic operators guiding the search: selection, crossover, and mutation. We chose these operators according to the evaluation of each string. The evaluation determined which set of eigenvectors were better. Selection of a string depended on the string's fitness relative to those of other strings. The string with better fitness was more likely to be selected. With a crossover probability, we crossed over two selected strings to form a new string. With a mutation probability, we just flipped each binary value in the string to generate a new string.

The GA method setting was as follows:

Accuracy range: 0.5-0.999

Initial population size: 2500

Selection: Parent population size was N, the offspring size was 2N. We selected the best N from the 3N that was the combination of parent population (N) and offspring (2N).

Crossover: Uniform crossover that each binary value of the offspring was selected randomly from the corresponding parental strings. The crossover probability was 0.66.

Mutation: The mutation probability was 0.04.

5.7.1 Multiple eigenspaces vs. single eigenspace

The goal of feature selection was to enhance the classification accuracy with a more compact feature set. Using our eigenvector selection method, we built specific eigenspaces for each class (truck, bus, minivan, and sedan). Fig. 5.10 showed four eigenspaces containing different sets of eigenvectors that were constructed by GA from different classes of training images and the single eigenspace that was built by GA from the whole set of training images. As shown in Fig. 10, the multiple eigenspaces were much more compact than the single eigenspace – 46 eigenvectors out of 200 for truck

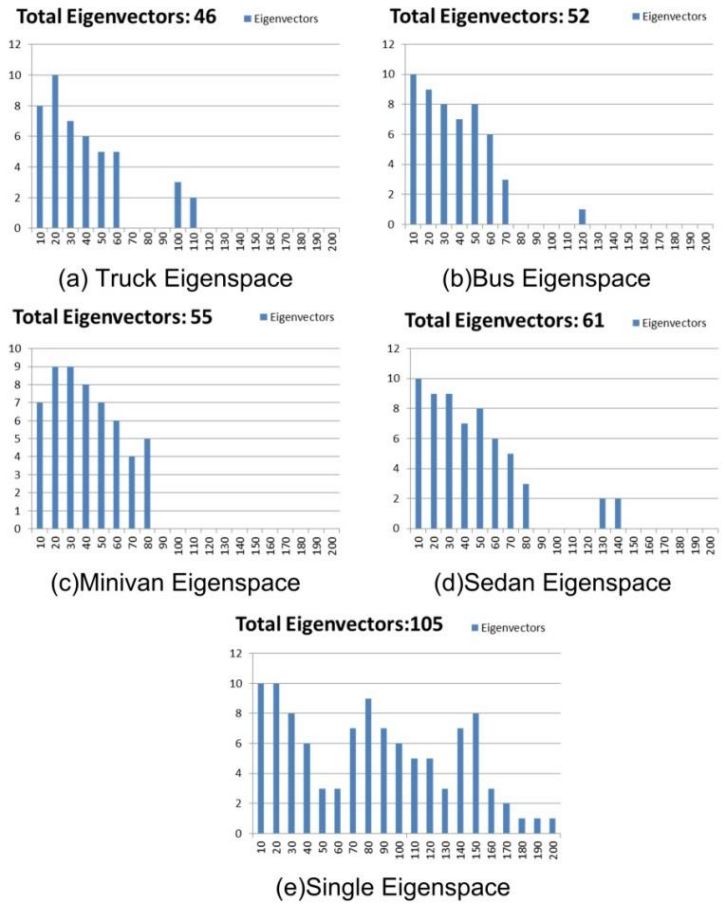


Fig. 5.10 Eigenvectors distribution in multiple eigenspaces and single eigenspace, respectively.

eigenspace, 52 eigenvectors out of 200 for bus eigenspace, 55 eigenvectors out of 200 for minivan eigenspace, 61 eigenvectors out of 200 for sedan eigenspace; and on the other hand, 105 eigenvectors out of 200 for a single eigenspace.

Different eigenvectors encoded different characteristics of target objects. Each vehicle front image can be represented by eigenvectors weighted by the projection coefficients after being projected into the corresponding eigenspace.

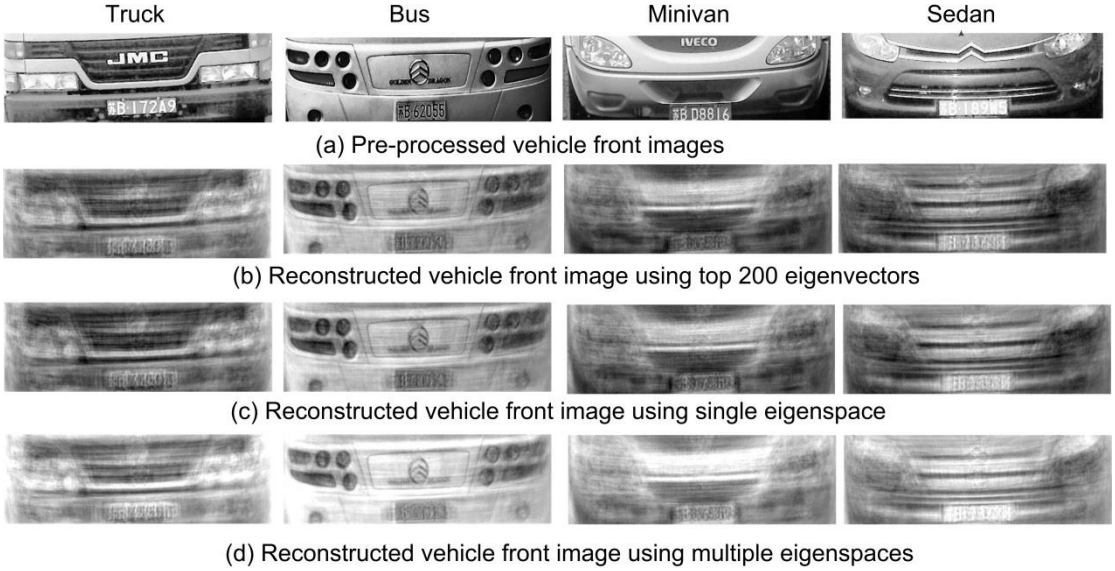


Fig. 5.11 Reconstructed images from eigenvectors: (a) Normalised vehicle front images (b) Reconstructed images from eigenspace consisting of top 200 eigenvectors (c) Reconstructed images from single eigenspace (d) Reconstructed images from multiple eigenspace.

For demonstrating the accuracy of the proposed eigenvector selection method, we reconstructed vehicle front images using selected eigenvectors only. In order to compare with single eigenspace, we also reconstructed the same vehicle front images using the top 200 eigenvectors and a uniform set of eigenvectors selected by the GA in (Sun, Bebis et al. 2004). The first row in Fig. 5.11 showed the normalised vehicle front images. The fourth row shows the images reconstructed by proposed multiple eigenspaces. The second row and the third row showed the images reconstructed by top 200 eigenvectors and a uniform set of GA selected eigenvectors. In the fourth row, all reconstructed images seemed to be normalised regarding illumination. Of particular interest were the reconstructed images in the fourth column which were brighter than the others. It appeared that the eigenvectors encoding illumination

information were excluded in the procedure of eigenvector selection, as illumination was not a critical factor for the discrimination among different vehicle types. Furthermore, the reconstructed images in the second and third rows show more details. All these images looked quite different. All images in the fourth row, by contrast, are smoother. As we explained before, discriminative eigenvectors should encode the specific characteristics for each class. In our multiple eigenspaces, irrelevant and redundant information were removed for each class. Therefore, the reconstructed vehicles showed the common characteristics in the class rather than individual identity information. In next section, the advantage of multiple eigenspaces over single eigenspace will be further analysed via comparison experiments.

5.8 Experiments

5.8.1 Front View Data Collection

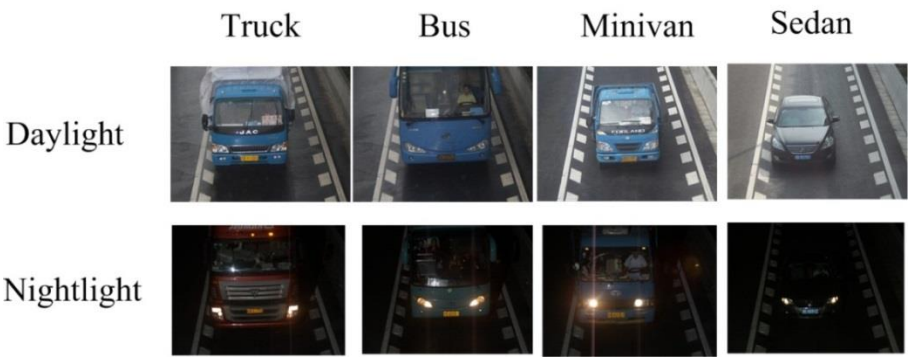


Fig. 5.12 Experiment images including 4 types of images: truck, bus, minivan and sedan.

As we don't have a public database with a sufficient number of images capturing vehicle frontal/rear views, we have established our own database, which includes 3327 images of vehicle frontal views. Everyone can access it for research purposes from http://dl.dropbox.com/u/52984000/Database1.rar. We hope it can be beneficial for continuing research in this topic.

We collected images of passing vehicles on a highway using a Sony HDR-SR12 camera. The camera, still-mounted on a pole, looks down and faces oncoming vehicles on a highway. The images are taken from 20th June, 2011 to 22nd June, 2011.

The images were taken in both of daylight with sunny and partly cloudy conditions and Night-time. As there are many more vehicles during day than during the night, there was a large difference between the numbers of daylight images and Night-timeimages. The total number of daylight images and night images are 2583 and 744, respectively. All images were with a size of 1600*1264 pixels. All images were manually labeled with one of the four classes: truck, bus, minivan, and sedan. Fig.5.12 shows examples of captured images. All collected images were separated into two groups: 2183 daylight images and 344 nightlight images for training; 400 daylight images and 400 nightlight images for testing. Table 5.1 indicates the numbers of each vehicle class in our experiments.

Table 5. 1 Experimental data collection.

	Daylight		**Nightlight**	
	Training	**Test**	**Training**	**Test**
Truck	551	100	156	100
Bus	410	100	56	100
Minivan	390	100	55	100
Sedan	832	100	77	100
Total	2183	400	344	400

5.8.2 License plate localisation evaluation

We localised LP with a rejection cascade, which consists of 3 classifiers based on line segment features and another 4 classifiers based on Haar-like features. For classifiers training, our experimental data consisted of 800 images containing LPs and 1000 images without LPs. Among LP contained images, 800 images were taken as positive samples, in which there were 800 visible LPs. The LP regions of 800 images were cropped manually, resized to images of size 60×0. We then employed illumination normalisation on all images to reduce negative effects caused by various lighting. For negative samples, 6000 image blocks of size 60×0 pixels were drawn from 1000 background images. Some examples of the cropped LP images are shown in Fig. 5.13.

Fig. 5.13 Cropped license plate images for training.

As shown in Table 5.2, our method utilised much fewer features than the traditional Haar-like feature based methods and therefore saved considerable time in the training stage. After adding line segment features into the Viola-Jones framework, a seven-node rejection cascade was obtained with only 180 features including 3 line segment features and 177 strong features based on Haar-like features. With the Viola-Jones framework alone, the rejection cascade consisted of 13 nodes with 412 features. Moreover, the training time required by our method was 5 days, compared to 14 days needed by traditional methods.

Table 5. 2 Comparison between Haar-like features based method and adaptive Haar-like features based method.

		HaLF	**LSF+HaLF**
Training	**nodes**	13	7
	features	412	180
	Time	14 days	5 days
Testing	**PD**	86.2%(250/290)	93.4%(271/290)
	FD	7.2%(21/290)	3.1%(9/290)
	Missed	6.6%(20/290)	3.5%(11/290)

*HaLF is Haar-like Feature, LSF is Line Segment Feature, PD is Positive Detection, and FD is False Detection

We compared our method with traditional Haar-like features based methods by implementing them on the same database. We used a public database containing 291 images taken in various parking lots in San Diego, California. All test images were a size of 640×80 pixels; and each image only contained one LP. The database link is http://vision.ucsd.edu/belongiegrp/research/carRec/car_data.html. Correct detection

was defined as detected areas overlaid with at least 85% of ground truth regions. Using our method, we found 271 LPs correctly while missed 11 LPs and detected 9 other objects as LPs falsely. By contrast, using the traditional method, 250 LPs were detected correctly while 20 LPs were missed and 21 were false detections. In addition, the average time for processing one image was 25 ms/frame vs. 70 ms/frame by our method and the traditional method, respectively. Obviously, the improvement in processing speed was important as we can save much time for following processing.

5.8.3 Vehicle front extraction evaluation

Table 5. 3 Comparison between proposed automatic vehicle front extraction and pre-defined vehicle front.

	Truck	**Bus**	**Minivan**	**Sedan**	**Total**
Test Images	189	190	195	180	754
CVFE by our method	180(95.2%)	190(100%)	190(97.4%)	171(95%)	731(96.9%)
CVFE by pre-definition	150(79.4%)	190(100%)	187(96.0%)	130(72.2%)	657(87.1%)

* Test Images are the test images with correct license plate detection; CVFE stands for correct vehicle front extraction

With license plate localisation, the vehicle front was extracted by background extraction and noise removal. Correct vehicle type classification was rested on accurate vehicle front extraction. It was extremely necessary to evaluate our proposed vehicle front extraction method. To avoid the bias brought by incorrect localised license plates, we evaluated the images with correct license plate localisation. Correct vehicle front extraction was defined as extracted areas overlaid with at least 85% of ground truth regions. For comparison, we extracted vehicle front using pre-definition method (Conos 2006), which extracts vehicle fronts as $w_v = 4 \times w$ and $h_v = 3 \times h$, where w_v, h_v w_v, h_v were the width and the height of the vehicle front and w, h were the width and the height of the localised license plate, respectively.

As shown in Table 5.3, our method achieved much better results (96.9%) than the manual definition method (87.1%). It was because a fixed rectangle cannot cover various vehicle fronts. For example, as shown in Fig. 5.14., using a defined vehicle

front area, we extracted incorrect fronts in the cases of a big truck and a small sedan, because the front sizes of the two are either too big or too small. On the other hand, with our proposed method, all fronts of the two vehicle types can be extracted correctly.

Fig. 5.14 Incorrect vehicle front extraction by pre-definition.

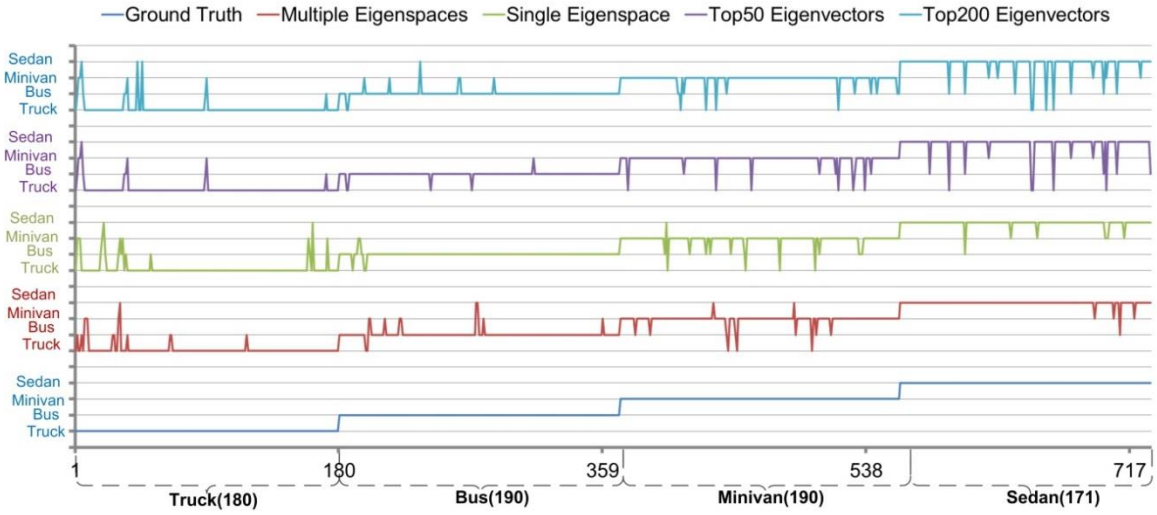


Fig. 5.15 Performance lines of different vehicle classification method; blue line is for ground truth, red is for MGA, green is for SGA, purple is for TES50, light blue is for TES200.

5.8.4 Vehicle type classification evaluation

In order to show the improvement of our proposed multiple-eigenspace method with genetic algorithm (MGA), we compared our method with single-eigenspace methods:

1. Top eigenvectors selection (TES): building an eigenspace using the top 50, 100, 150, and 200 eigenvectors, respectively.

2. Single-eigenspace with genetic algorithm (SGA): building an eigenspace with eigenvectors selected by GA (Sun, Bebis et al. 2004).

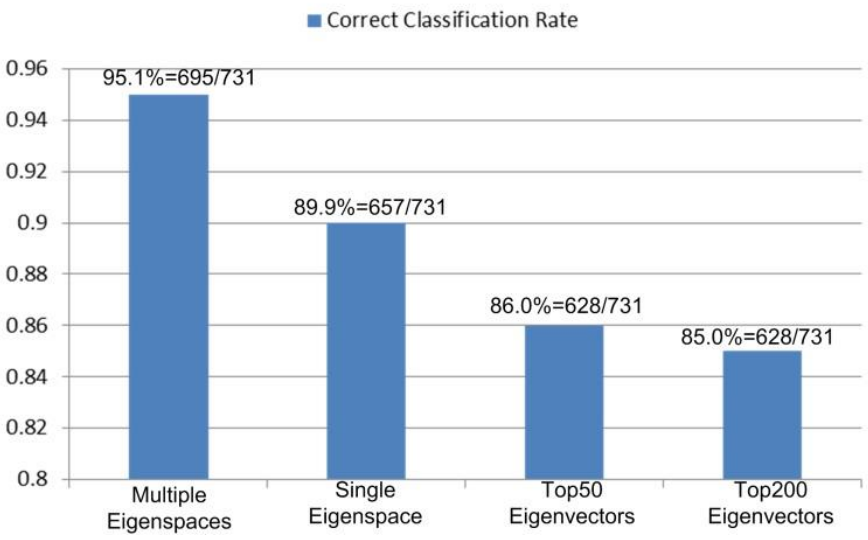


Fig. 5.16 Correct vehicle type classification rate using different methods, in order to avoid the bias brought about by incorrect vehicle front extraction, we evaluate classification on testing images with correct vehicle front extraction only (731) rather than the whole testing image set (800).

For avoiding bias brought about by incorrect vehicle front extraction, we used the vehicle front images extracted correctly in the last section for vehicle type classification evaluation. 731 testing images with correct vehicle front extraction contained trucks (180), buses (190), minivans (190), and sedans (171). In the Fig. 5.15, the blue line showed the ground-truth classes of each vehicle front; the red, green, purple and light blue lines represent the classifications using MGA, SGA, TES50 and TES200, respectively. In order to demonstrate the performance comparison clearly, we indicated the classification results using four steps: first step (truck), second step (bus), third step (minivan) and fourth step (sedan). On each

performance line, the fluctuations indicated incorrect classification. As shown, the ground truth line was wholly smooth on each step while fluctuations happened on other lines. All methods work better on truck and sedan images than on minivan and bus images. This corresponded to the visual observation; confusion was likely to happen between buses and minivans as sometimes they share some common characteristics in terms of their appearance. As shown in Fig. 5.15., the performance line (red) of multiple eigenspaces was much more stable than others, indicating that multiple eigenspaces method outperforms other methods.

Fig. 5.16. showed the correct classification rate for all the approaches tested. The first column to the fourth column are the results using MGA, SGA, TES50 and TES200, respectively. The results are 95.1%, 89.9%, 86.0% and 85.0%, respectively. Our proposed MGA method achieved the highest classification rate.

5.8.5 Degree of step compliance and method limitations

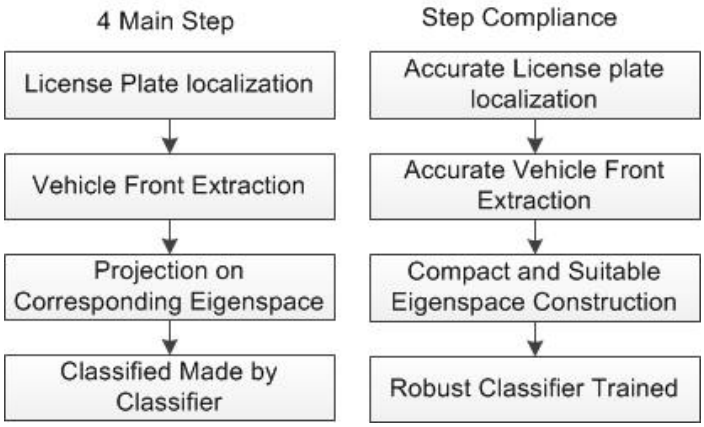


Fig. 5.17 Main steps and step compliance in our method.

As explained at the beginning of this section, the vehicle type classification system is a complicated system, which contains more than one step. Only good performance achieved in each step can lead to accurate and robust vehicle type classification. As shown in Fig. 5.17, our proposed vehicle type classification method contains 4 main steps: license plate localisation, vehicle front extraction, projection on eigenspace, and type classification. A high degree of compliance on each step contributes to a final and accurate result. Still referring to Fig. 5.17, each step should be implemented

correctly:

1) correct license plate localisation that is defined as localised license plates overlaid at least 85% of ground truth area.

2) correct vehicle front extraction that is defined as extracted vehicle fronts overlaid at least 85% of ground truth area.

3) extracted vehicle front image is projected into the constructed multiple eigenspaces.

4) correct determination made by trained classifier.

Though our method is robust in all these steps as shown by experimental analysis, we experienced two types of failure: incorrect license plate localisation and vehicle front extraction. Though the effective performance of both license plate localisation and vehicle front extraction were evaluated in Section 5.9.2 and Section 5.9.3, the failures of them are indeed nuisances as they are fundamental for the following processes.

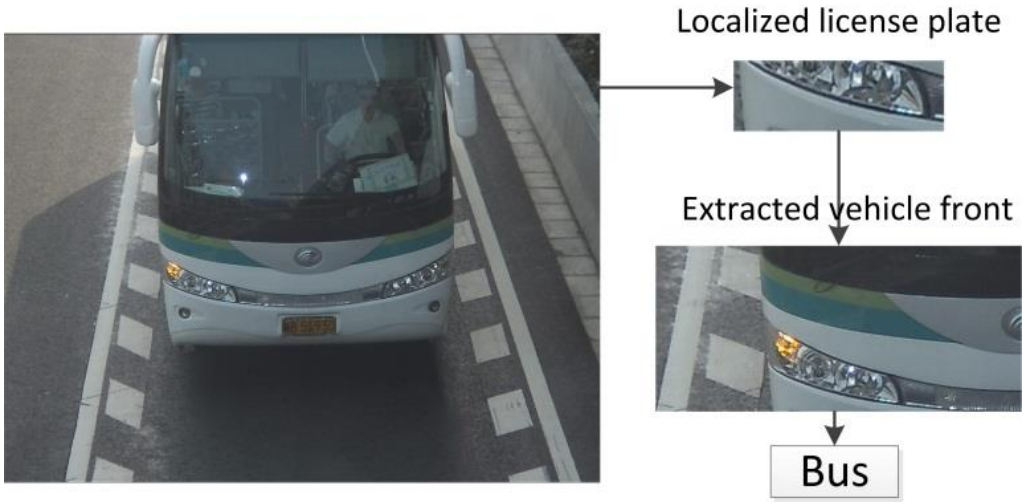


Fig. 5.18 "Lucky" Correct classification, some "lucky" correct classification happened even when a license plate or vehicle front is detected incorrectly.

When we used the proposed MGA on the whole set of test images, the correct classification is 88.8% (710/800). The result was much lower that the classification rate demonstrated in last section, because the adverse effects brought by wrong license plate localisation and vehicle front extraction. It was interesting to note that

the correct classification rate for the whole set of test images should be 86.9% (695/800) in theory, as we classified 695 images as shown in Fig. 5.16. The actual classification rate was a little higher because of some “lucky” correct classifications. As shown in Fig. 5.18, the system occasionally happens to make correct classifications even though it has wrongly localised the license plate and vehicle front.

5.9 Conclusion

This section validates the methodology of feature selection by a practical and robust vehicle type classification system. This system classifies a vehicle into one of four classes (truck, bus, minivan, and sedan), based on a vehicle’s front view. Our method extracted the vehicle front region automatically. To enhance the classification rate, we utilised an adaptive PCA method with multiple eigenspaces in classification. The comparison experiments demonstrated the advances of the proposed method. Finally, we made our database including 3327 vehicle images available to the public. This may help to further research in this area.

The schemes of maximum dependency and minimum redundancy guide the implementation of this system. According to the maximum dependency scheme, we propose Line Segment Feature (LSF) based on the special pattern of license plates. In order to speed up the process, we combine LSF with Haar-like features to generate a candidate set of features. Because Haar-like features are weak features, the generated candidate set contains large amount of features. For achieving the optimal feature set, we utilised AdaBoost to produce strong features. This procedure becomes much faster than the traditional Haar-like— AdaBoost method because of the introduction of LSF.

For vehicle type classification, we proposed an adaptive PCA method to select max-dependent and min-redundant features to present the vehicle front through building up multiple eigenspaces than the single eigenspace in traditional PCA methods.

Chapter 6 Feature Selection for Pedestrian Counting

The content in this chapter has been published in (Peng, Xu et al. 2012).

The content in this chapter has been included in the manuscript “Fast and Stable Pedestrian Counting System”, which has been submitted to the journal “Multimedia Tools and Applications”.

6.1 Background

Accurate detecting and counting of pedestrians in video sequences represents an essential component in a wide range of applications in ITS. In traffic control, automatic pedestrian detection and counting can be used to enhance safety and improve timing of traffic. In public place surveillance, pedestrian detection and counting is an important indicator for congestion, delay or other abnormalities. Furthermore, accurate pedestrian detection and counting provides pedestrians’ images for further analysis with the aim to estimate body poses, recognise actions or identify faces.

Although many advanced works on pedestrian detection and counting have been conducted in the previous two decades, many of them have restrictions: the background must be simple, occlusions must be few, image resolution must be high, or don’t require real-time processing speed. However, obviously, real scenes always include simple and complicated backgrounds, and occlusions usually happen. Moreover, real-time processing is substantially significant in visual surveillance system. In this chapter, we intend to propose an effective method for pedestrian detection and counting, which could be used to address the aforementioned limitations. Our objective is to achieve robustness and high accuracy rates whilst maintaining real-time (>25 fps) performance in both simple and complicated scenarios.

Current methods for people-counting can be classified into two categories: detection-based and map-based methods. Detection-based methods estimate the number of people by identifying individuals in the scene. These methods determine the number of pedestrians and localise them simultaneously whereas cope with high crowd density and occlusions at the same time. Map-based methods ascertain the number of pedestrians in the scene through establishing the relationship between the pedestrian

number and certain features extracted from the scene. Though their processing is usually quick, they are very sensitive to noise.

In this chapter, we propose a blob-based method. After segmenting the foreground, we detect the contours using multi-scale analysis. Then the foreground pixels are divided into different blobs independent from each other by detected contours. Precisely, each blob contains zero, one or several pedestrian(s). A new model with hybrid features consisting of low-level and high-level features was utilised to represent each blob. A classifier based on these extracted features is trained to estimate pedestrian numbers within each blob. Finally, blob tracking with the analysis of blob match, split or merge was proposed to optimise the estimation of the pedestrian number. Taking advantage of Graphics Processing Units (GPU) acceleration, we implemented our method using Computer Unified Device Architecture (CUDA), which was developed by Nvidia for graphics processing and parallel computing.

The remaining chapter was structured as follows: Section 6.2 presents previous work for pedestrian detection and counting. An overview of our method is then provided in Section 6.3. Section 6.4 describes the core part - features selection. We introduce the procedure of motion analysis for improving performance in Section 6.5. Comparison experiments with other popular methods are shown in Section 6.6. This chapter is concluded in Section 6.7.

6.2 Related work on pedestrian counting

Various approaches pertaining to pedestrian detection and counting have been proposed in the last twenty years. As aforementioned analysis describes, these methods usually belong to one of two categories: detection-based methods and map-based methods.

Human appearance models were usually used in detection-based methods. These methods usually slide the whole video frame with scalable window and classify the window image, after training classifiers based on extracted features with machine learning methods. Based on pedestrian body shape, (Dalal and Triggs 2005) proposed a pedestrian detector with Histogram of Oriented Gradient (HOG) features. Though the authors demonstrated impressive pedestrian detection performance, this method

was expensive in computation, sensitive to camera angles and very limited in the occlusion cases. In (Zhao, Delleandrea et al. 2009), a method based on face detection and tracking was proposed and free camera viewpoint was achieved. Undoubtedly, using the face as a reference rather than the entire body could be more reliable since faces are not generally obscure in surveillance cameras. Unfortunately, this method was invalid when pedestrians are not facing to camera. In (Li, Zhang et al. 2008), authors trained classifiers based on HOG heads and shoulders' features with Adaboost framework to detect people in each frame. Methods based on head detection were more robust than face-detection-based methods, as they did not require pedestrians to face towards the camera. However, neither method could obtain a promising result when video definition was low or environment was complex, as both the face and head were very small.

An interesting method for individual detection was raised in (Rodriguez and Shah 2007). The method learned a set of pedestrian posture clusters, and a codebook of local shape distribution for pedestrians in various postures. The method initialised the pedestrian segmentation by the corresponding posture clusters in the codebook and evolved contours to obtain precise and consistent segmentations. The paper showed impressive results even in crowd scenes. However, the segmentation initialisation was risky and contour evolvement was high in computational complexity.

Besides this, another major drawback of most individual detection and counting-based methods is that they assume there is a distinct visual separation between individuals. Only under this assumption, can the aforementioned methods of individual detection work satisfactorily. However, this assumption is not the common case in surveillance videos. Furthermore, pedestrians are often severely occluded and visually inseparable, as shown in Fig. 6.1.

Meanwhile, the map-based methods can be further classified into two groups. Some methods tried to establish the relationship between the number of pedestrians and the features of the whole image. The other methods counted the pedestrian number for each moving group, after segmenting moving objects from their backgrounds.

(a) Original Frame (b) Foreground segmentation

Fig. 6.1 Occlusion scenario in surveillance video.

Researchers utilised global features of the whole frame, such as texture information (Benfold and Reid 2009), fractal dimension (Benfold and Reid 2009) and invariant orthonormal Chebyshev moments (Rahmalan, Nixon et al. 2006) to estimate pedestrian numbers on each frame. Nevertheless, global features were highly sensitive to illumination changes. Therefore, these methods do not work well in the open air, where light changes correspondingly at different time points. Moreover, these methods can only estimate the numbers of pedestrians roughly in the video frame.

In (Kilambi, Ribnick et al. 2008), a method provided that used geometric projections to estimate pedestrian numbers in moving groups. The method required prior knowledge of camera setting and accurate camera calibration to learn the estimation for the pedestrian number. The method showed promising results in both outdoor and indoor environments, whereas prior knowledge was usually unavailable in reality. The method in (Albiol, Silla et al. 2009) assumed that each pedestrian, on average, had a particular number of corners and then determined pedestrian number in a moving group by analysing detected corners. While the number of corners detected on each pedestrian differed considerably due to variations in camera angle, distance from people to camera and video definition. (Kong, Gray et al. 2005) proposed a method that extracted features dependent on foreground segmentation and edge detection. Though the authors indicated their method took into account feature normalisation to deal with perspective projection and different camera orientation, the method deviated significantly by the edges detected from the background. Moreover, homograph calculation required in this method was difficult to achieve. Another two methods (Ryan, Denman et al. 2009), (Ryan, Denman et al. 2010) extracted several features

from each detected moving group. Based on these features, the classifier was trained using neural network or a linear model. Likewise, in consistence with most of aforementioned methods, these methods determined the pedestrian number of moving groups out of the assumption that segmented foregrounds were pedestrians. It was worth noting that in surveillance video, no foreground segmentation method can segment pedestrians only. Discriminating pedestrians from other segmented objects such as moving vehicles, shaking tree branches and running cats, is required.

Moreover, computational load always came to a bottleneck. Most of the aforementioned methods utilised repeated processing techniques such as the slide-window algorithm (Dalal and Triggs 2005) (Zhao, Delleandrea et al. 2009) (Li, Zhang et al. 2008) (Rodriguez and Shah 2007) and the pixel-wise calculation algorithm (Kilambi, Ribnick et al. 2008) (Albiol, Silla et al. 2009) (Kong, Gray et al. 2005) (Ryan, Denman et al. 2009) (Ryan, Denman et al. 2010), which were of significantly high computational complexity. Because of the expensive computation, some of these methods could hardly perform in real time.

As analyzed in Chapter 2, low-level features usually describe subtle details of an object but are time-consuming. On the other hand, high-level features represent shape and spatial information of objects, and usually have a low computational load. Pedestrian shapes have features discriminating them from other objects. That is why we can easily recognise pedestrians by silhouette. In order to guarantee the accuracy and real-time processing, we utilised hybrid features combining low-level features and high-level features in our approach to counting pedestrians. Moreover, we implement the parallel computing architecture in our system to boost the speed. Experiments showed our system can achieve real-time processing even on HD (1920×1080 pixel^2) surveillance video.

6.3 Method overview

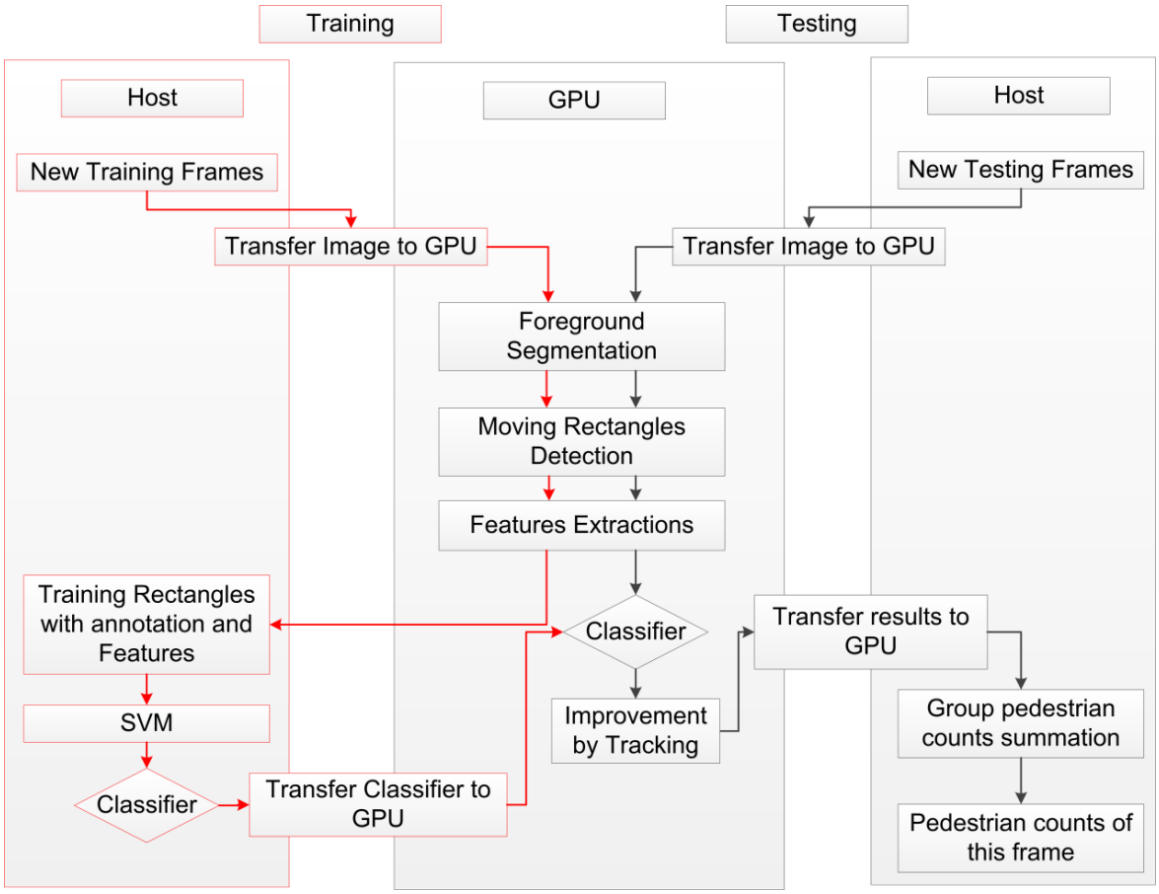


Fig. 6.2 Method Overview of our pedestrian counting system.

Fig. 6.3 Pedestrian counts are the summation of estimated counts of all moving groups.

As shown in Fig. 6.2, our method consisted of two stages: training, and testing; and two processing parts: host, and GPU processing. In both the training and testing stages, the image was acquired by the host, copied into GPU memory and then processed by the GPU.

In training stage, the method captured and copied training videos into GPU memory frame by frame. Extended Gaussian mixture model was implemented on frames for foreground segmentation. The method then removed noise from the segmented foreground as well as kept information of interest by median filter. Separate blobs are detected. It indicates moving objects, which included single pedestrian, several pedestrians or other objects. Relevant features were extracted from each moving blob. Moving blobs were annotated manually as the following: blobs including a single pedestrian was annotated with 1; blobs including several pedestrians were annotated with the pedestrian number of the corresponding blob, which were usually from 2 to 10; blobs of other objects were annotated with 0 unanimously. Then annotated blobs with extracted features were learned by Support Vector Machines (SVMs) for training the classifier. During testing stage, blobs were detected and features were extracted from testing frames based on the same methods as in training stage. After this, the blobs with extracted features were grouped according to trained classifier. In the next step, the classified pedestrian number of each moving blob was optimised by blobs tracking, which was based on a novel analysis of blobs split and merge Eventually, the total pedestrian number of each image was the summation of pedestrian numbers of all blobs, as shown in Fig. 6.3.

6.4 Feature selection scheme for counting pedestrian

Our method detected blobs, after segmenting foreground using extended Gaussian mixture model. Nine highly relevant features were then extracted from each blob.
According to the feature selection strategy, we choose features based on max-dependency and min-redundancy. By accumulating occurrences of gradient orientation in localised portions of an image, HOG is an impressive shape-based method for pedestrian detection. However, as the analysis in Section 6.2 discusses, HOG is not a suitable feature for our application, the aim of which is counting pedestrians, and beyond simply detecting them. In surveillance videos, there are

usually several pedestrians appearing within one frame. Moreover, occlusions happen frequently. In order to take the advantages of the pedestrian's shape, we count pedestrians based on blobs extracted from the background rather than detecting pedestrians directly. We extracted low-level and high-level features from each blob to indicate the number of pedestrians in this blob.

In order to avoid detrimental effects brought about by the changing angle between camera and pedestrian, we extract "Grid Index" to describe the position relationship between the camera and each blob. The ratio of the height and width of blobs is also an important feature indicating the pedestrian counts. Low-level features such as density, density variance, horizontal mean, horizontal variance, vertical mean, and vertical variance show the statistical description of blobs with different pedestrian counts.

6.4.1 Extended Gaussian mixture model

The Gaussian Mixture Model (GMM) stores for each pixel x_t M separated normal distributions during a time adaptation period T, which is parameterised by mean u_i, variance σ_i^2 and mixing weight w_i:

$$P(x_t) = \sum_{i=1}^{M} w_i \mathcal{N}(x_t; u_i, \sigma_i^2 I) \tag{6.1}$$

where $x_t$is a pixel intensity value at time t, M is between 3 and 5, depending on the complexity of the scene, and I is an identity matrix with proper dimensions. The estimated mixing weights w_i are non-negative and add up to one.

The Extended Gaussian Mixture Model (EGMM) using normalised mixture weights w_i to define an underlying multinomial distribution describing the probability that a sample pixel belongs to the i-th component of the GMM. Incoming pixels are weighted by the recursive update equations:

$$\begin{aligned} w_i &= \alpha n_i = \alpha \sum_{j=1}^{T} {o_i}^j \\ w_i &\rightarrow w_i + \alpha(o_i - w_i - \alpha c) \\ u_i &\rightarrow u_i + o_i \frac{\alpha}{w_i}(x_t - u_i) \\ \sigma_i^2 &\rightarrow \sigma_i^2 + o_i \frac{\alpha}{w_i}((x_t - u_i)^2 - \sigma_i^2) \end{aligned} \tag{6.2}$$

where α is a constant defining an exponentially decaying envelope that is used to limit the influence of the old data, o_i is an ownership notation that is set to 1 for the "close" component with largest w_i and the others are set to 0, -αc is a negative weight that imposing a minimal amount of evidence requirement before a component can be allowed to exist. The "close" component is defined when the Mahalanobis distance from the component is, for example, less than three standard deviations:

$$D_m^2(x_t) = (x_t - u_i)^2/\sigma_i^2 \tag{6.3}$$

If no "close" component is found, a new Gaussian component is created with

$$w_{M+1} = \alpha,\ u_{M+1} = x_t, \sigma_{M+1} = \sigma_0 \tag{6.4}$$

where σ_0 is a large initial variance. If the maximum number of components is exceeded, the component with the lowest weight will be discarded.

A foreign object appearing in the scene would be represented by some additional components with low weights and high variances. After implementing EGMM on frames, moving objects were segmented from background. A median filter was used to remove noise from segmented results, we detected blobs and found upright rectangles to separate these blobs. As shown in Fig. 6.4, these detected rectangles contained pedestrians as well as other objects (such as pigeons and moving waste garbage bags). In order to discriminate these rectangles and calculate pedestrian numbers within each rectangle, we trained the classifier based on extracted rectangle features.

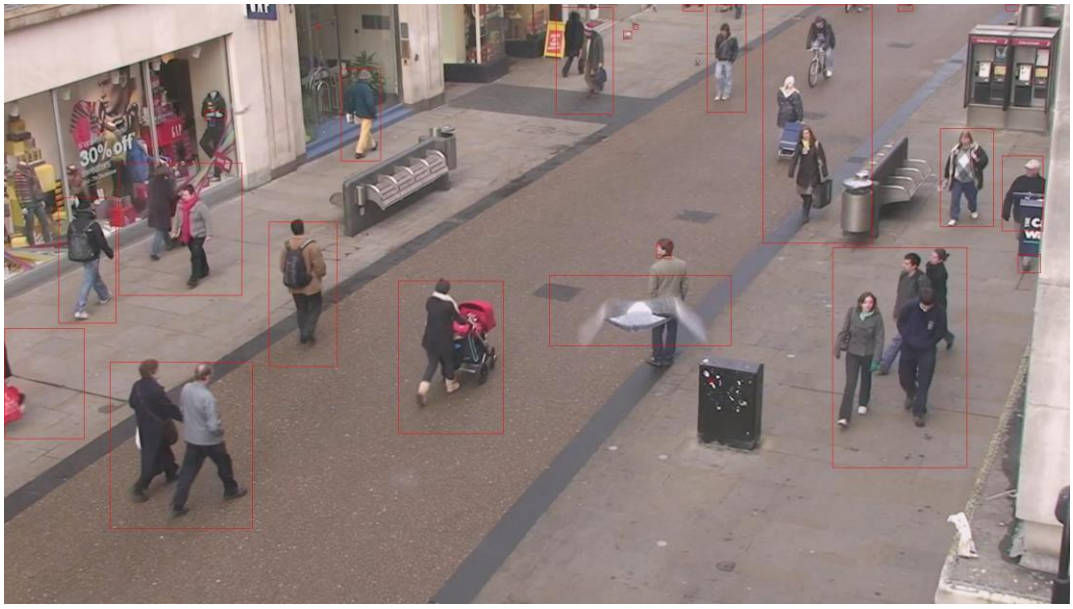


Fig. 6.4 Detected rectangles contain pedestrian(s) as well as other objects.

6.4.2 Hybrid features extraction based on maximum dependency

Nine highly relevant features were extracted from each rectangle in order to estimate the number of pedestrians in this rectangle. Please note that we defined rectangles containing no pedestrian equivalent to pedestrian count of 0. All features were examined on binary images obtained from foreground segmentation and noise removal, which was described in the last section. More specifically, these features were:

Grid Index: According to our observation, rectangles that contained the same number of pedestrians on different frame locations had different appearances. As shown in Fig. 6.5, both the two rectangles contain two pedestrians while they are with different sizes given that their distances to camera are different. Instead of using the locations of rectangles, we divided each frame into 10×10 grids and indicated the grids, to which rectangles belong:

$$argmin(|R(x,y) - G_i(x,y)|, i), i \in \{1, \dots, 100\} \quad (6.5)$$

We chose the grid indicator i when the distance between rectangle center $R(x,y)$and grid center $G_i(x,y)$ is minimized. Taking the advantage of grid indication, our method is robust to videos of different definitions.

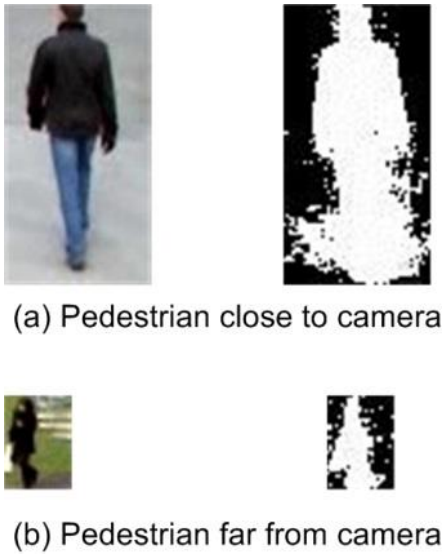


Fig. 6.5 Both rectangles contain one pedestrian while they are of different sizes.

Width and Height: Rather than using areas of rectangles as in previous methods (Ryan, Denman et al. 2009) (Ryan, Denman et al. 2010), we adopted both relative width and height as features. Since pedestrians usually walk upright, even rectangles of other objects have similar areas of pedestrian rectangles. Additionally, we took various video definitions into considerations. The relative width W and height H are defined as:

$$W = \frac{Width}{FrameWidth}, \quad H = \frac{Height}{FrameHeight} \tag{6.6}$$

where $Width$ and $Height$ are the width and height of moving rectangle, respectively, while $FrameWidth$ and $FrameHeight$ are the width and height of video frame, respectively.

Density: Density is used to describe the Foreground Pixels (FPs) density of each rectangle using:

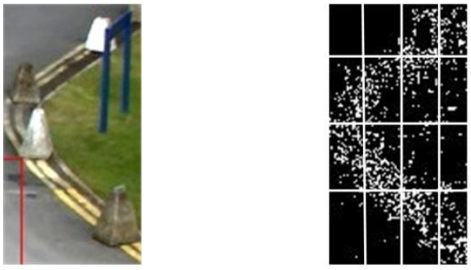
$$DS = F/N \tag{6.7}$$

where F is the number of FPs in the rectangle and N is the number of all pixels in the rectangle.

Density Variance: Besides the FPs density information, we should note that FPs in rectangles with the same pedestrian numbers usually distribute with certain common characteristics. Hence, we utilised FP density variance as a clue to discriminate

rectangles with different pedestrian numbers. To obtain this feature, each rectangle is divided into $n \times$ equal-sized sub-blocks, as shown in Fig. 6.6. Let g_i denote the

(a) Rectangle of pedestrians



(b) Rectangle of other objects

Fig. 6.6 Each rectangle is divided into 16 equal-sized sub-blocks for Density Variance Calculation.

foreground pixel number in sub-block i, and DS is the FP density described in (7). Then, the density variance V_g is defined as:

$$V_g = \frac{\sum_{i=1}^{n} |g_i - DS|}{n^2 * DS} \tag{6.8}$$

where n^2 is the number of foreground pixels in sub-blocks, e.g., $n = 4$ in our experiment.

Horizontal Mean, Horizontal Variance, Vertical Mean, and Vertical Variance: Due to variances in pedestrian shape and walking posture, some characteristics of rectangles can be described by Horizontal FP Mean (HM), Horizontal FP Variance (HV), Vertical FP Mean (VM), and Vertical FP Variance (VV):

$$\begin{aligned} HM &= F/Width \\ HV &= \frac{\sum_{i=1}^{Width} |I_i - HM|}{Width * HM} \\ VM &= F/Height \end{aligned} \tag{6.9}$$

$$VV = \frac{\sum_{j=1}^{Height} |I_j - VM|}{Height * VM}$$

In the training stage, all rectangles are annotated manually with 0, 1, 2,…, 10. With the above extracted features, the classifier is trained via SVM. We use libsvm library with radial basis function kernel (Chang and Lin 2011).

6.5 Improved counting with motion analysis

Our method treated each frame of video as an independent one, and estimated the pedestrian number based on features extracted from each moving rectangle. Although some tracking techniques such as the Kalman filter could smooth the rectangle trajectory, the splits and merges of rectangles are ignored. By contrast, in our method, we improved the tracked results according to the match, split and merge analyses.

Precisely, we first detected matches between each rectangle in two consecutive frames. For instance, given two rectangles *A*, *B*, two rectangles match is defined as:

$$D(A,B) < \frac{\sqrt{(A_W^2 + A_H^2)} + \sqrt{(B_W^2 + B_H^2)}}{2}$$

$$bottom\ threshold < \frac{B_W * B_H}{A_W * A_H} < up\ threshold \quad (6.10)$$

where $D(A,B)$ is the distances between centers of A and B, A_W, A_H, B_W and B_H are width and height of rectangle *A*, width and height of rectangle *B*, respectively. According to our experiment, we achieved best results when *bottom threshold* and *up threshold* are 0.8 and 1.2, respectively.

After matched rectangles are found, left rectangles are split or merged. To match A_n ,A_m in previous frame and B in current frame, we combine A_n and A_m to a joined region $M_A = A_n \cup A_m$. If M_A and B satisfy (10), we determine A_n ,A_m in the previous frame merges into B in the current frame. To match B in previous frame and A_n ,A_m in current frame, we combine A_n and A_m to a joined region$S_A = A_n \cap A_m$. If B and S_A satisfy (10), we determine A_n and A_m in current frame are split from B in previous frame.

Taking advantage of rectangles tracking with split-merge analysis, the pedestrian counting is improved. Specifically, the pedestrian counting of current rectangles is

optimised by the historical records of this rectangle according to the following four cases:

No match: If no matched rectangle was found in the previous frame, we only consider the current pedestrian counts of this rectangle.

Direct match: If a directly matched rectangle in the previous frame was found, we take the current pedestrian count as the median value of the counting record, which includes current counting and maximum -9 historical counts.

Merge: If two rectangles A_n ,A_m were found in the previous frame for one rectangle *B* in the current frame, a new counts record is formed by summing corresponding counts in the two lists of $A_n$and A_m. We take current pedestrian counts as the median value of this new record, which include current counts and maximum -9 historical counts.

Split: If one rectangle *B* was found in the previous frame for two rectangles A_n,A_m in the current frame, the current pedestrian counts are considered as:

$$C_{A_n}^o = (C_B * \frac{R_{A_n}}{R_{A_n}+R_{A_m}} + C_{A_n})/2, \qquad C_{A_m}^o = (C_B * \frac{R_{A_m}}{R_{A_n}+R_{A_m}} + C_{A_m})/2 \tag{6.11}$$

Where $C_{A_n}^o$ and $C_{A_m}^o$are optimised pedestrian counting, C_B is pedestrian counts of *B*, R_{A_n} and R_{A_m} are areas of $A_n$and A_m, C_{A_n} and C_{A_m} are current pedestrian counts.

6.6 Experiment and discussion

6.6.1 CUDA implementation

Our method implemented with a low-end GeForce 310M GPU on CUDA framework provides at least 10x speedup, which would be demonstrated in experiment section. The CUDA framework exposes the Single Instruction Multiple Data (SIMD) architecture of the GPUs by enabling the parallel operation of the program kernels on image pixels, divided them into multiple blocks consisted of several threads. Threads in a block can cooperate while blocks in the image are independent. This programming model allows for a very high degree of scalability. The highest performance is achieved when the threads avoid divergence and perform the identical operation on image. The threads can access data during the execution process for 6 different types of memory: register, local, shared, global, constant and texture

memory. All threads can access global, constant and texture memory while only threads in the same block can access a shared memory. Each thread has private local memory and registers. Constant and texture memory are read-only while all the others are read-write. The above basic knowledge guides the CUDA to implement our method. The details of CUDA memory hierarchy and heterogeneous programming model can be referred to in (Nvidia 2012).

6.6.2 Experimental data

In this section, we analyse the performance of our method. Experiments were performed on three benchmark public surveillance videos: two videos are from PETS 2009 (Database 2009) and the other one comes from Town Centre Database (Benfold and Reid 2011), and then compared with two current methods: fastHOG (Prisacariu and Reid 2009) and Crowd Counting using Multiple Local Features (CCMLF) (Ryan, Denman et al. 2009). Attributing to the open-ource code of fastHOG, we saved substantial time. The experimental videos are real-world pedestrian scenes with different backgrounds (both simple and complicated), viewpoints, and number of pedestrians within one frame ranging from a few individuals to over 50. Whereas in our experiment, the number of pedestrians ranges from 0 to 10. We should bear in mind that 10 was the maximum size of one group, not the total number of pedestrians in the frame. The total pedestrian count in a frame was the summation of the numbers of pedestrians within all the groups. All experiments were implemented on a Intel™ Core™2 Duo CPU 3.00GHz 2.00 GB memory PC with a Geforce 310M GPU card. Please refer to the supplementary material or via this link http://www.youtube.com/watch?v=wENabEdKDZo to watch a demonstration video.

(a) PETS2009_1 (b) PETS2009_2 (c) TownCenter

Fig. 6.7 Snapshots of the three experimental videos. The PETS2009 database is copyright University of Reading and permission is granted for free download for the purposes of academic and industrial research (Database 2009).

6.6.3 Classifier training

As mentioned previously, we used three videos in the experiment. Two videos came from PETS2009, the other one was from Town Centre Database. PETS2009 was a set of 768×576 pixels JPEG image sequences in outdoor condition. It had 4 subsets, each subset containing several sequences and each sequence contained views which ranged from 4 to 8. More precisely, we chose view 1 from subject 1, sequence 1 scenario and view 1 from subject 2, sequence 1 scenario to create two 768×576 at 20 fps videos. We named these two videos as PETS2009_1 and PETS2009_2 respectively, two frames of which are shown in Fig. 6.7 (a) and Fig. 6.7 (b). Meanwhile, Town Centre Database contained a video of a busy town centre street. The video was high definition (1920×1080 at 25fps). This video was used to test pedestrian detection and tracking performance in (Prisacariu and Reid 2009) (Benfold and Reid 2011). Fig. 6.7 (c) shows a snapshot of this video.

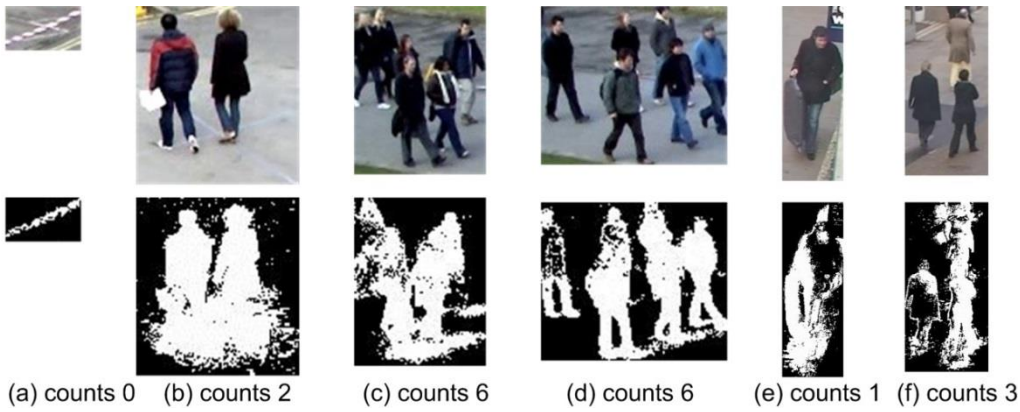


Fig. 6.8 Examples of binary rectangles and their colour corresponding.

The lengths of the three videos (PETS2009_1, PETS2009_2 and Town Centre Video) were 1584 frames, 478 frames, and 7500 frames respectively. In order to collect training data, we implemented a rectangle detection algorithm on the first halves of these videos, respectively, as described in the method overview. After the classifier was trained, we tested our method on the second halves of these three videos. As described in Table 6.1, we collected 3919 rectangles, 997 rectangles, and 45614 rectangles from 793 frames of PETS2009_1, 239 frames of PETS2009_2, and 3750 frames of Town Centre Video respectively. After extracting 9 features from each binary rectangle, we manually annotated these rectangles with pedestrian numbers (0

to 10). With the aim to guarantee correct pedestrian counting, annotation was made for each binary rectangle via viewing the corresponding colour rectangle, as shown in Fig. 6.8. It was worth noting that all rectangles contained no pedestrians, one pedestrian or several pedestrians. Also referred in Fig. 6.9, rectangles with no pedestrians were viewed as background objects like moving flag, vehicle, or tree branches.

Most detected rectangles from PETS2009_1 contained 0 to 4 pedestrians, as pedestrians in PETS2009_1 were usually sparse. Rectangles recognised from PETS2009_2 included more pedestrians, with numbers ranging from 0 to 10, as pedestrian groups were relatively dense in PETS2009_2. The pedestrian numbers of detected rectangles from Town Centre were distributed evenly as TownCentre contained many more frames than the previous two videos. When the annotation process was completed, all extracted features and pedestrian numbers were input into libsvm (Chang and Lin 2011) for training the classifier.

Table 6. 1 Detected rectangles with counts annotations for training classifier.

Video	TF	DR	PN0	PN1	PN2	PN3	PN4	PN5	PN6	PN7	PN8	PN 9	PN10
PETS 2009_1	792	3919	409	2446	801	230	26	5	1	0	1	0	0
PETS 2009_2	239	997	268	134	112	37	109	229	72	58	9	21	48
Town Centre	3750	45614	4500	24560	10549	1243	2690	1267	569	108	48	50	30

*TF denotes Training Frames, DR denotes Detected Rectangles, PNx denotes pedestrian number in rectangle is x

6.6.4 Comparison evaluation

(a) Pedestrian counting on PETS2009_1

(b) Pedestrian counting on PETS2009_2

(c) Pedestrian counting on TownCenter

Fig. 6.9 Pedestrian counts by proposed method on three videos (Please refer to Fig. 6.9 in appendix for high-resolution figures).

In this section, we assessed the performance of the proposed method against fastHOG (Prisacariu and Reid 2009) and CCMLF (Ryan, Denman et al. 2009), via testing it on the second halves of the three videos. In order to demonstrate the superiorities borne about by rectangles tracking, we also investigate the proposed method without rectangles tracking. Fig. 6.9 shows some sample frames and the counting results by our proposed method. The number at the top-left corner of each rectangle is its estimated pedestrian count. The number at the top-left corner of the image is the total estimated pedestrian count throughout entire image. Though PETS2009_1 (Fig.6.9 (a)) is usually sparse, it is still difficult as pedestrians walk in different directions. While PETS2009_2 (Fig.6.9 (b)) shows a scenario that pedestrians walk towards the same direction, the challenging part in this situation is that pedestrian groups are usually very dense. The tricky part of Town Centre (Fig. 6.9(c)) is that other disturbing objects such as pigeons and a trolley are present in the image.

Please note that CCMLF (Ryan, Denman et al. 2009) assume all detected blobs are pedestrians rather than to filter detected moving objects out from pedestrian blobs. Actually, this problem exists in a number of current pedestrian detection and counting methods. Background object discrimination was added into CCMLF by using the features and training method in CCMLF, and thus made the comparison experiments valid.

Ground truth was compared to the total estimated counts of our proposed method, of fastHOG, and of adaptive CCMLF, tested on the second halves of the three videos. Fig. 6.10, Fig. 6.11 and Fig. 6.12 depict the comparisons. Pertaining to the three figures, the red line describes ground truth pedestrian numbers per frame throughout the entire video. The blue, green, and black lines represent differences between estimated pedestrian counts and ground truth per frame throughout the entire video based on our proposed method, fastHOG and CCMLF, respectively. Fig. 6.10 illustrates the case of PET12008_1 video. The video is usually sparse in that it contains 1 to 10 pedestrians per frame. The proposed method (blue line) and fastHOG (green line) work much better than CCMLF (black line). In addition to this, our method is more stable than fastHOG. For example, between frame 150 and frame 700, fastHOG is prone to make mis-detections, which are usually caused by pedestrian overlay.

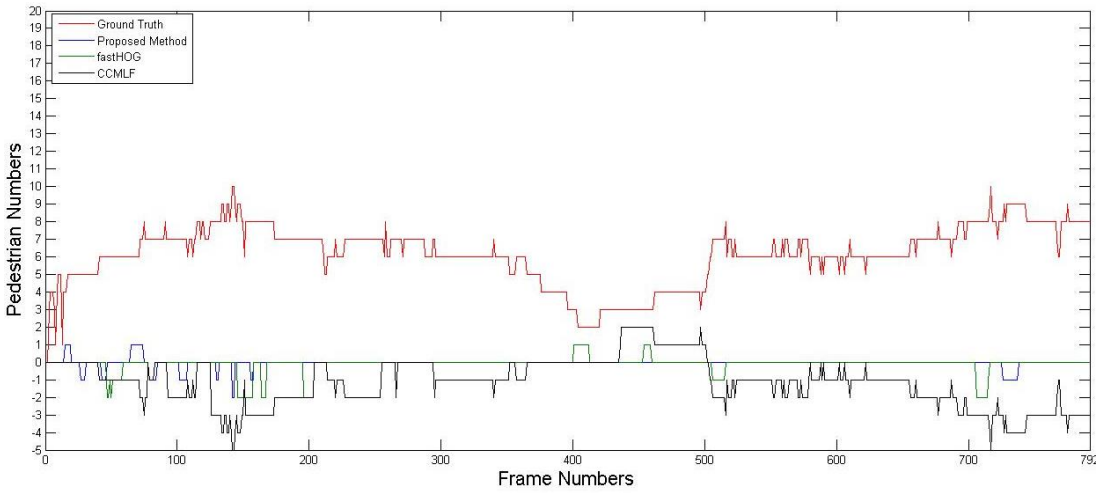


Fig. 6.10 Comparison evaluation of three methods by testing on PETS2009_1 (Please refer to Fig.6.10 in appendix for high-resolution figures).

Fig. 6.11 describes the case of PET12008_2 video, which usually contains pedestrian crowds of high density. The CCMCL (black line) is still unstable in this case. fastHOG (green line) usually misses pedestrians due to occlusions occurring, while the proposed method (blue line) works well (with exception as well, for example, around frame 90, when mistakes took place in the proposed method when the pedestrian densities of rectangles were too high). Fig. 6.12 depicts the case of TownCenter video, which contains a more complicated scenario than that of previous videos. CCMCL (black line) works poorly in this case. fastHOG (green line) performs well as the pedestrians usually walk sparsely but still misses pedestrians when occlusions happen. The proposed method works well on these tricky frames. However, a drawback of the proposed method should be noted in that it cannot deal well with stationary pedestrians. For example, around frame 1200, our method detects and counts the pedestrian correctly when he walked into the view of camera. But our method missed it after the pedestrian stopped on the centre of a street without any movement.

In order to demonstrate the improvement of pedestrian counting with rectangles tracking, we tested the proposed method with and without tracking on PETS2009_1 video. Fig. 6.13 shows the performances. The red line describes ground truth pedestrian numbers per frame throughout the entire video while the blue and green

lines represent the estimated pedestrian numbers by the proposed method with and without tracking, respectively. As shown, through the match, split and merge analyses, several mistakes were corrected by historical counts records because each pedestrian entering the video frame should walk out and disappear in that frame.

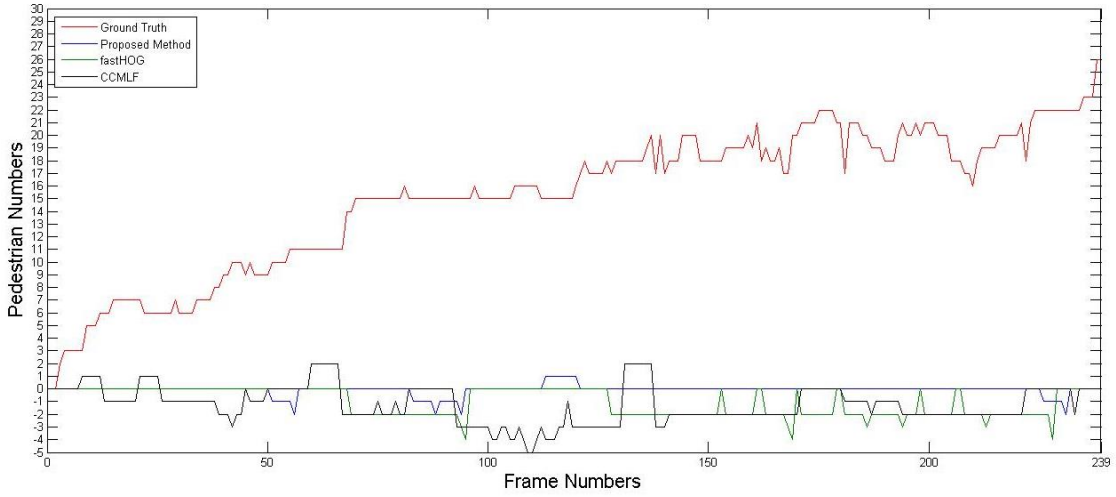


Fig. 6.11 Comparison evaluation of three methods by testing on PETS2009_2 (Please refer to Fig.6.11 in appendix for high-resolution figures).

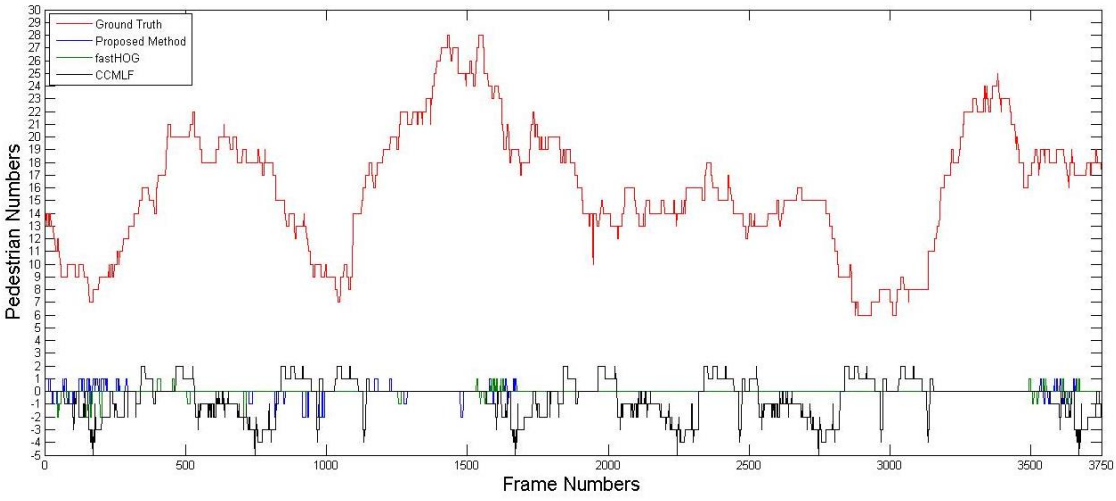


Fig. 6.12 Comparison evaluation of three methods by testing on TownCenter (Please refer to Fig.6.12 in appendix for high-resolution figures).

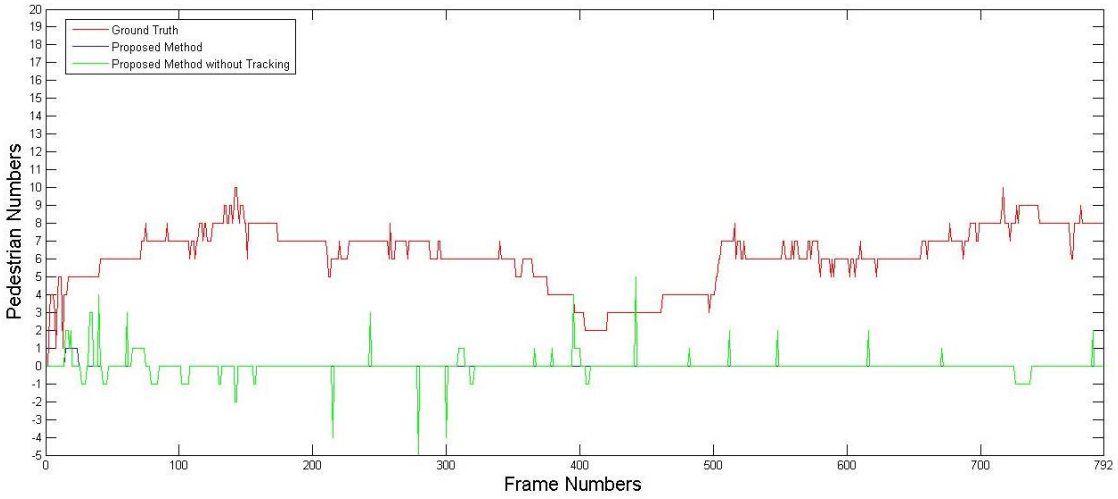


Fig. 6.13 Evaluation of counting improvement by adaptive tracking (Please refer to Fig.6.13 in appendix for high-resolution figures).

6.6.5 Robustness evaluation

Ideally, a promising method should guarantee robustness characterised by both low false alarm and mis-detection. In attempt to examine the robustness of our method, we compared the proposed method with fastHOG and CCMLF on the first 1000 frames of the TownCenter video. Table 6.2 shows the false alarm and mis-detection throughout the whole 1000 frames by these three methods. We should bear in mind that different false alarms and mis-detections for fastHOG and the other two methods were defined, as fastHOG was individual based while the other two were group based. For fastHOG:

$$false\ alarm = \frac{false\ pedestrian\ detection}{total\ pedestrian\ detection},$$
$$miss\ detection = \frac{miss\ pedestrian\ detection}{total\ pedestrian\ detection} \quad (6.12)$$

For proposed method and CCMLF:

$$false\ alarm = \frac{false\ rectangle\ detection}{total\ rectangle\ of\ pedestrian\ detection},$$
$$miss\ detection = \frac{miss\ rectangle\ detection}{total\ rectangle\ of\ pedestrian\ detection} \quad (6.13)$$

As shown in Table 2, our proposed method and fastHoG were similar in terms of false alarms (8.0% and 7.2%), which were much lower than that of CCMLF (16.3%). It indicated that the proposed method or fastHOG was more capable of discriminating pedestrians from other objects, because CCMLF did not discriminate other objects

from intended objects(pedestrians)after segmenting foreground On the other hand, both the proposed method and CCMLF had a lower mis-detection rate (2.7% and 3.8%) as they utilised similar blob detection methods. In contrast, fastHOG failed to detect pedestrians (17.5%) when occlusion happened.

Table 6. 2 Robustness evaluation regarding false alarms and mis-detection.

Rectangles	Pedestrian	Method	False alarm	Miss detection
11033	15121	Proposed	8.0%(1212/15121)	2.7%(401/15121)
11087	15890	CCMLF	16.3%(2590/15890)	3.8%(605/15890)
N/A	14629	fastHOG	7.2%(1056/14629)	17.5%(2560/14629)

6.6.6 Computational cost

In this section, we examined the running time of the proposed method. Results can be seen in Table 6.3. Processing speed was given in frames per second (fps). Processing speed was faster for our method than that of the other two methods. For all three methods, processing speed on TownCenter was faster than those on the other two videos using the same method, as TownCenter had a higher definition. Though both our proposed method and fastHOG were implemented on CUDA framework; fastHOG was slower as its siding-window technique required a substantially higher computational load. The processing speeds of our method on the three videos were 40fps, 42fps and 35fps, which were considerably faster than video playing speed (25fps). Compared to other methods, our method can work satisfactorily in real-time while other methods cannot be applied in real-time, since their processing speeds on three videos were slower than videos' playing speed.

Table 6. 3 Computation cost evaluation by testing on three videos.

Video	Proposed	fastHOG	CCMLF
PETS2009_1(768*576 at 20fps)	40 fps	11fps	14fps
PETS2009_2(768*576 at 20fps)	42fps	7fps	15fps
TownCentre(1920*1080 at 25fps)	35fps	3fps	1fps

6.6.7 Speedup demonstration

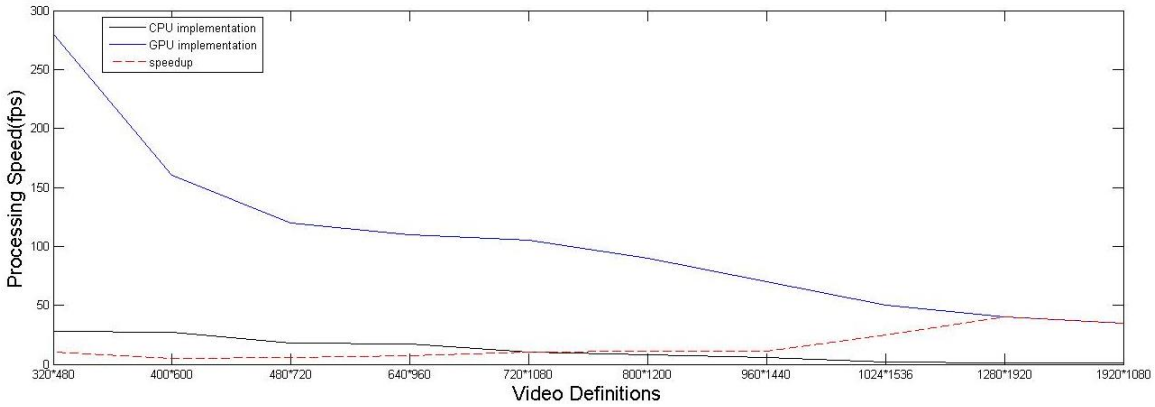


Fig. 6.14 Evaluation of Speedup by CUDA implementation.

Finally, we attempted to interpret the superiorities generated by CUDA implementation. The Town Center video was employed to test the speedup of the CUDA implementation over various frame resolutions. We down-sampled frames of TownCenter to get 10 videos in different resolutions: 320×480, 400×600, 480×720, 640×960, 720×1080, 800×1200, 960×1440, 1024×1536, 1280×1920, and 1920×1080. At each resolution level, we ran both methods with and without CUDA implementation, measured the processing speed and speedup correspondingly. As shown in Fig. 6.14, CPU implementation works well on extremely low-resolution videos (320×240 and 640×480) while accompanied by real-time processing speed (25fps). However, the processing speed of CPU implementation declined to 5fps when the video resolution increased to 1920×1080. On the other side, for CUDA implementation, the processing speed was greater than 25fps even if the highest-resolution video (1920×1080) was applied. The red line shows our CUDA implementation has at least a 10-times speed increase over traditional the CPU method.

6.7 Conclusion

Utilising the feature selection strategy, we have presented a novel method that estimates the numbers of pedestrians in surveillance videos. In order to best encode the characteristics of pedestrians in the moving blobs, we choose 9 maximum-dependent features. We first extract moving objects from the background per frame. Particularly, based on extracted hybrid features from each moving object, each object is classified into 0 to 10 pedestrians. Meanwhile, a traditional tracking method optimised by a novel analysis is proposed to improve the estimated pedestrian counts.

The total pedestrian number per frame is the summation of pedestrian counts of detected rectangles. In addition, CUDA implementation of our method provides at least a 10-times speed increase over traditional CPU methods. These aforementioned improvements enable our method to work well in real-time applications such as intelligent traffic systems and public place surveillance. The effectiveness of the proposed method using our feature selection methodology is demonstrated by comparison with other two methods on three benchmark surveillance videos.

Although we have demonstrated the stability and effectiveness of the pedestrian counting method in this chapter, our method cannot avoid certain limitations. First, our method was robust to limited camera angle changing (approximately 10 degrees) and variations from changing the camera-to-ground distance (approximately 1 metre). However, when the cameras are installed differently, we will need to collect new training data and train a new classifier then. Second, the method presented here would be more effective for scenes such as shopping malls, college campuses, and streets, where pedestrians walk as individuals or in social groups. Last, the proposed method was not intended for use in an environment that has immense crowds of people, such as mob scenes and political rallies, in which people cannot be segmented into individuals or groups.

Chapter 7 Feature Selection for Intelligent Vehicle

The content in this chapter has been published in (Peng, Xu et al. 2012) (Peng, Jin et al. 2012) (Peng, Xu et al. 2011).

The paper "3D Pose Estimation of Front Vehicle Towards a Better Driver Assistance System" was awarded Best Paper in 2012 IEEE International Conference on Multimedia and Expo Workshops (ICMEW).

7.1 Background

In last two chapters, we validate the effectiveness of our feature selection strategy in the two cognitive systems between surveillance cameras and the two main road-users—vehicles and pedestrians. In this chapter, we implement feature selection in a more complex cognitive system-intelligent vehicle. In this cognitive system, the camera is not fixed and background changes quickly. Moreover, this system strictly requires real-time processing.

Vehicle crashes occur every minute around the world. This makes vehicle collisions the leading cause of severe injuries worldwide, according to the report of the World Health Organization. With the aim of reducing the number of injuries and accident severity, crash-prevention systems is becoming an area of active research among automotive manufacturers, suppliers and universities. An on-board driver assistance system aiming to provide the driver with a 3D position of front vehicles is very attractive. Besides preventing collisions, the accurate position of the front vehicle can help a driver to make the right decisions on the road. This task includes two steps: front vehicle detection and 3D position calculation.

Vision-based vehicle detection has received considerable attention over the last 20 years. As shown in Fig. 7.1, these applications are grouped into two general categories, depending on the installation location of the camera: one is a fixed camera which is installed roadside; the other one is a camera mounted on a vehicle. For vehicle detection with a roadside camera, many vehicle detectors utilise background

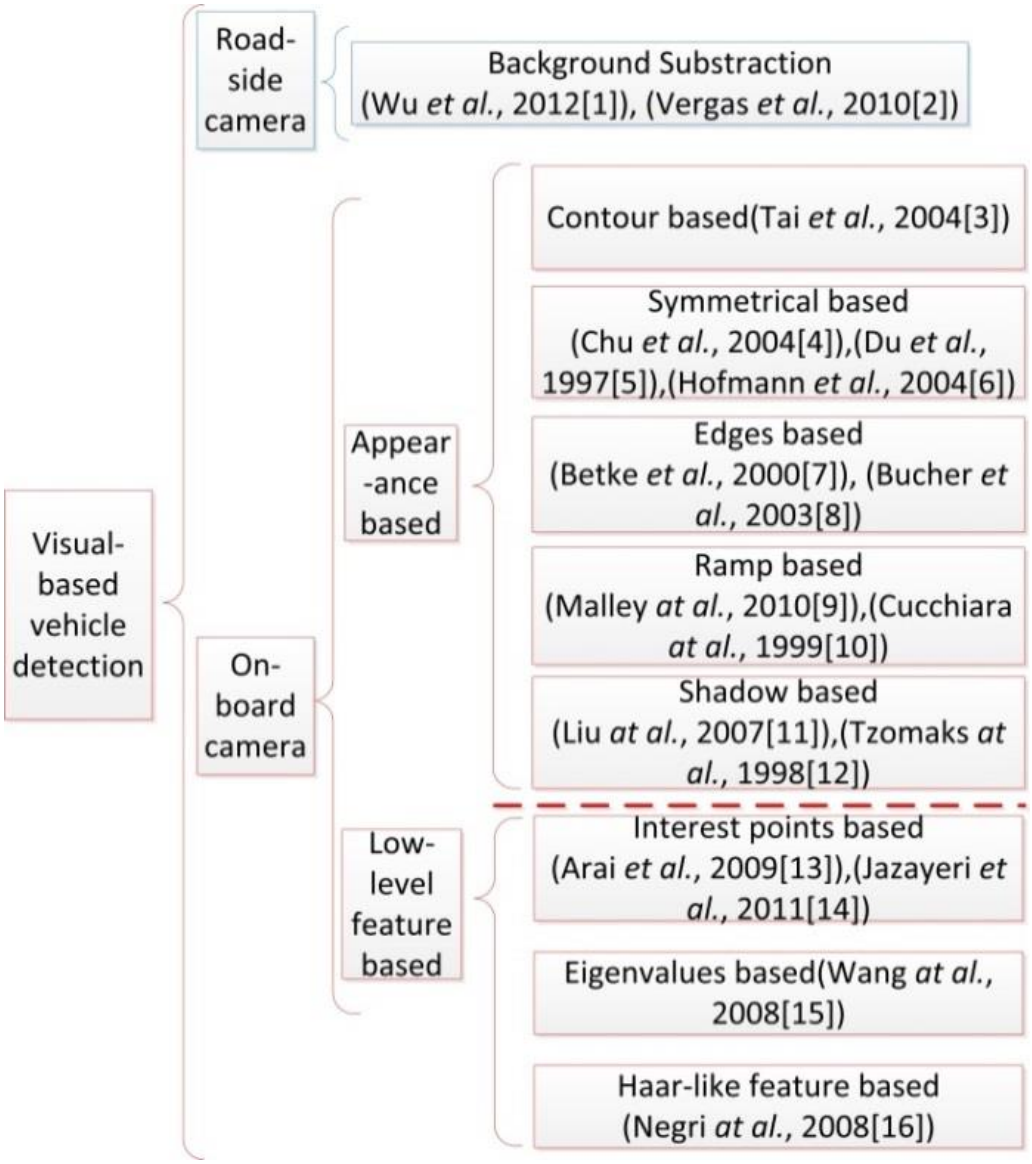


Fig. 7.1 Current vehicle detection methods.

subtraction methods (Vargas, Milla et al. 2010). Wu et al. (Wu and Juang 2012) proposed a grey-level differential value method to dynamically segment moving objects from the background. This method rests on the assumptions that road surfaces are grey, lane marks are yellow or white and that the remaining colours are to be regarded as moving objects on the road. Vargas et al. (Vargas, Milla et al. 2010) integrated a background subtraction algorithm with a sigma-delta filter, which has high computational efficiency. The proposed method attempts to achieve a background updating model at the pixel level by introducing a confidence measurement for each pixel.

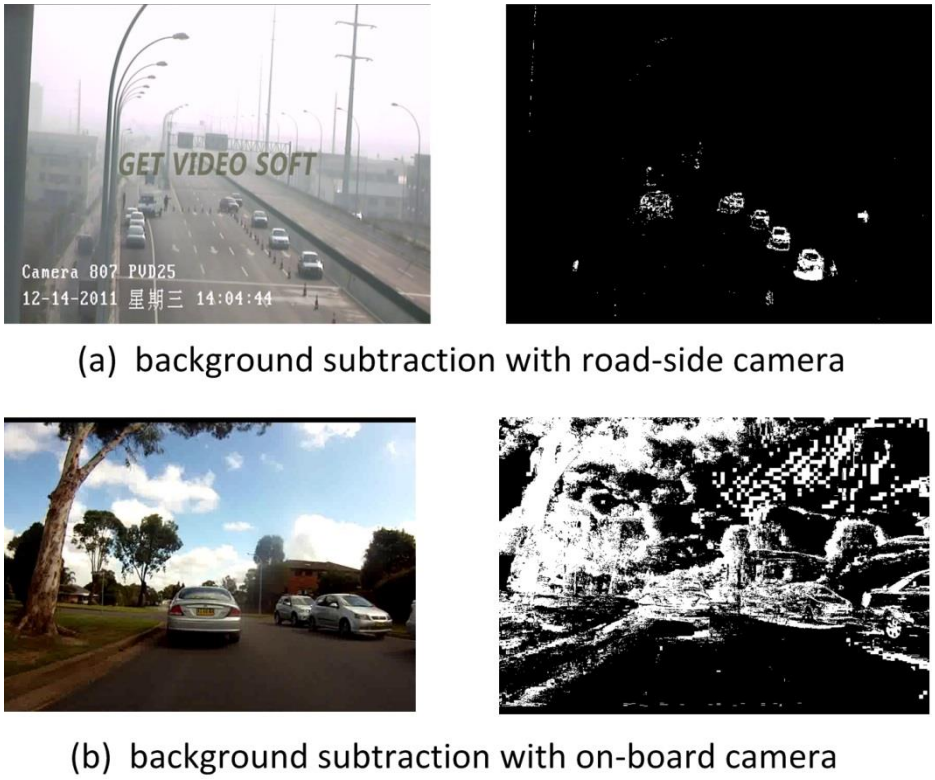


Fig. 7.2 Comparison between background subtraction techniques with a road-side camera and an on-board camera.

For vehicle detection with an onboard camera, however, background subtraction cannot be easily used because prior knowledge about the background is not available. For example, we achieve good results - as shown in Fig. 7.2 (a) - with a road-side camera using the background subtraction algorithm in (Vargas, Milla et al. 2010). However, this result cannot be obtained when we use the same algorithm with an onboard camera - as shown in Fig. 7.2 (b) - because the background is not static. The processing speed is extremely critical in onboard vehicle detection systems, since the prompt feedback from such a system could save time for a driver's reaction. Various approaches with low computational costs have been proposed in the literature, which can be classified into one of the following categories: appearance-based analysis (Betke, Haritaoglu et al. 2000) (Bucher, Curio et al. 2003) (Chu, Ji et al. 2004) (Cucchiara, Piccardi et al. 2000) (Du and Papanikolopoulos 1997) (Hoffman, Dang et al. 2004) (O'Malley, Jones et al. 2010) (Tai, Tseng et al. 2004) (Tzomakas and Seelen 1998) (Wei, XueZhi et al. 2007) and low-level-based features (Arai, Inoue et al. 2010) (Wang and Lien 2008) (Jazayeri, Hongyuan et al. 2011). Tai et al. (Tai, Tseng et al.

2004) used an active contour method with a Kalman filter to detect and track vehicles. As demonstrated in their research, the vehicles could be easily tracked with a low computation loading. However, the contour initialisation posed a critical risk. Chu et al. (Chu, Ji et al. 2004), Du et al. (Du and Papanikolopoulos 1997) and Hofmann et al. (Hoffman, Dang et al. 2004) reported that vehicles rear or frontal views are generally symmetrical in both horizontal and vertical directions. However, an important issue arises when computing symmetry from image intensity in that symmetry is quite prone to false detection, such as symmetrical background objects or partly occluded vehicles. It was observed that the rear or frontal views of vehicles usually contain many horizontal and vertical structures, such as rear windows and bumper; thus, Betke et al. (Betke, Haritaoglu et al. 2000) proposed a coarse-to-fine method to detect distant cars via searching for rectangular objects. A refined search was activated only for small regions on the image, after a whole image search. Within a predefined maximum distance from the detected road lanes, Bucher et al. (Bucher, Curio et al. 2003) found vehicle candidates with edge features by scanning the image from the bottom up to a certain vertical position, line-by-line. Although fast and positive results were made and the method appeared very attractive, its performance and robustness strictly depended on well-tuned parameters, such as the thresholds for detecting edges and choosing the most important vertical and horizontal edges. In addition, the use of vehicle lights as another clue for vehicle detection was studied. Malley et al. (O'Malley, Jones et al. 2010) detected and tracked vehicles by segmenting rear-facing lamps based on a red-colour threshold. Meanwhile, Cucchiara et al. (Cucchiara, Piccardi et al. 2000) employed a morphological analysis to detect vehicle light pairs in a narrow inspection area. This kind of method was very sensitive to illumination, where promising results could only be obtained at night. Shadow information as a sign pattern for vehicle detection was investigated in (Tzomakas and Seelen 1998) (Wei, XueZhi et al. 2007). Liu et al. (Wei, XueZhi et al. 2007) distinguished vehicle candidates using the shadow underneath a vehicle. Tzomaks et al. (Tzomakas and Seelen 1998) analysed the grey level around the detected lanes to segment shadow and then found a vehicle. However, there is no systematic way to choose the appropriate threshold values for shadow segmentation.

Texture patterns have also been used for vehicle detection. These texture patterns were usually presented by low-level features, such as feature points (Jazayeri,

Hongyuan et al. 2011), eigenvalues (Wang and Lien 2008) and Haar-like features (Negri, Clady et al. 2008), etc., rather than the structure features described above. Arai et al. (Arai, Inoue et al. 2010) proposed a vehicle detection system based on the shift of a feature plane, which was constituted by feature points on the front surface of the vehicle. They found the feature plane of the vehicle's front surface shifts in accordance with an affine transform. Jazayeriet et al. (Jazayeri, Hongyuan et al. 2011) extracted low-level features, such as corners, intensity peaks and horizontal line segments from images. These features were profiled to the temporal space. To identify tracked features, such as a car or a background, they estimated probability distributions for the motion properties of cars and the background. The Hidden Markov Model (HMM) was used to separate vehicles from the background and track them probabilistically. At the same time, a statistical model was used in (Wang and Lien 2008), performing vehicle detection by Principal Component Analysis (PCA). Negri et al. (Negri, Clady et al. 2008) compared the performances of different vehicle detectors. In their study, these detectors were trained from Haar-like features, a histogram of oriented gradient features and a fusion of the two feature sets, respectively. Subsequently, the best performance was achieved by the feature fusion. The low-level feature-based methods usually consist of feature extraction and classifier training. It is worth mentioning that in view of the various appearances of vehicles it is normally extremely difficult to construct explicit models.

Another interesting method was raised by Arrospide et al. in (Arrospide, Salgado et al. 2010). They found a vehicle via ground plane detection. The proposed method is based on the reliable estimation of the homography between ground planes in successive images. The homography calculation is grounded on a linear estimation framework, which predicates the ground plane transformation matrix while it is dynamically updated with new measurements. Disappointingly, the results showed the speed of the method to be only around 10fps, with an image resolution of 360 ×288, as implemented on a 2GHz 2GB Memory PC.

To the best of knowledge - with notable exceptions like (Bensrhair, Bertozzi et al. 2002) (Broggi, Caraffi et al. 2005) (SamYong, Se-Young et al. 2005) (Zielke, Brauchkmann et al. 1992) most of the previous vehicle detection works have been 2D. Even in (Sun, Miller et al. 2002), a proposed pre-crash system was based on vehicle rear detection in a 2D rather than a 3D space. With an onboard camera, Zielke et al.

(Zielke, Brauchkmann et al. 1992) measured the distance from front vehicles under a Time-to-Collision (TTC) model. This TTC model was represented by the distance between two images points on the rear surface of the front vehicle and the rate of the changing rate for this distance. However, this proposed method strictly rested on the assumption that the vehicle with the camera is moving up to the front vehicle rear vertically, which cannot be guaranteed in reality. Besides this, the stable detection and tracking of the two points in the marker-less scene was very difficult. Broggi et al. (Broggi, Caraffi et al. 2005) and Bensrhair et al. (Bensrhair, Bertozzi et al. 2002) detected and localized a front vehicle using stereo vision. With known stereo rig parameters, a 3D map of the viewed scene can be constructed via the differences in the corresponding pixels between left and right images. Though this method can obtain accurate measurements, it is extremely time-consuming. Kim et al. (SamYong, Se-Young et al. 2005) used a sonar sensor for vehicle detection and distance estimation. Although these non-visual sensors can measure distance directly, without requiring powerful computing resources, they have several drawbacks, such as a high cost, a low spatial resolution and a slow scanning speed. Furthermore, visual information is very important in a number of related applications, such as lane detection and traffic sign recognition.

In this chapter, our proposed approach recognises and tracks vehicle rears quickly based on license plate localisation. Then, a 3D pose is estimated with respect to the extracted vehicle rear. This has several advantages: 1, a license plate (LP) is smaller and much more standardised than a vehicle rear, making LP localisation quicker and more robust than directly detecting a vehicle; 2, because only the region around the LP rather than the whole vehicle is required for 3D pose estimation, we are able to achieve real-time performance; 3, pose measurement is not affected, even when the extracted regions of a vehicle's rear in successive frames are not exactly the same, since vehicle's rears are considered as planar.

The remainder of this chapter is structured as follows: we begin by reviewing the related work about 3D pose estimation from a planar object in Section 7.2. In Section 7.3 we present an overview of our own system. Subsequently, Section 7.4 overviews the feature selection scheme guides the map initialisation and maintenance. Section 7.5 interprets the process of map initialisation using extracted localised features. Next, vehicle rear detection, tracking and pose estimation are described in Section 7.5. In

Section 7.6 we present map updates and maintenance in detail. The experimental results are demonstrated in Section 7.7. Finally, we conclude in Section 7.8.

7.2 Related work about 3D pose estimation from a planar target

There are several approaches for pose estimation, where the 6 degrees of freedom of a camera's pose are calculated from correspondences between images and the real scene structure. Most of them work with the theory that the pose of a calibrated camera can be uniquely estimated by no less than four coplanar and no collinear points. Depending on how correspondences between images and the real world should be established, these methods could be divided into two categories: prior knowledge-based methods and self-initialisation-based methods.

With the prior model, the registration between an image and the real world can be performed directly. The camera pose can then be estimated from these corresponding points (Eade and Drummond 2009) (Kawano, Ban et al. 2003) (Mondragón, Campoy et al. 2010). Drummond et al. (Eade and Drummond 2009) found camera poses that correctly re-project some fixed features of a prior 3D model into the 2D image. These features can be edges, line segments or points. Through the least-squares minimisation of an error function, the best pose was found. Actually, a comprehensive prior model is not readily available. Some researchers have established a relationship between images and the real world using fiducial markers. For instances, Kawano et al. (Kawano, Ban et al. 2003) discussed a number of planar markers for Augmented Reality (AR). The salient markers with known pose information in the real world were easily detected in the images. Meanwhile, Mondragón et al. (Mondragón, Campoy et al. 2010) utilised 3D pose estimation techniques in Unmanned Aerial Vehicle (UAV) control. Specifically, their proposed method asked a UAV driver to select four points on the image that correspond to four corners on the helipad.

Normally, 3D pose estimation implements a previously unknown scene without any known models. This problem was solved in (Klein and Murray 2007) (Mouragnon, Lhuillier et al. 2006) by building an initial map from a five-point stereo (Stewénius, Engels et al. 2006). Mouragon et al. (Mouragnon, Lhuillier et al. 2006) tracked a camera using local bundle adjustment over the most recent camera poses and obtained accuracy over a long distance. Klein et al. (Klein and Murray 2007) established a

small AR workspace where the user will spend most of their time. As such, they built a long-time map in which features were constantly re-visited. This is similar to our own case, in which we focus on the extracted vehicle rear region. User cooperation is required in (Klein and Murray 2007) for map initialisation; however, it would be unsafe to ask a driver to manipulate this. In our method, the initial map is constructed with feature points tracked on two extracted vehicle rear regions at the beginning.

7.3 Method overview

As shown in Fig. 7.3, our system consists of two threads: tracking and mapping. Although these two threads are intimately linked, most of the time they perform in parallel in order to save processing time. For a mapping thread, as shown in Fig.7.3, and given the assumption that vehicle rears are detected and tracked on frames, a map is initialised from feature point correspondences found in the first two keyframes. The map consists of a collection of point features with their 3D information. Each map point has a coordinate in the world coordinate system, references to the source keyframe and to the patch source pixels. The map is updated by keyframes, rather than frame by frame. These map points can be considered as a "bridge", which relate image points to the real world.

For a tracking thread, and with the assumption that the map has been already initialised, video is captured by the camera mounted on the front of the vehicle. Because we only need information of the front vehicle rather than the whole image, we only process a Region of Interest (ROI), as shown in Fig.7.4. The size of the ROI in our experiment is 640×480 pixels. A rejection cascade of an AdaBoost classifier with line segment features and Haar-like features (Peng, Xu et al. 2011) is utilised in order to find a set of LP candidates quickly. A best detection is obtained, followed by non-maximal suppression. Based on a localised LP, a vehicle rear is extracted. FAST (Features from Accelerated Segment Test) corners (Rosten and Drummond 2006) are then detected on the extracted vehicle rear. After finding correspondences between the detected features and map points, the relative pose of the camera to the map is calculated.

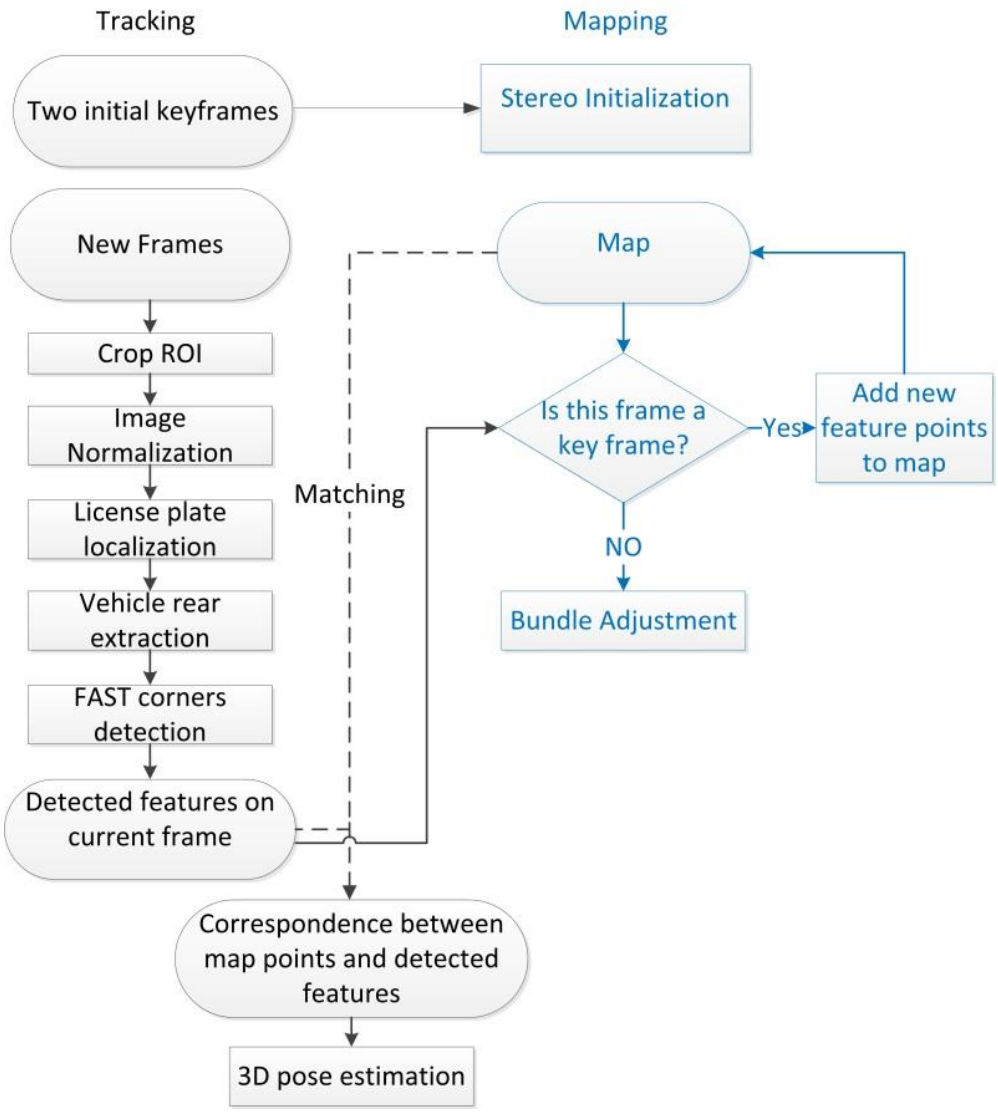


Fig. 7.3 Method overview of our 3D position estimation system.

Fig. 7.4 ROI on the whole frame.

7.4 Feature selection scheme for building reference map

The map is the most core part in this system. It is the "bridge" between recorded images and real world. Though we have demonstrated the combination of LSF and Haar-like in localising license plate in Chapter 5, these features are not suitable for the map initialisation and map maintenance in this system. We need features that can be processed extremely fast and represent the vehicle rear in sufficient detail. To achieve maximum dependency, FAST corner is utilised by us. According to the scheme of minimum redundancy, we use bundle adjustment to acquire the optimal set of features.

7.5 Map initialisation with selected features based on maximum dependency

As indicated in (Mouragnon, Lhuillier et al. 2006), the motion between two consecutive frames must be large enough to compute the epipolar geometry. For map initialisation, we select two frames at the beginning, relatively far from each other but with enough matched points. The first extracted vehicle rear is selected as a keyframe *K*. In the next *N* detected vehicle rears, we choose the second keyframe that is furthest from *K* and with the most matched interest points with *K*. In our experiments, we choose the shortest distance, which is 20 pixels, and the least number of matched points, which is 100.

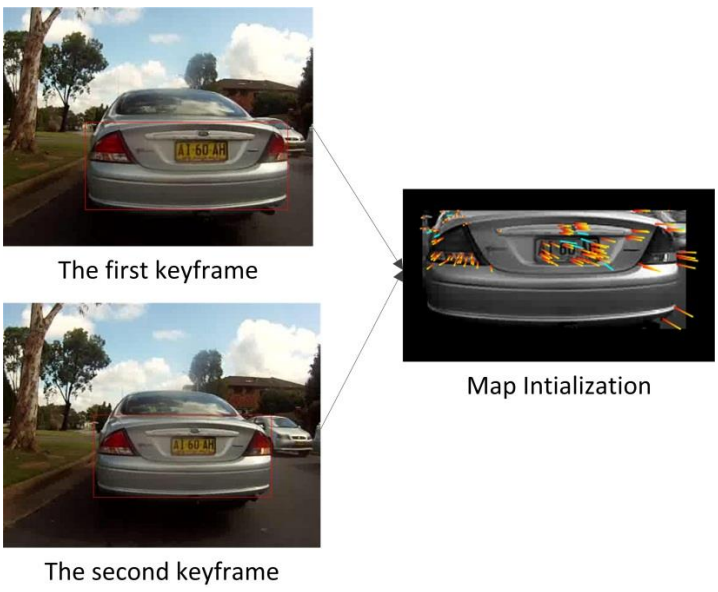


Fig. 7.5 Map initialisation.

As shown in Fig. 7.5, after obtaining the first two keyframes, map initialisation is done with a five-point algorithm (Arrospide, Salgado et al. 2010) and RANSAC (Hartley and Zisserman 2003). First, the essential matrix E between the two keyframes is calculated from a sample of 5 point correspondences:

$$\begin{cases} q_1^i E q_2^i = 0, i \in \{1 \cdots 5\} \\ EE^T E - \frac{1}{2} trace(EE^T)E = 0 \end{cases} \tag{7.1}$$

Where q_1^i and q_2^i are the 5 points projections on the two keyframes, respectively. At most, 10 solutions for E could be obtained from (1). Each E produces a solution for the camera's pose. We first discard the poses for which at least one of the 5 points is not reconstructed in front of the camera. Next, the remaining solutions for the camera pose are filtered with a RANSAC approach: a sample set of five points are selected to hypothesise a number of cameras' poses while the remaining matched points are tested for consensus. The best camera pose is chosen by computing the re-projection error over all the possible camera poses for all of the remaining matched points and keeping that with the highest number of matches. A map coordinate is constructed in such a way that the X axis and the Y axis are on the vehicle rear plane, while the Z axis is perpendicular to the vehicle rear plane.

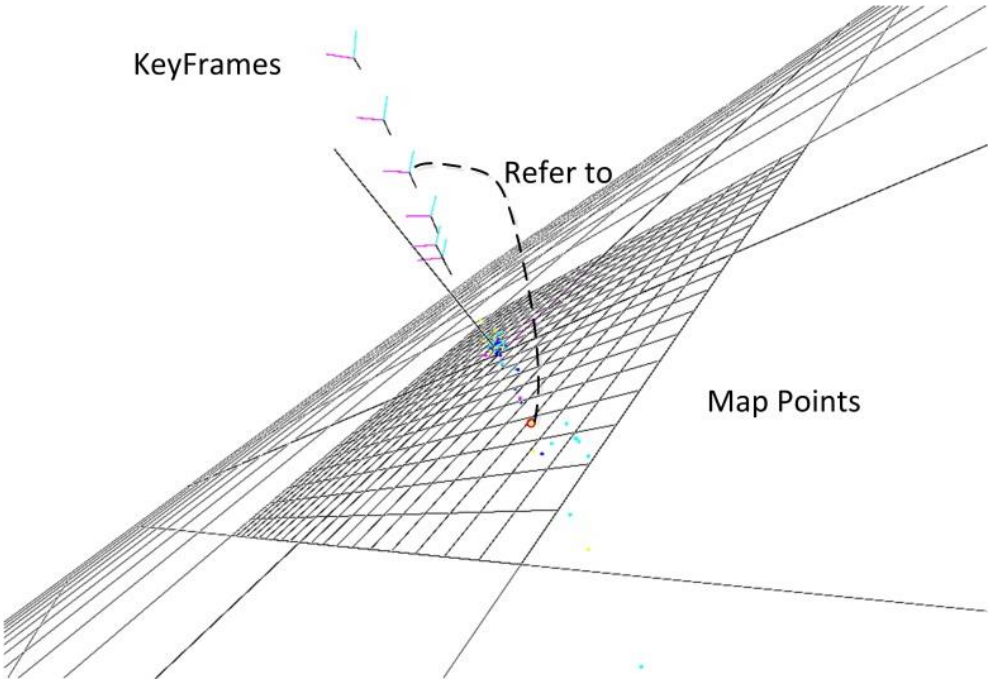


Fig. 7.6 Constructed map and keyframes.

As shown in Fig. 7.6, each map point refers to a single source keyframe, where the map point first takes place. For example, the map points generated in the procedure of map initialisation refer to the first keyframe. The relative 3D pose between the map point and its source keyframe are also recorded. Furthermore, each map point refers to its absolute 2D location on the source keyframe. FAST corners, as image features, are detected in our method as 8×8 pixel squares on greyscale frames. The centres of these pixel patches are recorded as the absolute 2D locations. Finally, because the sizes of the extracted vehicle rears change frame by frame, the relative 2D location of the map point on the source keyframe is also recorded , $\left(\frac{x}{W_v}, \frac{y}{H_v}\right)$, where (x, y), is an absolute 2D location of the map point on the source vehicle rear frame and (W_v, H_v), is width and height of the extracted vehicle rear.

7.6 Vehicle rear detection, tracking and 3D pose estimation

This section describes the procedure for vehicle rear detection, tracking and 3D pose estimation. In order to allow the reader to follow the idea of our method more easily, we present a procedure for map initialisation initially in the last section – which actually works in parallel with steps in this section. Therefore, the contents in this section build on the assumption that a map of 3D points has already been constructed.

7.6.1 Vehicle rear detection

We utilised the combination of line segment features and Haar-like features to detect license plate fast, which has been introduced in Chapter 4. Since we consider the vehicle rear as a plane, only a part of the vehicle rear is required for 3D pose estimation. We extract a vehicle rear as $W_v = 4 \times W$ and $H_v = 3 \times H$ where W_v and H_v are the width and the height of the vehicle rear and W and H are the width and the height of the localised LP, respectively.

7.6.2 Update search region for the license plate

The region prediction of the LP in the next frame is important. We use an alpha-beta filter to predict the possible region for the LP. It resembles a Kalman filter but is less complex and has less parameters to tune, having only alpha and beta values. The alpha controls the response to a new pose input while the beta controls how

responsive the filter is to a new velocity input. The alpha and beta gains range from [0, 1]. The alpha-beta filter update is as follows:

$$\begin{cases} r_t = x_t - \hat{x}_{t-1} \\ \hat{x}_t = \hat{x}_{t-1} + \alpha r_t \end{cases} \tag{7.2}$$

where $\hat{x}_t$ is the current smoothed out pose estimation, $\hat{x}_{t-1}$ is the previous estimation, x_t is the pose input, r_t is the residual and α is the gain.

7.6.3 Establishment of the correspondence between map points and frame points

For camera pose estimation, we need to find interest points on the current frame and match them up with map points. As described in the section on map initialisation, after transferring the extracted vehicle rear to greyscale, we run the FAST corner detector on it. Each FAST feature vector describes an 8×8 image patch. To match the map points with the detected points on the current frame, we use a k-means tree. The match is performed in a binary tree by comparing which centroid is closer to the query and going down the tree. The final leaf node will contain a handful of features that have to be searched linearly. Some matches will not be one to one correspondences when more than one FAST feature points to the same map point. To resolve this conflict, the FAST feature with the highest matching score is kept. To promote match accuracy, we perform a fix-range image search on the current frame. For every map point, we search on the current vehicle rear as:

$$\begin{cases} search\ center = (\frac{x}{W_v} W_v', \frac{y}{H_v} H_v') \\ search\ patch = \frac{1}{8} W_v' \times \frac{1}{16} H_v' \end{cases} \tag{7.3}$$

where $(\frac{x}{W_v}, \frac{y}{H_v})$ is described in the section of map initialisation as the relative location of the map point on the source keyframe and W_v' and H_v' are the width and height of the current vehicle rear, respectively.

7.6.4 The camera project model and pose estimation

To estimate camera pose, we should understand that the camera model points in the world coordinate are projected onto an image frame after being projected onto a camera-centred coordinate. The following equation describes the projection onto the camera-centred coordinate:

$$\begin{cases} P_{ic} = E_{cw} P_{iw} \\ E_{cw} = -R^T T \\ T = [t_x, t_y, t_z],\ R = [\theta, \psi, \phi] \end{cases} \tag{7.4}$$

where P_{iw} is a point in a world coordinate, P_{ic} is a transformed point in a camera-centred coordinate, E_{cw} is a 3×3 transform matrix that contains a translation T and a rotation R.

To transfer the point P_{ic} on a camera-centred coordinate to a point P_i image, the camera's intrinsic parameters are required. We obtain these parameters by camera calibration. A calibrated camera projection model is as follows:

$$\begin{cases} CamParas = [f_x, f_y, x_0, y_0, \omega] \\ \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{bmatrix} f_x & 0 \\ 0 & f_y \end{bmatrix} \frac{r'}{r} \left(\frac{X \times Y}{Z^2}\right) \\ r = \sqrt{\frac{x^2 + y^2}{z^2}},\ \ r' = \frac{1}{\omega} \tan^{-1}(2r \tan\frac{\omega}{2}) \end{cases} \tag{7.5}$$

where (f_x, f_y) is a camera focal length, (x_0, y_0) is a principal point, ω is a distortion parameter, (x, y) is a point P_i on an image and (x, y, z) is a point P_{ic} in a world coordinate.

For 3D pose estimation, we need to find values for R and T that minimise a re-projection error function:

$$E(T, R) = \sum_{i=1}^{n} \left\| \left(I - \frac{P_i {P_i}^T}{{P_i}^T P_i}\right) P_i \right\| \tag{7.6}$$

where n is the number of matched points on the current frame. We use the algorithm in (Schweighofer and Pinz 2006) to get a unique solution from the above equation.

7.7 Map update and maintenance

The initial map contains only two keyframes and a set of interest points. As the relative pose between a front vehicle and a camera changes accordingly, new keyframes and map features are added into the system, to let the map grow. When no new keyframes or features are added, we optimise the map using a bundle adjustment.

7.7.1 Map update

In section 5.3, we estimated the camera pose using the correspondence between map points and features on the current frame. After matching features on the current frame with map points, some features on the current frame may be leftover. When the tracking quality is good and the remaining non-near features are greater than 20, we add the current vehicle rear image as a new keyframe and each left feature is a candidate to become a new map point. We define the tracking quality as being good when more than 40 features are detected on the current vehicle's rear. At the same time, we define the remaining non-near features as those not in 16×16 patches around the centres of the successful matched points.

New map points require 3D information. This is not available from a single keyframe and so triangulation with another view is required. The closest keyframe already existing in the map is selected as the second view. The pixel patches around the FAST corners I_1 that lie along the epipolar line in the second view are compared with the candidate map points I_2 using the zero-mean sum of squared differences, as follows:

$$\sum_{(I_1,I_2)\in patch}(I_1 - {I_1}' - I_2 + {I_2}') \tag{7.7}$$

The feature on the second keyframe with the lowest sum of squared difference is selected for the candidate map point.

7.7.2 Map maintenance based on minimum redundancy

The map contains several keyframes associated with a set of map points. Using bundle adjustment, we optimise the map by simultaneously refining camera poses and map points through adjusting the associated respects of the 3D structure and the viewing parameters at the same time. In our system, bundle adjustment boils down to minimising the re-projection error in equation (7.6), with respect to the camera poses $u = (T, R)$ and map points P_{iw}. The operation is as follows:

$$\min_{u,P_{iw}} \sum_{i=1}^{N} \left\| (I - \frac{P_i {P_i}^T}{{P_i}^T P_i}) P_i \right\| \tag{7.8}$$

Where N is the number of feature points in the map. The minimisation is achieved using the Levenberg-Marquardt algorithm (Moré 1978), which implements an

effective damping strategy that lends it the ability to converge quickly from a wide range of initial guesses.

7.8 Experiment

In order to demonstrate the performance of the system described above, we evaluated our system with respect to five aspects: LP localisation, feature detection and mapping, real-time 3D pose estimation, map optimisation and lost tracking analysis. The performance of LP localisation has been validated in Chapter 5. So we don't repeat the experiment in this Chapter. We also discuss the degree of compliance with each stage of the proposed method as well as the limitations of the method. The experiments were implemented with a desktop PC with Intel™ Core™ 2 Duo CPU, E8400 3.00GHz, RAM 2GB. The videos were recorded by a camera, which was mounted on the front of a car, as shown in Fig. 7.7. We used a HD Motorsports HERO camera in the experiment. The recorded video is made with a high-resolution of 1920×1080 pixels, 30FPRS

Fig. 7.7 The camera mounted on the front of the car.

7.8.1 Camera calibration

As explained above, we need camera parameters in order to estimate the 3D pose of vehicle rear. These camera parameters include focal length, principal point and lens distortion. To obtain these parameters, we calibrated our camera with a chessboard, the corners of which are very easy to find and the geometry of which is very simple. We recorded a video containing different views of a chessboard. After 9×6

chessboard corners were found on the frames, the parameters of our camera were calculated as:

$$I=\begin{bmatrix} 4.35034 & 0 & 2.96018 \\ 0 & 4.69138 & 2.31044 \\ 0 & 0 & 1 \end{bmatrix}$$
$$D^T=[4.89049e-001,-3.04420e-001,-1.58325e-001,-7.42063e-002] \quad (7.9)$$

where I is an intrinsic matrix representing the focal length and the principal point and D^T is the matrix representing lens distortion. The camera parameters are metric.

7.8.2 Feature detection and mapping performance

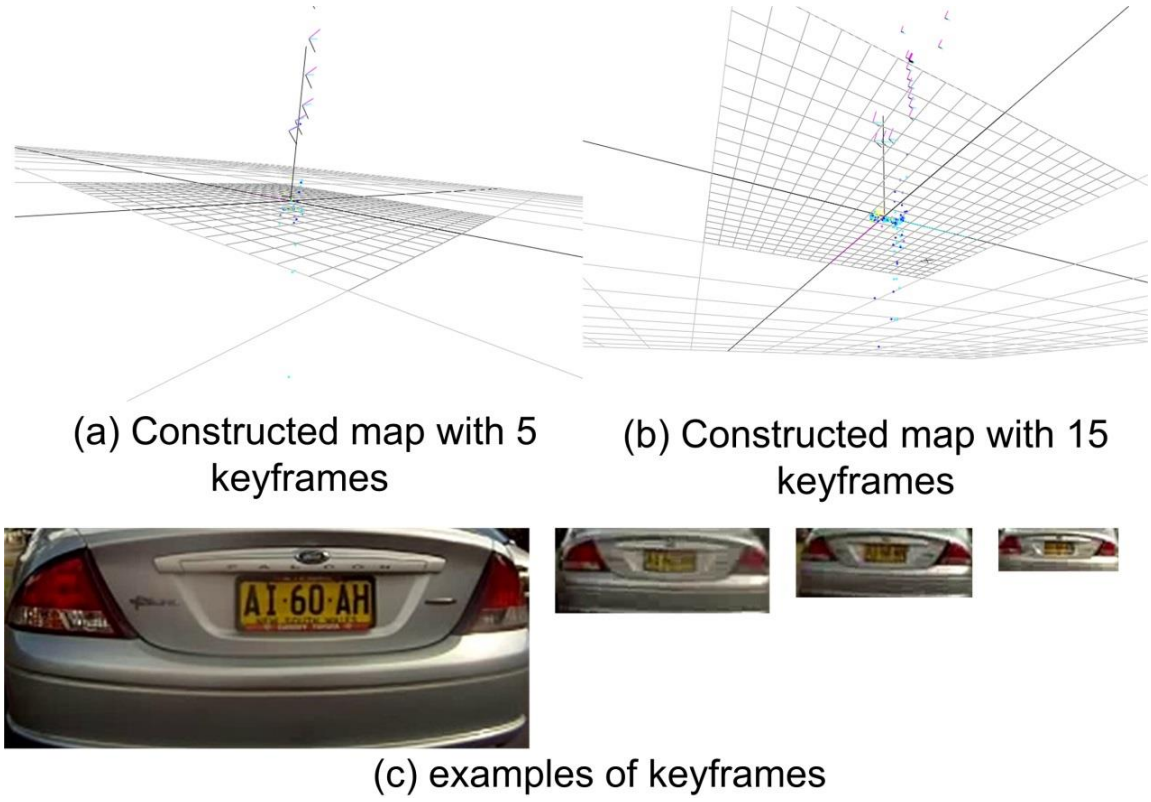


Fig. 7.8 Constructed map and keyframes.

In our method, vehicle rear tracking and mapping are very important for 3D pose estimation. In order to find sufficient and accurate correspondences between the detected features and the map points, excellent feature detection and mapping are a prerequisite. We assessed the performance of the feature detection and mapping in terms of detected features per frame and map growth. For its evaluation, our system was implemented with a video clip of 1093 frames, which recorded a car mounted

with the camera followed by another car on the road. The front car moved at various velocities and turned in numerous directions. Moreover, the distance between the two

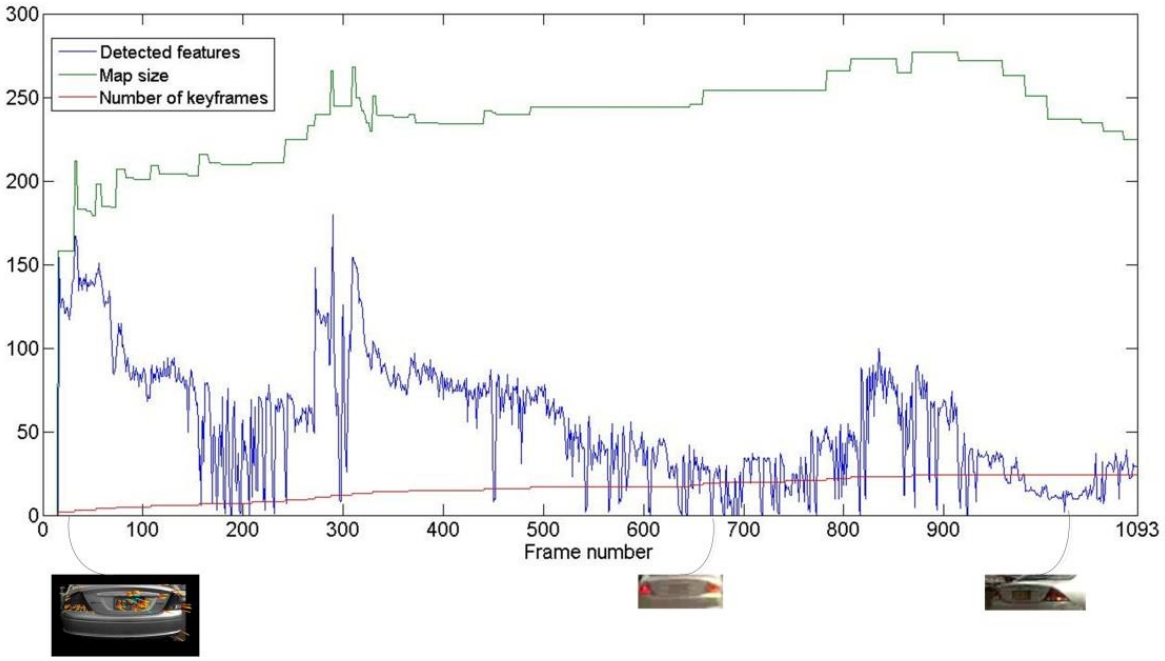


Fig. 7.9 Feature detection and mapping performance.

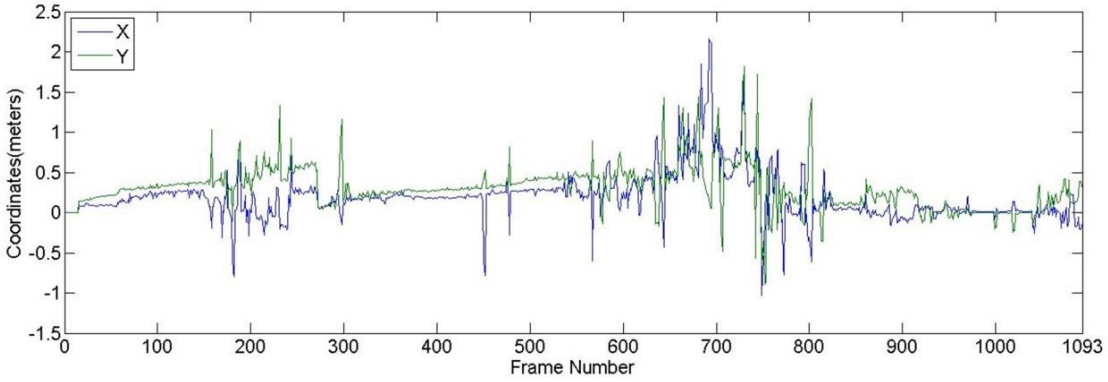
cars varied throughout. Fig. 7.8 illustrates the map and keyframes generated during tracking. Fig. 7.8 (a) and Fig. 7.8 (b) show the maps at two different times with the point features and keyframes drawn. There are 5 keyframes and 202 map points within the map in Fig. 7.8 (a) and 15 keyframes and 235 map points within the map in Fig. 7.8 (b). Fig. 7.8 (c) shows some corresponding vehicle rears when the keyframes were found. Fig. 7.9 describes the evolution of the tracked features and the map size with the frame number. Moreover, we plot the number of keyframes. During the experiment, the map initialisation was finalised in the first 15 frames. Concurrently, the map size and the amount of keyframes were increased with the frame numbers. For example, as the green line shows, the number of map points increased from 158 to 225. The red line represents the number of keyframes, which rose from 2 to 24.

The blue line shows the number of detected feature points on each frame. As explained above, at least 4 correspondences between the detected features and map points are needed. We can see from Fig.7.9 that there is no detected feature from the 1st frame to the 15th frame, because the initialised map is not yet implemented. After

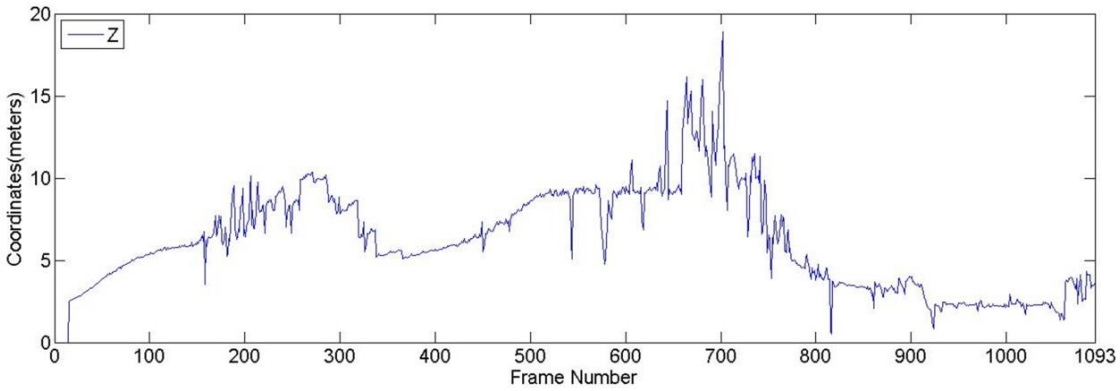
the map was constructed via 158 features, only 83 out of 1078 frames had less than 10 detected features and 46 out of 1078 frames had less than 4 detected features. However, when two cars were approaching one another, more features were detected because the vehicle rear became clearer. When the front car was distant or turning, fewer features could be obtained. Besides this, we can see the performance of the bundle adjustment from Fig. 7.9. When no new keyframes are found, the size of map might drop. This is because some bad map points were discarded when the map was optimised by a bundle adjustment.

7.8.3 3D pose estimation and real-time evaluation

The map in our method is constructed with the feature points found on the vehicle rear. The relative camera poses towards the vehicle rear are the coordinates of the camera in terms of the map coordinates. As shown in Fig. 7.10, the origin of the map coordinate is the centre of the extracted vehicle rear. The map coordinate is defined in



(a) X and Y coordinates of camera pose in Map coordinate frame



(b) Z coordinates of camera pose in Map coordinate frame

Fig. 7.10 3D pose estimation of the camera in a map coordinate frame.

Fig. 7.11 3D model added into the frame based on the estimated pose.

such a way that the *X* axis and the *Y* axis are on vehicle rear plane while the *Z* axis is perpendicular to the vehicle rear plane. Therefore, the measured *X* and *Y* values describe the offset of the camera from the origin of the map, namely the centre of the vehicle rear, and the measured *Z* value is the vertical distance between the camera and the map, namely the vehicle rear. Using the same video clip as that used in section 7.3, the *X* and *Y* coordinates of the camera are shown frame-by-frame in Fig. 7.10(a), while the updating distance between the camera and the map are shown in Fig. 7.10 (b). From Fig. 7.10 (b), we can see the track of these two cars: the two cars were close to each other at the beginning and moved away until the distance reached 18.5 m, and then that the two cars moved closer again and finally stopped when the distance between them was 3.5 m. To demonstrate the performance of our method, we used AR techniques that are strictly dependent on accurate 3D pose estimation in real-time. We used the same 3D model as that in (Klein and Murray 2007): 3D eyeball models are placed on the origin of the map. As shown in Fig. 7.11, the changing size of the eyeball models indicates the updating distance between the camera and the map. The changing direction of the eyeball models represents the varying offset between the camera and the origin of the map. We can see that within the entire 1093 frames the 3D models were added on to the accurate position.

As to the red line in Fig. 7.12, the average processing time for each frame was 38.707 ms. In particular, the time includes the license plate detection, the features detection, the 3D pose estimation and the map optimisation for each frame. From frame 1 to 15, it took round 25 ms for each frame for license plate detection. The map was constructed during frames 15 and 16, where the processing time jumped to above 90 ms per frame. After the map was constructed, the time for license plate detection, 3D

pose estimation and map optimisation for each frame did not vary substantially. We can observe that the figure shape of the processing time was similar to that of the feature detection in Fig. 7.9, where it can be seen that the feature detection primarily accounted for the variation in the processing time.

A demo video can be watched via http://www.youtube.com/watch?v=m5z5TBsTKwI.

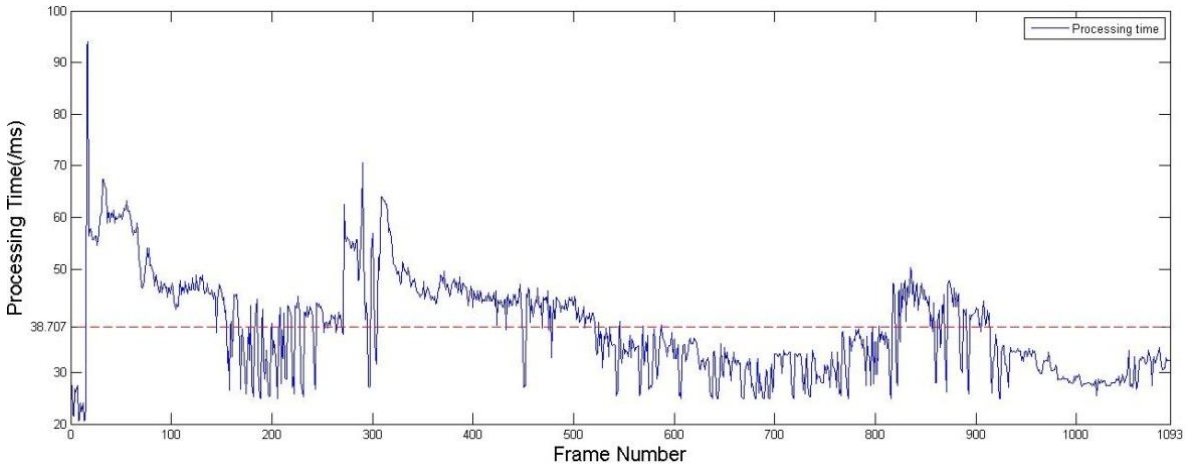


Fig. 7.12 Real-time evaluation.

7.8.4 Evaluation of the map optimisation

As explained above, the map is extremely important for accurate pose estimation. It would be highly risky to insert incorrect information into a map as it may expand when new keyframes are found. In order to demonstrate the superiorities flowing from our map optimisation method, we tested the proposed method both with and without bundle adjustment optimisation on the aforementioned video. As shown in Fig. 7.13, the red line and the green line denote the keyframe size and the map size with map optimisation and frames, respectively. Both of the lines were described in section 7.3: the map size increased when the new keyframes were discovered. Additionally, the map size might decrease when no new keyframe was yielded because some of the outliers were filtered out by bundle adjustment optimisation from the set of map points, as the examples show around frames 900 to 1093. On the other hand, the back line represents the evolving map size and the proposed method without map optimisation. As can be seen, the map size increased from 158 to 421 map points steadily, along with added keyframes. Furthermore, the black line remained flat when no additional keyframe was found. In this case, errors were prone to occur in pose

estimation. In order to put this point into perspective, we still demonstrated the estimated pose by adding the AR model into the video. Fig. 7.14 gives an example of wrong pose estimation by the proposed method without map optimisation: the AR model added in the wrong place, which should have been the front vehicle rear.

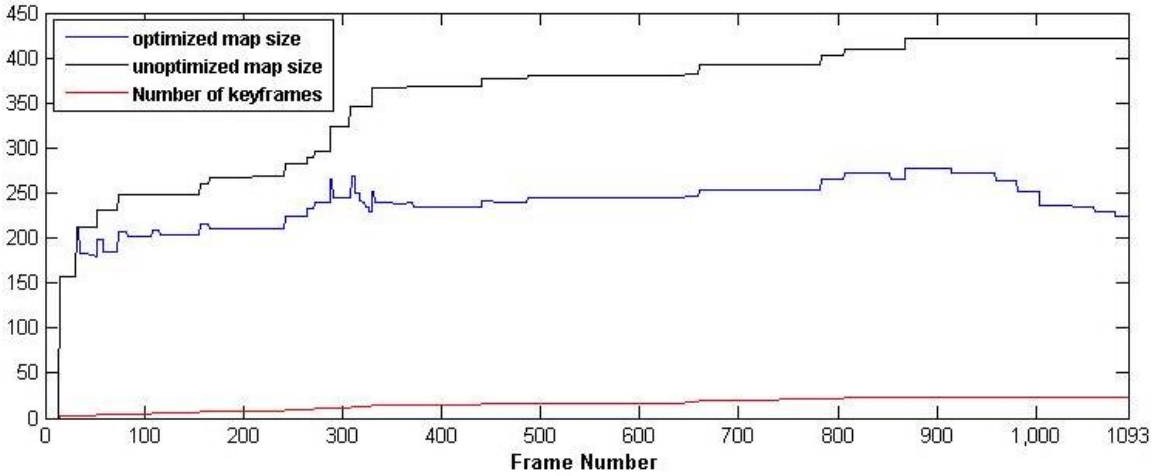


Fig. 7.13 Evaluation of the map optimisation.

Fig. 7.14 An example of wrong pose estimation by the proposed method without map optimisation.

7.8.5 Discussion on lost tracking

Fig. 7.15 In order to make the evaluations comparable, the occluded video is synthesised from the original video.

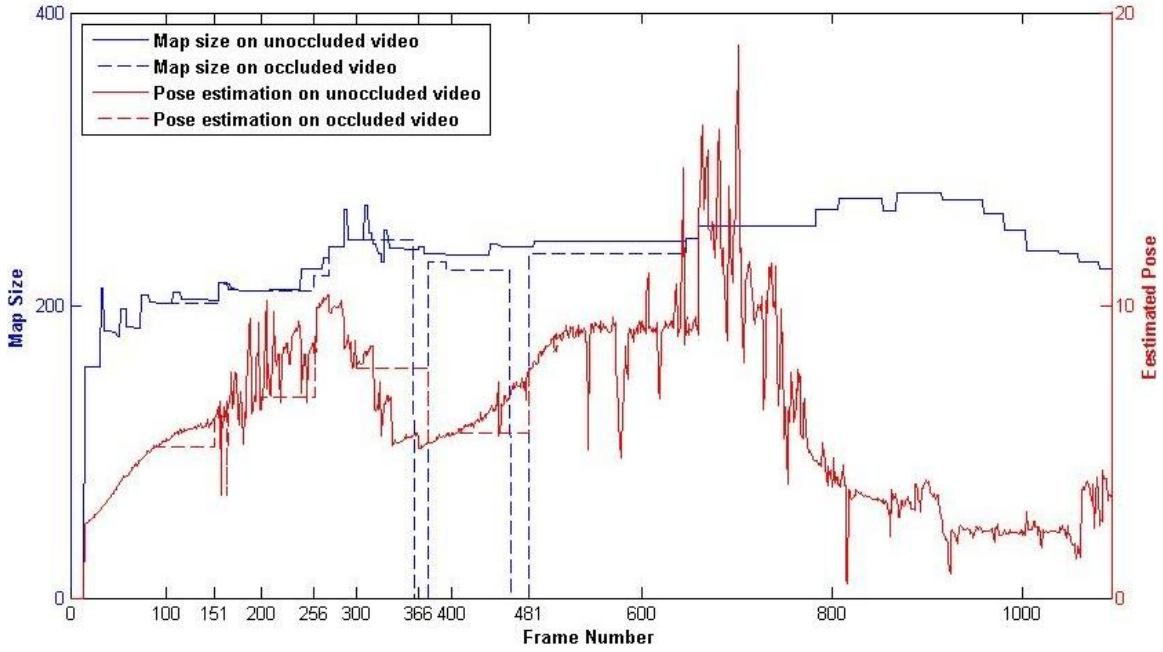


Fig. 7.16 Evaluation of the method by testing on the original and the synthesised videos.

The proposed method consists of two threads: tracking and mapping. The importance of accurate map construction and map optimisation will be discussed in the last section. This section evaluates the tracking thread of the proposed method. Tracking in our method includes LP tracking and features detection. Lost tracking happens when the LP cannot be found or else when less than 4 features are detected. These botherations are caused by the camera's being far away from the front vehicle, vehicle turning and occlusion. Our system is tolerant of temporary lost tracking, where the map and estimated pose can be kept up for a period. In our experiment, the tolerance

of lost tracking is 60 frames. To demonstrate the feasibility of the tolerance setting, we tested the proposed method on the video in which occlusion happened. To make the evaluation comparable, we synthesised a video from the video in Section 7.3 with four temporary occlusions by blocking the ROIs of the frames, which lasted for 50 frames (100 frame—151 frame), 55 frames (200 frame—256 frame), 65 frames (300 frame—366 frame) and 80 frames (400 frame –481 frame). An example of an occluded frame is shown in Fig. 7.15. Referring to Fig. 7.16, the red dashed line and the blue dashed line represent the estimated distance and the map on the synthesised video, respectively, while the red line and the blue line plot the estimated distance and the map on the un-occluded video, respectively. When the first and second occlusions happened, the estimated distances (red dashed line) and the map (blue dashed line) remain unchanging, which was acceptable because the distance would not change much over a very short period (around 1 s). After the occlusions end, pose was estimated again with the previous map. When the third and fourth occlusions happened, the distances remained unchanging. However, the map stayed unchanging for the first 60 frames (the set tolerance threshold) while it was initialised again after 60 frames. Pose was then estimated with the newly constructed map after the occlusion ended. Because the map initialisation was complemented very quickly (around 15 frames), we observed that the estimated poses (red dashed line) were not affected much in the case of occlusion after comparing it to the one (red line) on the original video.

7.8.6 Degree of compliance and the method's limitations

As shown in Fig. 7.17, the proposed method consists of 5 main stages: vehicle rear detection, map initialisation, feature detection, mapping and pose estimation. A high degree of compliance on each stage contributes to the final accurate result. Continuing to refer to Fig.7.17, the task at each stage should be to complement correctly: 1, the two initial keyframes are detected based on LP detection; 2, the initial stereo algorithm complements correctly, 3, more than 4 features are detected in tracking, 4, the detected features are correctly matched with map points, and 5, the pose estimation algorithm is applied correctly. Though our method is robust at all of these stages, as shown by the experimental analysis above, we experienced two types of failure: the first is a failure in LP localisation. Though the effective performance of

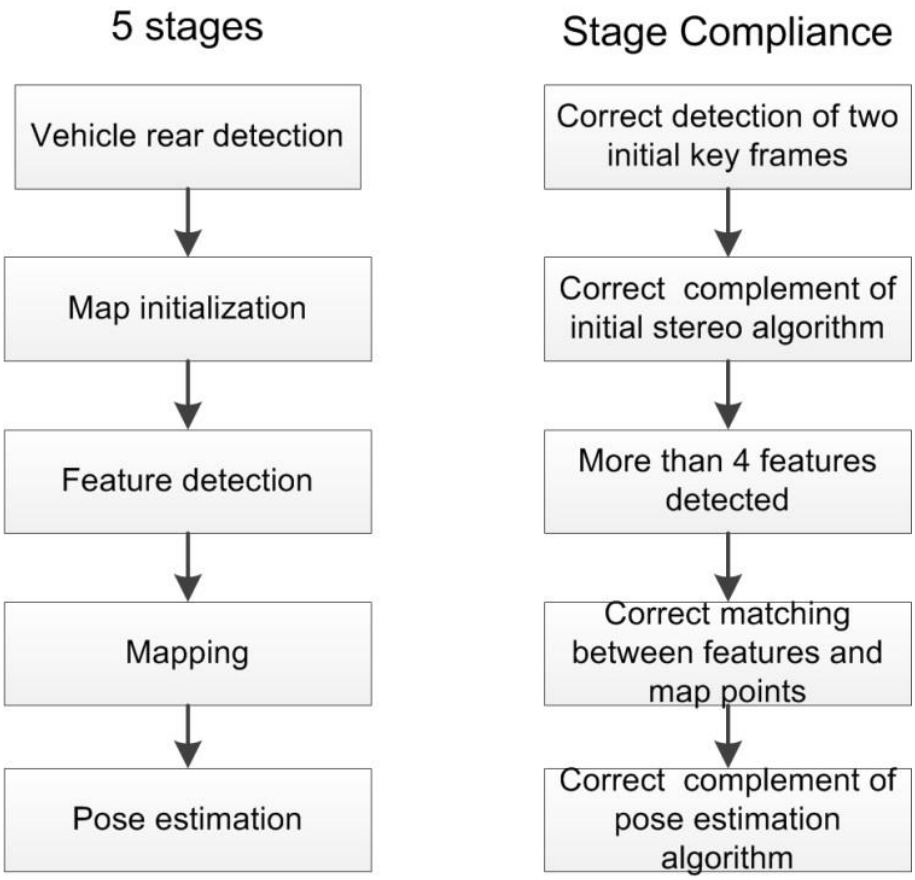


Fig. 7.17 Stage compliance of method.

LP localisation was evaluated in section 7.2, the failure of it is indeed a nuisance as it is fundamental to map initialisation and tracking. Fortunately, the failure of LP localisation always happens when the front vehicle is far away, where the estimate pose of the front vehicle is not especially necessary. The second is a failure of feature detection. This always happens at night. Though the LP can be localised at night because of the existence of vehicle rear lamps, the performance of feature detection is seriously inferior. Therefore, our proposed method is applicable only in daylight.

7.9 Conclusion

In this chapter, we propose a novel visual-based system to estimate the 3D pose of a front vehicle with an on-board camera. The maximum-dependency and minimum-redundancy schemes guide the map initialisation and map maintenance, which the core part in the whole system to bridge image and real world. Using a combination of line segment features and Haar-like features, the LP is quickly localised. The vehicle

rear is extracted based on the localised LP. FAST corners are detected on the vehicle rear. A map containing a set of reference points is initialised from two frames of the extracted vehicle rear at the beginning. After matching the map points with the detected feature points, the relative 3D pose between the current vehicle rear and the camera is calculated frame-by-frame. When new keyframes are found, new feature points would be added into the map. The map is optimised using bundle adjustment when no new keyframe is found. The robustness and accuracy of the proposed method is demonstrated by experimental results. The AR technique is utilised to make the estimated pose in a real-time video visible. With our current method, vehicle rear detection and feature points detection constitute two separate steps. In future work, we intend to find a quicker and more robust method to combine these two steps. Moreover, in order to obtain more accurate and robust pose estimation, an attempt will be made to optimise the process of map initialisation and map maintenance.

With our proposed method, the accurate distance of front vehicle can be obtained in real-time. With this information, the driver assistant can alert the driver to reduce their speed and thus avoid collisions when the front vehicle is too close. Furthermore, the future diver assistant can help drivers to make the right decision with a 3D pose estimation of the front vehicle.

Chapter 8 Feature Selection with Inference Bag of Features Model

The content in this chapter has been published in (Peng, Luo et al. 2012).

The content in this chapter has been presented in the paper "Inference Bag of Features Using Sparse Coding for Image Classification", had been accepted by the 21st International Conference on Pattern Recognition.

8.1 Introduction

Recent advances in computer vision allow us to robustly detect specific objects in the scenes of unrelated clutter, occlusion, and viewpoint changes such as face detection, pedestrian detection and vehicle detection. Object detection is always considered as a binary classification problem that discriminates foreground from background. Object detection methods always use global features or local features. Global features show that objects have some common global characteristics in the whole image and are estimated without invoking segmentation or grouping operations. Background subtraction with global features is a very popular method when the camera is stable. In (Liang, Tieniu et al. 2003), authors construct backgrounds from a small portion of image sequences even including moving objects. The moving pixels are then extracted by comparing a brightness distribution against a threshold value, which is decided by the conventional histogram method. The sliding windows method (Zhang, Marszalek et al. 2007), (Liu and Zhang 2011) is a very typical one using local features for object detection. Sliding windows detectors are always constructed from local features such as Haar-like features, Scale-Invariant Feature Transform (SIFT) descriptors and Histogram of Gradient (HOG) descriptors using machine learning methods such as Adaboost and Support Vector Machines (SVMs). Constructed sliding window object detectors consider small image windows at all locations and scales and perform a binary classification for each.

Further classifications within detected objects are desired in many applications. For example, beyond face detection, gender classification is desired in objective advertising and video surveillance; beyond vehicle detection, vehicle type classification is required by intelligent transportation systems for toll charges and

entrance-guarding. A large body of research has investigated the choices of treating within-class classification as traditional classification problems. For example, authors in (Zhou, Miller et al. 2011) proposed an impressive approach to age classification. They extracted image features using a difference of Gaussian filter followed by Radon transform. The classifier is afterwards trained with SVM using these features. In (Verschae, Ruiz-del-Solar et al. 2006), (Jain, Huang et al. 2005) and (Wu, Ai et al. 2003), authors utilised Haar-like features, Look Up Table (LUT) features or Radial Basis Functions (RBF) with SVM, AdaBoost, or neural-network to classify gender based on detected faces; in (Hsieh, Yu et al. 2006), (Loy and Eklundh 2006) and (Peijin, Lianwen et al. 2007), for vehicle type classification, authors extracted edge features or Gabor features from detected vehicle region and trained a classifier with machine learning methods. However, using these methods, we cannot usually achieve as good results as what we obtain in object detection. First, these methods are always very sensitive to view changes. For example, gender classification methods mentioned above are extremely dependent on frontal faces. Second, differences between sub-classes are much more trivial than that between foreground and background. Therefore, even using the same method on the same images, we achieve much lower accuracy rate in within-class classification than that in object detection. For example, using SIFT-based features in a Bayesian framework on CMU database, correct face detection rate is around 81% (Toews and Arbel 2006) but positive gender classification rate is around only 74% (Toews and Arbel 2009). On the other side, methods using tailor-made representation specific to the sub-classes are popular choices. For example, authors in (Zhang and Wang 2011) extracted geometric features from faces such as distances between eyebrow and chin, helix and earlobe, and inputted these features to SVM to perform gender classification. In (Feris, Siddiquie et al. 2012), authors classified vehicle as cars, minivans, van trucks, and trucks based on length, width and height of detected vehicles. These object specific methods cannot be applied to other problems without major modification.

Recently, Bag-of-Features (BoF) methods have recently demonstrated impressive performance in feature selection. Traditional BoF methods consist of five steps: 1) feature detection; 2) build visual words dictionary; 3) represent training image using visual words in the constructed dictionary; 4) train classifier; 5) represent testing image with visual words dictionary and classify. The concept of BoF is first raised in

(Sivic and Zisserman 2003) that inspired from the idea of text retrieval system. In last decades, many developments in BoF methods have been made with respect to answering three questions: Which feature is better? How do we build a Visual Words Dictionary (VDW)? How do we train a classifier? As to the first question, SIFT descriptor (Sivic and Zisserman 2003), dense descriptor (Ohbuchi and Furuya 2009), colour-based descriptor (Jiang, Ngo et al. 2007), and shape-based (Zhouhui, Godil et al. 2010) descriptors are usually utilised. Among them, a SIFT descriptor is the most popular one because it is more robust to view changes and varying illumination than colour-based and shape-based method; more computational efficient than dense descriptor.

For building VWD and representing images, which can be regarded as two reverse procedures, feature-pooling methods are required. These methods combine several nearby feature descriptors into a local or global BoF, in a way that preserves task-related information while removes irrelevant details (Boureau, Ponce et al. 2010). In (Sivic and Zisserman 2003), vector quantisation was carried with k-means clustering. However, this method is critiqued much as it may cause severe information loss by selecting each feature to the nearest visual word, especially for those features located at the boundary of several visual words. Though this method had been generalised in (Philbin, Chum et al. 2008) by keeping multiple nearest visual words, the number and weights of the visual words to be selected for each feature are easy to be determined. In (Yang, Yu et al. 2009), authors showed state of the art performance in image classification by replacing k-means with sparse coding. Compared with k-means, sparse coding automatically learns the optimal visual words dictionary, and concurrently assigns optimal weights to the visual words for each local feature. When representing images with visual words from VWD using sparse coding, image reconstruction can have a much lower error rate due to the less restrictive constraint, which is not found in k-means clustering. Furthermore, in (Gao, Tsang et al. 2010), a Laplacian sparse coding method was proposed to guarantee the selected cluster centres for similar features are also similar. Another strong criticism on traditional BoF model is the ignorance of location information of features. To overcome this, one extension of BoF proposed in (Lazebnik, Schmid et al. 2006) exploited the spatial information of location regions with Spatial Pyramid Matching (SPM) kernel. The method partitioned an image into finer segments in different scales, computed visual

words representation within each segment, and finally concatenated all representations to form a vector representation of the image.

After representing training images with visual words dictionary, we need to train classifiers. There are two basic kind methods for training classifier: generative methods and discriminative methods. Simply speaking, generative methods classify a testing image by considering every visual word representation of this image separately while discriminative methods make classification decisions by regarding the visual words representation of this testing image wholly. Bayesian hierarchical model (Li and Pietro 2005) is a typical generative method. The method learns a model that best represents the distribution of the visual words over each class. For every testing image, a class model is found that fits best the distribution of the visual word representation of this image. In discriminative methods, SVM (Sivic and Zisserman 2003), (Yang, Yu et al. 2009) and (Lazebnik, Schmid et al. 2006) is a very popular choice. With visual words representation of all training images, SVM found hyperplanes that best separate classes. After representing a testing image with visual words, these hyperplanes classify the image into a class.

To our best knowledge, all current BoF methods directly build visual words dictionary with feature descriptors extracted from training image. More training images are always desired for higher classification accuracy. Increased training data raises the size of VWD as well as the testing time. To guarantee processing speed, authors fix the amount of features descriptors or the size of VWD (Yang, Yu et al. 2009), (Lazebnik, Schmid et al. 2006). However, these constraints would miss much of the available information in training images. To deal with this dilemma, we propose an Inference BoF model that VWD is constructed from a set of inference images rather than training images.

In this Chapter, our proposed inference BoF model for within-class classification can exploit very large training database while guaranteeing testing speed. We use three sets of images with our method: training images, inference images and testing images. We first build VWD from a set of images, which covers all classes and have a fixed size. We call VWD as inference dictionary and these images as inference images. The inference dictionary is the core of our method. We consider it as a bridge between the training database and the testing image. Every visual word in the inference dictionary

can be considered as having an affinity with each class. The affinity is a posterior probability over each class that is learnt from training data based on a Bayesian framework. In testing, a testing image is represented by visual words in the inference dictionary. As posterior probabilities of these visual words over classes have been learned, the visual words representation of the testing image is classified. We use sparse coding for both inference dictionary construction and image representation. The complexity in testing is a constant $O(n)$, where n is the size of inference dictionary. More training images would increase classification rate but does not slow testing speed.

The rest of this chapter is organised as follows: In Section 8.2, we present the overview of our method. Section 8.3 describes inference BoF model. SIFT sparse coding is explained in Section 8.4. In Section 8.5, we present the learning procedure for inference parameters. In Section 8.6, we test our method on large databases of faces as well as comparing with two current popular methods. The chapter is finished by a conclusion in Section 8.7.

8.2 Method overview

As shown in Fig. 8.1, our method uses three set of images: training images, inference dictionary and testing image by two steps: learning and inferring. We first extract SIFT descriptors from inference images. Each SIFT is used to generate a feature vector that describes the local image region sampled relative to its scale-space coordinate frame (Zhou, Yuan et al. 2009). The inference images cover all classes and are with a fixed amount. All these SIFT vectors are then quantised using sparse coding into a visual word dictionary, which we name as inference dictionary. Secondly, we learn a posterior distribution over each class for every visual word in inference dictionary from training images. Every visual word in inference dictionary can be thought of having an affinity with each class. This affinity is learned in Bayesian framework as posterior distribution. Finally, we classify a testing image by inferring to the inference dictionary. The SIFT descriptors extracted on a test image are approximated by visual words in inference dictionary using sparse coding. The posterior distributions over all classes of chosen visual words then determine the class of the testing image.

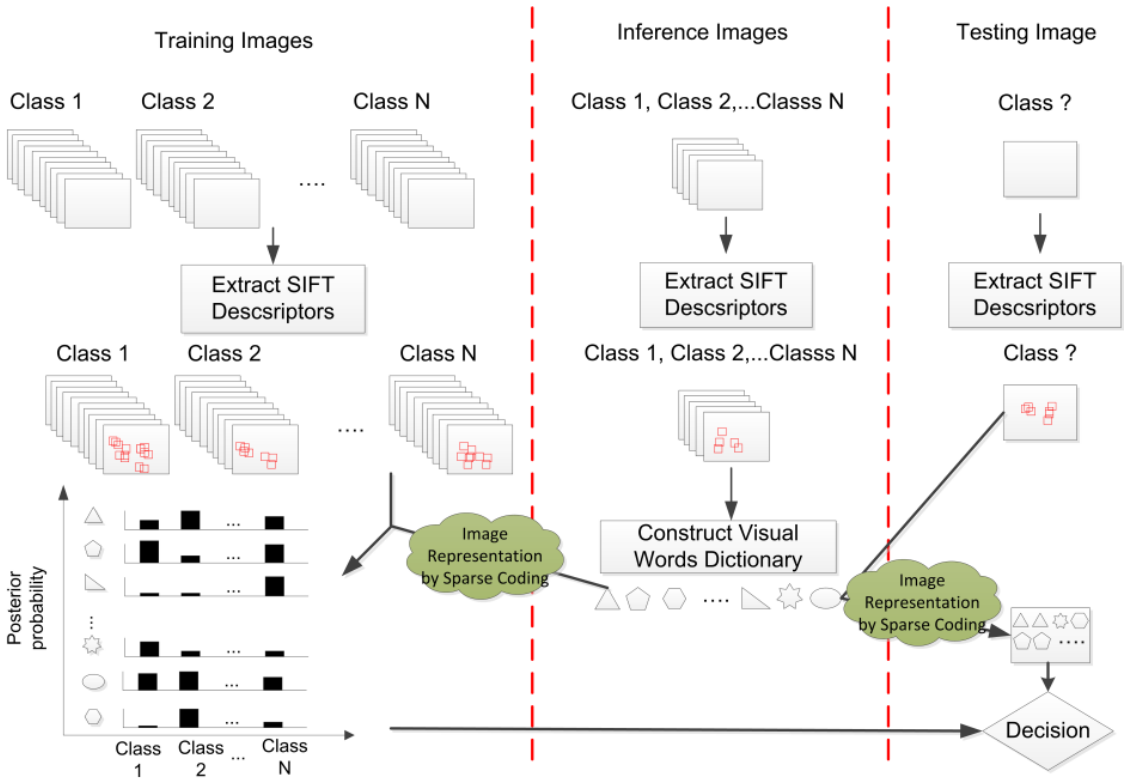


Fig. 8.1 Overview of Inference BoF method.

8.3 Inference BoF model

In current BoF methods, VWD is constructed from training data. The drawback of this method is complexity maybe hardly afforded when using a huge training data. On the other hand, we always desire more training data for higher accuracy. Our proposed inference BoF model addresses this problem. An inference VWD is deployed between testing images and training images. The complexity in testing becomes a constant, as we fix the size of the inference dictionary. More training images would lead to a higher classification rate but does not change the testing time.

The inference dictionary is constructed from a relative small set of images, which we call inference images. These inference images are unlabelled but cover all classes. In our experiment, we utilise 150 male and 150 female images as inference images in gender classification. SIFT descriptors are extracted from these images. SIFT descriptors are 16*16 pixel patches. Consider a set of all these descriptors a bag of features. It is impossible to use the bag directly due to the huge amount of features in this bag. In order to compress this bag, we need to cluster near features. Recently,

sparse coding is proven to outperform k-means method for cluster similar features. After building up visual dictionary using sparse coding, every training image and test image are represented by visual words in this dictionary using sparse coding again. We detail sparse coding in building visual dictionary and representing test images in next sections.

Every visual word chosen can be thought of as having a different affinity with each class. We learn these affinities during learning period with a huge training set. These affinities are represented as a set of parameters associated with each class. The learning process for these affinities is described in Section 8.5. After inferring to the visual dictionary, SIFT descriptors of a test image are approximated by visual words from inference dictionary. The affinity of every chosen visual word for each class is used to determine posterior probabilities of the test image over classes. The inferring process in testing is detailed in Section 8.6.

8.4 Sparse coding in building dictionary and presenting

k-means clustering is widely used for VWD construction in BoF model. Let *X* be a set of SIFT descriptors, where $X = [x_1, x_2, \dots, x_N]$. In k-means method, the SIFT set is partitioned into K clusters $C = [C_1, C_2, \dots, C_K]$. The centres of these clusters $U = [u_1, u_2, \dots, u_K]$ are visual words, which from the dictionary. As a hard assignment method, k-means assigns each local feature to one cluster centre only; the weight contributing to that centre is 1. k-means aims at finding these cluster centres and minimising the inter-class error as following:

$$\min_{U,C} \sum_{i=1}^{K} \sum_{x_j \in C_i} \left\| x_j - u_i \right\|^2 \tag{8.1}$$

The optimisation for the above formulation can be re-formulated into a matrix factorisation problem with a matrix of cluster indices, $V = [v_1, v_2, \dots, v_N]$,

$$\min_{U,V} \sum_{i=1}^{N} \|x_i - U v_i\|^2 \tag{8.2}$$

$$subject\ to: Card(v_i) = 1, |v_i| = 1, v_i \succcurlyeq 0, i = 1, \dots N$$

where $Card(v_i) = 1$ is a cardinality constraint, meaning that only one element of v_i is nonzero, $|v_i|$ is the summation of the absolute value of each element in v_i, $v_i \succcurlyeq 0$ means that all elements of v_i are nonnegative. After the optimisation, the index of the

only nonzero element in v_i indicates which cluster the SIFT x_i belongs to. In the phase of building visual dictionary U, the optimisation (2) is solved with respect to both U and V. After dictionary U built up, image is represented by visual words u_i. In the phase of representing image, optimisation (2) is solved with respect to both V only and with a new set of x_i.

However, the constraint that each SIFT only contributes to one visual word is too strict. This constraint could cause severe information loss, especially for those features located at the boundary of several clusters. To this end, $Card(v_i) = 1$ is removed for soft assignment, which optimally selects cluster centres for each SIFT. Moreover, to avoid each SIFT to be assigned to too many clusters, a sparse constraint on v_i is used that enforces v_i to have a small number of nonzero elements. The optimisation problem is turned into a sparse coding problem:

$$\min_{U,V} \sum_{i=1}^{N} \|x_i - Uv_i\|^2 + \lambda\|v_i\|_1 \tag{8.3}$$

$$subject\ to: |u_k| \leq 1, k = 1, \dots, K$$

Equation (3) is not convex for U and V simultaneously, but it is convex for U when V is fixed and it is also convex for V when U is fixed. We optimise U and V alternatively. Fixing V , the optimisation over U can be solved as a least square problem with quadratic constraints:

$$\min_{U} \|X - UV\|_F^2$$

$$subject\ to: |u_k| \leq 1, k = 1, \dots, K \tag{8.4}$$

The optimization of (4) is done by Lagrange dual in [20]. When U is fixed, we used the linear regression algorithm in (Lee, Battle et al. 2007) to optimize over each coefficient v_i individually:

$$\min_{V} \sum_{i=1}^{N} \|x_i - Uv_i\|^2 + \lambda\|v_i\|_1 \tag{8.5}$$

Similar to k-means method, we need to optimise (3) over both U and $V$when building the visual dictionary. When representing training image or test image with visual words, we need to optimise (3) with respect to V only.

8.5 Affinity learning

As explained above, to overcome the problem that a lot of training data slows testing time, we construct an inference visual dictionary from a relative small set of unlabelled inference images. Every visual word has an affinity with each class. These affinities are learned from labelled training images, which are with large amount and are different from inference images.

SIFT descriptors are detected on training images. We denote the descriptor from the i^{th}training image of the m^{th} class by t_{mi}. Using sparse coding, t_{mi} is assigned to a visual word in dictionary $U = [u_1, u_2, \ldots., u_K]$ ($u_i \in R^{d \times K}$). The affinity of a visual word with class is denotes as θ_{mu}, which represents the tendency for the visual word u to be picked when considering SIFT descriptors of class m .

In training period, we denote all SIFT descriptors belonging to the m^{th} class training images that as t_m. When t_m are picked for a visual word u_i, we learn a posterior distribution $p(\theta_{mu} \,/\, t_m)$ using Bayes' rule:

$$p(\theta_{mu}/t_m) = \frac{p(t_m/\theta_{mu})p(\theta_{mu})}{p(t_m)} \tag{8.6}$$

To simplify notation, we describe this process for just one of the K visual words and one of m classes and drop the indices u and m. Equation (6) is presented:

$$p(\theta/t) = \frac{p(t/\theta)p(\theta)}{p(t)} \tag{8.7}$$

where $t = [t_1, t_2, \ldots, t_T]$ is all descriptors from training data for this visual word and this class; θ. is the parameter associated with this visual word for this class. Substituting t with u into equation (7), we get a posterior distribution $p(\theta/u_i)$ over the affinity θ:

$$p(\theta/u) = \frac{p(u/\theta)p(\theta)}{p(u)} \tag{8.8}$$

The likelihood part is a categorical distribution over the visual word that is one sample from a multinomial. We get:

$$p(u/\theta) = \prod_{i=1}^{T} p(u_i/\theta) = \prod_{i=1}^{T} \theta_{u_i} = \prod_{u=1}^{K} (\theta_u)^{f_u} \tag{8.9}$$

where T is the amount of all descriptors detected from training data and K is the number of all visual words in the inference dictionary. f_u is defined as:

$$f_u = \sum_{i=1}^{T} \lambda \|v_i\|_1 \tag{8.10}$$

$$subject\ to: \min_V \sum_{i=1}^{T} \|t_i - Uv_i\|^2 + \lambda \|v_i\|_1,\ \ ubject\ to: |u_k| \leq 1, k = 1, \dots, K$$

where $\lambda \|v.\|_1$, as described in equation (3), are cluster membership indicators with a L1-norm regularisation. U is the constructed inference VWD. $t.$ is SIFT descriptors extracted from training images. In other words, f_u is the frequence that visual word u_i is picked up during training process.

We choose a Dirichlet prior over parameters θ in equation (8) as it is conjugate to the categorical likelihood:

$$p(\theta) = \frac{\Gamma(\sum_u \alpha_u)}{\prod_u \Gamma(\alpha_u)} \prod_{u=1}^{K} (\theta_U)^{\alpha_u - 1} \tag{8.11}$$

where Γ represents a Gamma distribution and $\{\alpha_1, \alpha_2, \dots, \alpha_K\}$ are the parameters of this Dirichlet distribution. These parameters are learned from a validation set. Substituting the likelihood part (9) and conjugate prior (10) into (8), we get a form of a Dirichlet distribution to describe posterior distribution over visual words' affinity:

$$p(\theta/u) = \frac{\Gamma(\sum_u (\alpha_u + f_u))}{\prod_u \Gamma(\alpha_u + f_u)} \prod_{u=1}^{K} (\theta_U)^{f_u + \alpha_u - 1} \tag{8.12}$$

We calculate the distribution for each visual word with respect to each class.

8.6 Inferring in testing

For every unknown testing image, we approximate SIFT descriptors detected on this image with visual words in inference dictionary. Every visual word has an affinity with classes that has been learned from training images. Chosen visual words determine the class of testing image. We call this process inferring.

After detecting SIFT, we represent a test image Y as a set of SIFT descriptors $Y = [y_1, y_2, \dots, y_P]$. The possible classes are $C = [C_1, C_2, \dots, C_K]$ and all training SIFT descriptors are $t = [t_1, t_2, \dots, t_T]$. The posterior probability over class label C for a test image is:

$$p(C = c|Y, t) \tag{8.13}$$

We get a further expression with the assumption every SIFT descriptor is independent.

$$p(C = c|Y, t) = \frac{\prod_{p=1}^{P} p(y_p|C=c,t.)p(C=c)}{p(Y)} \tag{8.14}$$

After obtaining the closest match from visual word dictionary using sparse coding, we substitute y_p with $u.$ into the likelihood part of (12). We take a Bayesian method and marginalise over model parameters, so the likelihood terms become:

$$p(y_p|C = c, t) = p(u.|C = c, t) = \int p(u|\theta)p(\theta|u)d\,\theta \tag{8.15}$$

$$p(\theta|u) = \theta_u \tag{8.16}$$

where θ_u, as explained above, represents the probability for a visual word to be chosen in training process. We substitute (12) and (16) into (15):

$$p(y_p|C = c, t\,) = \theta_u * \int \frac{\Gamma(\sum_u(\alpha_u+f_u)}{\prod_u \Gamma(\alpha_u+f_u)} \prod_{u=1}^{K}(\theta_U)^{f_u+\alpha_u-1}\, d\theta \;=$$
$$\int \frac{\Gamma(\sum_u(\alpha_u+f_u)}{\prod_u \Gamma(\alpha_u+f_u)} \prod_{u=1}^{K}(\theta_U)^{\lambda\|v\|_1+f_u+\alpha_u-1}\, d\theta \tag{8.17}$$

Turning the term inside the integral to a complete Dirichlet distribution which sums to one, we get:

$$p(y_p|C = c, t\,) = \frac{\Gamma(\sum_u(\alpha_u+f_u)}{\prod_u \Gamma(\alpha_u+f_u)} \frac{\prod_u \Gamma(\lambda\|v\|_1+\alpha_u+f_u)}{\Gamma(\sum_u(\lambda\|v\|_1+\alpha_u+f_u))} = \frac{f_u+\alpha_u}{\sum_u(f_u+\alpha_u)} \tag{8.18}$$

8.7 Experiment and discussion

In this section, we implement and evaluate three BoF methods on challenging tasks: gender classification on images of FERET database (Facial Recognition Technology Database) (Phillips, Moon et al. 2000). Three BoF methods are:

1. Inference BoF: our proposed method that uses an inference dictionary to infer classes' probabilities of testing images in a Bayesian framework.
2. Bayesian BoF: calculate class probabilities of testing images in a Bayesian framework, without inference images.
3. ScSPM BoF: very impressive discriminative method proposed in (Yang, Yu et al. 2009) that uses SVM linear kernel on spatial-pyramid pooling.

All these three methods use sparse coding in the phases of visual world dictionary construction and image representation.

8.7.1 Data collection

All training and testing image for gender classification are based on the standard, publicly available colour FERET database (Phillips, Moon et al. 2000). The FERET database consists of a total of 11338 facial images, which are collected by photographing 994 subjects of various ethnicities, age, sex, acquired from various viewpoints, illumination conditions, with/without glasses, etc. We build a database of 4970 images, each subject with 5 images, which are chosen at viewpoint of 0, 15, 22.5, 45, and 67.5 degrees. The male: female ratio in our database is approximately 3:2 (2955:2015). Due to other objects such as background, clothes, hair, etc. in FERET images, we need to extract face images. Faces are detected in two steps: first, we used a face detector on all images in our database. Thanks to the simple scenario of FERET images for detection and localisation, we do not get many mis- or false detections. Second, we manually crop the missed faces, most of which are with viewpoints of 45 or 67.5 degrees.

We standardized all extracted face image in following five steps: 1, transfer all images into greyscale; 2, resize all images into 120 ×120 pixels; 3, band-pass filter all images; 4, smooth all images using a Gaussian function centred on the image; 5, normalise all images to have zero mean and unit standard deviation. Fig. 8.2 shows some face image examples.

8.7.2 Parameters setting

In this section, we investigate sparse coding parameter λ in equation (3) and Dirichlet parameters $\{\alpha_1, \alpha_2, \ldots, \alpha_K\}$. Sparse coding parameter λ enforces the sparsity of the solution: the bigger λ is, the more sparse the solution will be. As (Yang, Yu et al. 2009) indicated, keeping the sparsity around 10% yields good results. Therefore, for both training and inferring, we simply fixed λ to be 0.3~0.4 and the mean number of supports is around 10.

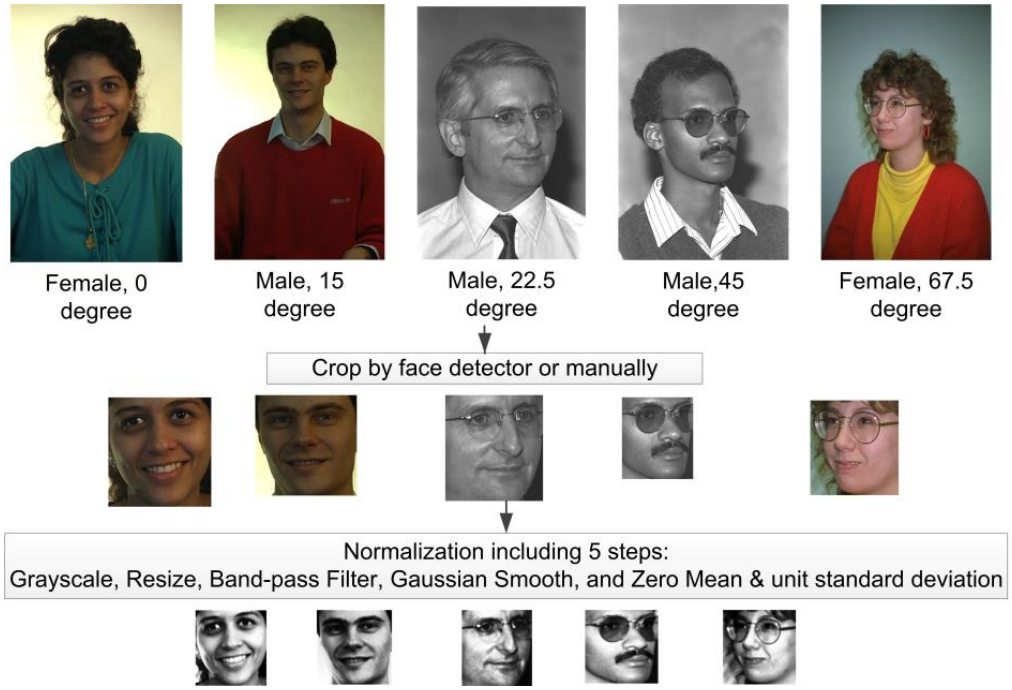


Fig. 8.2 Face image collection. Experimental images that are used in this chapter are from the Facial Recognition Technology (FERET) database (Phillips, Moon et al. 2000).

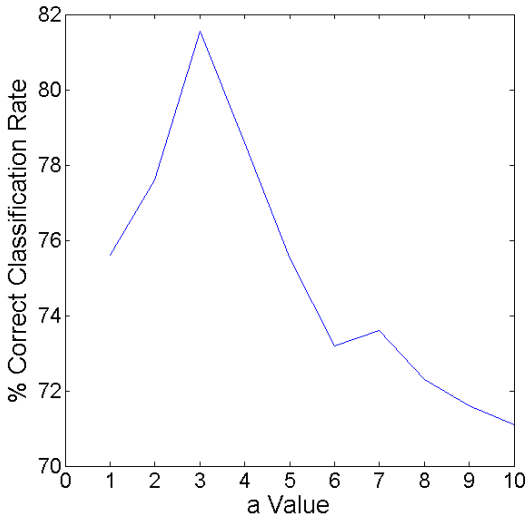


Fig. 8.3 Classification rate varies with changing Dirichlet distribution parameter, which is noted as α.

As a very common case of Dirichlet distribution, we constrain all of the elements making up the parameter vector $\{\alpha_1, \alpha_2, \ldots, \alpha_K\}$ to have the same value. In order to learn the best α for classification, we used a training set of 1000 male and 1000 female images, a validation set of 200 male and 200 female images, and an inference dataset made up of 100 male and 100 female images. We empirically chose α from 1

to 10. From Fig. 8.3, the classification performance jumps significantly from 1 to 3 but then declines. We choose α as 3. This result also shows the advance of the Baysian Inference framework, as the maximum-likelihood solution can be seen as a special case of Bayesian inference when $a = 1$.

8.7.3 Visual word dictionary size

The visual word dictionary is constructed from extracted SIFT features. Intuitively, we obtain a higher classification accuracy when we extract more SIFT features from more training images. However, classification time cannot be afforded when the size of the dictionary is extremely large. To address this problem, most current methods fixed the amount of features or the size of dictionary. In (Lazebnik, Schmid et al. 2006), the authors fixed the dictionary size to 200 and 400. In (Yang, Yu et al. 2009), the authors used 50,000 SIFT features to train dictionaries of three sizes: 256, 512 and 1024, separately. They found their method achieved the best performance when the dictionary contained 1024 visual words. The cost of fixed-size SIFT descriptors and dictionary is classification accuracy.

Our Inference BoF method addresses this problem. We use a high number of training images to guarantee a high classification rate while using a relatively small set of inference images to constrain the size of the dictionary.

8.7.4 Gender classification using inference BoF

For gender classification, we used a training set of 1800 male and 1800 female images. The inference images set consists of 150 male and 150 female images and the testing set is made up of 605 male and 402 female images. All images come from our faces database. As shown in Table 8.1, we implemented the three methods on the same images.

Table 8. 1 Implement three methods on the same images for gender classification.

Methods	Training Images	Inference Images	Testing Image
Inference Bof	1800 male, 1800 female	150 male, 150 female	605 male, 402 female
Bayesian BoF	1800 male, 1800 female	N/A	605 male, 402 female
ScSPM BoF	1800 male, 1800 female	N/A	605 male, 402 female

Table 8. 2 The effect of dictionary size on Inference method.

SIFT(Training)	SIFT(Inference)	Dictionary Size	Dictionary Size	Dictionary Size	Dictionary Size
252,019	**21,037**	256	512	**1024**	2048
		65.52%	71.25%	**85.60%**	80.54%

Table 8. 3 The effect of dictionary size on Bayesian BoF method and ScSPM BoF method.

	SIFT (Training)	SIFT (Inference)	Dictionary Size	Dictionary Size	Dictionary Size	Dictionary Size
			1024	2048	**4096**	**8192**
Bayesian BoF	252,019	N/A	65.18%	68.20%	70.62%	**75.5%**
ScSPM BoF	252,019	N/A	70.20%	81.50%	**85.14%**	72.41%

Table 8. 4 Computation complexity comparison.

	Inference BoF	Bayesian BoF	ScSPM BoF
Average Testing Speed	830 ms/ frame	2400 ms/frame	1300 ms/frame

With the Inference BoF method, the dictionary is constructed from inference images. As shown in Table 8.2, we extracted 252,019 SIFT features from 3600 training images and 21,037 SIFT features from 300 inference images. We tried four dictionary sizes: 256, 512, 1024, and 2048. Inference BoF achieves best performance when the dictionary size is 1024.

In Bayesian BoF and ScSPM BoF, dictionary is constructed from training images. As shown in Table 8.3, we extract the same amount of SIFT feature from training images as that in Inference BoF. As we extracted many more SIFT features from training images that that from inference images, we tried four bigger dictionary sizes for Bayesian BoF and SVM BoF: 1024, 2048, 4096, and 8192. Bayesian BoF achieve

best classification rate with the dictionary size 8192. ScSPM BoF obtains best performance when the dictionary size is 4096.

As shown in Table 8.4, the best performances of Inference BoF and ScSPM BoF are better than that of Bayesian BoF. There is no great difference between Inference BoF and ScSPM BoF. However, the processing speeds are discriminative. The dictionary size is 1024 when our method achieves best performance. The processing time for an incoming image in classification is $O(1024)$. On the other side, the processing time is $O(8192)$ and $O(4096)$ for Bayesian BoF and ScSPM BoF, respectively, when they achieve best classification rate.

8.7.5 Computation complexity evaluation

As explained in Section 8.4, Inference BoF, Bayesian BoF and ScSPM BoF achieved their best performances when the dictionary sizes are 1024, 8192 and 4096. Bigger size of VWD leads more testing time. In order to confirm this, we compared the testing time when three methods achieved their highest classification rate. We tested three methods on the same 1007 faces images. As Table 8.4 indicates, our proposed method processed testing images at a speed of around 830 ms/frame. The average testing speeds of Bayesian BoF and ScSPM BoF are around 2400ms/frame and 1300ms/frame, respectively. We calculated the testing time from input image into methods to make classification decision. We implemented three methods in C++ on a desktop with Intel(R) Duo CPU 3.00GHz 2GB memory. Please refer to the supplementary video to watch the demonstration of gender classification with our method.

8.8 Conclusion

In this chapter, we proposed an inference BoF method for selecting the optimal set of features, which is the aim of minimum-redundancy scheme. The method constructs visual words dictionary from inference images instead of training images in traditional methods. The size of inference dictionary is fixed when we don't change the amount of inference images. This method dramatically improves the scalability of training data and the speed of testing, and improves the classification accuracy. Our experiments on the changeling task—gender classification, demonstrated the effectiveness of this method.

Chapter 9 Discussion and Future Work

9.1 Summary

The primary focus of this thesis is our proposed scheme of feature selection for vision-based applications in an ITS. Feature selection schemes with maximum dependency and minimum redundancy were presented. In extensive experiments on large real-world datasets, we demonstrated a significant improvement in performance over state-of-the-art methods for all ITS applications considered in this thesis, i.e., license plate detection, vehicle type classification, and pedestrian counting. Finally, we proposed the inference Bag-of-Features (BoF) method for feature selection that outperformed the traditional BoF in terms of both the processing time and accuracy.

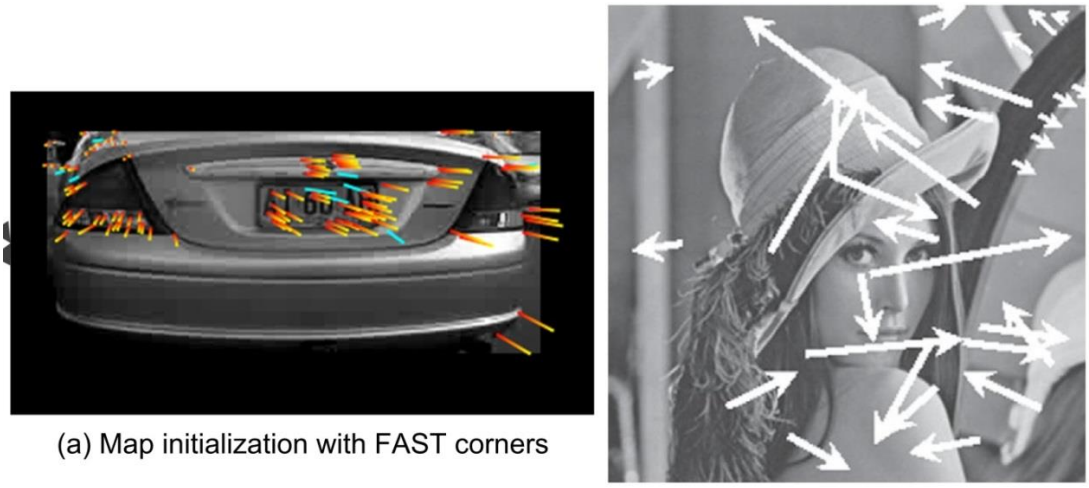
Chapter 1 introduced cognitive systems with machine learning into ITS system. By a set of algorithms than can retain knowledge from past interactions with the environment, transform this knowledge into experience, and plan future actions accordingly, the cognitive systems enabled prior perceived potential and amend behaviours with respect to traffic. Cognitive systems exploited the intelligence that was accumulated through the exchange of information among ITS nodes: vehicles, drivers, pedestrians, and infrastructure. Hence, the efficient and lossless exchanging of information was extremely critical. To avoid noise and guarantee the effectiveness of the information transfer in a cognitive system, finding suitable selected features was critical. In Chapter 2, we proposed the feature selection scheme of maximum dependency and minimum redundancy, which guided the development of visual applications in an ITS. In Chapter 3, we presented a comprehensive review that involved state-of-the-art feature selection methods utilised in the ITS. In Chapter 4, the outline and main contributions of this thesis were presented.

To evaluate the advancement of our feature selection scheme with these ITS applications, we implemented the experiments on a large-scale real-world dataset in Chapter 5 to Chapter 8. We obtained significant performance boosts over previous state-of-the-art methods for license plate detection, vehicle type classification, pedestrian counting, 3D pose estimation of the front vehicle, and face recognition. Chapter 5 presented license plate detection and vehicle type classification with a traffic surveillance camera. We utilised a combination of line segment features and

Haar-like features to achieve fast and robust license plate detection. In vehicle type classification, we showed the advance of our proposed multiple eigenspaces over a traditional eigenspace in selecting feature subsets. Every vehicle front was represented by eigenvectors after being projected into multiple eigenspaces. A higher classification rate was produced with our method. In Chapter 6, hybrid features that were low-level and high-level features were extracted for pedestrian counting with a linear SVM. Rather than detecting the pedestrians directly, we transferred the pedestrian counting problem to a classification problem. In Chapters 5 and 6, we mainly addressed the cognitive systems with a fixed camera. In Chapter 7, we implemented a feature selection scheme with a more complex cognitive system, the intelligent vehicle. With an on-board camera, we proposed an application to estimate the real-time position of the front vehicle toward a driver assistance system. The 3D position of the front vehicle on the road was calculated by a map, which decoded the relationship between the real world and the 2D features. In Chapter 8, we improved a popular feature selection method---Bag-of-Features using our scheme and demonstrated the improvement by addressing a challenging ITS problem that gender recognition of pedestrians.

9.2 Discussion

According to the proposed maximum dependency feature selection strategy, features should be selected with full consideration of the aim of the applications, the environment, and the processing speed requirement. We grouped features that were typically utilised in an ITS into three groups: low-level features, high-level features, and hybrid features (see Tables 3.1 and 3.2). Low-level features were those basic features that could be extracted from an image without any information about the spatial relationship, such as the pixel, edge and localised features. These features can describe subtle details about objects.



(a) Map initialization with FAST corners

(b) SIFT features

Fig.9. 1 Low-level features represent object by exploiting local neighbourhood properties. They are able to capture subtle characteristics but prone to include outliers. (a) In Chapter 7, FAST corners were extracted to initialize the map. (b) In Chapter 8, SIFT descriptors were utilized for face recognition.

In Chapter 7, the 3D position of the front vehicle was achieved by calculating the relationship between the real world and the features. Because the texture characteristics of the vehicle rear were needed and the extremely unstable background, FAST corners were selected as discrimination and robustness of this feature. The motivation of choosing FAST corners was that they were calculated extremely fast and represented the vehicle rear in sufficient detail. As shown in the Fig.9.1 (a), the texture in detail was essential for establishing the relationship between 2D image and 3D real world. In Chapter 8, another low-level feature, SIFT descriptor, was utilized for constructing the visual words dictionary because of its robustness to view change and illumination vary. The image of a person's face exhibits many variations which may affect the ability of a computer vision system to recognize the gender. It was due to the characteristics of a person, such as age, ethnicity and facial expressions, and the accessories being worn. Geometric-based and appearance-based methods hardly capture all these variations. We utilized the SIFT descriptors to represent the face as a collection of features. It made our method to capture the local characteristics as many as possible.

We have demonstrated the advanced performance of the utilized low-level features through the experiments of Chapter 7 and Chapter 8. On the other hand, high-level

features were incompetent for these two tasks for two reasons: first, high-level features were not suitable for capturing local characteristics. Low-levels features represented the objects in a collection of points or patches by exploiting local neighbour properties, such as the FAST corners and SIFT descriptors in Chapter 7 and Chapter 8. High-level features concerned finding shapes of objects. For example, human face can be recognized automatically using high-level features. The major face features such as the eyes, the ears, and the nose were extracted. To find them, we could use their shapes: the white part of the eyes was ellipsoidal; the mouth can appear as two lines, as do the eyebrows. These features were enough to recognize human faces from other objects. However, they cannot discriminate the male face from female one that required much more subtle characteristics. Second, because high-level features were based on the spatial information of targets, they were sensitive to viewing changes. In a planar image an object viewed from a different angle will appear different, but points which represented it still appeared in a similar arrangement. In Chapter 7, the relative view of the front vehicle to the camera changed frequently. Consequently, the spatial information of vehicle rear varied constantly. The high-level features cannot be extracted accurately in this case.

However, high-level features extraction process can be viewed similar to the way we perceive the world. More complex objects can be decomposed into a structure of simple shapes. As aforementioned discussion, the human face could be decomposed into a structure of nose, eyes, eyebrows, and mouth. The vehicle could be recognized as a combination of wheels (circles), frames (polygons), and lamps (ellipses). In many applications, analysis can be guided by the way the shapes are arranged. In Chapter 5, by observing the pedestrian and pedestrian groups in the surveillance videos, which were ground-based in the scene, shape-related features were extracted to classify the number of pedestrians in each foreground blob. It was worth noting that many high-level features were produced from low-level features, i.e., the shape of an object was described by a collection of low-level features. For example, HOG was a very successful feature for pedestrian detection. It described the pedestrian's shape by counting the occurrences of the gradient orientation in localised portions of an image. Moreover, hybrid features that combined low-level features and high-level features were usually chosen in an ITS application to take advantage of both of the two methods. In Chapter 5, line segment features were extracted to represent the texture

appearance of license plate which were generated by detected edges on license plate. In addition, the combination of line segment features and Haar-like features was used to detect license plate.

As discussed before, the computation load of single low-level feature was low. However, in low-level feature based method, objects were represented by a collection of low-level features. The computation load increased with the increasing amount of features. When multiple objects were targeted in the scene, a slide window method was utilized. Usually, a scalable window slid through the whole image. For each window, the low-level features were extracted. The processing speed became slow. Because the amount of high-level features were usually relatively small, such as three line segment features in Chapter 6 and nine foreground blob features in Chapter 7, the processing speed of the collection of these features was fast. Hence, in the application of multiple objects, high-level features were usually preferred.

After extracting suitable types of features, reducing the size of the feature set was another important step according to the proposed minimum redundancy scheme. To achieve the best performance, the optimal set of features was in demand that included the most informative features while excluding the redundant features. The choice of feature subset selection method mainly based on the features, which have been extracted from objects in the last step. In Chapter 7 and Chapter 8, low-level features were extracted without considering spatial information. As shown in Fig. 9.1**,** some FAST corner features which were outside the vehicle rear were included while many SIFT features which were outside the face were detected. These outliers needed to be excluded. In Chapter 7, after extracting FAST corners to build the map that recorded the relationship between the real world and 2D image, the Levenberg-Marquardt algorithm was utilized to reduce the size of feature set in map maintenance.

Bag-of-Features (BoF) outperformed many other approaches in object classification. However, the processing time of the BoF methods increased substantially with the additional training data. In Chapter 8, we addressed this problem by proposing an inference bag-of-features method. Traditional BoF methods constructed a visual word dictionary from the training images. A substantial amount of time was required for both training and testing for large-scale training data. A fixed size for the VWD in the current methods guaranteed the processing speed but would miss the available

training data. Our method addressed this dilemma by introducing the inference algorithm. VWD was constructed from inference images, the number of which was fixed. Posterior probabilities of visual words over classes were learned from training images in a Bayesian framework.

On the contrary, high-level features were relevant to objects because of the way they were generated. The optimal feature sets were usually generated by ranking the significance of features. In Chapter 5, the line segment features with Harr-like features were weighted by AdaBoost during the classifier training. Top eigenvectors were selected by PCA method. In Chapter 6, the extracted features of foreground blobs were scored by SVM. The top ranked features were highly relevant to the objects that performed better than other features in the predication task.

	Maximum Dependency Scheme				**Minimum Redundancy Scheme**	
	Input				Output/Input	Output
	Sub-class recognition	View Change	Real Time Requirement	Multiple Targets	Low-level/High-level /Hybrid features	Optimal feature set selection
Vehicle Type Classification	No	No	No	No	Hybrid--Line segment features, Haar-like features, Eigenvectors	AdaBoost, PCA
Pedestrian Counting	No	No	Yes	Yes	High-level--Blob	SVM

					features	
Pose Estimation of front Vehicle	No	Yes	Yes	No	Low-level--FAST corners	Levenberg-Marquardt
Pedestrian Gender Classification	Yes	No	No	No	Low-level--SIFT descriptors	Inference Bag of Features

Table 9. 1 ITS applications under the guidance of maximum dependency and minimum redundancy scheme. In the first step, several factors such as application environment and object character were input while extracted features were output. In the second step, the extracted features were input while the feature selection method was the output.

In summary, the best performance of ITS system can be expected with the best optimal feature set. As shown in Table 9.1, the achievement of best optimal feature set included two steps: extracting most suitable features and subset selection, which were under the guidance of maximum dependency and minimum redundancy. In the first step, several factors such as application environment and object character were input while extracted features were output. In the second step, the extracted features were input while the feature selection method was the output. By referring to the aforementioned discussion, high-level features were suitable for real-time response and multiple-targets task. But they were not expressive in the task of sub-class recognition because of their poor ability of representing local characteristics. Low-level features performed well in the task of sub-class recognition and view change. A collection of low-level features allowed for recognition where view of object changed or part of object was obscured, by exploiting local neighbourhood properties. In addition, the combination of low-level and high-level features could boost the performance in some tasks by utilizing the advantages of both of them. In the second step, the minimum redundancy scheme aimed at an optimal feature set, which was formed from the extracted features in last step. Usually a substantial amount of low-level features was extracted. Outliers would be excluded in this step. On the other hand, because high-level features were generated based on the spatial information of objects, they would be ranked according to their performance in predication rather than be excluded.

9.3 Future work

There is significant potential to improve and extend the algorithms presented in this thesis. Assumptions such as a static background, which was required for the surveillance system in Chapters 5 and 6, limited the applications and robustness significantly. Background models worked poorly when the surveillance cameras were shaking. The surveillance systems in general could benefit from moving away from the use of background to more generic approaches. Full scene understanding and object recognition without strong assumptions are undoubtedly more difficult tasks, but emerging technology that masters these tasks will enable more applications.

Fig.9. 2 The influence of scene context on object recognition. The recognition of the highlighted pedestrian and the car become much easier with the consideration of scene context.

In human visual processing, knowledge of scene context has a tremendous effect on object recognition. Objects do typically not appear in isolation but interact actively or passively with the environment. In Figure 9.2, it is hard to recognise the pedestrian and the car without considering the scene. Once scene context is available such as the relative position of the objects to the road, the recognition becomes much easier. Most current ITSs do not consider an image as a whole but operate locally on the interesting objects such as pedestrian and vehicle. Significant performance boosts

could be expected from incorporating a model of contextual feedback on object recognition and hypotheses generation.

One of the strongest critiques of the traditional BoF was that it disregarded information regarding the spatial layout of the features. To address this problem, (Lazebnik, Schmid et al. 2006) proposed a spatial pyramid matching method, and (Yang, Yu et al. 2009) developed this method using sparse coding. Future work concerning integrating the spatial information of objects into our inference BoF method is necessary and highly encouraged.

Reference

Abdi, H., Valentin, D., Edelman, B. and O'Toole, A. J. (1995). "More about the difference between men and women: evidence from linear neural network and the principal-component approach." Perception **24**: 539-539.

Albiol, A., Silla, M. J. and Mossi, J. M. (2009). "Video analysis using corner motion statistics." Proceedings of IEEE International Workshop on Performance Evaluation of Tracking and Surveillance**:** 31-38.

Arai, S., Inoue, O. and Ozawa, S. (2010). "Following vehicle detection based on shift of feature plane using affine transform." Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on**:** 881-886.

Arbab-Zavar, B. and Nixon, M. S. (2011). "On guided model-based analysis for ear biometrics." Computer Vision and Image Understanding **115**(4): 487-502.

Arrospide, J., Salgado, L., Nieto, M. and Mohedano, R. (2010). "Homography-based ground plane detection using a single on-board camera." Intelligent Transport Systems, IET **4**(2): 149-160.

Balci, K. and Atalay, V. (2002). "Pca for gender estimation: which eigenvectors contribute?" Proceedings of the 16th International Conference on Pattern Recognition, IEEE. **3:** 363-366.

Baró, X., Escalera, S., Vitrià, J., Pujol, O. and Radeva, P. (2009). "Traffic sign recognition using evolutionary adaboost detection and forest-ECOC classification." IEEE Transactions on Intelligent Transportation Systems **10**(1): 113-126.

Bay, H., Ess, A., Tuytelaars, T. and Van Gool, L. (2008). "Speeded-up robust features (SURF)." Computer Vision and Image Understanding **110**(3): 346-359.

Bell, D. A. and Wang, H. (2000). "A formalism for relevance and its application in feature subset selection." Machine Learning **41**(2): 175-195.

Benfold, B. and Reid, I. (2009). "Guiding visual surveillance by tracking human attention." Proceedings of the 20th British Machine Vision Conference. **459**.

Benfold, B. and Reid, I. (2011). "Stable multi-target tracking in real-time surveillance video." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2011, IEEE**:** 3457-3464.

Bensrhair, A., Bertozzi, A., Broggi, A., Fascioli, A., Mousset, S. and Toulminet, G. (2002). "Stereo vision-based feature extraction for vehicle detection." Intelligent Vehicle Symposium, 2002. IEEE. **2:** 465-470 vol.462.

Bensrhair, A., Bertozzi, M., Broggi, A., Miche, P., Mousset, S. and Toulminet, G. (2001). "A cooperative approach to vision-based vehicle detection." Proceedings of Intelligent Transportation Systems, 2001, IEEE**:** 207-212.

Bertozzi, M., Broggi, A., Cellario, M., Fascioli, A., Lombardi, P. and Porta, M. (2002). "Artificial vision in road vehicles." Proceedings of the IEEE **90**(7): 1258-1271.

Besbes, B., Rogozan, A. and Bensrhair, A. (2010). "Pedestrian recognition based on hierarchical codebook of surf features in visible and infrared images." Proceedings of Intelligent Vehicles Symposium (IV), 2010, IEEE**:** 156-161.

Betke, M., Haritaoglu, E. and Davis, L. S. (2000). "Real-time multiple vehicle detection and tracking from a moving vehicle." Machine Vision and Applications **12**(2): 69-83.

Biesiada, J. and Duch, W. (2007). "Feature selection for high-dimensional data—a Pearson redundancy based filter." Computer Recognition Systems. Second Edition. Springer**:** 242-249.

Bishop, C. M. (2006). "Pattern recognition and machine learning." New York, NY, USA, Springer.

Boureau, Y., Ponce, J. and LeCun, Y. (2010). "A theoretical analysis of feature pooling in visual recognition." Proceedings of the 27th International Conference on Machine Learning, 2010**:** 111-118.

Broggi, A., Caraffi, C., Fedriga, R. I. and Grisleri, P. (2005). "Obstacle Detection with Stereo Vision for Off-Road Vehicle Navigation." Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on**:** 65-65.

Buch, N., Velastin, S. A. and Orwell, J. (2011). "A review of computer vision techniques for the analysis of urban traffic." IEEE Transactions on Intelligent Transportation Systems **12**(3): 920-939.

Bucher, T., Curio, C., Edelbrunner, J., Igel, C., Kastrup, D., Leefken, I., Lorenz, G., Steinhage, A. and von Seelen, W. (2003). "Image processing and behavior planning for intelligent vehicles." IEEE Transactions on Industrial Electronics **50**(1): 62-75.

Carey, J. W. (2009). "Communication as culture: essays on media and society." Revised. New York, NY, USA, Routledge.

Chang, C. C. and Lin, C. J. (2011). "LIBSVM: a library for support vector machines." ACM Transactions on Intelligent Systems and Technology (TIST) **2**(3): 27.

Chellappa, R., Gang, Q. and Qinfen, Z. (2004). "Vehicle detection and tracking using acoustic and video sensors." Proceedings of Acoustics, Speech, and Signal Processing (ICASSP '04), 2004, IEEE. **3:** 793-796.

Chen, X. (2003). "An improved branch and bound algorithm for feature selection." Pattern Recognition Letters **24**(12): 1925-1933.

Chu, J., Ji, L., Guo, L., Libibing and Wang, R. (2004). "Study on method of detecting preceding vehicle based on monocular camera." Intelligent Vehicles Symposium, 2004 IEEE**:** 750-755.

Conos, M. (2006). "Recognition of vehicle make from a frontal view." Czech Tech. Univ., Prague, Czech Republic. **Master**.

Cormen, T. H., Leiserson, C. E., Rivest, R. L. and Stein, C. (2001). "Introduction to algorithms." Third Edition. Cambridge, Massachusetts, USA, MIT press.

Cover, T. M. and Thomas, J. A. (2006). "Elements of information theory." Second Edition. Hoboken, NJ, USA, Wiley-interscience.

Cucchiara, R., Piccardi, M. and Mello, P. (2000). "Image analysis and rule-based reasoning for a traffic monitoring system." IEEE Transactions on Intelligent Transportation Systems **1**(2): 119-130.

Cui, X., Liu, Y., Shan, S., Chen, X. and Gao, W. (2007). "3d haar-like features for pedestrian detection." Proceedings of IEEE International Conference on Multimedia and Expo, 2007, IEEE**:** 1263-1266.

Cui, Y., Jin, J. S., Park, M., Luo, S., Xu, M., Peng, Y., Wong, W. S. F. and Santos, L. D. (2010). "Computer aided abnormality detection for microscopy images of cervical tissue." Proceedings of IEEE/ICME International Conference on Complex Medical Engineering, 2010 63-68.

Cui, Y., Luo, S., Tian, Q., Zhang, S., Peng, Y., Jiang, L. and Jin, J. (2013). "Mutual Information-Based Emotion Recognition." The Era of Interactive Media. Springer New York**:** 471-479.

Dalal, N. and Triggs, B. (2005). "Histograms of oriented gradients for human detection." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, IEEE. **1:** 886-893.

Damavandi, Y. B. and Mohammadi, K. (2004). "Speed limit traffic sign detection and recognition." Proceedings of IEEE Conference on Cybernetics and Intelligent Systems, 2004, IEEE. **2:** 797-802.

Das, S. (2001). "Filters, wrappers and a boosting-based hybrid for feature selection." Proceedings of International Conference Machine Learning, 2001**:** 74-81.

Dash, M., Choi, K., Scheuermann, P. and Liu, H. (2002). "Feature selection for clustering-a filter solution." Proceedings of IEEE International Conference on Data Mining, 2002, IEEE**:** 115-122.

Dash, M. and Liu, H. (1997). "Feature selection for classification." Intelligent Data Analysis **1**(1-4): 131-156.

Database. (2009). "PETS 2009 benchmark data." from http://www.cvg.rdg.ac.uk/PETS2009/a.html.

Dickmanns, E. D. (2002). "The development of machine vision for road vehicles in the last decade." Proceedings of IEEE Conference on Intelligent Vehicle Symposium, 2002. **1:** 268-281 vol.261.

Dimitrakopoulos, G. and Demestichas, P. (2010). "Intelligent transportation systems." Vehicular Technology Magazine, IEEE **5**(1): 77-84.

Du, Y. and Papanikolopoulos, N. P. (1997). "Real-time vehicle following through a novel symmetry-based approach." Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on. **4:** 3160-3165.

Duan, T. D., Duc, D. A. and Du, T. L. H. (2004). "Combining Hough transform and contour algorithm for detecting vehicles' license-plates." Proceedings of International Symposium on Intelligent Multimedia, Video and Speech Processing, 2004, IEEE**:** 747-750.

Dubuisson Jolly, M. P., Lakshmanan, S. and Jain, A. K. (1996). "Vehicle segmentation and classification using deformable templates." IEEE Transactions on Pattern Analysis and Machine Intelligence **18**(3): 293-308.

Dy, J. G. and Brodley, C. E. (2000). "Feature subset selection and order identification for unsupervised learning." Proceedings of International Conference on Machine Learning, 2000**:** 247-254.

Eade, E. and Drummond, T. (2009). "Edge landmarks in monocular SLAM." Image and Vision Computing **27**(5): 588-596.

Enzweiler, M. and Gavrila, D. M. (2008). "A mixed generative-discriminative framework for pedestrian classification." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2008**:** 1-8.

Enzweiler, M. and Gavrila, D. M. (2009). "Monocular pedestrian detection: survey and experiments." IEEE Transactions on Pattern Analysis and Machine Intelligence **31**(12): 2179-2195.

Etemad, K. and Chellappa, R. (1997). "Discriminant analysis for recognition of human face images." Journal of Optical Society of America A **14**(8): 1724-1733.

Fan, X., Sun, Y., Yin, B. and Guo, X. (2010). "Gabor-based dynamic representation for human fatigue monitoring in facial image sequences." Pattern Recognition Letters **31**(3): 234-243.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2005). "Pictorial structures for object recognition." International Journal of Computer Vision **61**(1): 55-79.

Feris, R. S., Siddiquie, B., Petterson, J., Yun, Z., Datta, A., Brown, L. M. and Pankanti, S. (2012). "Large-Scale Vehicle Detection, Indexing, and Search in Urban Surveillance Videos." IEEE Transactions on Multimedia **14**(1): 28-42.

Figueiredo, L., Jesus, I., Machado, J. A. T., Ferreira, J. R. and Martins de Carvalho, J. L. (2001). "Towards the development of intelligent transportation systems." Proceedings of IEEE Conference on Intelligent Transportation Systems, 2001**:** 1206-1211.

Fischler, M. A. and Bolles, R. C. (1981). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." Communications of the ACM **24**(6): 381-395.

Fischler, M. A. and Elschlager, R. A. (1973). "The representation and matching of pictorial structures." IEEE Transactions on Computers **100**(1): 67-92.

Furugori, S., Yoshizawa, N., Iname, C. and Miura, Y. (2005). "Estimation of driver fatigue by pressure distribution on seat in long term driving." Review of Automotive Engineering **26**(1): 053-058.

Gao, S., Tsang, I. W., Chia, L. and Zhao, P. (2010). "Local features are not lonely-Laplacian sparse coding for image classification." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010**:** 3555-3561.

García-Garrido, M., Sotelo, M. and Martín-Gorostiza, E. (2005). "Fast road sign detection using Hough transform for assisted driving of road vehicles." Computer Aided Systems Theory–EUROCAST 2005: 543-548.

Gelmose, A., Trivedi, M. M. and Moeslund, T. B. (2012). "Vision-based traffic sign detection and analysis for intelligent driver assistance systems: perspectives and survey." IEEE Transactions on Intelligent Transportation Systems(99): 1-14.

Gonzalez, R. C. and Richard, E. (2002). "Digital image processing." Third Edition. New Jersey, USA, Prentice Hall Press.

Gupta, P., Doermann, D. and DeMenthon, D. (2002). "Beam search for feature selection in automatic svm defect classification." Proceedings of the 16th International Conference on Pattern Recognition, 2002, IEEE. **2:** 212-215.

Gupte, S., Masoud, O., Martin, R. F. K. and Papanikolopoulos, N. P. (2002). "Detection and classification of vehicles." IEEE Transactions on Intelligent Transportation Systems **3**(1): 37-47.

Guyon, I. and Elisseeff, A. (2003). "An introduction to variable and feature selection." The Journal of Machine Learning Research **3**: 1157-1182.

Haag, M. and Nagel, H.-H. (1999). "Combination of edge element and optical flow estimates for 3D-model-based vehicle tracking in traffic image sequences." International Journal of Computer Vision **35**(3): 295-319.

Hall, M. A. (1999). "Correlation-based feature selection for machine learning." The University of Waikato, New Zealand. **PhD**.

Hall, M. A. (2000). "Correlation-based feature selection for discrete and numeric class machine learning." Proceedings of the 17th International Conference on Machine Learnin, Stanford, CA, Morgan Kaufmann, San Francisco**:** 359-366.

Hancock, J., Hoffman, E., Sullivan, R. M., Ingimarson, D., Langer, D. and Hebert, M. (1998). "High-performance laser range scanner." Intelligent Systems & Advanced Manufacturing. International Society for Optics and Photonics**:** 40-49.

Handmann, U., Kalinke, T., Tzomakas, C., Werner, M. and Seelen, W. (2000). "An image processing system for driver assistance." Image and Vision Computing **18**(5): 367-376.

Harris, C. and Stephens, M. (1988). "A combined corner and edge detector." Proceedings of Alvey Vision Conference, 1988, Manchester, UK. **15:** 50.

Hartigan, J. A. and Wong, M. A. (1979). "Algorithm AS 136: A k-means clustering algorithm." Applied statistics: 100-108.

Hartley, R. and Zisserman, A. (2003). "Multiple view geometry in computer vision." Second Edition. New York, NY, USA, Cambridge University Press.

Haselhoff, A. and Kummert, A. (2009). "A vehicle detection system based on haar and triangle features." Proceedings of IEEE Conference on Intelligent Vehicle Symposium, 2009, IEEE**:** 261-266.

Hebert, M. (2000). "Active and passive range sensing for robotics." Proceedings of IEEE International Conference on Robotics and Automation, 2000. **1:** 102-110.

Hinz, S., Schlosser, C. and Reitberger, J. (2003). "Automatic car detection in high resolution urban scenes based on an adaptive 3D-model." Proceedings of 2nd GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas, 2003, IEEE**:** 167-171.

Hoferlin, B. and Zimmermann, K. (2009). "Towards reliable traffic sign recognition." Proceedings of IEEE Conference on Intelligent Vehicle Symposium, 2009, IEEE**:** 324-329.

Hoffman, C., Dang, T. and Stiller, C. (2004). "Vehicle detection fusing 2D visual features." Intelligent Vehicles Symposium, 2004 IEEE**:** 280-285.

Hommels, A. (2005). "Studying obduracy in the city: toward a productive fusion between technology studies and urban studies." Science, Technology, & Human Values **30**(3): 323-351.

Hoos, H. H. and Stützle, T. (2004). "Stochastic local search: foundations & applications." First Edition. Burlington, Massachusetts, USA, Morgan Kaufmann.

Hough, P. V. C. (1962). "Method and means for recognizing complex patterns." Google Patents **No. 3,069,654**.U. S. P. a. T. Office.

Hsieh, C., Juan, Y. and Hung, K. (2005). "Multiple license plate detection for complex background." Proceedings of the 19th International Conference on Advanced Information Networking and Applications, 2005, IEEE. **2:** 389-392.

Hsieh, J., Yu, S., Chen, Y. and Hu, W. (2006). "Automatic traffic surveillance system for vehicle tracking and classification." IEEE Transactions on Intelligent Transportation Systems **7**(2): 175-187.

Hsu, S. H. and Huang, C. L. (2001). "Road sign detection and recognition using matching pursuit method." Image and Vision Computing **19**(3): 119-129.

Hua, J., Tembe, W. and Dougherty, E. R. (2008). "Feature selection in the classification of high-dimension data." Proceedings of IEEE International Workshop on Genomic Signal Processing and Statistics, 2008, IEEE**:** 1-2.

Huan, L. and Lei, Y. (2005). "Toward integrating feature selection algorithms for classification and clustering." IEEE Transactions on Knowledge and Data Engineering **17**(4): 491-502.

Huber, D., Kapuria, A., Donamukkala, R. and Hebert, M. (2004). "Parts-based 3d object classification." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004, IEEE. **2:** 82-89.

Jain, A., Huang, J. and Shiaofen, F. (2005). "Gender identification using frontal facial images." IEEE International Conference on Multimedia and Expo, 2005**:** 4-10.

Jain, A. and Zongker, D. (1997). "Feature selection: evaluation, application, and small sample performance." IEEE Transactions on Pattern Analysis and Machine Intelligence **19**(2): 153-158.

Jazayeri, A., Hongyuan, C., Jiang Yu, Z. and Tuceryan, M. (2011). "Vehicle Detection and Tracking in Car Video Based on Motion Model." IEEE Transactions on Intelligent Transportation Systems **12**(2): 583-595.

Ji, P., Jin, L. and Li, X. (2007). "Vision-based vehicle type classification using partial gabor filter bank." Proceedings of IEEE International Conference on Automation and Logistics, 2007, IEEE**:** 1037-1040.

Jia, W., Zhang, H. and He, X. (2007). "Region-based license plate detection." Journal of Network and Computer Applications **30**(4): 1324-1333.

Jiang, Y., Ngo, C. and Yang, J. (2007). "Towards optimal bag-of-features for object categorization and semantic video retrieval." Proceedings of the 6th ACM International Conference on Image and Video Retrieval. Amsterdam, The Netherlands, ACM**:** 494-501.

Jin, X., Xu, A., Bie, R. and Guo, P. (2006). "Machine learning techniques and chi-square feature selection for cancer classification using SAGE gene expression profiles." Data Mining for Biomedical Applications. Springer**:** 106-115.

Jirawimut, R., Prakoonwit, S., Cecelja, F. and Balachandran, W. (2003). "Visual odometer for pedestrian navigation." IEEE Transactions on Instrumentation and Measurement **52**(4): 1166-1173.

John, G. H., Kohavi, R. and Pfleger, K. (1994). "Irrelevant features and the subset selection problem." Proceedings of the 11th International Conference on Machine Learning, San Francisco. **129:** 121-129.

Jolliffe, I. (2005). "Principal component analysis." Second Edition. Hoboken, New Jersey, USA, John Wiley & Sons, Ltd.

Jones, W. D. (2002). "Building safer cars." Spectrum, IEEE **39**(1): 82-85.

Kafai, M. and Bhanu, B. (2012). "Dynamic Bayesian networks for vehicle classification in video." IEEE Transactions on Industrial Informatics **8**(1): 100-109.

Kastrinaki, V., Zervakis, M. and Kalaitzakis, K. (2003). "A survey of video processing techniques for traffic applications." Image and Vision Computing **21**(4): 359-381.

Kawano, T., Ban, Y. and Uehara, K. (2003). "A coded visual marker for video tracking system based on structured image analysis." Mixed and Augmented Reality, 2003. Proceedings. The Second IEEE and ACM International Symposium on**:** 262-263.

Ke, Y. and Sukthankar, R. (2004). "PCA-SIFT: A more distinctive representation for local image descriptors." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004, IEEE. **2:** 506-513.

Kearns, M. J. and Mansour, Y. (1998). "A fast, bottom-up decision tree pruning algorithm with near-optimal generalization." Proceedings of the 15th International Conference on Machine learning, Madison, Wisconsin, USA. **98:** 269-277.

Ki, Y. K. and Baik, D. K. (2006). "Vehicle-classification algorithm for single-loop detectors using neural networks." IEEE Transactions on Vehicular Technology **55**(6): 1704-1711.

Kilambi, P., Ribnick, E., Joshi, A. J., Masoud, O. and Papanikolopoulos, N. (2008). "Estimating pedestrian counts in groups." Computer Vision and Image Understanding **110**(1): 43-59.

Kim, Y., Street, W. N. and Menczer, F. (2000). "Feature selection in unsupervised learning via evolutionary search." Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM**:** 365-369.

Klein, G. and Murray, D. (2007). "Parallel Tracking and Mapping for Small AR Workspaces." Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on**:** 225-234.

Kluge, K. (1994). "Extracting road curvature and orientation from image edge points without perceptual grouping into features." Proceedings of IEEE Conference on Intelligent Vehicle Symposium, 1994**:** 109-114.

Kohavi, R. and John, G. H. (1997). "Wrappers for feature subset selection." Artificial Intelligence **97**(1): 273-324.

Kong, D., Gray, D. and Tao, H. (2005). "Counting pedestrians in crowds using viewpoint invariant training." Proceedings of the 16th British Machine Vision Conference, 2005, Oxford U.K.

Kovesi, P. (2000). "Phase congruency: A low-level image invariant." Psychological Research **64**(2): 136-148.

Kumar, P., Mittal, A. and Kumar, P. (2008). "Study of robust and intelligent surveillance in visible and multi-modal framework." Informatica (Slovenia) **28**(1): 63.

Kuwabara, K., Yano, Y. and Okuma, S. (2009). "Vehicle Appearance Model for Recognition system considering The Change of Imaging Condition." Journal of Advanced Computational Intelligence and Intelligent Informatics **13**: 463-469.

Kwak, N. and Choi, C. (2002). "Input feature selection by mutual information based on Parzen window." IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(12): 1667-1671.

Lai, A. H. S., Fung, G. S. K. and Yung, N. H. C. (2001). "Vehicle type classification from visual-based dimension estimation." Proceedings of IEEE Conference on Intelligent Transportation Systems, 2001, IEEE**:** 201-206.

Lazebnik, S., Schmid, C. and Ponce, J. (2006). "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories." Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on. **2:** 2169-2178.

Le, Q. V., Ranzato, M. A., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J. and Ng, A. Y. (2012). "Building high-level features using large scale unsupervised learning." Proceedings of the 29th International Conference on Machine Learning, 2012, Edinburgh, Scotland, GB**:** 81--88.

Lee, H. (2006). "Neural network approach to identify model of vehicles." Advances in Neural Networks-ISNN 2006: 66-72.

Lee, H., Battle, A., Raina, R. and Ng, A. Y. (2007). "Efficient Sparse Coding Algorithms." Advances in Neural Information Processing Systems (19): 801--808.

Li, F.-F. and Pietro, P. (2005). "A Bayesian hierarchical model for learning natural scene categories." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. **2:** 524-531.

Li, M., Zhang, Z., Huang, K. and Tan, T. (2008). "Estimating the number of people in crowded scenes by MID based foreground segmentation and head-shoulder detection." Proceedings of 19th International Conference on Pattern Recognition, 2008 IEEE**:** 1-4.

Liang, W., Tieniu, T., Huazhong, N. and Weiming, H. (2003). "Silhouette analysis-based gait recognition for human identification." Pattern Analysis and Machine Intelligence, IEEE Transactions on **25**(12): 1505-1518.

Liao, C., Li, S. and Luo, Z. (2007). "Gene selection using wilcoxon rank sum test and support vector machine for cancer classification." Computational Intelligence and Security. Springer**:** 57-66.

Liu, H., Dougherty, E. R., Dy, J. G., Torkkola, K., Tuv, E., Peng, H., Ding, C., Long, F., Berens, M. and Parsons, L. (2005). "Evolving feature selection." Intelligent systems, IEEE **20**(6): 64-76.

Liu, H., Mei, T., Luo, J., Li, H. and Li, S. (2012). "Finding Perfect Rendezvous On the Go: Accurate Mobile Visual Localization and Its Applications to Routing." Proceedings of ACM International Conference on Multimedia, 2012, Nara, Japan**:** 9-18.

Liu, H. and Motoda, H. (1998). "Feature selection for knowledge discovery and data mining." Kluwer International Series in Engineering and Computer Science Norwell, MA, USA, Kluwer Academic Publishers.

Liu, H. and Yu, L. (2005). "Toward integrating feature selection algorithms for classification and clustering." IEEE Transactions on Knowledge and Data Engineering **17**(4): 491-502.

Liu, Z. and Zhang, Z. (2011). "Face geometry and appearance modeling." First Edition. Cambridge, United Kingdom, Cambridge University Press.

Lowe, D. G. (2004). "Distinctive image features from scale-invariant keypoints." International Journal of Computer Vision **60**(2): 91-110.

Loy, G. and Eklundh, J. (2006). "Detecting symmetry and symmetric constellations of features." Proceedings of the 9th European conference on Computer Vision - Volume Part II. Graz, Austria, Springer-Verlag**:** 508-521.

Ma, X. and Grimson, W. E. L. (2005). "Edge-based rich representation for vehicle classification." Proceedings of the 10th IEEE International Conference on Computer Vision, 2005, IEEE. **2:** 1185-1192.

Mao, K. (2004). "Feature subset selection for support vector machines through discriminative function pruning analysis." IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics **34**(1): 60-67.

Matthews, N. D., An, P. E., Charnley, D. and Harris, C. J. (1996). "Vehicle detection and recognition in greyscale imagery." Control Engineering Practice **4**(4): 473-479.

Mitra, P., Murthy, C. and Pal, S. K. (2002). "Unsupervised feature selection using feature similarity." IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(3): 301-312.

Mondragón, I. F., Campoy, P., Martinez, C. and Olivares-Méndez, M. A. (2010). "3D pose estimation based on planar object tracking for UAVs control." Robotics and Automation (ICRA), 2010 IEEE International Conference on, Ieee**:** 35-41.

Monteiro, G., Peixoto, P. and Nunes, U. (2006). "Vision-based pedestrian detection using Haar-like features." Robotica(67): 16-20.

Moré, J. (1978). "The Levenberg-Marquardt algorithm: implementation and theory." Numerical Analysis. Springer Berlin Heidelberg. **630:** 105-116.

Morel, J. M. and Yu, G. (2009). "ASIFT: A new framework for fully affine invariant image comparison." SIAM Journal on Imaging Sciences **2**(2): 438-469.

Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F. and Sayd, P. (2006). "Real Time Localization and 3D Reconstruction." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006. **1:** 363-370.

Münz, G., Li, S. and Carle, G. (2007). "Traffic anomaly detection using k-means clustering." Proceedings of Performance, Reliability and Dependability Evaluation of Communication Networks and Distributed Systems, 4 GI / ITG Workshop MMBnet. Hamburg, Germany. **4**.

Narendra, P. M. and Fukunaga, K. (1977). "A branch and bound algorithm for feature subset selection." IEEE Transactions on Computers **100**(9): 917-922.

Negri, P., Clady, X., Hanif, S. M. and Prevost, L. (2008). "A cascade of boosted generative and discriminative classifiers for vehicle detection." EURASIP Journal on Advances in Signal Processing **2008**: 136-146.

Negri, P., Clady, X., Hanif, S. M. and Prevost, L. (2008). "A Cascade of Boosted Generative and Discriminative Classifiers for Vehicle Detection." EURASIP Journal on Advances in Signal Processing **2008**.

Nistér, D., Naroditsky, O. and Bergen, J. (2006). "Visual odometry for ground vehicle applications." Journal of Field Robotics **23**(1): 3-20.

Nixon, M. S. and Aguado, A. S. (2012). "Feature extraction & image processing for computer vision." Third Edition. Oxford, UK, Elsevier Science.

Nvidia, C. (2012) "C best practices guide." DOI: http://docs.nvidia.com/cuda/cuda-c-best-practices-guide/index.html.

O'Malley, R., Jones, E. and Glavin, M. (2010). "Rear-Lamp Vehicle Detection and Tracking in Low-Exposure Color Video for Night Conditions." Intelligent Transportation Systems, IEEE Transactions on **11**(2): 453-462.

O'Toole, A. J., Abdi, H., Deffenbacher, K. A. and Valentin, D. (1993). "Low-dimensional representation of faces in higher dimensions of the face space." Journal of Optical Society of America A **10**(3): 405-411.

Ohbuchi, R. and Furuya, T. (2009). "Scale-weighted dense bag of visual features for 3D model retrieval from a partial view 3D model." Proceedings of IEEE 12th International Conference on Computer Vision Workshops, 2009**:** 63-70.

Optics, L. B. (2012). "Light Speed™ enhances road safety and changes the rules of automotive design." Retrieved 21/08/2013, from http://lightblueoptics.com/products/light-speed/.

Otsu, N. (1975). "A threshold selection method from gray-level histograms." Automatica **11**(285-296): 23-27.

Paetzold, F. and Franke, U. (2000). "Road recognition in urban environment." Image and Vision Computing **18**(5): 377-387.

Pai, C., Tyan, H., Liang, Y., Liao, H. M. and Chen, S. (2004). "Pedestrian detection and tracking at crossroads." Pattern Recognition **37**(5): 1025-1034.

Papadimitratos, P., La Fortelle, A., Evenssen, K., Brignolo, R. and Cosenza, S. (2009). "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation." Communications Magazine, IEEE **47**(11): 84-95.

Park, M., Jin, J. S., Peng, Y., Summons, P., Yu, D., Cui, Y., Luo, S., Wang, F., Santos, L. and Xu, M. (2010). "Automatic cell segmentation in microscopic color images using ellipse fitting and watershed." Proceedings of IEEE/ICME International Conference on Complex Medical Engineering, 2010**:** 69-74.

Park, S. J., Kim, T. Y., Kang, S. M. and Koo, K. H. (2003). "A novel signal processing technique for vehicle detection radar." Proceedings of IEEE MTT-S International Microwave Symposium Digest, 2003. **1:** 607-610.

Peijin, J., Lianwen, J. and Xutao, L. (2007). "Vision-based Vehicle Type Classification Using Partial Gabor Filter Bank." Automation and Logistics, 2007 IEEE International Conference on**:** 1037-1040.

Peng, H., Long, F. and Ding, C. (2005). "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy." IEEE Transactions on Pattern Analysis and Machine Intelligence **27**(8): 1226-1238.

Peng, Y., Jin, J., Luo, S. and Park, M. (2011). "Understanding video sequences through super-resolution." Advances in Multimedia Modeling. Springer Berlin Heidelberg. **6524:** 25-34.

Peng, Y., Jin, J., Luo, S., Xu, M., Au, S., Zhang, Z. and Cui, Y. (2013). "Vehicle type classification using data mining techniques." The Era of Interactive Media. Springer New York**:** 325-335.

Peng, Y., Jin, J. S., Luo, S. and Xu, M. (2010). "Learning priors for super-resolution in video sequence." Proceedings of the 2nd International Conference on Internet Multimedia Computing and Service. Harbin, China, ACM**:** 163-166.

Peng, Y., Jin, J. S., Luo, S., Xu, M. and Cui, Y. (2012). "3D pose estimation of front vehicle towards a better driver assistance system." Proceedings of IEEE International Conference on Multimedia and Expo Workshops, 2012 522-527.

Peng, Y., Jin, J. S., Luo, S., Xu, M. and Cui, Y. (2012). "Vehicle type classification using PCA with self-clustering." Proceedings of IEEE International Conference on Multimedia and Expo Workshops, 2012 384-389.

Peng, Y., Luo, S., Xu, M., Ni, Z., Jin, J. S., Wang, J. and Zhao, G. (2012). "Bag of features using sparse coding for gender classification." Proceedings of the 4th International Conference on Internet Multimedia Computing and Service. Wuhan, China, ACM**:** 80-83.

Peng, Y., Park, M., Xu, M., Luo, S., Jin, J. S., Cui, Y., Wong, W. F. and Santos, L. D. (2010). "Clustering nuclei using machine learning techniques." Proceedings of IEEE/ICME International Conference on Complex Medical Engineering, 2010, IEEE**:** 52-57.

Peng, Y., Park, M., Xu, M., Luo, S., Jin, J. S., Cui, Y. and Wong, W. S. F. (2010). "Detection of nuclei clusters from cervical cancer microscopic imagery using C4.5." Proceedings of the 2nd International Conference on Computer Engineering and Technology, 2010 **3:** 593-597.

Peng, Y., Xu, M., Jin, J. S., Luo, S. and Zhao, G. (2011). "Cascade-based license plate localization with line segment features and Haar-Like features." Proceedings of the 6th International Conference on Image and Graphics, 2011**:** 1023-1028.

Peng, Y., Xu, M., Ni, Z., Jin, J. and Luo, S. (2012). "Accurate pedestrian counting system based on local features." Advances in Multimedia Information Processing – PCM 2012. Springer Berlin Heidelberg. **7674:** 850-860.

Peng, Y., Xu, M., Ni, Z., Jin, J. S. and Luo, S. (2012). "Combining front vehicle detection with 3D pose estimation for a better driver assistance." International Journal of Advanced Robotic Systems **9**: 93-108.

Petrovic, V. and Cootes, T. (2004). "Analysis of features for rigid structure vehicle type recognition." Proceedings of British Machine Vision Conference, 2004. **2:** 587-596.

Philbin, J., Chum, O., Isard, M., Sivic, J. and Zisserman, A. (2008). "Lost in quantization: Improving particular object retrieval in large scale image databases." Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2008**:** 1-8.

Phillips, P. J., Moon, H., Rizvi, S. A. and Rauss, P. J. (2000). "The FERET evaluation methodology for face-recognition algorithms." IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(10): 1090-1104.

Prati, A., Mikic, I., Trivedi, M. M. and Cucchiara, R. (2003). "Detecting moving shadows: algorithms and evaluation." IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(7): 918-923.

Prisacariu, V. and Reid, I. (2009). "fastHOG-a real-time GPU implementation of HOG." University of Oxford Technical Report **2310**(09).

Psyllos, A., Anagnostopoulos, C. and Kayafas, E. (2011). "Vehicle model recognition from frontal view image measurements." Computer Standards Interfaces **33**(2): 142-151.

Pudil, P., Novovičová, J. and Kittler, J. (1994). "Floating search methods in feature selection." Pattern Recognition Letters **15**(11): 1119-1125.

Rahmalan, H., Nixon, M. S. and Carter, J. N. (2006). "On crowd density estimation for surveillance." Proceedings of the Institution of Engineering and Technology Conference on Crime and Security, 2006, IET**:** 540-545.

Robnik-Šikonja, M. and Kononenko, I. (2003). "Theoretical and empirical analysis of ReliefF and RReliefF." Machine Learning **53**(1-2): 23-69.

Rocchi, L., Chiari, L. and Cappello, A. (2004). "Feature selection of stabilometric parameters based on principal component analysis." Medical and Biological Engineering and Computing **42**(1): 71-79.

Rodriguez, M. D. and Shah, M. (2007). "Detecting and segmenting humans in crowded scenes." Proceedings of the 15th international conference on Multimedia, ACM**:** 353-356.

Rosenberg, C. (2001). "The Lenna story: imaging experts meet Lenna in person." Retrieved 21/08/2013, from CMU.edu.

Rosten, E. and Drummond, T. (2006). "Machine Learning for High-Speed Corner Detection Computer Vision – ECCV 2006." Springer Berlin / Heidelberg. **3951:** 430-443.

Ryan, D., Denman, S., Fookes, C. and Sridharan, S. (2009). "Crowd counting using multiple local features." Proceedings of Digital Image Computing: Techniques and Applications, 2009, IEEE**:** 81-88.

Ryan, D., Denman, S., Fookes, C. B. and Sridharan, S. (2010). "Crowd counting using group tracking and local features." Proceedings of the 7th IEEE International Conference on Advanced Video and Signal Based Surveillance, 2010, IEEE**:** 218-224.

Ryan, W. T., Daniel, H. F., Luiz, A. D. and Allen, B. M. (2006). "Cognitive networks: adaptation and learning to achieve end-to-end performance objectives." Communications Magazine, IEEE **44**(12): 51-57.

Saeys, Y., Inza, I. and Larrañaga, P. (2007). "A review of feature selection techniques in bioinformatics." Bioinformatics **23**(19): 2507-2517.

SamYong, K., Se-Young, O., JeongKwan, K., YoungWoo, R., Kwangsoo, K., Sang-Cheol, P. and KyongHa, P. (2005). "Front and rear vehicle detection and tracking in the day and night times using vision and sonar sensor fusion." Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on**:** 2173-2178.

Scaramuzza, D., Fraundorfer, F. and Siegwart, R. (2009). "Real-time monocular visual odometry for on-road vehicles with 1-point ransac." Proceedings of IEEE International Conference on Robotics and Automation, 2009, IEEE**:** 4293-4299.

Schmid, C., Mohr, R. and Bauckhage, C. (2000). "Evaluation of interest point detectors." International Journal of Computer Vision **37**(2): 151-172.

Schneiderman, H. and Kanade, T. (2004). "Object detection using the statistics of parts." International Journal of Computer Vision **56**(3): 151-177.

Schweighofer, G. and Pinz, A. (2006). "Robust Pose Estimation from a Planar Target." IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(12): 2024-2030.

Sebastiani, F. (2002). "Machine learning in automated text categorization." ACM computing surveys (CSUR) **34**(1): 1-47.

Shan, Y., Sawhney, H. S. and Kumar, R. (2005). "Unsupervised learning of discriminative edge measures for vehicle matching between non-overlapping cameras." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, IEEE. **1:** 894-901.

Sichitiu, M. L. and Kihl, M. (2008). "Inter-vehicle communication systems: a survey." Communications Surveys & Tutorials, IEEE **10**(2): 88-105.

Siedlecki, W. and Sklansky, J. (1989). "A note on genetic algorithms for large-scale feature selection." Pattern Recognition Letters **10**(5): 335-347.

Sivaraman, S. and Trivedi, M. M. (2010). "A general active-learning framework for on-road vehicle recognition and tracking." IEEE Transactions on Intelligent Transportation Systems **11**(2): 267-276.

Sivic, J. and Zisserman, A. (2003). "Video Google: a text retrieval approach to object matching in videos." Proceedings of the 9th IEEE International Conference on Computer Vision, 2003. **2:** 1470-1477.

Soetedjo, A. and Yamada, K. (2005). "Fast and robust traffic sign detection." Proceedings of IEEE International Conference on Systems, Man and Cybernetics, 2005. **2:** 1341-1346.

Stewénius, H., Engels, C. and Nistér, D. (2006). "Recent developments on direct relative orientation." ISPRS Journal of Photogrammetry and Remote Sensing **60**(4): 284-294.

Sun, Y., Todorovic, S. and Goodison, S. (2010). "Local-learning-based feature selection for high-dimensional data analysis." IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(9): 1610-1626.

Sun, Z., Bebis, G. and Miller, R. (2004). "Object detection using feature subset selection." Pattern Recognition **37**(11): 2165-2176.

Sun, Z., Bebis, G. and Miller, R. (2006). "On-road vehicle detection: A review." IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(5): 694-711.

Sun, Z., Miller, R., Bebis, G. and DiMeo, D. (2002). "A real-time precrash vehicle detection system." Proceedings of the 6th IEEE Workshop on Applications of Computer Vision, 2002**:** 171-176.

Tai, J.-C., Tseng, S.-T., Lin, C.-P. and Song, K.-T. (2004). "Real-time image tracking for automatic traffic monitoring and enforcement applications." Image and Vision Computing **22**(6): 485-501.

Tan, F., Fu, X., Wang, H., Zhang, Y. and Bourgeois, A. (2006). "A hybrid feature selection approach for microarray gene expression data." Computational Science–ICCS 2006. Springer**:** 678-685.

Toews, M. and Arbel, T. (2006). "Detection Over Viewpoint via the Object Class Invariant." Proceedings of the 18th International Conference on Pattern Recognition IEEE Computer Society. **18:** 765-768.

Toews, M. and Arbel, T. (2009). "Detection, localization, and sex classification of faces from arbitrary viewpoints and under occlusion." IEEE Transactions on Pattern Analysis and Machine Intelligence **31**(9): 1567-1581.

Turk, M. and Pentland, A. (1991). "Eigenfaces for recognition." Journal of Cognitive Neuroscience **3**(1): 71-86.

Tzomakas, C. and Seelen, W. v. (1998). " Vehicle detection in traffic scenes using shadows." Technical Report 98-06, IR-INI, Institut fur Nueroinformatik, Ruhr-Universitat Bochum, FRG, . Germany.

Urazghildiiev, I., Ragnarsson, R., Ridderstrom, P., Rydberg, A., Ojefors, E., Wallin, K., Enochsson, P., Ericson, M. and Lofqvist, G. (2007). "Vehicle classification based on the radar measurement of height profiles." IEEE Transactions on Intelligent Transportation Systems **8**(2): 245-253.

Valentin, D. and Abdi, H. (1996). "Can a linear autoassociator recognize faces from new orientations?" Journal of Optical Society of America A **13**(4): 717-724.

Valera, M. and Velastin, S. A. (2005). "Intelligent distributed surveillance systems: a review." Proceedings of IEE Conference on Vision, Image and Signal Processing, 2005 **152**(2): 192-204.

Vargas, M., Milla, J. M., Toral, S. L. and Barrero, F. (2010). "An Enhanced Background Estimation Algorithm for Vehicle Detection in Urban Traffic Scenes." IEEE Transactions on Vehicular Technology **59**(8): 3694-3709.

Vasconcelos, M. and Vasconcelos, N. (2009). "Natural image statistics and low-complexity feature selection." IEEE Transactions on Pattern Analysis and Machine Intelligence **31**(2): 228-244.

Verschae, R., Ruiz-del-Solar, J. and Correa, M. (2006). "Gender classification of faces using Adaboost." Progress in Pattern Recognition, Image Analysis and Applications. Springer Berlin Heidelberg. **4225:** 68-78.

Viola, P. and Jones, M. (2001). "Rapid object detection using a boosted cascade of simple features." Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001, IEEE. **1:** I-511-I-518 vol. 511.

Viola, P. and Jones, M. J. (2004). "Robust real-time face detection." International Journal of Computer Vision **57**(2): 137-154.

Wang, C.-C., Thorpe, C. and Suppe, A. (2003). "LADAR-based detection and tracking of moving objects from a ground vehicle at high speeds." Proceedings of Intelligent Vehicles Symposium, 2003, IEEE**:** 416-421.

Wang, C. R. and Lien, J. (2008). "Automatic vehicle detection using local features—A statistical approach." IEEE Transactions on Intelligent Transportation Systems **9**(1): 83-96.

Wang, F. Y., Herget, C. and Zeng, D. (2005). "Guest Editorial Developing and Improving Transportation Systems: The Structure and Operation of IEEE Intelligent Transportation Systems Society." IEEE Transactions on Intelligent Transportation Systems **6**(3): 261-264.

Wang, J. M., Chung, Y. C., Chang, C. and Chen, S. W. (2004). "Shadow detection and removal for traffic images." Proceedings of IEEE International Conference on Networking, Sensing and Control, 2004, IEEE. **1:** 649-654.

Wang, S., Cui, L., Liu, D., Huck, R., Verma, P., Sluss, J. and Cheng, S. (2012). "Vehicle Identification Via Sparse Representation." IEEE Transactions on Intelligent Transportation Systems **13**(2): 955-962.

Wang, S. Z. and Lee, H. J. (2007). "A cascade framework for a real-time statistical plate recognition system." IEEE Transactions on Information Forensics and Security **2**(2): 267-282.

Wang, Y., Zou, Y., Shi, H. and Zhao, H. (2009). "Video image vehicle detection system for signaled traffic intersection." Proceedings of the 9th International Conference on Hybrid Intelligent Systems, 2009, IEEE. **1:** 222-227.

Webb, A. R. (2003). "Statistical pattern recognition." Third Edition. Hoboken, NJ, USA, Wiley.

WebSource. (2012). "Kapsch TrafficCom." Retrieved 21/08/2013, from http://www.kapsch.net/en/Pages/default.aspx.
WebSource. (2012). "Traficon." Retrieved 21/08/2013, from http://traficon.com/.
WebSource. (2012). "Virage." Retrieved 21/08/2013, from http://www.virage.com/.
Wei, L., XueZhi, W., Bobo, D., Huai, Y. and Nan, W. (2007). "Rear Vehicle Detection and Tracking for Lane Change Assist." Intelligent Vehicles Symposium, 2007 IEEE**:** 252-257.
Wei, W., Zhang, Q. and Wang, M. (2001). "A method of vehicle classification using models and neural networks." Proceedings of the 53rd IEEE Vehicular Technology Conference, 2001, IEEE. **4:** 3022-3026.
Wood, D. and Graham, S. (2006). "Permeable boundaries in the software-sorted society: Surveillance and the differentiation of mobility." Mobile Technologies of The City: 177-191.
Wu, B., Ai, H. and Huang, C. (2003). "LUT-based Adaboost for gender classification." Proceedings of the 4th International Conference on Audio and Video based Biometric Person Authentication. Guildford, UK, Springer-Verlag**:** 104-110.
Wu, B. and Juang, J. (2012). "Adaptive Vehicle Detector Approach for Complex Environments." IEEE Transactions on Intelligent Transportation Systems **13**(2): 817-827.
Xie, Y., Liu, L., Li, C. and Qu, Y. (2009). "Unifying visual saliency with HOG feature learning for traffic sign detection." Proceedings of IEEE Conference on Intelligent Vehicle Symposium, 2009, IEEE**:** 24-29.
Xing, E. P., Jordan, M. I. and Karp, R. M. (2001). "Feature selection for high-dimensional genomic microarray data." Proceedings of International Conference on Machine Learning, 2001**:** 601-608.
Xu, M., He, X., Jin, J., Peng, Y., Xu, C. and Guo, W. (2011). "Using scripts for affective content retrieval." Advances in Multimedia Information Processing - PCM 2010. Springer Berlin Heidelberg. **6298:** 43-51.
Xu, M., He, X., Peng, Y., Jin, J., Luo, S., Chia, L.-T. and Hu, Y. (2012). "Content on demand video adaptation based on MPEG-21 digital item adaptation." EURASIP Journal on Wireless Communications and Networking **2012**(1): 104.
Yambor, W. S., Draper, B. A. and Beveridge, J. R. (2002). "Analyzing pca-based face recognition algorithms: Eigenvector selection and distance measures." Empirical Evaluation Methods in Computer Vision. Singapore, World Scientific. **50:** 39-60.
Yan, Z. and Yuan, C. (2004). "Ant colony optimization for feature selection in face recognition." Biometric Authentication. Springer**:** 221-226.
Yanchao, D., Zhencheng, H., Uchimura, K. and Murayama, N. (2011). "Driver inattention monitoring system for intelligent vehicles: a review." IEEE Transactions on Intelligent Transportation Systems **12**(2): 596-614.
Yang, J., Yu, K., Gong, Y. and Huang, T. (2009). "Linear spatial pyramid matching using sparse coding for image classification." Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2009, IEEE**:** 1794-1801.
Yang, Y. and Pedersen, J. O. (1997). "A comparative study on feature selection in text categorization." Proceedings of International Conference on Machine Learning, 1997, Morgan Kaufmann Publisher, INC.**:** 412-420.
Yu, L. and Liu, H. (2003). "Feature selection for high-dimensional data: A fast correlation-based filter solution." Proceedings of International Conference on Machine Learning, 2003. **20:** 856.
Yu, L. and Liu, H. (2004). "Efficient feature selection via analysis of relevance and redundancy." The Journal of Machine Learning Research **5**: 1205-1224.

Zhang, C., Chen, X. and Chen, W. (2006). "A PCA-based vehicle classification framework." Proceedings of the 22nd International Conference on Data Engineering Workshops, 2006, IEEE**:** 17-17.

Zhang, G. and Wang, Y. (2011). "Hierarchical and discriminative bag of features for face profile and ear based gender classification." International Joint Conference on Biometrics, 2011 1-8.

Zhang, H., Jia, W., He, X. and Wu, Q. (2006). "Learning-based license plate detection using global and local features." Proceedings of the18th International Conference on Pattern Recognition, 2006, IEEE. **2:** 1102-1105.

Zhang, J., Marszalek, M., Lazebnik, S. and Schmid, C. (2007). "Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study." International Journal of Computer Vision **73**(2): 213-238.

Zhang, W., Wu, Q., Wang, G. and You, X. (2012). "Tracking and pairing vehicle headlight in night scenes." IEEE Transactions on Intelligent Transportation Systems **13**(1): 140-153.

Zhang, Z., Cai, Y., Huang, K. and Tan, T. (2007). "Real-time moving object classification with automatic scene division." Proceedings of IEEE International Conference on Image Processing, 2007, IEEE. **5:** V-149-V-152.

Zhang, Z. and Peng, Y. (2013). "Eyeglasses removal from facial image based on MVLR." The Era of Interactive Media. Springer New York**:** 101-109.

Zhang, Z. and Zhang, J. (2006). "A new real-time eye tracking for driver fatigue detection." Proceedings of the 6th International Conference on ITS Telecommunications, 2006, IEEE**:** 8-11.

Zhao, X., Delleandrea, E. and Chen, L. (2009). "A people counting system based on face detection and tracking in a video." Proceedings of the 6th IEEE International Conference on Advanced Video and Signal Based Surveillance, 2009, IEEE**:** 67-72.

Zhou, H., Miller, P. and Zhang, J. (2011). "Age classification using Radon transform and entropy based scaling SVM." Proceedings of the British Machine Vision Conference, 2011, Dundee, UK, BMVA Press. **28:** 1-12.

Zhou, H., Yuan, Y. and Shi, C. (2009). "Object tracking using SIFT features and mean shift." Computer Vision and Image Understanding **113**(3): 345-352.

Zhouhui, L., Godil, A. and Xianfang, S. (2010). "Visual Similarity Based 3D Shape Retrieval Using Bag-of-Features." Shape Modeling International Conference (SMI), 2010**:** 25-36.

Zielke, T., Brauchkmann, M. and von Seelen, W. (1992). "CARTRACK: computer vision-based car following." Applications of Computer Vision, Proceedings, 1992., IEEE Workshop on**:** 156-163.

Appendix Publications

This thesis has led to a number of publications that are listed as following.

Reviewed Journals: (Peng, Xu et al. 2012) (Xu, He et al. 2012)

Peng, Y., Xu, M., Ni, Z., Jin, J. S. and Luo, S. (2012). "Combining front vehicle detection with 3D pose estimation for a better driver assistance." International Journal of Advanced Robotic Systems **9**: 93-108.

Xu, M., He, X., **Peng, Y.**, Jin, J., Luo, S., Chia, L.-T. and Hu, Y. (2012). "Content on demand video adaptation based on MPEG-21 digital item adaptation." EURASIP Journal on Wireless Communications and Networking **2012**(1): 104.

Peng, Y., M. Xu, J. S. Jin, "Vehicle Type Classification with adaptive eigenspace," *IEEE Transaction on Intelligent Transportation System*, 2013(Summitted)

Reviewed Book Chapters: (Cui, Luo et al. 2013) (Peng, Jin et al. 2011) (Peng, Jin et al. 2013) (Peng, Xu et al. 2012) (Xu, He et al. 2011) (Zhang and Peng 2013)

Peng, Y., Jin, J., Luo, S. and Park, M. (2011). "Understanding video sequences through super-resolution." Advances in Multimedia Modeling. Springer Berlin Heidelberg. **6524:** 25-34.

Peng, Y., Jin, J., Luo, S., Xu, M., Au, S., Zhang, Z. and Cui, Y. (2013). "Vehicle type classification using data mining techniques." The Era of Interactive Media. Springer New York**:** 325-335.

Peng, Y., Xu, M., Ni, Z., Jin, J. and Luo, S. (2012). "Accurate pedestrian counting system based on local features." Advances in Multimedia Information Processing – PCM 2012. Springer Berlin Heidelberg. **7674:** 850-860.

Xu, M., He, X., Jin, J., **Peng, Y.**, Xu, C. and Guo, W. (2011). "Using scripts for affective content retrieval." Advances in Multimedia Information Processing - PCM 2010. Springer Berlin Heidelberg. **6298:** 43-51.

Zhang, Z. and **Peng, Y.** (2013). "Eyeglasses removal from facial image based on MVLR." The Era of Interactive Media. Springer New York**:** 101-109.

Cui, Y., Luo, S., Tian, Q., Zhang, S., **Peng, Y.**, Jiang, L. and Jin, J. (2013). "Mutual Information-Based Emotion Recognition." The Era of Interactive Media. Springer New York**:** 471-479.

Reviewed Conference Proceedings: (Park, Jin et al. 2010) (Peng, Jin et al. 2010) (Peng, Luo et al. 2012) (Peng, Jin et al. 2012) (Peng, Jin et al. 2012) (Peng, Xu et al. 2011) (Peng, Park et al. 2010) (Peng, Park et al. 2010) (Cui, Jin et al. 2010)

Peng, Y., Luo, S., Xu, M., Ni, Z., Jin, J. S., Wang, J. and Zhao, G. (2012). "Bag of features using sparse coding for gender classification." Proceedings of the 4th International Conference on Internet Multimedia Computing and Service. Wuhan, China, ACM**:** 80-83.

Peng, Y., Jin, J. S., Luo, S., Xu, M. and Cui, Y. (2012). "3D pose estimation of front vehicle towards a better driver assistance system." Proceedings of IEEE International Conference on Multimedia and Expo Workshops, 2012 522-527.

(Rewarded Best paper in the conference)

Peng, Y., Jin, J. S., Luo, S., Xu, M. and Cui, Y. (2012). "Vehicle type classification using PCA with self-clustering." Proceedings of IEEE International Conference on Multimedia and Expo Workshops, 2012 384-389.

Peng, Y., Xu, M., Jin, J. S., Luo, S. and Zhao, G. (2011). "Cascade-based license plate localization with line segment features and Haar-Like features." Proceedings of the 6th International Conference on Image and Graphics, 2011, IEEE**:** 1023-1028.

Peng, Y., Park, M., Xu, M., Luo, S., Jin, J. S., Cui, Y., Wong, W. F. and Santos, L. D. (2010). "Clustering nuclei using machine learning techniques." Proceedings of IEEE/ICME International Conference on Complex Medical Engineering, 2010, IEEE**:** 52-57.

Peng, Y., Jin, J. S., Luo, S. and Xu, M. (2010). "Learning priors for super-resolution in video sequence." Proceedings of the 2nd International Conference on Internet Multimedia Computing and Service. Harbin, China, ACM**:** 163-166.

Peng, Y., Park, M., Xu, M., Luo, S., Jin, J. S., Cui, Y. and Wong, W. S. F. (2010). "Detection of nuclei clusters from cervical cancer microscopic imagery using C4.5." Proceedings of the 2nd International Conference on Computer Engineering and Technology, 2010 **3:** 593-597.

Cui, Y., Jin, J. S., Park, M., Luo, S., Xu, M., **Peng, Y.**, Wong, W. S. F. and Santos, L. D. (2010). "Computer aided abnormality detection for microscopy images of cervical tissue." Proceedings of IEEE/ICME International Conference on Complex Medical Engineering, 2010 63-68.

Park, M., Jin, J. S., **Peng, Y**., Summons, P., Yu, D., Cui, Y., Luo, S., Wang, F., Santos, L. and Xu, M. (2010). "Automatic cell segmentation in microscopic color images using ellipse fitting and watershed." Proceedings of IEEE/ICME International Conference on Complex Medical Engineering, 2010**:** 69-74.

Appendix High-resolution Figures

(a) Pedestrian counting on PETS2009_1

(b) Pedestrian counting on PETS2009_2

(c) Pedestrian counting on TownCenter

Fig. 6.9 Pedestrian counts by proposed method on three videos.

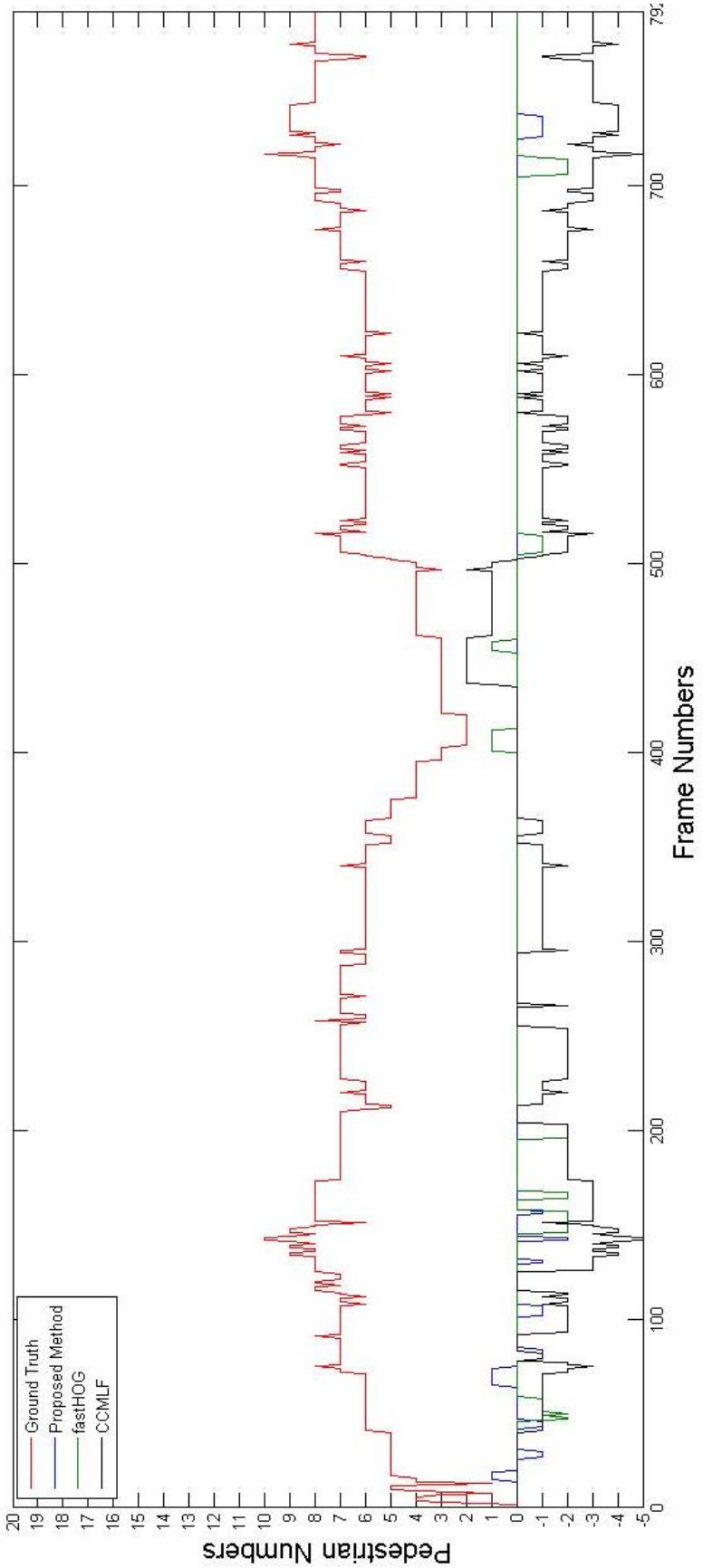


Fig. 6.10 Comparison evaluation of three methods by testing on PETS2009_1.

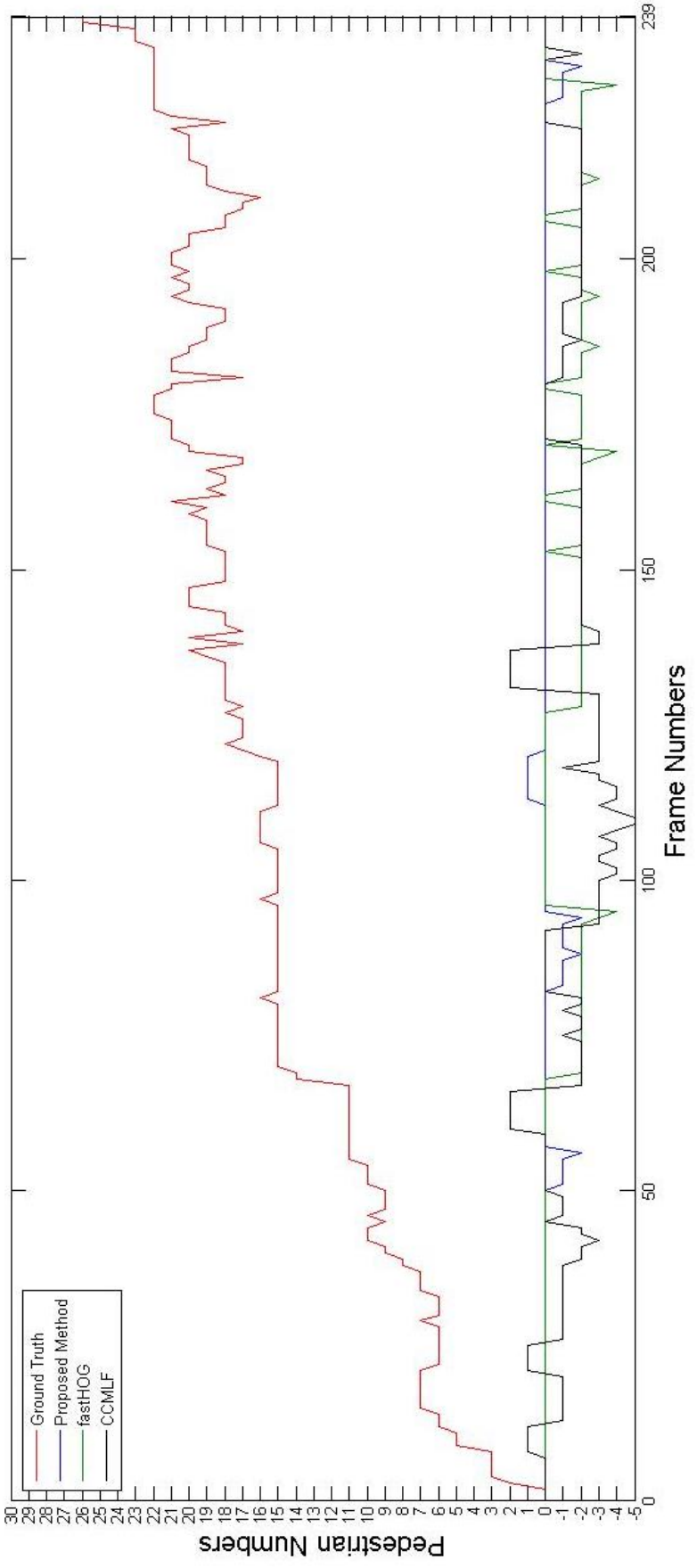


Fig. 6.11 Comparison evaluation of three methods by testing on PETS2009_2.

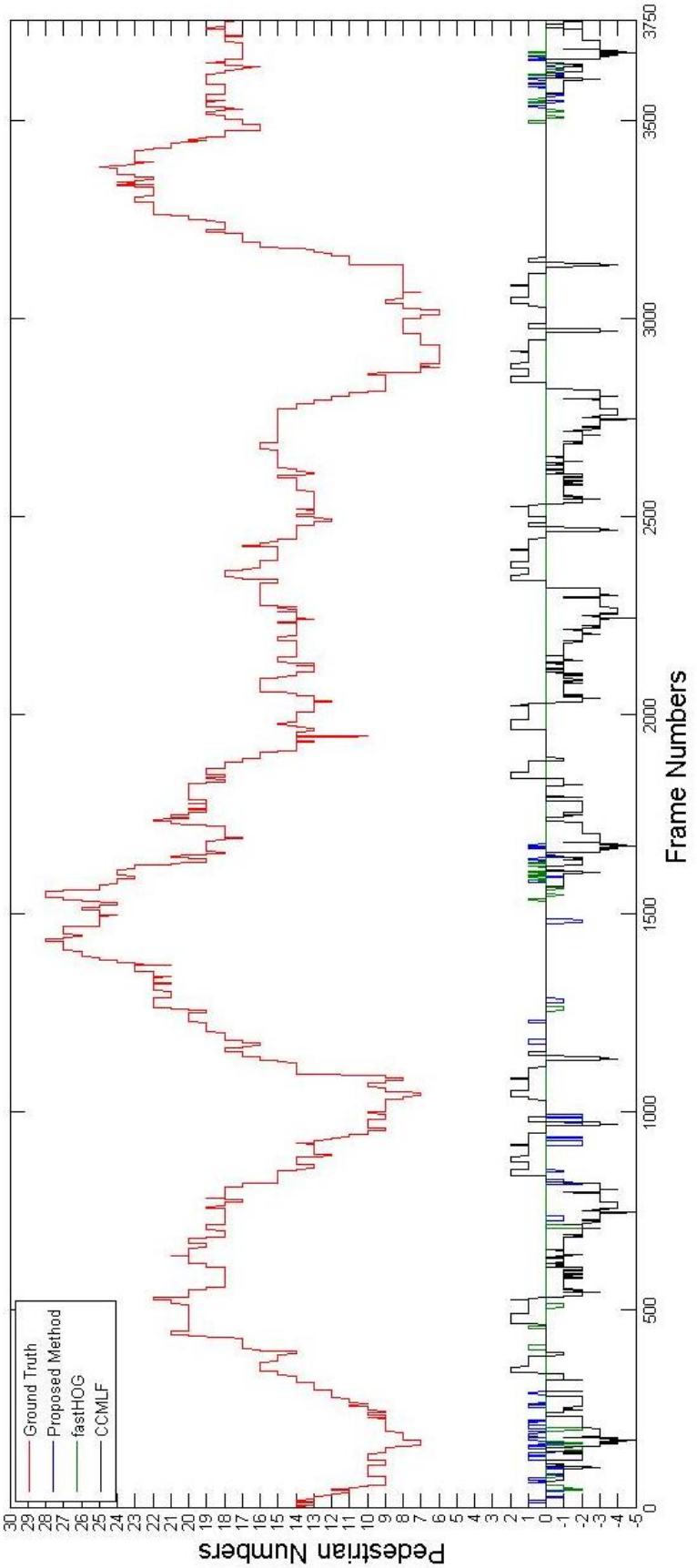


Fig. 6.12 Comparison evaluation of three methods by testing on TownCenter.

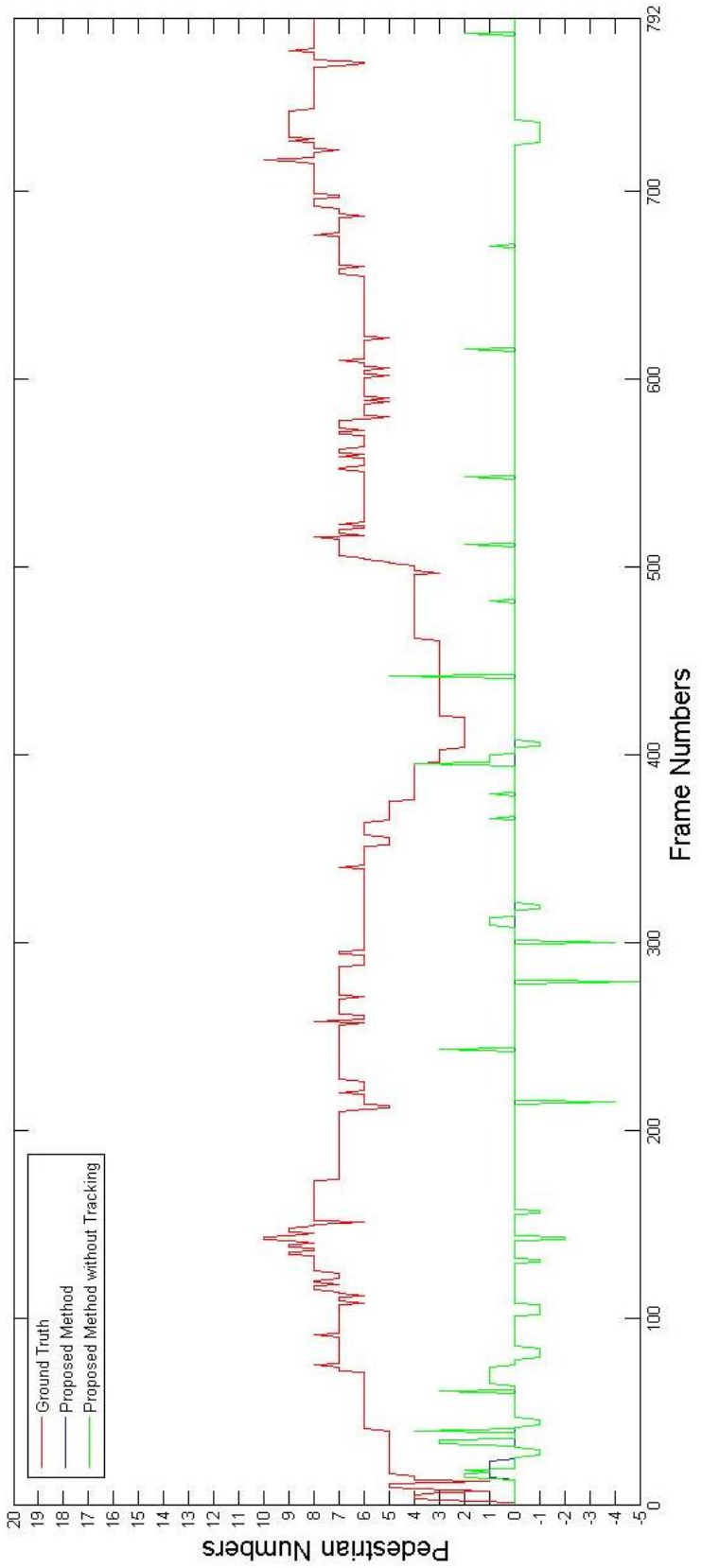


Fig. 6.13 Evaluation of counting improvement by adaptive tracking.