An Efficient Vision-Based Pedestrian Detection and Tracking System for ITS Applications

by

Tianyu Zuo

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.A.Sc. degree in
Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

In this thesis, a novel Pedestrian Protection System (PPS), composed of the Pedestrian Detection System (PDS) and the Pedestrian Tracking System (PTS), was proposed. The PPS is a supplementary application for the Advanced Driver Assistance System, which is used to avoid collisions between vehicles and pedestrians.

The Pedestrian Detection System (PDS) is used to detect pedestrians from near to far ranges with the feature-classifier-based detection method (HOG + SVM). To achieve pedestrian detection from near to far ranges, a novel structure was proposed. The structure of our PDS consists of two cameras (called C_S and C_L separately). The C_S is equipped with a short focal length lens to detect pedestrians in near-to-mid range; and, the C_L is equipped with a long focal length lens to detect pedestrians in mid-to-far range. To accelerate the processing speed of pedestrian detection, the parallel computing capacity of GPU was utilized in the PDS. The synchronization algorithm is also introduced to synchronize the detection results of C_S and C_L. Based on the novel pedestrian detection structure, the detection process can reach a distance which is more than 130 meters away without decreasing detection accuracy. The detection range can be extended more than 100 meters without decreasing the processing speed of pedestrian detection. Afterwards, an algorithm to eliminate duplicate detection results is proposed to improve the detection accuracy.

The Pedestrian Tracking System (PTS) is applied following the Pedestrian Detection System. The PTS is used to track the movement trajectory of pedestrians and to predict the future motion and movement direction. A $C++$ class (called PedestrianTracking class, which is short for PTC) was generated to operate the tracking process for every detected pedestrian. The Kalman filter is the main algorithm inside the PTC. During the operation of PPS, the final detection results of each frame from PDS will be transmitted to the PTS to enable the tracking process. The new detection results will be used to update the existing tracking results in the PTS. Moreover, if there is a newly detected pedestrian, a new process will be generated to track the pedestrian in the PTS. Based on the tracking results in PTS, the movement trajectory of pedestrians can be obtained and their future motion and movement direction can be predicted. Two kinds of alerts are generated based on the predictions: warning alert and dangerous alert. These two alerts represent different situations; and, they will alert drivers to the upcoming situations. Based on the predictions and alerts, the collisions can be prevented effectively. The safety of pedestrians can be guaranteed.

Publications Related to Thesis

Accpeted Conference Papers

- Abdelhamid Mammeri, Tianyu Zuo, Azzedine Boukerche. *Extending the Detection Range of Vision-based Driver Assistance Systems Application to Pedestrian Protection System.* In *Globecom 2014 - Communications Software, Services and Multimedia Symposium.*

Acknowledgements

It would not have been possible for me to write this Master thesis without the help and support from the mentors around me, to some of whom it is possible to give particular gratitude here.

I would like to express my sincere appreciations and thanks to my supervisor Professor Azzedine Boukerche, for his patience, enthusiasm and constant financial support throughout my Master research. It is his critical comments and insightful feedbacks that inspired me to any steps towards the completion of my research. Prof. Azzedine Boukerche have been a tremendous mentor for me for all my master time and will be a bright light to guide me in the future. I would also like to thank him for believing in me and for giving me the opportunity to collaborate with great people and researchers in the PARADISE Research Laboratory and the NSERC DIVA Research centre. It is a great honor of me to be a member in this group and to work in such an excellent environment.

I am grateful to Dr. Abdelhamid Mammeri, for his invaluable advices and inspirations, for all that I learned from him and for the experiences which he has taught to me in research experiments. He has provided assistances in numerous ways to help me find my way to the final point of my Master's thesis. His attitude to research inspired me enormously to complete my master research. Without his guidance and assistance, it would have been longer and more difficult for me to complete my thesis. I should also thank him for encouraging my research and for guiding me to grow as a research scientist. His advices on both research as well as on my career have been priceless.

My sincere thanks also goes to all the colleagues in the Paradise Laboratory and the members in Mobile Vision Group. It is the assistance and help from all of you that make me firmly believe I can complete my thesis and master research. I am grateful to all of you for giving me significant advices and for showing me the confidence to achieve the master research.

A special thanks to my families. Words cannot express how grateful I am to my mother and father for all of the sacrifices that they have made on my behalf. It is my mother's belief and father's inspiration that sustained me to complete my research and to finish my thesis. Without their support and comprehension, I would not have the opportunity to complete the research. I would also like to thank all of my friends. I am grateful to my friends whom have helped me to correct the thesis, whom have helped me to implement the experiments and whom have encouraged me to strive towards my goal.

Table of Contents

List of Tables

List of Figures

Nomenclature

Abbreviations

CPP	Collision Prevention Process
EDR	Elimination of Duplicate Restuls
HOG	Histogram of Oriented Gradients
ITS	Intelligent Transportation Systems
LBP	Local Binary Pattern
MaxDD	Maximum Detection Distance
MinDD	Minimum Detection Distance
PDS	Pedestrian Detection System
PPS	Pedestrian Protection System
PTS	Pedestrian Tracking System
SURF	Speeded Up Robust Features
SVM	Support Vector Machine
TSD	Total Stopping Distance

Mathematical Symbols

θ_h	horizontal field of view		degree
θ_v	vertical field of view		degree
C_L	long focal length camera		
C_s	short focal length camera		
D_b	braking distance	$\frac{V_c^2}{2\mu g}$	m
D_r	minimum required distance to capture ground in the picture		m
D_s	total stopping distance	$D_b + D_t$	m
D_t	thinking distance	$T_p \times V_c$	m

D_w	minimum distance for the system to cover the width of lanes in a picture	m
D_{max}	maximum theoretical detection distance	m
D_{min}	minimum theoretical detection distance	m
H	real height of the entire subjects captured in picture	m
h	real height of pedestrian	m
H_c	height of vehicle	m
H_p	total vertical pixels of the picture	pixel
h_p	height of pedestrians in picture	pixel
T_p	perception time	s
V_c	vehicle velocity	km/h
V_p	the speed of the pedestrian	
W	real width of the entire subjects captured in a picture	m
W_c	width of vehicle	m
W_L	width of lane	m
W_p	total horizontal pixels of the picture	pixel

Chapter 1

Introduction

1.1 Background and Problems

With the increase of vehicles since the end of the last century, accidental collisions between vehicles and pedestrians increased exponentially. With this increasing in the quantity of vehicles, the safety of pedestrians has been severely threatened. Thus, pedestrian safety in modern society is becoming a significant challenge. Researchers and engineers have placed a great amount of concern on pedestrian protection. Since the twenty-first century, the Pedestrian Protection System (PPS) has become an active research area aimed at protecting pedestrians by assisting inattentive drivers. Numerous pedestrian detection methods have been proposed as research on pedestrian protection is becoming deeper, and detection range is addressed in some cases.

To make the PPS suitable for most situations and compatible with most devices, this system has been mainly developed on vision-based techniques. Meanwhile, the whole PPS will be used in the VANET network [11, 12, 13] to cooperate with the vehicle-to-vehicle communication [13, 14] in the future. All these new technologies will be cooperated to serve the Intelligent Transportation System (ITS) [15, 16] in the future. In general, the vision-based pedestrian detection methods can be divided into two main categories: the monocular-camera-based detection method and the stereo-camera-based detection method. The technologies which utilize the techniques of stereo cameras can provide pedestrian detection within the range of 50 meters. On the other hand, the production cost of stereo cameras is a serious disadvantage. In contrast, pedestrian detection based on monocular cameras can ensure the detection accuracy the same with the detection accuracy based on stereo cameras, at a much lower price. However, there is also a shortcoming inherent in the monocular camera detection method. The detection range based on monocular cameras

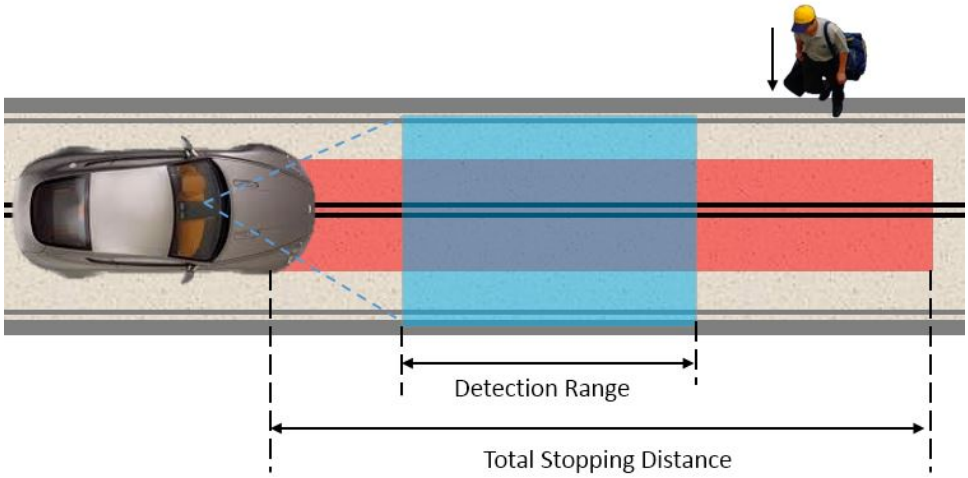


Figure 1.1: An example of pedestrian detection problem.

does not extend as far as the one based on stereo cameras.

Most of the traffic accidents are caused by late braking. Based on statistics, the stopping distance increases exponentially with the increase of velocity. Besides that, the stopping distance in wet conditions is even fifty percent longer than the one in dry conditions. At the same time, the increasing braking distance will result in collisions between vehicles. If the front car brakes too fast, the later car will have too little reaction time to brake himself; thus, the collision can be hardly avoided. Despite the enormous number of research work, most current systems are designed to recognize near-scale pedestrians, like the blue region in Figure 1.1. However, when they are performing pedestrian detection in mid-scale and in far-scale scenarios, these systems cannot perform as well as they do in near-scale scenarios. Moreover, in certain cases, pedestrians cannot be detected. Both the stereo-camera-based detection method and the monocular-camera-based detection methods cannot provide detection too far. It means that the detection range of most PPSs is not large enough to cover the total stopping distance. When a pedestrian is detected, the distance is not large enough to give effective braking. The collision will then happen.

To extend the detection range of the PPS, some researchers utilize the LIDAR detection technique. But the devices are expensive which is a big disadvantage. Some other researchers use smaller sizes of training features with higher image resolution; this can extend the detection range a bit further. However, the detection accuracy decreases greatly and the processing speed of the detection process drops exponentially. Besides, some researchers use the combination of far and near infrared sensors to detect pedestrians. However, there is an area in which pedestrians cannot be accurately detected as the detection range of near infrared sensors and far infrared sensors cannot be connected smoothly.

Moreover, the infrared sensor detection method cannot be used at night. This is because the infrared sensors are easily affected by illumination and environment. Our objective is to extend the detection range, without decreasing the detection speed and detection accuracy, to thus avoid the collisions between vehicles and pedestrians. Meanwhile, the PPS is going to be embedded in the emergency vehicle notification system [17, 18] and the collision avoidance system [19, 20], which are two signification categories of Intelligent Transportation System. To make PPS cooperate with ITS, the protocols [21, 22] used in VANET are undertaking important roles. By the cooperation between PPS and VANETs [23, 24], collision caused by late braking between vehicles can be prevented as well.

1.2 Goals

According to the problems described in previous section, we are determining to develop a method to extend the detection range without yielding detection accuracy and processing speed. Therefore, we proposed a novel architecture to perform pedestrian detection which will be followed by the pedestrian tracking system.

1.3 Contribution

As the development of the PPS is restricted by the detection range, our goal is to extend the total detection range of PPS and to make it suitable to most situations. In this thesis, we describe a novel pedestrian detection structure which utilizes two identical monocular cameras, each of which is equipped with a zoom lens. With the help of this novel structure, the PPS can detect pedestrians nearby, as well as those in mid and far ranges. Each of the cameras detects separated ranges. One camera detects the near-to-mid region located in front of vehicles. The other one detects the mid-to-far region. The system then combines the results and transmits the final results to a pedestrian tracking process. Meanwhile the detection range of each camera can be changed according to the driving speed, to give the highest protection level to pedestrians.

Meanwhile, we combined Speeded Up Robust Features (SURF) into our system, to test the duplicate detection results elimination. When the novel PPS is in operation, it is possible that one or more pedestrians has been detected by both cameras. The duplication elimination procedure is required to eliminate repeated detection results then. During the experiment, the SURF-based elimination method can exhibit good elimination precision; but, the robustness of the processing speed is a problem. We want to increase the robustness

of the processing speed of the elimination process, as well as to increase the elimination precision. So we proposed a new duplication elimination method. In the experiment, the testing results indicate that our own elimination method can guarantee the robustness of the processing speed and that they can provide better elimination precision.

After the PPS finishes its detection process, a tracking process is required to perform pedestrian tracking and to predict future motion and movement direction. By introducing the tracking process after the detection process, two helpful functions can be added to the PPS. One of them is the detection results refinement. By using the tracking process, the detection results are transmitted to the tracking system and the position of pedestrians can be refined by the tracking process with its tracking ability. The other helpful function is the prediction of future motion and movement direction. Based on the innate prediction ability of the tracking process, the motion and behavior of pedestrians in the next frame can be predicted. If the tracking results and the prediction results are obtained, the PPS can give judgements to each detected pedestrian. The PPS can determine whether the detected pedestrians are in warning situations or in danger. If the judgement from PPS is positive, the PPS will send alerts to drivers in advance to tell them there is a warning or dangerous situation in front of the vehicles.

Based on the three main contributions, we have proposed the novel PPS to achieve our motivation.

1.4 System Overview

The PPS is composed of two main subsystems: the Pedestrian Detection System (PDS) and the Pedestrian Tracking System (PTS). The PDS is used on pedestrian detection and duplicate results elimination. After the repeated results elimination, the final detection results are transmitted into the PTS. In the PTS, each detection result is followed by a tracking process. The tracking process will update its tracking results with the new detection results, or it will be updated by the former prediction results within several frames. Based on the tracking results, the PPS will judge the situation in front of the vehicle and decide whether to give a warning or danger alerts to the driver. After the PTS, the refined tracking results are transmitted to the in-car monitor to show the PPS detection results to drivers. The whole system will iterate the whole process infinitely during the operation to protect drivers and pedestrians.

1.5 Thesis Outline

The whole thesis consists of five chapters. In the first chapter, we provide some descriptions on the backgrounds of pedestrian detection and present the problem which we are mainly concerned about. The motivation and contribution are then discussed to show what was undertaken and what we did to tackle it. In Chapter 2, the background on pedestrian detection and pedestrian tracking are specifically described. The related works for our research are summarized and referenced. What the predecessors have done can be clearly summarized from this chapter.

The Chapter 3 and Chapter 4 fully describe our novel PPS. Chapter 3 talks about the algorithms and structure and Chapter 4 demonstrates the implementation environment and the experiment results analysis. In Chapter 3, the PDS and the PTS are separately introduced in Section 3.1 and Section 3.3. Section 3.1 mainly describes the algorithms and structure of our detection system. Meanwhile, Section 3.2 gives detailed introductions to the detection results synchronization the duplicate detection results elimination. Both the SURF-based elimination method and our own elimination method are explained. The PTS are introduced in Section 3.3, in which the tracking procedures and algorithms we utilized are discussed. Likewise, the judgement of alerting situation is also introduced in Section 3.3. In Chapter 4, the implementation environment, the implementation results and results analysis are exhibited. Chapter 4 explains the calculation procedure of the the detection range and gives a total computing example of the detection range calculation. Meanwhile, the specifications of the implementation environment and the coordination of devices are exhibited in detail. The selections of image resolutions and HOG descriptor sizes are talked about as well. The tracking samples and results are exhibited at the end of the Chapter 4. The content in the last chapter (Chapter 5) includes the conclusion about our PPS, followed by a future work that we plan to work on .

Chapter 2

Related Work

Pedestrian detection has been a popular research field over the recent years due to its wide utilization in areas such as visual surveillance, intelligent transportation, virtual reality and intelligent vehicle systems. It is one of the most challenging tasks in the research of computer vision due to many different types of noise, created by the appearance of pedestrians and the changes of postures and illumination, in addition to being brought about by complex backgrounds and perspectives. Occlusion is, moreover, a difficult problem.

According to the devices which are used in the detection process, the pedestrian detection approaches can be approximately divided into two aspects: the vision-based pedestrian detection approach and the radar-based pedestrian detection approach (shown in Table 2.1). In this thesis, we are mainly concerned with the vision-based pedestrian detection methods which have been widely used in intelligent vehicle systems. In the following, the vision-based detection achievements over recent years are discussed.

2.1 Pedestrian Detection

Many research works related to pedestrian detection have been proposed [1, 51, 52, 53, 54, 55, 56, 57, 39]. Typically, vision-based pedestrian detection can be achieved by the static detection approach or the dynamic detection approach. The definition of the detection approach depends on whether the motion information of pedestrians is utilized in the system. The former approach is widely used in vehicular systems and the latter approach is mostly used in systems with fixed cameras, such as surveillance systems. Besides the pedestrian detection approaches, the pedestrian dataset is another crucial component in the PDSs.

Table 2.1: Comparison of detection sensors.

Sensor	Functions	Advantages	Disadvantages	Related References
Optical Camera	Pedestrian feature extraction.	— Can obtain the specific information of detection reuslts: silhouette, texture, color, etc. — Larger detection range and wider field of view.	— Need to manage various features. — Affected by weather, illumination, etc. — Inaccurate distance information of detection results.	2013 [25, 26], 2012 [27, 28, 29, 30], 2007 – 2011 [31, 32, 33].
Infrared Camera	Pedestrian contour extraction.	— Easily obtain the contours of pedestrians. — Suitable to work at night.	— Easily affected by different heat source. — Inaccurate distance information of detection results.	2010 – 2012 [34, 35], 2004 – 2008 [36, 37, 38, 39].
Millimeter Radar	Distance measurement, moving pedestrians detection.	— Outstanding abilities of distance measurement, speed measurement and self-speed feedback. — Not affected by color, illumination, posture, etc.	— Cannot obtain the texture information. — Not suitable to classification. — Highly false-positive rate due to various kinds of noise.	2012 – 2013 [40, 41, 42, 43], 2005 – 2010[44, 45].
LIDAR	Distance measurement, environment modeling.	— Outstanding abilities of distance measurement, speed measurement and self-speed feedback. — Not affected by color, illumination, posture, etc.	— Cannot obtain the texture information. — Not suitable to classification. — Hard to popularize.	2009 – 2010[46, 47], 2011 [48, 49, 50].

2.1.1 Open Source Pedestrian Databases

One of the essential parts of this vision-based PDS is the dataset. In Table 2.2, we listed some of the widely used datasets for PDSs. In general, the MIT dataset and INRIA dataset are mostly used and tested by researchers. The Caltech pedestrian dataset is currently the largest dataset, which consists of both the positive and negative training sets and the testing sets. The CVC pedestrian dataset is composed of several subsets. Some of these subsets are composed of virtual pedestrians and the others consist of pedestrians in the real-world. In addition, the CAVIAR dataset includes many video clips, as well as the frames in each video.

Table 2.2: Pedestrian detection datasets.

Dataset	Dataset Specifications	Brief Comments
MIT CBCL [58]	— Training set: 924 positive. — No negative images or testing images.	— Only 64 × 128 PPM format images. — The pose is limited to frontal and rear views.
Daimler Pedestrian Detection Benchmark Dataset [53]	— Training set: 15,560 positive + 6744 negative. — Testing set: 21,790 images with 56,492 pedestrian labels.	— Captured from 27m video. — Monocolor.
INRIA Person Dataset [6]	— Training set: 2416 positive + 1218 negative.	— Only upright persons. — Color image.
CVC Pedestrian Dataset [59]	— Real-world dataset: 01, 02 and 2005. — Virtual dataset: 03, 04 and 06.	— Virtual and real-world pedestrian datasets.
CAVIAR [60]	— 79 video clips with corresponding images.	— Including many postures and activities.
Caltech Pedestrian Dataset [61]	— Training set: 67000 positive + 61000 negative. — Testing set:65000 positive + 56000 negative.	— Largest dataset: 6GB training sets and 5GB testing sets.

2.1.2 Static Detection Approaches

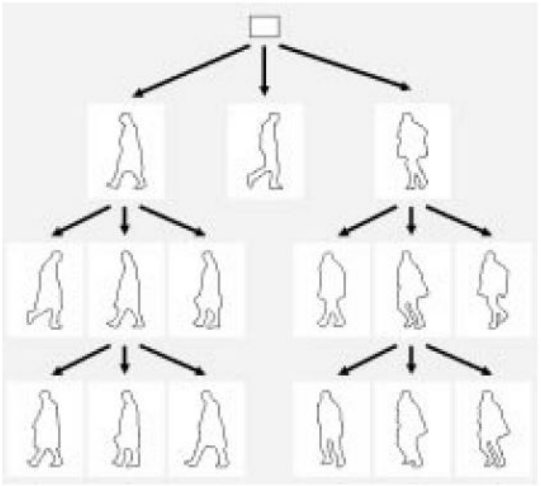
In the static detection approach of pedestrian detection, there is a model-based detection method and a feature-classifier-based detection method; they correspond to the generative model and the discriminative model in pattern classification respectively.

Figure 2.1: A sample of model-based pedestrian detection [2].

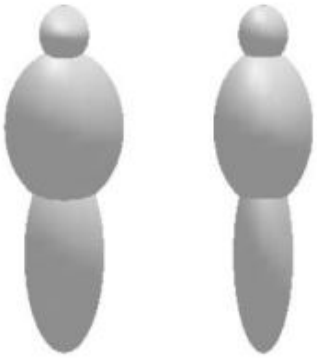
2.1.2.1 Model-Based Detection

The model-based pedestrian detection is operated based on a model dataset which consists of vast pedestrian models. When the model-based detection method is in operation, the detection system needs to search the matched region in the captured frames based on the models in the model dataset. A threshold is set before the comparison. If the comparison results between the model and the region in the captured frame are larger than the threshold, this region will be understood as being a pedestrian. An example in the model dataset is shown in Figure 2.1.

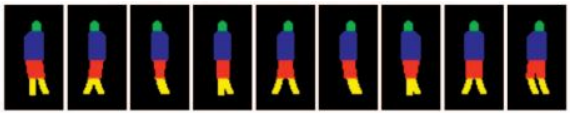
Over recent years, many types of models have been proposed by researchers. One of the widely used pedestrian detection models is the shape model. In 1999, Gavrila, D.M. et al. [2] proposed a real-time pedestrian detection algorithm by using the "Chamfer" system to apply a coarse-to-fine model matching detection method. Afterwards, Gavrila, D.M. et al. developed the pedestrian detection method a further step [62]. To perform accurate pedestrian detection, a large amount of exemplars are required to represent the various postures of pedestrians. However, the large number of models greatly increase the matching time. Gavrila, D.M. et al. proposed a hierarchical exemplar tree of pedestrian shapes to further implement the matching process. The hierarchical exemplar tree has decreased the computation of model matching greatly and has increased the processing speed significantly. Tao Zhao et al. described a pedestrian detection method which is used in crowded environments [63, 3]. In [63], they used the 3D model-based detection approach to detect pedestrians with shape models. After that, they proposed a new model-based approach, that made use of multiple partially occluded human features to form hypotheses in a Bayesian framework, to detect humans [3]. Rittscher, J. et al. [64] focused on the integration of feature grouping and on the model based segmentation into one consistent framework. Guang Chen et al. [65] proposed a conservative model adaptation method to



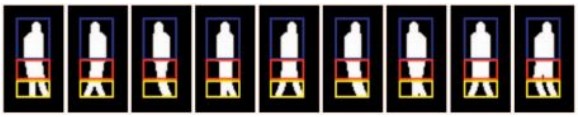
(a) Hierarchical tree of exemplar



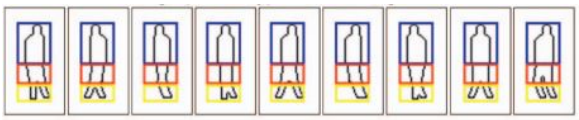
(b) 3D model example



(c) Parts of partial global shape model



(d) Silhouettes of partial global shape model



(e) Contour of partial shape model

Figure 2.2: Examples of models of model-based pedestrian detection [2, 3, 4].

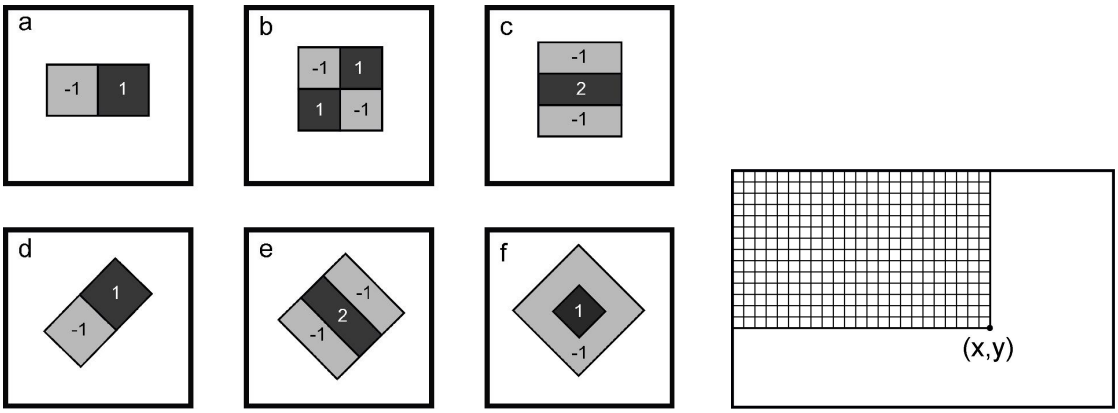
Figure 2.3: A flow chart of feature-classifier-based pedestrian detection.

determine the selection of the adaption rate to increase detection performance. Gavrila, D.M. [66] presented a new pedestrian detection approach, in which a novel template tree was used to perform the hierarchical, exemplar-based shape matching process. Zhe Lin et al. proposed a partial hierarchical template matching methodology [4]. Their detection method relies on the idea of matching a partial-template tree to images hierarchically. Based on this approach, the pedestrian detection and pedestrian segmentation can be processed simultaneously. Ying Wu et al. [67] proposed a detection approach based on a two-layer statistical field model. They used the two-layer statistical field model to increase the robustness of shape-based pedestrian representation. Some model examples are shown in Figure 2.2. Weina Ge et al. [68] proposed a pedestrian detection method by using a multi-camera system. This system is a model-based pedestrian detection system as well.

The model-based pedestrian detection method is one of the real-time pedestrian detection approaches. However, there are still disadvantages to the model-based pedestrian detection method as a great amount of work is required to build the model dataset. Meanwhile, the different postures and heights of pedestrian results in a large model dataset size, which gives rise to complex computation. In addition, the model-based pedestrian detection method is easily affected by shape extraction. If the contour is not clearly extracted, the detection accuracy will drop greatly.

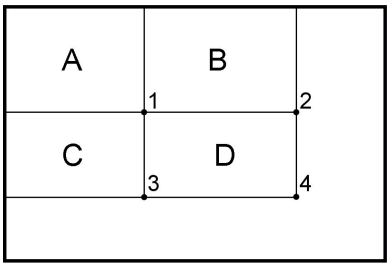
2.1.2.2 Feature-Classifier-Based Detection

In the feature-classifier-based detection methods, a pedestrian is defined as an entirety. The detection process draws a rectangle around the detected pedestrian to show the position of this pedestrian. For each given frame in a video, the detection process divides the frame into several regions. This dividing process is determined by the size of the trained features. After this division, the detection process needs to extract the useful features from each region, such as HOG features, haar-like features, LBP features, etc. Classifiers are utilized to proceed with the classification process and this is done by using these extracted features. The flow chart of the feature-classifier-based pedestrian detection process is exhibited in Figure 2.3. By using the feature-classifier-based detection methods, the detection process can be easily processed. Meanwhile, the features which can effectively



(a) Haar features

(b) Original image



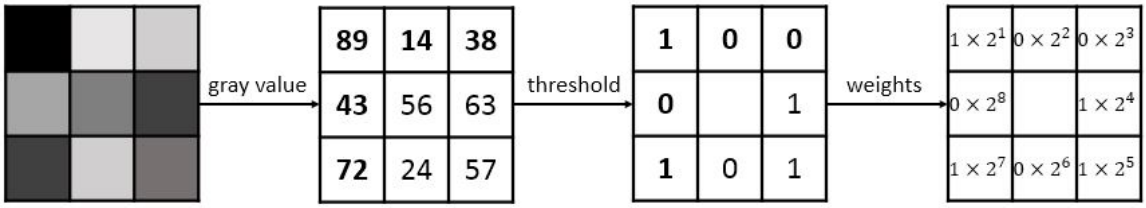
(c) Integral image

Figure 2.4: Examples of the Haar features and the integral image [5].

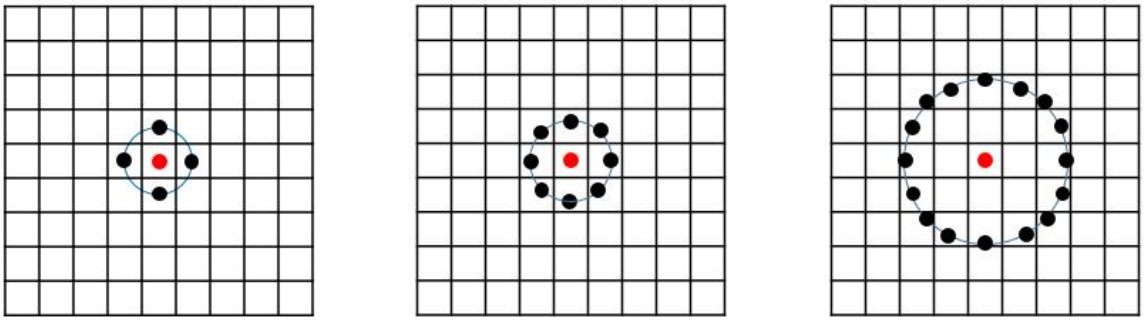
describe pedestrians need to be selected carefully; this is because their various appearances and different standing or walking postures are hard to describe.

The features commonly used to describe pedestrians over the recent years are the following features: HOG features, Haar-like features, LBP features, edgelet features and shapelet features.

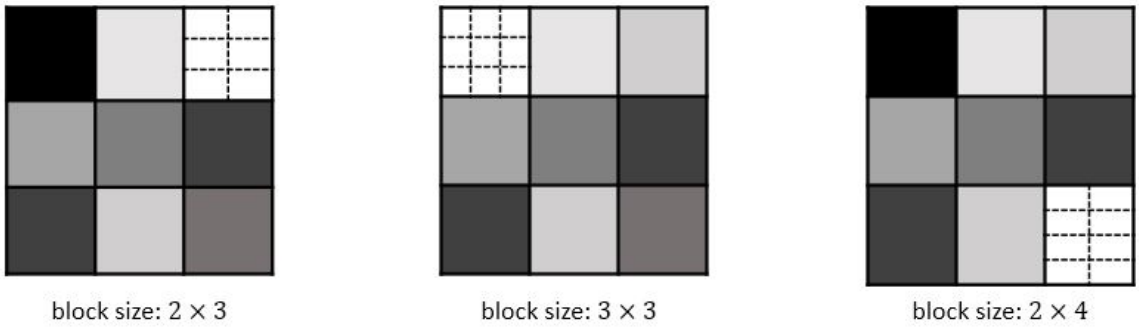
The Haar features were first proposed by Papageogiou et al. [69]. In the beginning, the Haar features were only able to be calculated by two methods: through a set of horizontal and vertical rectangles or through a square set of four rectangles. After this, more computation methods have been proposed for the calculation of the Haar-like features, as shown in Figure 2.4a. The computation of the Haar-like features utilizes the integral image to accelerate computing speed. The value of each region in the integral image is accumulated from the original image. Each point in Figure 2.4c represents the accumulation result of position $(0, 0)$ to position (x, y) in the original image (Figure 2.4b). The particular attributes of objects in each image can be accurately described by the Haar-like features. Inspired by [69], Viola, P. et al. introduced the Haar-like features in [5]; and, they implemented it on the face detection process. Furthermore, Viola, P. et al. used the Haar-like features on pedestrian detection in [70]. Since then, the Haar-like features have



(a) Basic LBP features [71]



(b) Generic LBP features [72]



(c) MB-LBP features [73]

Figure 2.5: Different examples of LBP features.

been utilized in many PDSs, as shown in Table 2.4.

When the Local Binary Pattern (LBP) was first introduced by Ojala et al. in [71], it was mainly used on texture classification. An example of the basic LBP features is shown in Figure 2.5a. Thereafter, LBP was utilized in the field of pedestrian detection. The computation of LBP follows an ordered set of grid pixels. The value of each pixel around the center pixel is compared with the value of the center pixel; this is done in order to get the 1 or 0 result. The final value of this grid is computed based on the compared results. Through these means, the computation of the LBP features is easily implemented and is computationally efficient. The processing speed of LBP-based pedestrian detection is increased as well. Timo Ojala et al. extended the basic LBP operator into a more generic

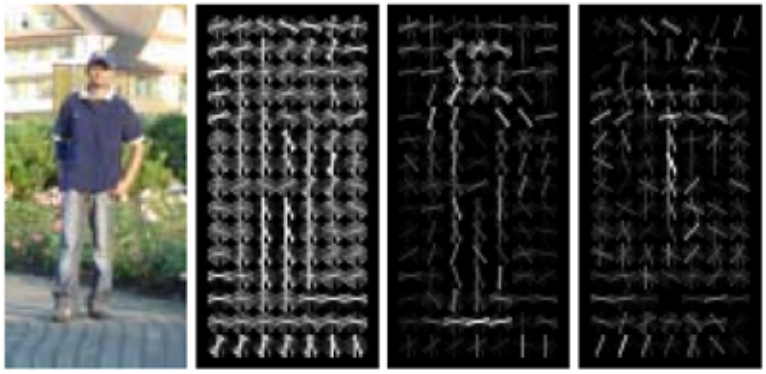


Figure 2.6: An example of HOG features [6].

version in [72, 74]. Unlike the basic LBP which is only computed by a 3×3 pixel block, the computation of the generic LBP features utilizes different block sizes to capture dominant features in different scales. In addition, the shape of the generic LBP has been changed into a circle. This new shape allows researchers to choose an arbitrary number of neighborhoods in desired distances. The example of the generic LBP is shown in Figure 2.5b. Inspired by the integral image of Haar features, the multi-scale Block LBP (MB-LBP) (shown in Figure 2.5c) was proposed by Zhang et al. [73]; the function of the MB-LBP is to detect differently sized the objects in the image. Some LBP-based pedestrian detection methods are exhibited in Table 2.4.

HOG features were first proposed by Dalal, N. et al. [6], as the example shown in Figure 2.6. One of the advantages of HOG features is that they can be used all day long because they can describe objects without being affected by changes in illumination. Another advantage of the HOG features is the good tolerance of different pedestrian postures, such as the posture of standing, walking, extending arms outwards or bicycling. Moreover, the different appearances of pedestrians can also be ignored through the use of HOG features during the detection process. Attracted by the advantages of HOG features, many researchers have proposed their pedestrian detection approaches by using HOG features, some of which are shown in Table 2.4.

The contour of pedestrians is one of the most obvious and easily represented features. Bo Wu et al. proposed an edgelet-based pedestrian detection approach in [7]. Unlike the definition of pedestrians in [62], in which pedestrians are represented by an entire shape model, the authors utilized small lines and curves to describe the edge of pedestrians, as shown in Figure 2.7. Each edgelet is one small line or one tiny curve. The contour of a pedestrian is described by the combination of these small lines and tiny curves. By using this edgelet features, the repeated computation of similar shape models can be avoided. The edgelet-based pedestrian detection method does not need to transform the original

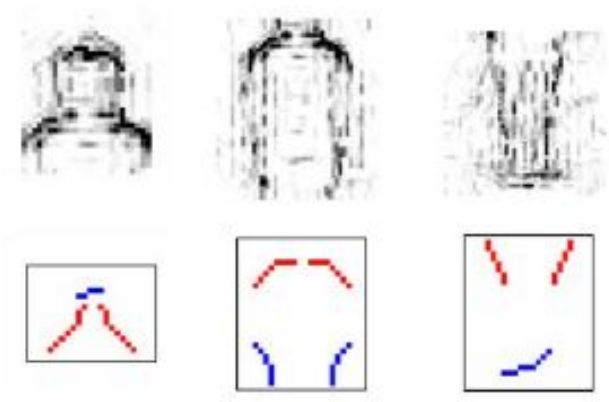


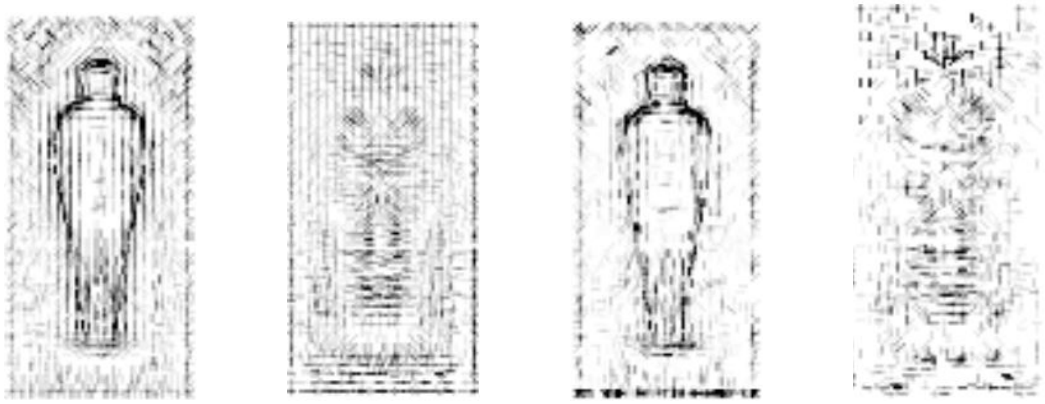
Figure 2.7: Head-shoulder, torso and leg comparison between shapelet features and edgelet features – the upper row represents the shapelet features and the lower row represents the edgelet features [7, 8].

image into a binary image. It uses the Sobel filter to compute the contours of pedestrians in the image. By doing this, the influence of illumination can be decreased. Meanwhile, the orientation information of edgelets can be used to reject the false similar edgelets in the image.

The shapelet features were proposed by Sabzmeydani, P. and Mori, G. in 2007 [8]. In their method, the pedestrian description features were extracted by machine learning algorithms. The Adaboost was used twice to train the final features. For the first step, the low-level gradient features are selected by the Adaboost to generate the mid-level shapelet features. Secondly, the Adaboost is used again to train mid-level shapelet features into becoming the final shapelet features. An example of the training of shapelet features is shown in Figure 2.8. By using the shapelet features, the detection accuracy of the shapelet-feature-based detection approach becomes higher than the edgelet-feature-based detection approach. The comparison was shown in Figure 2.7. Moreover, the shapelet features from the selection of machine learning algorithm are more precise than the edgelet features from artificial selection.

Besides the pedestrian description features described above, other features such as the Shape Context [75, 76], the Scale Invariant Feature Transform (SIFT) [77, 78] and the Speeded Up Robust Features (SURF) [79] can also provide effective pedestrian detection.

Other than the monocular-camera-based pedestrian detection approaches, stereo-camera-based pedestrian detection methods are also widely used. Suzuki, K. et al. [80] proposed a detection strategy by utilizing a probabilistic inference engine. The probabilistic inference engine can detect the context of an individual pedestrian and the context of a group of



(a) (b) (c) (d)

Figure 2.8: Samples of shapelet features – (a) and (c) are the pedestrian class features; (b) and (d) are the non-pedestrian class features; (a) and (b) are the illustration of low-level features selected and weighted in shapelet features across the detection window; (c) and (d) are the illustration of low-level features belonging to only the shapelet features inside the final classier [8].

pedestrians. Kawabe, M. et al. [81] proposed a pedestrian detection method by segmenting foreground regions to detect pedestrians with the help of a stereo camera. The pedestrian behaviors are then analysed by the detection system to arrange for an incoming emergency response in crowded scenarios. Al-Mutib, K. et al. [82] proposed their stereo-camera-based pedestrian detection and particle filter based tracking strategy. Krotosky, S.J. et al. [38] presented an analysis of color-, infrared- and multimodal-stereo approaches to pedestrian detection. Four cameras were used, of which two are infrared cameras and the other two are color cameras. They coordinated these four cameras to proceed the pedestrian detection. The stereo features can be obtained as well by the use of these four cameras. The experimental results demonstrated that the detection performance is good when color and infrared cameras are both used to extract features.

2.1.2.3 Part-Based Pedestrian Detection

Part-based pedestrian detection is commonly used in crowded situations for surveillance. In crowded scenes, pedestrians are always occluded due to so many pedestrians and obstacles. The pedestrian detection approaches which are based on the features of entire pedestrian bodies cannot provide accurate pedestrian detection in these occluded scenarios. To achieve pedestrian detection in occluded scenarios, researchers have proposed the detection methods that use features of partial bodies in [83, 84, 85, 8, 7]. In this process

of part-based pedestrian detection, pedestrians are segmented according to the number of parts used in the detection algorithms; an example of this is shown in Figure 2.7. The features of each part of this approach are trained separately. Each part of the pedestrian is detected by each specific partial features. The detection results for distinct partial features are merged together to provide the final position of detected pedestrians. This merging approach is processed based on geometric constraints.

Leibe et al. [84] proposed an implicit shape model to describe pedestrians. They utilized both the local and the global cues via probabilistic top-down segmentation. In this method, Hough transform was used to obtain the possible position of detected pedestrians; this was followed by the use of the Chamfer matching method to restrict the contour of pedestrian bodies. Finally, the MDL [86] was used to confirm the final position of the detected pedestrian. Gall et al. [87] proposed a detection method via the generalized Hough transform method. In their proposed approach, a class-specific Hough forest was used to obtain the possible location of the object centroid by mapping image patch appearances to the probabilistic vote. Maji, S. et al. [88] proposed a discriminative Hough transform based object detection method; in this method, a max-margin framework was used to train the weights to optimize the classification performance. Usually, pedestrians are divided into four parts during classification. Some segmentation processes classify the set of parts as head, legs, left and right arms [89]; and some other segmentation processes classify the set in terms of full body, head-shoulder, torso and legs [7]. The approaches described in [90] and [91] perform pedestrian detection by using more than 4 parts of the pedestrian segments. The pedestrian detection method based on different viewpoints is also discussed in [92]. The features used to describe entire pedestrian bodies are tested in the part-based pedestrian detection as well [93, 94]. The part-based pedestrian detection approaches can perform accurate pedestrian detection in crowded scenes. However, the segmented features of pedestrians can only be effectively extracted within high resolution frames. This results in a decrease in the processing speed.

2.1.3 Dynamic Detection Approaches

Dynamic pedestrian detection is commonly used in the pedestrian surveillance application, with a fixed camera system. The fixed camera does not have any intense movements. If a camera can be turned, the background of the images should be first processed. In general, the dynamic detection approach is used to detect moving objects. If the pedestrian is standing in one position without moving, the system can hardly detect this pedestrian. The dynamic detection approach is achieved by utilizing the motion features in videos. Based on the motion features, the detection system can extract objects from captured

frames and can continuously track the detected objects. Three types of moving pedestrian detection methods are commonly used. They are the frame difference method, the optical flow method and the background subtraction method. The correlated references are listed in Table 2.3.

Table 2.3: Dynamic detection approaches.

Methods	Advantages	Disadvantages	Related References
Frame Difference	— Lower computing complexity. — High processing speed.	— Easily affected by the background luminance. — Sensitive to object moving speed.	[95][96] [97][98]
Optical Flow	— No need to know the informaiton of background in advance. — Motion features are included in optical flow features.	— High computing complexity. — Low processing speed.	[99][100] [101][102]
Background Subtraction	— Have ability to perform self-adaptive updates. — Can be used with other features.	— Sensitive to changes in illumination. — Required background modeling.	[103][31] [104][105]

The method which utilizes the frame difference features [106, 107] is widely used in moving object detection. In this method, a difference image is generated. This difference image is calculated by subtraction between two adjacent frames [106]. Based on the difference image, the detection process regards a region as relatively static if the subtraction result of this region is very small. By contrast, if the subtraction result of a region is large, the detection process regards this region as a part of the dynamic object. After the regions of the dynamic object are acquired, the position of the moving object can be confirmed. However, the size of moving objects and the differences of background luminance have an effect on the detection results. Specifically, the size of the moving object can affect the size of the detected dynamic region. If the moving object is very large, a scenario in which a static region is surrounded by dynamic regions may occur. The background luminance has an effect on the correlating features. Thus, the detection approach that uses the frame difference is required to be adjusted during the operation.

Optical flow [108, 109, 110] is described as the presentation of the brightness of each pixel of the moving object. The motion features and the shape features of the moving

object can be extracted easily, from the features of the optical flow. The optical flow detection method can be processed without knowing the information of the background. It can be quickly implemented after system set-up. However, because of the detection noise, the object shadow and the occlusion, the detection accuracy of optical flow methods is not robust. Moreover, the computation of the optical flow is complex and time-consuming. Real-time detection is then difficult to achieve without hardware acceleration.

The background subtraction detection method compares the grey value or the histogram information of the captured frame with the grey value or the histogram information of the original background model. Because the background model varies according to the changes in illumination, the background model should then have the ability to perform a self-adaptive update. Over recent years, two methods have been discovered to process the self-adaptive update. One of the methods is based on the Gaussian background model, which uses the Gaussian distribution to describe the changes of background. Christof Ridder et al. [111] proposed a Kalman-filter-based method to enable the system to continuously operate with the changes in illumination. However, one Kalman filter is not enough to process the complexity of the background. Stauffer, Chris et al. [112] used multiple Kalman filters to increase the robustness. P. Kaewtrakulpong et al. [113] proposed an improved adaptive background mixture model to achieve the moving object detection. Afterwards, the moving object detection method introduced by D.R. Magee [114] made use of the Kalman filter on foreground objects. This method is proven to provide better moving object detection results. The other self-adaptive update method is first proposed by Long, W. et al. [115], which is based on the estimation of pixel value. This method utilizes an estimated pixel value according to a sequence of frames. They believe that if the grey value of the pixels in the video sequence does not change, this grey value of the pixels is then the background grey value. However, this estimation can cause false detection results if the object is moving at a much lower moving speed. To solve this problem, Gutchess et al. [116] proposed an improved method which utilizes the optical flow in the detection process. D. Gutchess et al. [117] proposed the Time Median Background Initializing method to improve the robustness of the detection results.

2.1.4 Hybrid Detection Approaches

The combination of different features is introduced to tackle the complex features of pedestrians. Dalal et al. [118] introduced the Histogram of Optical Flow (HOF) to compute the HOG features of the optical flow. Walk et al. [119] used the combination of HOG, HOF and color self-similarity to process pedestrian detection. Wang et al. [120] proposed a novel type of pedestrian description features, which is composed of HOG features and LBP

features. Chengbin Zeng et al. [121] proposed a rapid head-shoulder detection method for pedestrian counting, by utilizing the combined features of HOG features and LBP features. Viola et al. [122] proposed a new robust pedestrian detection method by utilizing both motion features and appearance features, which are the Haar-like features and the frame difference. Jones, M.J. et al. [123] improved [122] by extracting the motion information within mutliple frames. Yao et al. [124] extracted the covariance features of the probabilistic foreground image to improve the detection performance. Schwartz et al. [125] proposed a pedestrian detection method which uses the edge, the color and the texture information to represent pedestrians.

2.1.5 Features Classification

The most used feature training algorithms are presented in Table 2.4. In the classification

Table 2.4: Classifier training algorithms.

Algorithm	Features	Advantages	Disadvantages	Related References
SVM	Linear and non-linear SVMs.	— Processing classification in high-dimensional space. — Can avoid over-learning.	— Time-costing on classification for large-scale model datasets. — Can only be used to binary classification problems.	HOG[126][10][127] Haar[33][128] LBP[28][129]
Adaboost	Cascade weak classifiers.	— High processing speed. — Can classify multiple pedestrian features.	— Can be affected by data noise.	HOG[130][131] Haar[132][30] LBP[129][133]
Neural Networks	Multi-layers and feed-forward.	— Suitable for non-linear problems. — Have self-learning ability.	— Can cause over-fitting or under-fitting problems. — Easily affected by layers and number of neurons.	[51][134] [135][136]

process, the Support Vector Machine (SVM) and the Adaptive Boosting (Adaboost) are the two most used training algorithms in the research of pedestrian detection. Besides SVM and Adaboost, the neural network (NN) algorithm is also used to implement the classification process.

The SVM algorithm was first introduced by Vapnik et al. in 1995 [137] to analyze data and to classify patterns. Papageorgiou, C. et al. [138] combined the SVM with the Haar-like features to implement the pedestrian detection process in static images.

Afterwards, the SVM algorithm was widely used in the research of pedestrian detection. The difference between various kinds of SVMs is the kernel technique. The different types of SVMs can be classified into linear SVMs and non-linear SVMs. The linear SVMs are mostly used to train the description features of pedestrians to obtain the final classification features (related works are shown in Table 2.4). The non-linear SVMs can increase the classification accuracy slightly. However, the non-linear SVMs decrease the computation speed due to the increase in computation complexity [139, 140, 89, 141, 142, 143]. Before the classification, the extracted pedestrian features are transformed into vectors. The main task of SVM is to map the vectors of both positive features and negative features to a coordinate with higher dimensions. In the coordinate, the classification is processed by maximizing the boundary between the vectors of positive features and the vectors of negative features. Currently, some non-commercial open source SVM softwares have been published on their websites, such as the SVMlight [144], libHIK SVM [145, 146], SVMtool [147], libSVM [148] and SVMlin [149].

The Adaboost learning algorithm is achieved by assembling weak classifiers to a strong classifier under the framework of ensemble learning [150]. Since Viola, P. et al. utilized the Adaboost learning algorithm to process face detection and pedestrian detection [151], Adaboost has been used in the research of pedestrian detection. The operating procedures inside Adaboost are easy to comprehend. During the classification operation, a large amount of features are simultaneously transmitted into the first weak classifier. After the first weak classifier finishes its classification, the selected features are transmitted into the second weak classifier. Meanwhile, the latter classifier has a chance to correct the errors made by the former classifier. After the classification process of several weak classifiers, the final features are regarded as the best pedestrian description features. Those weak classifiers are also combined together to generate the final strong classifier. As a result, the whole process of Adaboost classification is speeded up; this is because the classifiers only concentrate the left features [8, 152]. Adaboost is not only used with a single type of pedestrian description features, but also used with multiple types of pedestrian description features to detect pedestrians. Because one Adaboost is not enough to provide effective classification among different pedestrian features, the method of training multiple strong classifiers is proposed in [153] and [154]. Besides the Adaboost learning algorithm, there are variants of boosting algorithms used in pattern classification; the LogitBoost used in [155, 156, 157] and the Gentle Adaboost used in [130, 131, 158]. Some of the classical pedestrian detection methods which used the Adaboost algorithm are listed in Table 2.4.

The neural network is another feature training algorithm. Munder, S. et al. [51] combined the feed-forward multi-layer neural network with local receptive fields (NN-LRFs) to detect pedestrians. Zhao et al. [134] proposed a stereo-camera-based pedestrian seg-

mentation approach that utilizes the intensity gradients and uses the feed-forward neural network. Gavrila, D. M. and Munder, S. [135] proposed their real-time pedestrian detection and pedestrian tracking method. In this method, they used a tight integration of consecutive modules and a feed-forward neural network to achieve the detection process and tracking process, which has been proven to balanced robustness and efficiency.

2.2 Detection Distance and Detection Range Concern

In the beginning, two kinds of concepts are required to be specified: the detection range and the detection distance. The detection range is the interval between the position where detection starts and the position where detection ends. The detection distance is the distance measured from the system position to the position where a pedestrian can be detected. For instance, the PDS can detect a pedestrian in a range between 5 and 30 meters. The range from 5 to 30 meters is the detection range; and, the distance of 5 meters and 30 meters represent the minimum and maximum detection distance respectively.

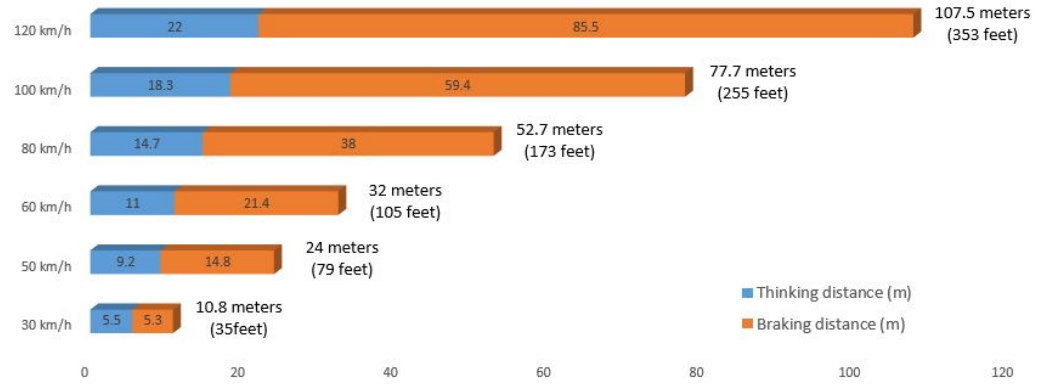
In the research on pedestrian detection, the majority of researchers primarily focus on improving the detection accuracy, rather than on extending the detection range of the detection system. The PDS is an application for the advanced driver assistance system, which is used on vehicles to prevent traffic accidents. The detection distance and the detection range are required to be larger than the stopping distance of vehicles. Thus, the system can prevent collisions by early pedestrian detection, as the example shown in Figure 1.1. Meanwhile, the emergency communication between vehicles can prevent the accidents caused by urgent braking.

2.2.1 Necessity of Extending the Detection Range

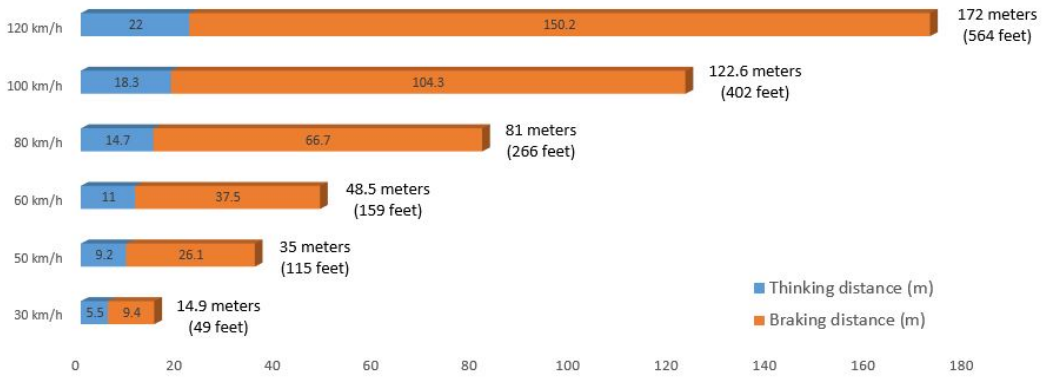
The required detection range is related to the stopping distance of vehicles in different scenarios. For a given driving speed, the stopping distance D_s should be considered. The D_s is mainly determined by two crucial factors: thinking distance D_t and braking distance D_b [159]. The D_t is defined as the distance traveled during a reaction time. It is determined by the vehicle velocity V_c and the perception time T_p, which is defined as the minimum time taken by drivers to recognize a threat. The thinking distance is described as $D_t = T_p \times V_c$. The T_p usually takes around 1.5s. The braking distance D_b refers to the distance required to halt the car after applying the brakes. The time T_b is the time interval required for the act of applying the brakes which results in the vehicle being totally halted. D_b depends on car velocity V_c and the friction coefficient between vehicles and the ground.

The braking distance is described as $D_b = \frac{V_c^2}{2\mu g}$, where μ is the friction coefficient and g is the gravitational acceleration. The stopping distance $D_s = D_t + D_b$ can then be obtained. We note that the D_s may also be affected by some other factors, such as: vehicle conditions (tires, brakes), pavement conditions (paved, not paved, holes), weather conditions (snow, wet, dry, frozen) and the driver's health (physical and mental).

The stopping distances in different scenarios are illustrated in Figure 2.9, according to the data from the Transport Research Laboratory (UK), ©Road Safety Authority 2007 [9] . The stopping distance in wet conditions is 50% longer than the one in dry conditions. The detection range of the PPS is then required to be larger than the stopping distance in wet conditions.



(a) Stopping distance in dry conditions



(b) Stopping distance in wet conditions

Figure 2.9: Stopping distance in different scenarios [9].

Table 2.5: General detection range [1].

Authors	Detection Range	Publications
Gavrilla et al.	5 – 25 meters	[160][161] [135]
Grubb et al.	till 30 meters	[162]
Shashua et al.	3 – 25 meters	[90]
Soga et al.	till 40 meters	[163]

2.2.2 Approaches of Extending Detection Range

The detection range is not widely discussed in the research on pedestrian detection. The detection distance is mentioned in [1, 164, 135, 165, 53]. In general, the detection distance should be considered with the corresponding driving speed. If the driving speed is fast, the maximum detection distance should be larger than the total stopping distance in order to avoid collision. In [1], the authors classified the pedestrian databases into different ranges. Likewise, they also summarized the detection range of a few published pedestrian detection methods (see Table 2.5), which emphasize that the detection range does need to be extended. Dollar et al. [164] focused on the detection distance; and, they exhibited the relationship between the detection distance and the height of pedestrians. Moreover, they defined detection range as the near scale, the medium scale and the far scale determined by the pedestrian height in pixels (height in frames). In their conclusion, most pedestrians (69%) are detected by systems at the medium scale (30 – 80 pixels). They also claimed that the medium scale pedestrian is only 4 seconds away from the car which is traveling at a speed of 55km/h. Their conclusion indicates that that detection distance and the detection range are extremely important.

To achieve large range pedestrian detection, the LIDAR or millimeter radars are commonly used. However, the cost of these kinds of devices is extremely expensive; and, the detection accuracy of millimeter radar is not robust. In contrast, the cost of the vision-based pedestrian detection methods is affordable and the detection accuracy performs at a high level. Meanwhile, improved vision-based pedestrian detection can provide detection within the range of 100 meters. Based on some publications, the methods of extending the detection range can be classified into two groups. One of the groups is the stereo-camera-based extending approach, which is discussed in [166, 167, 168]. The other group consists of the monocular-camera-based extending methods, introduced in [169, 170, 171, 172]. The method proposed in [166] is achieved by using a combination of stereo cameras and the convolution neural network (CNN). The authors in [166] novelly used two sets of stereo cameras and coordinated these two sets of stereo cameras with two CNNs. One of the CNN, is used to classify the near range detection results, which are the appearances of

pedestrians and stereo disparity-based features. The other CNN is used for long-range detections which use appearances of pedestrians only. However, the cost of two sets of stereo cameras is expensive. The image resolution used in the system is 2400×2000 pixels, which results in slower processing speed. The authors in [167] proposed a stereo-based pedestrian detection and distance measurement system. To measure the distance, triangulation is used when identical features of the same pedestrian are located. Yang et al. [168] introduced their pedestrian detection method, which uses 3D and 2D cameras simultaneously. They constructed a surface parallax map (SPM), which calculates the parallax; this is done to detect pedestrians who do not belong to the road plane. An occlusion image is then generated to detect pedestrians by locating high density areas. In monocular-camera-based detection, the extension of the detection range is usually achieved by utilizing either the far-infrared cameras [170, 172] or the smaller sized pedestrian features [169, 171]. The performance of infrared-camera-based detection method is easily affected by the changes in luminance. The detection accuracy of this method is not robust either. Likewise, the detection approach which uses smaller sized pedestrian features cannot guarantee detection accuracy. More false positive detection results will occur with the use of smaller sized features. In conclusion, both of these methods cannot meet requirements, due to detection range or detection accuracy. Thus, novel approaches are needed to achieve the extension of detection ranges without yielding detection accuracy and processing speed.

Because there are not many reviews about extending the detection range or about undergoing pedestrian detection over long distances, it is difficult to give a comparison of the stereo-camera-based methods and the monocular-camera-based methods. From the materials we obtained, there are two main difficulties which have constrained the extension of the detection range. One of them is the feature extraction and the other difficulty is the resolution problem. Pedestrians who are 50 meters away are usually less than 40 pixels in 960×720 frames [164]. Feature extraction is a tough problem for 640×480 image resolution. Smaller feature sizes and higher image resolution are utilized cooperatively to detect pedestrians in distant ranges. However, the utilization of smaller feature sizes results in high false positive rates and lower detection accuracy. The enlarged image resolution decreases the processing speed; this has not been effectively solved up to this time. So the extension of the detection range is still a difficult problem to be solved.

2.3 Pedestrian Tracking

The tracking process is supplementary part of the PPS following the detection process. The Pedestrian Tracking System (PTS) is utilized following the PDS; it acts as an important

component of the PPS. The PTS refines the detection results of pedestrians. The position of the detected pedestrian can be thus more precise. Meanwhile, PTS is used to predict the motion and the behavior of pedestrians in the next frame. The predictions help the PPS judge the danger level and alert drivers about upcoming situations.

The tracking process is used to fulfil several objectives: to predict future pedestrian positions, to track the movement trajectory of pedestrians, to estimate pedestrian motions and behaviors and to decrease the influence of occlusions [173]. Multi-tracking in complex environment is discussed in [174]. Authors in [175] proposed an object tracking method in the multi-camera-based system. The tracking process in crowded scenarios is discussed in [176]. The multi-pedestrian tracking process in the mobile vision system in crowded scenarios is described in [177]. During the operation of the tracking process, if a pedestrian can be detected in several sequential frames, the tracking process can then predict the future position and movement direction of this pedestrian.

The tracking process can be used following the static detection approach or the dynamic detection approach. Some researchers have given their reviews in [178, 55, 54, 53, 1, 179]. In these publications, the following widely used tracking methods are discussed: the Kalman filter, the particle filter, the Meanshift and the Continuously Adaptive Meanshift (Camshift). The Kalman filter and the particle filter can be regarded as different branches of recursive Bayesian estimation. Moreover, the Meanshift and Camshift are non-parametric feature-space analysis techniques which perform the estimation based on gradient density.

In general, Meanshift was first proposed in [180]. Yizong Cheng [181] improved the original mean shift algorithm by introducing the weighted kernel-based estimation tracking method. Camshift [182] is the improvement of the Meanshift. The adaptive adjustment was added to Camshift to adjust the size of the detection window and the rotation of the detection window. During the implementation of Meanshift, this algorithm computes the density of each area. The area with the maximum density can then be located, based on a given discrete data sample. The initial discrete data is the detection window given by the detection result. Once the initial detection window is transmitted to the Meanshift tracking process, the partial image in this window is then transformed into a histogram image, according to the density function. The whole original image is then transformed into a new confidence map based on the histogram of the partial image input. The density calculation function of Meanshift algorithms compares the density of the original partial image with the new confidence map; this is done to locate the maximum density in the confidence map. The position of the maximum density in the confidence map is the probable object position of the partial image input in the original image. In the pedestrian tracking process, Meanshift is commonly used in color videos after part-based pedestrian detection. Comaniciu et

al. [183] utilized the Meanshift to track objects in irregular shapes. Liping Yu et al. [184] proposed their Meanshift-based tracking approach to fuse pedestrian detection results. Li et al. [185] described their occluded pedestrian tracking method by using the Meanshift algorithm. After the different parts of pedestrians are detected, the detection results are fused by the Meanshift tracking process into a integrated result [186]. Camshift is also used in part-based pedestrian detection and tracking systems as well [187, 188, 189].

The Kalman filter [190, 191] is widely used in the pedestrian tracking process. The Kalman filter can be designed into a framework for the tracking process, to predict the future position of detected pedestrians and to increase the precision of the detected pedestrian positions. The Kalman filter is established on the recursive Bayesian estimation. The covariance of the Kalman filter is minimized when the error terms and the measurements are in the Gaussian distribution. During the processing of Kalman filter, the position of the detected pedestrian can be refined based on the current and previous detection results. The Kalman filter can also predict the future motion and movement direction, based on the updating of latent functions in the Kalman filter algorithm. The original Kalman filter is only used in linear Gaussian models, which restricts the utilization of the Kalman filter. To expand the usable range of the Kalman filter, the Extended Kalman filter (EKF) and Unscented Kalman filter (UKF) [192], which are both non-linear or non-Gaussian models, have been proposed. The basic idea of the EFK and UKF is the same: to transform the non-linear problem into the linear problem and to use the original Kalman filter to solve the transformed linear problem. The EKF and UKF have been used in many pedestrian detection systems, [193, 194, 195, 196]. Franke et al. [197] proposed their pedestrian localization and tracking system, in which the Kalman filter was configured as a tracker to reconstruct an interpretation of pedestrian positions in the scene. Bertozzi et al. [198] used Kalman filter to track silhouette features. Furthermore, they proposed a system in which a labeler and a predictor was used [199]. The labeler is used to memorize the past history of the detector and predictor. The predictor, a Kalman filter, is used to estimate the new position of pedestrians by utilizing the previous movements, to help the labeler to promote matching results.

The particle filter [200] was proposed (it has been names the bootstrap filter) to improve the tracking performance in the situation of non-linear and non-Gaussian tracking. It was developed by combining the Beyesian recursion equation and Sequential Monte Carlo (SMC) methods. The particle filter uses a grid-based approach and a set of sampling particles to represent the posterior density. These sampling particles are randomly and unrestrictively assumed from the state-space. The particle filter updates its posterior density with the newly detected results. Afterwards, these particles are distributed, regenerated and weighted according to the Bayesian recursive estimation; this is done to

obtain the posterior density of the future state. When the particle filter tracking process is in operation, these procedures are continuously iterated to track the object. The particle filter is not restricted by the linear problems and the Gaussian models, so it can be used in the estimation of non-linear and non-Guassian distribution problems. Philomin et al. [201] utilize the particle filter to track pedestrians and they have obtained good tracking performance. The particle filter has been thus increasingly used in PPSs to enable the tracking process [202, 203, 204, 205]. Besides, different variants of particle filters have been introduced over the recent years, which have improved the compatibility of the original particle filter (some of the variants of particle filters are shown in Table 2.6).

Table 2.6: Variants of the particle filter.

Year	**Variants**	**Related References**
1999	Auxiliary particle filter	[206]
2004	Appearance adaptive particle filter	[207]
2005	Monte Carlo filters	[208]
2009	Motion-based particle filter	[209] [210]

Apart from the mostly used tracking methods, there are also other tracking methods proposed and described by researchers. Porikli et al. [211] proposed the covariance tracker method. Multiple instance learning [212] is a significant new tracking approach. Andriluka et al. [213] proposed the tracking-by-detection and detection-by-tracking method. These tracking approaches discussed above have been widely utilized in the research on computer vision, in the areas of pattern recognition, machine learning and image processing. Meanwhile, researchers are continuously interested in the innovation of the tracking process.

Chapter 3

Design of Pedestrian Protection System

The vision-based PPS is composed of the PDS and the PTS. The PDS is used to proceed pedestrians detection in front of the vehicles. The detection range of the PDS is extended by our novel multi-camera-based detection approach. After the optimization process, the detection results are then transmitted to the PTS, which is used to track the detected pedestrians. Based on the PTS, the movement trajectory of pedestrians can be tracked and the future movement direction can be predicted. The tracking and predicting results are a great assistance in the protecting of pedestrians enabled by alerting drivers concerning the upcoming dangerous situation or by providing a warning. The process of our PPS is drawn in the flow chart in Figure 3.1.

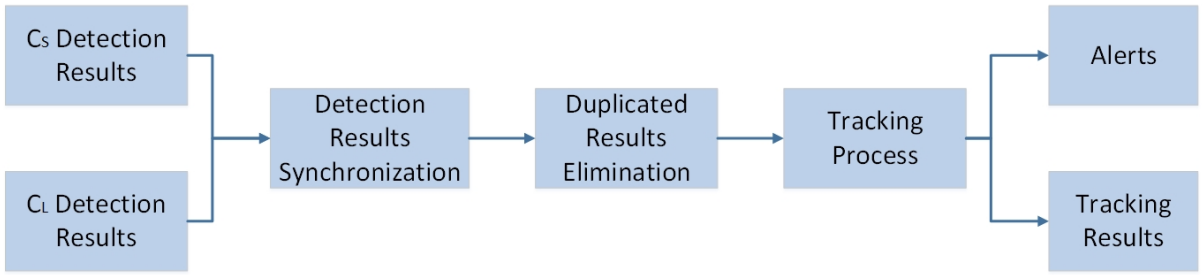


Figure 3.1: Design of the Pedestrian Protection System.

3.1 A Novel Pedestrian Detection System (PDS) – Extending the Pedestrian Detection Range

3.1.1 Overview of the Novel Pedestrian Detection System

In the PDS, the HOG features are used as the pedestrian descriptor; and, the SVM is used as the classifier. HOG descriptor is composed of HOG features to describe pedestrians. The combination of HOG and SVM performs well on pedestrian detection in the aspect of detection accuracy efficiency. The influence of changing of illumination and the impact of different body postures can be reduced to a minimum level. Meanwhile, we can estimate the maximum and minimum detection distance for each specified scenario based on the HOG descriptor size. In our PDS, the HOG features are computed by GPU. By this means, the processing speed of GPU-based pedestrian detection can be 10 times faster than the one of CPU-based pedestrian detection, which is acceptable in practical implementation.

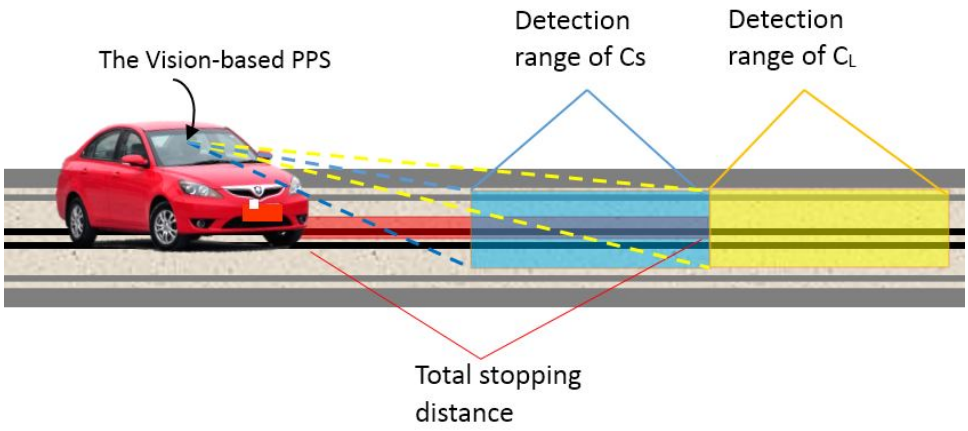


Figure 3.2: Overview of the Pedestrian Detection System.

The structure of the detection system is composed of two identical cameras, as shown in Figure 3.2. One camera is equipped with a lens with a long focal length, called C_L. This camera is used to detect the pedestrians from mid to far-scale. The other camera is equipped with a lens with a short focal length, called C_s. The C_s is used to detect pedestrians from the near to mid-scale. The detection results of C_s and C_L are then synchronized by the synchronization process. Meanwhile, as the detection range of each camera will change according to the driving speed, there will be an overlapping detection area, in which pedestrians will be repeatedly detected. The repeated detection results will be eliminated by our elimination process, in which the algorithm will not decrease the processing speed.

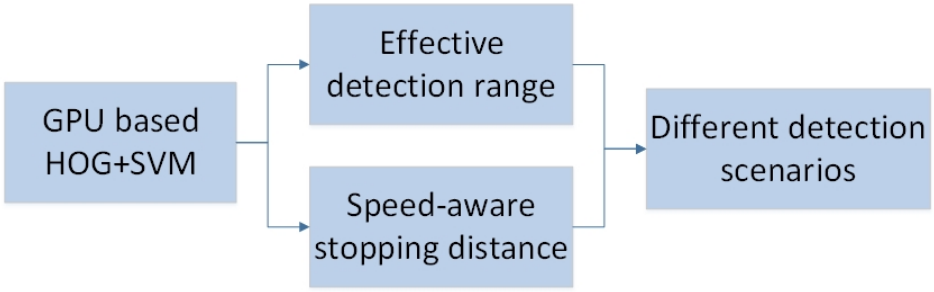


Figure 3.3: Detection scenarios determination.

In different driving scenarios, we set the detection range according to the stopping distance and the effective detection range of HOG descriptor, as the example shown in the flow chart of Figure 3.3.

3.1.2 Pedestrian Detection Algorithm

After comparing the advantages and disadvantages of difference feature-classifier-based detection methods, we have finally decided to use the HOG + SVM technique to implement the novel PDS. For a successful pedestrian detection application, accuracy and processing speed are the two most crucial factors considered in this thesis. The combination of HOG descriptors and SVM classifier can guarantee the detection accuracy; and, we use the GPU-based parallel computing technique to increase the detection processing speed. Thus, the detection accuracy and processing speed can be satisfied simultaneously.

3.1.2.1 Histogram of Oriented Gradients

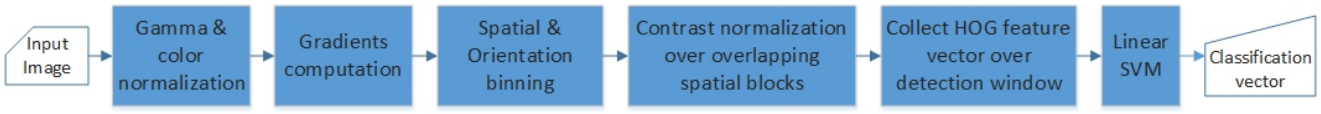


Figure 3.4: Flow chart of HOG features extraction [6].

To make our PDS robust, we use the HOG features to describe pedestrians. The computation of HOG features is composed of 6 main procedures (Figure 3.4):

- Gamma & color normalization
- Gradients computation
- Spatial & Orientation binning

- Contrast normalization over overlapping spatial blocks
- Collect HOG feature vector over detection window
- Linear SVM

Usually the first step is the gamma and color normalization of the input image. The horizontal and vertical gradients computations for each pixel are then achieved by using the operator $\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$ and $\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}^T$:

$$F_x(x,y) = V(x+1,y) - V(x-1,y) \tag{3.1}$$

$$F_y(x,y) = V(x,y+1) - V(x,y-1) \tag{3.2}$$

Where $F_x(x,y)$ is the the horizontal gradient of pixel (x,y) and $F_y(x,y)$ is the vertical gradient of pixel (x,y); $V(x,y)$ is the gray value at (x,y) pixel.

From Equations (3.1) and (3.2), we derive the norm which is the weight of pixel (x,y) and the orientation for each pixel:

$$F(x,y) = \sqrt{F_x(x,y)^2 + F_y(x,y)^2} \tag{3.3}$$

$$\theta(x,y) = \tan^{-1}(\frac{F_y(x,y)}{F_x(x,y)}) \tag{3.4}$$

Where $F(x,y)$ is the weight of pixel (x,y) and $\theta(x,y)$ is the orientation of pixel (x,y).

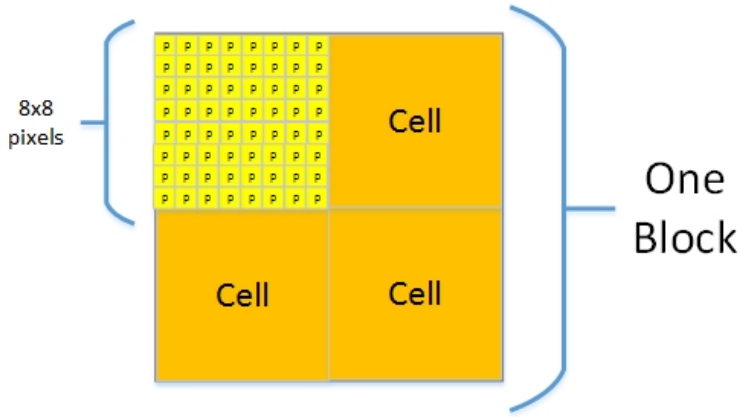


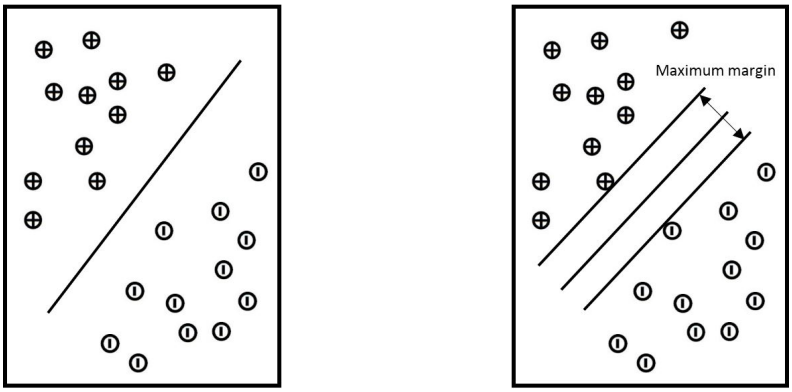
Figure 3.5: Components of HOG features.

At the same time, the input image is divided into cells and blocks. Each cell contains 8×8 pixels and each block contains 2×2 cells (shown in Figure 3.5). We classify $\theta(x,y)$ into 9 bins in each cell and each bin represents one orientation (we divide $180°$ into 9 bins and each bin has a range of $20°$). The weight of each pixel will then be grouped according

to bins in which the weight the each pixel is located. The next step is to accumulate all the weights in each bin. For each cell, a vote is set to decide which bin has the largest weight. The orientation and weight in this bin should be kept to represent this cell. A block acts as a conclusion for all the cells and bins, and it keeps the data of all the cells.

After finishing HOG feature extraction, the HOG descriptor size for the pedestrian detection should be decided upon; this is also known as the minimal sliding window size. Each HOG descriptor size has a distinct influence on the detection results. Usually the size of the HOG descriptor is set as 64×128 or 48×96 in pixels; this is because the size of 64×128 or 48×96 can provide good detection accuracy [6]. Besides setting up the HOG-feature-based pedestrian descriptor, there are two other parameters that should be determined as well: the increasing times of the size of the sliding window and the increasing ratio (nlevel and scale). These two parameters will affect how many times the sliding detection window will increase and have influence on the final detection results. Still, based on the research of predecessors [6, 164], when the nlevel is equal to 64 and the scale is equal to 1.05, the detection process can provide the best performance.

3.1.2.2 Support Vector Machine



(a) (b)

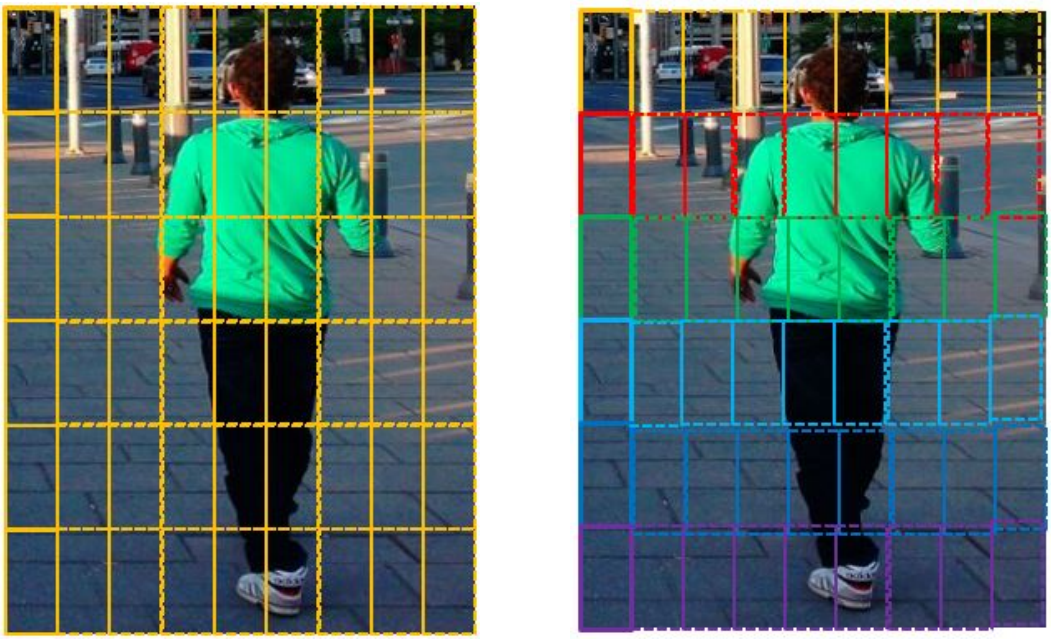
Figure 3.6: An example of SVM-based classification in 2D plane — the positive and negative training data are mapped to the hyperplane coordinate system, separated by the SVM. Meanwhile, the SVM will find a hyperplane to maximize the margin between positive data and negative data.

Support Vector Machine (SVM) is a supervised learning method which can be widely used in classification and regression analysis. SVM is also an efficient classification method used to minimize the empirical error and to maximize the geometric edge of the area (shown in Figure 3.6). The version of SVM we are using is the SVMlight, a linear classifier

and a maximum marginal zone classifier, which can classify the positive data and negative data quickly. SVMlight [144, 214] is an open source software with good classification performance. We use it to obtain the classification results of pedestrian HOG descriptors.

3.1.3 GPU-Based Acceleration of Pedestrian Detection

Throughout the whole detection process, the system spent most of the time on calculating the HOG features of the image, which caused the decrease of the detection rate. Currently, the maximum cores of a CPU in PCs are 8, but a normal GPU has more than 100 cores. Kari Pulli et al. [10] declared that the processing rate of pedestrian detection with the acceleration of NVIDIA CUDA can be 8 times faster than the original processing rate on average (shown in Figure 3.8). In OpenCV, there are libraries which can directly support the NVIDIA CUDA. We therefore decide to use the external OpenCV library. By this means, we can utilize the mutli-core parallel computing capacity of NIVIDA GPU and to use GPU as the main computing processor [126].



(a) CPU-based HOG features computation process

(b) GPU-based HOG features computation process

Figure 3.7: Differences of CPU and GPU based HOG features computation process.

The CPU-based calculation can perform each single task at a good computing speed, but the whole process cannot be processed that quickly due to the sequencing computation(shown in Figure 3.7a). However, when GPU is used in the calculation process, its

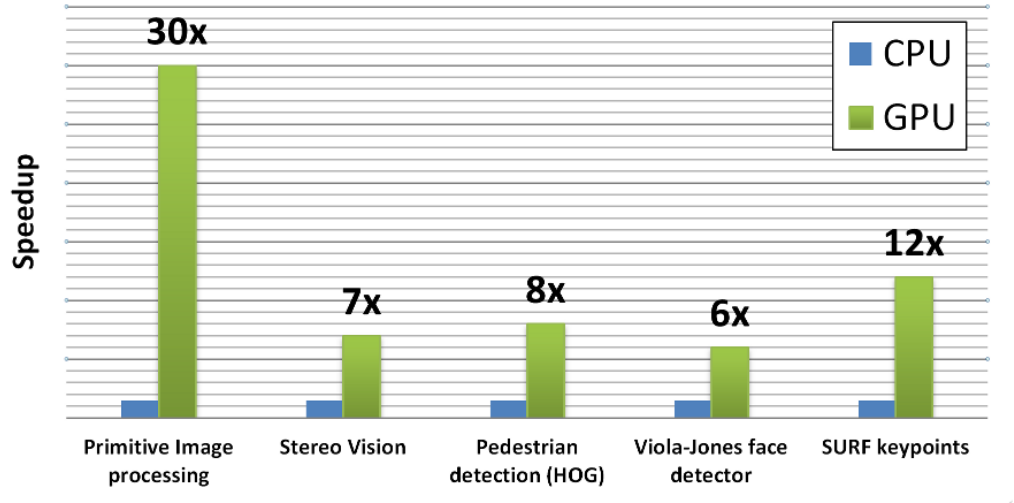


Figure 3.8: Tesla C2050 versus Core i5-760 2.8Ghz, SSE, TBB [10].

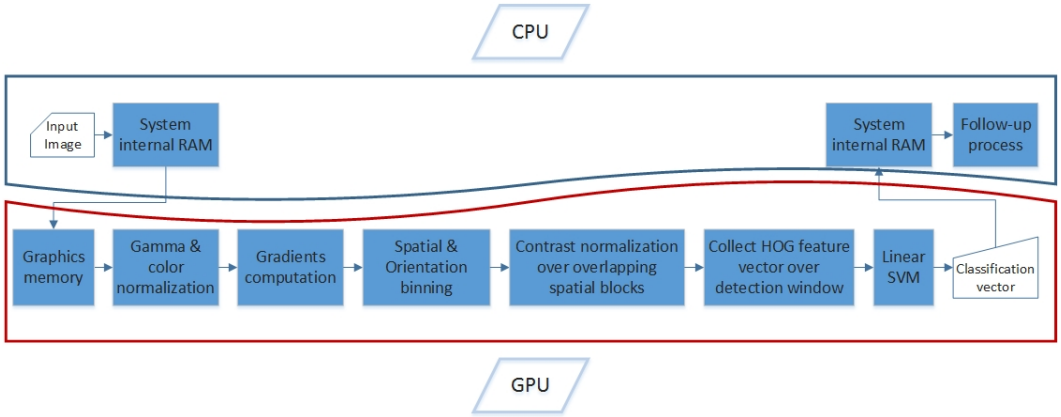


Figure 3.9: GPU-based calculation: the upper-most illustrations show how CPU works when GPU is used in the system. The process illustrated in the bottom half of the figure shows that the HOG features are mainly computed by GPU which transmits the results back to the internal RAM of the system.

parallel computing ability can be fully utilized (shown in Figure 3.7b). The system transmits the captured image from RAM to graphics memory, this image is then divided into several portions by the CUDA internal algorithm, on the basis of how many cores there are in the graphics card. All the graphic processing units read partial image information and process the HOG feature computation on this portion. When all the calculations for these portions are done, the HOG features of this image are transmitted from graphics memory back to RAM. The system can continue to the next procedure. The whole procedures of GPU-based calculation is shown in Figure 3.9.

3.1.4 Effective Detection Range of PDS

Recall that there are two different concepts: detection range and detection distance. The detection range is the interval from a detection start position to a detection end position. Meanwhile, the maximum or minimum detection distance is the detection distance from the system position to the maximum or minimum position where pedestrians can be detected by the system. The PDS is fixed on the middle of windshield of vehicles. From our experiment, the original HOG + SVM detection method can only detect pedestrians within a range of less than 30 meters (in the 960×720 resolution). We want to extend the detection range and to make it suitable for practical utilization. As aforementioned, we use two lens-equipped cameras with unequal focal length to provide pedestrian detection in different ranges. In what follows, we determine analytically the range (minimal and maximal distance) covered by our system.

For a PDS, the detection range has two dimensions. One is the width range (detection range in width) and the other is the distance range (detection range in length). For pedestrian detection, we need to ensure both ranges satisfy the requirements. The schematic diagram of detection range in length is shown in Figure 3.10 and the schematic diagram of detection range in width is shown in Figure 3.11. The parameter transformation from parameters in the real world to image is shown in Figure 3.12.

3.1.4.1 The calculation of detection range in length

When talking about detection range in length, we need to ensure that the maximum detection distance is larger than the stopping distance and that the minimum detection distance is smaller than the nearest braking distance. Hence, pedestrians can be protected when the vehicle is braking or accelerating. The calculation of the detection range is based on the detection algorithm and the hardware we use. Before we calculate the detection

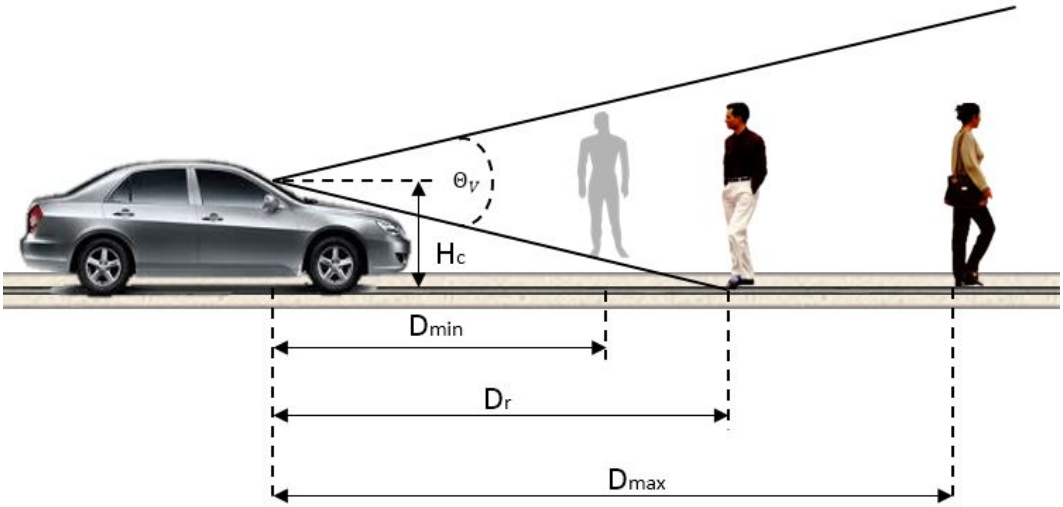


Figure 3.10: An example of detection range in length — H_c is the height of vehicle (m); D_{max} is the maximum theoretical detection distance (m); D_{min} is the minimum theoretical detection distance for camera (m); D_r is the minimum required distance to capture ground in the picture (m); θ_v is the vertical field of view (°).

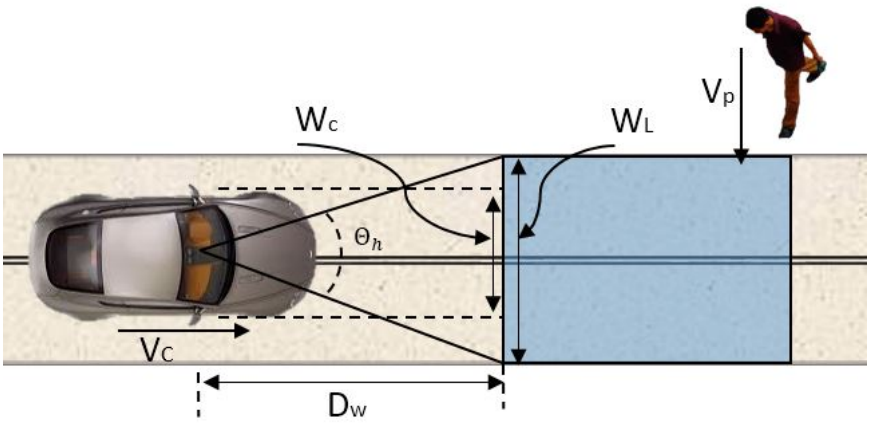


Figure 3.11: An example of detection range in width — W_L is the width of lane (m); W_c is the width of vehicle (m); D_w is the minimum distance for the system to cover the width of lanes in a picture (m); θ_h is the horizontal field of view (°); V_p and V_c represent the speed of the pedestrian and the car respectively.

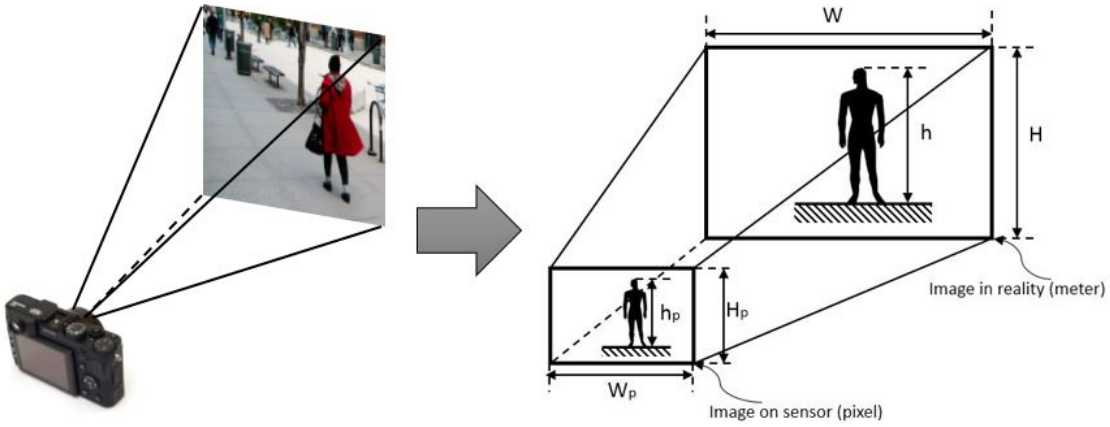


Figure 3.12: An example of parameter transformation — h is the real height of pedestrian (m); H is the real height of the entire subjects captured in picture (m); W is the real width of the entire subjects captured in a picture; h_p is the height of pedestrians in picture (pixel); H_p is the total vertical pixels of the picture (pixel); W_p is total horizontal pixels of the picture (pixel).

distance, we need to obtain the horizontal and the vertical field of view as follows:

$$\theta_h = 2 \times \arctan(\frac{d_h}{2f}) \tag{3.5}$$

$$\theta_v = 2 \times \arctan(\frac{d_v}{2f}) \tag{3.6}$$

Where θ_h is the horizontal field of view (FOV); θ_v is the vertical field of view (FOV); d_h represents the horizontal size of the sensor; d_v represents the vertical size the sensor; f is the focal length.

Note that the detection distance D from a camera to a pedestrian can be given by:

$$\tan(\frac{\theta_v}{2}) \times D = \frac{H}{2} \tag{3.7}$$

Where H is the height of the real image in meters and θ_v is the vertical FOV of the camera.

From Figure 3.12, the relationship between the real height h of a pedestrian and its correspondent height in pixels h_p is:

$$\frac{h}{H} \times H_p = h_p \tag{3.8}$$

Where H is the height of the real image in meters and H_p is the height of the image in pixels.

From Equations (3.7) and (3.8), the distance D from a camera to a pedestrian can be given as follows:

$$D = \frac{H_p \times h}{2 \times \tan(\frac{\theta_v}{2}) \times h_p} \tag{3.9}$$

In fact, the distance D varies from the minimum detection distance D_{min} to the maximum detection distance D_{max} regarding the values of $\theta_v \in [\theta_{v-min}, \theta_{v-max}]$ and $h_p \in [h_{p-min}, h_{p-max}]$. θ_{v-min} and θ_{v-max} correspond to the minimal vertical FOV and the maximal vertical FOV, respectively (shown in Figure 3.13). To detect near pedestrians, the FOV of the camera is set to θ_{v-max}, whereas the FOV of the camera is set to θ_{v-min} to detect far pedestrians. Hence, it is easy to deduce the calculation of D_{min} and D_{max} as follows:

$$D_{max} = \frac{H_p \times h}{2 \times \tan(\frac{\theta_{v-min}}{2}) \times h_{p-min}} \tag{3.10}$$

$$D_{min} = \frac{H_p \times h}{2 \times \tan(\frac{\theta_{v-max}}{2}) \times h_{p-max}} \tag{3.11}$$

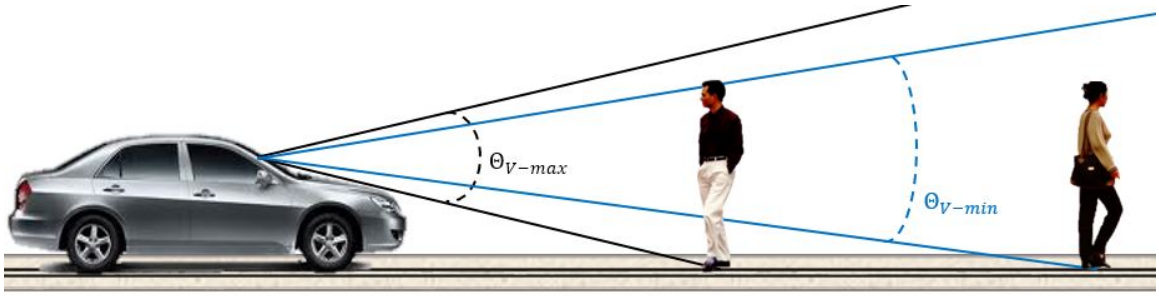


Figure 3.13: Description of θ_{v-min} and θ_{v-max}.

In general, detection should begin at the distance where the ground is captured in the picture. More precisely, by using HOG, the pedestrian who is located at D_{min} from the vehicle cannot be recognized since their lower body is not detected. This is due to the fact that the HOG-feature-based pedestrian detection is considered as a holistic approach. In practice, when the HOG features are used to implement pedestrian detection, the real (true) minimal detection distance is not only determined by D_{min} given by Equation (3.11), but also determined by the height of a car H_c. In this case, the minimal required distance D_r (as shown in Figure 3.10), for which the whole body is detected is as follows: :

$$D_r = \frac{H_c}{\tan(\frac{\theta_v}{2})} \tag{3.12}$$

As a conclusion, the detection range in which pedestrians can be detected by the PDS is from $Max\{D_r, D_{min}\}$ to D_{max}. Experimentally, we find that D_r is always greater than D_{min}.

(a) Recognizable example (b) Unrecognizable example

Figure 3.14: Examples that can or cannot be recognized by PDS.

3.1.4.2 The calculation of detection range in width

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

(k) (l) (m) (n) (o) (p) (q) (r) (s) (t)

Figure 3.15: Examples of positive training images [6].

For driving vehicles, if the horizontal field of view is not wide enough, collisions may occur between vehicles and pedestrians (shown in Figure 3.11); thus, importance should be attached to the horizontal FOV as well. This helps to ensure that the width of each frame can cover the width of the road in front of vehicles. For the horizontal FOV, there are two issues that should be premeditated.

The first issue is the width of lanes. The system should guarantee that the horizontal detection range is wide enough to cover the width of lanes. Based on the Equation (3.11) for

the minimum detection in length, the minimum horizontal detection range is understood as:

$$D_w = \frac{W_L}{2 \times \tan(\frac{\theta_{h-max}}{2})} \tag{3.13}$$

To ensure that the horizontal detection range is wide enough to protect pedestrians, the minimum detection distance in length D_{min} (Equation (3.11)) should be smaller than D_w (Equation (3.13)). This means that when detection begins, all the required width and height is already covered.

The second issue is the interaction between pedestrian velocity and vehicle velocity. There is a possible concern that pedestrians may not be detected until a collision has occurred. The occurrence of this lapse is due to vehicle velocity and the limited horizontal detection range. If the horizontal detection range is not wide enough and vehicle velocity is fast, pedestrians may not be detected when a car is driving towards them. This means that pedestrians are always outside of the horizontal detection range until the moment when the collision has occurred. To prevent missing the detection for a foreseeable collision, the PDS should guarantee that pedestrians are detected before they walk into the driving direction of a vehicle. This means that the time $t_c = \frac{D}{V_c}$ required by the car to travel the distance D should be less than the time $t_p = \frac{D \times \tan(\theta/2)}{V_p}$ taken by the pedestrian before meeting the car, where V_c and V_p are the velocity of vehicle and pedestrian, respectively. The requirement of the horizontal detection range can be formulated as follows:

$$\frac{D}{V_c} < \frac{\tan(\frac{\theta_h}{2}) \times D - \frac{W_c}{2}}{V_p} \tag{3.14}$$

Where V_c is the vehicle driving velocity and V_p is the pedestrian walking speed, which is usually less than 1.5m/s. W_c is the width of vehicles, which is usually less than 2.6 meters [215].

From Equation (3.14), we can see that the horizontal FOV θ_h is mainly influenced by vehicle speed V_c, pedestrian speed V_p, distance between vehicle and pedestrian, and width of the lane. The horizontal FOV θ_h should be:

$$\theta_h > 2 \times \arctan(\frac{W_c \times V_c + 2 \times D \times V_p}{2 \times D \times V_c}) \tag{3.15}$$

Equation (3.15) shows that the required horizontal FOV is increasing with the decreasing speed of the car V_c.

There are two parameters that can be changed: the distance between vehicles and pedestrians and the velocity of vehicles. For the first case, when the velocity is changing, the detection range in length should be guaranteed to be larger than the total stopping distance. The total stopping distance should be regarded as the D in Equation (3.14). At

the total detection distance D, the horizontal detection range should satisfy the requirement based on Equation (3.14). At this time, the width of lane W_c cannot be ignored as the $\tan(\frac{\theta_h}{2}) \times D$ is not much larger than it. If the horizontal field of view is large enough to cover the required horizontal detection range at the total stopping distance, the required horizontal field of view will decrease with the extending of the detection distance D. So, we could conclude that the required horizontal FOV is mainly influenced by the driving speed V_c, and that it has a relationship with the detection distance as well.

3.1.4.3 Calculation Explanation

One important element we need to emphasize is that the detection distance is only calculated by the vertical field of view and by the height of HOG descriptors (the height of the sliding window). There are three explanations for why the horizontal field of view and the width of HOG descriptor are not used to calculate the detection distance.

The first reason is caused by the training image datasets. The sliding window is mostly defined as the ratio 1:2: 48×96 or 64×128 pixels. The training images of our PDS are from INRIA France. The positive training database from which most images are derived does not physically equal to 64×128 or 48×96. So a shrinking and extraction process is required for all the images. When doing the shrinking and extraction procedure, the entire bodies of persons in the positive training database are strictly extracted using the ratio of 1:2 (width over height). However, the width-height ratio of persons in the positive training image are smaller than 1:2, so the width and height of pedestrians in the final extracting-ready images are not the same size as the sliding window, as illustrated in Figure 3.15a. Consequently, the pedestrians in positive training images are as tall as the sliding window in most cases; but, they are not as wide as the sliding window. So the detection distance calculation should use the vertical parameter because the vertical parameter can get much more accurate results than the horizontal parameters (shown in Figure 3.15b – 3.15t).

The second reason involves the pedestrian standing postures and the contrast normalization procedure of HOG feature computation (talked about in Section 3.1.2.1). As the positive training images consist of a large amount of distinct standing gestures, the width of the HOG descriptor representing persons in each positive training image is different. Therefore, the horizontal size of HOG descriptor (the width of the sliding window) is not suitable to perform detection distance calculation; and, it cannot give accurate calculation results. During the HOG feature computation process, the contrast normalization procedure normalized the disparities, which decreased the describing accuracy of the horizontal size of HOG features in the case of specific persons; however, it increased the universal applicability. After the normalization, the width of the HOG descriptor which represents

the width of pedestrians is not predictable. So the detection distance cannot be calculated by the width of the HOG descriptors.

(a) Pedestrian detected by small sliding window

(b) Pedestrian detected by large sliding window

Figure 3.16: An example of sliding window resizing.

The third reason is the unequal ratio of the sliding window and the streaming frames. For most cameras, the frames are captured in the ratio of 16:9 or 4:3; this means that the width of the frame is larger than the height of the frame. Meanwhile, the sliding window of pedestrian detection in our system is 1:2. When the detection process is running, the sliding window slides one by one in the frame. The sliding window size will change from the size set (64×128 or 48×96) to the maximum size it can reach (based on the scale value and the scale levels we set). The scale value used in the system is 1.05 since [6] has proven that the performance of the parameter and the scale level is set in 41 which is large enough to make the sliding window cover the entire height of frames. The sliding window increases based on the scale value and scale levels. The height of the sliding window will reach the height of the frame first while the width of sliding window will much smaller than the width of the frame, like the one shown in Figure 3.16. For instance, one level of resolution we used in the implementation is 960×720; and, the size of the HOG descriptor we used is 48×96. The ratio between the frame and the size of HOG descriptor is 7.5 vertically and 20 horizontally. The sliding window will give the iteration after each active sliding. For the equations $1.05^x = 7.5$ or $1.05^x = 20$, x is equal to 41 or 62. When the scale iteration is over 41 times, the height of the sliding window is larger than the height of the frame; this means the scale iteration process is redundant and it will decrease the whole detection process rate. So the detection distance calculation is mainly based on the vertical parameters rather than on the horizontal parameters.

3.1.5 Determination of the Boundaries of C_s and C_L

3.1.5.1 Speed-aware Detection Scenarios

According to the data concerning the driving speed limits in Canada, in most provinces, default speed limits are 50 km/h in urban areas, 80 km/h in rural areas, and 100 km/h on grade-separated expressways [216, 217]. Default speed limits for school zones tend to be 30 or 40 km/h in urban areas and 50 km/h in rural areas [218]. So the driving speed can be divided into 5 scenarios (shown in Table 3.1). According to the scenarios and stopping distance for each driving speed, the detection process of PDS is designed for 3 cases: for low speed area, for urban and suburban areas and for expressways. When talking about highway or arterial roads, the possibility of pedestrian appearance in front of vehicles is very small, but we still need to take this into consideration.

Table 3.1: Canadian roadway velocity limits.

Driving scenario	**Kilometers/hour**	**Miles/hour**
Parking areas	Less than 15	Less than 10
School zones	30	20
Residential streets	40	25
Major urban and suburban roads	50	30
Most 2-lane highways outside cities	80	50

To make the detection cases suitable for most scenarios, the detection range for each case is devided into two sections (shown in Table 3.2): One covers the range from the Minimum Detection Distance (MinDD) to the Total Stopping Distance (TSD) at the current velocity, the other covers the range from the Total Stopping Distance (TSD) to the Maximum Detection Distance (MaxDD). The details of the detection case configuration is going to be talked about in Section 3.1.5.2.

3.1.5.2 Coordination of Two cameras

Recall that we use two unequal focal length cameras to detect different partial ranges (shown in Table 3.2). For each driving speed, the detection range of C_s spans from the MinDD to the TSD. And the detection range of C_L is from the TSD to the MaxDD. The reason we set TSD as the critical value can be explained by two justifications. On the one hand, deriving detection from the measurements obtained by MinDD to TSD, enable the system to ensure that pedestrians in this range can be detected in advance if the car is driving at a lower speed or slowing down its speed. On the other hand, based on the

Table 3.2: Detection cases of PDS.

Different cases	**Detection range of C_s**	**Detection range of C_L**
Case 1:in low speed area (less than 30kph)	$MinDD_{30} - -TSD_{30}$	$TSD_{30} - -MaxDD_{30}$
Case 2:in urban and suburban areas (around 50kph)	$MinDD_{50} - -TSD_{50}$	$TSD_{50} - -MaxDD_{50}$
Case 3:on highways and expressways (around 80kph)	$MinDD_{80} - -TSD_{80}$	$TSD_{80} - -MaxDD_{80}$

stopping distance talked about in Section 3.1.5.1, pedestrians in the detection range of C_L can be detected in advance if the car is accelerating. Meanwhile, the extended detection range can cover the range which is the stopping distance achieved by a car at a velocity 10 to 20kph higher than the current velocity. According to Equations (3.10) and (3.11), the calculation of MinDD and MaxDD is based on the focal length and vertical FOV. Then the focal length and the vertical FOV in each case is based on the TSD at different speed.

From Equations (3.6), (3.9) and (3.12), the formula for calculating the vertical field of view of the detection distance belonging to each camera can be obtained:

$$\theta_v = 2 \times \arctan(\frac{H_p \times h}{2 \times D_s \times h_p}) \tag{3.16}$$

or

$$\theta_v = 2 \times \arctan(\frac{H_c}{D_r}) \tag{3.17}$$

The selection of the vertical field of view θ_v calculation is determined by which partial detection range is calculated. If the calculation is for C_L, both Equation (3.17) and Equation (3.16) should be used to get the maximum θ_v. And, Equation (3.16) is used for C_S calculation. The focal length of each camera is then:

$$f = \frac{d}{2 \times \tan(\frac{\theta_v}{2})} \tag{3.18}$$

Finally, based on the stopping distance of each driving speed and the formulas above, we can calculate the vertical FOV and focal length for both cameras in different cases.

Calculation for C_s : the detection range of C_s spans from MinDD to TSD (shown in Figure 3.17); so, TSD is regarded as the maximum detection distance of C_s. According to Equations (3.16) and (3.18), the vertical field of view and the focal length can be obtained:

$$\theta_{v-C_s} = 2 \times \arctan(\frac{H_p \times h}{2 \times TSD \times h_{p-min}}) \tag{3.19}$$

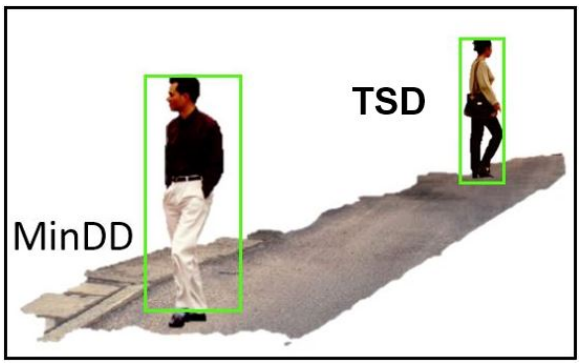


Figure 3.17: Detection range of C_s — span from MinDD to TSD.

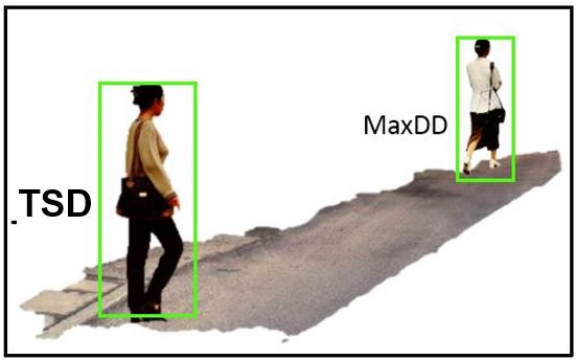


Figure 3.18: Detection Range of C_L — span from TSD to MaxDD.

$$f_{C_s} = \frac{d_v}{2 \times \tan(\frac{\theta_{v-C_s}}{2})} \tag{3.20}$$

Where TSD is the stopping distance. From Equations (3.11) and (3.12), the MinDD of C_s is:

$$MinDD = Max\left\{\frac{H_c}{\tan(\frac{\theta_{v-C_s}}{2})}, \frac{H_p \times h}{2 \times \tan(\frac{\theta_{v-C_s}}{2}) \times h_{p-max}}\right\} \tag{3.21}$$

Calculation for C_L : the minimum detection range of C_L is set from TSD to MaxDD (shown in Figure 3.18); and, it is shown in Equations (3.17), (3.16) and (3.18), that the vertical FOV of C_L is determined by the minimum required detection distance D_r and the minimum theoretical detection distance D_{min}:

$$\theta_{v-C_L} = Max\left\{2 \times \arctan(\frac{H_p \times h}{2 \times TSD \times h_{p-max}}), 2 \times \arctan(\frac{H_c}{TSD})\right\} \tag{3.22}$$

The calculation of focal length of the C_L is then the same as Equation (3.18):

$$f_{C_L} = \frac{d_v}{2 \times \tan(\frac{\theta_{v-C_L}}{2})} \tag{3.23}$$

From Equation (3.10), the MaxDD of C_L is:

$$MaxDD = \frac{H_p \times h}{2 \times \tan(\frac{\theta_{v-C_L}}{2}) \times h_{p-min}} \tag{3.24}$$

3.2 Optimization of the Detection Results

3.2.1 Detection Results Synchronization

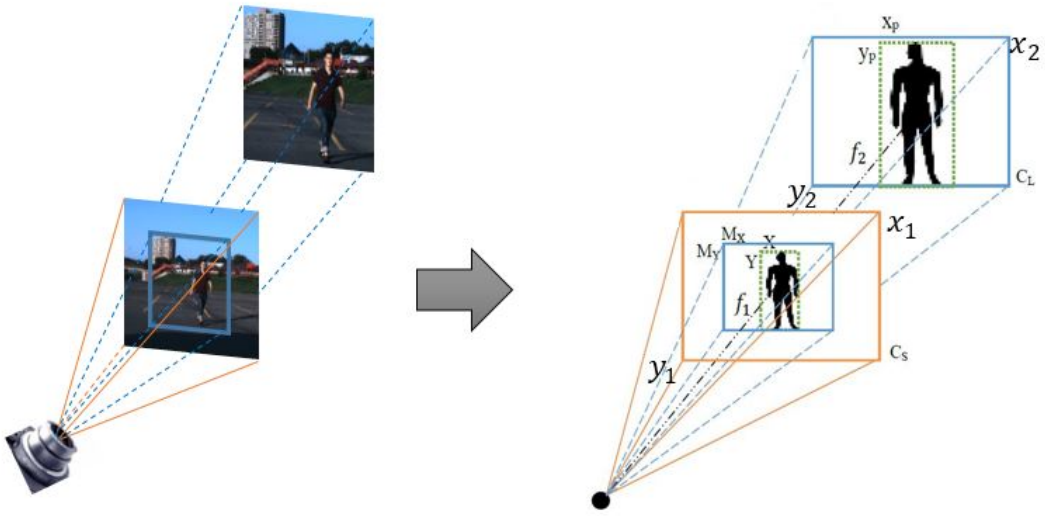


Figure 3.19: An example of C_s and C_L synchronization — f_1 is the focal length of C_s; f_2 is the focal length of C_L; x_1 is the total horizontal pixel of C_s; y_1 is the total vertical pixel C_s; x_2 is the total horizontal pixel of C_L; y_2 is the total vertical pixel C_L; x_p is the abscissa of one of the results in C_L; y_p is the ordinate of one of the results in C_L; X is the abscissa of x_p in C_s; Y is the ordinate of y_p in C_s; M_x is the horizontal shift offset of C_L in C_s; M_y is the vertical shift offset of C_L in C_s.

As two cameras are used in this system, the images captured by different cameras are not of equal focal length (shown in Figure 3.19). The pedestrians which can be detected in the C_L cannot be detected in the C_s images, and vice versa. So we need to combine the detection results together, which means the synchronization method is required to

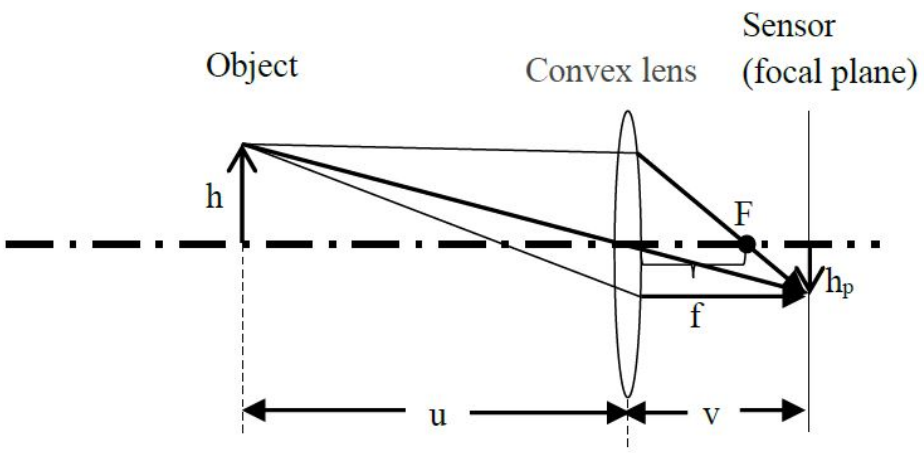


Figure 3.20: Amplification and focal length — u is the object distance; v is the image distance; F is the focal point; f is the focal length; h is the real height of pedestrian; h_p is the height of pedestrian in picture.

ensure that the system keeps operating correctly. The image captured by C_s has a shorter focal length than the image captured by C_L. The detection results of C_L and C_s can be synchronized appropriately, based on the relationship between the different focal length.

The relationship can be explained like this: the image taken by C_L can be seen as an amplification of the center of the image of C_s. So the detection results from C_L can be shrunk to the appropriate size and can be synchronized in the picture of C_s. Based on the basic physical principles of the amplification theorem [219], shown in Figure 3.20, the relationship between amplification and focal length can be acquired:

$$\frac{1}{u} + \frac{1}{v} = \frac{1}{f} \tag{3.25}$$

$$\frac{h}{h_p} = \frac{u}{v} \tag{3.26}$$

Where the definition of the notations are explained in Figure 3.20.

From Equations (3.25) and (3.26), we get:

$$\frac{h}{h_p} = \frac{u}{f} - 1 \tag{3.27}$$

In the calculation, the ratio of $\frac{h}{h_p}$ and $\frac{u}{f}$ is much larger than the number 1. When the h and u change, number '1' gives little influence. The number '1' can then be elided. If h and u is fixed, h_p will change with the changing of focal length f. Then the relationship between images of different focal length (C_s and C_L) is the ratio of short and long focal length. To perform the amplification synchronization, the equation is:

- Horizontal Offset

$$M_x = \frac{x_1}{2} - \frac{f_1 \times x_2}{2 \times f_2} \tag{3.28}$$

- Vertical Offset

$$M_y = \frac{y_1}{2} - \frac{f_1 \times y_2}{2 \times f_2} \tag{3.29}$$

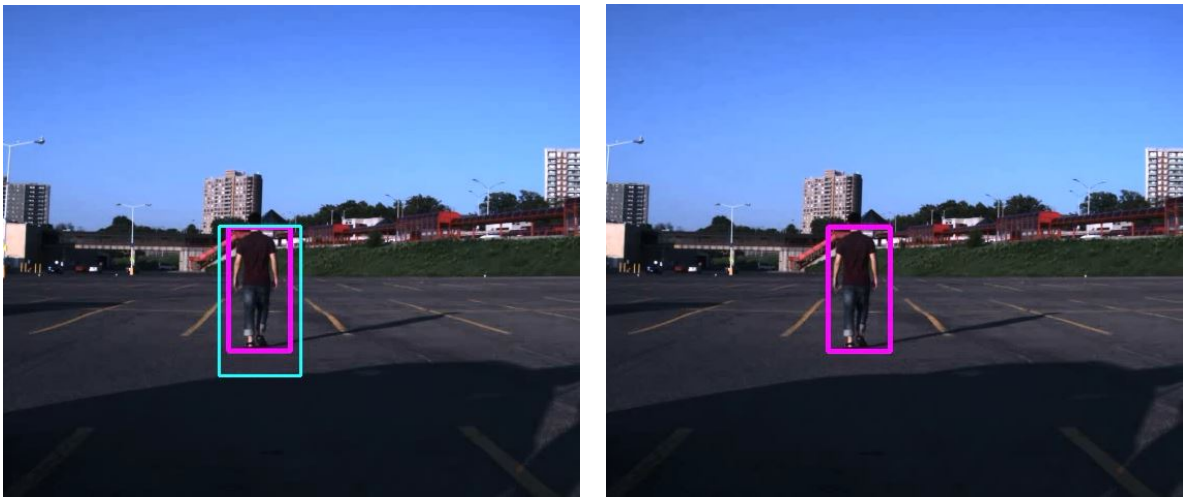
After that, based on Equations (3.28) and (3.29), the synchronization results of C_L in the position of image C_s is:

$$X = M_x + \frac{f_1 \times x_p}{f_2} \tag{3.30}$$

$$Y = M_y + \frac{f_1 \times y_p}{f_2} \tag{3.31}$$

After the synchronization of detection results of both cameras, the synchronized results are processed in the duplicate results elimination process. In the elimination process, the repeated detection results are eliminated. By doing this, the detection accuracy of the two-camera-based detection structure can be kept in a high level.

3.2.2 Duplicate Results Elimination



(a) Results without elimination (b) Results with elimination

Figure 3.21: Examples of results with and without elimination – the pedestrian in in the distance around TSD are detected by both C_s and C_L, the 3.21a is the result without elimination and the 3.21b is the result with elimination.

Normally, in the PDS, C_s detects the range from MinDD to TSD and C_L detects the range from TSD to MaxDD. There should be no overlapping detection range between them. However, in our experiment, when the PDS undergoes normal operation, there are chances that the pedestrian, who is standing at the distance of around TSD from the car, can be detected by both C_s and C_L, as the example shown in Figure 3.21. These duplicate detection results will decrease the safety level. To handle this problem, we first thought about the Scale-invariant feature transform (SIFT). However, the processing speed of SIFT is so slow that it cannot be used in real-time. We then focussed on the Speeded Up Robust Features (SURF), which is used in the research of object recognition. SURF uses the features of the sum of the Haar wavelet response around the point of interest. Meanwhile, SURF can be computed extremely quickly with the aid of the integral image. We combined SURF with our detection system mainly to use it in the research area. The elimination results then revealed that SURF performs good duplication elimination.

Figure 3.22: Enlarged elimination area.

Although the SURF method performs good elimination results, the processing speed of SURF is not dependable. During the operation, the processing speed of SURF will change greatly if the image resolution or the feature size changes. So we introduced a new elimination algorithm to make an improvement on both the robustness of the processing speed and on the elimination results. The detection results from C_s have a higher priority

Require: Detection process of C_s and C_L are both completed

1: Import the enlarged elimination areas of C_s into dynamic array V_{C_s};
2: Import synchronized detection results of C_L into dynamic array V_{C_L};
3: The dynamic array V_{E-C_L} was declared to store the remaining synchronized detection results of C_L;
4: **for** i = 0 ; i < $V_{C_L}.size$; ++i **do**
5: j = 0;
6: **for** ; j < $V_{C_s}.size$; ++j **do**
7: x_{C_L} = synchronized x value of the top-left point of $V_{C_L}.i$;
8: y_{C_L} = synchronized y value of the top-left point of $V_{C_L}.i$;
9: x_{C_s} = x value of the top-left point of $V_{C_s}.j$;
10: y_{C_s} = y value of the top-left point of $V_{C_s}.j$;
11: $width_{C_L}$ = width of $V_{C_L}.i$;
12: $height_{C_L}$ = height of $V_{C_L}.i$;
13: $width_{C_s}$ = width of $V_{C_s}.j$;
14: $height_{C_s}$ = height of $V_{C_s}.j$;
15: **if** $(x_{C_L} > x_{C_s})$**and**$(y_{C_L} > y_{C_s})$**and**$(x_{C_L} + width_{C_L} < x_{C_s} + width_{C_s})$**and**$(y_{C_L} + height_{C_L} < y_{C_s} + height_{C_s})$ **then**
16: break;
17: **else**
18: continue;
19: **end if**
20: **end for**
21: **if** j = $V_{C_s}.size$ **then**
22: $V_{E-C_L}.pushback(V_{C_L}.i)$;
23: **end if**
24: **end for**

Figure 3.23: Algorithm of duplicate results elimination.

than the ones from C_L. An enlarged elimination area is set up based on the size of the detected pedestrian plus the half-size of the trained feature size, as shown in Figure 3.22. If the detection results from C_L are in the enlarged elimination area of C_s detection results, they will be eliminated to avoid duplicate or inaccurate results. When the detection results are extracted and synchronized, the system will give a comparison procedure. This will be done to confirm whether there are results from the C_L which are repeated in the results of C_s. If this is the case, the system will only keep the result from C_s. If the results represent two different people, both of them will be kept in the detection results. The elimination algorithm is described in Figure 3.23; and, the analysis of elimination results is discussed in Section 4.2.5.

The duplication elimination process consist of three procedures. Each procedure is responsible for its own assignment. In the first procedure, after C_s and C_L finishing detection on their current frames, the detection results of C_L should be shrink according to the synchronization relationship. Then the system transfers the detection results of C_L into a dynamic array V_{C_L}, and transfers the detection results of C_s into dynamic array V_{C_s}. The vectors contains all the information of detection results, such as parameters, coordinates and positions. In the second procedure, the system compares every vector in V_{C_L} with all the V_{C_s} vectors, to find if there are detection results indicating the same pedestrian. If there are results indicating the same person, the results from C_L will be eliminated and the other one will be kept. Then the system process the same procedure for every vector in V_{C_L}. After finishing the comparison procedure, most duplicate detection results can be eliminated. The results after the elimination process will be then transmitted to the PTS, which is the third procedure.

3.3 Pedestrian Tracking System (PTS)

For a completed PPS, pedestrian detection processes need to be followed by pedestrian tracking to predict the possibility of collision in advance. In our PPS, the pedestrian detection is not the only required process; the prediction of pedestrians is also a primary requirement. The prediction of pedestrians is achieved by pedestrians tracking process to analyze their dynamics and behaviors. In the tracking process, we mainly use the Kalman filter (also known as Linear Quadratic Estimation) to operate the tracking procedure. The Kalman filter can predict the pedestrian position in frame $t+1$, based on the pedestrian detection results in frame $t-1$ and frame t. We then compared the detection result of the frame $t+1$ with the predicted results, to give more accurate results. With detection results and prediction results, an accurate movement trajectory of each pedestrian can

be obtained (as the exapmle shown in Figure 3.24). Based on the temporal movement trajectory of one pedestrian, the movement direction and the pedestrian's motion can be predicted; these are essential for our PPS to avoid collisions. Based on the prediction of pedestrian motion and behavior, the PPS can alert the drivers if a warning situation is coming or a dangerous situation is in front of them.

(a) Pedestrian is under tracking process (b) The other side of the under-trancked pedestrian

Figure 3.24: An example of pedestrian tracking.

3.3.1 Tracking Algorithms of the Pedestrian Tracking System

When the whole system is operating, four dynamic vectors are declared; two of them are used for the detection results (mentioned above as V_{C_L} and V_{C_s}) and the other two dynamic vectors, named V_{T-C_s} and V_{T-C_L}, are used for the tracking results. Each detection result is placed in a correlating dynamic vector according to which camera it belongs. The tracking process then takes the detection results from the detection vectors and puts them into the tracking vectors via the Kalman filter algorithm. After two or three steps, the tracking process can provide accurate predictions on the motion of pedestrians. The final results based on the tracking and detection systems are then transmitted to the in-car monitor to inform the driver of the situation in front of the vehicle.

3.3.1.1 Kalman Filter

The Kalman filter algorithm consists of a two-step process: the prediction step and the update step. In the prediction step, at time 1, the current state variables are estimated by the Kalman filter, along with the uncertainties regarding the current state variables.

Once the outcome of the measurement at time 2 is observed, the update step begins to work. These estimates at time 1 are updated using a weighted average. The estimates with the higher level of certainty are given more weight. The prediction step will then give the prediction again for the state variable at time 3; this is followed by the update step. When the two-step process begins to iterate, the Kalman filter tracking and predicting function shows up. Due to the inherent recursive ability of the algorithm, it can run in real-time by using the measurements at current time as the input, its previously priori state and its uncertainty matrix. The algorithms of the Kalman filter are described in the following text.

A basic model of the Kalman filter [190][220] The true state of a model at time k evolves from the state of $(k-1)$, as the equation below demonstrates:

$$x_k = F_k x_{k-1} + B_k u_k + W_k \tag{3.32}$$

Where F_k is the state transition model which is applied to the previous state x_{k1}; B_k is the control-input model which is applied to the control vector u_k; W_k is the process noise which is assumed to be drawn from a zero mean multivariate normal distribution with a covariance of Q_k. That is:

$$W_k \sim N(0, Q_k)$$

At time k, the measurement z_k of the true state x_k is made according to

$$z_k = H_k x_{k-1} + v_k \tag{3.33}$$

where H_k is the observation model. The true state space is mapped through H_k into the observed space and v_k is the observation noise assumed to be a zero mean Gaussian white noise with a covariance of R_k. That is

$$v_k \sim N(0, R_k)$$

The initial state and the noise vectors at each step are all assumed to be mutually independent.

Details of Kalman filter [191] Recall that there are two steps in the Kalman filter. One is the act of prediction and the other is the process of updating. In the prediction step, the Kalman filter uses the state variables at time k-1 to predict the state at time k. In the update step, the Kalman filter utilizes the measurement of the state at time k to update the prediction of the state at time k; this is done to acquire an accurate state value.

Prediction:

Predicted (a priori) state estimate:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k \tag{3.34}$$

Predicted (a priori) estimate covariance:

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k \tag{3.35}$$

Update:

Innovation or measurement residual:

$$\tilde{y}_k = z_k - H_k \hat{x}_{k|k-1} \tag{3.36}$$

Innovation (or residual) covariance:

$$\mathbf{S}_k = H_k P_{k|k-1} H_k^T + R_k \tag{3.37}$$

Optimal Kalman gain:

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{3.38}$$

Updated (a posteriori) state estimate:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \tilde{y}_k \tag{3.39}$$

Updated (a posteriori) estimate covariance:

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{3.40}$$

The formula for the updated estimate and covariance above is only valid for the optimal Kalman gain.

3.3.1.2 Tracking and Predicting

In the PTS, the methodology used to improve pedestrian tracking accuracy and to smoothen tracking results is formalized in a process. Based on the Opencv image processing library, we made a C++ class (called PedestrianTracking class — short for PTC) which mainly works on the pedestrian tracking process. We put all the work related to pedestrian tracking into the PTC to separate the pedestrian detection process and the pedestrian tracking process. The correlation between these two processes is the rectangle coordinate of the pedestrians. This rectangle is the one drawn around the detected pedestrians, like the one shown in Figure 3.24. All the parameters related to this rectangle are utilized by our PTC. When the PPS is working, the rectangles of the detection results are transmitted to the PTC. The PTC will operate the Kalman filter processing on the detection results. Finally, the optimized tracking results are transmitted back to the system to judge if there is a danger or a warning situation in front of the vehicle (the process of the PTS is shown in Figure 3.25).

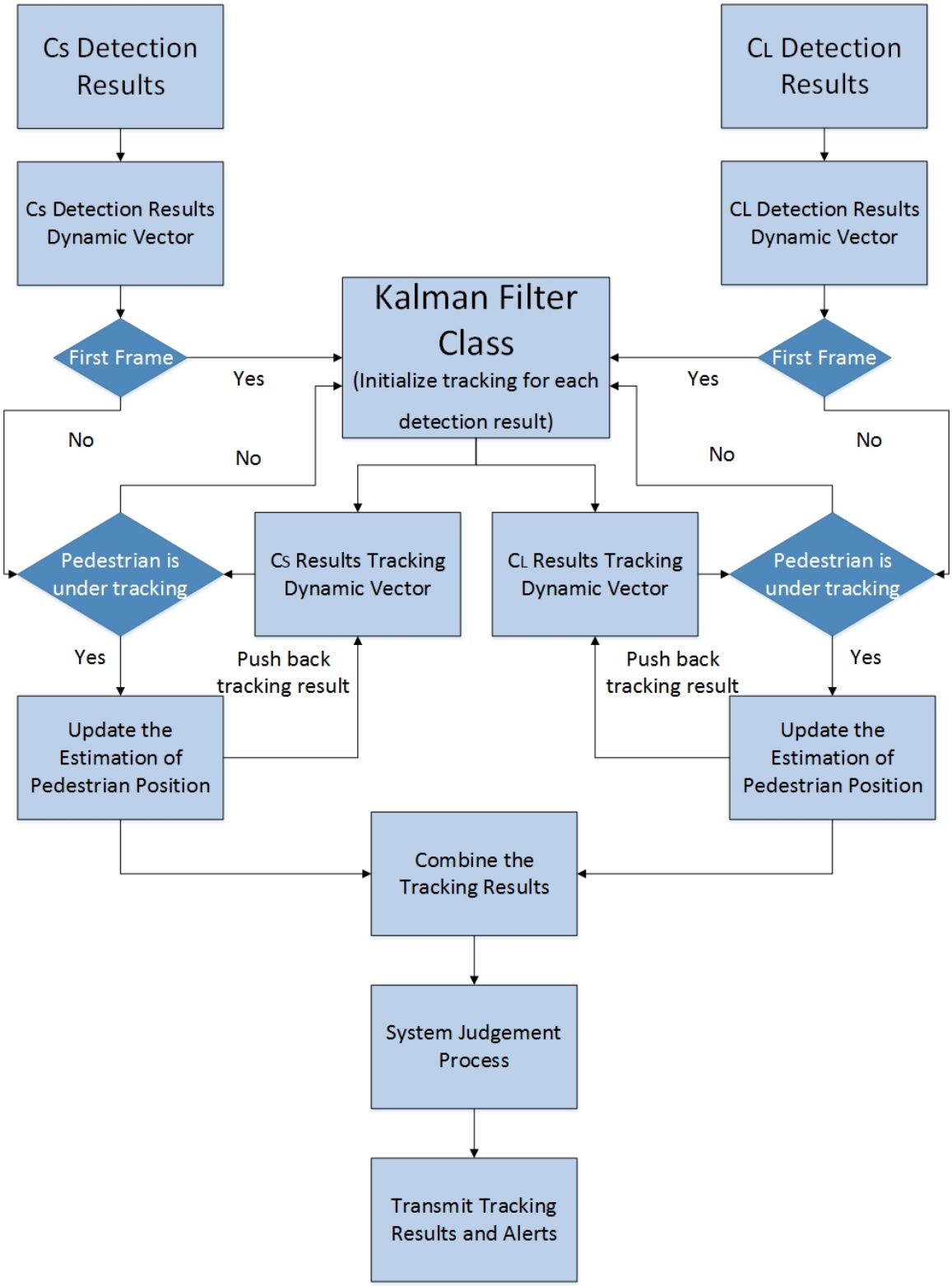


Figure 3.25: Pedestrian tracking process.

Detection-to-Tracking When a pedestrian is detected by our system, a rectangle is generated to memorize the position of the detected pedestrian. There are several parameters used to describe the rectangle. The parameters include the coordinates of the top-left point of the rectangle in the image. Also, the parameters have the description of the width and height of the rectangle. So we regard the abscissa and ordinate of the top-left point and the width and height of the rectangle as the measurements of the Kalman filter. These four parameters of each detection results are then transmitted to the PTC so that the tracking process can be proceeded.

PedestrianTracking Class In the PTC, we initialize the Kalman filter by using the input of the detection results. In the beginning, several matrices were generated. We have 8 state variables and 4 measurement variables. So the transition matrix F_k is set as:

$$F_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The state variables and measurement variables are set as:

$$state_{variables} = \begin{bmatrix} width \\ height \\ position.x \\ position.y \\ delta\_width \\ dealta\_height \\ delta\_position.x \\ delta\_position.y \end{bmatrix}$$

$$measurement_{variables} = \begin{bmatrix} delta\_width \\ dealta\_height \\ delta\_position.x \\ delta\_position.y \end{bmatrix}$$

In the beginning, we set the measurement matrix as the identity matrix, the process noise covariance matrix and the measurement noise covariance matrix as the diagonal matrix

with the value of e^{-1}. When the detection results are transmitted into the class, the PTC will compare the detection results with the results in tracking processes. If a corresponding tacking process is found for one detection result in the PTC, PTC will transmit this new detection result into the existing tracking process to update the state variables. Otherwise, the PTC will make a new tracking process for the new detection result; and, it will redo the initializing step in order to operate tracking process. The Kalman filter will proceed with the tracking processes on all the detection results. As the system is operating, if one of the tracking process has not been updated for several frames, this tracking process will be deleted; and, the tracking system will assume that this pedestrian has left the front area of the car.

Tracking Trajectory and Movement Prediction During the tracking process, both the detection results and the tracking results are stored in separate dynamic vectors. What we want to do with these dynamic vectors is to track pedestrians and to predict their future motion or movement direction. From the memorized tracking results in the tracking dynamic vectors, the movement trajectory of each pedestrian can be obtained. Based on the movement trajectory of each pedestrian, the approximate movement direction of each pedestrian can be estimated. Sequentially, the Kalman filter will provide the prediction of the pedestrian future motion and the movement direction in the next frame. When the prediction has been undergone, the system will provide judgements based on the tracking results, through the collision prevention process. These judgements will then be transmitted to the in-car monitor to show the driver if there is dangerous or warning situation in front area.

3.3.2 Collision Prevention Process

The Collision Prevention Process (CPP) is also embedded in the PedestrianTracking class. It is used to assist the system in judging situations, based on the current detection and tracking results. This CPP assists in the alerting of drivers to the possibility of upcoming warnings or dangerous situations. There are two different kinds of alerts. One is the Warning Alert which stands for less emergent situations. The other one is the Danger Alert which stands for the emergent situation.

Warning Alert In the tracking process, the dynamic vectors of tracking results are significant components used to perform predictions. If the pedestrian is walking towards the middle of the lane or to the car, like the example shown in Figure 3.2, this pedestrian is regarded as being in warning situation. To judge the situation of pedestrians, we make

Require: Pedestrians have been detected and tracked for more than 3 frames;

Calculate the Sub_{latest};
Calculate the Sub_{half};
for $i = 0; i < V_{T-C_s}.size; ++i$ **do**
  x_{T-C_s} = x value of the top-left point of $V_{T-C_s}.i$, $width_{T-C_s}$ = width of $V_{T-C_s}.i$;
  $height_W$ = height of the frame, $width_W$ = width of the frame;
  $latest_{CS} = V_{T-C_s}.i.Sub_{latest}$, $half_{CS} = V_{T-C_s}.i.Sub_{half}$;
  if $|latest_{CS}| < |half_{CS}|$ **then**
    label $V_{T-C_s}.i$ with warning situation;
    continue;
  else if $(x_{T-C_s} > \frac{width_W}{2} - \frac{height_W}{4})$**and** $(x_{T-C_s} + width_{T-C_s} < \frac{width_W}{2} + \frac{height_W}{4})$ **then**
    label $V_{T-C_s}.i$ with warning situation;
    continue;
  else
    continue;
  end if
end for
for $i = 0; i < V_{T-C_L}.size; ++i$ **do**
  x_{T-C_L} = x value of the top-left point of $V_{T-C_L}.i$, $width_{T-C_s}$ = width of $V_{T-C_s}.i$;
  $height_W$ = height of the frame, $width_W$ = width of the frame;
  $latest_{CL} = V_{T-C_L}.i.Sub_{latest}$, $half_{CL} = V_{T-C_L}.i.Sub_{half}$;
  if $|latest_{CL}| < |half_{CL}|$ **then**
    label $V_{T-C_L}.i$ with warning situation;
    continue;
  else if $(x_{T-C_L} > \frac{width_W}{2} - \frac{height_W}{4})$**and** $(x_{T-C_L} + width_{T-C_L} < \frac{width_W}{2} + \frac{height_W}{4})$ **then**
    label $V_{T-C_L}.i$ with warning situation;
    continue;
  else
    continue;
  end if
end for
Show the "Warning" label on pedestrians who are in warning situation;

Figure 3.26: Algorithm of Warning Alert.

full use of the coordinate parameters stored in the dynamic vector of tracking results. For the tracking results of each pedestrian, we make a subtraction between the coordinates of the latest tracking result and the center of the frame, called Sub_{latest}. Likewise, we do another subtraction between the coordinates of the tracking result in the half size of the dynamic vector and the center of the frame, called Sub_{half}. The absolute value of both subtraction results are compared to see which one is smaller. If the subtraction of Sub_{latest} is smaller, it means that this pedestrian is in the warning situation and the PPS will give Warning Alert to the driver. Otherwise, it means that this pedestrian is leaving the warning area and there will be no collision. Besides the subtraction comparison, we defined an area in the frames as the warning area. The height of the warning area is equal to the height of the frames, and the width of the warning area is equal to half the size of its height. Once a pedestrian is detected inside the warning area, he will be regarded as within the warning situation, even if he is not moving. The algorithm for the evaluation of the warning situation is exhibited in Figure 3.26.

Danger Alert When the Warning Alert is explained clearly, the danger alert can be specified easily. In the PPS, the Danger Alert means that there are highly dangerous situations in front of the car and that the driver needs to give great attention to the upcoming situations and to react immediately. The Danger Alert comes after the Warning Alert. Unlike the Warning Alert which is operated on both C_s and C_L results, the Danger Alert is mainly operated on the C_s results. If a pedestrian detected by C_s are judged as being in the warning situation, the system will determine that this pedestrian is in a dangerous situation. The algorithm is elucidated in Figure 3.27.

Require: The warning situation judgement for both V_{T-C_s} and V_{T-C_L} are finished;

1: **for** $i = 0; i < V_{T-C_s}.size; ++i$ **do**
2: **if** $V_{T-C_s}.i$ is labeled with warning situation **then**
3: label $V_{T-C_s}.i$ with danger situation;
4: **else**
5: continue;
6: **end if**
7: **end for**
8: Show the "Danger" label on pedestrians who are in danger situation;

Figure 3.27: Algorithm of Danger Alert.

Chapter 4

System Implementation and Results Analysis

In this chapter, we describe the devices used in the implementation. We then exhibit the implementation results of the PDS and the PTS. Afterwards, we analyze the results. The calculation parameters can be confirmed by the devices used. After this, the other data regarding the detection distance and distance range can be obtained easily.

4.1 Implementation Environment

4.1.1 Devices Specification

The equipment used to implement the PPS are all ordinary devices which can be bought easily. We used traditional CMOS sensor cameras with CS-Mount zoom lenses. We are going to give the detailed specification of the equipment and illustrate the equipment coordination in the following.

Cameras The specifications for cameras are illustrated in Table 4.1.

Lenses Specifications are shown in Table 4.2.

Computer Specifications for the implementing computer and the software versions are shown in Table 4.3.

Table 4.1: Camera parameters.

Camera attribute	**Camera parameters**
Company	Point Grey
Model No.	FL3-U3-13S2C-CS
Image Sensor	Sony IMX035 CMOS
Image Sensor Size	Diagonal 1/3", $4.8mm \times 3.6mm (H \times V) = 4:3$
Unit Size	3.63 μm
Maximum Resolution	1328×1048 at 120fps
Interface	USB 3.0
Image Buffer	32 MB frame buffer
Flash Memory	1MB

Table 4.2: Lens parameters.

Lens attribute	**Lens parameters**
Company	Edmund Optics
Model No.	#55-256
Focal Length (mm)	5.0 – 50.0
Maximum camera sensor format	1/3"
Aperture (f/#)	F1.3 – 16C
Field of View(°), for 1/3" Sensor	51.8 – 5.6
Working Distance (mm)	800 – ∞
Mount	CS-Mount

Table 4.3: Computer hardware.

Hardware attribute	**Hardware parameters**
Computer Model	Thiknpad T430
CPU Model	Intel Core i5-3210M @ 2.50GHz dual core
GPU Model	NVIDIA NVS-5400M 2GB
Memory	8 GB
Operating System	Windows7 Pro
OpenCV version	v2.44
Camera SDK	FlyCapture2-2.5.3.4-x64

4.1.2 Equipment Coordination

Recall that the cameras, lenses and other devices are ordinary equipments and bought separately. So they are not like some "products" sold by corporations which are compatible to most computers and can be used directly after purchased. To make the equipments cooperate mutually, some problems need to be solved.

4.1.2.1 Cameras and Lenses Calibration

The cameras and lenses can collaborate correctly; however, the issue is the scale of the focal length. We need to know how to adjust the focal length of the lens to the required parameters or what the focal length is when the lens is changing. To solve this problem, we need to perform the lens calibration. Thanks to achievements of our predecessors, we have found the calibration algorithm easily and achieved the calibration Matlab toolbox from the Phd thesis of Jean-Yves Bouguet [221]. Based on the toolbox, the cameras and lenses are coarsely calibrated and can collaborate with our PPS appropriately.

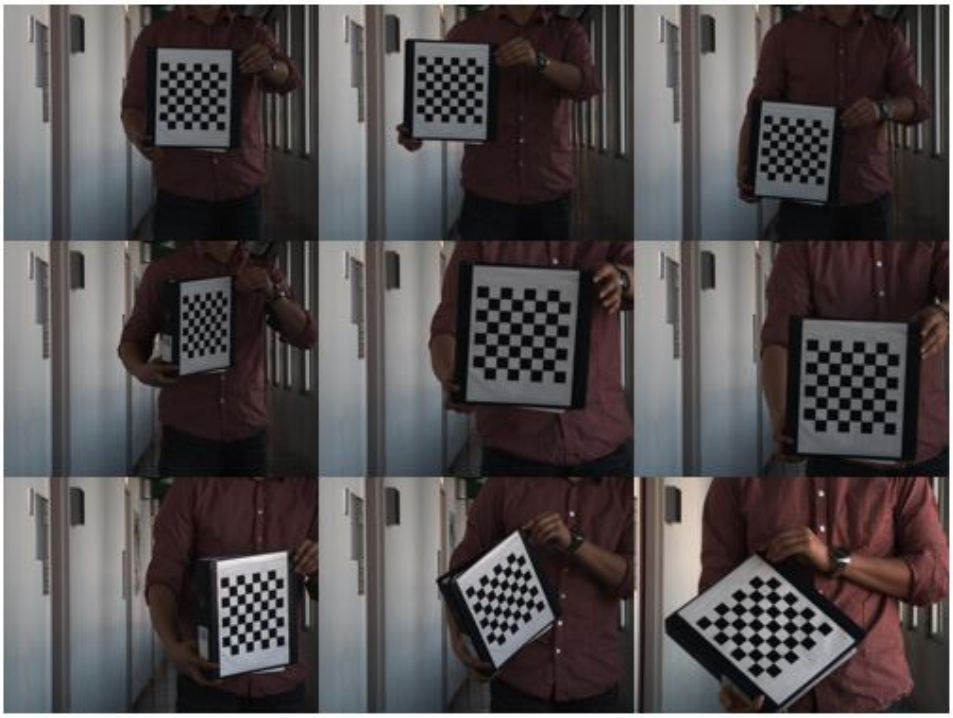


Figure 4.1: Cameras and lenses calibration.

4.1.2.2 Cameras and Computer Synchronization

The second problem is the video transmission. At first, the cameras cannot transmit the frames to the computer at a high speed through the USB3.0 interface, which means that the frames shown on the computer were lagged around 20 – 30 frames. Based on some trouble-shooting instructions found in technical documents, the specific drive for the cameras should then be installed first; this is because they are not totally compatible with the Intel USB 3.0 transport protocol. After the installation of the specific USB driver, the frame lagging decreases to an uncritical level. The photos in Figure 4.2 exhibit the equipments we use to implement the PPS.

Figure 4.2: Cameras and computer synchronization.

4.1.2.3 Image Format Converting

Recall that the main linking external image processing library used in the system is the OpenCV. The frames transmitted from the cameras are not compatible with the OpenCV library. Before performing the detection process on the frames, the frames should be converted into an OpenCV compatible format. Based on the functions in the camera's SDK and OpenCV, the format converting can be then performed efficiently and properly.

4.1.2.4 Field of View

From the lens specification in Table 4.2, the given field of view of the lens is the horizontal field of view for the 1/3" image sensor. We need to thus calculate the vertical field of

view because it is one of the significant parameters for calculating the detection range. According to Equation (3.6) and the camera specifications in Table 4.1, the vertical field of view can be calculated as follows:

$$\theta_{v-max} = 2 \times \arctan\left(\frac{d_v}{2 \times f_{min}}\right) = 39.6^{\circ} \tag{4.1}$$

$$\theta_{v-min} = 2 \times \arctan\left(\frac{d_v}{2 \times f_{max}}\right) = 4.1^{\circ} \tag{4.2}$$

From Equations (4.1) and (4.2), we obtained the following the vertical field of view of the lenses: 4° – 40°; we then can calculate the maximum and minimum detection distance and detection range for the PDS.

4.2 Implementation and Results Analysis of PDS

4.2.1 Detection Accuracy of Pedestrian Detection System

In the commencement of implementing our system, we trained 10 different sizes of HOG descriptors. As mentioned in the previous chapter, HOG descriptors affect the detection distance because the size of the HOG descriptor is one of the parameters in the system. However, the accuracy of the system is more important as it determines the detection precision and accuracy of the system. So the miss-rates of each size of HOG descriptor are required to be computed and compared with each other to determine which size is more suitable. To obtain the comparison result, 10 distinct sizes of HOG descriptors are extracted and trained. These 10 different sizes of HOG descriptors are compared together with the false positive per window (FPPW) rate.

In the implementation of miss-rate comparison, the training images are from INRIA Person database [6] and the test images are from MIT CBCL [222]. The test images from MIT are only in 64×128 pixels. We thus resized the original image dataset into ten copies – from 16×32 to 96×192 pixels. Then we test the miss-rate of all the sizes of the image dataset. In the Figure 4.3, the miss-rate of each size of HOG descriptors is drawn in the same coordinate. The interval between 10^{-4} and 10^{-3} is the critical comparing region; this is because the values of different HOG descriptors in the region larger than 10^{-3} are too close to be compared. In the interval between 10^{-4} and 10^{-3}, the HOG descriptors of 48×96 and 64×128 can provide better detection accuracy than other different sizes of HOG descriptors (the specifics are shown in Figure 4.4).

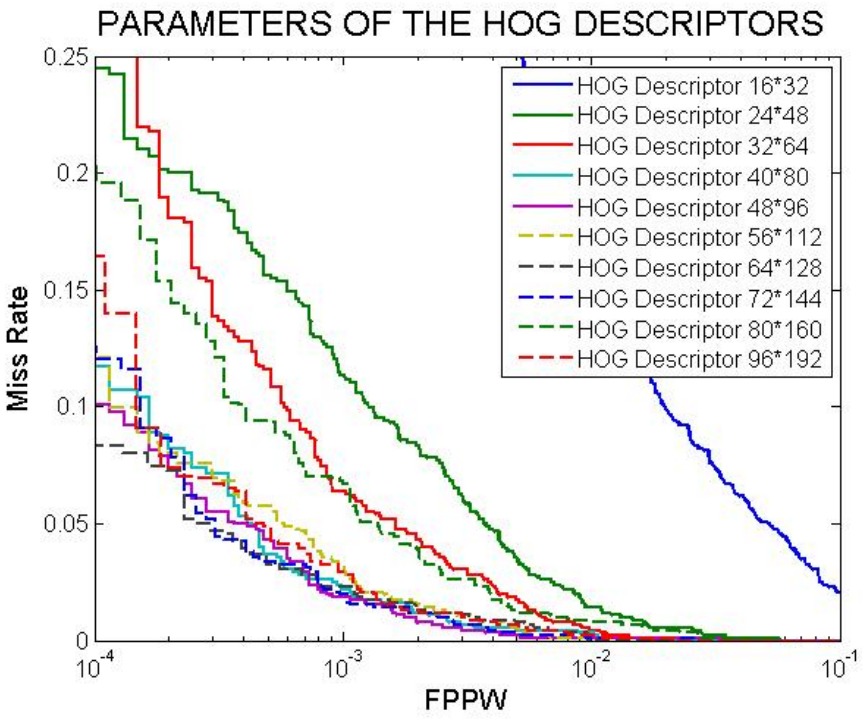


Figure 4.3: Detection miss-rate of different HOG descriptor sizes.

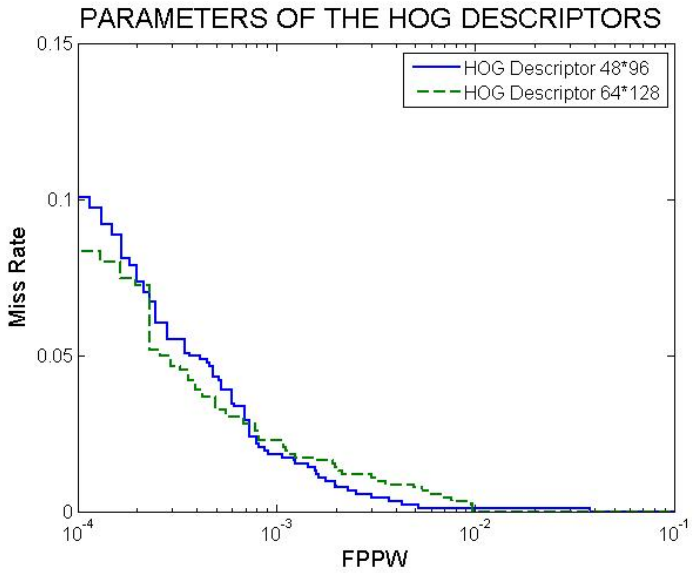


Figure 4.4: Detection miss-rate of 64×128 and 48×96 HOG descriptors.

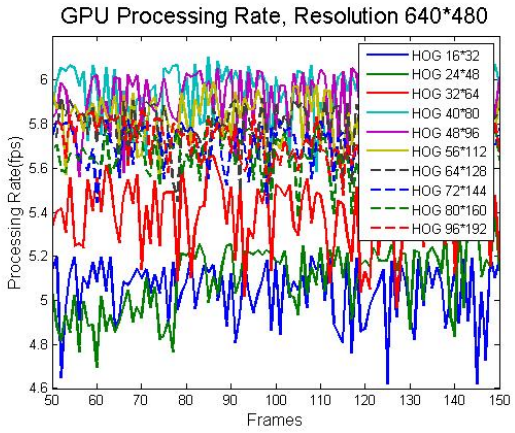
4.2.2 Processing Rate of PDS

Recall that the detection procedure of the PDS is operated by GPU in our system. So in our implementation, we tested both the processing speed of GPU-based pedestrian detection and CPU-based pedestrian detection. There are two kinds of implementation results, one is the processing rate and the other is the processing time. Processing rate represents the frames per second: the number of frames that can be processed in each second. The processing time represents the milliseconds per frame: the time required to process each frame. There are two figures used to demonstrate the disparity of processing time and processing rate implemented by CPU and GPU respectively. Because the image resolution is one of the parameters which may affect the processing rate, three different types of image resolution are testes in the implementation. And the reason for using these three different types of image resolution is going to be explained in Section 4.2.3.

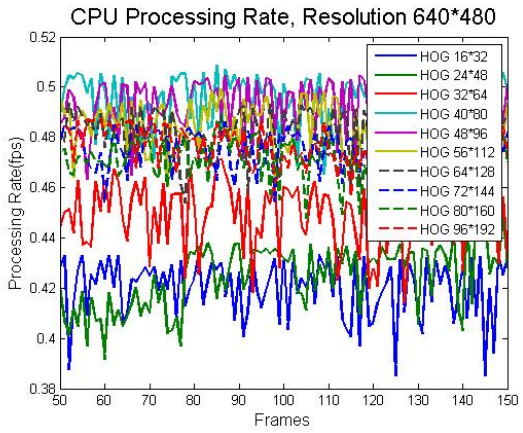
The processing rate of the implementation operated by CPU and GPU are shown in Figure 4.5. The abscissa represents the frame and the ordinate represents the processing rate (fps – frames per second) at the current frame. Figures 4.5b, 4.5d and 4.5f demonstrate the detection processing rate of different HOG descriptor sizes in CPU-based detection mode with an image resolution of 960×720, 800×600 and 640×480 respectively. Figures 4.5a, 4.5c and 4.5e elucidate the processing rate of different HOG descriptor sizes in GPU-based detection mode within distinct image resolutions. From the results in Figure 4.5, we can conclude that the detection process operated by GPU is 10 times faster than the detection process operated by CPU. As shown in the subfigures in Figure 4.5, the average processing rate declines with the increase in the image resolution. We can thus summarize that the image resolution is a significant parameter which can have a serious impact on the pedestrian detection processing rate as well, besides the CPU-based or GPU-based processing mode.

Meanwhile, the HOG descriptor size can have a small influence on the detection processing speed. In these processing rate figures, we can obtain information stating that the processing rate decreases with the increase of the HOG descriptor size. And if we take the accuracy of different HOG descriptor sizes into consideration, we can summarize that 48×96 and 64×128 HOG descriptors sizes are the two best selections as they can perform better detection accuracy and have a higher processing rate. Based on the conclusion, in the following implementation tests, we mainly use the 48×96 and 64×128 HOG descriptor sizes.

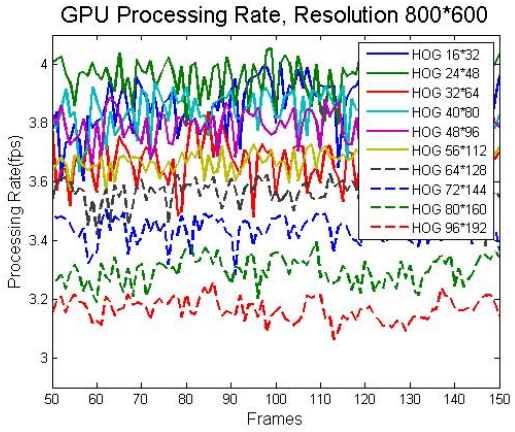
The processing time of the implementation are demonstrated in Figure 4.6. The abscissa represents each frame and the ordinate represents the processing time for current frame (millisecond per frame). The processing time graph is the supplementary instruction for the



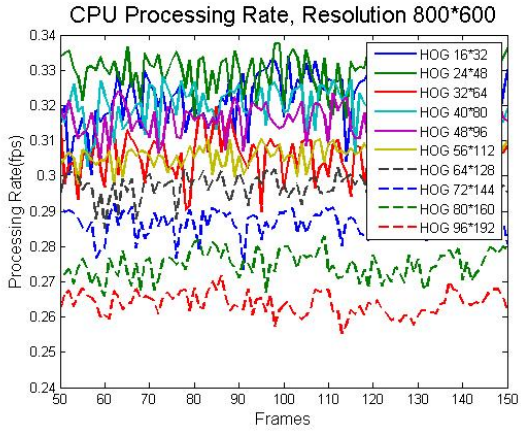
(a) Image resolution: 640 × 480 on GPU



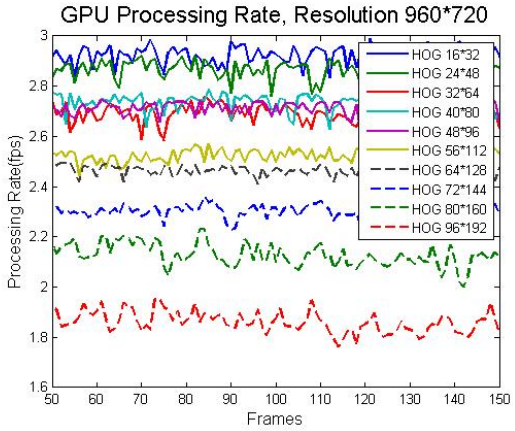
(b) Image resolution: 640 × 480 on CPU



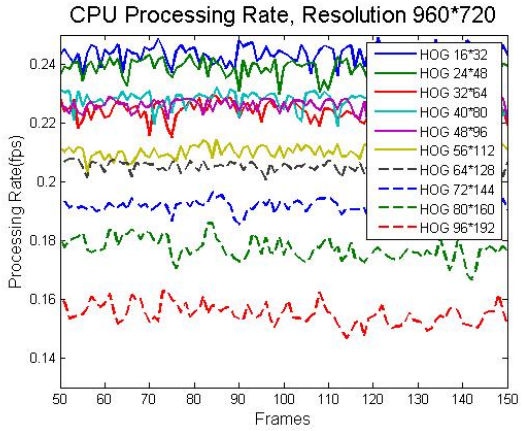
(c) Image resolution: 800 × 600 on GPU



(d) Image resolution: 800 × 600 on CPU

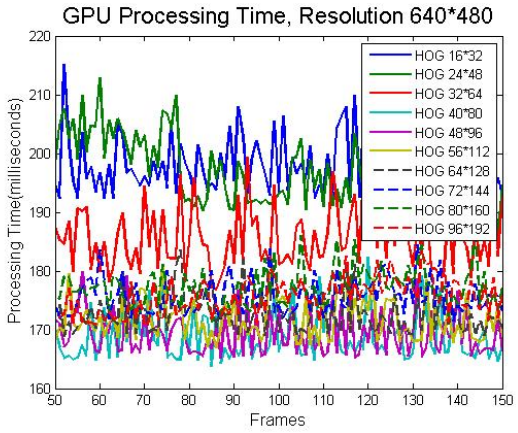


(e) Image resolution: 960 × 720 on GPU

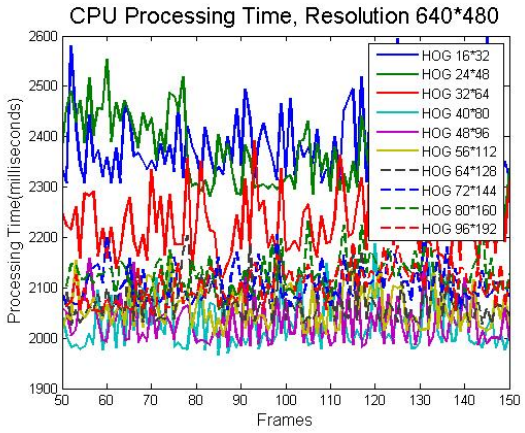


(f) Image resolution: 960 × 720 on CPU

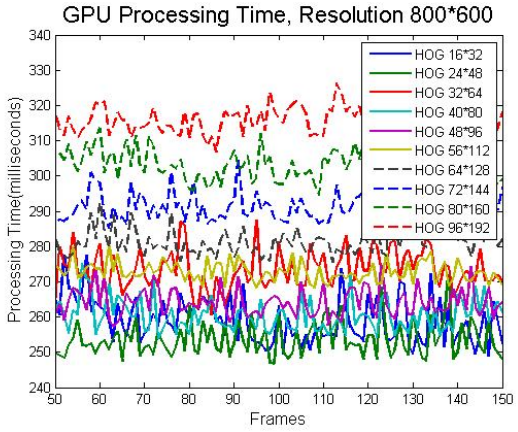
Figure 4.5: Processing rate comparison between GPU-based and CPU-based detection processes.



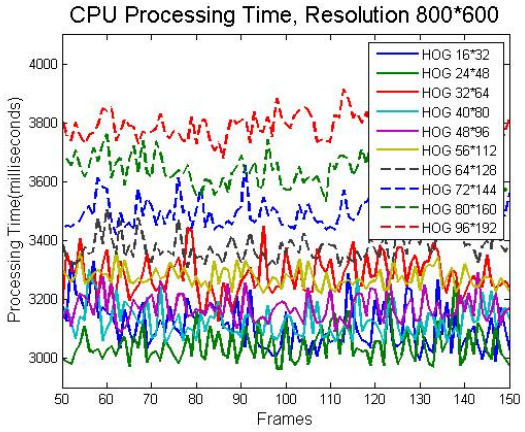
(a) Image resolution: 640 × 480 on GPU



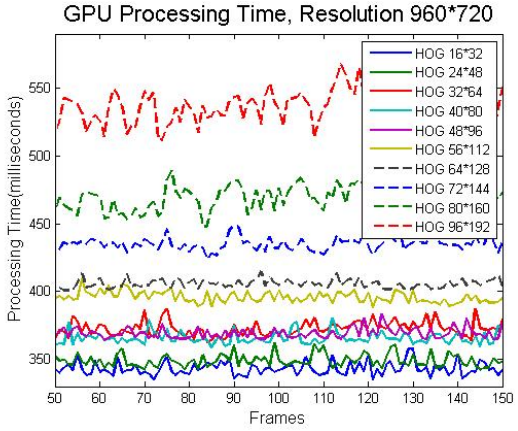
(b) Image resolution: 640 × 480 on CPU



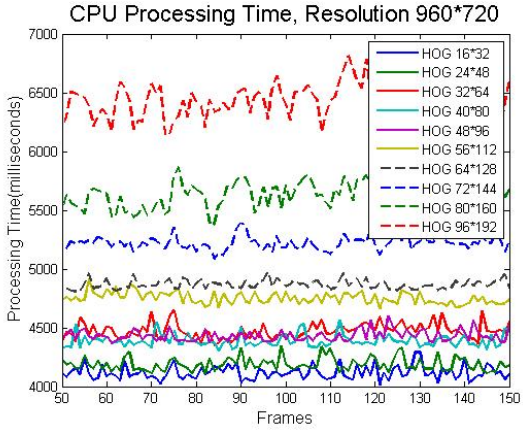
(c) Image resolution: 800 × 600 on GPU



(d) Image resolution: 800 × 600 on CPU



(e) Image resolution: 960 × 720 on GPU



(f) Image resolution: 960 × 720 on CPU

Figure 4.6: Processing time comparison between GPU-based and CPU-based detection processes.

processing rate graph; this is mainly used to describe the distinct influences resulting from the disparities of image resolution and HOG descriptor sizes. The lower value drawn on the processing time graph represents the better performance. Based on the supplementary figure (Figure 4.6), we can firmly determine that the processing mode and image resolution can have a great impact on the detection speed, as well as the HOG descriptor sizes.

4.2.3 Calculation for Effective Detection Range of PDS

When the devices are fixed, the parameters used in the calculations can be confirmed. We can then calculate the exact theoretical detection range. Based on the testing results in Section 4.2.1 and Section 4.2.2, we have determined to use 48×96 and 64×128 HOG descriptors in the following implementation. Before the implementation, we calculated the maximum and minimum detection range in different image resolutions with the 48×96 and 64×128 HOG descriptors. The calculation is based on the equations described in Section 3.1.4. The calculation results of all the combinations of image resolutions and HOG descriptors are shown in Table 4.4. An example of calculating the detection distance on an image of 960×720 image resolution with a 64×128 HOG descriptor is explained in the followings.

4.2.3.1 Detection Range in Length

According to Equations (3.10) and (3.11), the maximum and minimum detection distance is listed below (where the pedestrian height is set to 160cm):

$$D_{max} = \frac{720 \times 1.6}{2 \times \tan\left(\frac{4.1^\circ}{2}\right) \times 128} = 126 \quad meters \tag{4.3}$$

$$D_{min} = \frac{720 \times 1.6}{2 \times \tan\left(\frac{39.6^\circ}{2}\right) \times 720} = 2.2 \quad meters \tag{4.4}$$

Then we calculate the minimum required distance D_r. For most vehicles, the height of a car ranges from 1.3 to 1.5m, so H_c is set as the average height of vehicles: 1.4m. According to Equation (3.12), we can calculate the minimum required distance:

$$D_r = \frac{1.4}{\tan\left(\frac{39.6^\circ}{2}\right)} = 3.9 \quad meters \tag{4.5}$$

The minimum required distance needed to capture the ground by using the minimum focal length is 3.9m, which is bigger than D_{min}. So the detection range of the PDS in this implementation is from 3.9 to 129 meters; this means the PDS can guarantee pedestrians' safety when it is installed on vehicles and is operating normally. Besides the 960×720

image resolution and 64×128 HOG descriptor, we have calculated the maximum and minimum detection distance of other image resolutions and HOG descriptor sizes, which is shown in Table 4.4.

Table 4.4: Calculation results of detection range for different image resolutions and HOG descriptors.

Image resolution	HOG descriptor size	D_{max} meters	D_{min} meters
1280×960	64×128	168	3.9
1280×960	48×96	223	3.9
960×720	64×128	126	3.9
960×720	48×96	167	3.9
800×600	64×128	105	3.9
800×600	48×96	140	3.9
640×480	64×128	84	3.9
640×480	48×96	112	3.9
480×360	64×128	63	3.9
480×360	48×96	84	3.9
320×240	64×128	42	3.9
320×240	48×96	56	3.9

According to the Table 4.4, we can easily obtain the detection distance data for each image resolution. In this table, the image resolutions of 320×240 and 480×360 are not suitable for our objective as their maximum detection distance is too short. The image resolution 1280×960, which can reach the longest detection distance, is also not suitable for our current system; this is because the processing speed is much lower, even within the GPU-based processing mode. So the image resolutions which we are going to implement our system with are: 640×480, 800×600 and 960×720 pixels.

Sequentially, we implemented our PPS on these three different types of image resolutions to test the maximum and minimum detection distances in real world. The detection distance measurement is implemented in the Costco parking lot, where the distance can be estimated by the length of the parking space and the width of lanes. In the Costco parking lot, the longest distance is 170 meters. The maximum detection distance is the distance in which a pedestrian can be detected by the PDS. During implementation, there are some false negative results, which means that pedestrians are not detected. So the implementation results may vary among different devices and environments. The results are shown in Table 4.5.

Table 4.5: Detection range for different image resolutions and HOG descriptors in the real world.

Image resolution	HOG descriptor size	D_{max}	D_{min}
960×720	64×128	163 meters	around 4 meters
960×720	48×96	170 meters	around 4 meters
800×600	64×128	139 meters	around 4 meters
800×600	48×96	159 meters	around 4 meters
640×480	64×128	109 meters	around 4 meters
640×480	48×96	137 meters	around 4 meters

4.2.3.2 Detection Range in Width

Most lanes are less than 3.7m in width in North America [223]. The width of 3.7m attributed to lanes should be regarded as the minimum horizontal detection range needed to be covered by PDS. At the begining of the detection point, the horizontal detection range of the PDS should be larger than 3.7 meters to make system guarantee the safety for most situations. According to Equation (3.13), we can achieve the minimum distance to make the detection range in width cover the width of the car:

$$D_w = \frac{3.7}{2 \times \tan\left(\frac{51.8^\circ}{2}\right)} = 3.8 \quad meters \tag{4.6}$$

The minimum distance needed to cover the width of lane D_w is larger than the minimum detection distance D_{min}; and, it is a bit smaller than the minimum required distance – D_r. Thus, when the pedestrian detection process begins at D_r, the horizontal detection range is already bigger than the width of the lanes. This means that the width of vehicles and the width of lanes in front of vehicles are surely covered when the detection process begins at the distance of D_r. Therefore, the protection of pedestrians can be guaranteed and collisions can be prevented.

Another problem is whether the horizontal FOV is wide enough to cover the width in front of the car when it is driving at a specific velocity. If the horizontal FOV is not wide enough, the collision described by Equation (3.14) in Section 3.1.4.2 may occur. From Equation (3.14), we can derive a table which shows the relation between V_c and θ_h (shown in Table 4.6).

For most low speed driving, the speed is 30kph, which is 8.3m/s. The stopping distance of 30kph in wet conditions is 15 meters. Based on the driving speed and stopping distance, the required θ_h should be 30°. The horizontal FOV of 30° is smaller than the horizontal FOV of 45°, which the system used at 30kph (this is going to be discussed in Section 4.2.4).

Table 4.6: Focal length and velocity.

Driving velocity V_c	**Required horizontal field of view θ_h**
30 km/h	30°
50 km/h	16.5°
80 km/h	9.5°
100 km/h	7.4°

The collision can be thus prevented effectively because the system can provide detection process with large enough horizontal range to warn the drivers before pedestrians walk into the collision range (the area in front of the car).

4.2.4 Calculation for Detection Boundaries of C_s and C_L

In the previous sections, the following essential ingredients of the novel PDS have already been calculated: the maximum and minimum detection distances, horizontal detection range, focal length and field of view of the lenses. Recall that the detection operation process has three cases, and in each case, the detection range is divided into two parts: from MinDD to TSD and from TSD to MaxDD. For this implementation, a calculation example for Case 1 of the detection range for C_s and C_L is shown below, with a 960×720 resolution image and a 64×128 HOG descriptor. Furthermore, all the implementation results are shown in Table 4.7.

Calculation for C_s From Equations (3.19), (3.20) and (3.21), the parameters for the designed detection range of C_s can be obtained:

$$\theta_{v-C_s} = 2 \times \arctan\left(\frac{720 \times 1.6}{2 \times 15 \times 128}\right) = 34^\circ \tag{4.7}$$

$$f_{C_s} = \frac{3.6}{2 \times \tan\left(\frac{34^\circ}{2}\right)} = 6mm \tag{4.8}$$

$$\begin{aligned} MinDD_{30} &= Max\left\{\frac{1.4}{\tan\left(\frac{34^\circ}{2}\right)}, \frac{720 \times 1.6}{2 \times \tan\left(\frac{34^\circ}{2}\right) \times 720}\right\} \\ &= 4.6 \quad meters \end{aligned} \tag{4.9}$$

Calculation for C_L From Equations (3.22), (3.23) and (3.24), the parameters for the designed detection range of C_L can be obtained:

$$\begin{aligned} \theta_{v-C_L} &= Max\left\{2 \times \arctan\left(\frac{720 \times 1.6}{2 \times 15 \times 720}\right), 2 \times \arctan\left(\frac{1.4}{15}\right)\right\} \\ &= 11^\circ \end{aligned} \tag{4.10}$$

$$f_{C_L} = \frac{3.6}{2 \times \tan\left(\frac{11^\circ}{2}\right)} = 18.7mm \tag{4.11}$$

$$MaxDD_{30} = \frac{720 \times 1.6}{2 \times \tan\left(\frac{14.5^\circ}{2}\right) \times 128} = 46 \quad meters \tag{4.12}$$

After the detection range for C_L and C_s obtained, the focal length of lenses can be fixed and the synchronization can be configured quickly. Recall that we are testing the 960×720 resolution in this example and using s 64×128 HOG descriptor. The synchronization process is then:

- Horizontal shift offset (pixels)

$$M_x = \frac{960}{2} - \frac{6 \times 960}{2 \times 18.7} = 326 \tag{4.13}$$

- Vertical shift offset (pixels)

$$M_y = \frac{720}{2} - \frac{6 \times 720}{2 \times 18.7} = 244 \tag{4.14}$$

The synchronization results of C_L in the position of image C_s:

- the abscissa of C_L in C_s

$$X = 326 + \frac{6 \times x_p}{18.7} \tag{4.15}$$

- the ordinate of C_L in C_s

$$Y = 244 + \frac{6 \times y_p}{18.7} \tag{4.16}$$

To see all the calculation results of PDS implementation, please reference Table 4.7.

In the Table 4.7, we can obtain all the parameters needed to implement our PDS. However, the parameters for Case 3 in each combination are beyond the maximum capacity of our devices. So during the implementation, we tested the parameters in Case 1 and Case 2. The parameters in Case 3 may be tested after the update of our devices.

Table 4.7: Pedestrian detection system implementation parameters.

Image resolution	HOG descriptor size	Velocity	Speed-aware detection range		C_s & C_L Synchronization	
			C_S	C_L	X (pixels)	Y (pixels)
960 × 720	64 × 128	Case1:30kph	$\theta_{v-C_s} = 34^\circ$ $f_{C_s} = 6mm$ 4.6 to 15 meters	$\theta_{v-C_L} = 11^\circ$ $f_{C_L} = 18.7mm$ 15 to 46 meters	$X = 326 + \frac{6 \times x_p}{18.7}$	$Y = 244 + \frac{6 \times y_p}{18.7}$
		Case2:50kph	$\theta_{v-C_s} = 15^\circ$ $f_{C_s} = 14mm$ 11 to 35 meters	$\theta_{v-C_L} = 4.6^\circ$ $f_{C_L} = 45mm$ 35 to 112 meters	$X = 330 + \frac{14 \times x_p}{45}$	$Y = 248 + \frac{14 \times y_p}{45}$
		Case3:80kph	$\theta_{v-C_s} = 6.4^\circ$ $f_{C_s} = 32mm$ 25 to 80 meters	$\theta_{v-C_L} = 2^\circ$ $f_{C_L} = 103mm$ 80 to 258 meters	$X = 330 + \frac{32 \times x_p}{103}$	$Y = 248 + \frac{32 \times y_p}{103}$
	48 × 96	Case1:30kph	$\theta_{v-C_s} = 40^\circ$ $f_{C_s} = 5mm$ 3.9 to 16 meters	$\theta_{v-C_L} = 11^\circ$ $f_{C_L} = 18.7mm$ 15 to 62 meters	$X = 352 + \frac{5 \times x_p}{18.7}$	$Y = 264 + \frac{5 \times y_p}{18.7}$
		Case2:50kph	$\theta_{v-C_s} = 19^\circ$ $f_{C_s} = 11mm$ 8.4 to 35 meters	$\theta_{v-C_L} = 4.6^\circ$ $f_{C_L} = 45mm$ 35 to 149 meters	$X = 363 + \frac{11 \times x_p}{45}$	$Y = 272 + \frac{11 \times y_p}{45}$
		Case3:80kph	$\theta_{v-C_s} = 8.6^\circ$ $f_{C_s} = 24mm$ 19 to 80 meters	$\theta_{v-C_L} = 2^\circ$ $f_{C_L} = 103mm$ 80 to 340 meters	$X = 368 + \frac{24 \times x_p}{103}$	$Y = 276 + \frac{24 \times y_p}{103}$
800 × 600	64 × 128	Case1: 30kph	$\theta_{v-C_s} = 28^\circ$ $f_{C_s} = 7.2mm$ 5.6 to 15 meters	$\theta_{v-C_L} = 11^\circ$ $f_{C_L} = 18.7mm$ 15 to 39 meters	$X = 246 + \frac{7.2 \times x_p}{18.7}$	$Y = 184 + \frac{7.2 \times y_p}{18.7}$
		Case2: 50kph	$\theta_{v-C_s} = 12.3^\circ$ $f_{C_s} = 16.8mm$ 13 to 35 meters	$\theta_{v-C_L} = 4.6^\circ$ $f_{C_L} = 45mm$ 35 to 93 meters	$X = 250 + \frac{16.8 \times x_p}{45}$	$Y = 188 + \frac{16.8 \times y_p}{45}$
		Case3:80kph	$\theta_{v-C_s} = 5.3^\circ$ $f_{C_s} = 38mm$ 30 to 80 meters	$\theta_{v-C_L} = 2^\circ$ $f_{C_L} = 103mm$ 80 to 215 meters	$X = 250 + \frac{38 \times x_p}{103}$	$Y = 188 + \frac{38 \times y_p}{103}$
	48 × 96	Case1: 30kph	$\theta_{v-C_s} = 36.8^\circ$ $f_{C_s} = 5.4mm$ 4.2 to 15 meters	$\theta_{v-C_L} = 11^\circ$ $f_{C_L} = 18.7$ 15 to 52 meters	$X = 284 + \frac{5.4 \times x_p}{18.7}$	$Y = 213 + \frac{5.4 \times y_p}{18.7}$
		Case2: 50kph	$\theta_{v-C_s} = 16^\circ$ $f_{C_s} = 12.6mm$ 9.8 to 35 meters	$\theta_{v-C_L} = 4.6^\circ$ $f_{C_L} = 45mm$ 35 to 124 meters	$X = 288 + \frac{12.6 \times x_p}{45}$	$Y = 216 + \frac{12.6 \times y_p}{45}$
		Case3:80kph	$\theta_{v-C_s} = 7^\circ$ $f_{C_s} = 29mm$ 23 to 80 meters	$\theta_{v-C_L} = 2^\circ$ $f_{C_L} = 103mm$ 80 to 285 meters	$X = 287 + \frac{29 \times x_p}{103}$	$Y = 215 + \frac{29 \times y_p}{103}$
640 × 480	64 × 128	Case1: 30kph	$\theta_{v-C_s} = 23^\circ$ $f_{C_s} = 8.8mm$ 7 to 15 meters	$\theta_{v-C_L} = 11^\circ$ $f_{C_L} = 18.7mm$ 15 to 31 meters	$X = 169 + \frac{8.8 \times x_p}{18.7}$	$Y = 127 + \frac{8.8 \times y_p}{18.7}$
		Case2: 50kph	$\theta_{v-C_s} = 9.8^\circ$ $f_{C_s} = 21mm$ 16 to 35 meters	$\theta_{v-C_L} = 4.6^\circ$ $f_{C_L} = 45mm$ 35 to 75 meters	$X = 171 + \frac{21 \times x_p}{45}$	$Y = 128 + \frac{21 \times y_p}{45}$
		Case3:80kph	$\theta_{v-C_s} = 4.3^\circ$ $f_{C_s} = 48mm$ 37 to 80 meters	$\theta_{v-C_L} = 2^\circ$ $f_{C_L} = 103mm$ 80 to 172 meters	$X = 170 + \frac{48 \times x_p}{103}$	$Y = 128 + \frac{48 \times y_p}{103}$
	48 × 96	Case1: 30kph	$\theta_{v-C_s} = 30^\circ$ $f_{C_s} = 6.7mm$ 5.2 to 15 meters	$\theta_{v-C_L} = 11^\circ$ $f_{C_L} = 18.7$ 15 to 31.4 meters	$X = 205 + \frac{6.7 \times x_p}{18.7}$	$Y = 154 + \frac{6.7 \times y_p}{18.7}$
		Case2: 50kph	$\theta_{v-C_s} = 13^\circ$ $f_{C_s} = 15.8mm$ 12.3 to 35 meters	$\theta_{v-C_L} = 4.6^\circ$ $f_{C_L} = 45mm$ 35 to 100 meters	$X = 207 + \frac{15.8 \times x_p}{45}$	$Y = 155 + \frac{15.8 \times y_p}{45}$
		Case3:80kph	$\theta_{v-C_s} = 5.7^\circ$ $f_{C_s} = 36mm$ 28 to 80 meters	$\theta_{v-C_L} = 2^\circ$ $f_{C_L} = 103mm$ 80 to 229 meters	$X = 208 + \frac{48 \times x_p}{103}$	$Y = 156 + \frac{48 \times y_p}{103}$

Detection Range Implementation in the Real World To implement the calculation of the detection range of PDS from Section 4.2.4, we conducted many experiments on an open-air parking lot. To make the detection results clear and accurate, one person acted as the pedestrian walking slowly in the parking lot; and, we took videos with the two lens-equipped cameras when this person walked from a close proximity to far off, and return as well. As the parking lot has parking spaces drawn on the ground, the detection range can be estimated. When the detection results begin to appear on one point, the distance between the camera and this point may be understood as the minimum detection distance. When the detection results disappear at another point, the distance between the camera and the disappearing point is then the maximum detection distance. The detection range is the space range between the minimum detection distance and the maximum detection distance. Based on the practical test in the parking lot, we obtained the real detection range of the PDS. Again, we should emphasize that the distance is calculated by the parking place and the lane width: it is a coarse estimation of the detection range. After the calculation, we made a table to demonstrate the detection range of each case, which is shown in Table 4.8.

Table 4.8: Practical detection range of PDS.

Image resolution	HOG descriptor size	Scenario	Practical Detection Range	
			C_S	C_L
960×720	64×128	Case1:30kph	From 5 – 23 meters	From 16 – 86 meters
		Case2:50kph	From 12 – 61 meters	From 35 – 163 meters
	48×96	Case1:30kph	From 5 – 30 meters	From 16 – 96 meters
		Case2:50kph	From 9 – 41 meters	From 35 – 170 meters
800×600	64×128	Case1:30kph	From 6 – 30 meters	From 16 – 71 meters
		Case2:50kph	From 13 – 55 meters	From 35 – 139 meters
	48×96	Case1:30kph	From 5 – 24 meters	From 15 – 83 meters
		Case2:50kph	From 6 – 41 meters	From 35 – 158 meters
640×480	64×128	Case1:30kph	From 8 – 30 meters	From 16 – 60 meters
		Case2:50kph	From 21 – 67 meters	From 35 – 119 meters
	48×96	Case1:30kph	From 5 – 18 meters	From 15 – 71 meters
		Case2:50kph	From 13 – 54 meters	From 37 – 137 meters

The implementation results in Table 4.8 are organized according to Table 4.7. In Table 4.8, we can see that the practical detection range in each case is larger than the ones we calculated. The reason for these results may be due to inaccurate size of the training data and the measurement error in Google maps. Besides, the calibration of the cameras

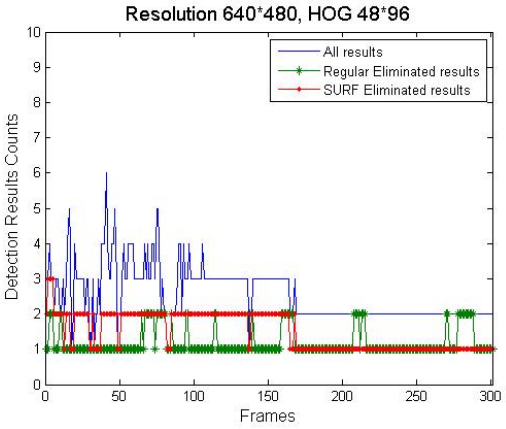
Figure 4.7: Detection results of PDS – detection results derived from C_s: a) d) and g); detection results derived from C_l: b) e) and h); detection results combination of C_s and C_L: c) f) and i).

and lenses are tuned by hand, so the calibration of the cameras and lenses may not be as accurate as we expect. As a result, due to the overlapping detection range, the duplication elimination process is extremely necessary. In Figure 4.7, we show some results related to pedestrian detection in our system through road tests.

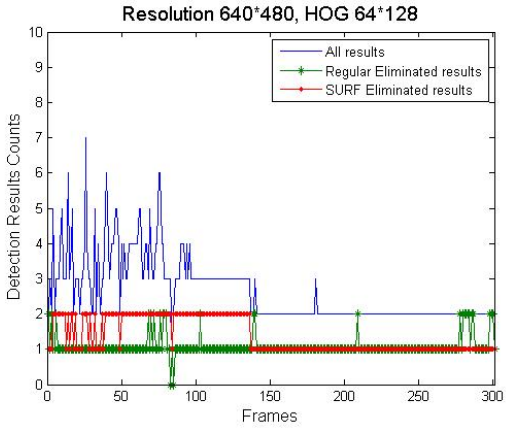
4.2.5 Elimination of duplicate detection results

Because the PDS uses multiple cameras, the detection results from different cameras may indicate the same one person. Meanwhile, the designed detection range of one camera is connected with the other one, so the detection around the connection area may result in duplicate detection results. As the implementation results of the detection range are demonstrated in Section 4.2.4, the detection range of C_s and C_L overlaps at the area of Total Stopping Distance (TSD). Due to the factors above, duplication elimination is an indispensable procedure that occurs before the final detection results are transmitted when the whole system is on.

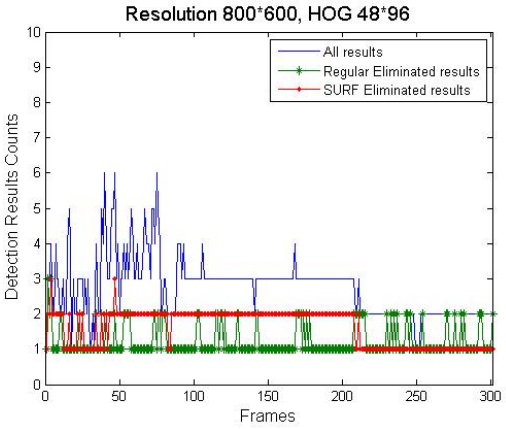
When implementing the system, we tested the duplication elimination procedure with SURF method and our own elimination method. Both of the methods can eliminate the duplicate detection results with a good performance. In Figure 4.8, the abscissa indicates the frame number and the ordinate indicates the total amount of the detection results in the frame. When the system is on, we put the threshold into a lower level to get more detection results; some of the results are even false positive results. In the duplication procedure, we then can see if the SURF or our own elimination method can provide good elimination performance. All results of these six subfigures are derived from one video and each subfigure represents the detection results from a kind of combination of image resolution and HOG descriptor. The blue curves represent the total detection results. The green curves represent the detection results after our own elimination process. The red curves represent the detection results after the SURF elimination process. The graphs demonstrate clearly that the detection process may achieve multiple detection results; and, the detection results vary greatly in the overlapping range between C_s and C_L. Normally, from 0 – 50 frames, the detection range is within the C_s detection range; and, in this range, C_L may give some inaccurate detection results if pedestrians are standing in front of the cameras directly. The inaccurate results should then be eliminated, as shown in the graphs. The range between 50 to around 160 frames (a disparity based on different HOG descriptor sizes), represents the overlapping range between C_s and C_L. The overlapping range may cause plenty of duplicate detection results and our objective is to eliminate them and to show exactly the corrected results. When the range is from around 160 – 300 frames, this range is the detection range of C_L; and, the detection of C_s will have little influence on



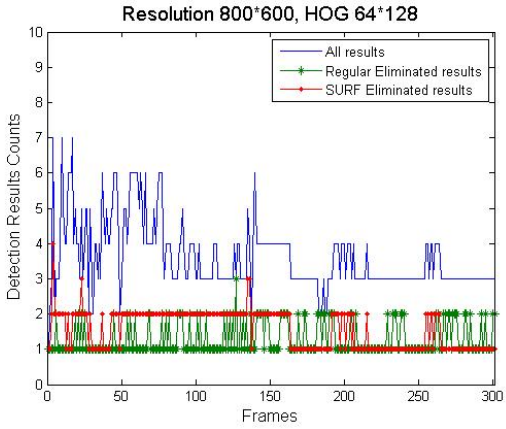
(a) Image resolution: 640 × 480, HOG descriptor: 48 × 96



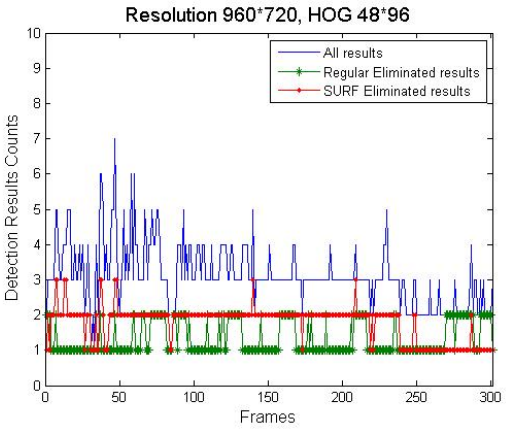
(b) Image resolution: 640 × 480, HOG descriptor: 64 × 128



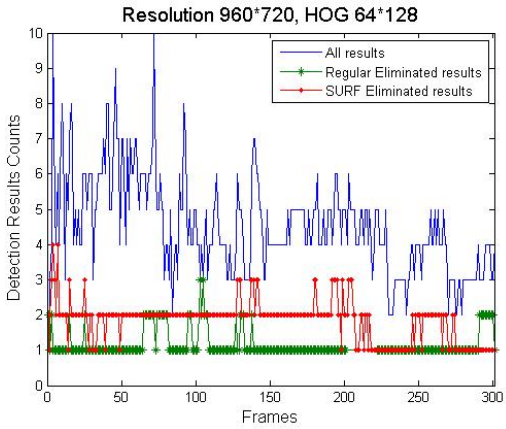
(c) Image resolution: 800 × 600, HOG descriptor: 48 × 96



(d) Image resolution: 00640 × 600, HOG descriptor: 64 × 128

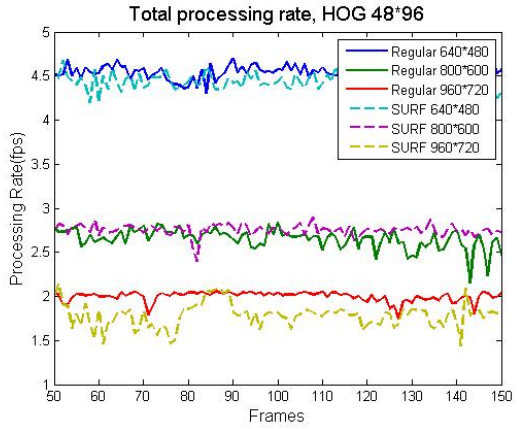


(e) Image resolution: 960 × 720, HOG descriptor: 48 × 96

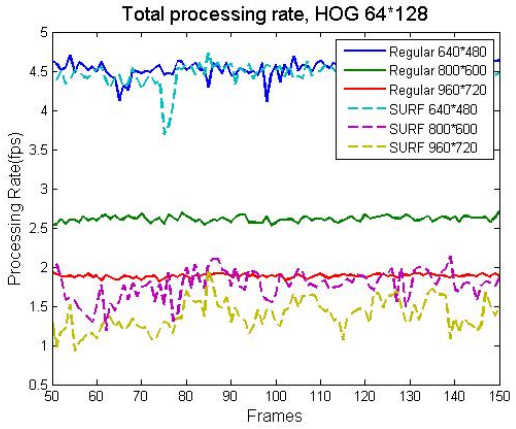


(f) Image resolution: 960 × 720, HOG descriptor: 64 × 128

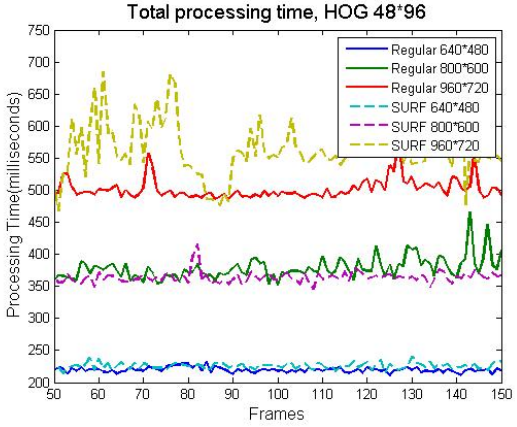
Figure 4.8: Comparison of elimination results of different elimination methods.



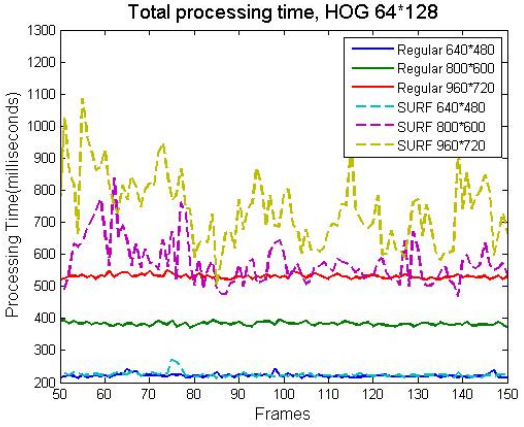
(a) HOG descriptor: 48×96



(b) HOG descriptor: 64×128



(c) HOG descriptor: 48×96



(d) HOG descriptor: 64×128

Figure 4.9: Comparison of processing rate and processing time of different elimination methods.

this range. The total detection results and eliminated detection results can then be almost the same.

After comparing the elimination results of SURF and our regular methods, we make a comparison of the total processing rate and the total processing time of our novel PDS when the SURF or our own elimination methods is in operation. This total processing rate and total processing time is not the same as the processing rate we compared in Section 4.2.2. In this section, the total processing rate and the total processing time is the rate and time of the total time cost of each frame when the system is operating. And based on the implementation results in Section 4.2.2, we used GPU-based pedestrian detection only when the elimination procedure was implemented because the processing speed of CPU-based pedestrian detection is slow. In Figure 4.9, we compared the processing speed

of SURF and our own elimination method with the same HOG descriptor size in different image resolutions. From the graphs, we can obtain that the processing speed of SURF and our own elimination method are similar in the same image resolution. However, the duplication elimination processing speed implemented by our own elimination method is more smooth than the one implemented by SURF elimination method.

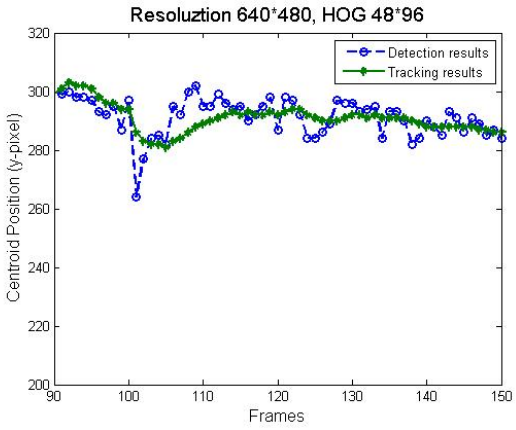
4.3 Implementation and Results Analysis of PTS

In this section, we show some implementation results to validate the tracking system. During the implementation, we compared the tracking results and detection results which are implemented in the same environment by using the same hardware. The comparison between the detection results and the tracking results clearly demonstrated that the detection accuracy was improved by using Kalman filter tracking methodology. Meanwhile, pedestrian behavior and motion can be predicted correctly by the tracking system.

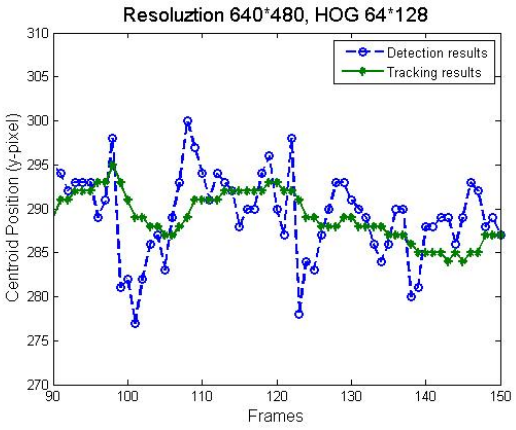
4.3.1 Tracking Process Evaluation

In the implementation of the PTS, there are two kinds of experiment results used to validate the performance of the tracking process. One is the beginning point (top-left point) of the rectangle around the pedestrian in the frame, with horizontal and vertical coordinates. The other is the area value of the pedestrian in the frame.

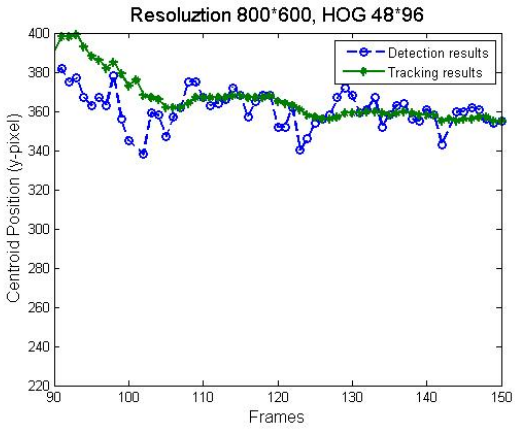
Top-left point comparison between detection results and tracking results In our experiment, there is only one pedestrian in the whole video. So the results are based on the detection results and tracking results of this pedestrian. The comparison graphs of top-left point coordinates for detection results and tracking results are exhibited in Figure 4.10. In each graph, we compared the results from the same image resolution and HOG descriptor size, to make clear the benefits of the tracking process in the whole system. The value in the graphs is the vertical coordinates of the top-left point of detection result, whose changing range is higher and can easily show the detection and tracking differences. From the figures, the changing range of tracking results and detection results can be observed. The graphs clearly demonstrate that the changing range of tracking results is smoother than the changing range of detection results. Meanwhile, the changes in the tracking results are not as steep as those in the detection results.



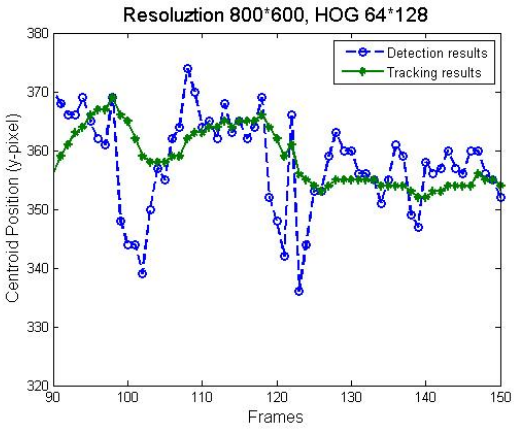
(a) Resolution: 640×480 HOG: 48×96



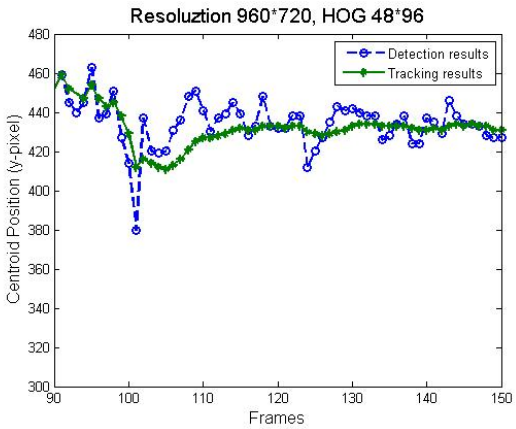
(b) Resolution: 640×480 HOG: 64×128



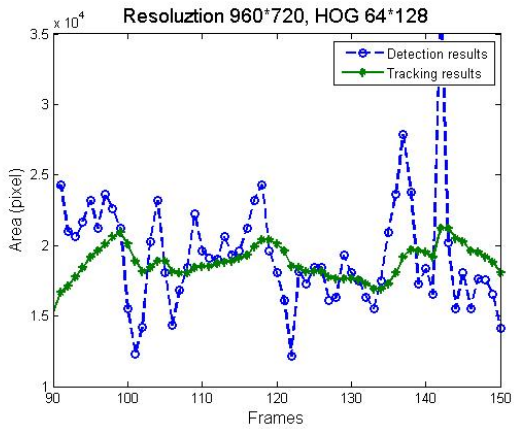
(c) Resolution: 800×600 HOG: 48×96



(d) Resolution: 800×600 HOG: 64×128

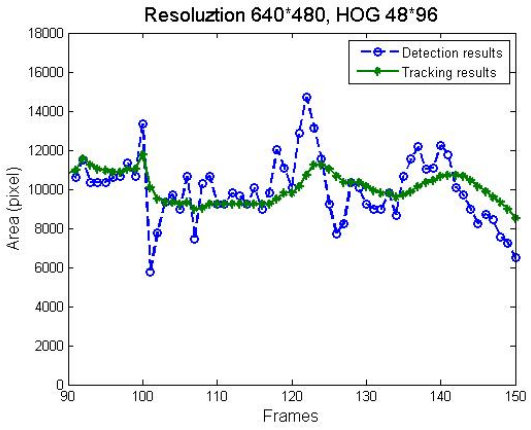


(e) Resolution: 960×720 HOG: 48×96

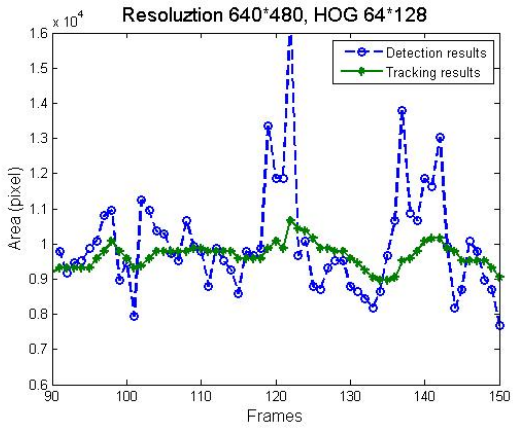


(f) Resolution: 960×720 HOG: 64×128

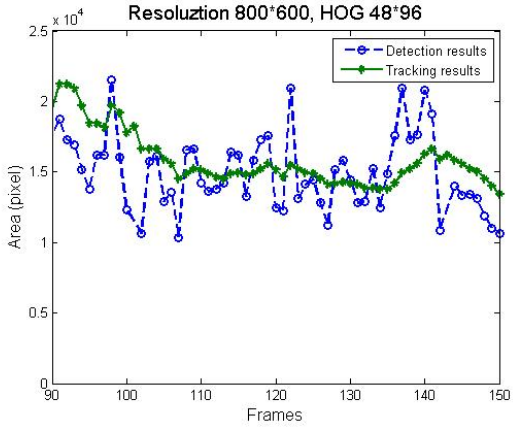
Figure 4.10: Comparison of detection and tracking results: top-left point coordinate.



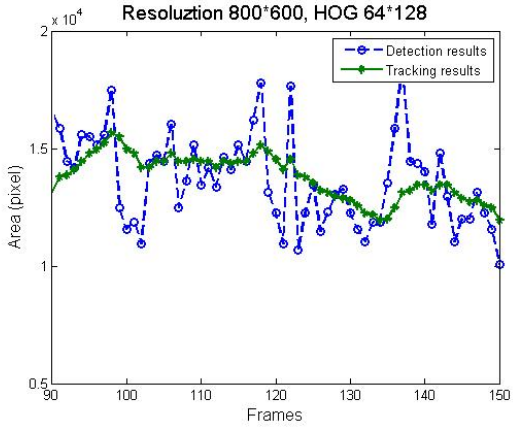
(a) Resolution: 640 × 480 HOG: 48 × 96



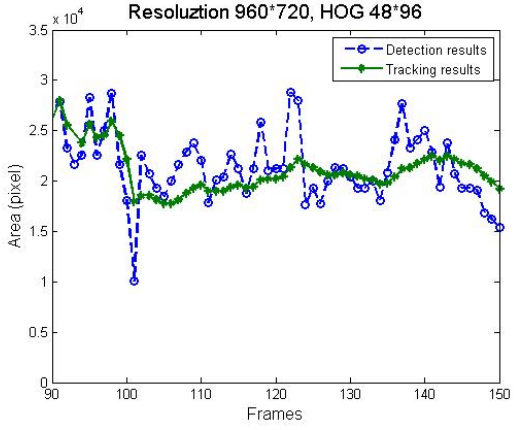
(b) Resolution: 640 × 480 HOG: 64 × 128



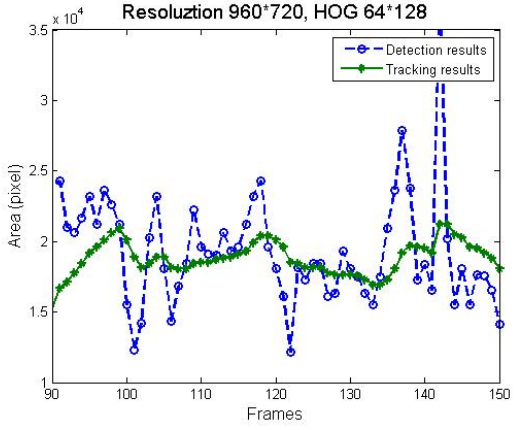
(c) Resolution: 800 × 600 HOG: 48 × 96



(d) Resolution: 800 × 600 HOG: 64 × 128



(e) Resolution: 960 × 720 HOG: 48 × 96



(f) Resolution: 960 × 720 HOG: 64 × 128

Figure 4.11: Comparison of detection and tracking results: size of detection area.

Area comparison between detection results and tracking results The area comparison between detection results and tracking results is a complementary proof used to validate the previous statement, shown in Figure 4.11. The graphs for the area changing range expanded the differences between tracking results and detection results. The area value, which is the width multiplied by the height, makes the disparity larger so that the smoothness of the changing range for tracking results is shown clearly. As a matter of fact, both the graphs for the changing range of area values and the graphs for the changing range of top-left point coordinates can be regarded as proof of the benefits of this tracking process.

4.3.2 Examples of Warning Alert and Danger Alert

The system can alert the driver if there is a danger or a situation that merits warning in the area ahead, which can greatly prevent collisions. There are some screenshot samples that show the operation of this process (shown in Figure 4.12). The movement of the pedestrian in the video can be tracked accurately and efficiently. Likewise, the motion and behavior of this pedestrian can be predicted by our PTS; the warning and danger alerts can be then shown correctly to the drivers. Drivers can then react immediately to prevent the collision occurrence, which is the most important thing we want to achieve in our system.

Figure 4.12: Alert samples of PPS.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this thesis, a novel PPS, a supplementary application to Driver Assistance Systems, was proposed to prevent collisions between vehicles and pedestrians.

In the first chapter, we described the research the background in the context of pedestrian detection areas. The problems addressed emerged based on this background. To solve these problems, we presented our motivation which seeks to extend the pedestrian detection range. By doing this, the common shortcoming of most PDSs can be solved. Based on our objective, we proposed a novel system structure; so that our objectives can be achieved.

In this thesis, three main contributions are made and described: extending the pedestrian detection range; eliminating duplicate detection results; and tracking detected pedestrians. In the PDS, a new hardware-based architecture that detects pedestrians in near-, mid- and far-scale was proposed. To ensure the detection accuracy of our system, we used HOG-SVM algorithm as the detection algorithm. Likewise, to accelerate the processing speed of the detection system, we utilized the technology from NVIDA: the parallel computing ability of the graphics card. To fully utilize the parallel computing ability of the NVIDIA graphics card, the toolkit called CUDA was required to be installed. The hardware part is based on two identical cameras, each of which is equipped with a zoom lens. We have shown the analytical and experimental possibility of accurately detecting pedestrians located up to 130 meters away from the vehicle.

To further improve our system, we focused on the elimination of redundancy between the two cameras of our system. To eliminate the duplicate detection results, a SURF-based elimination method was proposed. It can provide good elimination results and

efficient processing time. Additionally, we want to refine the elimination results and to speed up the processing time. So we proposed our own elimination method which has better elimination results than the SURF-based one with a faster processing rate.

When the PDS was specifically proposed, the tracking system was required, to refine the detection results and to predict the future motion and behaviour of detected pedestrians. The PTS is embedded in a PedestrianTracking class. All the detection results processed through the elimination procedure are transmitted into this class. The tracking system can then perform tracking processes on these detection results and can provide tracking results. By doing this, the position of pedestrians in front of vehicles can be predicted. The motion and behaviour of the pedestrians in the next moment can be also predicted. Based on the predictions, the collisions can be prevented effectively. The safety of pedestrian can be guaranteed.

According to the implementation results demonstrated in Chapter 4, the performance of our whole PPS can be validated. The safety of pedestrians can be guaranteed and collisions can be prevented effectively.

5.2 Future Work

An entire vision-based PPS is proposed in this thesis; however, there are some other functions which can help the system to enhance further performance.

1. The compatibility of the proposed novel architecture of PDS can be improved. By improving the compatibility of the structure, the structure of this novel PDS may be used as a framework by other types of vision-based applications. The detection range of other types of vision-based applications can then be extended without yielding the detection accuracy.

2. The PPS and the vehicular networks can cooperate together to improve the traffic flow [224, 225] and to decrease the traffic congestions [226, 16]. Meanwhile, the detection results can be transmitted to other nearby vehicles [227, 228, 229, 230] to warn the drivers about the upcoming situation.

References

[1] D. Geronimo, A.M. Lopez, A.D. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(7):1239–1258, July 2010.

[2] D.M. Gavrila and V. Philomin. Real-time object detection for ldquo;smart rdquo; vehicles. In *Computer Vision, The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 87–93 vol.1, 1999.

[3] T. Zhao, R. Nevatia, and B. Wu. Segmentation and tracking of multiple humans in crowded environments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(7):1198–1211, July 2008.

[4] Z. Lin and L.S. Davis. Shape-based human detection and segmentation via hierarchical part-template matching. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(4):604–618, April 2010.

[5] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1, 2001.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, volume 1, pages 886–893 vol. 1, 2005.

[7] B. Wu and R. Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *Computer Vision, Tenth IEEE International Conference on*, volume 1, pages 90–97 Vol. 1, Oct 2005.

[8] P. Sabzmeydani and G. Mori. Detecting pedestrians by learning shapelet features. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–8, June 2007.

[9] Driving School Ireland. Stopping distance for cars. http://www.drivingschoolireland.com/speed.html#5, 2013. [Online; accessed 22-May-2014].

[10] K. Pulli, A. Baksheev, K. Kornyakov, and V. Eruhimov. Real-time computer vision with opencv. *Communications of the ACM*, 55(6):61–69, June 2012.

[11] A. Boukerche and L. Bononi. Simulation and modeling of wireless, mobile, and ad hoc networks. *Mobile ad hoc networking*, pages 373–409, 2004.

[12] A. Boukerche and X. Fei. A coverage-preserving scheme for wireless sensor network with irregular sensing range. *Ad hoc networks*, 5(8):1303–1316, 2007.

[13] A. Boukerche, S.K. Das, and A. Fabbri. Swimnet: a scalable parallel simulation testbed for wireless and mobile networks. *Wireless Networks*, 7(5):467–486, 2001.

[14] R.W. Pazzi and A. Boukerche. Mobile data collector strategy for delay-sensitive applications over wireless sensor networks. *Computer Communications*, 31(5):1028–1039, 2008.

[15] A. Boukerche and Y. Ren. A secure mobile healthcare system using trust-based multicast scheme. *Selected Areas in Communications, IEEE Journal on*, 27(4):387–399, 2009.

[16] E. El Ajaltouni, A. Boukerche, and M. Zhang. An efficient dynamic load balancing scheme for distributed simulations on a grid infrastructure. In *Distributed Simulation and Real-Time Applications, 12th IEEE/ACM International Symposium on*, pages 61–68. IEEE, 2008.

[17] A. Boukerche, R.W. Pazzi, and J. Feng. An end-to-end virtual environment streaming technique for thin mobile devices over heterogeneous networks. *Computer Communications*, 31(11):2716–2725, 2008.

[18] A. Boukerche, N.J. McGraw, C. Dzermajko, and K. Lu. Grid-filtered region-based data distribution management in large-scale distributed simulation systems. In *Simulation Symposium, Proceedings. 38th Annual*, pages 259–266. IEEE, 2005.

[19] A. Boukerche and R.W.N. Pazzi. Remote rendering and streaming of progressive panoramas for mobile devices. In *Proceedings of the 14th annual ACM international conference on Multimedia*, pages 691–694. ACM, 2006.

[20] A. Boukerche, S. Hong, and T. Jacob. A distributed algorithm for dynamic channel allocation. *Mobile Networks and Applications*, 7(2):115–126, 2002.

[21] M. Elhadef, A. Boukerche, and H. Elkadiki. A distributed fault identification protocol for wireless and mobile ad hoc networks. *Journal of Parallel and Distributed Computing*, 68(3):321–335, 2008.

[22] M. Elhadef, A. Boukerche, and H. Elkadiki. Diagnosing mobile ad-hoc networks: two distributed comparison-based self-diagnosis protocols. In *Proceedings of the 4th ACM international workshop on Mobility management and wireless access*, pages 18–27. ACM, 2006.

[23] Y. Ren and A. Boukerche. Modeling and managing the trust for wireless and mobile ad hoc networks. In *Communications, IEEE International Conference on*, pages 2129–2133. IEEE, 2008.

[24] H.A.B. de Oliveira, E.F. Nakamura, A.A.F. Loureiro, and A. Boukerche. Directed position estimation: A recursive localization approach for wireless sensor networks. In *Computer Communications and Networks, Proceedings. 14th International Conference on*, pages 557–562. IEEE, 2005.

[25] B. Leng, Q. He, H. Xiao, B. Li, H. Wang, Y. Hu, W. Wu, G. Guan, H. Zou, and L. Liang. An improved pedestrians detection algorithm using hog and vibe. In *Robotics and Biomimetics, IEEE International Conference on*, pages 240–244, Dec 2013.

[26] R. Brehar and S. Nedevschi. Local information statistics of lbp and hog for pedestrian detection. In *Intelligent Computer Communication and Processing, IEEE International Conference on*, pages 117–122, Sept 2013.

[27] Y. Cao, S. Pranata, M. Yasugi, Z. Niu, and H. Nishimura. Stagged multi-scale lbp for pedestrian detection. In *Image Processing, 19th IEEE International Conference on*, pages 449–452, Sept 2012.

[28] A. Boudissa, J.K. Tan, H. Kim, and S. Ishikawa. A simple pedestrian detection using lbp-based patterns of oriented edges. In *Image Processing, 19th IEEE International Conference on*, pages 469–472, Sept 2012.

[29] W. Yao and Z. Deng. A robust pedestrian detection approach based on shapelet feature and haar detector ensembles. *Tsinghua Science and Technology*, 17(1):40–50, Feb 2012.

[30] G.R. Rakate, S.R. Borhade, P.S. Jadhav, and M.S. Shah. Advanced pedestrian detection system using combination of haar-like features, adaboost algorithm and

edgelet-shapelet. In *Computational Intelligence Computing Research, IEEE International Conference on*, pages 1–5, Dec 2012.

[31] B. Wang, Z. Chen, J. Wang, and L. Zhang. Pedestrian detection based on the combination of hog and background subtraction method. In *Transportation, Mechanical, and Electrical Engineering, International Conference on*, pages 527–531, Dec 2011.

[32] P. Kelly, N.E. O'Connor, and A.F. Smeaton. A framework for evaluating stereo-based pedestrian detection techniques. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(8):1163–1167, Aug 2008.

[33] C. Cui, Y. Liu, S. Shan, X. Chen, and W. Gao. 3d haar-like features for pedestrian detection. In *Multimedia and Expo, IEEE International Conference on*, pages 1263–1266, July 2007.

[34] D. Xia, H. Sun, and Z Shen. Real-time infrared pedestrian detection based on multi-block lbp. In *Computer Application and System Modeling, International Conference on*, volume 12, pages V12–139–V12–142, Oct 2010.

[35] H. Zou, H. Sun, and K. Ji. Real-time infrared pedestrian detection via sparse representation. In *Computer Vision in Remote Sensing, International Conference on*, pages 195–198, Dec 2012.

[36] U. Meis, M. Oberlander, and W. Ritter. Reinforcing the reliability of pedestrian detection in far-infrared sensing. In *Intelligent Vehicles Symposium, IEEE*, pages 779–783, June 2004.

[37] Y. Fang, K. Yamada, Y. Ninomiya, B. K P Horn, and I. Masaki. A shape-independent method for pedestrian detection with far-infrared images. *Vehicular Technology, IEEE Transactions on*, 53(6):1679–1697, Nov 2004.

[38] S.J. Krotosky and M.M. Trivedi. On color-, infrared-, and multimodal-stereo approaches to pedestrian detection. *Intelligent Transportation Systems, IEEE Transactions on*, 8(4):619–629, Dec 2007.

[39] R. O'Malley, M. Glavin, and E. Jones. A review of automotive infrared pedestrian detection techniques. In *Signals and Systems Conference, 208. IET Irish*, pages 168–173, June 2008.

[40] T. Kishigami, T. Morita, H. Mukai, and Y. Nakagawa. Millimeter-wave radar combining tx beamforming and rx array processing for precise pedestrians detection. In *Microwave Conference Proceedings, Asia-Pacific*, pages 1016–1018, Dec 2012.

[41] K. Kobayashi, T. Morita, H. Mukai, T. Kishigami, and Y. Nakagawa. Advanced code sequences design for pedestrian detection in millimeter-wave pulse-compression radar system. In *Microwave Conference Proceedings, Asia-Pacific*, pages 682–684, Dec 2012.

[42] T. Kishigami, K. Kobayashi, M. Otani, T. Morita, H. Mukai, A. Saito, and Y. Nakagawa. Advanced millimeter-wave radar system using coded pulse compression and adaptive array for pedestrian detection. In *Radar Conference, IEEE*, pages 1–6, April 2013.

[43] K. Kobayashi, T. Morita, H. Mukai, T. Kishigami, and Y. Nakagawa. 79ghz-band coded pulse compression radar system performance in outdoor for pedestrian detection. In *Microwave Conference, European*, pages 1639–1642, Oct 2013.

[44] D.T. Linzmeier, M. Skutek, M. Mekhaiel, and K.C.J. Dietmayer. A pedestrian detection system based on thermopile and radar sensor data fusion. In *Information Fusion, 8th International Conference on*, volume 2, pages 8 pp.–, July 2005.

[45] H. Rohling, S. Heuel, and H. Ritter. Pedestrian detection procedure integrated into an 24 ghz automotive radar. In *Radar Conference, IEEE*, pages 1229–1232, May 2010.

[46] G. Gate and F. Nashashibi. Fast algorithm for pedestrian and group of pedestrians detection using a laser scanner. In *Intelligent Vehicles Symposium, IEEE*, pages 1322–1327, June 2009.

[47] C. Premebida, O. Ludwig, M. Silva, and U. Nunes. A cascade classifier applied in pedestrian detection using laser and image-based features. In *Intelligent Transportation Systems, International IEEE Conference on*, pages 1153–1159, Sept 2010.

[48] Z. J. Chong, B. Qin, T. Bandyopadhyay, T. Wongpiromsarn, E. S. Rankin, M.H. Ang, E. Frazzoli, D. Rus, D. Hsu, and K. H. Low. Autonomous personal vehicle for the first- and last-mile transportation services. In *Cybernetics and Intelligent Systems, International IEEE Conference on*, pages 253–260, Sept 2011.

[49] O. Aycard, Q. Baig, S. Bota, F. Nashashibi, S. Nedevschi, C. Pantilie, M. Parent, P. Resende, and Trung-Dung Vu. Intersection safety using lidar and stereo vision sensors. In *Intelligent Vehicles Symposium, IEEE*, pages 863–869, June 2011.

[50] F. Garcia, B. Musleh, A. De la Escalera, and J.M. Armingol. Fusion procedure for pedestrian detection based on laser scanner and computer vision. In *Intelligent*

Transportation Systems, International IEEE Conference on, pages 1325–1330, Oct 2011.

[51] S. Munder and D.M. Gavrila. An experimental study on pedestrian classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(11):1863–1868, Nov 2006.

[52] S. Paisitkriangkrai, C. Shen, and J. Zhang. Performance evaluation of local features in human classification and detection. *Computer Vision, IET*, 2(4):236–246, December 2008.

[53] M. Enzweiler and D.M. Gavrila. Monocular pedestrian detection: Survey and experiments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(12):2179–2195, Dec 2009.

[54] J. Junior, S. Raupp Musse, and C.R. Jung. Crowd analysis using computer vision techniques. *Signal Processing Magazine, IEEE*, 27(5):66–77, Sept 2010.

[55] J. Candamo, M. Shreve, D.B. Goldgof, D.B. Sapper, and R. Kasturi. Understanding transit scenes: A survey on human behavior-recognition algorithms. *Intelligent Transportation Systems, IEEE Transactions on*, 11(1):206–224, March 2010.

[56] B. Zhan, D.N. Monekosso, P. Remagnino, S.A. Velastin, and L. Xu. Crowd analysis: A survey. *Machine Vision and Applications*, 19(5-6):345–357, September 2008.

[57] T.B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2):90–126, November 2006.

[58] M. Oren, C.P. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. pages 193–99, 1997.

[59] J. Marin, D. Vazquez, A.M. Lopez, J. Amores, and L.I. Kuncheva. Occlusion handling via random subspace classifiers for human detection. *Cybernetics, IEEE Transactions on*, 44(3):342–354, March 2014.

[60] CAVIAR project/IST 2001 37540. Caviar test case scenarios. http://homepages.inf.ed.ac.uk/rbf/CAVIAR/, Sep 2005. [Online; accessed 22-May-2014].

[61] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 304–311, June 2009.

[62] D.M. Gavrila and J. Giebel. Shape-based pedestrian detection and tracking. In *Intelligent Vehicle Symposium, IEEE*, volume 1, pages 8–14 vol.1, June 2002.

[63] T. Zhao, R. Nevatia, and F. Lv. Segmentation and tracking of multiple humans in complex situations. In *Computer Vision and Pattern Recognition, Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–194–II–201 vol.2, 2001.

[64] J. Rittscher, P.H. Tu, and N. Krahnstoever. Simultaneous estimation of segmentation and shape. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, volume 2, pages 486–493 vol. 2, June 2005.

[65] G. Chen, T.X. Han, and S. Lao. Adapting an object detector by considering the worst case: A conservative approach. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1369–1376, June 2011.

[66] D.M. Gavrila. A bayesian, exemplar-based approach to hierarchical shape matching. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(8):1408–1421, Aug 2007.

[67] Y Wu and T. Yu. A field model for human detection and tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(5):753–765, May 2006.

[68] W. Ge and R.T. Collins. Crowd detection with a multiview sampler. In *Proceedings of the 11th European Conference on Computer Vision: Part V*, ECCV'10, pages 324–337, Berlin, Heidelberg, 2010. Springer-Verlag.

[69] C.P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Computer Vision, Sixth International Conference on*, pages 555–562, Jan 1998.

[70] P. Viola, M.J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *Computer Vision, Proceedings. Ninth IEEE International Conference on*, pages 734–741 vol.2, Oct 2003.

[71] T. Ojala, M. Pietikäinen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996.

[72] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, 2002.

[73] L. Zhang, R. Chu, S. Xiang, S. Liao, and S.Z. Li. Face detection based on multi-block lbp representation. In *Advances in biometrics*, pages 11–18. Springer, 2007.

[74] Topi Mäenpää. *The Local binary pattern approach to texture analysis: Extenxions and applications*. Oulun yliopisto, 2003.

[75] M. Andriluka, S. Roth, and B. Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1014–1021, June 2009.

[76] F. Yang, Y. Lu, and B. Li. Human body detection using multi-scale shape contexts. In *Bioinformatics and Biomedical Engineering, International Conference on*, pages 1–4, June 2010.

[77] P. Liu, J. Wang, M. She, and H. Liu. Human action recognition based on 3d sift and lda model. In *Robotic Intelligence In Informationally Structured Space, IEEE Workshop on*, pages 12–17, April 2011.

[78] W.S. Leoputra, S. Venkatesh, and T. Tan. Pedestrian detection for mobile bus surveillance. In *Control, Automation, Robotics and Vision, 10th International Conference on*, pages 726–732, Dec 2008.

[79] A.Q.M. Sabri, J. Boonaert, S. Lecoeuche, and E. Mouaddib. Human action classification using surf based spatio-temporal correlated descriptors. In *Image Processing, 19th IEEE International Conference on*, pages 1401–1404, Sept 2012.

[80] K. Suzuki, K. Takashio, H. Tokuda, M. Wada, Y. Matsuki, and K. Umeda. Toward real-time extraction of pedestrian contexts with stereo camera. In *Networked Sensing Systems, 5th International Conference on*, pages 115–118, June 2008.

[81] M. Kawabe, J.K. Tan, H. Kim, S. Ishikawa, and T. Morie. Extraction of individual pedestrians employing stereo camera images. In *Control, Automation and Systems, 11th International Conference on*, pages 1744–1747, Oct 2011.

[82] K. Al-Mutib, M. Emaduddin, M. Alsulaiman, H. Ramdane, and E. Mattar. Motion periodicity based pedestrian detection and particle filter based pedestrian tracking using stereo vision camera. In *Mechatronics and Machine Vision in Practice, 19th International Conference*, pages 32–37, Nov 2012.

[83] J. Yan, Z. Lei, D. Yi, and S.Z. Li. Multi-pedestrian detection in crowded scenes: A global view. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 3124–3129, June 2012.

[84] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, volume 1, pages 878–885 vol. 1, June 2005.

[85] B. Wu, R. Nevatia, and Y. Li. Segmentation of multiple, partially occluded objects by grouping, merging, assigning part detection responses. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–8, June 2008.

[86] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on statistical learning in computer vision, ECCV*, volume 2, pages 7–14, 2004.

[87] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1022–1029, June 2009.

[88] S. Maji and J. Malik. Object detection using a max-margin hough transform. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1038–1045, June 2009.

[89] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(4):349–361, Apr 2001.

[90] A. Shashua, Y. Gdalyahu, and G. Hayun. Pedestrian detection for driving assistance systems: single-frame classification and system level performance. In *Intelligent Vehicles Symposium, IEEE*, pages 1–6, June 2004.

[91] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–8, June 2008.

[92] C. Huang, H. Al, B. Wu, and S. Lao. Boosting nested cascade detector for multi-view face detection. In *Pattern Recognition, Proceedings of the 17th International Conference on*, volume 2, pages 415–418 Vol.2, Aug 2004.

[93] P. Dollr, B.S.B. Babenko, P. Perona, and Z. Tu. Multiple component learning for object detection. In *In Proc. of ECCV*, 2008.

[94] Z. Lin and L.S. Davis. A pose-invariant descriptor for human detection and segmentation, 2008.

[95] H. Wang, R. Lu, X. Wu, L. Zhang, and J. Shen. Pedestrian detection and tracking algorithm design in transportation video monitoring system. In *Information Technology and Computer Science, International Conference on*, volume 2, pages 53–56, July 2009.

[96] T. Du, D. Xu, N. Li, L. He, and J. Liu. Research on dynamic vehicle detection and image capture based on adaptive background. In *Intelligent Control and Automation, The Sixth World Congress on*, volume 2, pages 8577–8581, 2006.

[97] D. Park, C.L. Zitnick, D. Ramanan, and P. Dollar. Exploring weak stabilization for motion feature extraction. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 2882–2889, June 2013.

[98] S. Kuriyama, G. Hasegawa, and H. Nakano. Real-time estimation method of the number of pedestrians in video sequences. In *Digital Telecommunications, Fourth International Conference on*, pages 65–70, July 2009.

[99] Q. Liu, O. Osechas, and J. Rife. Optical flow measurement of human walking. In *Position Location and Navigation Symposium, IEEE/ION*, pages 547–554, April 2012.

[100] B. Krausz and C. Bauckhage. Analyzing pedestrian behavior in crowds for automatic detection of congestions. In *Computer Vision Workshops, IEEE International Conference on*, pages 144–149, Nov 2011.

[101] Y. Lin, F. Guo, and S. Li. Improvement of optical flow in pedestrian detection based on pictorial structure. In *Intelligent Computing and Intelligent Systems, IEEE International Conference on*, volume 3, pages 917–921, Oct 2010.

[102] T. Nakada, S. Kagami, and H. Mizoguchi. Pedestrian detection using 3d optical flow sequences for a mobile robot. In *Sensors, 2008 IEEE*, pages 776–779, Oct 2008.

[103] D. Schreiber and M. Rauter. Gpu-based non-parametric background subtraction for a practical surveillance system. In *Computer Vision Workshops, IEEE 12th International Conference on*, pages 870–877, Sept 2009.

[104] Z. Jiang, D.Q. Huynh, W. Moran, and S. Challa. Combining background subtraction and temporal persistency in pedestrian detection from static videos. In *Image Processing, 20th IEEE International Conference on*, pages 4141–4145, Sept 2013.

[105] Y. Xu, L. Xu, D. Li, and Y. Wu. Pedestrian detection using background subtraction assisted support vector machine. In *Intelligent Systems Design and Applications, 11th International Conference on*, pages 837–842, Nov 2011.

[106] A.J. Lipton, H. Fujiyoshi, and R.S. Patil. Moving target classification and tracking from real-time video. In *Applications of Computer Vision, Proceedings., Fourth IEEE Workshop on*, pages 8–14, Oct 1998.

[107] C. H. Anderson, P. J. Burt, and G. S. van der Wal. Change detection and tracking using pyramid transform techniques, 1985.

[108] D. Meyer, J. Denzler, and H. Niemann. Model based extraction of articulated objects in image sequences for gait analysis. In *Image Processing, Proceedings., International Conference on*, volume 3, pages 78–81 vol.3, Oct 1997.

[109] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *Int. J. Comput. Vision*, 12(1):43–77, February 1994.

[110] A. Verri, S. Uras, and E. De Micheli. Motion segmentation from optical flow. In *Alvey Vision Conference*, pages 1–6, 1989.

[111] C. Ridder, O. Munkelt, and H. Kirchner. Adaptive background estimation and foreground detection using kalman-filtering. In *Proceedings of International Conference on recent Advances in Mechatronics*, pages 193–199, 1995.

[112] C. Stauffer and W. E L Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on.*, volume 2, pages –252 Vol. 2, 1999.

[113] P. Kaewtrakulpong and R. Bowden. An improved adaptive background mixture model for realtime tracking with shadow detection, 2001.

[114] D.R. Magee. Tracking multiple vehicles using foreground, background and motion models. *Image and Vision Computing*, 22:143–155, 2001.

[115] W. Long and Y.H. Yang. *Stationary background generation: an alternative to the difference of two images*, volume 23. Elsevier, 1990.

[116] D. Gutchess, M. Trajkovics, E. Cohen-Solal, D. Lyons, and A. K. Jain. A background model initialization algorithm for video surveillance. In *Computer Vision, Proceedings., Eighth IEEE International Conference on*, volume 1, pages 733–740 vol.1, 2001.

[117] D. Gutchess, E. Cohen-solal, D. Lyons, and A.K. Jain. In *Computer Vision, Proceedings., Eighth IEEE International Conference on*, volume 1, pages 733–740. IEEE, 2001.

[118] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *Proceedings of the 9th European Conference on Computer Vision - Volume Part II*, ECCV'06, pages 428–441, Berlin, Heidelberg, 2006. Springer-Verlag.

[119] S. Walk, N. Majer, K. Schindler, and B. Schiele. New features and insights for pedestrian detection. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1030–1037, June 2010.

[120] X. Wang, T.X. Han, and S. Yan. An hog-lbp human detector with partial occlusion handling. In *Computer Vision, IEEE 12th International Conference on*, pages 32–39, Sept 2009.

[121] C. Zeng and H. Ma. Robust head-shoulder detection by pca-based multilevel hog-lbp detector for people counting. In *Pattern Recognition (ICPR), 20th International Conference on*, pages 2069–2072, Aug 2010.

[122] P. Viola, M.J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *Computer Vision, Proceedings. Ninth IEEE International Conference on*, pages 734–741 vol.2, Oct 2003.

[123] M.J. Jones and D. Snow. Pedestrian detection using boosted features over many frames. In *Pattern Recognition, 19th International Conference on*, pages 1–4, Dec 2008.

[124] J. Yao and J.M. Odobez. Fast Human Detection from Videos Using Covariance Features. In *The Eighth International Workshop on Visual Surveillance - VS2008*, Marseille, France, 2008. Graeme Jones and Tieniu Tan and Steve Maybank and Dimitrios Makris.

[125] W.R. Schwartz, A. Kembhavi, D. Harwood, and L.S. Davis. Human detection using partial least squares analysis. In *Computer Vision, IEEE 12th International Conference on*, pages 24–31, Sept 2009.

[126] T. Machida and T. Naito. Gpu & cpu cooperative accelerated pedestrian and vehicle detection. In *Computer Vision Workshops, IEEE International Conference on*, pages 506–513, Nov 2011.

[127] Y. Said, M. Atri, and R. Tourki. Human detection based on integral histograms of oriented gradients and svm. In *Communications, Computing and Control Applications, International Conference on*, pages 1–5, 2011.

[128] L. Guo, M. Zhang, L. Li, Y. Zhao, and R. Wang. Research of pedestrian detection for intelligent vehicle based on machine vision. In *Robotics and Biomimetics, IEEE International Conference on*, pages 1172–1177, Dec 2009.

[129] W.J. Park, D.H. Kim, S. Suryanto, C.G. Lyuh, T.M. Roh, and S.J. Ko. Fast human detection using selective block-based hog-lbp. In *Image Processing, 19th IEEE International Conference on*, pages 601–604, Sept 2012.

[130] J. Jiang and H. Xiong. Fast pedestrian detection based on hog-pca and gentle adaboost. In *Computer Science Service System, International Conference on*, pages 1819–1822, Aug 2012.

[131] Q. Liu and Y. Qu. Hog and color based adaboost pedestrian detection. In *Natural Computation, Seventh International Conference on*, volume 1, pages 584–587, July 2011.

[132] V.D. Hoang, A. Vavilin, and K.H. Jo. Pedestrian detection approach based on modified haar-like features and adaboost. In *Control, Automation and Systems, 12th International Conference on*, pages 614–618, Oct 2012.

[133] Z. Li and Y. Zhao. Pedestrian detection in single frame by edgelet-lbp part detectors. In *Advanced Video and Signal Based Surveillance, 10th IEEE International Conference on*, pages 420–425, Aug 2013.

[134] L. Zhao and C.E. Thorpe. Stereo- and neural network-based pedestrian detection. *Intelligent Transportation Systems, IEEE Transactions on*, 1(3):148–154, Sep 2000.

[135] D. M. Gavrila and S. Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. *International Journal of Computer Vision*, 73(1):41–59, June 2007.

[136] V. Neagoe, A. Ciotec, and A. Barar. A concurrent neural network approach to pedestrian detection in thermal imagery. In *Communications, 9th International Conference on*, pages 133–136, June 2012.

[137] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[138] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, pages 193–199, Jun 1997.

[139] S. Maji, A.C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–8, June 2008.

[140] I.P. Alonso, D.F. Llorca, M.A. Sotelo, L.M. Bergasa, P. Revenga de Toro, J. Nuevo, M. Ocana, and M.A.G. Garrido. Combination of feature extraction methods for svm pedestrian detection. *Intelligent Transportation Systems, IEEE Transactions on*, 8(2):292–307, June 2007.

[141] S. Munder and D.M. Gavrila. An experimental study on pedestrian classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(11):1863–1868, Nov 2006.

[142] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, June 2000.

[143] C. Nakajima, M. Pontil, B. Heisele, and T. Poggio. Full-body person recognition system. *Pattern recognition*, 36(9):1997–2006, 2003.

[144] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 133–142, New York, NY, USA, 2002. ACM.

[145] J. Wu, W.C. Tan, and J.M. Rehg. Efficient and effective visual codebook generation using additive kernels. *Journal of Machine Learning Research*, 12:3097–3118, November 2011.

[146] J. Wu. A fast dual method for hik svm learning. In *Proceedings of the 11th European Conference on Computer Vision: Part II*, ECCV'10, pages 552–565, Berlin, Heidelberg, 2010. Springer-Verlag.

[147] J. Giménez and L. Màrquez. Svmtool: A general pos tagger generator based on support vector machines. In *Proceedings of the 4th LREC*, 2004.

[148] C. Chang and C. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[149] Vikas S. Svmlin. http://mloss.org/software/view/32/, Nov 2007. [Online; accessed 22-May-2014].

[150] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[151] P. Viola and M. Jones. Robust real-time face detection. In *Computer Vision, Proceedings. Eighth IEEE International Conference on*, volume 2, pages 747–747, 2001.

[152] A. Shashua, Y. Gdalyahu, and G. Hayun. Pedestrian detection for driving assistance systems: single-frame classification and system level performance. In *Intelligent Vehicles Symposium, IEEE*, pages 1–6, June 2004.

[153] F. Harirchi, P. Radparvar, H.A. Moghaddam, F. Dehghan, and M. Giti. Two-level algorithm for mcs detection in mammograms using diverse-adaboost-svm. In *Pattern Recognition, 20th International Conference on*, pages 269–272, Aug 2010.

[154] T.H. Wing, W.L. Hao, and Y.H. Tay. Two-stage license plate detection using gentle adaboost and sift-svm. In *Intelligent Information and Database Systems, First Asian Conference on*, pages 109–114, April 2009.

[155] G. Gualdi, A. Prati, and R. Cucchiara. Covariance descriptors on moving regions for human detection in very complex outdoor scenes. In *Distributed Smart Cameras, Third ACM/IEEE International Conference on*, pages 1–8, Aug 2009.

[156] S. Duan, X. Wang, and W. Wan. The logitboost based on joint feature for face detection. In *Image and Graphics, Seventh International Conference on*, pages 483–488, July 2013.

[157] C. Demirkr and B. Sankur. Multi pose face detection and pose estimation using multiclass logitboost algorithm. In *Signal Processing and Communications Applications Conference, IEEE 18th*, pages 17–20, April 2010.

[158] J. Wu, S. Yang, and L. Zhang. Pedestrian detection based on improved hog feature and robust adaptive boosting algorithm. In *Image and Signal Processing, 4th International Congress on*, volume 3, pages 1535–1539, Oct 2011.

[159] L.B. Fricke. *Traffic Accident Reconstruction, Volume 2*. Northwestern Univ Center for Public, 1990.

[160] D.M. Gavrila, J. Giebel, and S. Munder. Vision-based pedestrian detection: the protector system. In *Intelligent Vehicles Symposium, IEEE*, pages 13–18, June 2004.

[161] D. Gavrila. Pedestrian detection from a moving vehicle. In *Proceedings of the 6th European Conference on Computer Vision-Part II*, ECCV '00, pages 37–49, London, UK, UK, 2000. Springer-Verlag.

[162] G. Grubb, A. Zelinsky, L. Nilsson, and M. Rilbe. 3d vision sensing for improved pedestrian safety. In *Intelligent Vehicles Symposium, IEEE*, pages 19–24, June 2004.

[163] M. Soga, T. Kato, M. Ohta, and Y. Ninomiya. Pedestrian detection with stereo vision. In *Data Engineering Workshops, 21st International Conference on*, pages 1200–1200, April 2005.

[164] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(4):743–761, 2012.

[165] D. Gerónimo, A. López, A.D. Sappa, and D. Ponsa. Model features and horizon line estimation for pedestrian detection in advanced driver assistance systems.

[166] Z. Kira, R. Hadsell, G. Salgian, and S. Samarasekera. Long-range pedestrian detection using stereo and a cascade of convolutional network classifiers. In *Intelligent Robots and Systems, IEEE/RSJ International Conference on*, pages 2396–2403, Oct 2012.

[167] B. Bhowmick, S. Bhadra, and A. Sinharay. Stereo vision based pedestrians detection and distance measurement for automotive application. In *Intelligent Systems, Modelling and Simulation, Second International Conference on*, pages 25–29, Jan 2011.

[168] Y. Yang, J. Yang, and D. Guo. Pedestrian detection on moving vehicle using stereovision and 2d cue. In Jian Yang, Fang Fang, and Changyin Sun, editors, *IScIDE*, volume 7751 of *Lecture Notes in Computer Science*, pages 466–474. Springer, 2012.

[169] G. Ma, D. Muller, S.-B. Park, S. Muller-Schneiders, and A. Kummert. Pedestrian detection using a singlemonochrome camera. *Intelligent Transport Systems, IET*, 3(1):42–56, March 2009.

[170] O. Tsimhoni, M.J. Flannagan, T. Minoda, and J. Bärgman. *Pedestrian Detection with Near and Far Infrared Night Vision Enhancement.* UMTRI: Transportation Research Institute. University of Michigan, Transportation Research Institute, 2004.

[171] x. Cao, H. Qiao, and J. Keane. A low-cost pedestrian-detection system with a single optical camera. *Intelligent Transportation Systems, IEEE Transactions on*, 9(1):58–67, March 2008.

[172] W. Ma, P. He, L. Huang, and C. Liu. Context inspired pedestrian detection in far-field videos. In *Pattern Recognition, 20th International Conference on*, pages 3009–3012, Aug 2010.

[173] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4), December 2006.

[174] L.M. Fuentes and S.A. Velastin. People tracking in surveillance applications. *Image and Vision Computing*, 24(11):1165–1171, 2006.

[175] N. Ning and T. Tan. A framework for tracking moving target in a heterogeneous camera suite. In *Control, Automation, Robotics and Vision, 9th International Conference on*, pages 1–5, Dec 2006.

[176] R. Eshel and Y. Moses. Homography based multiple camera detection and tracking of people in a dense crowd. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–8, June 2008.

[177] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. A mobile vision system for robust multi-person tracking. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–8, June 2008.

[178] S. Haykin and N. deFreitas. Special issue on sequential state estimation. *Proceedings of the IEEE*, 92(3):399–400, Mar 2004.

[179] P.F. Gabriel, J/.G. Verly, J.H. Piater, and A. Genon. The state of the art in multiple object tracking under occlusion in video sequences. In *In Advanced Concepts for Intelligent Vision Systems*, pages 166–173, 2003.

[180] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *Information Theory, IEEE Transactions on*, 21(1):32–40, Jan 1975.

[181] Y. Cheng. Mean shift, mode seeking, and clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(8):790–799, Aug 1995.

[182] G.R. Bradski. Computer vision face tracking for use in a perceptual user interface. 1998.

[183] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):564–577, May 2003.

[184] L. Yu and W. Yao. Pedestrian detection fusion method based on mean shift. In *Machine Vision, Second International Conference on*, pages 204–207, Dec 2009.

[185] Z. Li, Q.L. Tang, and N. Sang. Improved mean shift algorithm for occlusion pedestrian tracking. *Electronics Letters*, 44(10):622–623, May 2008.

[186] L. Yu, W. Yao, H. Liu, and F. Liu. A monocular vision based pedestrian detection system for intelligent vehicles. In *Intelligent Vehicles Symposium, IEEE*, pages 524–529, June 2008.

[187] G. Yang and H. Liu. Visual tracking algorithm based on camshift and multi-cue fusion for human motion analysis. In *Systems, Man and Cybernetics, IEEE International Conference on*, pages 1280–1285, Oct 2009.

[188] K. Chen, X. Zhao, G. Jiang, and H. Joo. Video tracking of human-faces with occlusion using enhanced camshift approach. In *Progress in Informatics and Computing, IEEE International Conference on*, volume 2, pages 883–887, Dec 2010.

[189] Y. Luo, H. Yang, and Z. Hu. Human limb motion real-time tracking based on camshift for intelligent rehabilitation system. In *Robotics and Biomimetics, IEEE International Conference on*, pages 343–348, Dec 2009.

[190] R.E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[191] G. Welch and G. Bishop. An introduction to the kalman filter. Technical report, Chapel Hill, NC, USA, 1995.

[192] S.J. Julier and J.K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, Mar 2004.

[193] Y.W. Xu, X.B. Cao, and T. Li. Extended kalman filter based pedestrian localization for collision avoidance. In *Mechatronics and Automation, International Conference on*, pages 4366–4370, Aug 2009.

[194] M. K. Bhuyan, B.C. Lovell, and A. Bigdeli. Tracking with multiple cameras for video surveillance. In *Digital Image Computing Techniques and Applications, 9th Biennial Conference of the Australian Pattern Recognition Society on*, pages 592–599, Dec 2007.

[195] D. Olmeda, A. De la Escalera, and J.M. Armingol. Detection and tracking of pedestrians in infrared images. In *Signals, Circuits and Systems, 3rd International Conference on*, pages 1–6, Nov 2009.

[196] M. Meuter, U. Iurgel, S.B. Park, and A. Kummert. The unscented kalman filter for pedestrian tracking from a moving host. In *Intelligent Vehicles Symposium, IEEE*, pages 37–42, June 2008.

[197] M. Bertozzi, A. Broggi, A. Fascioli, A. Tibaldi, R. Chapuis, and F. Chausse. Pedestrian localization and tracking system with kalman filtering. In *Intelligent Vehicles Symposium, IEEE*, pages 584–589, June 2004.

[198] M. Bertozzi, A. Broggi, M. Carletti, A. Fascioli, T. Graf, P. Grisleri, and M. Meinecke. Ir pedestrian detection for advanced driver assistance systems. In *Pattern Recognition*, pages 582–590. Springer, 2003.

[199] E. Binelli, A. Broggi, A. Fascioli, S. Ghidoni, P. Grisleri, T. Graf, and M. Meinecke. A modular tracking system for far infrared pedestrian recognition. In *Intelligent Vehicles Symposium, Proceedings. IEEE*, pages 759–764, June 2005.

[200] N.J. Gordon, D.J. Salmond, and A. F M Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107–113, Apr 1993.

[201] V. Philomin, R. Duraiswami, and L. Davis. Pedestrian tracking from a moving vehicle. In *Intelligent Vehicles Symposium, Proceedings of the IEEE*, pages 350–355, 2000.

[202] R. Arndt, R. Schweiger, W. Ritter, D. Paulus, and O. Lohlein. Detection and tracking of multiple pedestrians in automotive applications. In *Intelligent Vehicles Symposium, IEEE*, pages 13–18, June 2007.

[203] Z. Wang, Z. Ji, and H. Xie. A novel method for pedestrian tracking for infrared image sequences. In *Test and Measurement, International Conference on*, volume 1, pages 200–203, Dec 2009.

[204] G. Chetty. Comparative evaluation of two multisensory video surveillance techniques for pedestrian tracking. In *Signal Processing and Communication Systems, 2nd International Conference on*, pages 1–6, Dec 2008.

[205] S. Gidel, P. Checchin, C. Blanc, T. Chateau, and L. Trassoudaine. Pedestrian detection and tracking in an urban environment using a multilayer laser scanner. *Intelligent Transportation Systems, IEEE Transactions on*, 11(3):579–588, Sept 2010.

[206] M.K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599, 1999.

[207] S.K. Zhou, R. Chellappa, and B. Moghaddam. Visual tracking and recognition using appearance-adaptive models in particle filters. *Image Processing, IEEE Transactions on*, 13(11):1491–1506, Nov 2004.

[208] J. Vermaak, S.J. Godsill, and P. Perez. Monte carlo filtering for multi target tracking and data association. *Aerospace and Electronic Systems, IEEE Transactions on*, 41(1):309–332, Jan 2005.

[209] N. Bouaynaya and D. Schonfeld. On the optimality of motion-based particle filtering. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(7):1068–1072, July 2009.

[210] C. Chen and D. Schonfeld. A particle filtering framework for joint video tracking and pose estimation. *Image Processing, IEEE Transactions on*, 19(6):1625–1634, June 2010.

[211] F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on lie algebra. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, volume 1, pages 728–735, June 2006.

[212] B. Babenko, M.H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 983–990, June 2009.

[213] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–8, June 2008.

[214] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[215] N. Carey. Establishing pedestrian walking speeds. *Karen Aspelin, Portland State University*, 2005.

[216] TourismPEI.com. Driving to prince edward island. http://www.tourismpei.com/drive-to-pei. [Online; accessed 22-May-2014].

[217] novascotia.ca. Rules of the road. http://www.novascotia.ca/snsmr/rmv/handbook/DH-Chapter2.pdf. [Online; accessed 22-May-2014].

[218] novascotia.ca. Safer school zone. http://novascotia.ca/tran/roadsafety/schoolzonesafetyq&a.asp, Sep 2012. [Online; accessed 22-May-2014].

[219] E. Hecht and A. Zajac. *Optics*. Pearson Education (US), 2002.

[220] P. Protopapas. Hidden markov models. http://iacs-courses.seas.harvard.edu/courses/am207/blog/lecture-18.html, April 2014. [Online; accessed 22-May-2014].

[221] J.Y. Bouguet. *Visual Methods for Three-dimensional Modeling*. PhD thesis, Pasadena, CA, USA, 1999. AAI9941097.

[222] C. Papageorgiou, T. Evgeniou, and T. Poggio. A trainable pedestrian detection system. In *Proceeding of Intelligent Vehicles*, pages 241–246, October 1998.

[223] US Department of Transportation Federal Highway Administration. Lane width. http://safety.fhwa.dot.gov/geometric/pubs/mitigationstrategies/chapter3/3_lanewidth.htm. [Online; accessed 22-May-2014].

[224] A. Boukerche, R. W.N. Pazzi, and R.B. Araujo. Hpeq a hierarchical periodic, event-driven and query-based wireless sensor network protocol. In *Local Computer Networks, The IEEE Conference on*, pages 560–567. IEEE, 2005.

[225] A. Boukerche and X. Fei. A voronoi approach for coverage protocols in wireless sensor networks. In *Global Telecommunications Conference, IEEE*, pages 5190–5194. IEEE, 2007.

[226] A. Bamis, A. Boukerche, I. Chatzigiannakis, and S. Nikoletseas. A mobility aware protocol synthesis for efficient routing in ad hoc mobile networks. *Computer Networks*, 52(1):130–154, 2008.

[227] A. Boukerche. *Algorithms and protocols for wireless, mobile Ad Hoc networks*, volume 77. John Wiley & Sons, 2008.

[228] H.A. de Oliveira, A. Boukerche, E. Freire Nakamura, and A.A.F. Loureiro. An efficient directed localization recursion protocol for wireless sensor networks. *Computers, IEEE Transactions on*, 58(5):677–691, 2009.

[229] T. Antoniou, I. Chatzigiannakis, G. Mylonas, S. Nikoletseas, and A. Boukerche. A new energy efficient and fault-tolerant protocol for data propagation in smart dust networks using varying transmission range. In *Proceedings of the 37th annual symposium on Simulation*, page 43. IEEE Computer Society, 2004.

[230] H.A. de Oliveira, E.F. Nakamura, A.A.F. Loureiro, and A. Boukerche. Localization in time and space for sensor networks. In *Advanced Information Networking and Applications, 21st International Conference on*, pages 539–546. IEEE, 2007.