Graduate School ETD Form 9
(Revised 12/07)

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Pingge Jiang

Entitled
A new approach for pedestrian tracking and status analysis

For the degree of Master of Science in Electrical and Computer Engineering

Is approved by the final examining committee:

Eliza Yingzi Du
Chair

Brian King

Rizkalla Mahar

To the best of my knowledge and as understood by the student in the *Research Integrity and Copyright Disclaimer (Graduate School Form 20)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): Eliza Yingzi Du

Approved by: Brian King 09/18/2013
Head of the Graduate Program Date

A NEW APPROACH FOR PEDESTRIAN TRACKING AND STATUS ANALYSIS

A Thesis

Submitted to the Faculty

of

Purdue University

by

Pingge Jiang

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Electrical and Computer Engineering

December 2013

Purdue University

Indianapolis, Indiana

ACKNOWLEDGMENTS

I would like to gratefully thank Dr. Eliza Du. As my thesis advisor, she gave me consistent encouragement and guidance in my courses and research projects during the past two years. Without her wholehearted support, I can never accomplish my academic work successfully.

I would like to thank my advisory committee members, Dr. Brian King and Dr. Maher Rizkalla for their time and help during the completion of this thesis.

I would also like to thank all my lab mates at the Pattern Recognition and Biometric Laboratory, Dr. Zhi Zhou, Mr. Kai Yang, Mr. Shan Cong and Ms. Feng Jiang for their support and help during the chapter of my life. I thank Ms. Sherrie Tucker for assisting me in formatting this thesis.

Finally I thank all my family, who mean more to me than words can possibly express.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

ABSTRACT

Jiang, Pingge. MSECE, Purdue University, December 2013. A new approach for pedestrian tracking and status analysis. Major Professor: Eliza Du.

Pedestrian and vehicle interaction analysis in a naturalistic driving environment can provide useful information for designing vehicle-pedestrian crash warning and mitigation systems. Many researchers have used crash data to understand and study pedestrian behaviors and interactions between vehicles and pedestrian during crash. However, crash data may not provide detailed pedestrian-vehicle interaction information for us.

In this thesis, we designed an automatic pedestrian tracking and status analysis method to process and study pedestrian and vehicle interactions. The proposed pedestrian tracking and status analysis method includes pedestrian detection, pedestrian tracking and pedestrian status analysis modules.

The main contributions of this thesis are: we designed a new pedestrian tracking method by learning the pedestrian appearance and also their motion pattern. We designed a pedestrian status estimation method by using our tracking results and thus helped estimate the possibility of collision.

Our preliminary experiment results using naturalistic driving data showed promising results.

1. INTRODUCTION

1.1 Background

Vehicle-pedestrian crashes are often fatal and have serious consequences [1–3]. In the United States, 4280 pedestrian were killed in traffic crashes in 2010, with around 70,000 injuries according to the National Highway Traffic Administration (NHTSA). In Europe, more than 30,000 people were killed on the road in 2011 based on the European Commission data [4]. In Asia, more than 130,000 deaths reported in India alone with more than 25 percent being pedestrians [5]. Recently, the crash numbers in Asia are increasing every year with more and more vehicles on the road [6].

There are various factors that contributed to the vehicle-pedestrian accidents, including careless drivers and pedestrians, speeding, bad weather, low illumination, and/or poor road design, etc. [7, 8]. Reducing vehicle-pedestrian crashes and improving pedestrian safety becomes an important research topic.

Many researchers have used crash data to understand and study pedestrian behaviors during crashes. These results have been used in the development of regulations and laws [6], improvement of road design [9–11], development of active transportation systems with some crash warning/mitigation capability [12–14], and various educational programs to raise awareness of transportation safety among the general population [15, 16].

However, there are many limitations and challenges just by using the crash data. First of all, accident situations can be very complicated and beyond what we can describe in the accident records, it is difficult to get the comprehensive knowledge of

the whole process. Also, the information we get from accident data may not applicable for everyone. Some of the accidents may only be able to happen in particular situations.

On the other hand, naturalistic driving data can provide us comprehensive information by recording what has happened. It provides objective information about the driving environment, pedestrian behavior and vehicle-pedestrian interaction for every minute and also the whole process of potential accidents.

Pedestrian behavior can give us a lot of information on what the pedestrian is likely to do in the future and help us to improve the accuracy of collision prediction. In order to learn pedestrian behavior and fully understand the collision scenarios, we hired 110 cars to collect naturalistic driving data for a one year period. Until now, we have collected more than 90 terabytes of data with around 400,000 videos and over 33,333 driving hours. Pedestrian status learning is an important basis of pedestrian behavior analysis. To handle such a large amount of data, it would be desirable to have an automatic way for estimating pedestrian status. In this thesis, we proposed a new pedestrian tracking algorithm to help us efficiently determine the pedestrian status distribution.

1.2 Challenges

Its a quite challenging task to process the large-scale naturalistic driving data on account of the following reasons:

- Complex driving environment. Varying road types, weather and illumination conditions make it difficult to accurately recognize the pedestrians and their behavior changes, especially with low contrast or complex environment backgrounds. Figure 1.1 shows some driving environment examples.

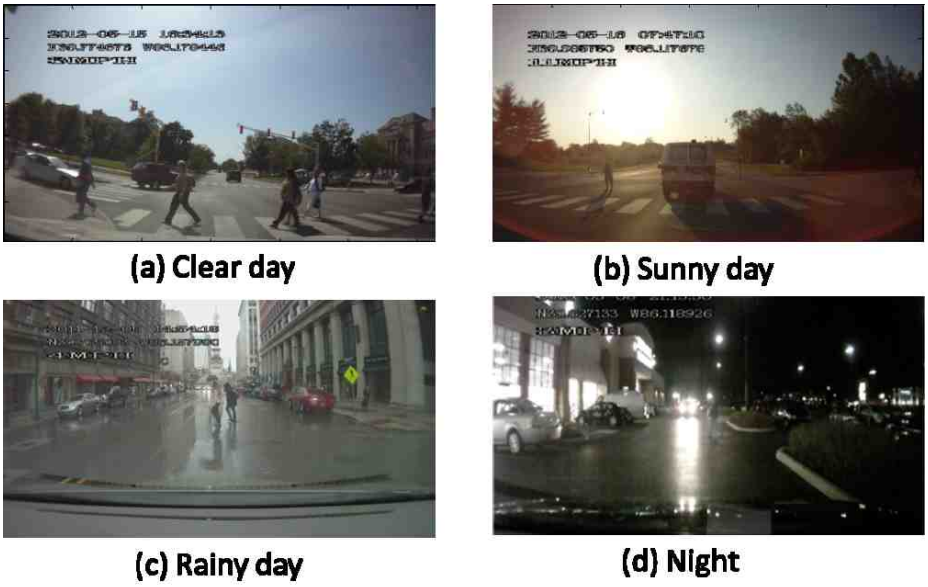

Fig. 1.1. Various driving environment

- Complex pedestrian appearance/viewpoint. The appearance of pedestrians in naturalistic driving data is constantly changing. The difference of size, height and pose make it difficult to accurately track the people. Also the same pedestrian captured from different viewpoints may not look the same. Figure 1.2 shows some pedestrians in our database.

Fig. 1.2. Different pedestrian appearance/viewpoint

- Complex data quality. Sometimes data quality is not good for feature extraction. Blurred images of pedestrians make it very hard to detect even for human eyes. Figure 1.3 shows some example of blurred pedestrians in the videos.

Fig. 1.3. Blurred pedestrians in the videos

1.3 Summary of contributions

The main contributions of our work are summarized as:

- We developed a new robust pedestrian tracking algorithm based on feature and motion learning for monocular vision.
- We proposed pedestrian status estimation method based on pedestrian tracking results and real-time GPS information.
- We developed a pedestrian pre-collision analysis method, which can help estimate the possibility of collision by evaluating the vehicle-pedestrian relationship.

1.4 Organization

The thesis is organized as follows: Chapter 2 introduces our naturalistic data collection and analysis process. Chapter 3 provides background information about pedestrian classification and pedestrian tracking. In Chapter 4, we propose the feature matching and motion learning based pedestrian tracking method. In Chapter 5, we present experiment results. Finally, Chapter 6 draws the conclusion and discuss about the future work.

2. NATURALISTIC DATA COLLECTION AND ANALYSIS

2.1 Apparatus

In this project, we installed a DOD GS600 camera on each subject vehicle to collect the naturalistic driving data. Three types of data were collected: video, GPS location information and G-sensor acceleration information. In this study, the videos are set to be 5 minutes long and each will have corresponding log files to save GPS and G-sensor information during the period. DOD GS600 camera is a 120^0 wide angle lens camera with $720 * 1280$ resolutions, 30 frames per second. Figure 2.1 shows a sample video frame. From the left top corner of the frame, we can get current time and data, real-time GPS information and current car velocity. Also, we will have the log files with recorded GPS and G-sensor information as shown in Figure 2.2 and Figure 2.3.

Fig. 2.1. Sample frame of our recorded videos

2012-09-08 08:47:23	N39.693758	W86.137968	0	7.20	176
2012-09-08 08:47:26	N39.693751	W86.137895	207.05	7.70	211
2012-09-08 08:47:27	N39.693746	W86.137911	207.93	7.20	236
2012-09-08 08:47:29	N39.693746	W86.137961	208.13	5.00	262
2012-09-08 08:47:31	N39.693736	W86.137970	209.61	0.00	262
2012-09-08 08:47:33	N39.693685	W86.137948	203.87	7.90	174
2012-09-08 08:47:35	N39.693638	W86.137935	203.11	9.40	172
2012-09-08 08:47:37	N39.693595	W86.137930	201.50	9.30	167
2012-09-08 08:47:39	N39.693521	W86.137866	199.50	16.00	144
2012-09-08 08:47:41	N39.693468	W86.137721	197.67	25.30	118
2012-09-08 08:47:43	N39.693430	W86.137506	197.51	33.10	106
2012-09-08 08:47:45	N39.693418	W86.137283	197.25	34.70	98
2012-09-08 08:47:47	N39.693426	W86.137056	197.09	33.80	91
2012-09-08 08:47:49	N39.693398	W86.136896	197.68	26.70	97
2012-09-08 08:47:51	N39.693268	W86.136830	198.24	24.10	138
2012-09-08 08:47:53	N39.693090	W86.136756	199.52	34.90	158
2012-09-08 08:47:55	N39.692863	W86.136666	199.21	46.00	162
2012-09-08 08:47:57	N39.692606	W86.136566	198.52	53.20	163
2012-09-08 08:47:59	N39.692358	W86.136460	197.56	52.40	163
2012-09-08 08:48:01	N39.692126	W86.136356	196.97	49.20	162
2012-09-08 08:48:03	N39.691926	W86.136278	197.16	41.00	164
2012-09-08 08:48:05	N39.691786	W86.136228	196.79	29.80	166
2012-09-08 08:48:07	N39.691698	W86.136180	197.24	21.30	163
2012-09-08 08:48:09	N39.691655	W86.136078	197.02	17.80	135
2012-09-08 08:48:11	N39.691643	W86.135990	195.39	14.00	112
2012-09-08 08:48:13	N39.691640	W86.135895	195.79	13.10	97
2012-09-08 08:48:15	N39.691643	W86.135803	195.62	13.40	92
2012-09-08 08:48:17	N39.691651	W86.135675	195.61	20.00	92
2012-09-08 08:48:19	N39.691650	W86.135521	194.94	23.40	93
2012-09-08 08:48:21	N39.691648	W86.135388	195.25	18.90	89
2012-09-08 08:48:23	N39.691673	W86.135300	193.50	10.80	60
2012-09-08 08:48:25	N39.691771	W86.135280	193.35	13.90	10
2012-09-08 08:48:27	N39.691846	W86.135331	193.70	16.30	336
2012-09-08 08:48:29	N39.691881	W86.135458	194.52	19.80	304
2012-09-08 08:48:31	N39.691883	W86.135588	195.71	17.20	280
2012-09-08 08:48:33	N39.691838	W86.135660	195.97	11.10	236
2012-09-08 08:48:35	N39.691803	W86.135701	196.08	0.00	218

Fig. 2.2. Sample GPS log

[S]	133	51	867			
[S]	118	67	918			
[S]	133	51	918			
[S]	118	51	918			
[S]	133	33	918			
[S]	133	85	900			
[S]	118	33	918			
[S]	118	33	900			
[S]	118	51	900			
[G]	2012-09-07 17:31:41			N39.782758	W86.155803	41	92
[S]	100	18	867			
[S]	100	67	900			
[S]	133	51	900			
[S]	166	33	918			
[S]	118	33	900			
[S]	133	51	951			
[S]	133	51	885			
[S]	133	51	885			
[S]	151	51	900			
[S]	100	51	918			
[G]	2012-09-07 17:31:42			N39.782755	W86.155668	41	92
[S]	133	51	900			
[S]	133	33	918			
[S]	100	67	900			
[S]	118	51	885			
[S]	151	51	918			
[S]	133	51	867			
[S]	133	67	900			
[S]	133	51	885			
[S]	133	-33	966			
[S]	151	-18	900			
[G]	2012-09-07 17:31:44			N39.782748	W86.155380	44	91
[S]	118	33	933			
[S]	151	51	885			
[S]	133	51	900			
[S]	151	67	885			
[S]	118	67	918			

Fig. 2.3. Sample G-sensor log

2.2 Data collection

We recruited 110 cars in the Indianapolis area for a one years naturalistic driving study that began in 2012. The study participants span a wide range of age, education, vehicle model and driving habits to make the collected data varied and informative. We installed the DOD GS600 on each vehicle behind the rear-view camera. It records the real-time GPS information, G-sensor acceleration information and high-resolution videos. We developed several tools and algorithms to automatically and efficiently process the large amount of data.

For each human subject, we assigned several SD cards to save the video and log files from their driving and they return the cards to us every weekend. Managing the large amount of data is not a trival task. We developed an automatic data management tool to help us collect and classify different kinds of data. Figure 2.4 shows the user graphic interface of the data manage tool.

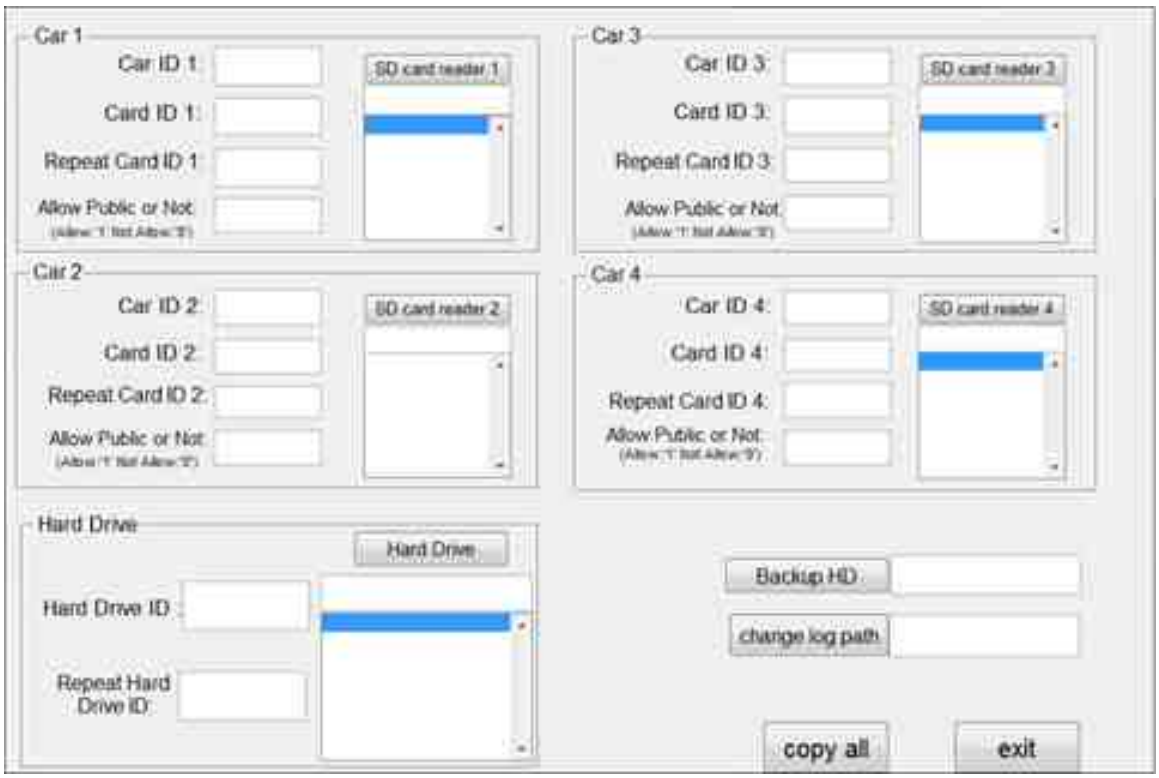

Fig. 2.4. Graphic user interface for data management

The capabilities of the data management tool are:

- Properly store and organize information, such as hard drive location, data type, process date, etc. We separate the video files, GPS log files and G-sensor files for each human subject based on data acquisition date. The file names are changed to a fixed 18 digits structure: 3 digits car information, 6 digits record date information, 4 digits record time information and 5 digits index. The structure is shown as in Figure 2.5.

Fig. 2.5. Filename structure

- Properly record, calculate and organize information related to each car such as recorded mileage, recorded time, etc. For every 5 minutes of video, we find the corresponding GPS information and calculate the mileage the car drove in this period.
- Provide accurate information for human subject payment.
- Create logs; such as mileage log, daily log, car log, overall log, etc. Several logs are automatically created to record our process. The car log will be created and record the driving information for each car, including the videos, GPS and G-sensor logs, data type, record and process date, time and mileage. Daily log and overall log are focusing on the data for all the cars we process in one day/one period and regardless which car it comes from. Mileage log will record the calculated mileage information of each car, including the total mileage and total driving time we processed, data record date and data process date. Count log indicates remain us how many files weve processed.
- Separate the files that are public and non-public.

2.3 Data processing

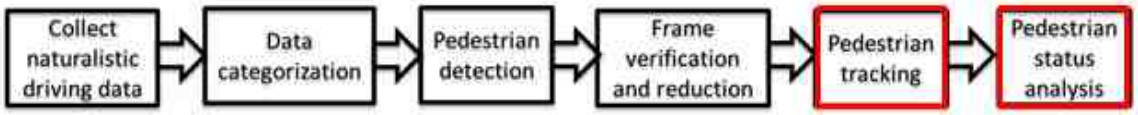

Fig. 2.6. Data processing steps

Figure 2.6 shows the overall chart of data processing steps before our pedestrian status analysis. After collecting the naturalistic driving data, we designed the categorization algorithms to automatically classify the driving scenarios into several categorizations, which include location categorization, time categorization and weather categorization, etc. Different categorization will result in different pedestrian appearance probability and potential conflict probability. Based on this information, an automatic pedestrian detection algorithm is developed to detect the pedestrians in the videos. No automatic pedestrian system can achieve 100% accuracy thus the results are verified and processed by reductionists. The tasks of reductionists include verifying the correctly detected pedestrians and eliminating the falsely detected frames and repeatedly detected pedestrian using our frame reduction tool.

2.3.1 Data categorization

The collected data (including video files and relevant data files) will be firstly categorized based on the GPS, G-sensor and data information etc. the categorization will focus on classifying the driving event, driving location and weather condition which will help to improve the efficiency in pedestrian recognition. The overall architecture is shown in Table 2.1.

Table 2.1 Categorization architecture [17]

Category	Characteristics	Designed method
Low probability of pedestrian (freeway, rural places, highway and suburb areas, etc.)	Low pedestrian appearance probability; high FAR	Pedestrian constrains based fast algorithm
High probability of pedestrian (downtown, communities, schools, shopping areas, etc.)	High pedestrian appearance probability; high FRR	Feature based classification

Two kinds of locations are categorized: locations with a low chance to see pedestrians, such as highway, rural and suburb; and locations with a high chance to see pedestrians, such as downtown, shopping malls and school area. Before doing pedestrian tracking, we label the locations by GPS information.

1) Low probability location pedestrian detection

For the locations with low chance to see pedestrian, such as highway and rural area, backgrounds are not as complicated as the urban area. Thus, we use background subtraction method to efficiently generate the binary foreground as regions of interest.

2) High chance location pedestrian detection

For the locations with a high chance to see pedestrians, such as downtown and schools, more complicated feature based descriptor is needed. We use feature descriptors to represent the searching area and use trained classifiers to determine whether it is pedestrian or not.

2.3.2 Categorization based automatic pedestrian detection

Our lab, Yang *et al.* designed the categorization based automatic pedestrian method [17] to process the large-scale naturalistic driving data to detect pedestrians. Figure 2.7 shows the diagram of the pedestrian detection method.

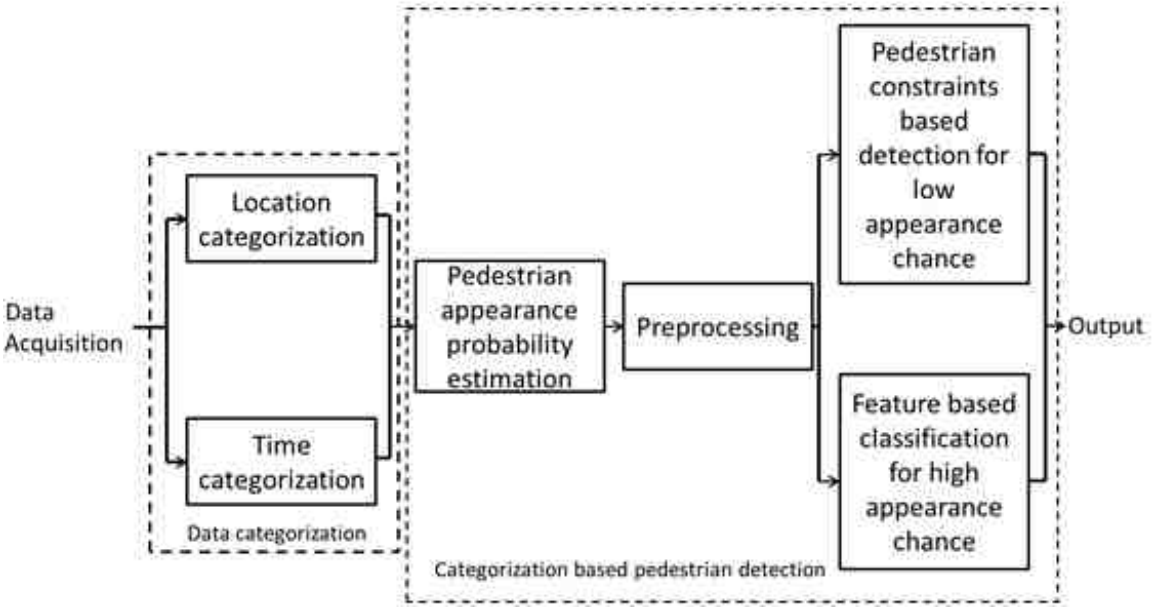

Fig. 2.7. Pedestrian detection diagram [17]

For the situations with low pedestrian appearance probability or vehicle is stopping or slow moving, we will first extract the ROIs for further classification from the background. We observed from the acquired test data that the top (sky) and bottom part (dash board) of the frame cannot include any pedestrian information; therefore we can omit these two regions to improve efficiency. We slide the search window in this area with the prior knowledge of the pedestrian, such as size, position, aspect ratio, etc. And use the silhouette matching method to match a pedestrian with an existing template. The binary template matching algorithm is sufficient for our objective here since in these scenarios, the chance of pedestrian appearance is very low and the variance of pedestrian shape and appearance is very limited.

For the situations with high probability of pedestrian appearance, we will use a new pedestrian detection method based on Histogram of Oriented Gradient and Kernel based Extreme Learning Machine to improve the efficiency and accuracy. HOG is a popularly used pedestrian detector. It relies on computing the overlapping local oriented histograms by learning the distribution of intensity gradients and edge directions. The computed distribution of histograms will be concatenated and trained by Support Vector Machine. A number of positive (patches containing pedestrian) and negative training samples (random selected patches without pedestrian) are used to train the classifier to determine the decision boundary between them. After training, the classifier processes unknown samples and decides the presence or absence of the object based on which side of the decision boundary the feature vector lies.

2.3.3 Frame verification and reduction

The goal of this step is to verify the correctness of pedestrian detection results and eliminate the repeated pedestrians in same scenarios. As the pedestrian detection algorithm is based on frame domain, it will output the detection results for every frame detected with pedestrian. Our trained data reductionists will check each frame to confirm if there was a pedestrian or not. Also for the frames with the same pedestrian in the same scenario, the reductionists will choose the middle frame to represent the whole scenario. We developed a graphic user interface to help the reductionists to choose the frames and automatically record the results in the log.

2.3.4 Pedestrian tracking and pedestrian status analysis

Pedestrian tracking and pedestrian status analysis algorithms are the main contribution in this thesis. The pedestrian found with the detection algorithms is the target pedestrian to track. We proposed a feature and motion based pedestrian track-

ing method to locate the pedestrian in every frame of video. The tracking results are used for vehicle-pedestrian relationship analysis and thus help us to estimate the pedestrian status in the particular scenarios.

2.3.5 Summary

We introduce the overall steps of our data collection and analysis. The data collection is accomplished by a high resolution camera with GPS and G-sensor information. We develop a data management tool to efficiently organize the large amount of data. Then, categorization based pedestrian detection system is performed to detect pedestrians in our collected videos. We categorize the data into two categories with high and low pedestrian appearance probability respectively and apply different pedestrian tracking method for each category. Reductionists verify the pedestrian detection quality and select the desired scenarios for pedestrian tracking. We propose a feature and motion based pedestrian tracking method in this thesis to accurately predict pedestrian location. Pedestrian statuses are analyzed based on tracking results.

3. REVIEW OF PEDESTRIAN TRACKING METHODS

Many researchers have proposed different methods in pedestrian tracking. Markus *et al.* summarized the pedestrian tracking methods into three steps: search Region of Interest (ROI), to hypothesis pedestrian location and improve efficiency; pedestrian classification/registration, to determine pedestrian characteristics; and tracking.

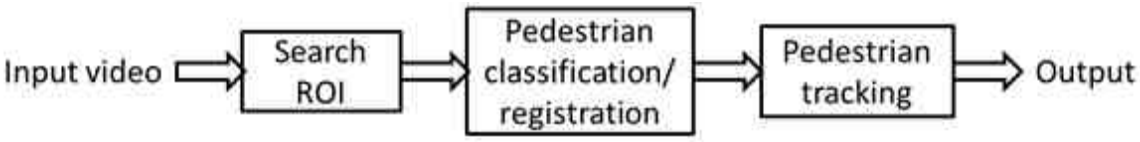

Fig. 3.1. Pedestrian tracking steps

3.1 Search ROI

The general way for existing ROI search method is based on sliding windows at all possible scales and locations in the image [18]. It performs feature matching and classification inside each window [19]. The computational cost of searching throughout the image is often too high for implementation. A number of ways are investigated to improve the efficiency.

Search ROI by moving objects detection. Searching moving objects in the video is a very efficient way in finding pedestrians. Background subtraction method is popularly employed in surveillance approaches for static cameras. For example, Elzein *et al.* [20] computed the variation between two consecutive frames to find the different area as the candidate pedestrian location and ROI will only within the pixels

higher than a pre-set threshold. Ennzweiler *et al.* [21] also used the moving object detection method to find ROI, but he extended its application to moving cameras by using optical flow technique. The regions with unique motion flow were selected.

Search ROI by feature detection. A serious problem of motion detection method is that they will miss the standing or slow moving pedestrians. Thus the approach based on coupling window sliding with feature classifiers is proposed. Shashua *et al.* [22] filtered out most of the search windows with less distinctive properties and incompliance with the specified pedestrian constrains before performing pedestrian detection, and only a small number of windows were left per frame. Violat *et al.* [23] used several Haar-like appearance and motion filter to select the possible regions of interest.

3.2 Pedestrian classification/registration

Once the ROIs are obtained, feature matching and pattern classification methods will be applied for pedestrian classification and registration. Many researches have designed feature-based pedestrian classification/registration methods in the past decades. The development has experienced several stages: 1) model/template matching. 2) discriminative feature matching. 3) component-based matching. And 4) multimodal feature matching

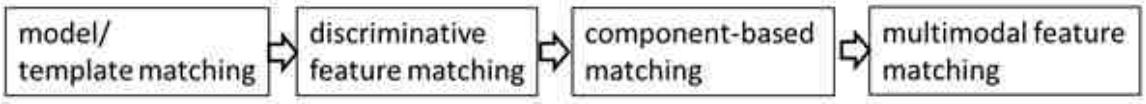

Fig. 3.2. Pedestrian classification/registration methods

Model/template matching. Model/template matching method has wide applications. The advantage of model/template matching method in pedestrian matching is it can eliminate the influence from various clothing and luminance. For example, Gavrila *et al.* [18] applied a coarse-to-fine template matching method to find the best

matches of the pre-defined models. He built a hierarchical architecture offline to label several kinds of shape templates. The matching step involves measure Chamfer distance between the templates with candidate windows. Pedestrian location will be found when the distance within a specified threshold. Later on, Enzweiler *et al.* [21] made it more robust and accurate by constructing shape and texture models separately. However, such models require large amount of examples to cover all the possible shapes of pedestrian.

Discriminative feature matching. It is difficult to apply model/template matching directly without exploiting the appearance features of the pedestrian. Thus, discriminative feature models are attracting more attention. Papageorgiou *et al.* [24] first proposed to use the Haar-wavelet to extract the local shape and structure, only invariant features are captured by computing the derivative of two different regions. Later, Dalal *et al.* [25] proposed to classify a human by Histogram of Oriented Gradients (HOG), which represents local edge structure by calculating gradient and orientation within multiple overlapping blocks and concatenate together. HOG achieved promising results in human detection and has become a classic human descriptor. Similar gradient histogram based approaches are also developed. Wu *et al.* [26] developed a class of feature which is called Edgelet features, which use the gradient and orientation of short segmentations of edge to represent the components of pedestrian. Some other interesting point based pedestrian representation methods are also investigated. For example, Leibe *et al.* [27] used the Scale-Invariant Feature Transform (SIFT) method to find the local interesting points, and the appearance and structure information around the points were used to find the similar regions.

Component-based matching. Component-based approaches are also used to address the general problems. These kinds of methods decompose the complex pedestrian structure into several parts and make it easier to address their training and matching steps. Mohan *et al.* [28] proposed to separate the human body into four parts: face, leg, right arm and left arm. And detect them separately by feature

matching. Ramanan *et al.* [29] also used several parts of people (torso, arm, leg and head) to detect people, but alternatively, he focused on statistical point of view by collecting a dense of patches for each part and generate hypothesis model.

Multimodal feature matching. Nowadays, more works are investigating on incorporating texture, color, structure, statistical features and descriptive features together to explore the best combination of the features. Wojek *et al.* [30] proposed to use Shapelets and HOG to descriptor pedestrian and the output was better than any single feature. Local structure, HOG and model based approaches are proposed by Xu *et al.* [31] to efficiently detect sudden crossing pedestrians.

3.3 Pedestrian tracking

After pedestrian classification and registration, the next step is to infer trajectory information for the target pedestrians. Although extensive work has been done in recent years, its still a challenge task to track the pedestrian accurately throughout the video by monocular cameras. Some researchers predict pedestrian location in videos as frame-by-frame matching and detection. We call it tracking-by-detection. Another popular way is use geometry and dynamics for location prediction.

Tracking-by-detection. Tracking-by-detection is based on pedestrian detection in each frame and calculates the distance between newly detected pedestrian and pedestrian in previous frames. Ramanan *et al.* [29] proposed to find several pre-defined body parts for each frame and collect together for clustering. Each cluster was used as training samples for specific model construction. The pedestrian location was determined when all parts can meet feature and structure matching criteria. Wu *et al.* [26, 32] proposed to use Bayesian framework, by combing appearance similarity measurement and probabilistic inference to decide whether two responses from two frames belong to the same person.

Dynamic tracking. Dynamic tracking is a kind of tracking method that predicts moving trajectories based on position measurement, motion learning and prediction. Mean-shift, Kalman filter and particle filter are the commonly used motion prediction methods in tracking. Meuter *et al.* [33] proposed to use unscented Kalman filter, which is a kind of non-linear Kalman filter to project the movement of vehicle and pedestrian. Pedestrian location and walking speed were updating frame by frame to realize location prediction.

3.4 Unique challenges of our research

Most of the existing pedestrian tracking algorithms are designed based on the public database, which may only include limited scenarios and pedestrian appearance with very small amount of data. But naturalistic driving data has all the situations we may meet in our daily life, unexpected scenarios and driving environments will be shown in our database. These methods were not designed to work on such diverse backgrounds and large-scale data sets. In this research, we propose feature and motion based method to robustly track pedestrians at different scenarios.

We introduce the previous work of each steps of pedestrian tracking. In the ROI searching step, we introduce moving objects detection and feature detection. This step is aimed to find the candidate pedestrian location before performing tracking algorithm and thus improve the efficiency. the model/template matching, discriminative feature matching, component-based matching and multimodal feature matching algorithms are introduced in the pedestrian classification/registration step, the performance is improving gradually. Finally, two kinds of tracking methods are introduced based on state-of-art tracking literatures. Our tracking method takes advantage of previous work but more robust and suitable for our naturalistic driving data.

4. PROPOSED TRACKING AND STATUS ANALYSIS METHOD

4.1 Preprocessing

The Preprocessing Module first categorizes the driving environment based on GPS and G-sensor information. We will separate the videos with different weather conditions, driving environments, pedestrian appearance probability, etc. Then the video with low luminance will be enhanced by using Power-law before pedestrian detection.

4.2 Feature and motion based pedestrian tracking

The commonly used three stage pedestrian tracking method is also adopted in our algorithm, which include ROI search, pedestrian registration and pedestrian tracking. However, since naturalistic driving data shares more complex background, more various pedestrian appearance and more unpredictable scenarios, we did a lot of improvements at each step to make it more robust and accurate. On one hand, we extracted the most stable features from the pedestrian and weaken the influence of background and moving limbs. On the other hand, we take use of both feature matching and motion learning method to provide better outputs.

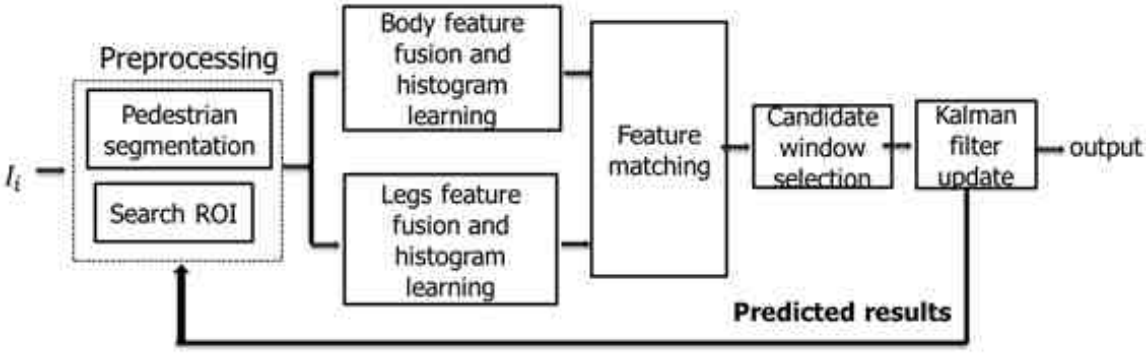

Figure 4-1 Proposed pedestrian tracking algorithm

Fig. 4.1. Proposed pedestrian tracking algorithm

Figure 4.1 shows the diagram of the pedestrian tracking module, which includes: 1) Pedestrian segmentation. We separate pedestrian into three parts (head, body and legs) by using the Anatomical Properties [34]. 2) Search region of interest. Fast video leap segmentation method [35, 36] is used here to quickly find the candidate pedestrian location in the consecutive frames. 3) Feature fusion. We use the covariance matrix method [37, 38] to fuse multiple features of the pedestrian. 4) Histogram learning. This step is a supplement of feature covariance representation to overcome its disadvantages. 5) Feature matching. We compare the feature of pedestrian with feature of environment to determine if the candidate region is a pedestrian or not. 6) Window selection. Several candidate windows will be selected after previous steps. Two window selection methods will be discussed to determine the final pedestrian location by feature matching algorithm. 7) Kalman Filter learning and update. This step draw the motion pattern of the pedestrian, help to check the accuracy of feature matching results, determine pedestrian size and also predict motion model.

4.2.1 Analysis of pedestrian location in videos

Before designing pedestrian tracking algorithm, the transformation between pedestrian location on ground plane and image plane should be investigated. We grouped the scenarios into two classes to better understand the transformations in different cases: 1) subject vehicle is moving straight. Pedestrian is walking, running, playing or standing in the right or left side of the vehicle and his or her moving direction is parallel to the vehicle. 2) Subject vehicle is turning right or left. Pedestrian is walking, running, playing or standing on the right or left side of the vehicle and moving either along the traffic way or cross the street.

Figure 4.2 to Figure 4.5 illustrates some of the scenarios. For Figure 4.2, the pedestrian is moving along the traffic way and vehicle is moving straightly. The corresponding pedestrian location change in the video frames is shown in the bottom right image. As shown in the figure, pedestrian location will move from middle of the frame to the right if the pedestrian is on the right side of the vehicle. Pedestrian size will increase as their distance become shorter. If the pedestrians initial location is on the opposite side, say the left side of the vehicle, we can infer that the corresponding pedestrian location change should be exactly the mirror inverse of Figure 4.2. Figure 4.3 illustrates when the pedestrian is walking across the street and vehicle is stopping. Pedestrian will move from right to left in the frame sequences as we expected. Figure 4.4 shows the scenario when the vehicle is turning left and pedestrian is walking on the right side of the vehicle or just standing at the right corner of the intersection. For this situation, the corresponding pedestrian locations in each frame are shown as the bottom right image of Figure 4.4. The pedestrian will firstly on the very left side of the frame, and as the vehicle turning left, pedestrian location will go to the right side of the frame. Figure 4.5 shows when the vehicle is turning right and pedestrian is going to cross the street from the left side of the vehicle or just standing

on the left corner of the intersection. From the corresponding location distribution, we can find that the motion pattern is similar to the case in Figure 4.4 and just the difference in moving direction.

Fig. 4.2. Vehicle is driving straight and pedestrian is walking along traffic way

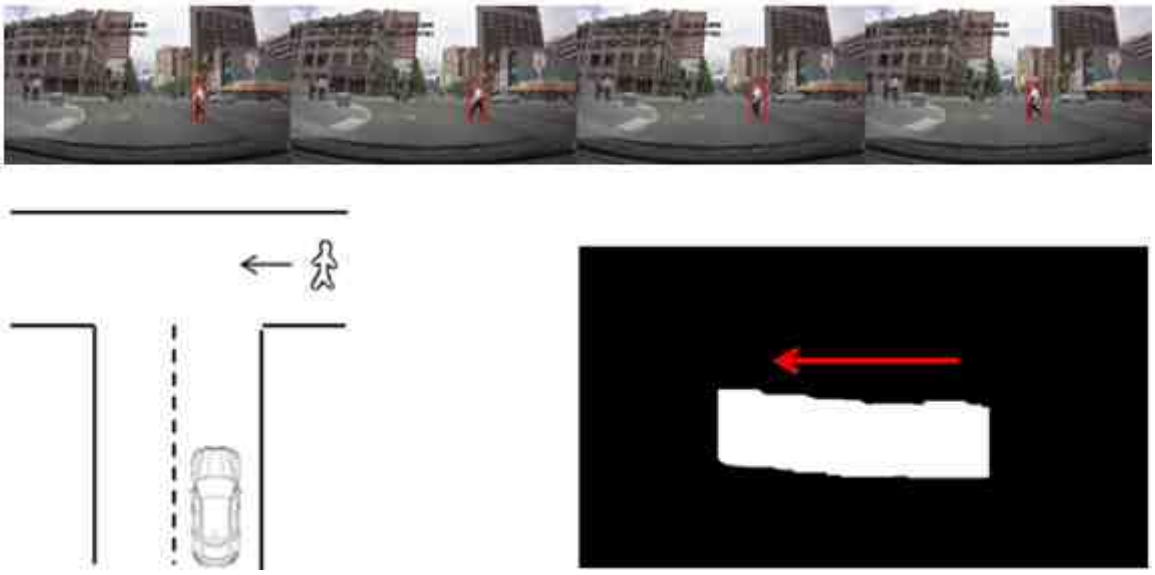

Fig. 4.3. Vehicle is stopping and pedestrian is crossing the road

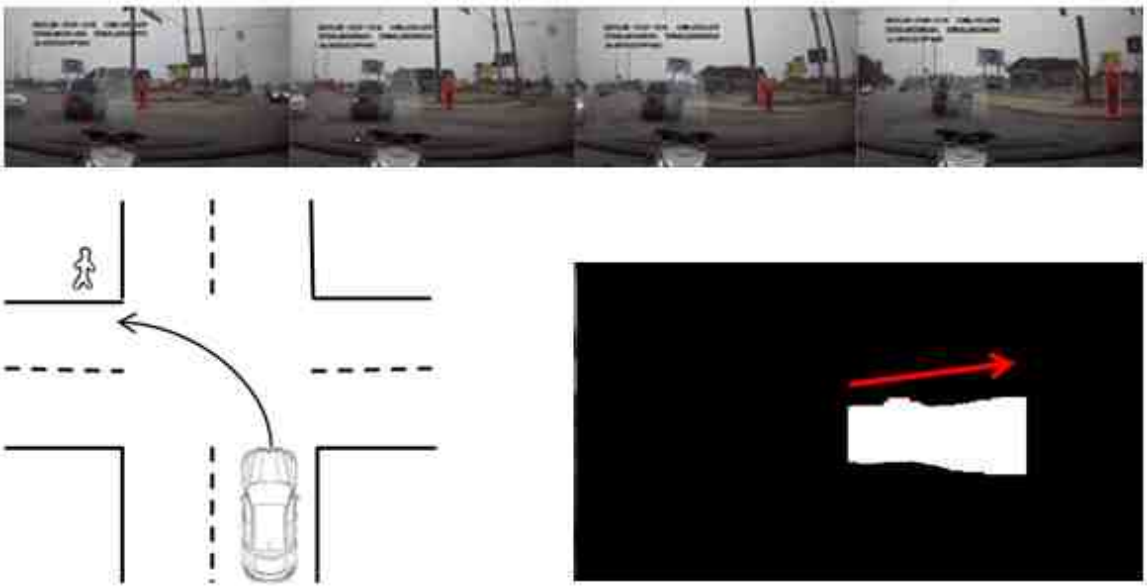

Fig. 4.4. Vehicle is turning left and pedestrian is on the right side of vehicle

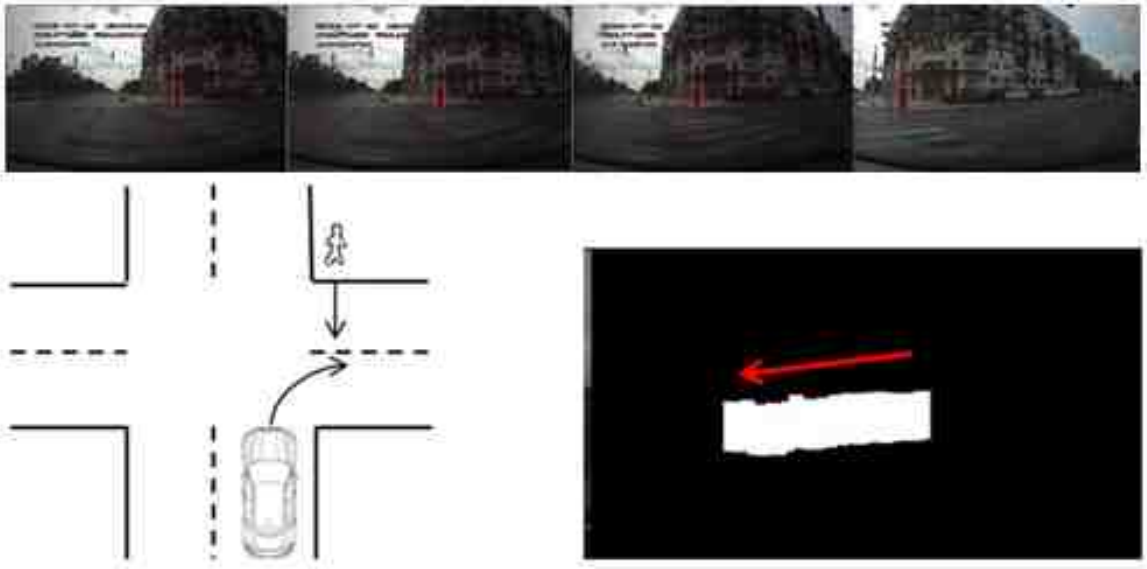

Fig. 4.5. Vehicle is turning right and pedestrian is on the right side of vehicle

4.2.2 Pedestrian segmentation

Part-based pedestrian learning method has proven its superiority in several literatures Andreas *et al.* proposed a two stage part-based pedestrian method and found that false positive were lower than whole body based method [39]. Andrei *et al.* proposed part-based pedestrian detection method with edge and orientation features to handle difficult database [40]. This kind of methods construct templates or feature vectors for each part separately and detect pedestrian in an unknown frame by combing several detectors. The outcomes are always better than just learning the whole pedestrian. However, the problem for implying existing part-based tracking technique on our naturalistic driving data is that the human parts are not guaranteed to be observable or detectable, which will lead to many mistakes. One way to overcome this problem is to search the most constant parts from the pedestrian and do feature matching, while weaken or ignore the inconstant parts. We take advantage of Anatomical Properties to separate the pedestrians into three parts as shown in Figure 4.6 (a): head, which covers top 13% of the total height of the person, body which covers 39% of the total height, and legs which covers 48% of the total height. The red line in Figure 4.6(b) shows the anatomical properties implemented in our naturalistic driving data. This may not be exactly accurate for all cases. However, as shown in the example, its an efficient way to separate the three parts in general situations. Body part is a more consistent part of the whole body across consecutive frames that we can use to perform feature matching.

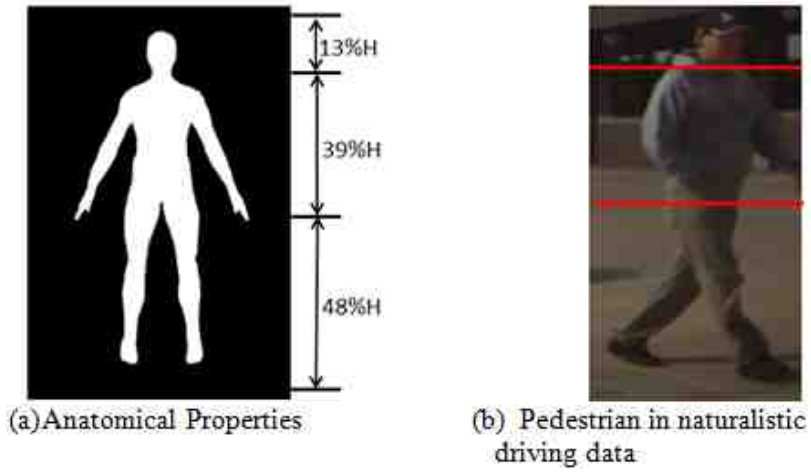

Fig. 4.6. Sample pedestrian appearance model

In order to minimize the influence of background and random moving arms, we also used a one-dimensional Gaussian filter on the body region to emphasize the middle area while weaken the edge area. Figure 4.7 shows some examples of the pedestrian in our naturalistic driving data. Body is the most consistent part either in color or structure.

Fig. 4.7. Sample pedestrian in our naturalistic data

The appearance of legs may change nonlinearly for a moving pedestrian. Its quite challenging to match legs. However, legs may provide some useful information if we get confused by just matching body region. On one hand, we match the color information from the legs or trousers as a prerequisite for region of interest searching.

On the other hand, we attempt to learn periodic appearance model for legs. In this research, we use an adaptive motion model to represent the legs. Let the current set of legs appearance models to be S, which include all the previous models we learned. After we get our tracking results at time t, we will first calculate the distance between current observation A and all the previous models in S to find the min distance. If we can find that the min distance is less than a threshold, we will update the appearance model set S with our new observation. Otherwise, a new appearance model will be added to S. Figure 4.8 shows the appearance variation of the legs part of a pedestrian we tracked.

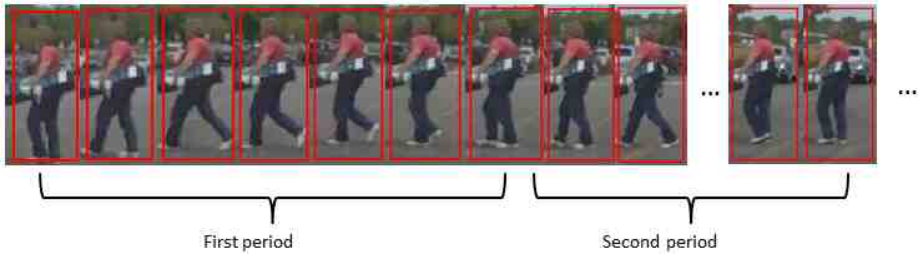

Fig. 4.8. Periodic appearance of legs

4.2.3 Search ROI

Region of interest (ROI) predetermination can help to get rid of unnecessary regions for searching as we mentioned before. At present, we extracted the cloth color information and try to extend out to find the whole cloth region in the frames. We innovatively used the fast video leap segmentation method in this step to help us efficiently find out the candidate regions.

Fast video leap segmentation

Fast video leap segmentation is a kind of method that finds similar regions based on color and structure information. The final segmentation is the groups of pixels that satisfy specific special and Chroma constraints. In particular, the related pixel

sets P1 and P2, also called as equivalent regions between two images, are recognized iff: 1) P1 and P2 are Chroma similar. 2) P1 and P2 are adjacent based on some special threshold T but not necessary contiguous (i.e. they can be neighbors in a T*T window centered on P1 or P2). The definition of Chroma similar in color images are based on evaluating the maximum difference between two pixels in three channels as:

$$max(|R_1 - R_2|\,,\ |G_1 - G_2|\,,\ |B_1 - B_2|\,) \leq\ \alpha \tag{4.1}$$

α is a threshold which may differ for different datasets.

If we define all the pixels in a region with 2*2 tiles with color information (RGB), order information (pixel ID) and structure information (region size, location and shape). The steps to find the similar regions are:

1. Search exactly in the same location of the initial segment. We extract the features in this new region to compare with the previous region features. If the result cannot meet our requirement, then:

2. We will widen the search window to include all the tiles containing the pixels segmented in the previous frame. The same criteria for this region. If match is still not available, then:

3. The search will widened again to include the neighbor tiles for matching until matching is obtained.

Leap segmentation for searching ROI

Although in consecutive frames the pedestrian location may not change dramatically, we have no idea about the pedestrian status, direction or speed. Leap segmentation method can help find the interesting regions more efficient than sliding search window randomly around the previous location. In our implementation, we initially learn the pixel information in the first frame and quickly exploit the similar regions in the consecutive frames by the steps we discussed in previous section.

Figure 4.9 shows the results by implementing video leap segmentation method on our dataset. Figure 4.10 shows the matching results for a frame sequence in the video as Figure 4.9. We can find that if the color of pedestrian cloth is obvious and distinctive from the background, leap method is good enough to track the pedestrian. However, with unknown environment information and pedestrian color information, leap segmentation is not a reliable method for pedestrian tracking. If the cloth color of the pedestrian is very similar to the background, or say, by using equation 4.1 a large area will be segmented, we cannot determine the exact pedestrian location. Thus, other features should be used to track the pedestrian. We use leap segmentation method in the first step because of its time efficiency and give us a rough idea about where the pedestrian should be.

Fig. 4.9. Leap segmentation results for a sample frame sequence

Fig. 4.10. Leap matching results

4.2.4 Feature matching scheme

As mentioned before, the body part is more consistent and reliable for feature information among consecutive video frames. Just searching the corresponding body part in the consecutive frames may be good enough if the pedestrian is not obscured and the body area provides distinct feature information, like edges and color. However, if the pedestrian is far away or body part gives low contrast with the background, it may fail in finding the most similar regions. To overcome this problem, we also use surrounding patches within radius R instead of learning the pedestrian alone. As shown in Figure 4.11, the body part is labeled as foreground. The surrounding patches of the environment with the same size are labeled as background. The number of background patches can depend on the complexity of the environment. All the features and labels of the patches are combined as training data and will be used to find the region in the subsequent frame with shortest distance in feature matching.

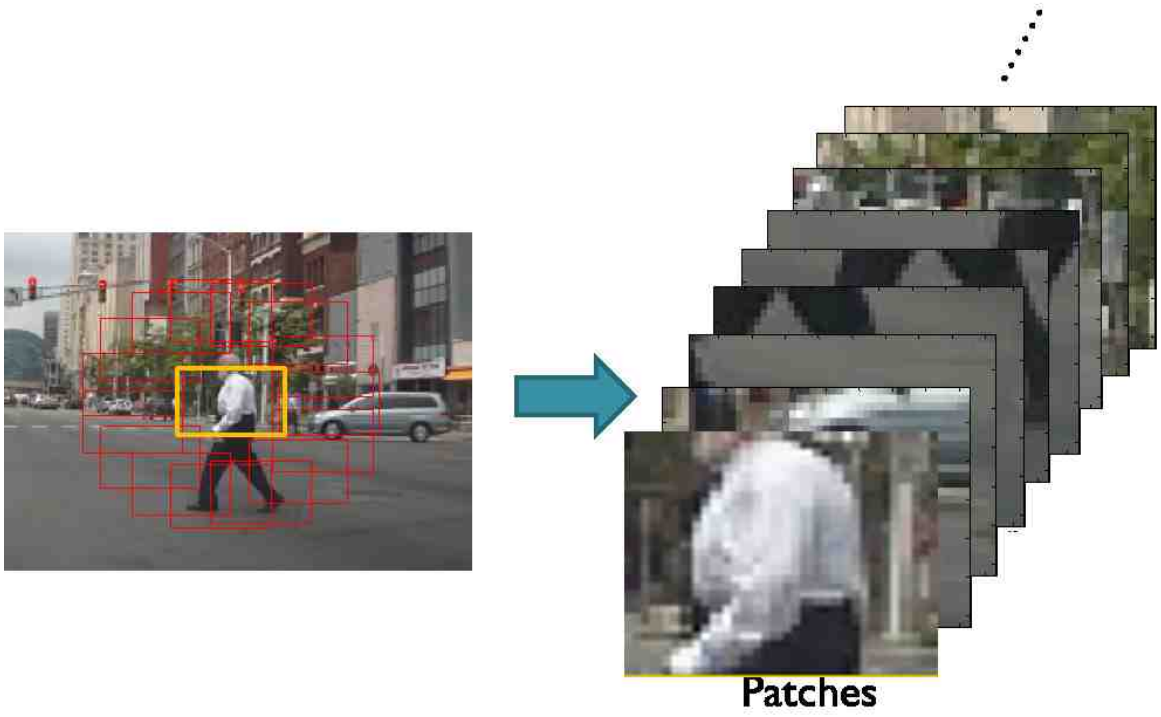

Fig. 4.11. Example patches used for background analysis (note the patches will be more than what is shown in the example)

The frame rate is 30 per second and we need to extract the pedestrian location in each frame. Pedestrian location wont change dramatically in that short period of time if pedestrian speed and vehicle speed are normal. We try to find an optimal radius to make sure the pedestrian location in the next frame will fall in the range of the environment patches we selected.

However, if the pedestrian or vehicle is moving very fast, using the same radius as slow moving scenarios may not be a wise choice. We use adaptive radius for environment patches selection for this kind of scenarios. On one hand, radius is adapted with the size change of pedestrian, shaper changes of pedestrian size will lead to shaper changes of radius. On the other hand, pedestrian and vehicle speed are made into consideration. Larger radius will be adopted for fast moving scenarios.

The matching scheme we proposed in this thesis is based on comparing the candidate region with the previous body patch and all the environment patches. If the distance between current region and previous body patch belong to the smallest (threshold determined by experiment) distances, we say this region should be the candidate new body location. To determine if current window is the target pedestrian, we will use following criterion:

$$patch = \begin{cases} 1, & D_{p-B} < min_{\beta_{D_{p-E}}} \\ 0, & else \end{cases} \tag{4.2}$$

where D_{p-B} is the distance between current patch and previous body patch, $min_{\beta_{D_{p-E}}}$ is the th minimal distance between current patch and previous environment patches. As shown in Figure 4.12, the window in (b) will compare with all the selected patches in previous frame (a), the resulting distances are indicated on each patch. If the distance to the previous body patch can meet our criterion, the window will be selected.

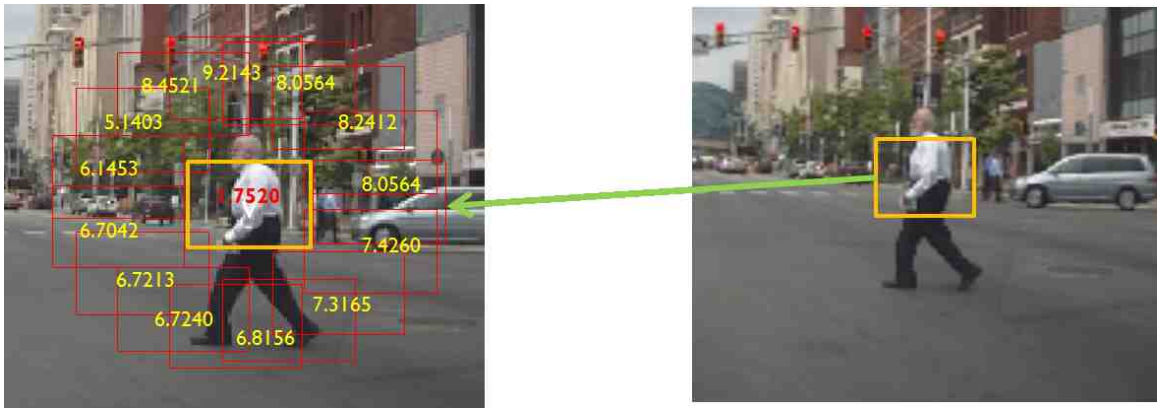

Fig. 4.12. Compare a new window with patches in previous frame

Usually, more than one candidate windows are left after this feature matching step and can hardly provide us the final pedestrian location. Two window selection methods will be introduced in the following sections.

4.2.5 Feature fusion by covariance matrix

Covariance matrix feature representation has attracted a lot of attentions since 2006. Several works have been done by using covariance matrix, such as object recognition (Tuzel *et al.* [38]), covariance tracking (Wu *et al.* [41]) and action recognition (Guo *et al.* [42]). The advantage of covariance matrix is it provides a natural way to fuse high-dimensional feature vectors which maybe correlated. It use only (d2+d)/2 different values to represent the high dimensional features and also reduce the influence of noise in the process of calculating the covariance. We use covariance matrix to combine color and structure feature together for matching.

Feature representation

Given a region R with m*n number of pixels, and its feature mapping F, which,

$$F\left(x,y\right)=M(R,x,y) \tag{4.3}$$

where (x,y) is the location of each pixel. They are directly associated with the feature vectors. M is the feature mapping function and can include color, magnitude and orientation, etc. The number of features in M corresponds to the dimension of F, denoted by d.

Note that its not true that more features will get better results or vice versa. The optimal feature combination we found for our datasets are:

$$F\left(x,y\right)=\left(x,y,\ I\left(x,y\right),I_{x}\left(x,y\right),I_{y}\left(x,y\right),U\left(x,y\right),\ V\left(x,y\right)\right)^{T} \tag{4.4}$$

Here x, y denote the coordinates of the pixel. They are directly associated with the feature vectors. $I(x,y)$ denotes the pixel intensity in grayscale at (x,y). I_x and I_y are the gradient in the horizontal and vertical directions respectively. Gradient will be calculated for three channels and the channel with largest norm will be selected

for that pixel. The gradients give us the edge and structure information. U and V provide the color information derived from the LUV color model since it provides more constant color information than the RGB model under different illumination. Thus for each pixel, we collected the location, structure and color information.

Figure 4.13 shows the feature representation architecture. We use large red boxes on the image to represent each pixel for visualization.

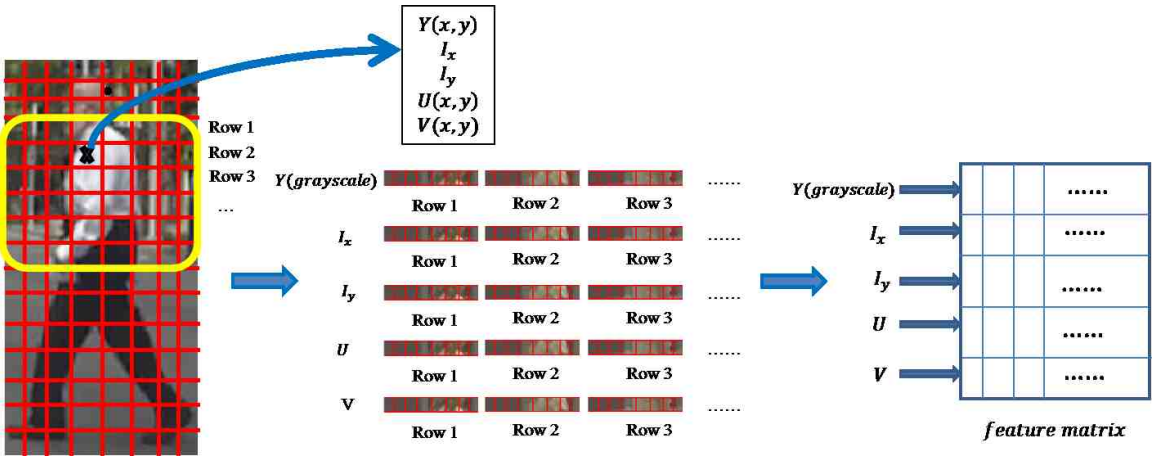

Fig. 4.13. Feature representation

Within,

- $Y(x,y) = 0.299 * R(x,y) + 0.587 * G(x,y) + 0.114 * B(x,y)$;
- $\frac{dI(x,y)}{x} = I(x+1,y) - I(x-1,y)$;
- $\frac{dI(x,y)}{y} = I(x,y+1) - I(x,y-1)$;
- $U(x,y) = -0.14713 * R(x,y) - 0.28886 * G(x,y) + 0.436 * B(x,y)$;
- $\mathrm{V}(x,y) = 0.615 * R(x,y) - 0.51499 * G(x,y) - 0.10001 * B(x,y)$;

For the constructed $m * n * d$ feature matrix, we calculate the covariance matrix as:

$$C = \frac{1}{m*n} \sum_{k=1}^{mn} (f_k - \mu_r)(f_k - \mu_r)^T \tag{4.5}$$

where (μ_r) represents the mean vector of the corresponding feature and f_k is the d dimensional feature vectors inside the region. In our experiment, the result of covariance matrix is a 5*5 matrix, the diagonal of the entries are the variance of each feature and the non-diagonal entries are the correlation between the features.

Distance measurement

The next step is to measure the distance between two covariance matrixes. Covariance matrix does not lie in Euclidean space, but we can make it lie in vector space by using log-covariance matrices. Thus we use the Log-Euclidean Riemannian Matric (LEARM) [43] proposed by Arsigny et al. [44] to calculate their distance:

$$D\left(C_i, C_j\right) = \left\|\log\left(C_i\right) \ - \log\left(C_j\right) \ \right\|_2 \tag{4.6}$$

As the size and appearance of moving pedestrian may vary overtime, its necessary to adapt these changes. The fastest way to update these changes is to find the mean value of selected previous states. At time T, we calculate the covariance matrix C_k and accumulate the entire previous covariance matrix in the Riemannian space:

$$C = \exp\left(\sum_{k=1}^{T} \log\left(C_k\right) \right) \tag{4.7}$$

4.2.6 Local histogram learning

It may not be sufficient to use covariance matrix alone in our real life implementation as it only represents the variance and correlation between each feature, the results cannot be guaranteed if no specific features are provided. Several kinds of situations will lead to very small covariance distance, such as large area overlapping, small feature correlation distance, etc. Figure 4.14 shows the situation that two patches with nearly 50% similar area lead to small covariance distance.

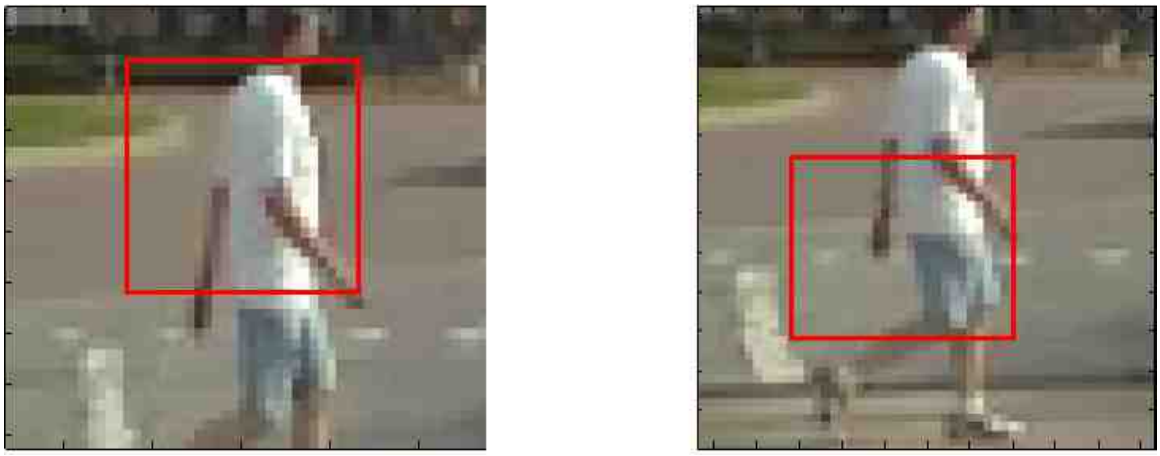

Fig. 4.14. Two regions with small covariance distance but not similar

To overcome this problem, we propose to use local histogram as an additional matching criterion for two patches. The local histogram takes advantage of summarizing local color information of an image and concatenating together to represent the whole image. This method is originally used in face recognition and has proven its good performance [45, 46].

We first normalize the patches to the size in previous frame, then separate them into 8 by 8 blocks and calculate the histogram of each block, the final image histogram is combing the sub histograms together (as shown in Figure 4.15). The similarity of two color histogram is calculated by using chi-square equation [47]:

$$d\left(h_1,\ h_2\right) = \sqrt{1/2 \sum_{m=1}^{k} \frac{\left[h_1\left(m\right) - h_2\left(m\right)\right]^2}{h_1\left(m\right) + h_2\left(m\right)}} \tag{4.8}$$

where h_1,h_2 are two histograms respectively.

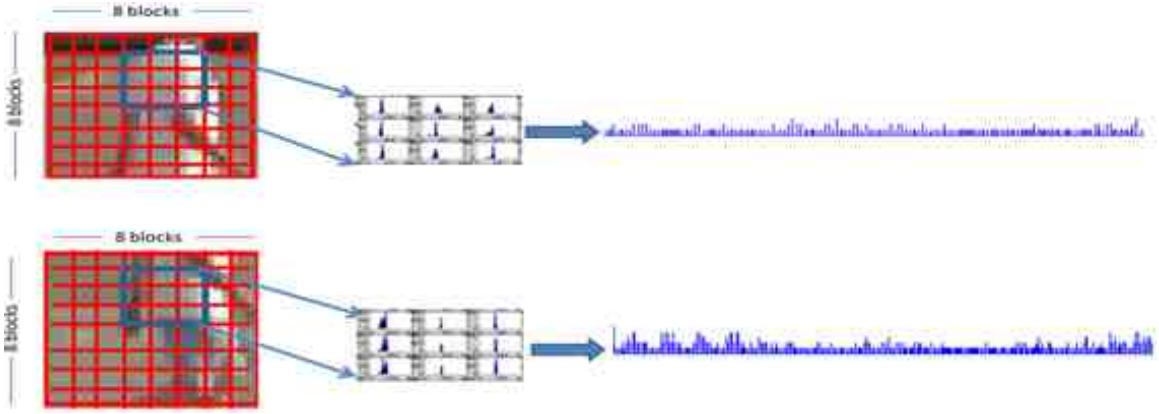

Fig. 4.15. Local histogram of two regions

We have done the feature matching steps. Unfortunately, more than one patch will be left after these efforts. We need to go further to determine the final pedestrian location. The pedestrian tracking algorithm needs to solve two problems: 1) if the selected regions are representing a pedestrian or a non-pedestrian. 2) if the selected regions are representing the same pedestrian. We have solved the second problem in the previous sections, the next is to verify the human structure within the patches. Figure 4.16 gives us an example of how the left patches look like.

Fig. 4.16. Sample patches selected

4.2.7 Window selection with association of HOG

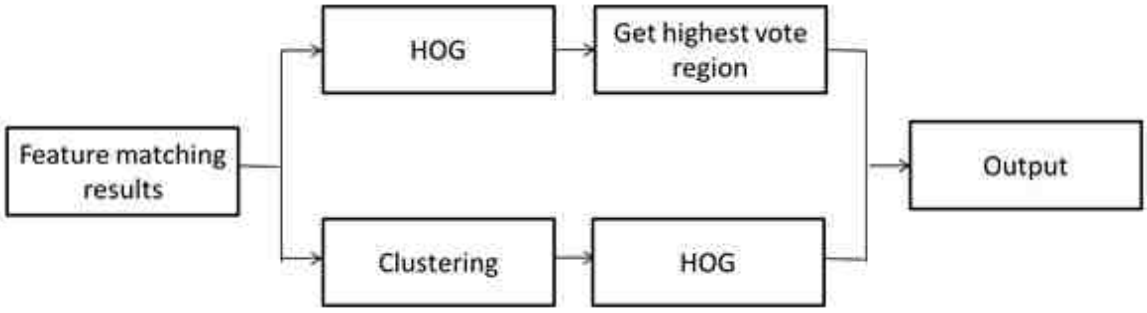

Fig. 4.17. Location selection scheme

Figure 4.17 shows how we determine the final pedestrian location based on the selected patches. We can find two kinds of results after the feature matching: 1) a bunch of windows are surrounding the target pedestrian we are trying to track. 2) more than one group of windows are surrounding several locations, which means some other locations are very similar to the pedestrian. Before demonstrating two methods to deal with the results, we will first introduce some more details about HOG descriptor.

Histogram of Oriented Gradient (HOG)

Histogram of Oriented Gradient (HOG) is firstly proposed by Dalal *et al.* [25] and is popularly used in pedestrian detection area. The idea of HOG is that people can be well characterized by the distribution of local intensity gradient and edge directions. HOG use histogram of gradient and orientation as the robust feature to represent people in an image. Given an image, HOG can be calculated by following steps: compute gradient, weighted into special and orientation cells, contrast normalization over overlapping spatial blocks and collect HOG over the detection window.

- Compute gradient of the original image. The magnitude and orientations can be calculated by:

$$magnitude: \quad mag = \sqrt{g_x^2 + g_y^2} \tag{4.9}$$

$$orientation: \quad angle = \arctan\left(\frac{g_y}{g_x}\right) \tag{4.10}$$

g_y and g_x are gradients in y and x directions respectively, with:

$$g_x = I\left(x+1,\ y\right) - I\left(x-1, y\right) \tag{4.11}$$

$$g_y = I\left(x,\ y+1\right) - I\left(x, y-1\right) \tag{4.12}$$

- Normalize the image to be 128*64
- Divide the image into several 16*16 blocks with overlap ratio 50%. The overlapping is to ensure consistency across the image (Figure 4.18 (a)).
- For each block, we will calculate 256 magnitude and orientations. 64 of them will be assigned to each cell.
- Quantize the orientation into 6 bins, so the correlation between the orientations and bins would be as Figure 4.18(b).

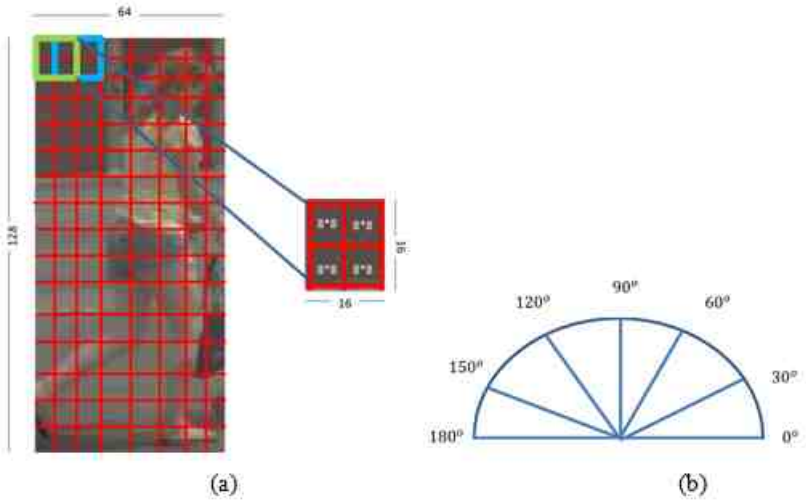

Fig. 4.18. Image blocks and bins

- To get the histogram of each block, we will calculate the voted histogram of each cell separately. 64 orientations in the cells mean 64 votes for each 6 bins, and the votes will be their magnitude with Gaussian weight. The concatenated histogram will be the final block histogram.

- The block histogram will be normalized before constructing the orientation histogram of the whole image. L2-norm is used in this experiment (is a very small value to be non-zero):

$$v = \frac{v}{\sqrt{||v||_2^2 + \tau^2}} \tag{4.13}$$

- Finally, all the vectors computed from the blocks are concatenated to represent the whole image as Figure 4.19. From our specified parameters during the process, the dimension of HOG descriptor is 2520. The 2520 dimensional HOG descriptors are trained by linear support vector machine for classification.

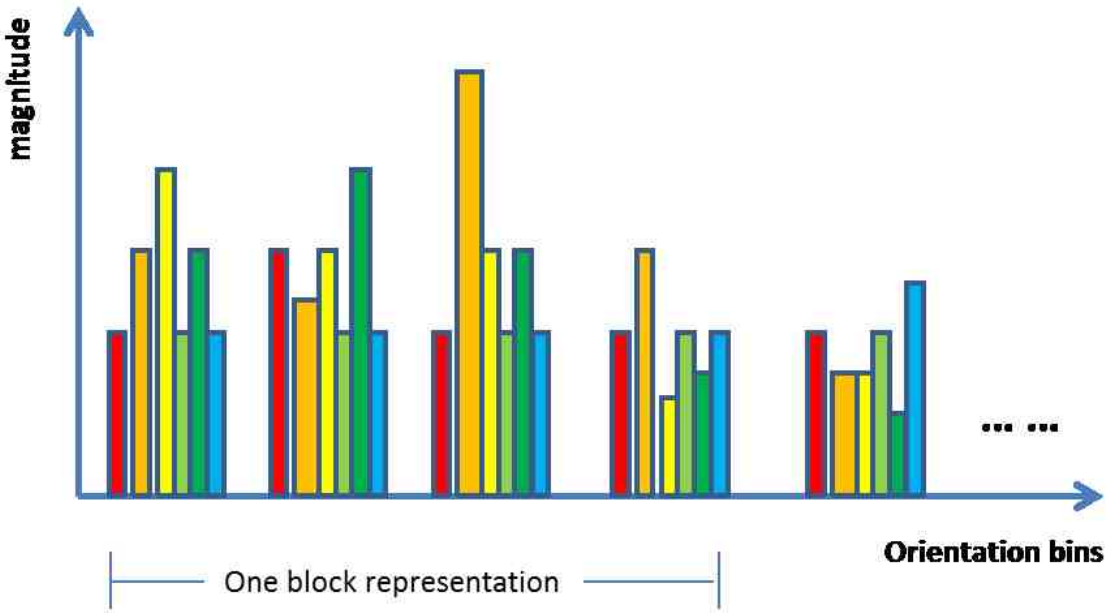

Fig. 4.19. HOG descriptor

Window selection by finding the region with highest vote

In this section, we discuss the feasibility of window selection by finding the highest vote region. After the previous feature matching step, we still have several candidate patches that meet our requirements. As shown in Figure 4.16, the blue windows on the frame are the candidate patches we selected (only indicated the body part). For this case, the simplest way is to give the same weight for each candidate window and vote for the entire region they located in. The region with the highest vote should be used as the central part of the pedestrian body.

However, not all the candidate patches are eligible for voting. Some patches are definitely not representing human but they share similar features as we get from the pedestrian in the previous frame. They also hold one vote to decide the pedestrian location and will affect the accuracy. To deal with this problem, we use HOG pedes-

trian detection with lower threshold to delete some non-pedestrian patches before we use the voting method. By using HOG descriptor, we can get rid of these patches to minimize their influence.

Window selection by clustering the candidate patches

Not all situations are suitable with voting method due to: 1) some background may share similar features as the pedestrian and attracts as many attentions as pedestrian. 2) Some mistaken windows sift the good result. Figure 4.20 shows an example for this situation. Obviously two groups of candidate patches in this example and they focus on two different regions separately. The upper groups of windows only have two members but their voting will make the tracked pedestrian location higher than the exact location. We use clustering method to separate them into two groups and find two potential pedestrian locations based on these groups. The final location will be determined by comparing their similarity to the previous pedestrian we tracked.

Fig. 4.20. Grouped candidate windows

4.2.8 Motion learning

We propose to use a Kalman filter for motion learning. Kalman filter is the optimal minimum mean squared error(MMSE) estimator and has been widely used in object tracking issues [48, 49]. Since real life scenarios can hardly be projected in linear functions, we prefer to use the unscented Kalman filter for our motion learning. The unscented Kalman filter is an extension of the linear Kalman filter based on the scaled unscented transformation [50]. It shows a superior performance in our pedestrian tracking problems compare to the linear Kalman filter.

Kalman filter can measure the movement based on learning the previous moving patterns and give us a prediction of current status. We suppose to use Kalman filter for three reasons: 1) check our feature matching results 2) adapt the size change of pedestrian in video frames efficiently 3) the motion model analyzed here will be used as part of behavior analysis system. It is not easy to adapt the size change of the pedestrian in the video clips just by feature matching method, and its kind of time consuming even if we can adjust it by searching each human structure time by time after each tracking cycle. However, the projection algorithm in Kalman filter provides us an efficient way to measure the size of pedestrian at the same time of calculating their speed and location. Here we will introduce the idea of Kalman filter and its implementation in our system.

Kalman filter update

The Kalman filter cycle is shown as Figure 4.21.

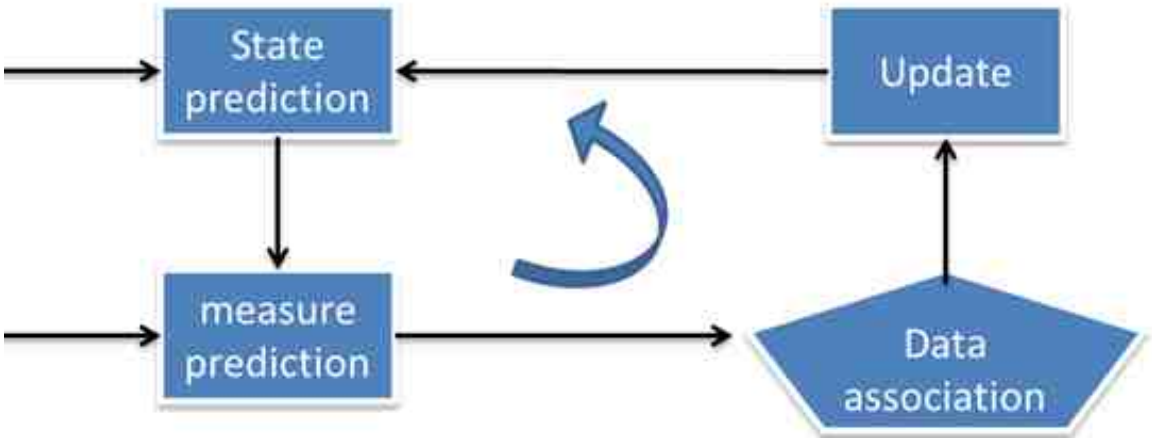

Fig. 4.21. Kalman filter cycle

Kalman filter deals with the discrete-time controlled process that described by a linear system as:

$$\hat{x}\left(k+1 \mid k\right) = F\hat{x}\left(k \mid k\right) + Q \tag{4.14}$$

$$\hat{p}\left(k+1 \mid k\right) = F\hat{p}\left(k \mid k\right) F^{T} \tag{4.15}$$

where $\hat{x}\left(k \mid k\right)$ is the current state. F is the transform function which predicts next state from previous state. $\hat{p}\left(k \mid k\right)$ is current covariance matrix. Q represents the process noise and assumed to be white and normal probability distribution. Suppose we want to model a straight walking pedestrian in the video, $\hat{x}\left(k \mid k\right)$ will provide us all the pedestrian information: location, speed and acceleration if he is not in constant speed. F will provide us the mathematical way to calculate the next pedestrian state.

In most cases, we cannot observe all the parameters in $\hat{x}\,(k \mid k)$. Instead of measure $\hat{x}\,(k \mid k)$ directly, we develop a measurement vector $\hat{z}$:

$$\hat{z}\,(\mathrm{k}) = \mathrm{H}\,(\mathrm{k})\,\hat{x}\,(k+1 \mid k) + E \tag{4.16}$$

where $H(k)$ helps to convert the state parameters in $\hat{x}\,(k \mid k)$ to measurable and observable vectors. The random vector E is the measurement noise and also to be Gaussian normal distribution. By error propagation law, we can infer the measurement covariance s as:

$$s_{k+1} = H\,(k)\,\hat{p}\,(k+1 \mid k)\,H^{T}\,(k) \tag{4.17}$$

In order to derive the Kalman filter update equations, we try to find a relationship between the posterior state estimation $\hat{x}\,(k+1 \mid k)$, prior estimation $\hat{x}\,(k \mid k)$, measurement prediction $hatz\,(\mathrm{k})$ and actual observation $z\,(\mathrm{k})$. Thus, we define the justification equation as:

$$\hat{x}\,(k+1 \mid k+1) = \hat{x}\,(k+1 \mid k) + K\,(k)\,v \tag{4.18}$$

where $K(k)$ is a weight and v is called measurement residual and defined as:

$$v = z\,(k) - \hat{z}\,(k) \tag{4.19}$$

This equation helps us to predict the state from the previous state and the difference between our observation and mathematically calculated prediction. The goal for Kalman filter is to find the optimize weight for this equation so to minimize the error propagation. In order to minimize the posteriori error covariance, we combine the previous equations and calculate optimize $K(k)$ by the following relationship:

$$\min\left(E\left[\left|x\,(k) - \hat{x}\,(k|k)\right|^{2}\right]\right) = \min\left(tr(\,\hat{p}\,(k \mid k))\right) \Rightarrow \frac{\partial\; tr(\hat{p}\,(k \mid k))}{\partial k_{k}} = 0 \tag{4.20}$$

which as a result, $K(k)$ is in the form of:

$$K\left(k\right)=\hat{p}\left(k+1\mid k\right)H^{T}\left(k\right)s(k)^{-1} \tag{4.21}$$

The updated covariance $\hat{p}\left(k+1\mid k+1\right)$ is:

$$\hat{p}\left(k+1\mid k+1\right)=\left(I-k\left(k\right)H\left(k\right)\right)\hat{p}\left(k+1\mid k\right) \tag{4.22}$$

Introduction of Unscented Kalman Filter

The fundamental of unscented transform is that a small set of points with known mean and covariance are good enough to represent the distribution of random variables [51], which make it easier to express the relationship between the datasets in the form of equations. Given a set of random variables x with mean $\overline{x}$ and covariance P_x, we can calculate the substitute points (sigma points) χ_i for a nonlinear function,$z=f(x)$, according to:

$$\chi_0=\overline{x} \tag{4.23}$$

$$\chi_i=\overline{x}-\left(\sqrt{\left(L+\lambda\right)p_x}\right)(i)\qquad i=1,...,L \tag{4.24}$$

$$\chi_i=\overline{x}+\left(\sqrt{\left(L+\lambda\right)p_x}\right)(i)\qquad i=L+1,...,2L \tag{4.25}$$

where λ is a scaling factor by $\lambda=\alpha^2(L+K)-L$. The parameter determines the distribution of sigma points around the mean $\overline{x}$ and is set to a small positive value(ex. $1e-3$). K is the second scaling parameter which is usually set to $k>=0$. $\left(\sqrt{\left(L+\lambda\right)p_x}\right)(i)$ is the i_th row of the matrix square root of $(L+\lambda)P_x$. The weight distributed to each sigma points are calculated by:

$$w_0^{(m)} = \frac{\lambda}{L+\lambda} \tag{4.26}$$

$$w_0^{(c)} = \frac{\lambda}{L+\lambda} + \left(1-\alpha^2\right) \tag{4.27}$$

$$w_i^{(m)} = w_i^{(c)} = \frac{1}{\{2\left(L+\lambda\right)\}} \qquad i = 1, ..., 2L \tag{4.28}$$

Finally the sigma points are propagated through the nonlinear function f to capture the weighted mean and covariance for z:

$$\overline{z} = \sum_{i=0}^{2l} w_i^{(m)} f\left(\chi_i\right) \tag{4.29}$$

$$p_z = \sum_{i=0}^{2l} w_i^{(c)} \left\{f\left(\chi_i\right) - \overline{z}\right\} \left\{f\left(\chi_i\right) - \overline{z}\right\}^T \tag{4.30}$$

Implementation of UKF in our tracking algorithm

In this section, we describe the implementation of unscented Kalman filter in our tracking system. We assume that the road surface and the optical axis of the camera are parallel and the camera will be fixed on the vehicle. The GPS information recorded in real time will be used to analysis the movement of vehicle (i.e. speed and direction). The state vector in our system is given by $x = [x, y, V_c, V_p, h, w]$ with the vehicle velocity V_c, pedestrian velocity V_p , pedestrian location, pedestrian height h and pedestrian width w in image plane.

We project the 3D scene view to 2D image plane based on the pinhole camera model, thus get the equation relationship between the vectors in the way of:

$$r * \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} [R \mid t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4.31}$$

Where (x, y) denotes the image coordinates of the pedestrian, r is the coefficient that converts pixel value to physical value. f is the focus length of the camera, (c_x, c_y) is the center of the image. Matrix $[R|t]$ represents the rotation of the vehicle, i.e. the direction change of the vehicle; it can be calculated from the GPS information. $[X^{'}, Y^{'}, Z^{'}]$ is the world coordinate of the pedestrian. Knowing the time interval t and initial distance between vehicle and pedestrian (we will go in detail of their calculation in the next section), we can get the pedestrian location of world coordinate in the new frame, thus update our image projection. We denote the non-linear projection as: $state_{t|t-1} = f(state_{t-1})$.

The transformation H in our algorithm is used to extract the observable vectors , tracking results will be used as the observation vector z_t, scaling factor λ equals: $\lambda = (\alpha^2 - 1) * L$.

Then:

- Calculate$\chi_i,\ w^{(m)},\ w^{(c)}$ from Equation 4.23 to 4.28
- The weighted mean and covariance of predicted state can be calculated by:

$$\overline{x_{t|t-1}} = \sum_{i=0}^{2l} w_i^{(m)} f\left(\chi_i\right) \tag{4.32}$$

$$p_{\overline{x_{t|t-1}}} = \sum_{i=0}^{2l} w_i^{(c)} \left\{f\left(\chi_i\right) - \overline{x_{t|t-1}}\right\} \left\{f\left(\chi_i\right) - \overline{x_{t|t-1}}\right\}^T \tag{4.33}$$

- The observable vectors and their distribution can be found as:

$$\widehat{z_{t|t-1}} = \sum_{i=0}^{2l} w_i^{(m)} H\left(f\left(\chi_i\right)\right) \tag{4.34}$$

$$p_{zz} = \sum_{i=0}^{2l} w_i^{(c)} \left\{H(f\left(\chi_i\right)) - \widehat{z_{t|t-1}}\right\} \left\{H(f\left(\chi_i\right)) - \widehat{z_{t|t-1}}\right\}^T \tag{4.35}$$

- After these steps, the data association and state update can be accomplished by:

$$p_{xz} = \sum_{i=0}^{2l} w_i^{(c)} \left\{f\left(\chi_i\right) - \overline{x_{t|t-1}}\right\} \left\{H(f\left(\chi_i\right)) - \widehat{z_{t|t-1}}\right\}^T \tag{4.36}$$

$$K_t = p_{xz} p_{zz}^{-1} \tag{4.37}$$

$$\widehat{x_{t|t}} = \overline{x_{t|t-1}} + K_t \left(z_t - \widehat{z_{t|t-1}} \right) \tag{4.38}$$

$$p_{t|t} = p_{\overline{x_{t|t-1}}} - K_t p_{zz} K_t^{-1} \tag{4.39}$$

4.3 Pedestrian status and pre-collision analysis

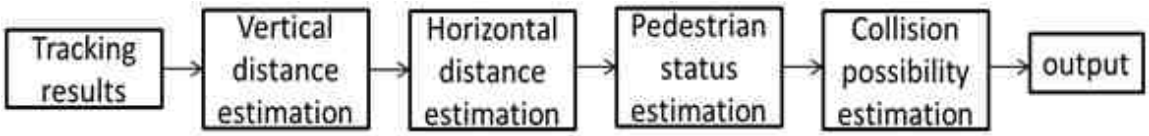

Fig. 4.22. Pedestrian status and pre-collision anlysis diagram

4.3.1 Pedestrian speed and pedestrian-vehicle relationship analysis

Figure 4.22 illustrates the diagram of the status analysis process, which includes Vertical distance estimation Module, Horizontal distance estimation Module, Pedestrian status estimation Module and Collision possibility estimation module.

Fig. 4.23. Pedestrian-vehicle interaction

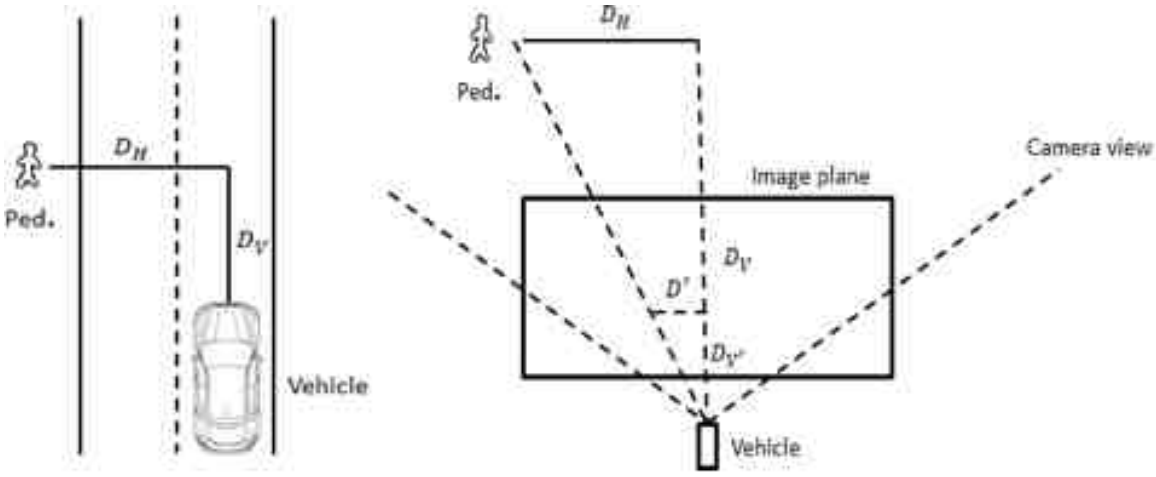

Fig. 4.24. Pedestrian-vehicle distance estimation

Figure 4.23 shows the pedestrian-vehicle interaction in our data. Figure 4.24 shows the schematic diagram of the imaging geometry. Pedestrian status in front of the vehicle can be easily estimated if we know the status of the vehicle, camera parameters and the pedestrian location in each frame. Vehicle speed can be calculated based on the GPS information the camera recorded. After getting the pedestrian size

in two frames and their time interval, we can easily get the distance between the vehicle and pedestrian by solving the two equations:

$$D_V = \frac{D_{V'} * H}{y_1} \tag{4.40}$$

$$D_V - V_c * t = \frac{D_{V'} * H}{y_2} \tag{4.41}$$

where D_v is the distance between the vehicle and pedestrian, $D_{V'}$ is the camera focus length, H is the height of the pedestrian, V_c is the velocity of the vehicle, t is the time interval between the two frames and y_1, y_2 are the height of the pedestrian shown in the pixel domain in two frames respectively.

Once we get the value of D_v, the horizontal distance between the pedestrian and vehicle can be obtained by:

$$\frac{D_V}{D_{V'}} = \frac{D_H}{|x_c - x| * s} \tag{4.42}$$

where s is the ratio between object length and pixel length.

Thus, we can get the relationship between the pedestrian and vehicle. By calculating the pedestrian-vehicle distance with known time interval t, we can estimate the pedestrian status (walking, running or standing) and pedestrian-vehicle relation (in front of vehicle or just in side of vehicle), and if they will hit together or not.

4.3.2 Collision probability projection

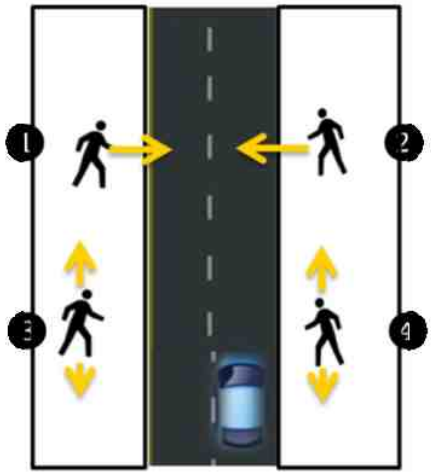

Fig. 4.25. Scenarios for pedestrian pre-collision analysis

Several kinds of scenarios will be discussed in this section to correlate our tracking and status analysis results with the pedestrian pre-collision analysis system. Figure 4.25shows some scenarios in our naturalistic driving data. Four kinds of pedestrians moving direction are presented. Pedestrian 1 and pedestrian 2 are crossing the street from two directions, pedestrian 3 and pedestrian 4 are just walking along the traffic way, either toward the vehicle or backward the vehicle. The collision will happen unless: 1) at the same location and 2) at the same time. We will discuss the feasibility for analyzing potential pedestrian-vehicle crashes based on these scenarios.

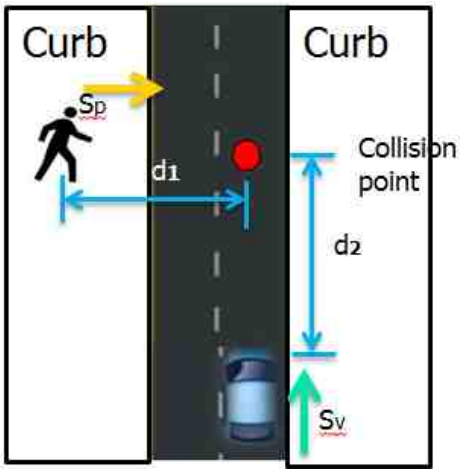

Fig. 4.26. Pedestrian is crossing the street

Figure 4.26 illustrates how we analysis the possibility of collision for pedestrian 1 in Figure 4.25. For this situation, we can get the pedestrian speed, horizontal and vertical distance between pedestrian and vehicle at the beginning based on tracking and status analysis results. Vehicle speed can be extracted from the GPS information and pedestrian direction can be learned from the frame sequence. We suppose the speed of pedestrian and vehicle are constant (no deceleration) and calculate the time for vehicle driving d2 distance and the time for pedestrian walking d1 distance. If they cannot get to the collision point at the same time, no possible collision will happen. Otherwise, the driver should be warned to brake.

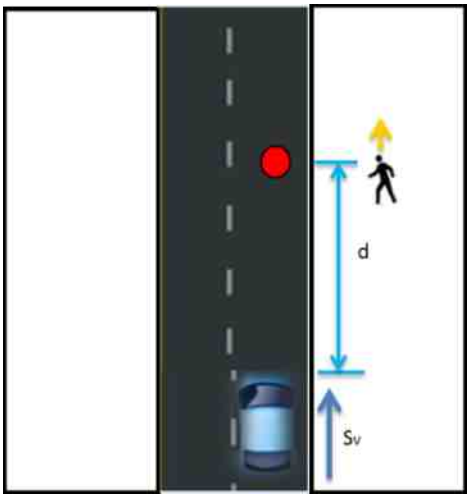

Fig. 4.27. Pedestrian is walking along the street

Figure 4.27 shows another scenario. Pedestrian is walking along the street and vehicle is moving straightly. Initially the vertical distance between the pedestrian and vehicle is d. By tracking the pedestrian in each frame, we can easily get the result that pedestrian is always along the street and horizontal distance almost keep constant. The possibility of collision is very low even if pedestrian is detected nearby the vehicle.

5. EXPERIMENT RESULTS

In the previous chapters, we introduced the overall approach for proposed pedestrian tracking and pedestrian status analysis. Specifically, we used the dynamic feature matching and motion learning method to track the pedestrian. Then, based on the tracking results, we can get the pedestrian motion information, pedestrian-vehicle location information and also the vehicle motion information from the GPS. This information can be used for: 1) pedestrian status estimation. Pedestrian speed in intersections will be calculated to infer if the pedestrian is walking, running or standing. 2) Estimate the possibility of potential conflict. Pedestrian speed, vehicle speed and initial pedestrian-vehicle distance can be used to evaluate if they will hit without warning.

Our performance evaluation was based on our naturalistic driving data. First, for each video sequence, we divided them into 150 frame segments. We specify pedestrian location and pedestrian size in one of the frames and extract pedestrian information from this frame. Completely tracking the pedestrian consists of two separate programs with different projection methods: one tracks the pedestrian forward from the frame we specified to the end; the other tracks the pedestrian backward from the frame we specified to the first frame. All the tracked location information is saved for the pedestrian-vehicle relationship analysis.

In the pedestrian status estimation step, we projected the pedestrian coordinates in image domain to ground plane to learn the distribution of pedestrian speed by collecting a large amount of data for different scenarios and learn the pedestrian status. Also, we find the pedestrian status can be used for potential collision analysis, thus, we extended our work for possibility of collision estimation.

5.1 Pedestrian tracking performance analysis

We evaluated our pedestrian tracking algorithm on the naturalistic driving videos. The details about the data collection process are discussed in the Chapter 2.

In the tracking, we set the color threshold α to be 30 for leap segmentation. Smaller α may help to get a more specific and accurate ROI in many cases, but could also miss some useful areas, especially when the illumination changes. For normal pedestrian size and vehicle speed, we used 30 patches around the specified pedestrian body to represent the environment. For high vehicle speed or close up pedestrians, adaptive numbers of patches are selected, with the overlapping criteria between each of patches not larger than 1/2. We used the paper specified parameters for constructing HOG descriptor to classify pedestrian and non-pedestrian. All the image patches are normalized to 128*64 and separate to 16*16 tiles proportionally.

We tested our pedestrian tracking algorithm on more than 2000 videos with various kinds of scenarios and driving environments. The performance evaluation is based on comparing the spatial overlapping rate between tracking output and the ground truth. Overlapping rate is calculated by finding the common area between tracked window and ground truth window, accurate tracking result is obtained when the rate is higher than a pre-defined threshold [52]. The threshold we defined is 15% for our performance evaluation. False positive rate (FPR) is calculated by dividing number of false positive frames by total number of frames with pedestrian, true positive rate (TPR) is calculated by dividing number of true positive frames by total number of frames with pedestrian.

5.1.1 Pedestrian tracking performance for 2000 videos

In Table 5.1, we evaluated our tracking performance for all the 2000 videos and categorized them into four classes: excellent, which accurately tracked more than 90% of the frames with pedestrian in the video; very good, which accurately tracked more

than 60% of the frames with pedestrian in the video; fair, which tracked around 30% of the frames; and poor, which tracked less than 30%. We mark the tracking results by red boxes and the actual pedestrian location by yellow boxes.

Table 5.1 Tracking performance for 2000 videos

	Excellent	Very good	Fair	Poor
Percentage	40.59%	22.34%	28.42%	8.65%

Fig. 5.1. Sample excellent tracking frame sequence (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.1 shows an example of excellent tracking frame sequence (150 frames). The pedestrian is walking along the traffic way and vehicle is driving straight. The tracking performance is good in this example and pedestrian window size is also adapted perfectly throughout the video. The false positive rate in this video is 0% and true positive rate is 100%.

Fig. 5.2. Sample vary good tracking frame sequence (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.2 shows the very good tracking frame sequence. The vehicle is turning left and there is a pedestrian standing on the right side of the vehicle. The tracking program stopped after 41 frames (total frame number with pedestrian is 55). Two reasons lead to pos. rejection: 1. Pedestrian viewpoint changed dramatically when the vehicle is turning. 2. Obvious distortion on the side of the frames due to our wide lens camera. The false positive rate is 0% and true positive rate is 75%.

Fig. 5.3. Sample good tracking frame sequence (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.3 shows the example of good tracking sequence. The pedestrian is walking cross the street. At the beginning tracking is good but stopped after the pedestrian was obscured by utility pole. Currently our algorithm is based on frame-by-frame tracking and mistake on any frame will affect the following frames. The false positive rate is 2% and true positive rate is 50%.

Fig. 5.4. Sample poor tracking frame sequence (every 4 frames)

Figure 5.4 shows the sample poor tracking sequence. Its a night scenario, vehicle is stopping and pedestrian is running from left to right. The illumination is extremely inconstant in this scenario. Pedestrian can just be tracked in several frames among the 150 frames. Although some night scenarios can be tracked accurately, night vision is still a challenge task for tracking because its illumination variation. Pedestrian even cannot be detected clearly by human eyes in the dark. Also, the camera does not have night vision capability and does not perform well in night. The false positive

rate for this video is 216%, which is calculated by dividing number of frames with false alarm (106 frames) with number of frames with pedestrian (49 frames) . True positive rate for this video is 5.4%.

5.1.2 Tracking performance comparison

Fig. 5.5. Tracking results by leaps method for excellent tracking category (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.5 shows the tracking results by using the leaps method. The results show that when the pedestrian is relatively small in the fames, leaps method is not sensitive and false rejection rate is high. The leaps method is a very efficient tracking method when the color of pedestrian is distinctive from the background. Thus, in this scenario, when the pedestrian size becomes bigger, the tracking performance is good. The false positive rate is 0% and true positive rate is 63.6%.

Fig. 5.6. Tacking results by covariance matrix for excellent tracking category (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.6 shows the tracking results by just using covariance matrix. The pedestrian can be accurately tracked in the example excellent video and true positive rate is 100%.

Fig. 5.7. Tracking results by leaps method for very good tracking category (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.7 shows the leap tracking results for the very good category example video. In this scenario, the color of pedestrian is not as distinctive to the background as the previous pedestrian. The pedestrian can be tracked at the beginning, but

become inaccurate in the following and finally stopped. Color information is not reliable without structure and other information in this scenario. The false positive rate is 9.2% and true positive rate is 29%.

Fig. 5.8. Tracking results by covariance matrix for very good tracking category (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.8 shows the covariance tracking results for example video in very good category. Pedestrian cannot be tracked accurately after several frames and pos. rejection is high. Without the histogram constrain, some non-pedestrian windows are selected as candidate window and affect the final results.

Fig. 5.9. Tracking results by leaps method for fair tracking category (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.9 shows the leap tracking results for the example fair tracking category. The false positive rate is very high in this scenario because when the pedestrian is obscured by the utility pole, the similar color of pole and pedestrian make it hard for leap to classify them. The search window stopped on the pole throughout the video.

Fig. 5.10. Tracking results by covariance matrix for fair tracking category (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.10 shows the results of just using covariance matrix for the example video in fair tracking category. The results are better than just using leap tracking but still have high false positive rate in this scenario.

Fig. 5.11. Tracking results by leaps method for poor tracking category (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.11 shows the leap tracking results for the example video in poor tracking category. The true positive rate is almost 0. As mentioned before, leaps method is highly depend on the color information and is not a good tracker in this scenario.

Fig. 5.12. Tracking results by covariance matrix for poor tracking category (every 4 frames with red boxes as tracked pedestrian location, blue boxes as tracking mistakes and yellow boxes as missed pedestrian)

Figure 5.12 shows the covariance matrix tracking for the example video in poor tracking category. Its also very challenging to accurately track the pedestrian in this scenario.

5.2 Pedestrian speed calculation and status analysis

Among the tracked 2,000 videos, around 1,200 videos are eligible for pedestrian status analysis because many of them do not have interaction between pedestrian and vehicle. We extracted pedestrian location, pedestrian size in frame domain and vehicle speed, calculate vertical distance changes and horizontal distance changes by projection during time interval T we specified.

Fig. 5.13. Walking pedestrian for status analysis

Figure 5.13 shows the result when the pedestrian is walking across the road. The vehicle is driving straight and going to stop at the intersection. The vertical distance D_v in the initial frame is calculated as 8.56 meters, horizontal distance D_H is 1.45 meters. Based on our tracking result, we calculated the average speed of the pedestrian is 1.43 m/s. The speed we calculated manually is 1.36 m/s. The accuracy rate is 95%. This shows that the automatic method generates reasonably good result.

Fig. 5.14. Running pedestrian for status analysis

Figure 5.14 illustrates the result when the pedestrian is running. It happened in front of a shopping mall. The pedestrian is crossing the street with a high speed. Our tracking method has a good performance on this situation and we extracted the pedestrian location in each frame. The moving speed of the pedestrian is calculated as 4.4 m/s., while the speed we calculated manually is 4.1m/s. The accuracy rate is 93%.

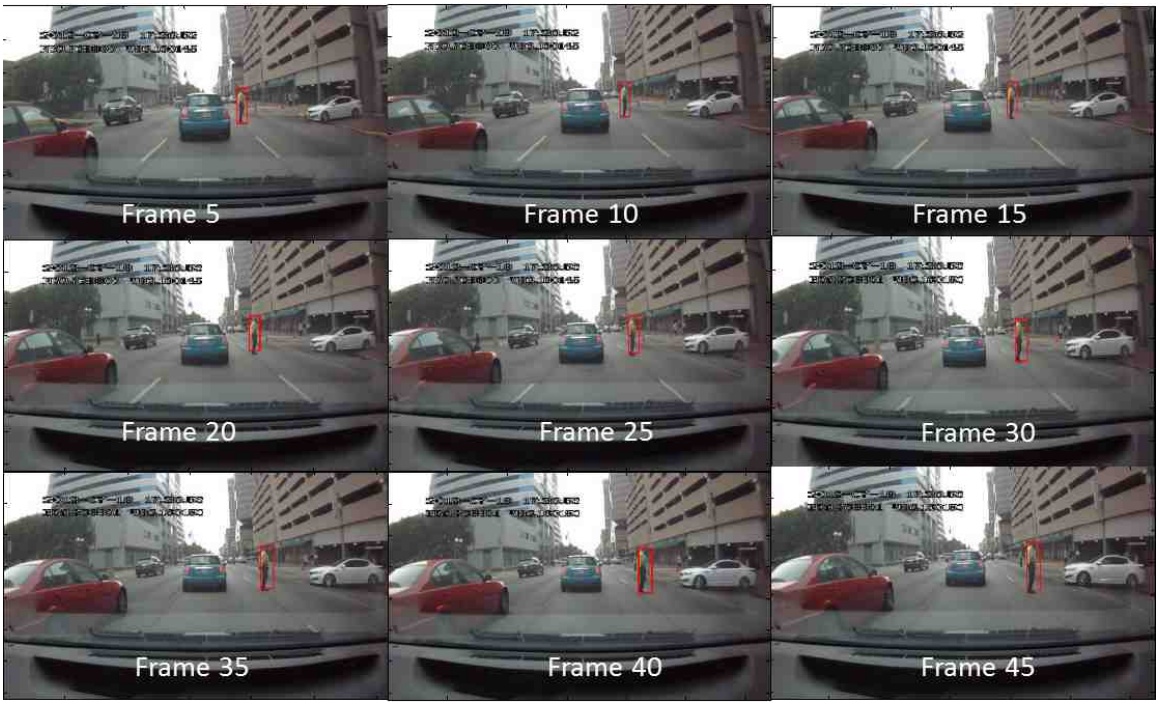

Fig. 5.15. Standing pedestrian for status analysis

Figure 5.15 shows another scenario when there is a person standing on the road. Its really close when the vehicle is passing by the pedestrian. But since the pedestrian is always standing beside the right side of the vehicle, he will be safe. Our tracking step also helps to detect the pedestrian location in each frame, and we find that the horizontal distance (D_H) is almost constant throughout the video. Thus, we can automatically get the information that the pedestrian is standing.

5.2.1 Pedestrian speed distribution by different location

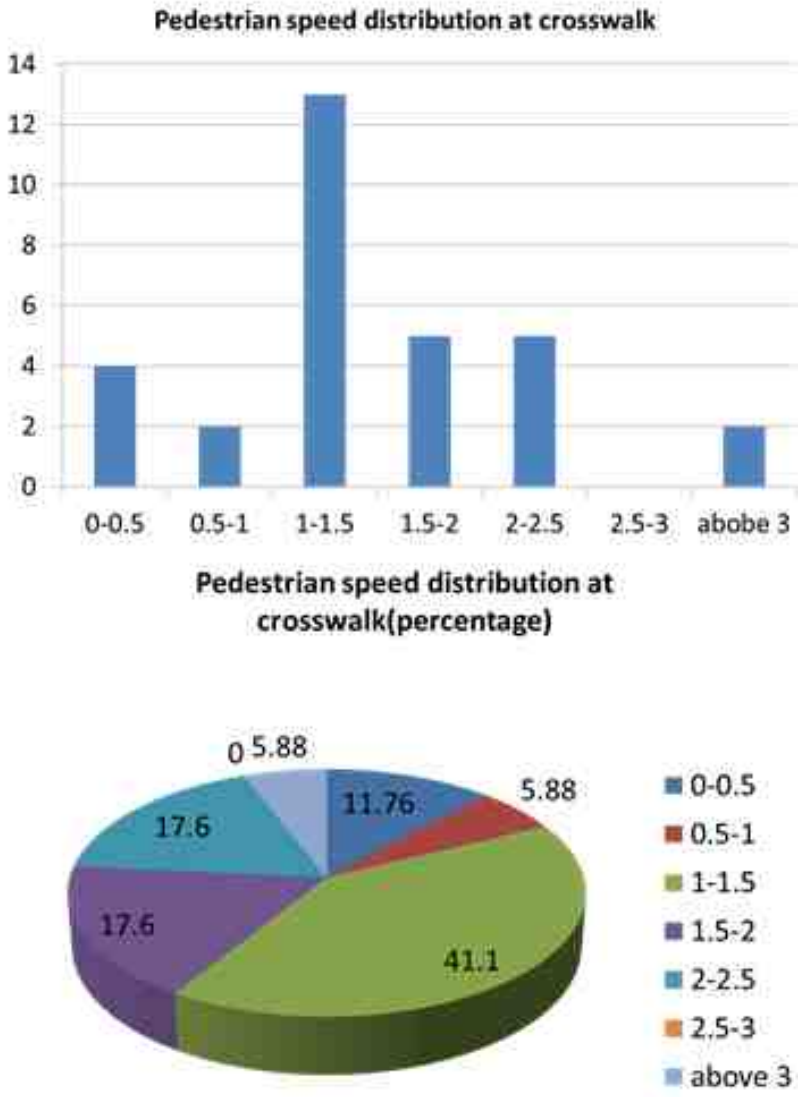

Fig. 5.16. Pedestrian speed distribution at crosswalk scenarios

The distribution of pedestrian speed at crosswalk is shown in Figure 5.16. More than 40 percent of pedestrians are walking between 1 to 1.5 m/s in this scenario. Equally around 17% pedestrians are walking at 1.5 to 2 m/s and 2 to 2.5 m/s. Around 10% pedestrian are walking between below 0.5 m/s. Most pedestrians in this category are just standing beside the road and waiting to cross the street. No scenarios with

speed 2.5 to 3 m/s are found in our tested data. The mean speed in this situation is 1.62 m/s and standard deviation is 0.69.

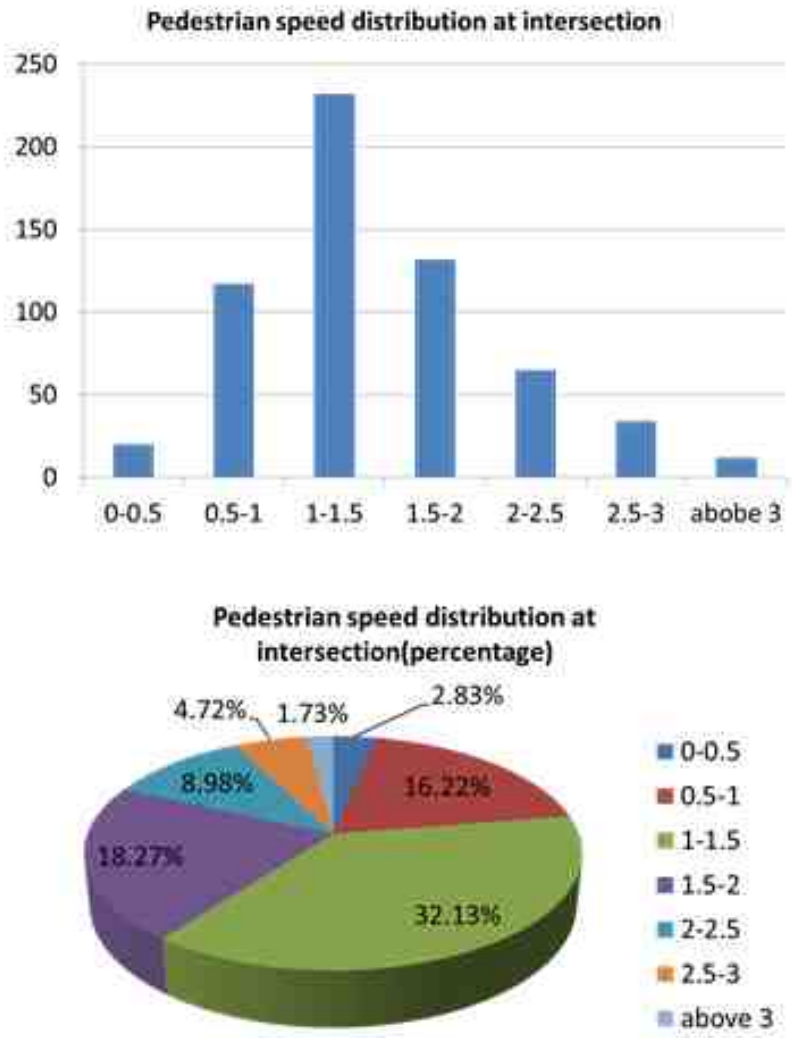

Fig. 5.17. Pedestrian speed distribution at intersection scenarios

Figure 5.17 shows the pedestrian speed distribution in intersection. Its a relatively normal distribution with centered at 1 to 1.5 m/s. Vehicle may go straight, turn left and turning right at intersection, while pedestrian also share more freedom at intersection than other locations. Thus, more complicated and various scenarios can be found in this situation. Besides the normal pedestrian speed, 0.5 to 1 m/s and

1.5 to 2 m/s are also very common. Very slow speed (below 0.5 m/s) and very high speed (above 3 m/s) can also be found at this location. The mean speed is 1.52 m/s and standard deviation is 0.64.

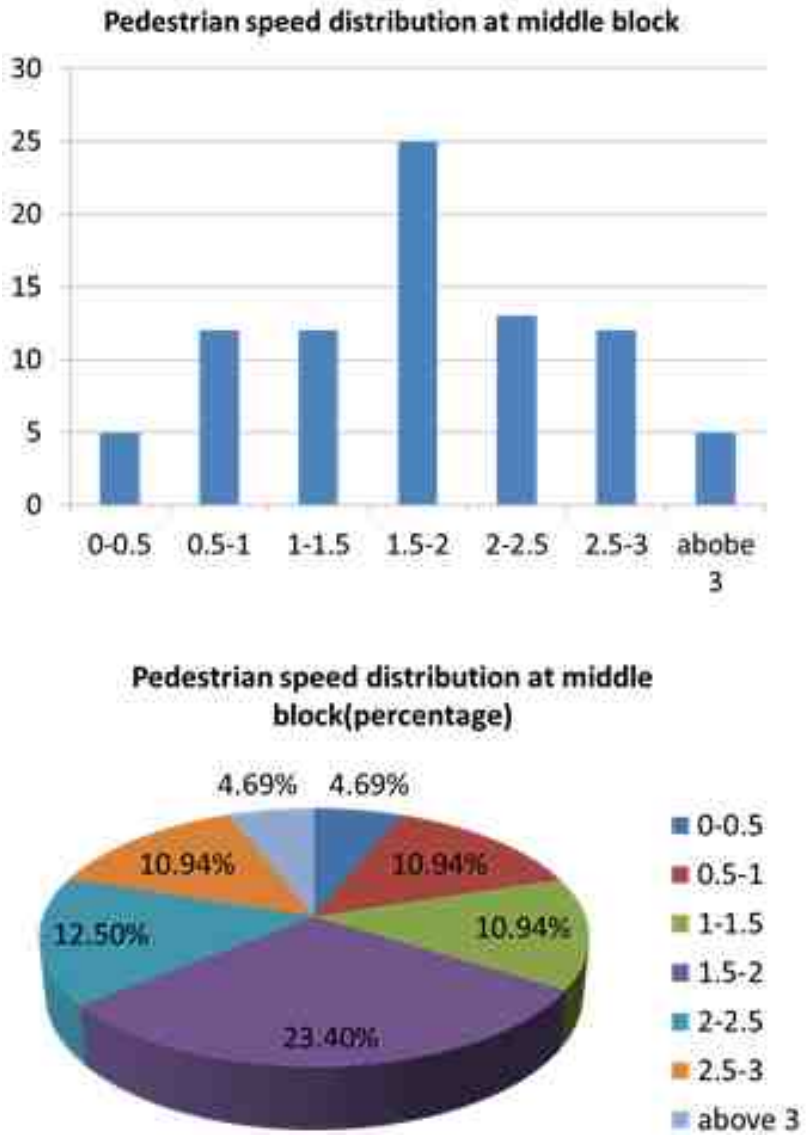

Fig. 5.18. Pedestrian speed distribution at middle block scenarios

Figure 5.18 shows the pedestrian speed at middle block. 1.5 to 2 m/s shares the largest percentage. Also, its very interesting that the percentage of speed around 2 to 3 or more meter per second is very high, some pedestrian are running and the others are walking with high speed. Without crosswalk or traffic control, people are intended to cross the street with relatively higher speed.

Table 5.2 shows the comparison of mean and standard deviation for pedestrian speed in different location. The result suggests that pedestrian will increase speed when they are in road. Their intention to cross the street may lead to a higher speed.

Table 5.2 Mean and standard deviation of pedestrian speed

	Mean	Standard deviation
Cross walk	1.62	0.69
Intersection	1.52	0.64
Middle block	1.73	0.8

5.3 Possibility of collision estimation

After pedestrian tracking and pedestrian status estimation, we can further perform collision analysis. The GPS information can be used for vehicle speed estimation; video can be used for pedestrian location detection and pedestrian speed estimated as we have discussed in the sections 4.2.1 to 4.3.1. We will use the method we introduced in section 4.3.2 for pre-collision analysis.

Fig. 5.19. Pedestrian is walking cross the street with potential conflict

Figure 5.19 shows the scenario when pedestrian is walking cross the street. At the beginning of the video, we get the vehicle velocity as 17 mph from the GPS information, vertical distance between pedestrian and vehicle is 8.7 meters, horizontal distance between pedestrian and vehicle is 1.75 meters. We assume the pedestrian speed is constant during that short period of time, thus we calculated the average pedestrian speed is 1.35 m/s based on the first 20 frames. If the vehicle still keeps driving with the initial speed, it will take 1.14 seconds to drive to the collision point. During the same time, pedestrians walking distance is around 1.5 meters, make the vertical distance zero and horizontal distance around 0.25 meters. Its a very dangerous distance for the pedestrian.

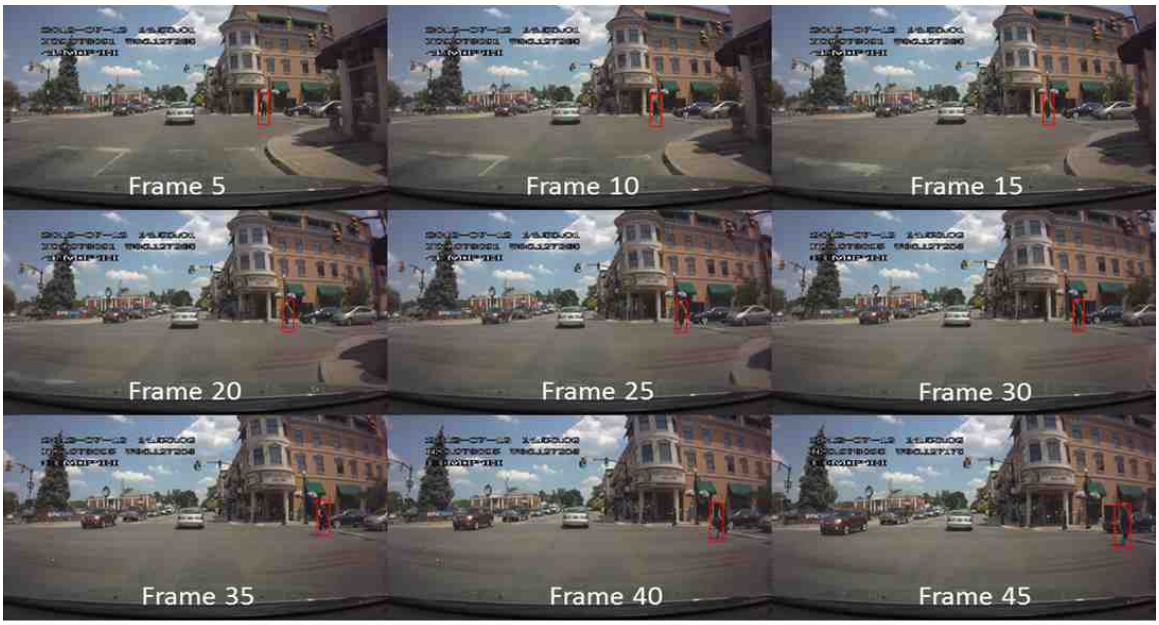

Fig. 5.20. Pedestrian is walking along the traffic way without potential conflict

Figure 5.20 shows the scenario when pedestrian is walking along the traffic way. The vehicle speed is 4 mph based on GPS information. At the beginning of the video, we calculated the vertical distance from the pedestrian to vehicle is 22 meters, the horizontal distance is 7 meters. During the 5 seconds video, we learned that the horizontal distance between pedestrian and vehicle is almost constant, thus no collision will happen.

6. CONCLUSION AND FUTURE WORK

In this thesis, we designed a new pedestrian tracking and status analysis method for naturalistic driving data. To robustly track pedestrians, we separate pedestrians into head, body and leg sections, and use body section as the main region for feature matching, since it is more stable and constant during the whole video. We also give a lower weight for legs and learn their color, and periodic movement pattern. In feature matching, we use leaps method to efficiently find the interesting area, and then use covariance matrix and histogram to find the most similar area in the current frame. The candidate regions are grouped by voting or clustering method to help us determine the final pedestrian location by feature matching method. We also used the Unscented Kalman Filter in the last tracking step to efficiently determine the pedestrian size and also check our feature matching accuracy.

By using the pedestrian tracking results, we designed an algorithm for pedestrian status and pre-collision analysis. We calculated pedestrian speed for several kinds of scenarios and discussed their difference. This information helps us analyze pedestrians status in different scenarios. We also discussed the collision possibility analysis by combining the tracked pedestrian location, pedestrian speed and the GPS information.

We evaluated around 2000 videos for pedestrian tracking analysis using our naturalistic driving data. The experimental results show that our proposed method is very promising with better performance than leap and covariance matrix tracking methods.

We also used the tracking results to analyze the pedestrian speed using the naturalistic driving data. We found those pedestrians are generally walking faster in the middle blocks than in crosswalks and intersections.

Designing an accurate pedestrian safety system still needs a lot of work. This thesis provides us a good basis for future work. In the future, the tracking algorithm can be improved to be more adaptive to different illumination. Second, we need to run more data and acquire more information about the distribution of pedestrian speed and pedestrian status in different scenarios. Third, it will be ideal if our tracking algorithm can work in real-time.

LIST OF REFERENCES

LIST OF REFERENCES

[1] Y. Zhang, T. Z. Q. D. Yao, and L. Peng, "Pedestrian safety analysis in mixed traffic conditions using video data," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1832–1844, 2012.

[2] J. Gresset and F. Meyer, "Risk of automobile accidents among elderly drivers with impairments or chronic diseases," *Canadian Journal of Public Health*, vol. 85, pp. 282–285, 1993.

[3] J. C. Stutts and W. W. Hunter, "Motor vehicle and roadway factors in pedestrian and bicyclist injuries an examination based on emergency department data," *Accident Analysis and Prevention*, vol. 31, pp. 505–514, 1999.

[4] E. Commission, "Road safety evolution in EU," *Directorate-General Press*, pp. 1–4, 2012.

[5] R. Dandona, "Patterns of road traffic injuries in a vulnerable population in hyderabad," *Injury Prevention 12.3*, pp. 183–188, 2006.

[6] C. V. Zegeer and M. Bushell, "Pedestrian crash trends and potential countermeasures from around the world," *Accident Analysis and Prevention 44.1*, pp. 3–11, 2012.

[7] K. Peltzer and W. Renner, "Superstition, risk-taking and risk perception of accidents among South African taxi drivers," *Accident Analysis and Prevention*, vol. 35, pp. 619–623, 2003.

[8] F. A. Whitlock, "Death on the road: a study in social violence," *Taylor and Francis*, vol. 2, pp. 1–204, 1971.

[9] H. Ying and S. Chunfu, "The design method of pedestrian crossing facilities on the urban expressway," *3rd International Conference on Digital Manufacturing and Automation*, pp. 899–902, 2012.

[10] K. W. Ogden, "Safer roads a guide to road safety engineering," *Avebury Technical*, pp. 1–536, 1996.

[11] R. Lamm, B. Psarianos, and T. Mailaender, "Highway design and traffic safety engineering handbook," *Mcgraw-Hill Columbus Ohio*, vol. 2, pp. 3–16, 1999.

[12] D. Llorca, M. Sotelo, I. Parra, and J. E. Naranjo, "An experimental study on pitch compensation in pedestrian-protection systems for collision avoidance and mitigation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, pp. 469–474, 2009.

[13] G. Leen and D. Heffernan, "Expanding automotive electronic systems," *Computer*, vol. 35, pp. 88–93, 2002.

[14] A. Jaraniene and G. Jakubauskas, "Improvement of road safety using passive and active intelligent vehicle safety systems," *Transport*, vol. 22, pp. 284–289, 2007.

[15] D. R. Mayhew and H. M. Simpson, "The safety value of driver education an training," *Injury Prevention*, vol. 8, pp. ii3–ii8, 2002.

[16] M. Limbourg and D. Gerber, "A parent training program for the road safety education of preschool children," *Accident Analysis and Prevention*, vol. 13, pp. 255–267, 1981.

[17] K. Yang, E. Y. Du, P. Jiang, and Y. Chen, "Automatic categorization-based multi-stage pedestrian detection," *15th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 451–456, 2012.

[18] D. M. Gavrila and S. Munder, "Multi-cue pedestrian detection and tracking from a moving vehicle," *International Journal of Computer Vision*, vol. 73, pp. 41–59, 2007.

[19] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 2179–2195, 2009.

[20] H. Elzein, S. Lakshmanan, and P. Watta, "A motion and shape-based pedestrian detection algorithm," *Intelligent Vehicles Symposium*, pp. 500–504, 2003.

[21] M. Enzweiler, P. Kanter, and D. Gavrila, "Monocular pedestrian recognition using motion parallax," *Intelligent Vehicles Symposium*, pp. 792–797, 2008.

[22] A. Shashua, Y. Gdalyahu, and G. Hayun, "Pedestrian detection for driver assistance system: single frame classification and system level performance," *IEEE Intelligent Vehicle Symposium*, pp. 1–6, 2004.

[23] P. Viola, M. J. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," *International Journal of Computer Vision*, vol. 63, pp. 153–161, 2005.

[24] C. Papageorgiou and T. Poggio, "A trainable system for object detection," *International Journal of Computer Vision*, vol. 38, pp. 15–33, 2000.

[25] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 886–893, 2005.

[26] B. Wu and R. Nevatia, "Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors," *International Journal of Computer Vision*, vol. 75, pp. 247–266, 2007.

[27] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *International Journal of Computer Vision*, vol. 77, pp. 259–289, 2008.

[28] A. Mohan, C. Papageorgiou, and T. Poggio, "Example-based object detection in images by components," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 349–361, 2001.

[29] D. Ramanan, D. A. Forsyth, and A. Zisserman, "Strike a pose: Tracking people by finding stylized poses," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 271–278, 2005.

[30] C. Wojek and B. Schiele, "A performance evaluation of single and multi-feature people detection," *Pattern Recognition, ed: Springer*, pp. 82–91, 2008.

[31] Y. Xu, D. Xu, S. Lin, and T. X. Han, "Detection of sudden pedestrian crossings for driving assistance systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 42, pp. 729–739, 2012.

[32] B. Wu and R. Nevatia, "Tracking of multiple, partially occluded humans based on static body part detection," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 951–958, 2006.

[33] M. Meuter, U. Iurgel, S.-B. Park, and A. Kummert, "The unscented kalman filter for pedestrian tracking from a moving host," *Intelligent Vehicles Symposium*, pp. 37–42, 2008.

[34] M. A. Hossain, Y. Makihara, W. Junqui, and Y. Yagi, "Clothes-invariant gait identification using part-based adaptive weight control," *19th International Conference on Pattern Recognition*, pp. 1–4, 2008.

[35] D. Forsthoefel, D. S. Wills, and L. M. Wills, "Unsupervised video leap segmentation for fast detection of salient segment transformations in mobile sequences," *15th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 728–733, 2012.

[36] D. Forsthoefel and D. S. Wills, "Fast leap segmentation," *International Journal on Computer Vision*, pp. 128–134, 2012.

[37] F. Porikli and P. Meer, "Covariance tracking using model update based on lie algebra," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 728–735, 2006.

[38] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," *Computer Vision ECCV*, pp. 589–600, 2006.

[39] M. Andreas, A. Prioletti, M. Trivedi, B. Alberto, and B. Moeslund, "Two-stage part-based pedestrian detection," *15th International IEEE Conference on Intelligent Transportation Systems*, pp. 73–77, 2012.

[40] A. C. Cosma, R. Brehar, and S. Nedevschi, "Part-based pedestrian detection using hog features and vertical symmetry," *IEEE International Conference on Intelligent Computer Communication and Processing*, pp. 229–236, 2012.

[41] Y. Wu, J. L. B. Wu, and H. Lu, "Probabilistic tracking on riemannian manifolds," *19th International Conference on Pattern Recognition*, pp. 1–4, 2008.

[42] K. Guo, P. Ishwar, and J. Konrad, "Action recognition using sparse representation on covariance manifolds of optical flow," *7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, pp. 188–195, 2010.

[43] A. Cherian, A. B. S. Sra, and N. Papanikolopoulos, "Jensen-bregman logdet divergence with application to efficient similarity search for covariance matrices," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, pp. 2161–2174, 2012.

[44] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, "Geometric means in a novel vector space structure on symmetric positive-definite matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 29, pp. 328–347, 2007.

[45] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 2037–2041, 2006.

[46] G. Zhang, X. Huang, Y. W. S. Z. Li, and X. Wu, "Boosting local binary pattern (lbp)-based face recognition," *Advances in Biometric Person Authentication*, pp. 179–186, 2005.

[47] X. Cai, B. Xiao, C. Wang, and R. Zhang, "Quadratic-chi similarity metric learning for histogram feature," *First Asian Conference on Pattern Recognition*, pp. 47–51, 2011.

[48] Y. Chan, A. G. Hu, and J. Plant, "A kalman filter based tracking scheme with input estimation," *IEEE Transactions on Aerospace and Electronic Systems*, pp. 237–244, 1979.

[49] B. Stenger, P. R. Mendona, and R. Cipolla, "Model-based hand tracking using an unscented kalman filter," *British Machine Vision Conference*, pp. 63–72, 2001.

[50] E. A. Wan and R. V. D. Merwe, "The unscented kalman filter for nonlinear estimation," *Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pp. 153–158, 2000.

[51] E. A. Wan and R. V. D. Merwe, "The unscented kalman filter," *Kalman Filtering and Neural Networks*, pp. 221–280, 2001.

[52] Y. Fei, D. Makris, and S. A. Velastin, "Performance evaluation of object tracking algorithms," *10th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, Rio de Janeiro, Brazil*, pp. 1–1, 2007.