UNIVERSITY OF SOUTHAMPTON

FACULTY OF ENGINEERING AND THE ENVIRONMENT

Civil and Environmental Engineering

Study of Pedestrian-Vehicle Interaction Behaviour by Microscopic Simulation Modelling

by

Tianjiao Wang

Thesis for the degree of Doctor of Philosophy
in Transportation Engineering

September 2012

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND THE ENVIRONMENT

Civil and Environmental Engineering

Doctor of Philosophy in Transportation Engineering

STUDY OF PEDESTRIAN-VEHICLE INTERACTION BEHAVIOUR BY MICROSCOPIC SIMULATION MODELLING

by Tianjiao Wang

Walking is healthy, environmentally beneficial and sustainable to human society. Travellers increasingly are being encouraged to walk more. However, pedestrians' interaction with motorised vehicles is a major constraint to their movement. Many innovative treatments have been developed to balance the two modes. Proper methods are required to evaluate and compare performances of different treatments to support decision making. Micro-simulation is a useful supplementary tool for such evaluation and comparison studies for its cost-effectiveness and non-intrusiveness. However, there is a significant gap between capabilities of existing simulation models and practical needs. New understandings of the Pedestrian-Vehicle Interaction (PVI) behaviour and corresponding micro-simulation models are required to conduct micro-simulation studies of the interaction process between the two modes to derive new knowledge of the mixed traffic.

This dissertation presents the development and application of a micro-simulation model, PVISIM (Pedestrian-Vehicle Interaction SIMulation), to study PVI behaviour in a range of circumstances in an urban street environment. Key contributions relate to the collection of a substantial data base, development and validation of the model, an appreciation of the value of the approach and new understandings of PVI behaviour. A series of studies to measure behaviour based on the data collected in Beijing, China have been detailed. Intra vehicle and pedestrian behaviour models were developed and validated separately, incorporating the best available understandings from existing published studies and in accordance with the specific local data. The two modes were integrated by interpreting new findings from the study of microscopic interaction behaviour of the two modes. The complete model was validated against field data independent of those used in model development, covering a number of typical scenarios, including both unsignalised and signalised situations.

The validated model was applied to study a typical unsignalised scenario by analysing system performances under different combinations of vehicular traffic and pedestrian crossing demand, in terms of efficiency, safety and environmental impact. Also, operations of different treatments including no-control, Zebra crossing, fixed-time signal crossing and Puffin crossing at two typical types of locations were compared. Interpretations and recommendations were given for each application. The results can be used to supplement existing guidelines for pedestrian related problems, and also contribute to the knowledge base to incorporate pedestrians into current micro-simulation tools in a more realistic way.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

DECLARATION OF AUTHORSHIP

I, Tianjiao Wang [please print name] declare that this thesis and the work presented in it are my own and has been generated by me as the result of my own original research.

[title of thesis] Study of Pedestrian-Vehicle Interaction Behaviour by Microscopic Simulation Modelling .

I confirm that:

This work was done wholly or mainly while in candidature for a research degree at this University;

Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;

Where I have consulted the published work of others, this is always clearly attributed;

Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;

I have acknowledged all main sources of help;

Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;

Either none of this work has been published before submission, or parts of this work have been published as: [please list references below]:

[1] Wang, T., Wu, J. and McDonald, M. (2012) A Micro-Simulation Model of Pedestrian-Vehicle Interaction Behaviour at Unsignalised Mid-Block Locations. *Proceedings of 15th International IEEE Conference on Intelligent Transportation Systems (ITSC 2012)*, Anchorage, Alaska, United States, 16-19,

2012.

[2] Wang, T., Wu, J., Zheng, P. and McDonald, M. (2010) Study of Pedestrians' Gap Acceptance Behavior when They Jaywalk outside Crossing Facilities. *Proceedings of 13th International IEEE Conference on Intelligent Transportation Systems (ITSC 2010),* Madeira Island, Portugal, September 19-22, 2010.

[3] Wang, T., Wu, J., Zheng, P. and McDonald, M. (2010) A Framework for Analysing and Modelling Pedestrian-Vehicle Interaction Behaviour in a Micro-simulation Environment. *Proceedings (CD) of 3rd Transport Research Arena Conference (TRA 2010),* Brussels, Belgium, 7-10, June, 2010.

Signed: ____.

Date: September 2012 .

ACKNOWLEDGEMENT

I would like to express my deepest thanks to my supervisory team, Prof. Mike McDonald, Prof. Jianping Wu and Prof. Pengjun Zheng for their patient guidance, encouragement and securing a number of major funding throughout this work.

I would also like to express my sincere gratitude to all my colleges in Transportation Research Group, University of Southampton. Their comments and questions have contributed much to this work. Special thanks must be given to our technicians, Mr. Doug Robinson and Mr. Brian Mould for their help with the equipments for data collection.

I gratefully acknowledge the School of Civil Engineering and the Environment (currently merged to the Faculty of Engineering and the Environment), University of Southampton for providing me the opportunity to conduct this research and the scholarship covering the entire period of this study. Special thanks are given to Ms. Melanie Hallford and Ms. Jacqui Holmes for their patient help for trivial administrative things.

Thanks are also given to the staff in Beijing Traffic Management Bureau, as well as the people who participated in my experiments for their help for collecting the necessary data at a number of sites in Beijing, China.

Also, I must thank my families and friends for their unconditional love and support.

Last, thanks are also given to those who devoted to this field of research. This project can never be accomplished without referring to their work.

FREQUENTLY USED ABBREVIATIONS

BCA	Bus or Coach (Articulated)
BCR	Bus or Coach (Regular)
CO_2	Carbon Dioxide
DfT	Department for Transport (UK)
DOT	Department Of Transportation (US)
EPA	Environmental Protection Agency (US)
FHWA	Federal HighWay Administration (US)
FIS	Fuzzy Inference System
FLOWSIM	Fuzzy logic based traffic micro-simulation
GHR	Gazis-Herman-Rothery (car-following model)
GPS	Global Positioning System
HC	Hydrocarbons
HCM	Highway Capacity Manual (US)
HCV	Heavy Commercial Vehicle
HGV	Heavy Goods Vehicle
LV	Light Vehicles
MCV	Medium Commercial Vehicle
MOHURD	Ministry Of Housing and Urban-Rural Development (China)
MUTCD	Manual of Uniform Traffic Control Devices (US)
NGSIM	Next Generation SIMulation program (US)
NOx	Nitrogen Oxide
O-D	Origin-Destination pair
O/D	Origin or Destination
OF	Older and Female
OM	Older and Male
pcu	passenger car unit
Pelican	Pedestrian light controlled crossings
PGA	Pedestrian Gap Acceptance
PM	Particle Matters
Puffin	Pedestrian user friendly intelligent crossings
PVI	Pedestrian-Vehicle Interaction

PVISIM	Pedestrian-Vehicle Interaction SIMulation
SPSS	Statistical Package for the Social Sciences
TRB	Transportation Research Board (US)
VISSIM	Verkehr In Stadten – SIMulationsmedell ("Traffic In Cities – Simulation Model" in German)
YF	Younger and Female
YM	Younger and Male

CHAPTER 1
INTRODUCTION

1.1 Background and motivation

Walking as a transport mode is healthy, environmentally beneficial and sustainable to human society. A significant amount of research has shown that people who use cars less and promote walking are much healthier than those who are excessively dependent on cars (Saelens et al 2003, Frank et al 2004, Pucher and Dijkstra 2003). Also, evidence shows that the motorised transport is harmful to the natural environment because it introduces a number of emissions consisting of many dangerous pollutants and consumes a large amount of energy mostly generated from valuable non-renewable fossil fuels (Highways Agency 2010, Dore et al 2005, Granovskii et al 2006). Therefore, for health and environmental reasons, travellers increasingly are being encouraged to walk more, either as a main mode of travel or as part of a multimodal trip.

However, roads can be barriers to pedestrian movement. Pedestrians' interaction with motorised vehicles, for example street crossing behaviour in urban traffic systems has been found to be one of the major constraints to pedestrian activities (Hine 1996). Vehicular traffic can cost pedestrians additional time or distance to complete a journey and pedestrians are extremely vulnerable when they cross the vehicle lanes.

On the other hand, pedestrian crossing activities also have impacts on the vehicle flow. Such impacts may include increased delays, decreased capacity, and harmful environmental influences. For example, the pedestrian jaywalking behaviour at unsignalised locations can lead to traffic chaos and serious safety problems. It may be argued that whilst pedestrian crossing facilities with inappropriate implementation of location or unsuitable signal patterns can still marginally improve the safety, they may result in making vehicle traffic less efficient, produce more emissions and fuel consumption due to increased inconstant flow and stop-and-go phenomena (Hamiltion-Baillie 2008, Cassini 2006, New Straits Times 2002). The high costs of installation and maintenance for some advanced crossing facilities are also factors that decision makers need to consider (Department for Transport 1995a). Limited budgets for building new crossing facilities are always expected to be spent where the effect of improvement is most significant.

Consequently, addressing the Pedestrian-Vehicle Interaction (PVI) related problems is a multi-criteria trade-off that the traffic engineers have been facing for a long time. Over the past decades, many innovative treatments, for example different pedestrian crossing facilities, have been developed to balance between pedestrians and vehicles. The emergence of various treatments demands proper methods for analysing, evaluating and comparing performances so that engineers can decide which plan is best for any specific scenario.

Typically, the selection of different treatments is determined by engineering guidelines and the evaluation of such treatments is carried out through empirical before and after studies. While these approaches are adequate, some researchers argue that these methods still have many drawbacks. For example, the evaluation based on field study is time-consuming and lacks microscopic detail; it is nearly impossible to compare several potential solutions and choose the best trade-off for a particular site by implementing all of them in the real system; and there is always a lack of foresight of how the system will behave if a specific treatment is implemented (Schroeder 2008, Du 2008).

To overcome the disadvantages mentioned above, more recently, engineers have started to apply traffic micro-simulation as a supplement to existing methods for the prediction and evaluation task. Such micro-simulation is an integration of several essential road users' behaviour models of car following, lane changing, gap acceptance, street crossing and route choice, etc. Different road user objects are generated for a scenario created in a simulation environment and act according to several behavioural and control models. For each simulation step, "the dynamics of each road user", for example instantaneous position, speed, acceleration, delay and emission, etc are calculated and recorded by the simulation system and then can be extracted and analysed (FHWA 1998). The behaviour of the real system is mirrored by the micro-simulation, which is then used as a convenient and economic tool to study the real system. If properly modelled and validated, micro-simulation can provide detailed prediction and evaluation results for different sites to assist existing methods for decision making. The performance of the real traffic system can be measured with several model indicators, in terms of efficiency, safety and environmental impact, providing a non-intrusive and cost effective way to investigate a real traffic system and permit traffic engineers to have a bird-eye view and an instant feel for the problems and potential solutions (Algers et al 1997).

At the core of a sound micro-simulation model are the road users' behavioural models. The deeper the road users' behaviour is understood, the closer the simulation model can be reality, so be more reliable. In recent years, although micro-simulation models for traffic applications have undergone a long development and most of the motorists' behaviour have been studied and modelled, there are still relatively few studies of pedestrians' behaviour and modelling, especially on the pedestrian-vehicle interaction process (Harney 2002). Consequently, there are still many limitations in the use of current micro-simulation tools to evaluate performances of pedestrian related traffic systems. For example, existing micro-simulation tools are of limited use to evaluate the interaction between pedestrians and vehicles at unsignalised locations or to conduct comparison studies between the operations of various signalised and unsignalised scenarios. It has been pointed out that "current simulation software usually ignores or oversimplifies the interaction behaviour between the two modes due to insufficient knowledge to build and validate the microscopic behaviour models" (Schroeder 2008). Some others claimed that "there is a need to study the pedestrian-vehicle interaction behaviour and develop models to incorporate pedestrian objects with vehicular traffic in existing micro-simulation tools and to use such tools to evaluate different pedestrian related treatments in a micro-simulation environment to support decision making" (Ishaque 2006).

In summary, there is a significant demand to study and model the pedestrian-vehicle interaction behaviour in the urban street environment, especially at unsignalised areas, and to investigate how to conduct a multi-criteria evaluation on different treatments for pedestrian related problems in a micro-simulation environment. The benefits of this research are twofold. First, some fundamental understandings of the pedestrian-vehicle interaction behaviour will be gained and this knowledge can be interpreted and integrated into existing micro-simulation models. Second, with the improved micro-simulation tool, various design, management and control plans can be tested and evaluated in details. Indicators of efficiency, safety and environmental impact, etc can be generated and presented more vividly, and the results of the simulation study can play as supplements for existing engineering guidelines.

1.2 Research objectives

The main aim of this research is to investigate the interaction behaviour between pedestrians and vehicles in urban street environment focused on mid-block locations with microscopic simulation modelling, in order to provide guidance and a micro-simulation tool for the design and evaluation of treatments for pedestrian related problems at such locations.

Understanding behaviour is crucial to the satisfactory completion of this aim and part of the research has related to the collection of a very substantial data base in China. As behaviour will vary by location and due to the cultural differences between the UK and China are substantial, the applications and treatments will have fundamental value for China in an area of growing importance. Based on the above discussion, the following research objectives are specified:

Objective 1 is to develop new understandings of the microscopic interaction behaviour between pedestrians and vehicles in the urban street environment in China;

Objective 2 is to develop a micro-simulation model that can fully simulate the interaction process between the two modes for both signalised and unsignalised scenarios, incorporating the understandings from the new behaviour study and best available research findings published previously;

Objective 3 is to apply the model to derive new understandings from a multi-criteria evaluation study of a pure unsignalised area and a comparison study of different treatments regarding pedestrian-vehicle interactions in some typical scenarios.

To achieve these objectives, this research involves the following main activities:

(1) To review current common pedestrian related treatments and the engineering guidance to build them, review the development of traffic micro-simulation models and identify the gap between the current models and the new increasing demand and how existing knowledge can inspire the development of the new PVI model;

(2) To collect field data for behaviour study and model development, calibration and validation;

(3) To develop new understandings of the microscopic pedestrian-vehicle interaction process from data analysis and behaviour study and use such knowledge to design and implement the regarding micro-simulation models;

(4) To calibrate and validate the complete micro-simulation model the relative sub-models in accordance with the local situation;

(5) To conduct several evaluation studies using the newly developed micro-simulation model for several typical scenarios, in terms of efficiency, safety and environmental impact.

1.3 Thesis structure

This dissertation is organised into the following 9 chapters.

Chapter 1 discusses the background and motivation of this research; clarifies the research objectives and main activities; and then introduces the structure of this dissertation.

Chapter 2 provides an overview of literature of pedestrian related treatments and the current situation of traffic micro-simulation modelling, identifies the gap between current micro-simulation tools and the research objectives, and inspires the methodology chosen in this research.

Chapter 3 introduces the methodology applied in this research. Emphasis is given to data collection to obtain a substantial data base.

Chapter 4 discusses the vehicle behaviour modelling and the development, calibration and validation of the relative vehicle models in this research.

Chapter 5 discusses the pedestrian behaviour modelling and the development, calibration and validation of the relative pedestrian models in this research.

Chapter 6 analyses and interprets the pedestrian-vehicle interaction behaviour and describes the development, calibration and validation of the relative sub-models.

Chapter 7 describes the validation process for the complete micro-simulation model developed in this research.

Chapter 8 showcases how this model can supplement existing engineering guidelines regarding pedestrian related treatments through case studies.

Chapter 9 draws conclusions of this research and discusses potential future work.

CHAPTER 2
OVERVIEW OF LITERATURE

2.1 Introduction

This chapter provides a general overview of literature in the areas of pedestrian and driver behaviour, treatments regarding their interaction and microscopic traffic simulation. As the detailed development of the models and their credibility is strongly related to both the new data sets collected in this research and on specific results presented in previous literature, most of the more fundamental research is referenced in subsequent chapters where its relevance can be most immediately seen.

2.2 Pedestrians and their interaction with motor vehicles

2.2.1 The position of walking as a transport mode

Walking is probably the most basic and common mode of transport. Nowadays it has been increasingly promoted in urban traffic systems, either as a main mode of travel or as part of a multimodal trip. Walking as a transport mode has many irreplaceable advantages compared to other means of transport.

First, walking has immense health benefits for people. People using walking as a transport mode can at the same time take exercise in the most natural way, which can help ward off a number of diseases, such as hypertension, diabetes and obesity (Hayashi et al 1999, Sample 2008, Saelens et al 2003). The switching from cars to walking can also reduce harmful side effect of motorised transport which generates many emissions that cause respiratory disease. Many researches have shown that a community with less car dependency but promoting walking is less likely to suffer from health problems (Hayashi et al 1999, Sample 2008, Saelens et al 2003).

Second, walking is environmentally friendly and sustainable for human society. Traditional transport systems, which are highly dependent on motor vehicles, have significant impacts on the environment, accounting for approximately 25% of world energy consumption and carbon dioxide emissions (World Energy Council 2007). Greenhouse gas emissions from transport sector are increasing at a faster rate than any other energy using sector (Kahn Ribeiro et al 2007). Road transport is also a major source to the air and noise pollution in towns (US EPA 2011). Those damages to the environment are not only from the motor vehicles themselves but also from the building and maintenance of the transport infrastructure. Therefore, an effective way to mitigate those detrimental impacts is to shift transport mode from motorised to non-motorised. Encouraging non-motorised transport such as walking can effectively reduce emissions and ease the pressure on the demand for non-renewable fossil fuels, which are unlikely to be replaced for many decades even with technological advances (Granovskii et al 2006).

Third, the cost of the walking transport is low compared to most of other modes of transport. Bouwman (2000) gave a comparison of the average cost of transport modes per passenger·km, given the average trip length per transport mode, as shown in Table 2.1, which showed that walking had zero monetary cost and a favourable energy performance based on their low direct and indirect energy input per passenger·km.

Table 2.1 Comparison of costs of various transport modes

Transport mode	Space used for infrastructure (10^{-2} m^2/pkm)	Direct and indirect energy use (MJ/pkm)	Average costs paid by traveller (Euro/pkm)	Travel time (min/pkm)
Petrol passenger car	0.55	1.79	0.170	1.34
Train	0.21	0.98	0.075	0.94
Bus, tram, metro	0.51	1.11	0.085	1.92
Bicycle	0.71	0.04	0.045	5.40
Walking	1.7	0.03	0.000	10.77

pkm = passenger·km

(Source: Bouwman 2000)

Fourth, walking is almost an inevitable element in a multi-modal transport chain. Every person may experience the walking mode to some extent in their daily travel.

For example, walking to and from the parking place, or the bus/railway station are necessary parts in the case of car trips or public transport respectively. When the various trip elements are considered, the share of walking, together with cycling were found much higher, due to their convenience, flexibility and the capability of providing door-to-door transport (Rietveld 2001). In addition, walking plays a substantial role in the transport systems in some developing countries such as China due to the low car ownership, insufficient public transport infrastructure and large population. In China, the pedestrian flows are most common and will continue to be one of the major modes of private transport in its urban transport system in the foreseeable future (Lu et al 2009).

2.2.2 The barriers to walking transport

Due to the advantages of the walking mode discussed above, many cities and smaller towns have begun to encourage their citizens to walk more. However, there are many obstacles for people to adopt walking in the choice of modes, for example, adverse weather condition, street crime, air pollution, insufficient pedestrian facilities, long walking distance and interaction with motorised traffic, etc. Of the many barriers, the interaction with motorised vehicles, focused around the street crossing activities is found a major constraint to pedestrians (Hine 1996).

First, motorised traffic prevents people from walking to the full extent that is possible to them and therefore results in additional costs of journey time or distance for pedestrians. Roads can act as barriers to pedestrian activities in the urban street environment, in which pedestrians often need to detour to locate appropriate crossing facilities for a safer crossing route, sacrificing their efficiency.

Second, the interaction with motorised vehicles can pose severe safety problems to pedestrians, who are regarded as the most vulnerable of all types of road users. In China, the mixture of motorised and non-motorised traffic is one of the major characteristics in urban transport systems and frequent conflicts between the two modes the main causes for most of road accidents. It is reported in China that pedestrian fatalities in traffic accidents are more than 20 thousand every year from

1998, approximately accounting for 30% of all traffic fatalities each year (Traffic Management Bureau, Ministry of Public Security of China 2007). The problem is more severe in some mega cities such as Beijing and Guangzhou, where the ratio of pedestrian fatalities of all traffic fatalities were more significant, being 38% and 40% respectively in 2007. Also, a recent research on pedestrians' opinions about the walking environment in Beijing showed that 74% of the interviewed pedestrians considered the walking environment in this city unsafe (Pan et al 2010).

In addition, the activities of pedestrian street crossing may in turn cause negative impact to vehicular traffic, resulting in efficiency loss and adverse environmental influence for the whole transport system. For example, although the pedestrian crossing facilities can separate the interaction between the pedestrians and motorised vehicles, it is argued that too many such facilities, inappropriate type, set-up location and signal timing can lower the efficiency for vehicular traffic and produce more fuel consumption and emissions due to more inconstant flow and more stop-and-go phenomena (Hamiltion-Baillie 2008, Cassini 2006, New Straits Times 2002). Further, the high costs of installation and maintenance for some advanced siganlised crossing facilities are also factors that the decision makers need to consider (Department for Transport 1995a).

2.2.3 The treatments regarding pedestrian-vehicle interaction

As the interactions between pedestrians and motorised traffic have significant influences on the efficiency, safety and environmental aspects for the urban transport system, for many decades, people have been searching ways to balance the two modes from various aspects, such as legislation, public policy, urban planning, rehabilitation of road users' behaviour and engineering treatments, etc. This research approaches this problem from the engineering aspect.

Nowadays an increasing number of innovative engineering solutions have been devised to accommodate pedestrians in urban transport systems. In terms of how pedestrians and motorised traffic are separated, those solutions mainly fall into two categories: full segregation, such as underpasses and overpasses, and partial segregation, such as at-grade pedestrian crossings. As the underground or over-ground crossings are less common and less popular with pedestrians (Ishaque 2006), this research focuses on surface crossings only. Common treatments regarding the interaction process are reviewed in the following sections.

2.2.3.1 Pedestrian refuge island

The pedestrian refuge island is also called “crossing median”, as shown in Figure 2.1. It provides a small waiting area in the middle of a road, where pedestrians can stop and wait in the middle before finishing crossing the entire road. It is usually applied when no traffic signals exist and pedestrians need to manage the conflicting traffic one direction at a time, or when a road is too wide for some pedestrians to finish the crossing task in one cycle of the pedestrian green light. It is a relatively inexpensive method for improving crossing facilities for pedestrians (FHWA 2003).

Figure 2.1 An example of pedestrian refuge island (Southampton, UK)

2.2.3.2 Zebra crossing

As shown in Figure 2.2, a Zebra crossing is usually equipped with alternating dark and light stripes on road surface, from which it derives its name. A Zebra crossing is typically without traffic lights and gives crossing priority to pedestrians. Pedestrians establish precedence by stepping onto the crossing and so delays to them are minimised. Zebra crossings usually do not require any advanced electronic devices and therefore is low-cost for construction and maintenance.

(a) Southampton, UK   (b) Beijing, China

Figure 2.2 Examples of Zebra crossings in the UK and China

Typically, pedestrians always have right of way on a Zebra crossing and motorists have to yield to pedestrians who have already established their precedence on the crossing. However, the road users' behaviour at Zebra crossings is closely related to the local traffic culture and the actual operations at these locations may vary. For example, in China, where the traffic is less disciplined, motorists seldom give way to pedestrians, even when they have already stepped onto the crossing facility. At zebra crossings in China, the drivers' yielding rate is significantly low and the pedestrian-vehicle interaction behaviour at such locations is more similar to pure uncontrolled locations (Chen et al 2008).

Although a Zebra crossing does not require any complex electronic devices and thus is a convenient and inexpensive solution, the use of Zebra crossings is somewhat argued in the traffic engineering field. First, if the perceived priority rule by road users is ambiguous, Zebra crossings can pose negative effects on traffic safety. A research undertaken in New Zealand found that a Zebra crossing without other safety features on average increases pedestrian accidents by 28% compared to a location without any crossings (New Zealand Transport Agency 2007). Second, if the priority rule is fully perceived and obeyed, the use of such crossings may cause delays to motorised vehicles when the conflicting pedestrian flow is high.

2.2.3.3 Signalised crossing

A signalised crossing is a type of pedestrian crossing facility where the right-of-way of pedestrians and vehicles is indicated by signal lights. The timing of a signalised crossing can be determined with two methods: fixed-time control and pedestrian actuated control.

1. Fixed-time control

Fixed-time control uses pre-defined traffic signal timing (may vary according to the time-of-day) to give the right-of-way to pedestrians and vehicles alternately regardless of the changes of the pedestrian or vehicle flow during the specific period of time. At such facilities the pedestrians do not need to exhibit extra actions (e.g. push-button) and both pedestrians and drivers have clear instructions. This type of crossing does not require any detectors and therefore is inexpensive to construct and maintain. However, one obvious disadvantage of this crossing is that the fixed sequence of signals sometimes stops the traffic for a long time even with only few pedestrian crossing activities. Although this type of crossing is not popular in some developed countries such as the UK, it is still widely implemented in China.

2. Pedestrian actuated control

The method of the pedestrian actuated control is based on providing the priority to vehicles to minimise the vehicle delay, while the pedestrian phase is only activated based on their crossing demand (Lyons et al 2001). In the UK, there are two common types of pedestrian actuated signal control in operation: Pelican crossing and Puffin crossing.

(1) Pelican crossing

“Pelican” is an abbreviation of “pedestrian light controlled crossings”, which have a typical setup of red/amber/green signals facing motorists and red/green signal heads on the opposite side of the road to the pedestrians waiting to cross, as shown in Figure 2.3.

Figure 2.3 A typical Pelican crossing (Southampton, UK)

A Pelican crossing operates in the following way. First, a pedestrian registers a crossing demand by pushing the button mounted on the traffic signal pole and a "wait" signal is shown to the pedestrian. Then, the "red man" signal on the far-side of the crossing changes to a "green man" to indicate to the pedestrian it is safe to cross the road. Meanwhile, a red light signal is shown to the conflicting traffic. Next, when the end of the period of "green man" signal to pedestrian is approaching, the "green man" begins to flash and then turn to "red man" again, warning pedestrians that they should no longer start crossing. A typical signal timing plan of the Pelican crossing is illustrated in Figure 2.4.

	A	B	C	D	E	F	G
Vehicle Signal	Green	Steady Amber	Red			Flashing Amber	
Vehicle Instruction	Proceed if clear	Stop if safe	Wait at stop line			Give way to peds	
Pedestrian Signal	Red			Green	Flashing Green		Red
Pedestrian Instruction	Wait			Proceed if clear	Do not start to cross		Wait

Figure 2.4 A typical signal timing plan of the Pelican crossing
(Source: Walker et al 2005)

Theoretically the Pelican crossing can improve the signal control compared to the fixed-time signal as unnecessary disruption to traffic is reduced. However, if a pedestrian pushes the button but walks down the road instead of crossing it, the crossing demand cannot be eliminated and therefore the signal may stop the traffic even without any pedestrian crossing. Also, compared to the fixed-time signal, it may cause problems to blind and partially sighted people due to the “push-button” action. In addition, extra costs are required for installation and maintenance.

(2) Puffin crossing

“Puffin” is an abbreviation of “pedestrian user friendly intelligent crossings”, which have a typical setup of nearside pedestrian signals, push-button devices and 2 types of pedestrian detectors, as shown in Figure 2.5.

Figure 2.5 A typical Puffin crossing (Southampton, UK)

Puffin crossings are designed as an improvement for Pelican crossings. Leaving aside their different appearances, they differ in operations in the following ways. A Puffin crossing does not have a flashing “green man” period for pedestrians or a flashing amber period for drivers but uses nearside pedestrian signal heads and an extendable all-red crossing period activated by a pedestrian push-button request accompanied by a pedestrian detector demand (Department for Transport 1995b). The kerb-side detectors detect the presence of pedestrians attempting to cross the road so that when a pedestrian has registered a demand to cross, but subsequently finds an opportunity to cross before the commencement of the pedestrian green period, or when the pedestrian moves away from the crossing, the signal demand for pedestrian green period can be cancelled, avoiding unnecessary stop for vehicular traffic. The on-crossing detectors are used to detect if there are still crossing pedestrians on the road, so that the signal can be notified to extend the all-red period, allowing pedestrians some additional time to complete their crossing of the road (Walker et al 2005).

	1	2	3	4	5	6	7	8	9
Vehicle Signal	Green	Amber	Red						Red Amber
Vehicle Instruction	Proceed if clear	Stop if safe	Wait at stop line						Wait at stop line
Pedestrian Signal	Red			Green	Red	Ext. Red	Red		
Pedestrian Instruction	Wait			Proceed if clear	Do not start to cross	Do not start to cross	Do not start to cross		

Figure 2.6 A typical signal timing plan of the Puffin crossing
(Source: Walker et al 2005)

A Puffin crossing is more intelligent than most of others in that its advanced detectors control the traffic lights so that the clearance period is variable, providing longer crossing periods when required, which is of particular benefits to people who walk slowly such as the elderly and disabled. Meanwhile the right-of-way for vehicular traffic is resumed as soon as the crossing is clear, effectively avoiding unnecessary delays for drivers. A number of advantages of Puffin crossings have been reported by Walker et al (2005). In the UK, Puffin crossings are meant to replace Pelican crossings as the standard stand-alone pedestrian crossings (Department for Transport 1995b). Recently, China is also showing interest to Puffin crossings. A number of trials have been conducted in showcase cities to investigate the feasibility of introducing such a facility in urban traffic systems.

The main drawback of the Puffin crossing is probably its high cost. The installation of extra detectors in Puffin crossings makes them more expensive and needs additional maintenance compared to Pelican crossings (McLeod et al 2004). It is estimated that the additional cost of installing a new Puffin crossing compared with a new Pelican crossing is approximately 2500 GBP and the cost of converting an existing Pelican crossing to a Puffin crossing may be higher (Department for Transport 2006). Therefore, limited budgets should be carefully spent on the sites with the most potential effective improvement. A detailed evaluation study is usually required before the installation of such a facility.

2.2.3.4 Shared space

Compared to the common engineering solutions mentioned above, “shared space” (or “naked street”) is a new concept to deal with the interactions between pedestrians and vehicles in urban areas. It was coined in 2003 following the research by Hamilton-Baillie (2001) who identified a trend in urban design in several countries on how to deal with the adverse impacts of traffic. The core concept of shared space method is to encourage the integration of various transport modes instead of segregation, allowing different road users to negotiate right-of-way with their own instinct instead of obeying traffic signals.

Several European towns have begun to advocate the shared space concept in their urban regeneration scheme. In these pioneer towns, along with landscape redesign, speed calming and removal of conventional traffic signals, pedestrians and motorists share urban streets and use their intuition to accommodate with each other instead of obeying signals which are often confusing. An example of the transition is shown in Figure 2.4. Although the impact of this integration have not been fully understood, some positive results have shown that at these locations accident rates are decreased, emissions and fuel consumption due to vehicles’ stop-and-go phenomenon are reduced, pedestrians’ level of service is increased, urban views are improved but traffic efficiency barely falls (Shared Space Project 2008, Hamiltion-Baillie 2008).

(a) Before   (b) After

Figure 2.7 An example of the Shared Space Project (Haren, Netherlands)
(Source: Shared Space Project 2008)

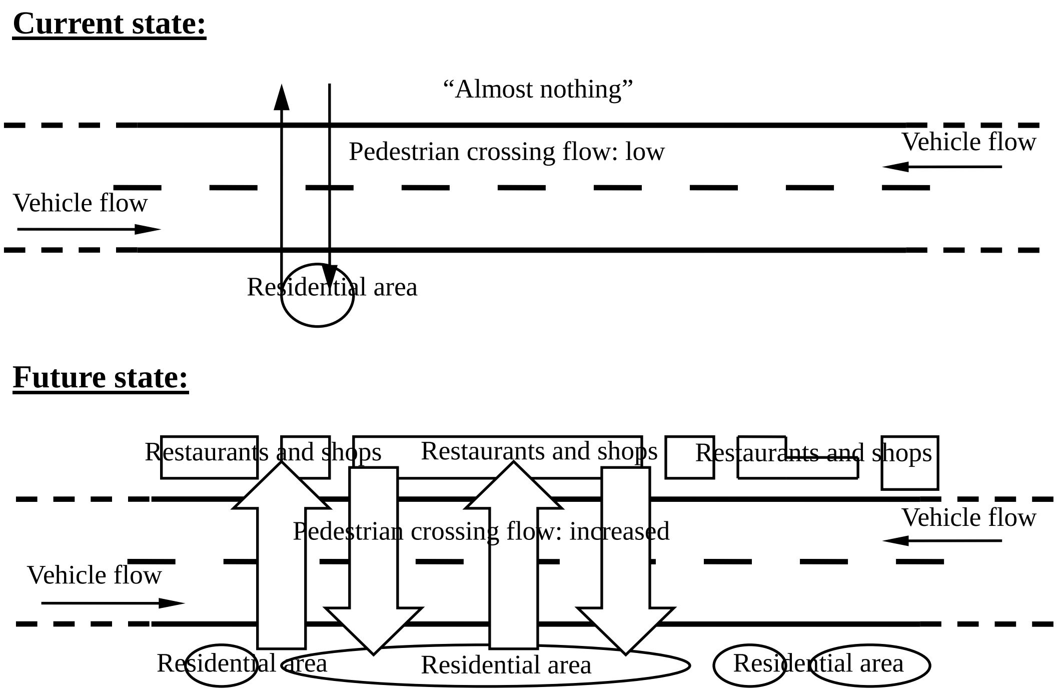
Although for most places shared space remains a hypothetical concept, it is mentioned here as it also can be seen as the “default” or “base” mode before any engineering separation method is applied to urban spaces. In some developing countries such as China where there is much lack of signalised pedestrian crossing facilities, the traffic pattern is more “aggregated” rather than “segregated” and mixed traffic is one of the main characteristics in urban street environment. Therefore, there is a significant need of developing multi-criteria evaluation method to study traffic operations under mixed traffic situations, as well as to compare system performances in unsignalised scenarios with signalised scenarios.

2.2.4 The evaluation of different treatments

The field of traffic engineering has traditionally focused on the mode of motorised transport. However, as discussed above, the walking mode has been increasingly in favour of today, due to its health benefits, environmental advantages, cost effectiveness and energy efficiency. A number of treatments have been developed to accommodate pedestrians with motorised vehicles. The growing emphasis of treatments for dealing with the interaction between the two modes demands proper methods of analysis, in order to choose the best trade-off plan for any specific scenario.

One of the challenging components of the analysis is to decide where to put a pedestrian crossing facility or what type of facility should be used. An example is that if a limited budget is given to improve the level of service for pedestrians at a number of unsignalised locations, the priority should usually be given to the site where the improvement will be the most cost-effective. Therefore, a full comparison study is needed among those unsignalised sites to determine which one currently has the worst performance and is likely to have the most improvement with a proposed plan.

Another example is that the change of land use in a certain area can result in a shift of road users’ travel pattern. For example, new buildings along either side of a street section may attract more pedestrian crossing demand, as shown in Figure 2.8. If this area is originally unsignalised but is likely to generate more interactions than before, some pedestrian crossing facilities may be required. The type of the proposed crossing facility should be decided by considering a trade-off between the two modes. As discussed before, a signalised crossing may bring controllable delays and fewer conflicts but worse environmental impact or more cost, whereas an unsignalised solution may result in better environmental impact and less cost but uncertain delays or more conflicts for the two modes. Therefore, a multi-criteria evaluation study for any proposed plan, against the base scenario, is needed to assist decision making.



Problems:

(1) Need special facilities in the future? (2) If so, what type and where should it be?

Possible solutions:

(1) Do-nothing; (2) Zebra crossing; (3) Signalised crossing (fixed-time, Puffin, etc)

Model indicators for evaluation of each solution:

(1) Efficiency: average pedestrian delay, average vehicle delay, capacity, etc
(2) Safety: frequency of conflicts, average severity of conflicts, etc
(3) Environmental impact: emissions and fuel consumption, etc

Figure 2.8 An example of evaluation of performances of different scenarios

Typically, the selection of different treatments is determined by some engineering guidelines and the evaluation of such treatments is carried out through empirical before and after studies. The methods for determining the criteria for a specific treatment are usually in accordance with local situations and therefore vary in different areas.

In the UK, the national guidance on the assessment method to be used when considering the provision and types of “stand-alone” at-grade pedestrian crossings is described in “Local Transport Note 1/95 – The assessment of pedestrian crossings” (Department for Transport 1995a). The assessment method in this note encourages decisions to be made as to whether a crossing is necessary and if so which type is appropriate by using a evaluation framework, which suggests the following main factors to have a bearing on the choice of the types of pedestrian crossing facilities:

(1) Difficulty in crossing, which can be indicated by the average time that a person normally has to wait at the site for an acceptable gap in the traffic before crossing;

(2) Vehicle delay during the process of getting through that road section;

(3) Carriageway capacity that could be reduced by introducing the proposed pedestrian crossing facility;

(4) Safety, which is assessed by reviewing historical accident record on the concerned site; and

(5) Other factors, such as traffic speeds, local representations and costs, etc.

In addition to the national guidance, some other areas in the UK also use another quantitative method called PV^2 as the basis to determine the degree of conflict between pedestrians and vehicles, where V is the “2-way total hourly flow of vehicles” and P is the “2-way total hourly flow of pedestrians crossing the road within 50 m on either side of the site at busy times” (Department for Transport 1995a). For example, in Northern Ireland, an average value of the 4 highest hourly rates of PV^2 exceeding 10^8 for an undivided road or 2×10^8 for a divided road is used for considering a certain type of pedestrian crossing; in Warwickshire, an adjusted PV^2 value considering the weights of pedestrian age type, vehicle type, waiting time, width of road, speed limit and accident record is used to justify pedestrian crossings, suggesting that the values of PV^2 greater than 0.4×10^8, 0.6×10^8 and 0.9×10^8 justifies a refuge, a Zebra crossing and a signalised crossing, respectively.

In the United States, the Highway Capacity Manual (TRB 2000) provides some quantitative analysis methods on the capacity and level of service of signalised and unsignalised crossings but does not specify the criteria for the provision of each type of pedestrian crossing. A general guidance is described in the Manual of Uniform Traffic Control Devices (FHWA 2003), which suggests that a pedestrian signal should be considered when both of the following conditions are met:

(1) “The pedestrian volume crossing the major street at an intersection or midblock location during an average day is 100 or more for each of any 4 hours or 190 or more during any 1 hour”; (FHWA 2003) and

(2) “There are fewer than 60 gaps per hour in the traffic stream of adequate length to allow pedestrians to cross during the same period when the pedestrian volume criterion is satisfied. Where there is a divided street having a median of sufficient width for pedestrians to wait, the requirement applies separately to each direction of vehicular traffic”. (FHWA 2003)

In China, there is currently a significant lack of guidance on the provision of pedestrian crossings on a national level. The warrant of the provision of a pedestrian crossing is determined mainly based on historical accident record, field survey and subjective opinions of the surveyors (Liang and Zou 2006).

From the discussion above, it can be seen that the decision on whether a pedestrian crossing is necessary and if so which type should be more appropriate needs in-depth analysis for any specific site; and the assessment methods and criteria for each type of crossing vary from place to place. However, the evaluation method based on field study is usually time-consuming and lacks microscopic details. This provides an opportunity for microscopic traffic simulation. If properly modelled and validated, microscopic simulation can provide detailed evaluation and prediction for any different site to assist existing methods for decision making. The concept of traffic micro-simulation, how they can function as a supplement for existing guidelines and current situation of micro-simulation modelling are discussed in the next section.

2.3 Microscopic traffic simulation

2.3.1 Introduction

In general, simulation is defined as “dynamic representation of some part of the real world achieved by building a computer model and moving it through time” (Drew 1968). The use of computer simulation to study traffic systems can date back to 1950s, when the development of computer technology made it possible to investigate the increasingly complex behaviour in traffic systems. Since then, simulation has become a widely used tool in transport engineering with a variety of applications from scientific research to planning, training and demonstration, etc. Nowadays, it is generally accepted that the type of traffic simulation can be classified as macroscopic, mesoscopic and microscopic. The macroscopic and mesoscopic simulation tend to model the traffic system in an aggregate fashion, for example using formulations inspired by flow equations from physics, while the microscopic simulation deals with the system in an agent based way, capturing the behaviour of each individual road user in much more details, for example, each individual’s instantaneous dynamics such as position, speed and acceleration, can be recorded by the micro-simulation system and then can be extracted and analysed.

For evaluation and comparison studies of traffic systems, compared to traditional empirical methods, which usually need collection and analysis of extensive field data with extremely high time and labour consuming, the main advantage of the micro-simulation is that it can generate various performance indicators under different combinations of system input quickly and without intruding the real system. In the case of evaluation and comparison studies of pedestrian-vehicle interaction related treatments, according to the common guidelines in some countries discussed previously, the quantitative analysis of delay, safety and environmental impact are of particular interests to assist decision making. A review of existing commonly used proprietary micro-simulation software is provided in the next section. As this research concerns the study of pedestrian-vehicle interaction behaviour, the review will be focused on their capabilities of pedestrian related modelling.

2.3.2 A review of common proprietary micro-simulation software

PTV AG (2010) claimed its flagship product VISSIM ("Verkehr In Städten - SIMulationsmodell", German for "Traffic in cities - simulation model"), which was developed in Karlsruhe, Germany, was the longest established simulation tool and the first multi-modal microscopic simulation software to include the interaction between pedestrians and vehicles.

In VISSIM, the basic traffic model ruling the movement of vehicles was developed by Wiedemann (1974) at Karlsruhe University. It is a car-following model that considers physical and psychological aspects of the drivers. The model underlying pedestrian dynamics is the social force model by Helbing and Molnár (1995). The social force model uses analogy of concepts in physics to describe microscopic dynamics of pedestrians. The motion of pedestrians, which are modelled as agents with different diameters and velocities, can be described as if "they would be subject to social forces" which are "a measure for the internal motivations of the individual to perform certain actions" (Helbing and Molnár 1995). In this model, a pedestrian's movement is determined by an attractive force towards the destination and repulsive forces from other pedestrians and obstacles. The forces are translated into accelerations used for the calculation of velocities and new positions. This model is a continuous-time-continuous-space simulation model mainly suitable for scenarios containing pure pedestrian flow, such as buildings, train stations, and pedestrian zones. Therefore, the developer of VISSIM, claimed that the areas of use of the pedestrian module in VISSIM are mainly around: (1) space optimisation and capacity planning; (2) evacuation analysis; (3) planning of mass attendee events; (4) analysis of travel and waiting times; (5) assessment of alternatives; (6) dwell time analysis (PTV AG 2012). A typical application example is that it was used to simulate extremely crowded event such as hajj in Mecca (PTV AG 2008). However, the capability of this software in scenarios with pedestrian-vehicle interactions is not as powerful as in scenarios with the sole pedestrian mode. A typical example for simulating the interaction of the two modes is described by the developer (Bönisch and Kretz 2009), which will be discussed as follows.

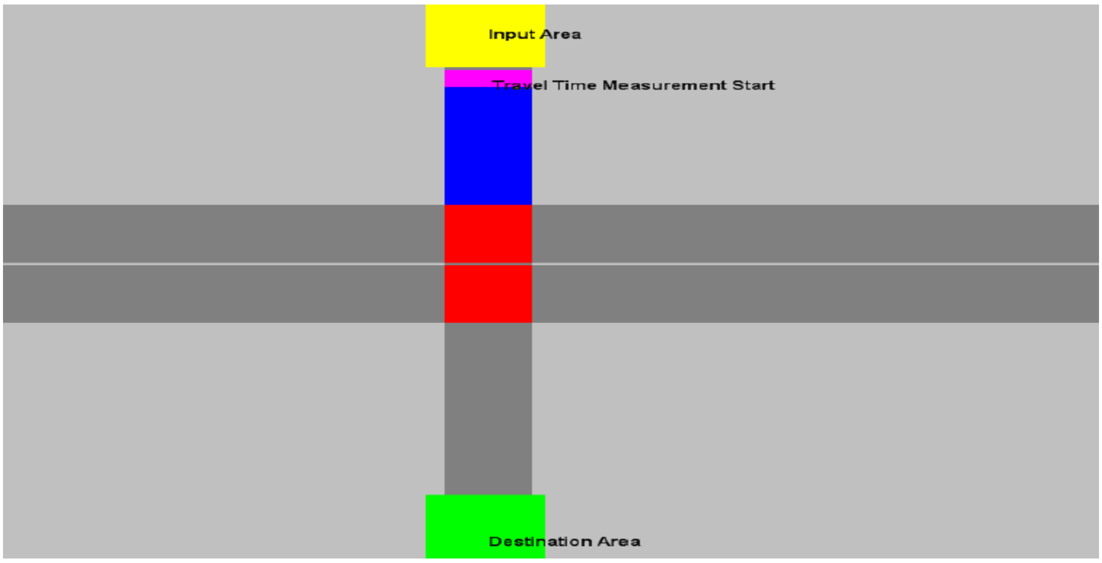


Figure 2.9 A typical example of pedestrian-vehicle interaction simulation in VISSIM (Source: Bönisch and Kretz 2009)

As shown in Figure 2.9, a conflict area needs to be predefined, usually either a signalised or unsignalised pedestrian crossing. Pedestrians are inserted with a given average frequency to the simulation on the yellow area and assigned the green area to be their destination (or vice versa for counter-flow). In the predefined conflicting area, the priority is given either to vehicles or to pedestrians. When approaching a conflicting area, pedestrians calculate if there is enough time to cross the street in time before the next vehicle arrives to decide whether or not to cross. However, sometimes pedestrians may misjudge the situation and encounter the conflicting vehicle on vehicle lanes. In this situation, the information of pedestrians being on the conflicting area is then given to the approaching vehicles, which in turn slow down, notwithstanding their right of way (Bönisch and Kretz 2009).

While this logic is sufficient to describe the friction effect of pedestrians to vehicular traffic at a pre-defined area such as a Zebra crossing or a signalised crossing, it adopts an oversimplified method to generate pedestrians only from one end of a pre-defined crossing route to the other end. This is unrealistic as not all pedestrians take the pre-defined route to cross the road in reality, Instead, given a crossing origin and a destination, they may cross anywhere along the road section, especially when the crossing origin and destination points are far away from each other. This route choice behaviour is related to many possible factors such as the presence of nearby crossing facilities, pedestrians' accumulative waiting time, number of pedestrians in a crossing group, or pedestrians' own characteristics like age and gender and it s crucial to evaluate pedestrians' overall crossing time (Chu and Baltes 2003). However, this is oversimplified in VISSIM, which can hardly be applied to model the phenomenon of pedestrian jaywalking outside the pre-defined area.

Another approach to model the pedestrian-vehicle interaction behaviour is to use "Legion for Aimsun", a combined solution for multi-modal micro-simulation. Legion is a microscopic, agent based intra-pedestrian model based on an assumption that individual pedestrian chooses his/her next step in an effort to find the best trade-off between directness of path, speed and comfort. These decisions take into account an agent's preferences and objectives as well as the context, environment and other pedestrians around them (Ronald et al 2005). The legion model is mainly applied for crowd dynamics, evacuation simulation and other scenarios concerning frequent interactions among pedestrians. Aimsun is an integrated transport modelling software, developed and marketed by Transport Simulation Systems (TSS) based in Barcelona, Spain. It integrates three types of transport models (macroscopic, mesoscopic and microscopic) into one software application. Therefore, capabilities of Aimsun include assigning traffic onto a network at a macroscopic level, adjusting Origin-Destination (O-D) matrices to reflect real-world data, the evaluation and recommendation of detector layout and locations, as well as detailed car-following and lane changing models (TSS 2012). From the version of 6.0.1, Aimsun has established a partnership with Legion to form "Legion for Aimsun", a pedestrian simulation application that is integrated inside Aimsun software as a plug-in.

As implied by the name itself, Legion for Aimsun works in such a way that the results of pedestrian simulations in Legion are loaded into Aimsun. The feature of the integrated model is primarily focused on evaluate of frictions of conflicting pedestrians to vehicular traffic, especially at locations with crossing facilities. In the integrated model, pedestrians are granted priority and a gap acceptance model is applied at crossings, ensuring that vehicles react to pedestrians (Legion 2012).

Similarly to VISSIM, the Legion for Aimsun also needs to pre-define the pedestrians' crossing route while they are crossing a road section. This is realised with the introduction of entrance centroid, exit centroid and several intermediate nodes. A snapshot from Legion for Aimsun as shown in Figure 2.10 illustrates this concept (entrance centroid in green, exit centroid in red and intermediate nodes in blue). However, it does not include the impact of various factors on pedestrians' crossing route choice. Pedestrians' behaviour is more focused on crossing facilities rather than considering their jaywalking behaviour along the entire road section.

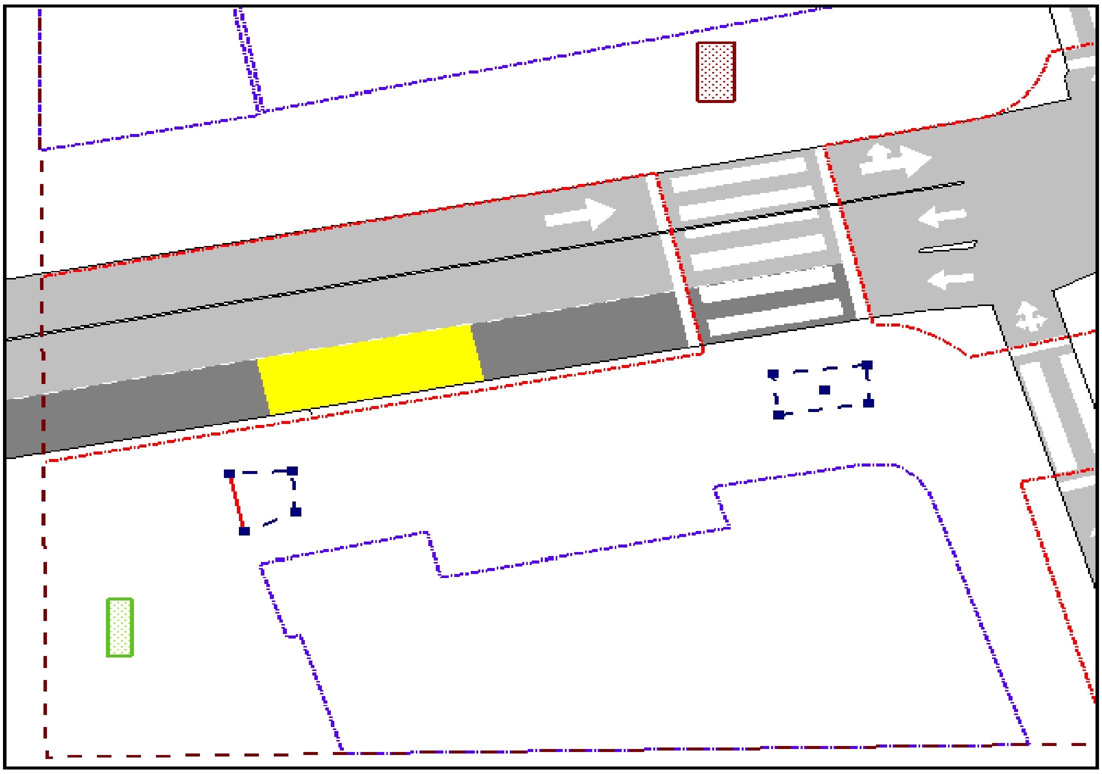


Figure 2.10 The concept of pre-defined crossing route in Aimsun
(Source: TSS 2009)

Another widely used micro-simulation software is Paramics developed by Quadstone Paramics. The module for pedestrian modelling in Paramics is called Urban Analytics Framework (UAF), which focuses on the simulation of pedestrian-vehicle interaction behaviour at crossing facilities.

In UAF, unsignalised pedestrian crossings are modelled using a combination of shared and vehicle aware space, as shown by the red area in Figure 2.11. However, how the pedestrian crossing origins and destination are represented or whether there exists a route choice model is not explicitly expressed in their technical specifications document. Signalised pedestrian crossings are modelled using combinations of shared and vehicle aware space but additional control is to be added using blocking regions connected to selected phases of signalised intersections (Quadstone Paramics 2012a).

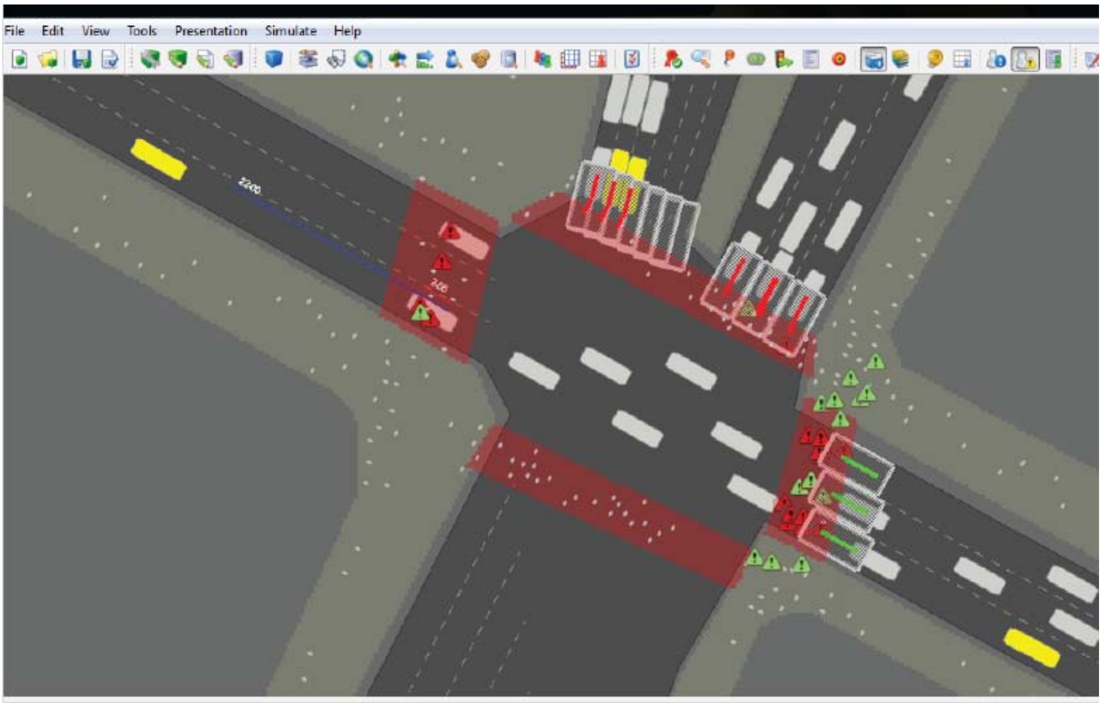
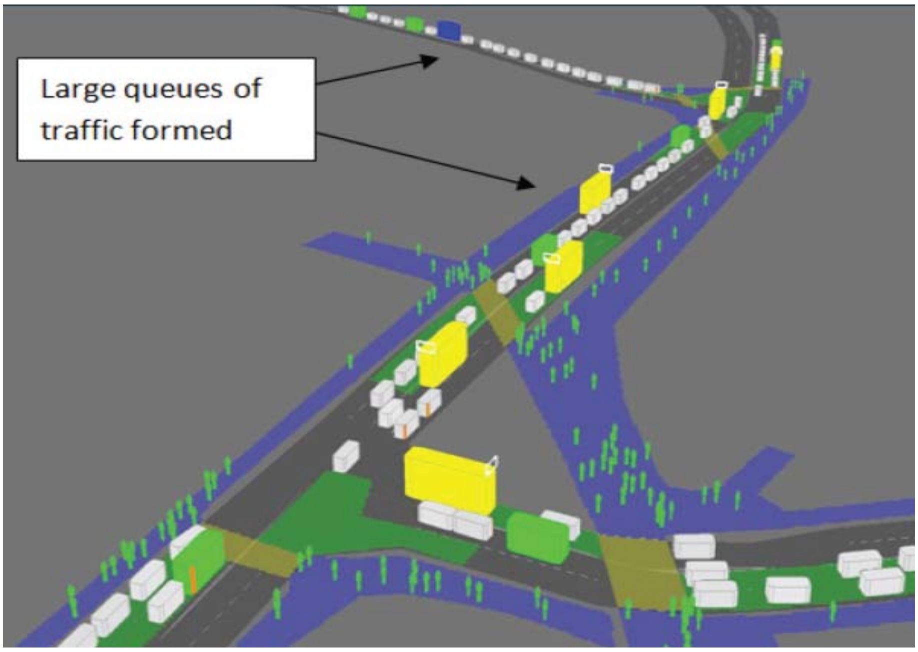
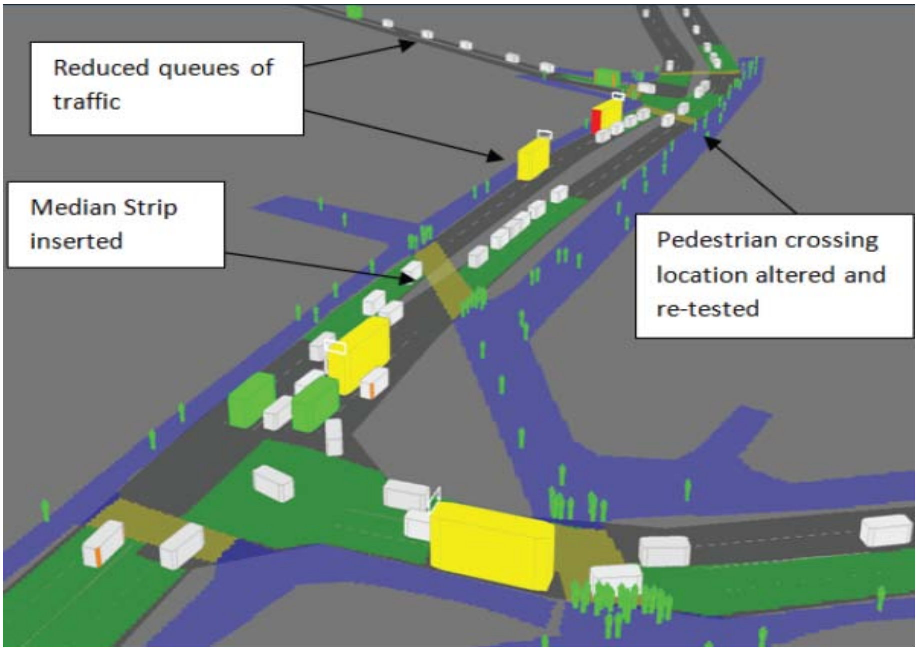


Figure 2.11 The concept of shared space used for modelling pedestrians in Paramics (Source: Quadstone Paramics 2012a)

Perhaps the most famous applications of UAF are evaluations of the effects of a number of shared space projects in Europe. A typical project is the evaluation of traffic performance at Haymarket, Newcastle Upon Tyne, UK, which compared two different designs in a shared space project in this area and gave preference to the alternative one with altered pedestrian crossing locations, based on less overall delay for all modes, as shown in Figure 2.12.



(a) Original design



(b) Alternative design with altered pedestrian crossing

Figure 2.12 Application of Paramics in shared space project in Newcastle, UK (Source: Quadstone Paramics 2012b)

In addition to the 3 popular micro-simulation packages reviewed above, there are a number of other tools capable of modelling pedestrians. However, they are more focuses on scenarios mainly containing the sole pedestrian mode. For example:

SimWalk is a micro-simulation tool mainly used for study intra-pedestrian behaviour, or their interactions with traffic at crossing facilities. This software gives focus to analysis of pedestrian flows or pedestrian panic behaviour in complex environments such as a train station. Typical applications include: pedestrian capacity and efficiency analysis of complex facilities, crowd flow analysis of public areas in urban planning, evacuation simulation of high rise buildings and other areas, and analysis of pedestrian crossings in traffic scenarios (SimWalk 2012).

PEDFLOW is a microscopic autonomous agent model of pedestrians' movement, in which each pedestrian is assumed as an agent capable to make decision making based upon a part of the observable scene local to that pedestrian (Kukla et al 2001).

PAXPORT is a pedestrian planning tool capable to model pedestrian flows and their interaction with urban environment, often used for evaluation and optimisation of building designs for pedestrian comfort such as airports, train stations and sporting venues (Halcrow Group 2008).

SimPed is a combined microscopic and macroscopic simulation tool, used to model pedestrian comfort over a different design of pedestrian facility and able to model public transport processes and their influences on pedestrian behaviour. (Daamen 2008).

However, even these models suitable for pedestrian behavioural simulations to some extent, it is yet not suitable enough to be used for analysis of pedestrian-vehicle interaction process as most of them can hardly model vehicular traffic and traffic control features (Ishaque and Noland 2007).

In conclusion, for most commercial micro-simulation software, they can provide friendly user interface and vivid animation for model output. However, there are still many limitations in the use of current micro-simulation tools to evaluate performances of pedestrian related traffic systems as stated in previous chapter.

First, existing commercial simulation tools are mostly focusing pedestrian crossing behaviour at a specific location such as a road junction. Few have combined the crossing process with the walking process, considering pedestrian behaviour in the vicinity of vehicle traffic. The pedestrian crossing route choice is lacked or oversimplified. Thus, they are not appropriate to be used at locations with less traffic disciplines, such as most developing countries like China, where pedestrian may cross anywhere on the road. This is backed up by Ishaque (2006) when he conducted a multi-modal trade-off analysis using VISSIM, who pointed that “there is a need to develop and integrate separate pedestrian algorithms in traffic micro-simulation tools or to develop new micro-simulation software that fully meets the needs to incorporate pedestrians with vehicular traffic” (Ishaque 2006).

Second, a number of commercial software use rather old models that were initially designed to simulate intra-pedestrian interactions as a basis to describe pedestrian behaviour during their interaction with vehicles, but with little changes. For example, the social force model that VISSIM utilises is developed in 1995 initially for modelling of pedestrian crowd dynamics. In such a way, a pedestrian’s movement is determined by an attractive force towards the destination and repulsive forces from other pedestrians and obstacles. However, in the context of road crossing behaviour, there are a number of factors this model did not consider. For example, the behavioural interpretation of effects of pedestrians’ accumulative waiting time, number of pedestrians in a crossing group and etc on pedestrians’ decisions of road crossing at unsignalised locations is missing in the social force model. Therefore, they are of limited use to evaluate the interaction between pedestrians and vehicles at unsignalised locations or to conduct comparison studies between the operations of various signalised and unsignalised scenarios (Schroeder 2008).

The third drawback is that most of existing software are developed based on data collected from developed countries where different modes of traffic are more segregated than integrated and road disciplines are well maintained. As this research focuses the pedestrian related problems in China, where the situation of mixed traffic is much more significant; there is a severe lack of discipline on road; and pedestrian crossing behaviour is much more aggressive, the validity of such models can be argued. Extensive calibration/validation work with local data is needed before they can be readily used. However, there is little successful calibration/validation regarding the situation in China. This is probably due to the sole modification of parameters in those models cannot fully reflect the significant behaviour difference. A modification in the internal behavioural logic is needed. Nevertheless, most proprietary simulation software do not provide users with access to the source code of their products.

In order to overcome the major constraints discussed above and achieve the objectives raised in Section 1.2, there is a need to develop new microscopic simulation model for pedestrian-vehicle interaction behaviour using the local data in China. The following section will provide a review regarding the current situation of micro-simulation modelling techniques, in order to inspire the methodology to be applied for this research.

2.3.3 Current situation of micro-simulation modelling

As the field of traffic engineering has traditionally focused on the operations of motorised transport, the modelling of intra vehicle behaviour is most common in the simulation field. A number of studies on intra vehicle behaviour such as car-following and lane-changing, etc have been carried out in past decades. There is a vast resource base on the development, calibration, validation and application regarding the vehicle behaviour models for both model developers and end users. Further, regarding the vehicle micro-simulation modelling, there is a large and vibrant community resulted from the Next Generation Simulation (NGSIM) program, which was initiated by Federal Highway Administration (FHWA), the United States Department of Transportation in the early 2000s, aiming to develop a framework for micro-simulation modelling for the future, to facilitate data sharing, technology transfer and communication between micro-simulation developers.

However, the pedestrian related micro-simulation modelling still gains little concerns and is currently not in the framework of NGSIM. Most of existing micro-simulation tools are developed considering only the sole vehicle mode; few have incorporated pedestrians or other modes of transport, Therefore, while most of existing micro-simulation tools are capable of analysing vehicular traffic, they do not offer a way for analysing the pedestrian-vehicle interaction behaviour, especially in an urban environment in developing countries such as China, where the interaction is too significant to be neglected.

Currently in the field of pedestrian behaviour modelling, many researchers followed the framework suggested by Hoogendoorn (2004) and further discussed by Airault et al (2004), Ishaque (2006) and Papadimitriou et al (2009), where pedestrian behaviour is considered to follow a structure with three levels including strategic, tactical and operational, as shown in Figure 2.13.

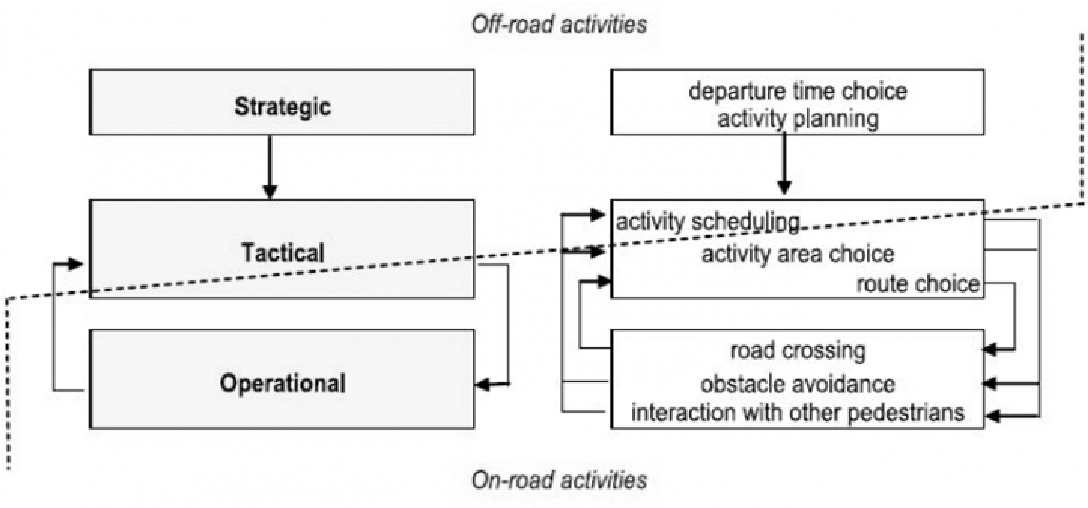


Figure 2.13 Levels of pedestrian behaviour modelling
(Source: Papadimitriou et al 2009)

The highest level mainly deals with strategic decision, such as mode choice and some other off-road activities or decisions made before the trip. The middle level concerns short term decisions on the road, such as activity scheduling and route choice. The lowest level involves operational behaviour such as instantaneous speed and direction choice. For microscopic modelling, the pedestrians' behaviour in the tactical level and operational level are of particularly interests for researchers. In reality, these two levels are usually integrated other than separated. The decisions at the operational level are often affected by the choices made at the tactical level and vice versa, for example, to use a nearby pedestrian crossing (tactical level) and then walk towards that crossing (operational level), or to decide jaywalking (tactical) and then walk towards the road (operational level) and meanwhile keep monitoring traffic condition to decide where to cross (tactical level) to save journey time.

A comprehensive modelling of pedestrian activities should ideally involve all the three aspects, that is, from the very beginning of any pedestrian's trip to the end of his/her entire journey. However, in practice, this needs extensive and accurate data which are not convenient to obtain with current technology, and complicates the modelling environment. Existing micro-simulation studies mainly focus on the lower two levels and model them separately. In the tactical level, most of existing studies focus on the pedestrian speed-density-flow relationship and their modelling objectives concern mainly the evacuation and crowd behaviour. In the operational level, current studies are mostly regarding pedestrian crossing behaviour at a specific location, usually a signalised pedestrian crossing; few have combined the crossing process with the walking process, considering pedestrian behaviour in the vicinity of vehicle traffic and vice versa, or involved the more complex interaction of the two modes at pure unsignalised areas, for example, the pedestrian jaywalking behaviour; and several observational studies do not include model development and implementation. Therefore, while current micro-simulation tools may be adequate for analysing signalised pedestrian crossings, they do not provide a way of evaluating the performances of unsignalised locations, or comparing the operational performances between signalised scenarios and unsignalised scenarios;

Therefore, although modelling the complete three levels is not practical at this moment, there is a need to extend the pedestrian behaviour at the operational level further with the tactical level. This means the modelling of pedestrian behaviour during the interaction with motorised vehicles should consider the entire process from the generation of any pedestrian's crossing demand to the end of his/her crossing journey along the entire concerned road section instead of just at a specific crossing facility. There is also a need to develop the corresponding micro-simulation model that can fully describe the interaction behaviour in both signalised and unsignalised situations.

2.4 Conclusion

This chapter reviewed the importance of walking as a transport mode, the main barriers to pedestrians in urban environment and common treatments regarding pedestrian-vehicle interaction. It identified a need to evaluate the operational performances at unsignalised locations as well as conduct comparison study between unsignalised areas and signalised areas, in order to assist decision making on the design of appropriate pedestrian related treatments to encourage the use of walking mode.

The pedestrian-vehicle interaction behaviour is closely related to the local culture. In developing countries, problems resulted from such phenomena are more severe and urgent and relative data are more likely to be obtained due to frequent interactions. Therefore, this research focuses on the situation in urban street environment in China.

The micro-simulation modelling approach is suggested to be employed for carrying out evaluation and comparison studies as it is cost-effective, non-intrusive to the real traffic system and can provide detailed indicators in terms of efficiency, safety and environmental impact. Sound and credible models are the key for any simulation study. As there is currently no appropriate models for such type of investigation, the data collection, behaviour study and model development should be conducted first. Then, the model is to be applied to solve some practical problems in the real world. The meaning of this research is two-fold. First, some fundamental understanding of pedestrian-vehicle interaction behaviour will be gained and this knowledge can form guidelines for future pedestrian transport planning and engineering. Second, this knowledge can be interpreted and integrated into existing microscopic traffic simulation models, not only making them more realistic but also making it possible to perform evaluation studies on some pedestrian related treatments in a micro-simulation environment. For short-term applications, the simulation model developed in this research can be used to evaluate traffic performances at unsignalised urban street areas or to compare operations between unsignalised and signalised solutions at any specific location. For long-term application, the result of this research can help form guidelines for future pedestrian transport planning and engineering and can also contribute to the knowledge base to develop more realistic micro-simulation models incorporating all types of road users such as motorists, cyclists and pedestrians, etc.

CHAPTER 3
METHODOLOGY

3.1 Introduction

The general methodology for study of road users' behaviour by micro-simulation modelling is well documented in existing literature, which suggests the following main procedures.

1. Behavioural study and model development: the pedestrian-vehicle interaction behaviour to be modelled in this research focuses on unsignalised mid-block locations where there is no pedestrian crossing facility. Zebra crossings are also considered as pedestrians are not given full priority and motorists hardly yield to them at such locations in China. The traffic operations in signalised scenarios are to be modelled mainly based on corresponding signal models assuming full compliance. The issues of non-compliance and misunderstanding at signalised locations are not considered. The necessary intra vehicle and pedestrian models is to be developed mainly on the basis of existing findings but with calibration and validation using local data. The interaction model is to be developed on the basis of new understandings of behaviour study and assumptions based on previous findings. Several sub-models are to be integrated and the complete model is to be validated with local data. A substantial data base is the key for model development, calibration and validation. The data collection method employed in this research will be detailed in the next section of this chapter.

2. Model implementation: during this phase, the logic model developed from behaviour studied is to be transformed into a simulation program. The C++ programming language and Visual C++ integrated development environment is to be used for the programming work. C++ supports object-oriented programming, which is generally considered as the best way to develop agent-based micro-simulation

program. Visual C++ is the most popular integrated development environment and compiler for C++ on Microsoft Windows operating system and it has been successfully applied to develop simulation programs by many researchers. The randomness of the simulation model is usually achieved by a certain random number generation method. As part of the simulation program, the random number generation module used in this research was implemented by the author based on the algorithms proposed by Knuth (1997).

3. Model validation: this process examines whether the complete simulation model is sufficiently credible to be used to study the real system. Some aggregated indicators are usually used for the validation process. Data collected for model validation should be independent from those used for model development. The similarity of the indicators from field survey and simulation is usually examined with statistical tests. In this research, Statistical Package for the Social Sciences (SPSS), a common statistical analysis tool is to be used for this purpose.

4. Model application: the developed and validated model can then be applied to solve engineering problems in the real world to showcase its value. In this research, the model is to be applied to derive new understandings from a multi-criteria evaluation study at a typical pure unsignalised area and from a comparison study of different treatments regarding pedestrian-vehicle interactions at some typical locations.

3.2 Data collection

3.2.1 Selection of data collection method

The data required for model development, calibration and validation include both the static site characteristics and the microscopic dynamics of pedestrians and vehicles. The static site characteristics can be easily obtained through field survey with some measuring tools, while the collection of dynamics of pedestrians and vehicles are more challenging. Existing research on microscopic road users' behaviour suggests the following possible approaches to obtain this type of data.

1. Video camera recording with automatic video image processing software: this method can be used to collect data of speed and distance in a relatively accurate way. However, it has two major disadvantages. First, the calibration process of the video processing software is time-consuming, especially when there are many synchronised cameras. Second, not all data of interests can be included due to limitations of the software and some of the necessary data may still have to be extracted manually.

2. Video camera recording with manual data extraction: this is a more flexible and practical method. In this method, the raw video data are collected with several synchronised video cameras. Then all data of interest can be extracted manually by comparing the relative positions of the subjects to nearby road markings, with the help of common video processing software. The drawback is that the data collection sites have to satisfy several rigorous conditions to ensure acceptable errors. Generally, the following two conditions must be fulfilled to obtain satisfactory data: (1) it is flexible for the surveyor to adjust the position, height and angle of any camera to cover all areas of interest with acceptable errors resulted from image distortion. Therefore, high buildings along data collection sites are usually needed to set up surveying cameras; (2) road markings at the data collection sites must be clearly identifiable in video images.

3. Radar or laser speed gun: these tools are usually applied to obtain any specific vehicle's instantaneous speed at any particular time of interest. Some modern laser speed gun includes a time stamp of the observation and also records the distance between the gun and vehicle. This approach balances data accuracy and processing effort but brings the need for additional data collection equipment. Other drawbacks include: (1) one speed gun can only trace one object at a particular time; (2) the accuracy of the speed gun may be unsure when the speed of the moving object is low and therefore it is not appropriate to collect behavioural data of vehicles in congestion or slow-moving pedestrians; (3) the researchers cannot trace back to the original traffic scenarios when they intend to consider more variables for analysis.

4. Instrumented vehicle: this is a new technology for driving behaviour studies in recent years. With multiple sensors, video cameras and computers equipped in an

instrumented vehicle, the very detailed dynamics of the vehicle and the driver can be obtained over a long period of time or distance, with a relatively high accuracy. This method is advantageous to study car-following behaviour because the dynamics of leading and following vehicles (relative to the instrumented vehicle) can be obtained at the same time. However, when researchers intend to collect more information outside the leader-follower system at the concurrent time, this method usually needs to be complemented with other devices such as roof-level video cameras.

In summary, as the dynamic data required in this research are mostly calculated based on the instantaneous positions of vehicles and pedestrians, it is necessary to record their behaviour over a large area and during a long time. The method of video camera recording with manual extraction is selected in this research to collect and extract all necessary data as it provides a good trade-off among flexibility, accuracy and efficiency. Besides, it also provides an opportunity for researcher to trace back the original event scenarios.

3.2.2 Data collection

As this research focuses on pedestrian-vehicle interaction behaviour in an urban street environment, most of the data were collected at 2-way-2-lane street sections with low speed limit of 30 km/h, where the interactions between the two modes were more frequent. The data collection sites were carefully selected to fulfil the requirements for the video camera method discussed in the previous section.

The static characteristics of each site (e.g. road geometry, length of road markings and width of lanes, etc) were surveyed using a measuring wheel with a range of 10 km, a minimum scale of 0.1 m and an error of ±1.0 m per 1.0 km, as shown in Figure 3.1.

Figure 3.1 Survey of site characteristics using a measuring wheel

The static and dynamic characteristics of pedestrians and vehicles were recorded with several roof-level synchronised cameras. A typical set-up of the cameras for raw video data collection is illustrated in Figure 3.2.

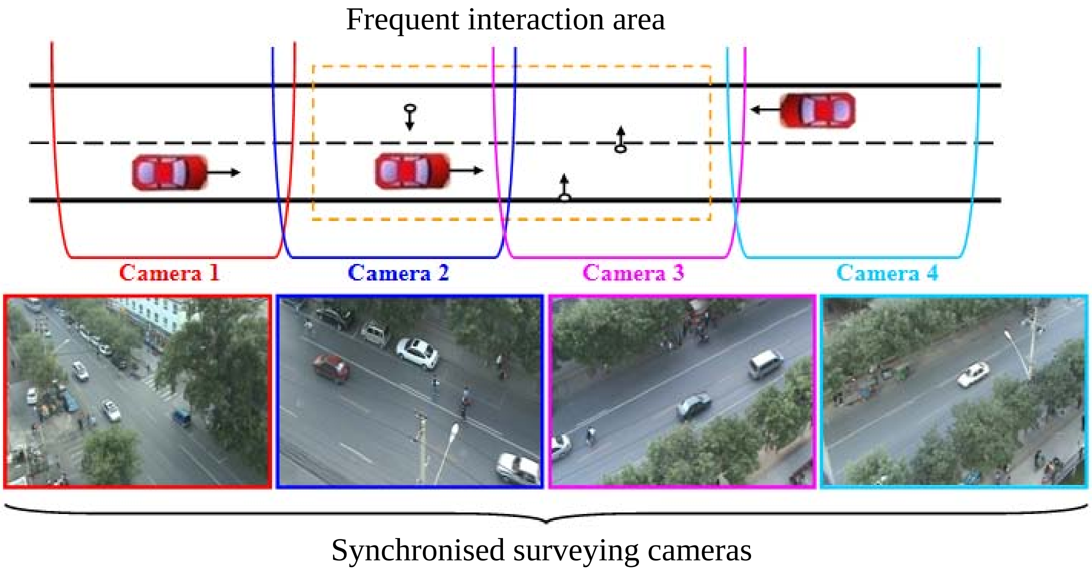


Figure 3.2 A typical set-up of the cameras for raw video data collection

The dynamic traffic scenario at each site was recorded by four synchronised video cameras, with 25 Hz recording frequency, covering a road section of approximate 300 m. Two cameras in the middle covered the frequent interaction area for most of the interaction behaviour data and two others covered both sides of the site with relatively long distances to record the concurrent vehicle dynamics. The frequent interaction area in each site was ensured sufficiently far from nearby intersections to allow vehicles have enough distance to reach their desired status.

3.2.3 Data extraction

The site characteristics were surveyed before setting up the video cameras each time. Therefore, the analyst can learn the relative position of any road user to nearby road markings from the video pictures to extract their dynamic characteristics.

The static data of any object such as the pedestrian's type of gender and age, his/her crossing Origin-Destination pair (*O-D*), and the vehicle's type can be directly observed from the video pictures. The dynamic traffic data such as the distance and instantaneous speed, etc of a pedestrian or a vehicle can be calculated based on the pedestrian or vehicle's position time series data. The following paragraphs will discuss the extraction of dynamic data using vehicle instantaneous speed at a given time as an example. Any other variables can be extracted from the video pictures following the same principle.

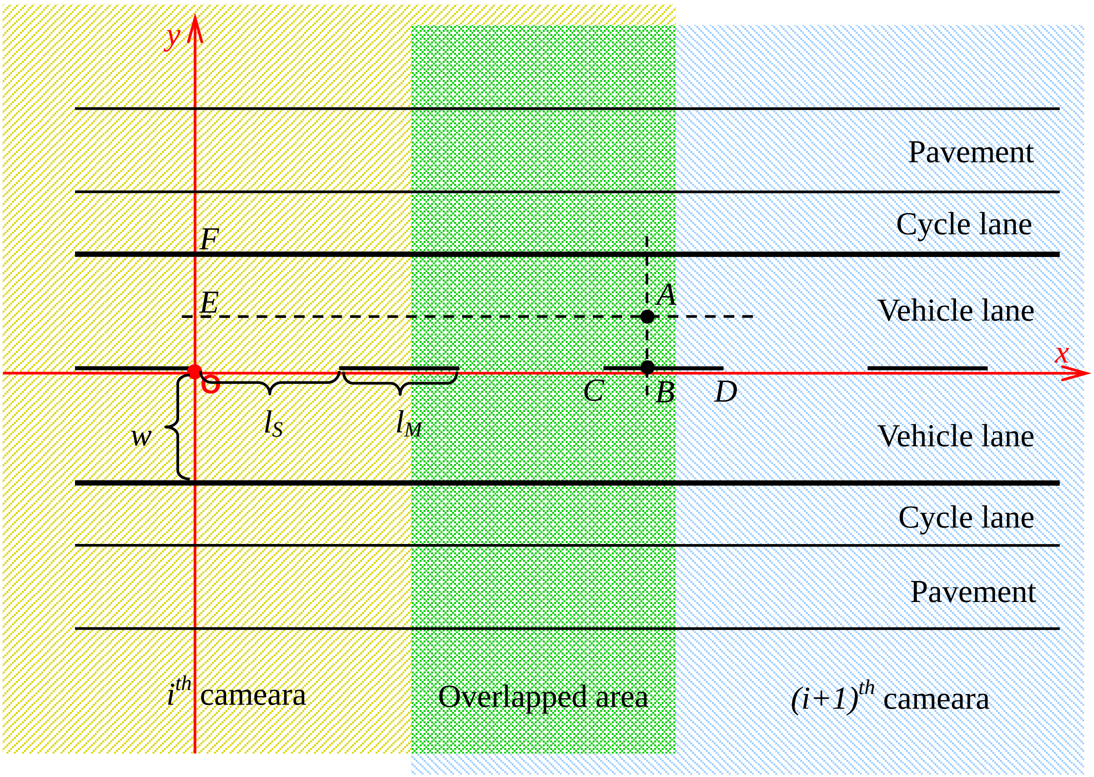


Figure 3.3 A schematic diagram of data extraction method

As shown in Figure 3.3, as the length of any road marking (or the interval space between any two adjacent markings, the width of any lane, etc, hereafter) has been measured, for a designated origin point (*Point O* in Figure 3.3), the coordinate of any point in the video picture can be obtained by comparing its screen coordinate to the screen coordinate of an adjacent road marking.

For example in Figure 3.3, assuming that *O* is the origin point with a screen coordinate and a real-world coordinate both to (*0*, *0*); *A* is an arbitrary point in the video picture with a screen coordinate of (x_A, y_A); *B* is the foot of perpendicular from *A* to an adjacent road marking, with a screen coordinate of (x_B, y_B) (*B* is achievable by adding a reference line parallel to the front bumper of an adjacent vehicle); *C* and *D* represent the two ends of the road marking, with screen coordinates of (x_C, y_C) and (x_D, y_D) respectively, the real-world coordinate of *A* in the *x* direction, x'_A, can then be calculated by Equation 3.1.

$$x'_A = l_S + l_M + l_S + \frac{x_B - x_C}{x_D - x_C} \cdot l_M \tag{3.1}$$

Where,
l_S is the length of the interval space between two adjacent road markings;
l_M is the length of the road marking.

Similarly, the real-world coordinate of *A* in the *y* direction, y'_A, can be calculated by Equation 3.2.

$$y'_A = \frac{y_E}{y_F} \cdot w \tag{3.2}$$

Where,
y_E is the screen coordinate of *E* in the *y* direction;
y_F is the screen coordinate of *F* in the *y* direction;
w is the width of the vehicle lane.

Therefore, at any given time, the real-world coordinate of any point in video pictures can be obtained, which then can be used as basis to calculate an object's dynamic parameter (when required, the length and width of each vehicle can also be obtained by comparing the relative position of its outline to an adjacent road marking).

Taking the instantaneous vehicle speed at point *A* as an example, when a vehicle arrives at point *A* ($x_{Veh} = x_A$) at time *t*, its instantaneous speed at this point can be estimated by the average speed between *DB* or *BC*, whichever is shorter. In Figure 3.3, assuming *BC* is shorter than *DB* and $x_{Veh} = x_C$ at time t_1, the vehicle's instantaneous speed $v_{Veh}(t)$ can then be estimated by Equation 3.3.

$$v_{Veh}(t) = \frac{x_B - x_C}{t_1 - t} \tag{3.3}$$

3.2.4 Error analysis

Since the data collection and extraction method employed in this research involved both system error and observation error, an analytical method was not suitable for error analysis. Therefore, the error was estimated using an empirical method.

A subject driver was instructed to drive a vehicle with a constant speed of 30 km/h along a road for several times. His instantaneous speed at a particular location was surveyed using both a road side radar speed gun and the video camera method discussed previously. Considering the accuracy of the radar speed gun was relatively high, the two sets of data were compared to examine whether the results from the video method was similar to those from radar speed gun. 50 pairs of speed data were obtained during the validation process.

Statistical tests showed that there was not enough evidence to reject that the speed data from radar speed gun and video camera yielded to independent normal distributions (see Appendix I). Therefore, the following 2-sample t test assuming different variances was carried out to check the similarity of the means from the two samples.

Test method: 2-sample t test.

Test hypotheses: H_0: the mean values of the speed data from radar speed gun and video camera method are equal; H_1: the mean values of the speed data from radar speed gun and video camera method are not equal.

The result of the test is shown in Table 3.1, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.819). Therefore, it is reasonable to accept H_0. The video camera method is sufficiently reliable to be used to obtain microscopic behaviour data.

Table 3.1 The result of 2-sample t test for vehicle instantaneous speed

Equal variances not assumed

	t-test for Equality of Means						
						95% Confidence Interval of the Difference	
	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Speed	-.230	68.884	.819	-.06460	.28146	-.62611	.49691

3.3 Conclusion

This chapter provided an overview of the general methodology to study road users' behaviour by micro-simulation. Emphasis was given to data collection method for model development, calibration and validation. The method of video camera recording with manual extraction was proposed to be applied in this research due to its balance among flexibility, accuracy and efficiency. The method for raw video data collection and extraction of the according quantitative data were detailed using vehicle instantaneous speed as an example. The error analysis showed that the proposed data collection method was sufficiently reliable to be used to obtain microscopic behaviour data. In this research, most of the interaction data were collected and extracted using the method described in this chapter. In addition, this method was also used to collect other necessary data to develop, calibrate and validate intra vehicle or pedestrian models at different types of locations. The sample size and use of each type of data will be detailed in subsequent chapters where the relevance can be most immediately seen.

CHAPTER 4
VEHICLE BEHAVIOUR AND MODELS

4.1 Introduction

Before incorporating pedestrian objects into the microscopic simulation, one should firstly ensure that the simulation is credible solely with vehicle objects. Therefore, a set of intra vehicle models have to be defined, calibrated and validated to properly represent the pure vehicle behaviour in accordance with the local situation. This chapter discusses the development, calibration and validation of such vehicle models applied in this research.

4.2 Vehicle static characteristics

Vehicle static characteristics are closely related to the local application conditions and include characteristics such as vehicle type and effective size. The definitions of these parameters are discussed in this section.

1. Type

The type of a motor vehicle is the most basic parameter to be defined as it relates to kinematic behaviour of acceleration and deceleration. Also, the type of a vehicle determines its size, which affects its road space occupancy.

Existing literature suggests that for simulation studies in urban street environment in China, it is appropriate that vehicles are classified into the following 4 categories, as shown in Table 4.1. According to the situation in China, buses and coaches are sub-categorised into regular and articulated ones. The percentage of each type is to be calibrated by the user according to the specific situation of the focus area when conducting the simulation study.

Table 4.1 Classification of vehicle types in this research

<table>
<tr><td colspan="2">Type</td><td>Description</td><td>Examples</td></tr>
<tr><td colspan="2">Light Vehicles (LV)</td><td>3-4 wheels</td><td>Passenger cars</td></tr>
<tr><td colspan="2">Medium Commercial Vehicles (MCV)</td><td>2 axles and more than 4 wheels</td><td>Vans, pick-ups</td></tr>
<tr><td colspan="2">Heavy Commercial Vehicles (HCV)</td><td>More than 2 axles</td><td>Heavy good vehicles</td></tr>
<tr><td rowspan="2">Buses and coaches</td><td>Regular (BCR)</td><td rowspan="2">Buses or coaches</td><td rowspan="2">Buses or coaches</td></tr>
<tr><td>Articulated (BCA)</td></tr>
</table>

(Source: Du 2008)

2. Effective size

The effective size of a vehicle in micro-simulation is characterised by its length, width and a margin, which is defined as "a short distance from its rear into which its following vehicle is not willing to intrude even when it is at rest" (Gipps 1981).

The lengths of vehicles of a certain type can be described by a normal distribution. The parameters of such normal distributions were calibrated by the author with the observed data in Beijing, China using the video camera method described in Chapter 3. The results are shown in Table 4.2. As there is no significant differences of the values of vehicle widths within one vehicle type, for simplification all vehicles in a certain type are assigned the same width, which is the average value from field observation. The sizes of buses and coaches are relatively standard within the same sub-category. Therefore, constant values averaged from field observation are assigned accordingly.

Table 4.2 Observed vehicle lengths and widths

Type	Sample size	Average length (m)	Standard deviation of length (m)	Minimum length (m)	Maximum length (m)	Average width (m)
LV	172	4.12	0.56	2.47	5.16	1.86
MCV	45	5.03	0.51	3.78	6.52	2.34
HCV	32	9.82	1.36	5.86	11.60	2.50
BCR	25	12.00	0.00	12.00	12.00	2.50
BCA	15	18.00	0.00	18.00	18.00	2.50

Besides, 155 datasets were collected by the author to calibrate the value of the vehicle margin suitable for simulation study in China. The following statistical test was carried out to check the type of distribution of the vehicle margins.

Test method: 1-sample K-S test.

Test hypotheses: H_0: the data of vehicle margins yield to a normal distribution; H_1: the data of vehicle margins do not yield to a normal distribution.

The result of the test is shown in Table 4.3, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.982). Therefore, it is reasonable to accept H_0 that the data of vehicle margins yield to a normal distribution.

Table 4.3 The result of 1-sample K-S test for vehicle margins

		VehMargin
N		55
Normal Parameters[a,b]	Mean	1.4081
	Std. Deviation	.41671
Most Extreme Differences	Absolute	.063
	Positive	.063
	Negative	-.050
Kolmogorov-Smirnov Z		.464
Asymp. Sig. (2-tailed)		.982

a. Test distribution is Normal.

b. Calculated from data.

In addition, it was also found from field survey that the minimum and maximum values of vehicle margin were 0.54 m and 2.33 m respectively. Therefore, the vehicle margin can be described with a truncated normal distribution: $l_M \sim N$ (1.41, $(0.42)^2$), ($0.54 \leq l_M \leq 2.33$).

4.3 Vehicle free flow characteristics

In micro-simulation, vehicle can be defined to be in a free flow condition if it moves "without any unimpeded vehicles or other objects such as pedestrians or signals" (Wu 1994). For example, a queue leader discharging from a traffic signal is regarded to be in a free flow condition. In this condition, the dynamics of the vehicle are not governed by car-following model, but need to be determined by some other mechanisms. The main characteristics to be determined under this condition include its desired speed, and free acceleration and deceleration. The method for deciding the values of these 3 parameters is discussed in this section.

4.3.1 Desired speed

Desired speed is the speed a vehicle attempts to keep when it is in a free flow condition and it is only limited by the capability of the vehicle and the willingness of its driver. Many factors may have influences on vehicle's desired speed, such as vehicle characteristics, trip characteristics, vehicle ownership and speed limit, etc (Hirsh 1986). However, there is no single model considering all possible factors. It is commonly suggested that vehicle desired speed can be described with a normal distribution (Du 2008, Wu 1994, Leutzbach and Wiedemann 1986). Further, studies have shown that the ratio of standard deviation to mean in such a distribution seems to be constant over a wide range of speeds (Mintsis 1982, McLean 1982, Taylor 1976). Du (2008) suggested that the values found by Mintsis (1982) were suitable for simulation studies in the urban street environment in China, as shown in Table 4.4.

Table 4.4 The ratio of standard deviation to mean for vehicle desired speed

<table>
<tr><td colspan="2" rowspan="2">Vehicle type</td><td colspan="2">The ratio of standard deviation to mean (k)</td></tr>
<tr><td>Single carriageways</td><td>Dual carriageways</td></tr>
<tr><td rowspan="2">Light vehicles</td><td>LV</td><td rowspan="2">0.137</td><td rowspan="2">0.125</td></tr>
<tr><td>MCV</td></tr>
<tr><td rowspan="3">Heavy vehicles</td><td>HCV</td><td rowspan="3">0.118</td><td rowspan="3">0.110</td></tr>
<tr><td>BCR</td></tr>
<tr><td>BCA</td></tr>
</table>

(Source: Mintsis 1982)

Therefore, the probability density function of vehicle desired speed can be expressed by Equation 4.1, where the parameters of mean, minimum and maximum are to be provided by the user with local values when carrying out the simulation study. The user can apply the video camera method discussed in Chapter 3 to collect the speed data from the on-road vehicles in apparent free flow conditions in order to calibrate these 3 parameters.

$$f\left(v_{Veh,DSR}\right) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(v_{Veh,DSR}-\mu)^2}{2\sigma^2}}, & v_{Veh,DSR,min} \le v_{Veh,DSR} \le v_{Veh,DSR,max} \\ 0, & v_{Veh,DSR} < v_{Veh,DSR,min} \; or \; v_{Veh,DSR} > v_{Veh,DSR,max} \end{cases} \quad (4.1)$$

Where,

$v_{Veh,DSR}$ is the vehicle desired speed, which is a random variable, (m/s);

$f(v_{Veh,DSR})$ is the probability density function of $v_{Veh,DSR}$;

μ is the mean value of $v_{Veh,DSR}$, which is calibrated from field observation, (m/s);

σ is the standard deviation of $v_{Veh,DSR}$, which is calculated by $\sigma = k\mu$, where k is the ratio of the standard deviation to mean with the values in Table 4.4, (m/s);

$v_{Veh,DSR,min}$ is the minimum value of $v_{Veh,DSR}$, which is to be calibrated from field observation, (m/s);

$v_{Veh,DSR,max}$ is the maximum value of $v_{Veh,DSR}$, which is to be calibrated from field observation, (m/s).

4.3.2 Free acceleration

Free acceleration is the acceleration rate the vehicle wishes to undertake when it intends to increase its speed in a free flow condition. When carrying out a simulation study in the urban street environment in China, Wu (2001) suggested the use of the model proposed by Anderson et al (1968) considering the combination of its satisfactory accuracy and relative simplicity. This model can be expressed by Equation 4.2. The values of the parameters in this model suggested by Wu (2001) according to the local situation in China are listed in Table 4.5.

$$a_{Veh,FRF}(t) = a_{Veh,init} \cdot \sqrt{1 - \frac{v_{Veh}(t)}{v_{Veh,DSR}}}, \qquad 0 \le v_{Veh}(t) \le v_{Veh,DSR} \tag{4.2}$$

Where,

$a_{Veh,FRF}(t)$ is the free flow acceleration rate (≥ 0) of this vehicle at a given simulation time t, (m/s^2);

$a_{Veh,init}$ is the initial acceleration rate (≥ 0), which should be calibrated according to local situation, (m/s^2);

$v_{Veh}(t)$ is the speed of this vehicle at time t, (m/s);

$v_{Veh,DSR}$ is the desired speed of this vehicle, (m/s).

Table 4.5 Parameters for vehicle free acceleration model

Vehicle type		$a_{Veh,init}$ (m/s^2)
Light vehicles	LV	1.95
	MCV	
Heavy vehicles	HCV	2.15
	BCR	
	BCA	

(Source: Wu 2001)

4.3.3 Free deceleration

A vehicle in a free flow condition stopped by traffic lights usually reduces its speed with a free deceleration rate, whose value varies in accordance with the driver's sense of safety and comfort during the braking process (Wu 1994). A review of free deceleration models pointed out that the linear deceleration with distance provides better results when applied in micro-simulation (Wu 1994). The expression of this model is shown in Equation 4.3. In this research, the parameters of this model are defined with the values suggested by Wu (1994), as shown in Table 4.6, as existing research similar to this project has shown that such values are suitable for micro-simulation studies in China (Du 2008, Sui et al 2008, Wu 2001).

$$\begin{cases} d_{Veh,FRF}(t) = d_{Veh,final}\left[1 - \frac{x_{Veh}(t)}{x_{Veh,max}}\right] \\ x_{Veh,max} = \frac{v_{Veh,DSR}^2}{d_{Veh,final}} \end{cases}, \quad 0 \le x_{Veh}(t) \le x_{Veh,max} \qquad (4.3)$$

Where,

$d_{Veh,FRF}(t)$ is the free flow deceleration of this vehicle (≥ 0) at time t, (m/s^2);

$d_{Veh,final}$ is the final deceleration (≥ 0) right before stopping, defined with values in Table 4.6, (m/s^2);

$x_{Veh}(t)$ is distance between vehicle front bumper and final stopping position at t, (m);

$x_{Veh,max}$ is the distance between the vehicle front bumper and its final stopping position right at the time when the speed change is just being affected, (m);

$v_{Veh,DSR}$ is the vehicle desired speed, (m/s).

Table 4.6 Parameters for vehicle free deceleration model

Vehicle type		$d_{Veh,final}$ (m/s^2)
Light vehicles	LV	1.798
	MCV	
Heavy vehicles	HCV	1.589
	BCR	
	BCA	

(Source: Wu 1994)

4.4 Vehicle generation model

In micro-simulation, vehicles are introduced into the simulation from the boundaries of the system at specific simulation times. The vehicle generation model decides when to generate a new vehicle object into the simulation and what initial status this new generated vehicle should be assigned to. After that, the dynamics of this vehicle will be governed by a car-following model, which will be discussed in Section 4.5.

4.4.1 Initial time headway

In micro-simulation, the time at which to introduce a new vehicle into the simulation area is determined by an initial time headway model. The vehicle time headway is “the time interval of passing a reference point between 2 consecutive vehicles and usually refers to front-bumper-to-front-bumper time headway” (Zheng 2003). The initial time headway is the time headway between 2 consecutive vehicles with the vehicle generation point as the reference point. Therefore, if the time headway of any 2 consecutive vehicles can be determined during the whole simulation time, the generation time of any vehicle can be decided. Many initial time headway models have been proposed and the most popular ones applied in existing simulation programs are discussed as follows.

1. Exponential distribution model

This model assumes that the vehicle time headway is a random variable and yields to an exponential distribution, as shown in Equation 4.4.

$$f(t) = \begin{cases} \lambda e^{-\lambda(t-\mu)}, & t \geq \mu \\ 0, & t < \mu \end{cases}, \quad \lambda > 0 \tag{4.4}$$

Where,

t is the vehicle time headway, (s);

$f(t)$ is the probability density function of t;

μ is the minimum value of t, which is provided by the user, (s);

λ is the other parameter in this model and from the property of exponential distribution the following relationship can be derived by Equation 4.5.

$$\mu + \frac{1}{\lambda} = E(t), \quad \lambda > 0 \tag{4.5}$$

Therefore, λ can be derived by Equation 4.6.

$$\lambda = \frac{1}{E(t)-\mu}, \quad E(t) > \mu \tag{4.6}$$

Where,

E(t) is the expectation of *t*, which is calibrated from field observation by the user, (s).

This model was originally proposed by Adams (1936) and had been widely applied in practice due to its simplicity and accuracy (Du 2008, Faghri and Egyhaziova 1999, Wu 1994, Chin 1983). In addition, Du (2008) suggested that the use of parameter μ with the value of 0.5 was reasonable to the local situation in China.

2. Lognormal distribution model

Another commonly used model is the lognormal distribution model, which assumes that the vehicle time headway yields to a lognormal distribution, as shown in Equation 4.7. This model is suitable for a high flow traffic stream on motorway (Mei and Bullen 1993). However, the calibration of this model is more complex as there are 3 parameters to be confirmed.

$$f(t) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma t} exp\left[-\frac{(lnt-\mu)^2}{2\sigma^2}\right], & t > 0 \\ 0, & t \le 0 \end{cases}, \quad -\infty < \mu < +\infty, \sigma > 0 \tag{4.7}$$

Where,

t is the vehicle time headway, (s);

f(t) is the probability density function of *t*; and according to the property of the lognormal distribution, the 2 parameters μ and σ can be derived by Equation 4.8.

$$\begin{cases} \mu = lnE(t) - \frac{1}{2}\left[1 + \frac{D(t)}{[E(t)]^2}\right] \\ \sigma = \sqrt{ln\left[1 + \frac{D(t)}{[E(t)]^2}\right]} \end{cases} \tag{4.8}$$

Where,

E(t) is the expectation of *t*, to be calibrated according to the local data, (s);

D(t) is the variance of *t*, to be calibrated according to local data, (s).

3. Mixed distribution model

A more sophisticated model is the mixed distribution model, as shown in Equation 4.9, in which the proportion, type and parameters of each distribution have all to be determined by the user. This model was proposed by researchers (Griffiths 1991, Buckley 1962, Schuhl 1955) who believed that there was more than one type of distribution for vehicle time headways in the traffic stream. For example, one distribution was used to describe vehicles in free flow conditions and another for vehicles in car-following conditions. Some of these models have shown that the statistical fits can be improved. However, they are less popular for simulation modelling because of the complexity of the form and the number of parameters to be calibrated.

$$f(t) = \sum_{i=1}^{n} r_i f_i(t), \quad \sum_{i=1}^{n} r_i = 1, \quad n > 1 \; and \; n \in \mathbb{N} \tag{4.9}$$

Where,

t is the vehicle time headway, (s);

f(t) is the probability density function of *t*;

$f_i(t)$ is the i^{th} sub probability density function of *t*;

r_i is the proportion of $f_i(t)$;

n is the total number of sub probability density functions to describe *t*.

In conclusion, various initial time headway models can be found in existing practices. The selection of such a model is related to the study purpose and local traffic characteristics. Since this research is similar to the one conducted by Du (2008), who studied the interaction behaviour of vehicle and cyclists in the urban street environment in Beijing, China, this research follows her suggestion using the model expressed in Equation 4.4 and value of 0.5 for parameter μ in this model. The other parameter λ can be easily calibrated from the traffic counts and the total corresponding period of time in the study area.

4.4.2 Initial dynamics

When a vehicle object enters the simulation system at its generation time, the initial status of this vehicle should be defined so that together with the car-following model the status of this vehicle at any given time during the whole period of simulation can be decided. This section discusses the methods to assign initial values to vehicle status parameters which are relative to simulation modelling.

1. Initial position

The position of the vehicle object in the micro-simulation is usually indicated by a coordinate (*Lane ID*, Δx), where *Lane ID* specifies which lane the vehicle is currently in and Δx is the vehicle departing distance from the generation point along the driving direction of this vehicle lane. If the physical space ahead of the newly generated vehicle is available, the value of its initial Δx is defined with 0, meaning this vehicle is about to enter the simulation area; otherwise Δx is below 0, meaning this vehicle is queuing outside the simulation area. This logic can be described by the algorithm shown in Figure 4.1.

```
/*
Assuming Veh_i enters the simulation system at time t;
its leading vehicle is marked as Veh_{i-1} (if exists);
the Δx of Veh_i at this time can be determined by the following algorithm:
*/

if (Veh_{i-1} does not exist)
        Δx_{Veh,i}(t) = 0;
else
        {
                if (Δx_{Veh,i-1}(t) - l_{V,i-1} - l_{M,i-1} ≥ 0)
                        Δx_{Veh,i}(t) = 0;
                else
                        Δx_{Veh,i}(t) = Δx_{Veh,i-1}(t) - l_{V,i-1} - l_{M,i-1} ;
        }

/*
Where,
Δx_{Veh,i}(t) is the Δx of Veh_i at time t, (m);
Δx_{Veh,i}(t) = 0 means the new generated vehicle is about to enter this lane, (m);
Δx_{Veh,i}(t) < 0 means the new generated vehicle is queuing outside the boundary of this lane, (m);
Δx_{Veh,i-1}(t) is the Δx of Veh_{i-1} at t, (m); and
l_{V,i-1} is the vehicle length of Veh_{i-1}, (m);
l_{M,i-1} is the vehicle margin of Veh_{i-1}, (m).
*/
```

Figure 4.1 The algorithm for determining vehicle initial Δx

2. Initial speed and acceleration

The initial speed of a newly generated vehicle is dependent on this vehicle's desired speed, as well as the distance, speed and acceleration of the vehicle ahead of it, if such a leading vehicle exists; otherwise, the initial speed of the vehicle is set to be its desired speed. The definition of the vehicle desired speed has been discussed in Section 4.2. The algorithm to describe a vehicle's initial speed is shown in Figure 4.2. This algorithm is further illustrated in the flow chart in Figure 4.3. The core of this algorithm is that the allocation of the initial speed to this vehicle cannot result in a collision between this vehicle and the one ahead of it.

```
/*
Assuming Veh_i enters the simulation system at time t;
 its leading vehicle is marked as Veh_{i-1} (if exists);
 the speed of Veh_i at this time can be determined by the following algorithm:
*/

if (Veh_{i-1} does not exist)
        v_{Veh,i}(t) = v_{Veh,DSR,i} ;
else
        {
                Calculate the maximum allowed speed for Veh_i at time t, marked as v_{Veh,max,i}(t);
                if (v_{Veh,max,i}(t) ≥ v_{Veh,DSR,i})
                        v_{Veh,i}(t) = v_{Veh,DSR,i} ;
                else
                        v_{Veh,i}(t) = v_{Veh,max,i}(t);
        }

/*
Where,
v_{Veh,i}(t)  is the speed of Veh_i at t, (m/s);
v_{Veh,DSR,i} is the desired speed of Veh_i, (m/s);
v_{Veh,max,i}(t) is the maximum allowed speed for Veh_i at t, (m/s); and
the method for calculating v_{Veh,max,i}(t) is discussed in the paragraphs following Figure 4.2
*/
```

Figure 4.2 The algorithm for determining vehicle initial speed

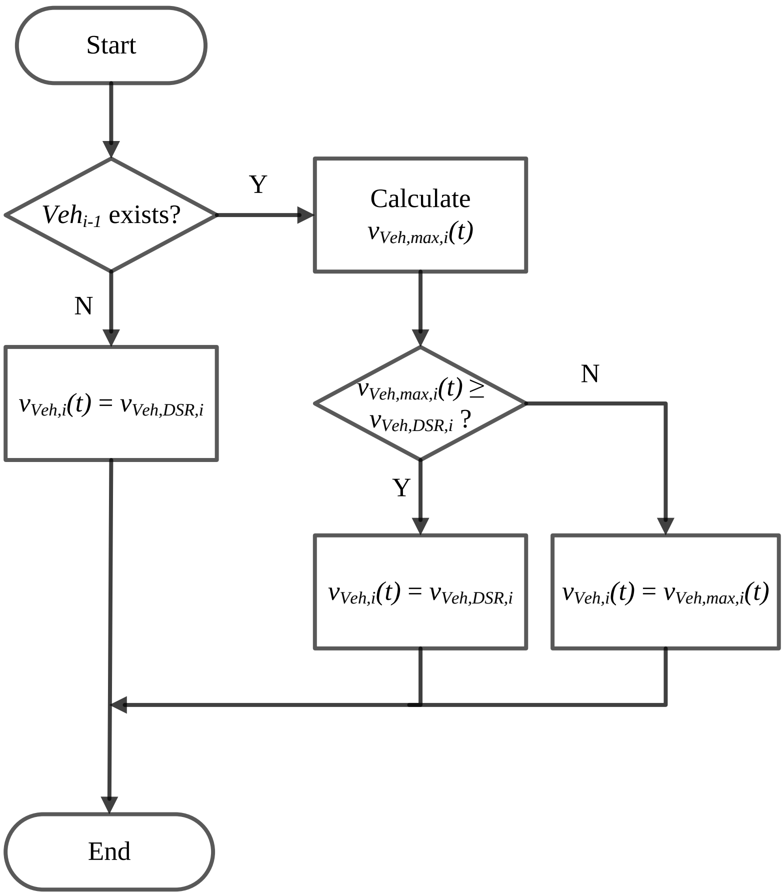


Figure 4.3 The flow chart of determining vehicle initial speed

The concept of calculating $v_{Veh,max,i}(t)$ is that, with this speed Veh_i will "marginally" collide with Veh_{i-1}, which is decelerating with its maximum deceleration $d_{max,i-1}$ at time t. The term "marginally" means that if Veh_i starts to decelerate with maximum deceleration $d_{max,i}$ at its next reaction time, assuming the decelerating Veh_{i-1} comes to a full stop at time $t+\Delta t$, Veh_i will also comes to a full stop at $t+\Delta t$ and just intrude the margin of Veh_{i-1}, as shown in Equation 4.10.

$$\Delta x_{Veh,i}(t+\Delta t) = \Delta x_{Veh,i-1}(t+\Delta t) - l_{V,i-1} - l_{M,i-1} \qquad (4.10)$$

Where,

$\Delta x_{Veh,i}(t+\Delta t)$ is the Δx of Veh_i at time $t+\Delta t$, (m);

$\Delta x_{Veh,i}(t+\Delta t)$ is the Δx of Veh_{i-1} at time $t+\Delta t$, (m);

$l_{V,i-1}$ is the vehicle length of Veh_{i-1}, (m);

$l_{M,i-1}$ is the margin length of Veh_{i-1}, (m).

And from vehicle kinetics, the following relationships can be derived, as shown in Equation 4.11 and Equation 4.12.

$$\Delta x_{Veh,i}(t+\Delta t) = \Delta x_{Veh,i}(t) + v_{Veh,max,i}(t) \cdot \Delta t_{Veh,TTR,i}(t) + \frac{v_{Veh,max,i}^2(t)}{2d_{max,i}} \quad (4.11)$$

Where,

$\Delta x_{Veh,i}(t+\Delta t)$ is the Δx of Veh_i at time $t+\Delta t$, (m);

$\Delta x_{Veh,i}(t)$ is the Δx of Veh_i at time t, (m);

$v_{Veh,max,i}(t)$ is the maximum allowed speed for Veh_i at time t, (m/s);

$\Delta t_{Veh,TTR,i}(t)$ is the time-to-react of Veh_i at time t, (s);

$d_{max,i}$ is the maximum deceleration rate of Veh_i, ($d_{max,i} > 0$), (m/s^2).

$$\Delta x_{Veh,i-1}(t+\Delta t) = \Delta x_{Veh,i-1}(t) + \frac{v_{Veh,i-1}^2(t)}{2d_{max,i-1}} \quad (4.12)$$

Where,

$\Delta x_{Veh,i-1}(t+\Delta t)$ is the Δx of Veh_{i-1} at time $t+\Delta t$, (m);

$\Delta x_{Veh,i-1}(t)$ is the Δx of Veh_{i-1} at time t, (m);

$v_{Veh,i-1}(t)$ is the speed for Veh_{i-1} at time t, (m/s);

$d_{max,i-1}$ is the maximum deceleration rate of Veh_{i-1}, ($d_{max,i-1} > 0$), (m/s^2).

Therefore, Equation 4.13 can be derived by putting Equation 4.11 and Equation 4.12 into Equation 4.10

$$\Delta x_{Veh,i}(t) + v_{Veh,max,i}(t) \cdot \Delta t_{Veh,TTR,i}(t) + \frac{v_{Veh,max,i}^2(t)}{2d_{max,i}} = \Delta x_{Veh,i-1}(t) + \frac{v_{Veh,i-1}^2(t)}{2d_{max,i-1}} - l_{V,i-1} - l_{M,i-1} \quad (4.13)$$

Where,

$\Delta x_{Veh,i}(t)$ is the Δx of Veh_i at time t, (m);

$v_{Veh,max,i}(t)$ is the maximum allowed speed for Veh_i at time t, (m/s);

$\Delta t_{Veh,TTR,i}(t)$ is the time to the next reaction time of Veh_i at time t, (s);

$d_{max,i}$ is the maximum deceleration rate of Veh_i, ($d_{max,i} > 0$), (m/s^2);

$\Delta x_{Veh,i-1}(t)$ is the Δx of Veh_{i-1} at time t, (m);

$v_{Veh,i-1}(t)$ is the speed for Veh_{i-1} at time t, (m/s);

$d_{max,i-1}$ is the maximum deceleration rate of Veh_{i-1}, ($d_{max,i-1} > 0$), (m/s^2);
$l_{V,i-1}$ is the vehicle length of Veh_{i-1}, (m);
$l_{M,i-1}$ is the margin length of Veh_{i-1}, (m).

From Equation 4.13, $v_{Veh,max,i}(t)$ can be solved as shown in Equation 4.14

$$v_{Veh,max,i}(t) = d_{max,i} \cdot \left\{ \sqrt{\Delta t_{Veh,TTR,i}^2(t) + \frac{2\left[\left(\Delta x_{Veh,i-1}(t) + \frac{v_{Veh,i-1}^2(t)}{2d_{max,i-1}} - l_{V,i-1} - l_{M,i-1}\right) - \Delta x_{Veh,i}(t)\right]}{d_{max,i}}} - \Delta t_{Veh,TTR,i}(t) \right\} \quad (4.14)$$

Where,
$v_{Veh,max,i}(t)$ is the maximum allowed speed for Veh_i at time t, s.t. Veh_i will just intrude the margin of Veh_{i-1} ($v_{Veh,max,i}(t) \geq 0$), (m/s);
$d_{max,i}$ is the maximum deceleration rate of Veh_i, ($d_{max,i} > 0$), (m/s^2);
$\Delta t_{Veh,TTR,i}(t)$ is the time to the next reaction time of Veh_i at time t, (s);
$\Delta x_{Veh,i-1}(t)$ is the Δx of Veh_{i-1} at time t, (m);
$v_{Veh,i-1}(t)$ is the speed for Veh_{i-1} at time t, (m/s);
$d_{max,i-1}$ is the maximum deceleration rate of Veh_{i-1}, ($d_{max,i-1} > 0$), (m/s^2);
$l_{V,i-1}$ is the vehicle length of Veh_{i-1}, (m);
$l_{M,i-1}$ is the margin length of Veh_{i-1}, (m);
$\Delta x_{Veh,i}(t)$ is the Δx of Veh_i at time t, (m).

To sum up, the vehicle initial speed can be decided by the algorithm described in Figure 4.2 combined with Equation 4.14. There are 2 parameters to be provided by the user, including the vehicle maximum deceleration rate, and vehicle reaction time. The vehicle reaction time is the same parameter required by the car-following model and therefore will be discussed in Section 4.5.2. As to the maximum deceleration rate, previous studies have seen the use of values listed in Table 4.7. This research adopts the value suggested in Du (2008)'s research involving urban traffic simulation in Beijing, China as it has the most similar scenario to this research.

Table 4.7 Vehicle maximum deceleration used in other studies

Researcher	Scenario	Suggested value (m/s^2)
Gipps (1981)	3-lane divided highway, Australia	N (3.4, (0.6)^2)
Wu (1994)	Urban junction, UK	2.5 to 5.94
Du (2008)	Urban street, China	4.2

The initial value of vehicle acceleration can be simply assigned with zero as it is just a temporary attribute and can be adjusted according to the car-following model as soon as the vehicle enters the simulation system (Wu 1994).

4.5 Vehicle car-following Model

The way in which "one vehicle in a stream of traffic reacts to the behaviour of its preceding vehicle" is defined as car-following behaviour (Gipps 1981). The car-following model is the mathematical model implemented in the micro-simulation program to mimic the vehicle following process in the real world. It plays a crucial part in the simulation program in this research. To select an appropriate model suitable for this research, a brief literature review of common car-following models is conducted in Section 4.5.1. After that, the implementation of the proposed model and the calibration of its parameters are discussed in Section 4.5.2.

4.5.1 A review of main car-following models

Car-following model is regarded as the "cornerstone for many important areas of traffic research such as micro-simulation modelling and advanced vehicle control, etc" (Brackstone and McDonald 1999). There have been vast amount of efforts to develop credible car-following models. Some of the commonly used car-following models are reviewed as follows and one of them is to be selected as the core of vehicle behaviour model for this research on the basis of this review.

4.5.1.1 Gazis-Herman-Rothery (GHR) model

Initially developed by General Motors research laboratory in Detroit, United States in 1958, the GHR model is one of the earliest models and regarded as the most well-known model to describe the car-following behaviour. The formulation of the model is shown in Equation 4.15 (Brackstone and McDonald 1999).

$$a_n(t) = cv_n^m(t) \cdot \frac{v_n(t-T) - v_{n-1}(t-T)}{[x_n(t-T) - x_{n-1}(t-T)]^l} \tag{4.15}$$

Where,

$a_n(t)$ is the acceleration of *Vehicle n* applies at time *t*, (m/s^2);

$v_n(t)$ is the speed of *Vehicle n* at time *t*, (m/s);

$v_n(t\text{-}T)$ and $v_{n-1}(t\text{-}T)$ are the speeds of V*ehicle n* and *n-1* respectively, at time *t-T*, (m/s);

$x_n(t\text{-}T)$ and $x_{n-1}(t\text{-}T)$ are the displacement in the lane to the same reference point of vehicle *n* and *n-1* respectively, at time *t-T*, (m);

Vehicle n-1 is the vehicle immediately ahead of *Vehicle n*;

c, *m* and *l* are the constants to be calibrated.

Researchers have shown significant interests in the GHR model for several decades and extensive work has been done on the calibration and validation of this model. Brackstone and McDonald (1999) summarised the most important results found, as shown in Table 4.8.

Table 4.8 Most reliable estimation of the parameters *m* and *l* for the GHR model

Source	*m*	*l*
Chandler et al (1958)	0	0
Herman and Potts (1959)	0	1
Hoefs (1972) (dcn no brk dcn brk / acn)	1.5/0.2/0.6	0.9/0.9/3.2
Treiterer and Myers (1974) (dcn / acn)	0.7/0.2	2.5/1.6
Ozaki (1993) (dcn / acn)	0.9/-0.2	1/0.2

dcn/acn: deceleration/acceleration, brk/no brk: deceleration with and without the use of brakes

(Source: Brackstone and McDonald 1999)

However, the GHR model is now being used less frequently, primarily for the following reasons. First, the parameters in this model have no obvious connection with the identifiable characteristics of vehicles, resulting that the model is less convenient to calibrate. In fact, a large number of contradictory findings have been presented in previous studies for correcting values of *m* and *l* in this model (Brackstone and McDonald 1999). Second, the model cannot differentiate between the car-following and free flow status. In other words, in this model, the follower is affected even if there is a long distance to the leader. Some researchers have tried using a deterministic space threshold as the separation between car-following and free flow status. However, this introduces new parameters, which are also less convenient to calibrate and many different values of thresholds have been suggested for use in practice (Aycin 2001, Toledo 2003).

4.5.1.2 Action point model

The underlying factors that lead to this type of model were firstly proposed by Michaels (1963), who suggested an angular velocity, expressed in Equation 4.16, as the action threshold. In this model, when the vehicle exceeds the thresholds defined by this angular velocity, it will take proper acceleration or deceleration actions. Otherwise, its acceleration is kept unchanged until another threshold is broken.

$$\frac{d\theta}{dt} = \frac{4W_l}{4\left[x_l(t) - x_f(t)\right]^2 + W_l^2} \cdot \left[\frac{dx_l(t)}{dt} - \frac{dx_f(t)}{dt}\right] \qquad (4.16)$$

Where,

$d\theta/dt$ is the angular velocity, (radius/s);

W_l is the width of the leading vehicle, (m);

$x_l(t)$ and $x_f(t)$ are the displacement in the lane to the same reference point of the leading and following vehicle respectively, at time *t*, (m).

Leutzbach and Wiedemann (1986) firstly incorporated the concept of perception threshold into their micro-simulation car following model called MISSION. This simulation model divides the car-following process into several sub-situations; for example, free driving, closing in, following and emergency situation. The vehicle agent is programmed in a way that it will apply different accelerations in different sub-situations once the according threshold is exceeded. For example, if a vehicle is approaching its leader, the action this vehicle takes when the threshold is broken can be expressed with Equation 4.17.

$$d_f(t+T) = -\frac{\left[\frac{dx_l(t)}{dt} - \frac{dx_f(t)}{dt}\right]^2}{2\left[x_l(t) - x_f(t) - D\right]} + d_l(t) \tag{4.17}$$

Where,

$d_f(t+T)$ is the deceleration rate (≥ 0) of the following vehicle at time $t+T$, (m/s^2);

T is the reaction time of the following vehicle, (s);

$x_l(t)$ and $x_f(t)$ are the displacement in the lane to the same reference point of the leading and following vehicle respectively, at time t, (m).

D is the effective size of the leading vehicle, which is the length plus the margin of the leading vehicle, (m);

$d_l(t)$ is the deceleration rate (≥ 0) of the leading vehicle at time t, (m/s^2);

Models of this type are considered more rational as it takes into account the human threshold of perception. However, it is much more complicated to calibrate the threshold values and the modification mechanism of the acceleration above each value, as well as how to switch between each pre-defined sub-situation. Some researchers (Brackstone and McDonald 1999, Zheng 2003) believed that “the calibration of such models had been less successful and it was difficult to come to a firm conclusion as to the validity of these models”.

4.5.1.3 Collision avoidance model

The core concept of this type of model is that for the following vehicle there exists a safe distance within which a collision cannot be avoided if the leading vehicle in front acts unpredictably. Therefore, the follower always tries to keep itself out of the safe distance when it is the time for it to make an action. Usually, the safe following distance can be obtained by applying basic Newtonian equations of motion.

The major development of this type of model was made by Gipps (1981), in which he set several behavioural limits to vehicles to calculate a safe speed as a decision output of the following vehicle, which would only selected a speed that could ensure a safe stop if the leading vehicle came to a sudden stop. By combining several constraints together, the model was achieved in the form of the following equation, as shown in Equation 4.18 (Gipps 1981).

$$v_n(t+\tau) = min\left\{v_n(t) + 2.5a_n\tau\left(1-\frac{v_n(t)}{V_n}\right)\sqrt{0.025+\frac{v_n(t)}{V_n}},\quad b_n\tau + \sqrt{b_n^2\tau^2 - b_n\left\{2[x_{n-1}(t) - s_{n-1} - x_n(t)] - v_n(t)\tau - \frac{v_{n-1}^2(t)}{\hat{b}}\right\}}\right\} \quad (4.18)$$

Where,

Vehicle n-1 is the leading vehicle to *Vehicle n*;

$v_n(t+\tau)$ is the speed of the *Vehicle n* at time $t+\tau$, (m/s);

$v_n(t)$ is the speed of *Vehicle n* at time t, (m/s);

a_n is the maximum acceleration which *Vehicle n* wishes to undertake ($a_n > 0$), (m/s^2);

V_n is the desired speed of *Vehicle n*, (m/s);

b_n is the maximum deceleration which *Vehicle n* wishes to undertake ($b_n < 0$), (m/s^2);

b_{n-1} is the maximum deceleration which *Vehicle n* wishes to undertake ($b_{n-1} < 0$), (m/s^2);

$x_{n-1}(t)$ is the location of *Vehicle n-1* at time t, (m);

$x_n(t)$ is the location of *Vehicle n* at time t, (m);

s_{n-1} is the effective size of *Vehicle n-1*, which is the physical length plus a margin into which the following vehicle is not willing to intrude, even when at rest, (m);

$\hat{b}$ is an estimation of b_{n-1}, Gipps (1981) suggested using $min\{-3.0, (b_n - 3.0)/2\}$, (m/s^2)

This type of model has been widely used in micro-simulation due to the reason that it correlates model parameters to vehicle characteristics and therefore it can be easily calibrated with common assumptions about vehicle behaviour, requiring only a few parameters to be confirmed. However, this model also has some drawbacks. For example, with the combined restraints it can only give a maximum allowed speed for the following vehicle, assuming that the vehicle wishes to travel as fast as it can. However, in reality this assumption cannot always stands as some vehicles may consider some other factors on the road such as several vehicles downstream to decide its action thus may not always wish to take the maximum allowed speed but a certain smaller value, which cannot be predicted by this model. In addition, the idea of using speed instead of acceleration as the decision output is not realistic to the situation in the real world.

4.5.1.4 Fuzzy logic model

The use of fuzzy logic is a more recent approach in modelling the car-following behaviour. This model has the following general form, as shown in Figure 4.4.

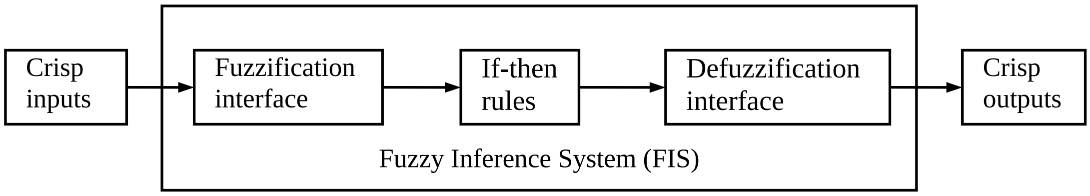


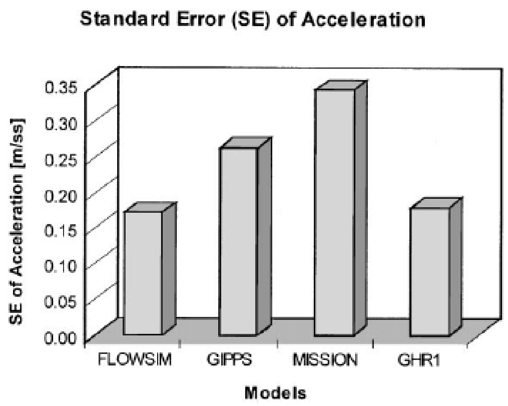
Figure 4.4 The general form of fuzzy logic car-following model

This model uses several driver perception variables as inputs, such as relative speed and distance, etc, and predicts control variable for the driver, usually in the form of acceleration. The prediction is achieved by using a “black box” called Fuzzy Inference System (FIS), which consists of 3 main modules including the fuzzification interface, the if-then rules and the defuzzification interface. The working process of this model can be described as follows. First, the crisp inputs describing the driver’s perceptions are derived from the vehicle and the concurrent traffic conditions. Then the crisp inputs are fuzzified into several fuzzy sets with a fuzzification interface defined by a series of membership functions. Next, a fuzzy reasoning process is performed by executing several if-then rules, usually defined by a series of linguistic expressions, such as “IF *x is A* AND *y is B* THEN *z is C*”. The output of each rule is a fuzzy set, which describes a certain type of driver’s decision. These output fuzzy sets are then aggregated into a single output fuzzy set. Finally, the defuzzification process is performed to resolve the ultimate single output fuzzy set to a crisp value that can be utilised by the simulation program. The FIS plays a key role in the fuzzy logic based model and should be calibrated with vast amount of local driving data. Wu (2000) et al developed a fuzzy logic based model called “FLOWSIM”, using relative speed and distance divergence as inputs and acceleration as output. The validation results of FLOWSIM have shown that this model performs better than GHR, Gipps and MISSION model (Wu 2000). Also, this model has been calibrated and validated using Chinese data by Du (2008) when she studied the driver-cyclist interaction behaviour in Beijing, China.

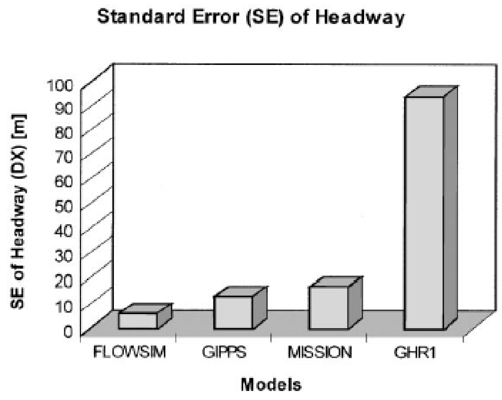
4.5.1.5 Selection of the car-following modelling suite

It can be seen from the above discussion that the modelling of car-following behaviour has been extensive. The underlying assumptions and forms of these models vary in many different ways and each of them has shown the ability to describe this behaviour to some extent.

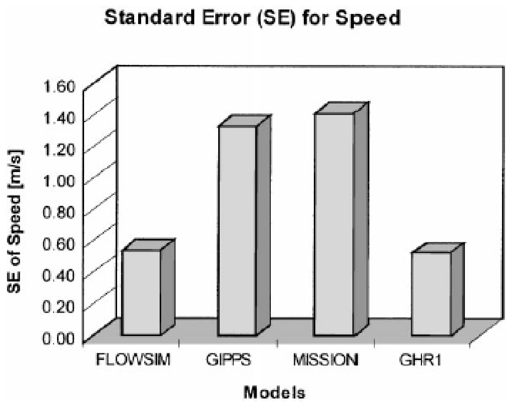
Wu et al (2000) conducted a study to compare the microscopic performance of four commonly used car-following models, including FLOWSIM, Gipps, MISSION and GHR, which were based on fuzzy logic, collision avoidance, action point and Gazis-Herman-Rothery modelling techniques respectively. The surveyed data from the subject vehicles and test traces on the UK roads were used for the comparison, which used Standard Error (SE) to measure the degree of fit between the simulated and actual data. Analysis examples for one test driver were shown in Figure 4.5.



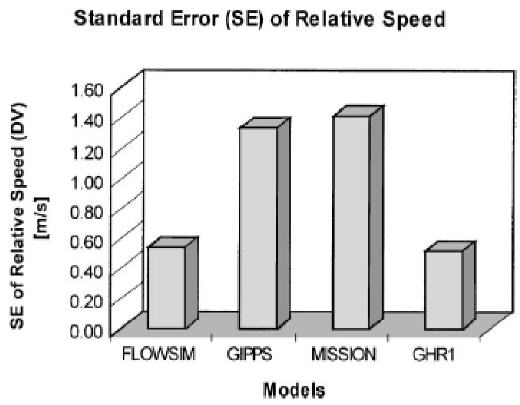
(a) Comparison of SE on acceleration rates for different car-following models



(b) Comparison of SE on headways for different car-following models



(c) Comparison of SE on speeds for different car-following models



(d) Comparison of SE on relative speeds for different car-following models

Figure 4.5 Comparison of different car-following models
(Source: Wu et al 2000)

It can be seen that FLOWSIM gave the best overall performance for all outputs compared to the other three. This is probably due to this model was realised based on such a knowledge base (called the “fuzzy inference system” in this model), which was established from extensive data collected from both road-side cameras and instrumented vehicles, a most recent advanced data collection technique, that ensured the mapping between the outputs and the inputs data was more accurate than other common models only using one or several equations to describe the behaviour. Gipps and MISSION models come second and third respectively in terms of overall performance. GHR model gives good results on outputs of speed, acceleration rate and relative speed, but much greater errors of headways.

In addition to its better accuracy, the fuzzy logic model are also behaviourally meaningful because of the simple linguistic form, and is mathematically rigid because of several analytical tools developed to explore properties of fuzzy logic model analytically (Zheng 2003). Besides, the source code of FLOWSIM, the fuzzy logic model developed at the University of Southampton, is fully available to the author. This is important as the incorporation of pedestrian objects will inevitably evolve the modification of the intra vehicle models. The access to the source code will facilitate the author to design the relative algorithms for the new simulation model more efficiently and flexibly.

In conclusion, the FLOWSIM model is employed to be the fundamental model for intra-vehicle behaviour because of its better overall performance and the flexibility it can provide for the author to modify the internal logic to incorporate pedestrians in the future.

4.5.2 The proposed car-following model

The FLOWSIM car-following model uses two variables as model inputs, which are relative speed (*DV*) and distance divergence (*DSSD*). The model output is vehicle's acceleration (*ACC*). In Figure 4.6, assuming Veh_{i-1} is the leading vehicle of Veh_i, *DV* is then defined with Equation 4.19 and *DSSD* is defined with Equation 4.20.

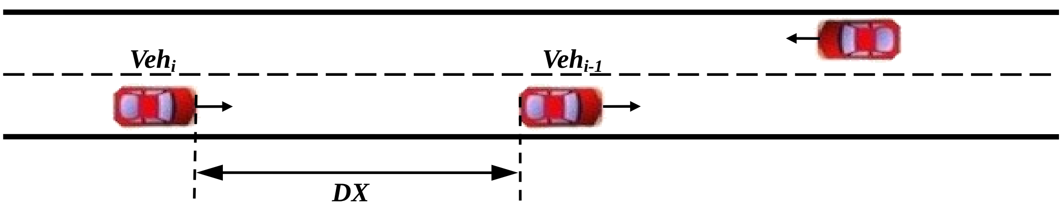


Figure 4.6 FLOWSIM car-following model

$$DV(t) = v_{i-1}(t) - v_i(t) \quad (4.19)$$

Where,

DV(t) is the relative speed between the leading vehicle *i-1* and the subject vehicle *i*, at time *t*, (m/s);

$v_{i-1}(t)$ is the instantaneous speed of the leading vehicle *i-1*, at time *t*, (m/s);
$v_i(t)$ is the instantaneous speed of the subject vehicle *i*, at time *t*, (m/s);

$$DSSD(t) = \frac{DX(t)}{D_{DSR,i}} = \frac{DX(t)}{v_i(t) \cdot t_{DSR,i}} \tag{4.20}$$

Where,
DSSD(t) is distance divergence between the leading vehicle *i-1* and the subject vehicle *i*, at time *t*;
DX(t) is the relative distance between the leading vehicle *i-1* and the subject vehicle *i*, at time *t*, (m);
$D_{DSR,i}$ is the desired headway of vehicle *i*, (m);
$v_i(t)$ is the instantaneous speed of the subject vehicle *i*, at time *t*, (m/s);
$t_{DSR,i}$ is the desired time headway of vehicle *i*, at time *t*, (s).

Therefore, at the reaction time of any subject vehicle, given a combination of *DV* and *DSSD*, the FIS in the FLOWSIM can generate an output in terms of vehicle acceleration. The typical fuzzy sets used in FLOWSIM are listed in Table 4.9. A typical fuzzy rule for the car-following behaviour has the following form: "IF *DSSD is too great* AND *DV is closing* THEN *ACC is no action*" (Wu 2000).

Table 4.9 Fuzzy sets used in FLOWSIM car-following model

Relative speed (*DV*)	Distance divergence (*DSSD*)	Acceleration (*ACC*)
Opening fast (*V1*)	Much too far (*S1*)	Strong acceleration
Opening (*V2*)	Too far (*S2*)	Light acceleration
About zero (*V3*)	Satisfied (*S3*)	No action
Closing (*V4*)	Too close (*S4*)	Light deceleration
Closing fast (*V5*)	Much too close (*S5*)	Strong deceleration

(Source: Wu 2000)

There are three parts in the FLOWSIM model requiring calibration according to the local situation when conducting simulation study, including the FIS, vehicle reaction time and vehicle desired time headway. This research employs the FIS calibrated by Du (2008) using Chinese data. A FIS reading class is implemented by the author to utilise this model in the simulation program in this research. The definition of reaction time follows the suggestion by Liu et al (2002), who studied vehicle's reaction time in 13 provinces in China and proposed the use of an average value of 0.89 seconds in Beijing. As to the desired time headway, Zheng (2003) suggested that the mean value of desired time headway among different drivers could be described with a lognormal distribution, while the value within a driver could also be described with a lognormal distribution, with the standard deviation and mean yielding to a linear relationship. This idea was backed up by Zhou and Chen (2004), who suggested the mean and standard deviation among different drivers be 2.0 and 0.4 seconds, respectively, as well as the linear relationship of the mean and standard deviation within one driver be "standard deviation = 0.41mean + 0.08".

4.6 Validation of the vehicle behaviour model

This section discusses the validation of the vehicle behaviour model developed in this research. The saturation flow and average vehicle journey time are examined in the validation process.

4.6.1 Validation of saturation flow

The saturation flow is defined as "the maximum flow, expressed in equivalent passenger car units (pcu), which can be discharged from a traffic lane when there is a continuous green indication and a continuous queue on the approach" (Salter 1985). It is an integrated index of several vehicle behaviour factors such as desired speed, desired time headway and reaction time, etc. Therefore, it is generally considered that the modelled vehicle behaviour in a simulation system is similar to the local community to be studied if the system can produce similar saturation flow to the one in the real world.

4.6.1.1 Data collection

The most reliable method of validating a simulation model is comparing the simulation outputs with the data from field survey. Therefore, a site is chosen for surveying the actual saturation flow from field survey. Then the site is modelled into the simulation and the saturation flow is surveyed again from the simulation system. These two sets of saturation flow data are then compared to check if the simulation system is sufficiently close to the real system.

<u>1. Saturation flow from field survey</u>

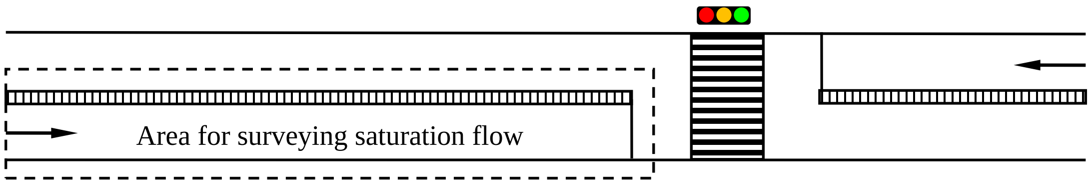


Figure 4.7 Data collection site for validation of saturation flow

As shown in Figure 4.7, the site chosen for the field survey of saturation flow is a 2-way-2-lane road section in Beijing, China, with a guard rail in the median to ensure there is no interference of pedestrian jaywalking activities. A fixed-time signal pedestrian crossing is in the middle of the road section. Four synchronised video cameras are deployed covering the area from one of the two stopping lines to the corresponding upstream traffic, in order to record the whole vehicle queuing and discharging behaviour in that lane. The survey was carried out during peak times in several different days when the traffic was likely to be saturated. The raw video data was then taken into the laboratory for analysis.

The saturation flow was estimated from the video data using the method described in Highway Capacity Manual (TRB 2000), which suggested using the stop line as the survey reference point and that the period of saturation flow began when the front axle of the 4^{th} vehicle in the queue crossed the stop line and ended when the front axle of the last vehicle in the queue crossed the stop line during the green time. The number of total vehicles and the type of each vehicle during this time were recorded and then transformed into the equivalent pcu with the values suggested by Ministry

Of Housing and Urban-Rural Development of China (MOHURD), as shown in Table 4.10. Then the saturation flow was calculated using Equation 4.21.

$$F_S = \frac{n-1}{t_i - t_4} \times 3600, \quad i > 11 \; and \; i \in \mathbb{N} \tag{4.21}$$

Where,

F_S is the saturation flow of this lane, (pcu/h);

t_4 is the time when the front axle of the 4^{th} vehicle in the queue crosses the stop line during the green time, (s);

t_i is the time when the front axle of the i^{th} and also the last vehicle in the queue crosses the stop line during the green time, (s);

n is the equivalent pcu crossing the stop line during the green time, (pcu)

In order to obtain a statistically significant value, a minimum of 15 signal cycles with more than 11 vehicles in the queue was required (TRB 2000). The saturation flow data from field survey are shown in Table 4.11.

Table 4.10 Values of equivalent pcu in China

Type		Description	Examples	pcu
Light Vehicles (LV)		3-4 wheeled vehicles	Passenger cars	1.0
Medium Commercial Vehicles (MCV)		2 axles and more than 4 wheels	Vans, pick-ups	1.6
Heavy Commercial Vehicles (HCV)		More than 2 axles	Heavy good vehicles	2.2
Buses and coaches	Regular (BCR)	Buses and coaches	Buses and coaches	2.0
	Articulated (BCA)			2.8

(Source: MOHURD 1995)

Table 4.11 Saturation flow from field survey

Number of observation	Start time (s)	End time (s)	Time interval (s)	Number of pcu	Saturation flow (pcu/h)
1	111.72	150.24	38.52	15.0	1439.23
2	189.04	214.84	25.80	10.0	1343.28
3	665.64	693.24	27.60	11.0	1434.78
4	745.96	766.68	20.72	8.0	1389.96
5	901.52	929.68	28.16	11.0	1358.02
6	1060.76	1086.24	25.48	10.0	1470.59
7	1299.56	1327.48	27.92	11.0	1369.29
8	1380.92	1404.24	23.32	9.0	1332.24
9	1774.60	1799.36	24.76	9.6	1454.55
10	2008.76	2031.68	22.92	9.0	1478.10
11	2168.60	2192.24	23.64	9.2	1401.02
12	2245.96	2271.40	25.44	10.0	1415.09
13	2322.88	2346.04	23.16	9.0	1462.09
14	2482.72	2505.84	23.12	9.0	1343.28
15	3508.68	3537.00	28.32	11.0	1398.31
Mean (pcu/h)					1405.99
Standard deviation (pcu/h)					49.41

2. Saturation flow from simulation

The site was then modelled into the simulation program and the same procedure was applied to calculate the saturation flow by running the simulation. The procedure was repeated for 30 times, twice as many as in the field survey as the simulation system was much more convenient for observation. The saturation flow data from simulation are shown in Table 4.12.

Table 4.12 Saturation flow from micro-simulation

Number of observation	Start time (s)	End time (s)	Time interval (s)	Number of pcu	Saturation flow (pcu/h)
1	173.9	204.8	30.9	12.0	1411.76
2	254.8	288.6	33.8	13.0	1397.01
3	338.4	369.5	31.1	12.0	1440.00
4	417.2	445.6	28.4	11.0	1394.37
5	497.4	528.4	31.0	12.0	1407.17
6	578.9	612.3	33.4	13.0	1401.20
7	659.8	695.7	35.9	14.0	1419.72
8	740.6	774.2	33.6	13.0	1435.58
9	823.7	854.5	30.8	12.0	1402.60
10	901.8	935.4	33.6	13.0	1397.01
11	982.7	1013.8	31.1	12.0	1421.05
12	1064.3	1095.2	30.9	12.0	1398.06
13	1145.2	1178.4	33.2	13.0	1409.64
14	1226.0	1257.1	31.1	12.0	1389.07
15	1306.5	1340.1	33.6	13.0	1392.86
16	1387.4	1418.4	31.0	12.0	1393.55
17	1468.3	1502.5	34.2	13.0	1409.64
18	1548.7	1579.5	30.8	12.0	1402.60
19	1630.2	1660.7	30.5	12.0	1416.39
20	1711.1	1746.7	35.6	14.0	1415.73
21	1792.2	1823.1	30.9	12.0	1398.06
22	1950.7	1981.8	31.1	12.0	1416.39
23	2031.9	2065.2	33.3	13.0	1405.41
24	2112.4	2145.6	33.2	13.0	1409.64
25	2193.3	2224.3	31.0	12.0	1440.00
26	2274.1	2305.3	31.2	12.0	1430.46
27	2355.0	2387.2	32.2	12.6	1408.70
28	2436.1	2466.9	30.8	12.0	1402.60
29	2516.3	2547.5	31.2	12.0	1430.46
30	2597.5	2630.5	33.0	13.0	1418.18
Mean (pcu/h)					1410.50
Standard deviation (pcu/h)					14.18

4.6.2.2 Analysis of results

It can be seen from Table 4.11 and Table 4.12 that the saturation flow estimated from field survey and simulation appear to be very close. Statistical tests showed that there was not enough evidence to reject that the saturation flow data from field survey and from simulation yielded to independent normal distributions (see Appendix I). Therefore, the following 2-sample t test assuming different variances was carried out to check the similarity of the means from the two samples.

Test method: 2-sample t test.

Test hypotheses: H_0: the mean values of observed saturation flow rate from field survey and simulation are equal; H_1: the mean values of observed saturation flow rate from field survey and simulation are not equal.

The result of the test is shown in Table 4.13, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.734). In other words, it is reasonable to accept H_0, implying that the saturation flow from the micro-simulation is valid and the vehicle behaviour model in the micro-simulation is capable of generating realistic traffic flows.

Table 4.13 The result of 2-sample t test for saturation flow

Equal variances not assumed

	t-test for Equality of Means						
						95% Confidence Interval of the Difference	
	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
SaturationFlow	-.346	15.163	.734	-4.50967	13.01870	-32.23235	23.21302

4.6.2 Validation of vehicle journey time

The vehicle journey time between two data collection points is another commonly used indicator for the validation of vehicle behaviour models (Park and Schneeberger 2003). Given the same system inputs, the distributions of vehicle journey times generated by a credible simulation model should be similar to the actual system. For this purpose, a site was chosen for surveying the actual times of individual vehicles travelling through a road section. Then the site was modelled into the simulation and individual vehicle journey times were obtained by running the simulation. These two sets of journey time data were then compared to check if the simulation system could generate satisfactory outputs. Park and Schneeberger (2003) suggested using the t-test to compare the means of the two data sets, or more rigorously using the Kolmogorov-Smirnov (K-S) test to compare the distributions of the two data sets. If there is not enough evidence to reject that the distributions of the two data sets are the same, the model can be regarded accurate in a statistical meaning. Otherwise, the model developer should examine and recalibrate the behavioural parameters inside the simulation model and repeat the validation process until the model is accurate.

4.6.2.1 Data collection

The layout of the site chosen for the field survey of vehicle journey time is illustrated in Figure 4.8. It is a 2-way-2-lane road section in Beijing, China, with a length of 300 m and guard rail in the median to ensure there is no interference of pedestrian jaywalking activities. A fixed-time signal pedestrian crossing is in the middle of the road section. Main parameters of this site are surveyed as shown in Table 4.14 and Figure 4.9, for the subsequent simulation modelling.

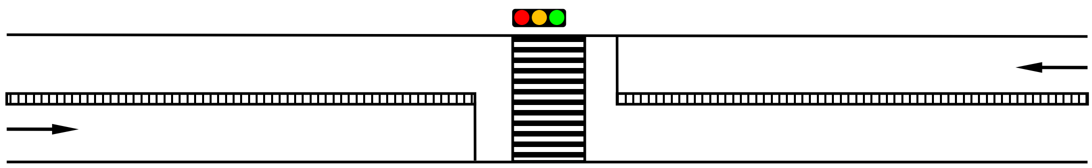


Figure 4.8 Data collection site for validation of vehicle journey time

Table 4.14 Parameters of the site for validation of vehicle journey time

Parameters	Description
Vehicle composition	75% LV, 20% MCV, 1% HCV, 2% BCR and 2% BCA
Vehicle desired speed mean	9.15 m/s
Signal timing configuration	Fixed-time signal as shown in Figure 4.9

Period	P1 (50 s)	P2 (3 s)	P3 (2 s)	P4 (20 s)	P5 (5 s)
Veh. signal	G	A	R	R	R
Ped. signal	R	R	R	G	R

Figure 4.9 Signal timing at the site chosen for validation of vehicle journey time

The field observation was carried out between 16:30 and 17:30 on a typical weekday. The traffic in both lanes of the road section was recorded using the video camera method discussed in Chapter 3, ensuring any vehicle travels between the two ends of this road section was traceable. The journey time of any passing vehicle was analysed with video processing software by subtracting the time when the vehicle entered this area from the time when the vehicle left. The traffic demand for each lane during the period of field survey was also recorded (eastbound = 610 veh/h and westbound = 588 veh/h during the data collection time on that day). The distribution of individual vehicle journey times from field survey is illustrated in Figure 4.10.

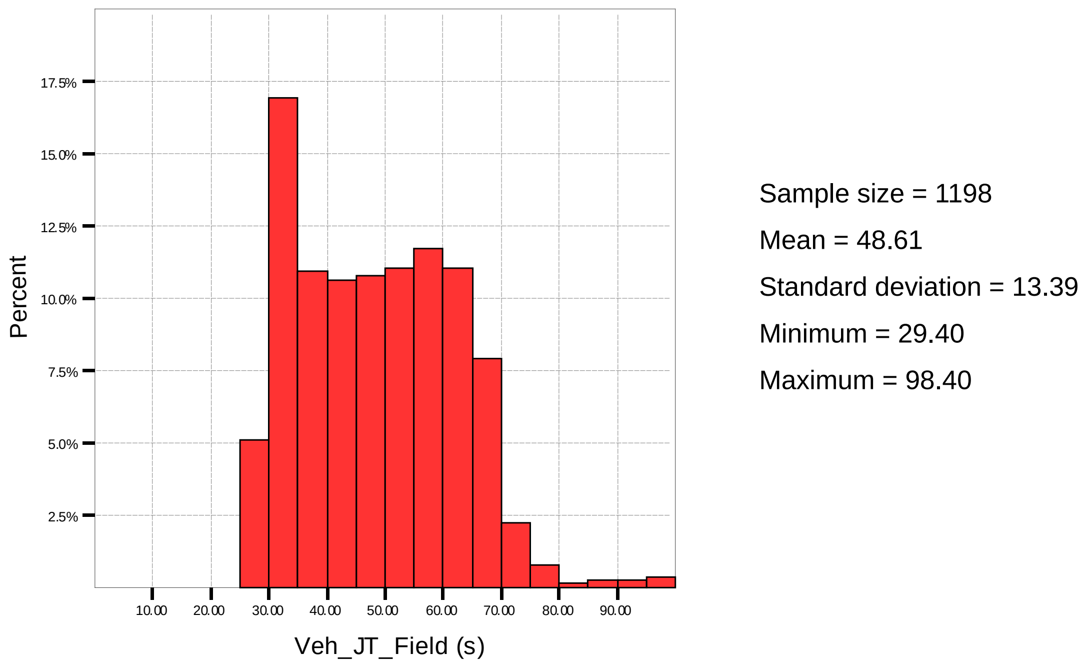


Figure 4.10 The distribution of vehicle journey times from field survey

Then, the scenario was modelled in the simulation with the surveyed traffic demand to obtain individual vehicle journey times from simulation. The simulation time was set longer than the actual surveying time in order to achieve a larger sample size. The distribution of individual vehicle journey times from simulation is illustrated in Figure 4.11.

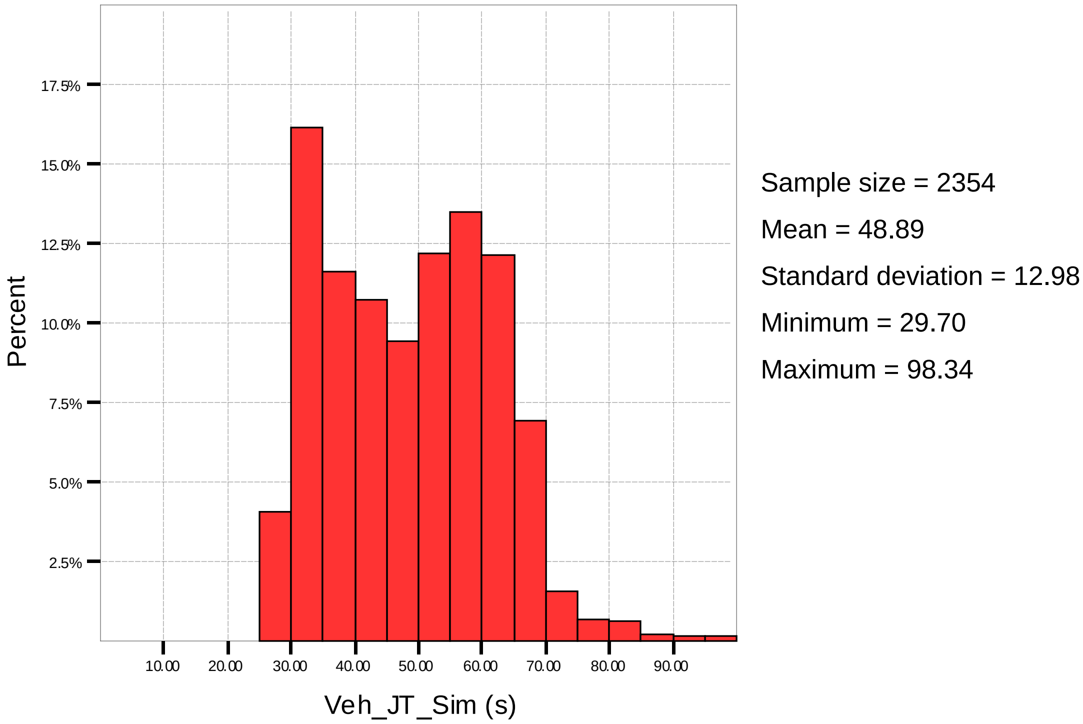


Figure 4.11 The distribution of vehicle journey times from simulation

4.6.2.2 Analysis of results

It can be seen from Figure 4.10 and Figure 4.11 that the distributions of actual and simulated vehicle journey times appear to be close. The following 2-sample K-S test was conducted to check the similarity of the distributions from the two samples.

Test method: 2-sample K-S test.

Test hypotheses: H_0: the distributions of the vehicle journey times from field survey and simulation are the same; H_1: the distributions of the vehicle journey times from field survey and simulation are not the same;

The result of the test is shown in Table 4.15, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.324). Therefore, it is reasonable to accept H_0. The distributions of vehicle journey times from field survey and simulation have no difference in a statistical meaning. In conclusion, the vehicle model can generate sufficiently accurate vehicle journey times.

Table 4.15 The result of 2-sample K-S test for vehicle journey times

Frequencies

	Type	N
Veh_JT_Field	Field	1198
	Sim	2354
	Total	3552

Veh_JT = Vehicle Journey Time, N = Sample Size

Test Statistics[a]

		Veh_JT_Field
Most Extreme Differences	Absolute	.034
	Positive	.017
	Negative	-.034
Kolmogorov-Smirnov Z		.953
Asymp. Sig. (2-tailed)		.324

a. Grouping Variable: Type

4.7 Conclusion

This chapter described the development, calibration and validation of the intra-vehicle model used in this research. First, the vehicles were divided into 4 categories and the static characteristics for each category were calibrated using on-road data collected in Beijing, China. Then, the vehicle free flow model generation model in the micro-simulation were introduced and defined. Next, the core car-following model was developed on the basis of a review of existing common models such as GHR, action point, collision avoidance and fuzzy logic models. The concepts of the fuzzy logic model in FLOWSIM, a micro-simulation tool developed at Transportation Research Group, University of Southampton were employed to develop the car-following model in this research. Series of parameters were calibrated according to the local situation in Beijing, China. Finally, the intra-vehicle model was validated in terms of saturation flow and journey times of individual vehicles passing through a typical road section in Beijing, China against field data independent of those used for model development. Statistical tests showed that the developed and calibrated intra-vehicle model was sufficiently reliable to be used in this research.

CHAPTER 5
PEDESTRIAN BEHAVIOUR AND MODELS

5.1 Introduction

Similar to the intra vehicle model, there is also a need to describe pedestrians' characteristics, their generation and basic movement behaviour in a micro-simulation environment. The intra pedestrian models are mainly used to describe pedestrian behaviour on pavements and signalised crossings (when the pedestrian signal is green) where there is no interaction with vehicles. This chapter discusses the development, calibration and validation of relative intra pedestrian behaviour models.

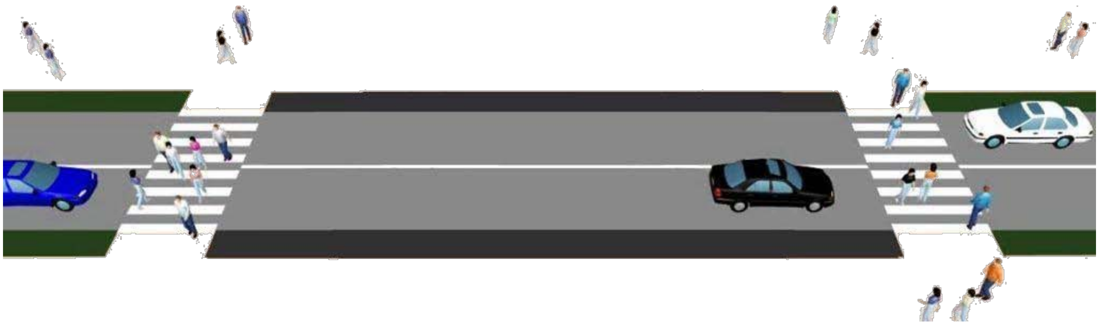
5.2 Pedestrian static characteristics

The Highway Capacity Manual (TRB 2000) recommends modelling the trace of a pedestrian as an ellipse of 0.5 m by 0.6 m. Several variances are found based on this recommendation. This research adopts Dell'Orco (2007)'s suggestion, which abstracts the pedestrian object as a circle with a diameter of 0.54 m in micro-simulation, having the same area as the ellipse.

In addition, pedestrians with different gender and age can have different behaviour. As to the age, in simulation modelling researchers usually divide pedestrians into several age groups, assuming that the behaviour from the same age group is more homogeneous with less variance (Sun et al 2003). In this research, the method proposed by Montufar et al (2007) is adopted. The age group of pedestrians is divided into "younger" and "older". The younger are those who appear to be between 20 and 64 years old, and the older are those who appear to be 65 years of age or older. Therefore, the type of pedestrians can be divided into 4 categories including Younger Male (YM), Younger Female (YF), Older Male (OM) and Older Female (OF).

5.3 Pedestrian generation model

Ideally, if pedestrian objects are to be incorporated into micro-simulation, their *O-D* matrix in a relatively large area should be surveyed and defined before the running of the simulation. However in reality, the *O-D* matrix of pedestrian activities is difficult to obtain than that of vehicles. Existing simulation tools usually adopt an oversimplified method, generating pedestrian objects from one end of a pre-defined crossing route, for example a crossing facility, to the other end, as shown in Figure 5.1. This is unrealistic as not all pedestrians in reality take the pre-defined route to cross the road. It cannot model the phenomenon of pedestrian jaywalking outside crossing facilities, where they can choose anywhere to cross the road. However, the jaywalking phenomenon is common in some developing countries like China, where there is a lack of traffic discipline on the road.



This is a snapshot taken from the simulation play in VISSIM, a commonly used micro-simulation tool for traffic analysis, showing that pedestrian crossing routes are pre-defined at two Zebra crossings only; the crossing behaviour between them is ignored.

Figure 5.1 An example of pre-defined pedestrian crossing routes adopted in VISSIM

Therefore, a trade-off between the full pedestrian activity *O-D* matrix and the oversimplified model has been applied. The pedestrian generation model proposed in this research is discussed as follows.

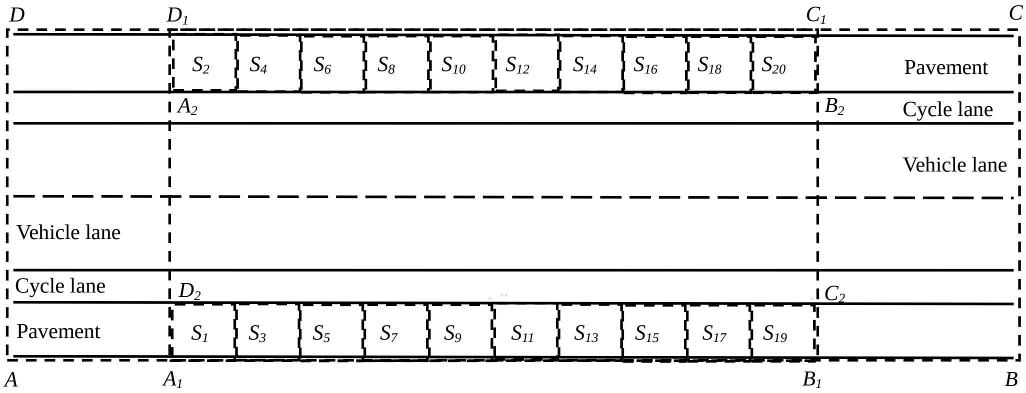


Figure 5.2 The proposed pedestrian generation model

As shown in Figure 5.2, *Rectangle ABCD* stands for the boundary of the abstracted simulation area and *Rectangle* $A_1B_1C_1D_1$ stands for the boundary of the area where the interaction between pedestrians and vehicles is to be studied. The overlapped area of *Rectangle* $A_1B_1C_1D_1$ and the pavement on either side of the road is divided into several smaller areas, each of which is marked as a pedestrian crossing *O/D* area S_i, ($i=2k-1$ for southern side and $i=2k$ for northern side, $k\in\mathbb{N}$). It is evident that any pedestrian crossing demand in area $A_1B_1C_1D_1$ can be uniquely marked with an *O-D* pair such as S_i-S_j ($i, j\in\mathbb{N}$ and $i+j=2k+1$, $k\in\mathbb{N}$). Therefore, the demand of pedestrian crossing activities in this area can be described with a crossing *O-D* matrix $M = [f_{i,j}]_{m\times m}$, ($i, j ,m \in\mathbb{N}$ and $i, j\leq m$), where $f_{i,j}$ indicates the crossing demand (ped/h) from S_i to S_j. The value of $f_{i,j}$ is related to the specific local situation and needs to be provided by the user when carrying out simulation study. Meanwhile, the distribution of the pedestrian appearing time interval for S_i-S_j is also to be provided from user defined distribution calibrated from field survey. Thus, with the crossing demand and the distribution of appearing time interval for each *O-D* pair, the global generation time can be determined for each pedestrian object in the micro-simulation. Apparently, the more *O/D* areas divided, the more explicitly the *O-D* matrix can describe the crossing demand but the more difficult for the user to calibrate the demand and the time interval distribution. For a trade-off, 10 *O/D* areas on either side of the road section are used in this research, as shown in Figure 5.2.

5.4 Pedestrian movement model

After the pedestrian is generated by the micro-simulation, his/her behaviour is then governed by a movement model. Similar to vehicle models, the intra pedestrian model is the basis to describe pedestrian behaviour on pavements. In this chapter, emphasis is on intra pedestrian behaviour when motorised traffic is not present. The influence of vehicle traffic on pedestrian movement behaviour and vice versa, as well as the integration of the two modes will be discussed in Chapter 6.

5.4.1 A review of main pedestrian movement models

This section provides a brief review of a few common modelling techniques for describing pedestrian's microscopic movement behaviour. The classification of these techniques is less rigid and the individual pedestrian behaviour can be reasonably modelled by a combination of several techniques, as long as the concerned model indicators are validated. A conclusion is made on how to model the pedestrian movement at the end of this section.

1. Social force model

The social force model is a type of model using analogy of concepts in physics to describe microscopic dynamics of pedestrians. It models pedestrians as objects with different diameters and velocities. The motion of pedestrians can be described as if "they would be subject to social forces" which are "a measure for the internal motivations of the individual to perform certain actions" (Helbing and Molnár 1995). In this model, there are three types of forces influencing the decision of any pedestrian, including the attractive force from the destination, the attractive force from other factors and the repulsive force from surrounding objects that conflict with the pedestrian. Each of the influencing force is aggregated and the pedestrian's decision can be then determined with Newtonian equations of mechanics. This model is a continuous-time-continuous-space simulation model suitable for scenarios containing pure pedestrian flow, such as buildings, train stations, and pedestrian zones. However, the form of this model is much more complex, containing many parameters to be calibrated and it is not convenient to integrate such a model with the proposed vehicle behaviour models discussed in the previous chapter.

2. Cellular automata model

The cellular automata model is a discrete-time-discrete-space simulation model consists of a grid of cells representing pedestrians. Each of the cells has a certain number of states. As the simulation time moves step by step, the state of each cell changes according to some transition rules involving neighbouring cells and previous state of the subject cell. There are a few pedestrian movement model based on cellular automata (Burstedde et al 2001, Li et al 2005). However, few have discussed the integration with vehicle models. The cellular automata is capable of simulating the pedestrian microscopic behaviour to some extent. However, the abstraction of pedestrians into cells lacks behavioural details and it is not flexible for such a model to be integrated with vehicle micro-simulation models without major modifications.

3. Discrete choice model

This type of model approaches the pedestrian modelling problems from a choice perspective. The pedestrian choice of next step is usually discretised into a set of possible alternatives, for example, a set of possible speeds with a set of possible directions. The pedestrian's decision output is then predicted based on several factors such as the individual's demographic characteristics, concurrent surrounding situations and some common assumptions such as collision avoidance and maximum utility (or minimum cost). A typical example was the model proposed by Antonini et al (2006), who discreted pedestrian speed into 3 regions including "constant speed", "accelerated" and "decelerated", and discreted pedestrian walking direction into 11 possible patterns characterised by angles of movement.

4. Multi-agent model

The multi-agent model treats pedestrians as different agents (or objects), which have their own characteristics and "artificial intelligence". The different agents can communicate with each other and with their surroundings under a set of defined rules.

This type of model is usually applied in combination with the discrete modelling technique. The decision of any agent is predicted from a set of possible outputs, under the constraints of several behavioural rules. The model allows for a true representation of an agent's behaviour in a microscopic level and therefore is also called micro-simulation model. Further, this modelling technique is the most natural way to work in conjunction with the object oriented computer programming.

5. Conclusion

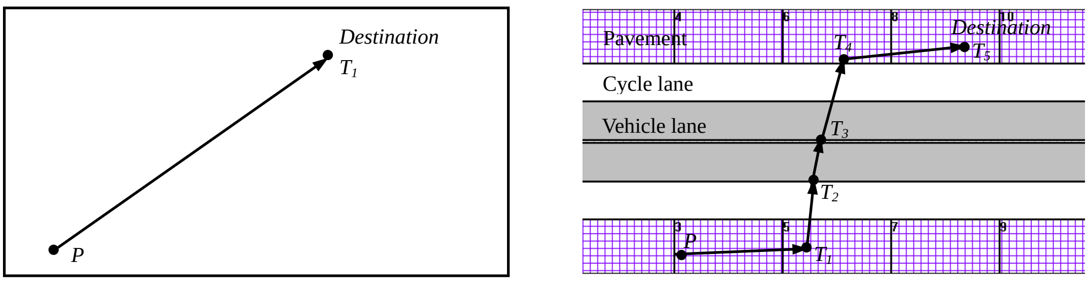
There are several modelling techniques to model pedestrian behaviour. Unlike the vehicle behaviour modelling, there is often an overlap between various techniques due to the complexity of the pedestrian behaviour. In this research, the concepts of agent, discrete modelling are employed to describe the intra pedestrian behaviour as such modelling techniques can work more naturally with object oriented programming. The proposed model is described in details in Section 5.4.2 and the validity of this model is discussed in Section 5.5.

5.4.2 The proposed pedestrian movement model

The pedestrian movement model used in this research is based on Wakim et al (2004) and Antonini et al (2006)'s research on pedestrian behaviour modelling with slight modifications. This is a discrete-time-discrete-choice model. It assumes that the pedestrian can change his/her dynamics from a set of choices at each reaction time. The pedestrian's reaction time follows Green (2000)'s suggestion of 0.7 second, which consists of 0.5 second perception time and 0.2 second movement time. The pedestrian's decision output at each reaction time is characterised by its velocity instead of acceleration. This idea is reasonable based on Blue and Alder (2000)'s suggestion on pedestrian modelling that "pedestrians can vary their speeds very quickly and they have almost instantaneous acceleration profiles" and Bierlaire and Robin (2009)'s suggestion that "the pedestrian speed choice set can be a list of possible absolute speeds, ranging from 0 to maximum possible speed that can be achieved by a pedestrian, discretised in some appropriate way". The velocity as a decision output is further discretised into a finite set of directions and a finite set of walking speeds, as discussed in the following paragraphs.

The position of a pedestrian at any given time is denoted by a 2-layer coordinate system. One type of coordinate, (x_p, y_p), represents the actual position of the pedestrian in the simulation area. The other type of coordinate is achieved by dividing the whole simulation area with a number of 3.0 m × 3.0 m cells, in any of which the maximum pedestrian occupancy is assumed 18 ped/cell (0.5 m^2/ped) in an urban street environment according to Li et al (2007). The row and column *ID* numbers (i, j) of the cell which the pedestrian is currently in is used as the second coordinate of the pedestrian. The use of the second layer coordinate is to ensure the basic pedestrian flow-speed-density relationship is reasonable, which will be discussed later in this chapter.

To continue the discussion, it is necessary to introduce the concept of “target positions” of a pedestrian. The target positions are several temporary “way points” in the pedestrian’s movement route. If there is no interference from vehicle traffic, the target position of a pedestrian is always his/her destination position at any time. However, in the interaction environment, the target position of a pedestrian may vary according to the concurrent traffic situation and should be determined in combination of the pedestrian’s final destination and the influence of motorised traffic. This concept is illustrated in Figure 5.3.



(a) No interference of vehicle traffic (b) With interference of vehicle traffic

P is the pedestrian’s current position; T_i represents several target positions along the pedestrian’s route.

Figure 5.3 The concept of pedestrian target positions

The determination of a pedestrian’s target positions near vehicle traffic will be discussed in Chapter 6. In this chapter, it is assumed to be exogenous to the intra pedestrian model. Assuming the current position of a pedestrian is $P(x_p, y_p)$ and the target position of this pedestrian is $T(x_t, y_t)$. A local coordinate system can be established as shown in Figure 5.4.

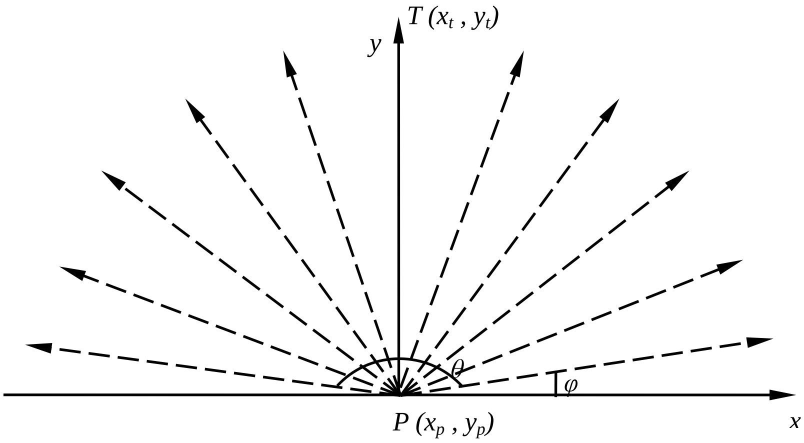


Figure 5.4 The discretisation of pedestrian walking direction

In Figure 5.4, the direction of vector $\overrightarrow{PT}$ is defined as the pedestrian's desired direction. Assuming the pedestrian's visual angle is θ, which is typically $\frac{17\pi}{18}$, and the region of interest is situated only in front of the pedestrian (Antonini et al 2006), the pedestrian's direction choice set can then be obtained by discretising the entire visual angle into a finite set $\Phi = \left\{\varphi \middle| \varphi = \frac{\pi}{2} + \frac{i-5}{10}\theta,\ i \in \mathbb{Z}\ and\ 0 \leq i \leq 10\right\}$, where φ indicates the pedestrian's walking direction. For example, $\varphi = \frac{\pi}{2}$ means that the pedestrian chooses its desired direction, which is denoted as $\varphi_{Ped,DSR}$.

Similarly, assuming the maximum walking speed of the pedestrian is $v_{Ped,max}$, the pedestrian's walking speed choice set can then be obtained by discretising the whole speed region into a finite set $V = \left\{v \middle| v = \frac{j}{10} v_{Ped,max},\ j \in \mathbb{Z}\ and\ 0 \leq j \leq 10\right\} \cup \left\{v_{Ped,DSR}\right\}$, where $v_{Ped,DSR}$ is the pedestrian's desired (free) walking speed.

Therefore, combining the speed set V and the direction set Φ, a decision choice set of $11 \times 11 = 121$ or $11 \times 12 = 132$ alternatives is generated, where each alternative corresponds to a combination of a walking speed $v \in V$ and a direction $\varphi \in \Phi$. Then, at each reaction time, the pedestrian makes a decision from its decision choice set based on the behavioural rules described in Figure 5.5.

Step (1): Calculate the desired speed and direction for each pedestrian and mark the decision status as “not updated”.

Step (2): Traverse the set of pedestrians whose decision status is “not updated” in a random order; if all pedestrians have been updated, go to step (8); else, find such a pedestrian Ped_i.

Step (3): For Ped_i, if it is not entering a new cell, assuming all pedestrians in the same cell can well negotiate their positions, update its speed and direction; return to Step (2). Else, Ped_i is entering a new cell; go to the next step.

Step (4): If the number of pedestrians in the target cell of Ped_i has not reached the maximum value, update the speed and direction of Ped_i; return to Step 2. Otherwise, go to the next step.

Step (5): In the target cell of Ped_i, if there is no pedestrian entering the cell which Ped_i is currently in, recalculate the next best choice of Ped_i from its decision choice set V and Φ; go to Step (3). Otherwise, go to the next step.

Step (6): There exists a pedestrian Ped_j entering the cell which Ped_i is currently in. Update the speeds and directions of Ped_i and Ped_j simultaneously with a probability parameter k ($0 \leq k \leq 1$). The introduction of the parameter k is learnt from the similar concept in cellular automata models (Schadschneider and Seyfried 2009), in order to mimic the friction among pedestrians when they are in a bi-directional flow condition in a limited space. It is apparent that a higher flow can be achieved as the value of k increases and vice versa. A valid model should be able to generate reasonable bi-directional pedestrian flow with a properly calibrated k according to the local situation. This will be discussed in the pedestrian model validation process in Section 5.5. If Ped_i and Ped_j have updated their speeds and directions, mark their decision status as “updated” and go to Step (2). Otherwise, repeat this step until all pedestrians in the target cell of Ped_i have been checked and Ped_i still has not updated its speed and direction; go to the next step.

Step (7): Ped_i waits at its current position; set its decision status to “updated” and return to Step (2).

Step (8): Finish.

Figure 5.5 The logic of the proposed pedestrian movement model

The above logic is illustrated in the flow chart shown in Figure 5.6.

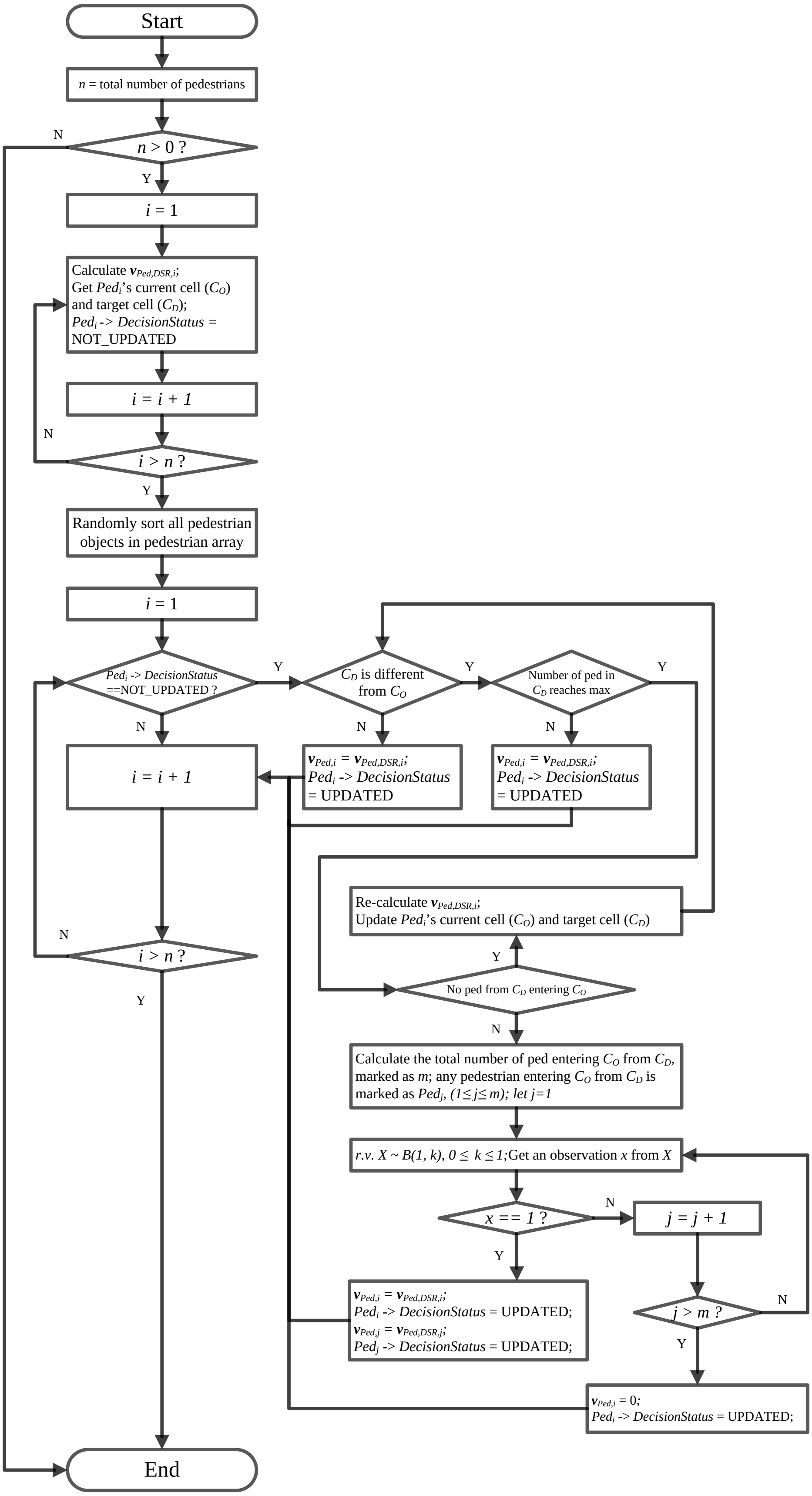


Figure 5.6 The flow chart of the proposed pedestrian movement model

The pedestrian's desired and maximum walking speeds were calibrated with the following experiment. The friction parameter was determined in the model validation process described in Section 5.5.

The experiment for the calibration of pedestrian's desired and maximum walking speed was carried out in an outdoor open space where there was no interference to the subject. A straight path with a distance of 30 m was designated, with the positions of 0 m, 10 m, 20 m and 30 m from the starting point being marked by the researcher, as shown in Figure 5.7.

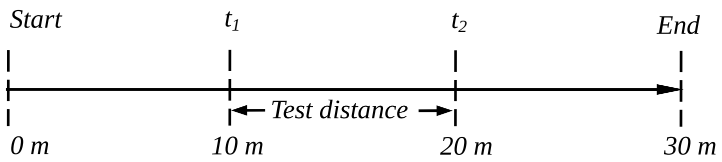


Figure 5.7 Estimation of pedestrian walking speed

First, the gender and age group of the subject was recorded by the researcher. Then, each subject was instructed to walk along the designated path with his/her desired and maximum walking speed, respectively. For the desired speed estimation, they were instructed to walk at their normal comfortable (natural) speed. For the maximum speed estimation, they were asked to walk as fast as they could safely without running (Bohannon 1997). The instantaneous times t_1 and t_2, when the subject just reached the positions of 10 m and 20 m respectively, were recorded by the observer non-intrusively with a stop watch. Then the desired/maximum walking speed of the subject was estimated by the average speed between t_1 and t_2.

A total of 200 people were selected in Beijing, China as subjects in this experiment. The result showed that the pedestrian desired and maximum walking speeds for each age-gender group can be described with a normal distribution, whose parameters could be estimated from the mean and standard deviation of the sample, as shown in Table 5.1 and Table 5.2 respectively.

Table 5.1 Observed pedestrian desired walking speed

Age group	Gender	Mean (m/s)	Standard deviation (m/s)	Minimum (m/s)	Maximum (m/s)	Sample size
Younger	Male	1.44	0.16	1.22	1.79	50
Younger	Female	1.38	0.15	1.14	1.63	50
Older	Male	1.29	0.17	0.98	1.55	50
Older	Female	1.22	0.19	0.90	1.50	50

Table 5.2 Observed pedestrian maximum walking speed

Age group	Gender	Mean (m/s)	Standard deviation (m/s)	Minimum (m/s)	Maximum (m/s)	Sample size
Younger	Male	2.38	0.35	1.83	3.08	50
Younger	Female	2.24	0.28	1.78	3.01	50
Older	Male	2.01	0.36	1.63	2.49	50
Older	Female	1.86	0.27	1.52	2.20	50

5.5 Validation of the pedestrian behaviour model

The common method to validate a pedestrian model is to test whether the model can generate reasonable pedestrian flow at a bottleneck. In the context of urban street environment, there exists a bottleneck at any signalised pedestrian crossing facility. Due to the limitation of its physical size, the total number of pedestrians (2-way) passing the crossing facility during a period of pedestrian green time has an upper limit, which is defined as the capacity of this crossing facility. Previous studies show that the capacity (2-way) of a fixed-signal pedestrian crossing in Beijing, China is around 2400 ped/m/h_G ("/h_G" stands for "per hour of pedestrian green time") (You 2004). Therefore, the friction parameter in the pedestrian model should be calibrated in order that the model can generate a pedestrian flow near this level at a pedestrian crossing facility.

A fixed-time signalised pedestrian crossing with a typical timing plan shown in Figure 5.8 was created in the simulation. A very large pedestrian crossing demand (2-way), 10000 ped/h, was assigned to the crossing during the pedestrian green period, in order to obtain the capacity of the crossing facility, as shown in Figure 5.9. The pedestrian crossing capacity (2-way) was then calculated by Equation 5.1.

$$\frac{\frac{1}{n}\sum_{i=1}^{n} N_i}{t_G \cdot w} \times 3600 \tag{5.1}$$

Where,

n is the number of pedestrian green periods during the simulation time;

i is the i^{th} pedestrian green period;

N_i is the total number of pedestrians passing the crossing facility (2-way) during i^{th} pedestrian green period;

t_G is the time of each pedestrian green period, (s);

w is the width of the pedestrian crossing facility, (m).

Period	P1 (50 s)	P2 (3 s)	P3 (2 s)	P4 (20 s)	P5 (5 s)
Veh. signal	G	A	R	R	R
Ped. signal	R	R	R	G	R

Figure 5.8 Signal timing for validation of pedestrian crossing capacity

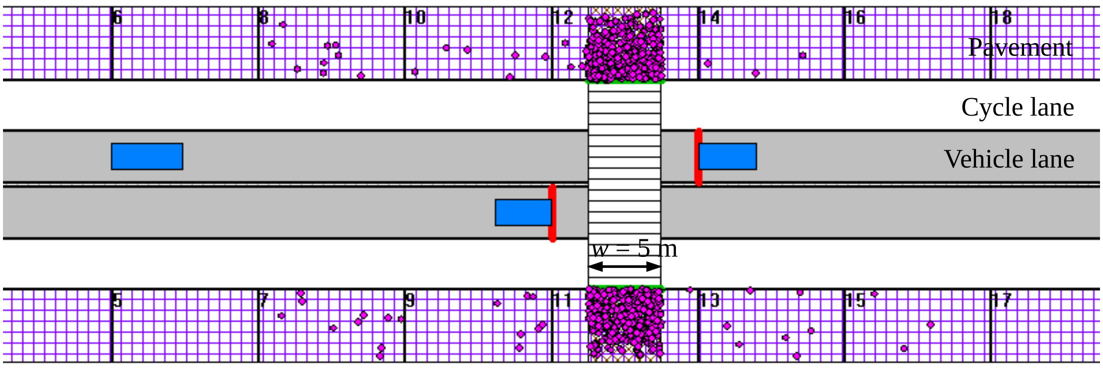


Figure 5.9 Validation of pedestrian crossing capacity

The simulation program was run for several times with different values of parameter k introduced in Figure 5.5. For each simulation run, the composition of types of pedestrians was set to 41% YM, 38% YF, 11% OM and 10% OF, which was observed from a typical signalised pedestrian crossing in Beijing, China. It was noted that when k was around 0.40, the capacity of the pedestrian crossing was close to the local value (2400 ped/m/h_G), as shown in Figure 5.10.

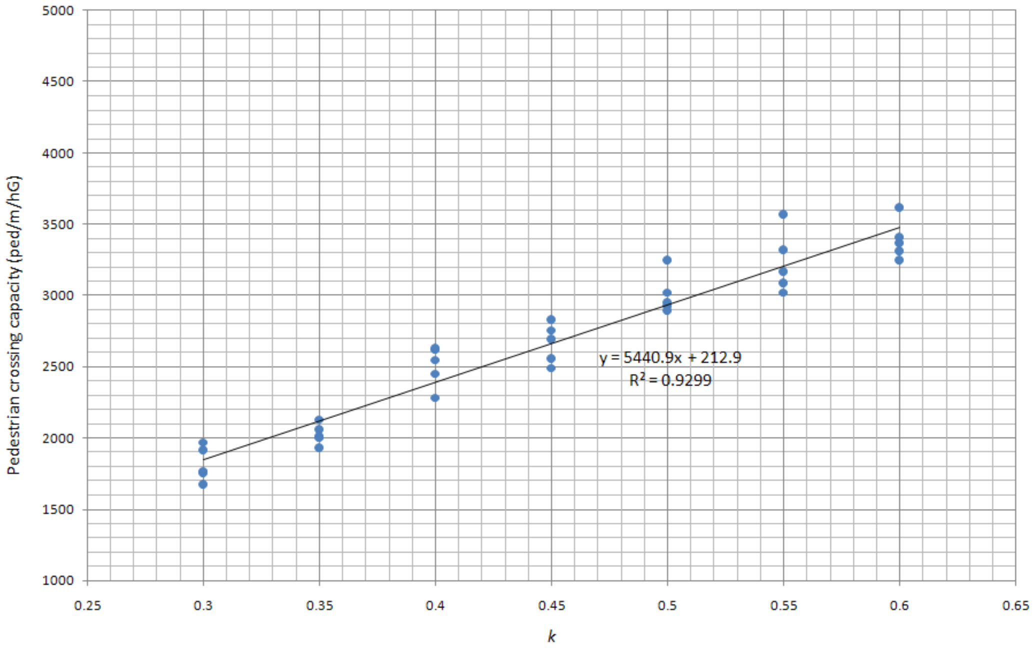


Figure 5.10 Pedestrian crossing capacity (2-way) varying with parameter *k*

Therefore, the simulation was run with *k*=0.40 for further 10 times to obtain a series levels of pedestrian crossing capacity, as shown in Table 5.3, in order to conduct a statistical test to check if the model is able to generate reasonable pedestrian flows in terms of pedestrian crossing capacity.

Table 5.3 Validation of pedestrian crossing capacity

Number of simulation	Pedestrian crossing capacity (2-way) (ped/m/h_G)
1	2433
2	2183
3	2333
4	2313
5	2256
6	2571
7	2584
8	2386
9	2354
10	2325

As a statistical test showed that there was not enough evidence to reject that the capacity data yield to a normal distribution (see Appendix I), the following 1-sample t test was conducted to test the mean of pedestrian crossing capacity data.

Test method: 1-sample t test.

Test hypotheses: H_0: the mean of the pedestrian crossing capacity equals to 2400; H_1: the mean of the pedestrian crossing capacity does not equal to 2400.

The result of the test is shown in Table 5.4, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.530). Therefore, it is reasonable to accept H_0. In conclusion, with a properly calibrated parameter *k*, the intra pedestrian model can generate reasonable pedestrian flows in terms of the capacity of the pedestrian crossing facility.

Table 5.4 The result of 1-sample t test for pedestrian crossing capacity

	Test Value = 2400					
					95% Confidence Interval of the Difference	
	t	df	Sig. (2-tailed)	Mean Difference	Lower	Upper
PedCrossingCapacity	-.652	9	.530	-26.200	-117.04	64.64

5.6 Conclusion

This chapter described the development, calibration and validation of the intra-pedestrian model used in this research. First, the pedestrians were divided into 4 categories based on their age and gender. Then, an innovative pedestrian generation model was proposed incorporating the concept of pedestrian crossing origin and destination matrix, which was used to describe the crossing demand along the road section being studied. Next, the pedestrian movement model was developed using discrete choice model approaches suggested by Wakim et al (2004) and Antonini et al (2006), with some modifications to enable it to be integrated with vehicle models. Key parameters of this movement model were calibrated by experiments designed by the author and actual on-road data collected at typical street sections in Beijing, China. Finally, the intra-pedestrian model was validated in terms of pedestrian crossing capacity. Results showed that the model could generate a reasonable level of pedestrian flow at the bottleneck of pedestrian crossing facility.

CHAPTER 6
PEDESTRIAN-VEHICLE INTERACTION BEHAVIOUR AND MODELS

6.1 Introduction

The pedestrian-vehicle interaction process to be modelled in this research is defined as the complete process starting at the moment when a pedestrian emerges at its crossing origin on one side of a road and ending at the moment when the pedestrian finishes the crossing at its crossing destination on the other side of the road. This process involves pedestrian behaviour from both tactical level (e.g. crossing route choice, whether to use a nearby crossing facility, etc) and operational level (e.g. gap acceptance, on-road movement, etc). Existing research mostly treats these two levels of pedestrian behaviour separately. Current simulation models on pedestrians' tactical behaviour are mainly oriented to crowd dynamics and they seldom incorporate the interactions between pedestrians and traffic. On the other hand, in the operational level, current simulation modelling are mostly regarding pedestrian crossing behaviour at a specific location, usually at a signalised crossing; few have combined the crossing process with the walking process, considering pedestrian behaviour near the vehicle traffic and vice versa, or involved the more complex interaction of the two modes at pure unsignalised areas (e.g. pedestrian jaywalking behaviour). Further, most observational studies do not include model development and existing simulation models can hardly be credible to be used to conduct applications in a highly mixed-traffic condition, which is one of the most significant characteristics in China's urban transport system.

As there is little guidance in the literature on how to model the complete interaction behaviour, a qualitative observation of the interaction was carried out in this research, in order to inspire a framework in a bottom-up way to model the interaction behaviour of the two modes. Based on the observation, the complete interaction process was abstracted into several sub-modules for modelling convenience. For each sub-module, the model was established based on behavioural interpretation from the new collected data or assumptions made based on previous studies or the qualitative observation. These will be discussed in subsequent sections in this chapter.

6.2 Model framework

The qualitative observation was carried out in a 2-way-2-lane road section in Beijing, China. A Cartesian coordinate system was established as shown in Figure 6.1 (Hereafter in this chapter, the analysis regards to pedestrian crossing activities from south to north and the coordinate system uses eastbound and northbound as the positive directions for *x* and *y* coordinates respectively. The same definition and principle apply to those who cross from north to south, but with a modification of the coordinate in the *y* direction accordingly.). The trajectories of any pedestrian crossing activity from *Area O* to *Area D* were recorded using the video camera method discussed in Chapter 3, in order to inspire a framework for modelling convenience. This work has been previously presented at a conference by the author during candidature (Wang et al 2010a).

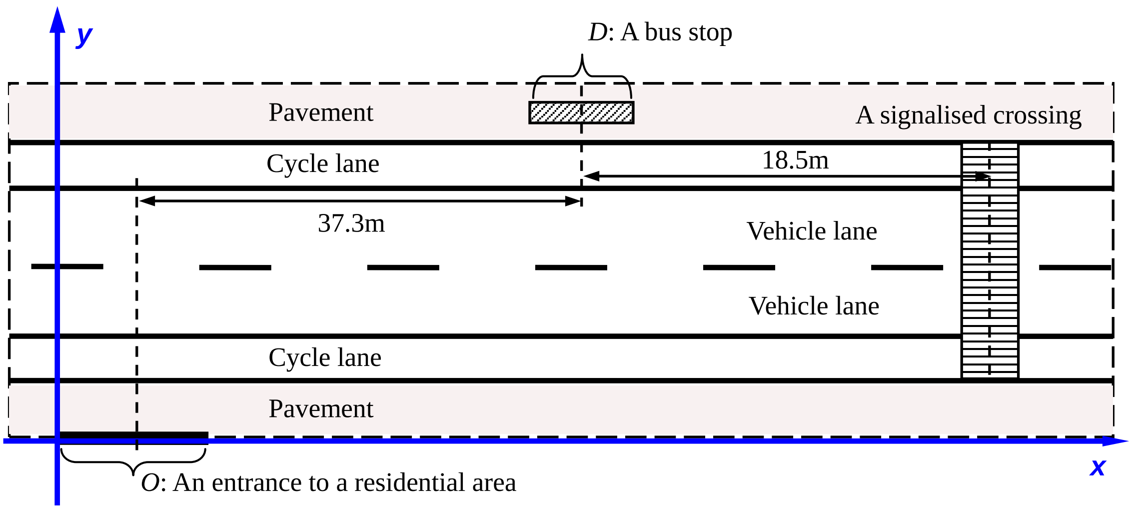


Figure 6.1 An illustration of the site chosen for qualitative observation of pedestrian crossing activities

(Source: Wang et al 2010a)

To exclude the influence of cyclists and buses, only samples without adjacent interference of cyclists and buses were recorded. Pedestrians walking with bikes, luggage, etc were also excluded. Some typical patterns of pedestrian trajectories are shown in Figure 6.2.

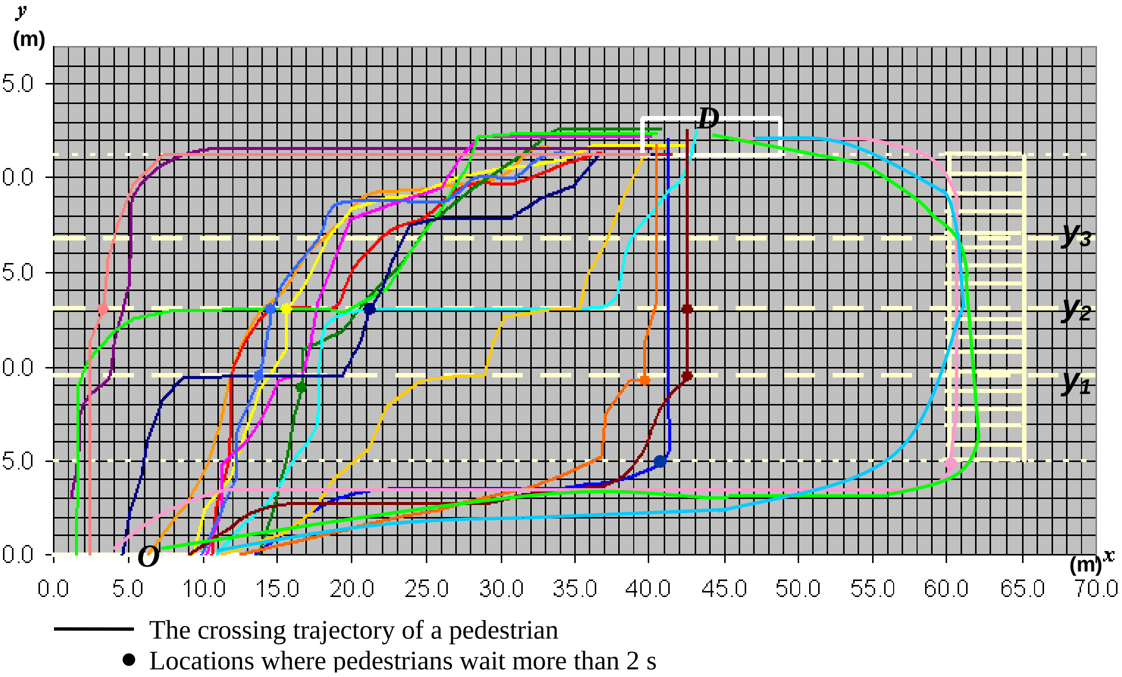


Figure 6.2 Some typical patterns of pedestrian moving trajectories
(Source: Wang et al 2010a)

In conjunction with Figure 6.2, the following main phenomena were noted during the qualitative observation.

First, pedestrians applied different tactics to approach the vehicle lanes (from *O* to y_1). Some pedestrians were likely to use the nearby crossing facility while others not. For those who chose to jaywalk, they seemed to keep monitoring the condition of the traffic from the beginning of the crossing process (observable according to their head movement) and somehow decided a location to leave the pavement for y_1 (the edge of the nearside vehicle lane).

Second, for pedestrians who did not use the pedestrian crossing, at y_1, they evaluated the condition of the road traffic and decided to step on the road when there was an appropriate time gap on the nearside vehicle lane. However, they did not necessarily wait until both of the vehicle lanes were cleared. Some pedestrians encountered unaccepted gaps and waited at or went along the median (y_2) of the road until the time gap on the next lane to be crossed was acceptable.

Third, while on the vehicle lanes, the trajectories of pedestrians were somehow attracted to their destinations. When the time gap was large, they were likely to target at their destinations to gain some advantages on the journey time whereas when the gap was small, they were likely to cross with a direction perpendicular to the road and keen to leave the vehicle lanes.

Fourth, when pedestrians finished crossing all vehicle lanes, they were likely to target directly to their destinations and their behaviour were not likely to be influenced by vehicle traffic behind them (from y_3 to D).

Last, sometimes pedestrians took risk and accepted smaller gaps, and forced the conflicting vehicle to slow down.

Based on the above discussion, the complete interaction process is divided into the following modules for analysis and modelling purpose.

(1) Pedestrian approaching vehicle lanes (from O to y_1), which concerns where a pedestrian decides to cross the road and whether he/she decides to use a nearby crossing facility;

(2) Pedestrian gap acceptance (at y_1 and y_2), which deals with how a pedestrian accepts a gap in the traffic and steps onto a vehicle lane;

(3) Pedestrian on-road movement (from y_1 to y_3), which focuses on how a pedestrian chooses its target position when he/she is on a vehicle lane;

(4) Pedestrian departing from vehicle lanes (from y_3 to D), which deals with pedestrian movement behaviour when he/she finishes crossing all the vehicle lanes;

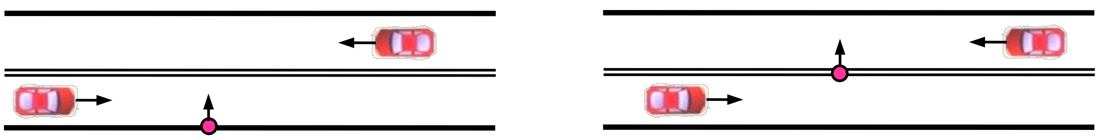
(5) Vehicle reactions to pedestrians, which models the vehicle reaction behaviour

when there is a conflicting pedestrian ahead of it on the vehicle lane.

It should be noted that there is no definite boundary between any two of the modules in the real world. This abstraction is only for the purpose of modelling convenience and its validity will be checked in further model validation process discussed in Chapter 7. The analysis and modelling for each module will be discussed in subsequent sections in this chapter. As the pedestrian gap acceptance behaviour is commonly considered as a major factor for pedestrian delay and safety problems and also may have influences to other aspects of behaviour during the road crossing process (Pan et al 2010), it will be analysed first in the following section.

6.3 Pedestrian gap acceptance

For the specific 2-way-2-lane road in this research, there are two types of pedestrian gap acceptance when he/she stands on the edge of the vehicle lane: the nearside gap acceptance, as shown in Figure 6.3 (a), in which the pedestrian faces two vehicle lanes to cross, and the far-side gap acceptance, as shown in Figure 6.3 (b), in which the pedestrian has already on the median of the road and faces only one vehicle lane to cross. As behaviour may be different in these two scenarios, they will be studied separately. In addition, pedestrian gap acceptance behaviour at Zebra crossings may be different from that at pure unsignalised locations. Therefore, the gap acceptance behaviour at Zebra crossings will be examined separately.



(a) Nearside gap acceptance   (b) Far-side gap acceptance

Figure 6.3 Pedestrian gap acceptance

6.3.1 At locations with no-control

1. Nearside gap acceptance

Existing literature (Schroeder and Rouphail 2007, Bierlaire and Robin 2009, Sun et al 2003) suggests that a pedestrian's decision whether to step onto the first vehicle lane to start to cross is related to the pedestrian's characteristics and the concurrent traffic dynamics. The modelling purpose for this behaviour is to establish the prediction relationship between the probability of a pedestrian accepting a gap and the possible influential factors. For this purpose, the initial idea is to follow the method in FLOWSIM to use fuzzy logic to model pedestrian gap acceptance. However, in reality, such a method is not suitable for this modelling purpose mainly due to the limitations of manual data collection. This is discussed as follows.

As discussed in Section 4.5.2, for FLOWSIM car-following model, the model output is the vehicle's acceleration (*ACC*) and the inputs are *DV* and *DSSD*, as defined in Section 4.5.2. As both the *DV* and *DSSD* are continuous variables and extend to a long range of possible values, extensive driving data, in the type of (*ACC*, *DV*, *DSSD*) are needed to establish the mapping relationship in the fuzzy inference system. In FLOWSIM, such data were collected via automated data collection method using an instrumented vehicle. Thanks to on-board sensors and computers, the instantaneous states of the subject vehicle, as well as both its leader and follower can be computed and recorded immediately. Thousands of (*ACC*, *DV*, *DSSD*) data can be collected within minutes. This is crucial as any fuzzy logic based model needs a huge amount of data to train its inference system (MathWorks 2000).

For pedestrian gap acceptance behaviour modelling, considering a least complex situation only involving two types of inputs such as the time gap in the vehicular lane to be crossed (abbreviated as X_1 hereafter) and the number of pedestrians in a crossing group (abbreviated as X_2 hereafter), which are the mostly mentioned possible influential factors in the literature (Schroeder and Rouphail 2007, Bierlaire and Robin 2009, Sun et al 2003), the amount of data for establish the fuzzy inference system can be estimated as follows.

For X_1, the accuracy and range of the sample data are usually set to be at least 0.1 s and [0.0 s, 10.0 s]. For X_2, which is an integral number, the range of the sample data should be set at least [1, 5] to reflect the actual situation in the real world. Therefore, there are at least 10 × 10 × 5 = 500 different combinations of values of (X_1, X_2). To establish the relationship between the probability of pedestrian accepting a gap (abbreviated as Y hereafter), which is the output of this fuzzy logic model and the given combinations of (X_1, X_2), each (X_1, X_2) should expect a resulting Y. However, as Y is a probability variable, a relatively large sample (sample size is commonly over 50) with the same value of (X_1, X_2) is required to estimate the according Y at this level of (X_1, X_2). Assuming at least half of the total points in the 2 dimensional (X_1, X_2) system have to be populated with Y in order to achieve a relatively accurate model, base on the above discussion, the total number of cases of pedestrian accepting/rejecting a gap can be estimated to be: 500 × 50 × 50% = 12500 at least.

It should be noted that the above discussion is based on a least complex situation considering only two variables as inputs. In reality, the number of possible influencing factors may be more, such as waiting time, pedestrians' type of age and gender, and types of scenarios such as with/without a Zebra crossing, etc. Therefore, there is likely to be a huge demand of data to establish a fuzzy logic model. However, as discussed in Chapter 3, existing data collection techniques are limited on pedestrians. Automated instruments such as civil Global Positioning System (GPS) is not accurate enough to collect pedestrian data to develop microscopic models. The more commonly used manual data collection method, as applied in this research, can ensure a better accuracy but is time consuming. For this research, it can hardly meet the need of vast amount of data for a fuzzy logic model with limited time. Therefore, an alternative modelling approach is needed for pedestrian gap acceptance behaviour. This is discussed in the following paragraphs.

Since the pedestrian has only two alternatives (accept or reject), it is appropriate to describe this process using a binary logit model, which is a commonly used modelling approach proposed by other researchers to study similar problems (Sun et al 2003). In this method, the probability of any pedestrian accepting a gap is given by Equation 6.1.

$$\begin{cases} P(Accept) = \frac{1}{1+\exp(-U)} \\ U = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \end{cases} \quad (6.1)$$

Where,

P(Accept) is the probability of this pedestrian accepting a gap in the traffic;

n is the total number of predictors;

x_i $(1 \leq i \leq n)$ is the value of the i^{th} predictor for this pedestrian;

β_0 is constant; β_i $(1 \leq i \leq n)$ is the i^{th} coefficient for the i^{th} predictor.

The calibration and validation of this binary logit model involves the determination of the parameters β_i $(0 \leq i \leq n)$, which can be achieved by using the data collected from field survey. Of all similar gap acceptance studies, the following factors are mostly mentioned in existing literature.

(1) Pedestrian's type of age (*Age*): 0 for younger and 1 for older (the concept of younger or older has been defined in Section 5.2);

(2) Pedestrian's type of gender (*Gender*): 0 for male and 1 for female;

(3) Pedestrian's accumulative waiting time on the edge of that lane (*WaitTime*), (s);

(4) The number of pedestrians in a crossing group (*PedNum*);

(5) The time gap in the nearside vehicle lane (*GapNear*), (s);

(6) The time gap in the far-side vehicle lane (*GapFar*), (s).

The nearside and far-side time gaps are defined as shown in Figure 6.4 and Figure 6.5 respectively.

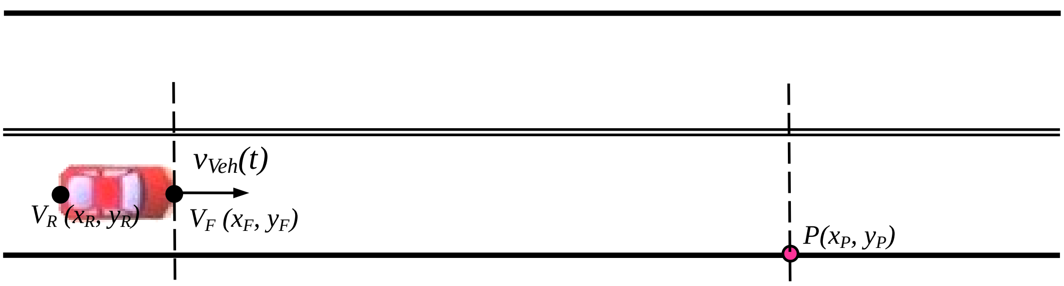


Figure 6.4 Definition of nearside time gap

As shown in Figure 6.4, at any given time t, assuming the instantaneous speed of the conflicting vehicle to the pedestrian is $v_{Veh}(t)$, the coordinates of its front and rear bumper centroid are V_F (x_F, y_F) and V_R (x_R, y_R) respectively, the coordinate of the pedestrian is P (x_P, y_P). The nearside time gap (t_N) is given by Equation 6.2.

$$\begin{cases} t_N = \frac{x_P - x_F}{v_{Veh}(t)}, & x_P > x_F \\ 0, & x_R \le x_P \le x_F \end{cases} \tag{6.2}$$

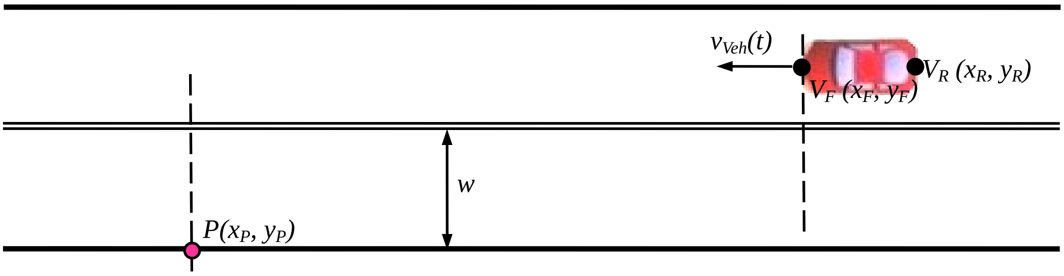


Figure 6.5 Definition of far-side time gap

The definition of the far-side time gap is similar to the nearside time gap with an exception that it considers the width (w) of the vehicle lane and pedestrian's desired speed ($v_{Ped,DSR}$). Assuming the pedestrian and the conflicting vehicle have the dynamics shown in Figure 6.5, the far-side time gap (t_F) can then be given by Equation 6.3.

$$\begin{cases} t_F = \frac{x_F - x_P - \frac{w \cdot v_{Veh}(t)}{v_{Ped,DSR}}}{v_{Veh}(t)}, & x_F - x_P - \frac{w \cdot v_{Veh}(t)}{v_{Ped,DSR}} > 0 \\ 0, & x_F - x_P - \frac{w \cdot v_{Veh}(t)}{v_{Ped,DSR}} \le 0 \; and \; x_R - x_P - \frac{w \cdot v_{Veh}(t)}{v_{Ped,DSR}} \ge 0 \end{cases} \tag{6.3}$$

If the pedestrian's gap acceptance is denoted by a variable *PGA* (1 for accept and 0 for reject), the general form of the binary logit model for pedestrian gap acceptance can then be express by Equation 6.4.

$$\begin{cases} P(PGA = 1) = \frac{1}{1+exp(-U)} \\ U = \beta_0 + \beta_1 Age + \beta_2 Gender + \beta_3 WaitTime + \beta_4 PedNum + \beta_5 GapNear + \beta_6 GapFar \end{cases} \tag{6.4}$$

The parameters $\beta_0, \beta_1, \beta_2, \cdots, \beta_6$ need to be calibrated according to the local data, which were collected in Jiaoda East Road, Beijing, China, a typical 2-way-2-lane road section with a speed limit of 30 km/h, as shown in Figure 6.6.

Figure 6.6 Data collection site for calibration of pedestrian gap acceptance model

The data were collected and extracted using the video camera method discussed in Chapter 3. For any pedestrian standing on the edge of the nearside lane, his/her gap acceptance decision and the six proposed variables in Equation 6.4 were recorded. For those who rejected a number of gaps, only one of the rejected decisions was randomly selected to eliminate sampling bias. The type of age of a pedestrian was estimated by the author subjectively, excluding those whose type of age could not be easily identified. Although this method was subjective than the pedestrians' self reported data, it was a commonly accepted method in practice to obtain such data whilst the subjects to be surveyed were kept unaware.

A total of 600 data sets were collected, of which 70% were used to estimate the coefficient of the regression function and 30% for model validation. A multiple variable regression analysis was performed in SPSS, which estimated the coefficients of the linear utility function using the maximum likelihood method. The results of the estimated coefficients are shown in Table 6.1.

Table 6.1 Estimated coefficients for nearside gap acceptance model at a location with no-control

	β	S. E.	Wald	*df*	Sig.	*Exp(β)*
Constant	-7.207	0.852	71.553	1	0.000	0.005
Age	-2.225	0.847	6.891	1	0.009	0.108
PedNum	0.619	0.309	4.007	1	0.045	1.858
GapNear	1.389	0.226	37.859	1	0.000	4.012

Only three variables of the proposed six are sufficiently significant to be included in the final model. Therefore, the model expressed with Equation 6.4 can then be calibrated to Equation 6.5.

$$\begin{cases} P(PGA = 1) = \frac{1}{1+exp(-U)} \\ U = -7.207 - 2.225\ Age + 0.619\ PedNum + 1.389\ GapNear \end{cases} \tag{6.5}$$

Table 6.2 shows the modelling and validation results. The validation was performed against 30% of the total data not used for model development. The validation result showed that the percentage of the overall correct prediction of the above binary logit model is 91.0%.

Table 6.2 Modelling and validation results of nearside gap acceptance model at a location with no-control

<table>
<tr><th colspan="2" rowspan="4">Observed</th><th colspan="6">Predicted</th></tr>
<tr><th colspan="3">Cases for modelling</th><th colspan="3">Cases for validation</th></tr>
<tr><th colspan="2">PGA</th><th rowspan="2">Correct %</th><th colspan="2">PGA</th><th rowspan="2">Correct %</th></tr>
<tr><th>0</th><th>1</th><th>0</th><th>1</th></tr>
<tr><td rowspan="2">PGA</td><td>0</td><td>300</td><td>10</td><td>96.8</td><td>98</td><td>8</td><td>92.5</td></tr>
<tr><td>1</td><td>20</td><td>114</td><td>85.1</td><td>6</td><td>44</td><td>88.0</td></tr>
<tr><td colspan="2">Overall %</td><td>N/A</td><td>N/A</td><td>93.2</td><td>N/A</td><td>N/A</td><td>91.0</td></tr>
</table>

In conclusion, for pedestrian nearside gap acceptance behaviour at a pure unsignalised location, three factors including nearside time gap, pedestrian's age type and number of pedestrians in a crossing group out of six proposed ones are identified sufficiently significant to be included into the binary logit model. The following aspects can be noted from the calibrated model.

(1) The pedestrian's waiting time is left out of the model. This fact is not surprising as it was found by different researchers when they studied the pedestrians' gap acceptance behaviour at marked crossing facilities that the increase of waiting time could result in either shorter or larger accepted gaps (Schroeder 2008). Some explained that pedestrians tended to exhibit more risky behaviour when waiting a longer time while others were in favour that pedestrians who still waited after long waiting time tended to be careful in nature and therefore would hardly accept a short or risky gap (Schroeder 2008).

(2) The far-side time gap is not significant to be included in the model. This means that pedestrians are likely to cross the road lane by lane to minimise delay instead of waiting until all lanes are cleared. This is in accordance with the qualitative observation conducted by the author. The raw video data showed that pedestrians hardly paid attention to the far-side incoming vehicles when they started to cross the first lane. Many pedestrians crossed the road regardless of the far-side time gap, even when the far-side time gap was too small to be accepted, resulting in that such pedestrians wait at the median of the road for the next possible gaps in the far-side lane to continue the crossing behaviour.

(3) Older pedestrians are more cautious and wait for longer nearside gaps to start the crossing. Therefore, higher proportion of older pedestrians can lead to more pedestrian delays when there are fewer available gaps on the nearside vehicle lane during busy hours.

(4) As the size of a waiting group increases, pedestrians become more aggressive and thus may accept smaller gaps. Video data show that sometimes they can even force the conflicting vehicles to slow down or stop. One possible explanation is that pedestrians in a large group may feel safer, more confident, or protected by each other and thus may act more aggressively to accept smaller gaps. Therefore, although the vehicular traffic has priority at such unsignalised locations, it may still be influenced by pedestrian crossing activities when pedestrian crossing demand is high.

(5) A drawback of this model is that a pedestrian may accept a rather small gap when the size of the waiting group is sufficiently large. To overcome this disadvantage, the following condition is added to the micro-simulation program: if $t_N \leq \frac{w}{v_{Ped,max}}$, then $P(PGA = 1) = 0$, where t_N is the nearside time gap, w is the width of the vehicle lane and $v_{Ped,max}$ is the maximum walking speed of that pedestrian. This condition represents that if a pedestrian cannot finish crossing the vehicle lane within the time gap with its maximum walking speed, it will not accept that gap.

2. Far-side gap acceptance

As discussed above, pedestrians are likely to cross the road one lane at a time and only the time gap in the immediate lane to be crossed is influential to their gap acceptance decisions. Therefore, it is reasonable to assume that the nearside gap acceptance model should be applicable to describe their far-side gap acceptance behaviour, with a modification that the nearside time gap is switched to the gap in the next immediate lane to be crossed.

To validate this assumption, 500 cases of pedestrian gap acceptance at the median of the road were surveyed by the author. The site and method for the survey were the same with the one used for modelling the nearside gap acceptance. For each case, the pedestrian's gap acceptance decision from the field observation and from the model were compared, the result showed that the nearside gap acceptance model was sufficiently accurate to be used to describe the far-side gap acceptance behaviour, as shown in Table 6.3.

Table 6.3 Validation result of far-side gap acceptance at a location with no control

Observed		Predicted		
		PGA		Correct %
		0	1	
PGA	0	238	22	91.5
	1	27	213	88.8
Overall %		N/A	N/A	90.2

6.3.2 At Zebra crossings

In China, pedestrians are not given full priority at Zebra crossings. Their behaviour at such locations is similar to pure uncontrolled location in that motorists seldom give way to pedestrians at Zebra crossings and pedestrians have to wait for acceptable gaps in the traffic to cross the road (Chen et al 2008). Therefore, the method to modelling pedestrian gap acceptance at Zebra crossings is the same to that applied at pure no-control location as discussed in the previous section, with an exception that the gap acceptance data are collected at a Zebra crossing.

1. Nearside gap acceptance

A total of 500 data sets were collected for calibration of the model parameters. The result of the estimated coefficients is shown in Table 6.4.

Table 6.4 Estimated coefficients for nearside gap acceptance at Zebra crossings

	β	S. E.	Wald	*df*	Sig.	*Exp(β)*
Constant	-7.056	0.924	58.314	1	0.000	0.005
Age	-2.134	0.919	5.392	1	0.006	0.106
PedNum	0.658	0.335	3.858	1	0.035	1.714
GapNear	1.407	0.245	32.980	1	0.000	4.379

Therefore, the gap acceptance model at Zebra crossings can be expressed with Equation 6.6.

$$\begin{cases} P(PGA=1)=\frac{1}{1+exp(-U)} \\ U=-7.056-2.134\ Age+0.658\ PedNum+1.407\ GapNear \end{cases} \quad (6.6)$$

It can be seen that the form of the gap acceptance model at Zebra crossings is similar to that at a location with no-control. Compared to a no-control location, the coefficients of the model at Zebra crossings are slightly higher, indicating that pedestrians are more aggressive at such locations. Table 6.5 shows the modelling and validation results at Zebra crossings. The validation was performed against 30% of the total data not used for model development. The validation result showed that the percentage of the overall correct prediction of the above binary logit model is 90.4%.

Table 6.5 Modelling and validation results of nearside gap acceptance model at Zebra crossings

Observed		Predicted					
		Cases for modelling			Cases for validation		
		PGA		Correct %	PGA		Correct %
		0	1		0	1	
PGA	0	201	9	95.7	101	10	91.0
	1	17	106	86.2	6	50	89.3
Overall %		N/A	N/A	92.2	N/A	N/A	90.4

2. Far-side gap acceptance

Following the discussion in Section 6.3.1, the validity of the nearside gap acceptance model at Zebra crossings was also examined for far-side gap acceptance behaviour.

A total of 500 cases of pedestrian gap acceptance at the median of the road at a Zebra crossing were surveyed for this purpose. For each case, the pedestrian's gap acceptance decision from the field observation and from the model expressed in Equation 6.8 were compared, the result showed that the nearside gap acceptance model was sufficiently accurate to be used to describe the far-side gap acceptance behaviour at Zebra crossings, as shown in Table 6.6.

Table 6.6 Validation result of far-side gap acceptance model at a Zebra crossing

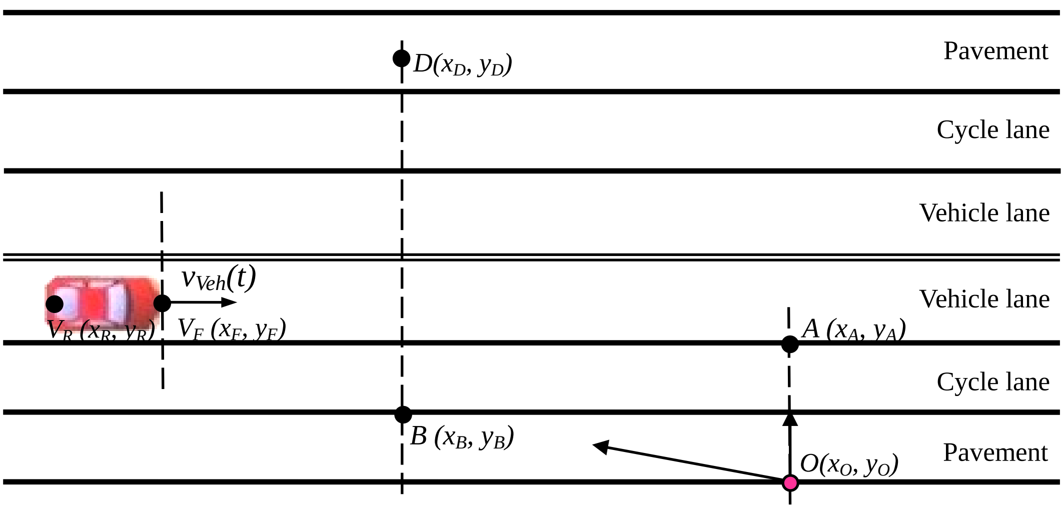
Observed		Predicted		
		PGA		Correct %
		0	1	
PGA	0	226	18	92.6
	1	24	232	90.6
Overall %		N/A	N/A	91.6

6.4 Pedestrian approaching vehicle lanes

This module concerns how a pedestrian plans his/her route to approach the vehicle lanes. It is a link between the intra pedestrian behaviour on pavements and the gap acceptance behaviour on the edge of vehicle lanes. The presence of nearby pedestrian crossing facility may have influence to this behaviour. Therefore, this process was modelled separately in three typical scenarios including locations with no-control, with a Zebra crossing and with a signalised crossing. As it was difficult to collect explicit data on such types of behaviour, some assumptions were made based on the qualitative observations conducted by the author. The validity of those assumptions will be tested in further model validation process presented in Chapter 7.

<u>1. A typical location with no-control</u>

As shown in Figure 6.7, the qualitative observation suggests that the pedestrian behaviour on the pavement can be abstracted to two patterns: if a sufficient gap in the nearside lane is perceived, he/she will move towards *A* to make use that gap; otherwise towards *B* to gain advantage of journey time. When there are no available gaps in the nearside lane, a pedestrian is less likely to stop but to keep moving towards *B* until he/she has reached that point.



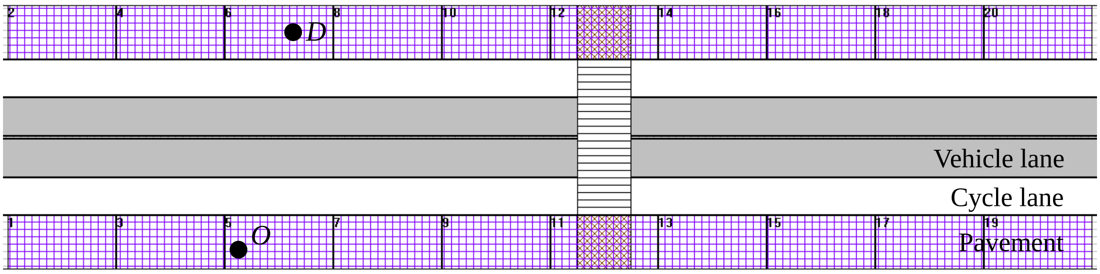
O and *D* are the pedestrian’s crossing origin and destination positions respectively

Figure 6.7 Pedestrian approaching vehicle lanes when there is no nearby crossing

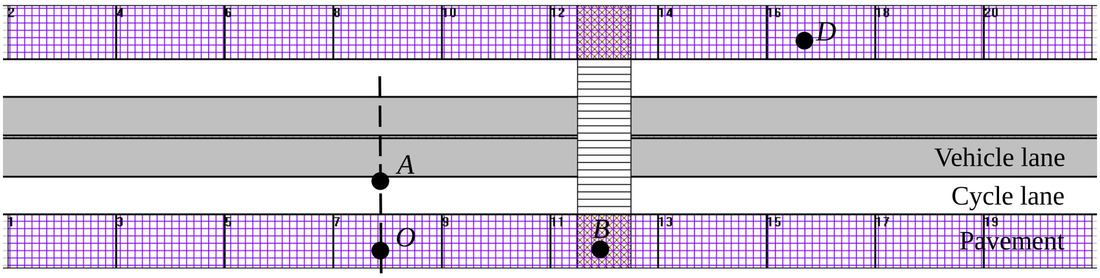
It can also be noted from the qualitative observation that during this process, pedestrians keep monitoring the conditions on the road, in order to make best use of available gaps. As it was found in Section 6.3 that pedestrians were likely to cross the road one lane at a time, it is proposed that the decision of a pedestrian moving towards *A* or *B* is predicted by the gap acceptance model developed in the previous section. If the expected nearside gap is accepted, the pedestrian moves towards *A*; otherwise, towards *B*. During the course from *O* to *B*, the pedestrian keeps monitoring the road condition and repeats this process, until he/she has reached *B*, where they will no longer move but wait for appropriate gap to cross the road. The expected nearside time gap is defined as the projected time gap in the nearside traffic lane if the pedestrian moves from *O* to *A* using his/her desired walking speed at this moment. Therefore, in this module, at any reaction time of a pedestrian, the decision of his/her target position is determined by the following logic: (1) if the expected nearside time gap is accepted, his/her target position is to be set to *A*; (2) otherwise, his/her target position is to be set to *B*.

2. A typical location with a nearby Zebra crossing

When there is a nearby Zebra crossing, pedestrian behaviour can be divided into two types according to the relative position between their crossing *O/D* and the Zebra crossing, as shown in Figure 6.8.



(a) The use of Zebra crossing involves detouring



(b) The use of Zebra crossing does not involve detouring

O and *D* are the pedestrian's crossing origin and destination positions respectively; *B* is a random position within the waiting area of the crossing facility on the same side with *O*

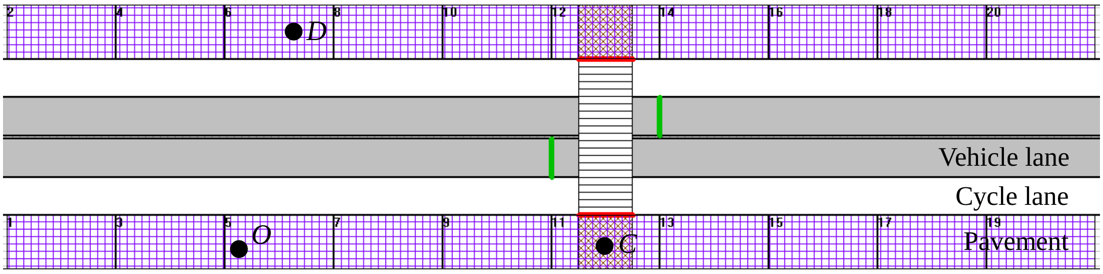
Figure 6.8 Two types of relative position between pedestrian crossing *O/D* and the nearby Zebra crossing

The qualitative field observation suggests that if the use of Zebra crossing involves detouring, pedestrians are not likely to use that facility in that: (1) pedestrians dislike detouring in nature; (2) pedestrians are not given full priority at Zebra crossings in China and thus the sacrifice in journey time cannot bring them much safety advantage. If the use of Zebra crossing does not involve detouring, a pedestrian tends to use that crossing to cross the road. However, if an appropriate expected gap appears on the nearside lane, he/she is likely to accept it and moves towards the vehicle lane instead of heading to the Zebra crossing. If no gaps have been accepted when he/she reaches the Zebra crossing, he/she will no longer move towards his/her destination but wait at the crossing facility. This can be explained as a pedestrian is likely to follow the behaviour of others when they are near a large group of people, which is more likely to emerge at a crossing facility. Therefore, in this module, at any reaction time of a pedestrian, the decision of his/her target position is determined by the following logic: (1) if the use of Zebra crossing involves detouring, as shown in Figure 6.8 (a), his/her target position is to be determined using the logic in a no-control scenario discussed in the previous section; (2) otherwise, as shown in Figure

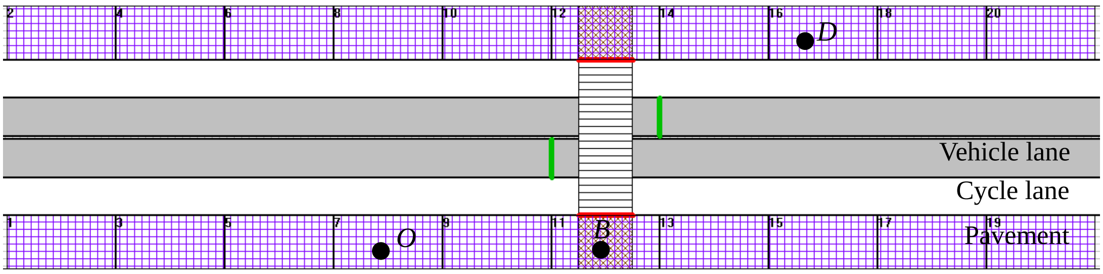
6.8 (b), if the expected nearside time gap is accepted, his/her target position is to be set to *A*; otherwise, to *B*.

3. A typical location with a nearby signalised crossing

Similarly, when there is a nearby signalised crossing, pedestrian behaviour can be divided into two types according to the relative position between their crossing *O/D* and the signalised crossing, as shown in Figure 6.9.



(a) The use of signalised crossing involves detouring



(b) The use of signalised crossing does not involve detouring

O and *D* are the pedestrian's crossing origin and destination positions respectively; *B* and C are random positions within the waiting areas of the crossing facility on the same side with *O*

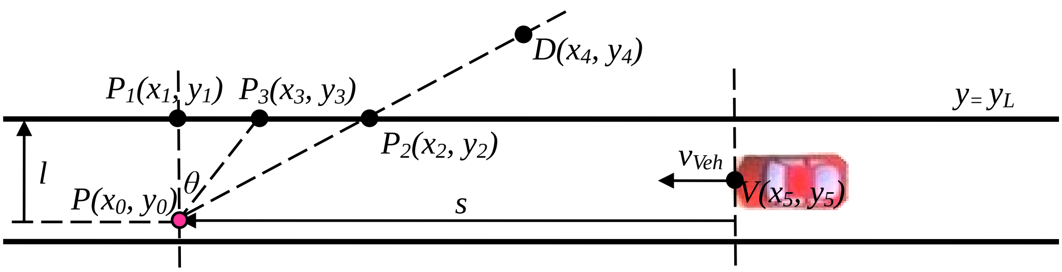
Figure 6.9 Two types of relative position between pedestrian crossing *O/D* and the nearby signalised crossing

The qualitative field observation suggests that if the use of signalised crossing does not involve detouring, pedestrians are likely to choose to cross at the nearby signalised crossing. This is probably because most signalised crossings are located in an area where the traffic is relatively busy. Pedestrians tend to use the signalised crossing to gain safety without losing much efficiency if they do not have to detour. Therefore, in this situation, the decision of a pedestrian's target position is simply assigned to *B*, as shown in Figure 6.9 (b).

For the scenario in which the use of signalised crossing involves detouring, previous research shows that pedestrians are likely to organise their crossing location and timing to minimise walking delay (Daff et al 1991, Sisiopiku and Akin 2003). Therefore, sometimes they may commit risky behaviour such as jaywalking if the perceived decrease in efficiency overweighs the demand for safety. Therefore, the following logic is employed to model this process. For any pedestrian in this situation, the expected journey time (t_E) from its origin position to destination position can be estimated according to the characteristics of this pedestrian and the signal timing plan of the nearby crossing facility. Then, the simulation is pre-run before the pedestrian makes a decision on his/her target position, with the current dynamics of all objects in the simulation, assuming that the pedestrian does not use the crossing facility, from current time stamp to a period of t_E time. If the pedestrian cannot reach its destination during this time, he/she will choose to use the signal crossing because the jaywalking behaviour cannot bring any advantage. In this case, his/her target position is set to *C*, as shown in Figure 6.9 (a). Otherwise, the jaywalking manoeuvre is applied to minimise delay. In this case, the pedestrian's target position is determined using the logic in a no-control scenario discussed previously. This pre-simulation method is rational in that it can mimic the anticipatory aspects in the crossing process. Previous research regarding human intelligence suggests that anticipation is one of the most primitive functions of human intelligence. The mechanism of anticipation has been previously justified by Rosen (1985) using rigorous mathematical methods. The anticipation function of human intelligence suggests that a pedestrian may be able to recognise the near future state of the road section based on a large set of percepts describing the present state and can take actions to minimise his/her delay.

6.5 Pedestrian on-road movement

This section presents the analysis of how a pedestrian chooses his/her speed and direction when he/she is on a vehicle lane.



$(x_4 - x_0)\ (x_5 - x_0) > 0$

Figure 6.10 Pedestrian on-road movement (Scenario 1)

As shown in Figure 6.10, assuming a pedestrian is on a vehicle lane (either nearside or far-side), his/her current position is $P\ (x_0, y_0)$; Line $y=y_L$ stands for the far-side edge of this lane; $P_1\ (x_1, y_1)$ is the foot of perpendicular from P to $y=y_L$; $D\ (x_4, y_4)$ is the crossing destination of this pedestrian; $P_2\ (x_2, y_2)$ is the point of intersection between PD and $y=y_L$; $P_3\ (x_3, y_3)$ is the actual point when the pedestrian arrives at the far-side edge of this lane; and $V(x_5, y_5)$ is the centroid of the vehicle's front bumper.

Although the Highway Code in China suggests pedestrian not cross the road diagonally, the qualitative observation shows that P_3 is likely to fall somewhere between P_1 and P_2 (i.e. $x_1 \leq x_3 \leq x_2$). This can be explained from the social force model's point of view. The pedestrian is attracted by a force perpendicular to the vehicle lane towards P_1 to escape from the vehicle lane, whilst it is also dragged by a force pointing from its current position P to its destination D to gain advantage on journey time. Assuming the instantaneous movement direction of the pedestrian is designated by an angle θ, which is the deviation angle from $\overrightarrow{PP_1}$ to $\overrightarrow{PP_3}$, the aggregation of the two forces results in $0 \leq \theta \leq min\{\angle P_1PD,\ \theta_F\}$ at any given time when the pedestrian is on the road, where θ_F is the maximum deviation angle when there is no conflicting vehicles on that lane.

In order to calculate the decisions of speed v and direction θ of a pedestrian when he/she reaches his/her reaction time on the vehicle lane, it is necessary to introduce another parameter t_M, the "minimum crossing time margin", which is defined as the minimum projected time-to-collision a pedestrian will accept when it just finishes crossing the vehicle lane if the speed and direction of the pedestrian and the conflicting vehicle keep unchanged. The pedestrian's decision can then be determined according to the following logic.

1. Scenario 1: (x_4-x_0) (x_5-x_0) > 0, as shown in Figure 6.10

(1) If $\frac{s}{v_{Veh}} - \frac{l}{v_{Ped,DSR}} > t_M$, where $v_{Ped,DSR}$ is the desired walking speed of the pedestrian, it is still possible for the pedestrian to gain advantage of journey time with a deviation angle from $\overrightarrow{PP_1}$. Assuming the maximum deviation angle is θ_{max}, the following relationship can be derived as shown in Equation 6.7.

$$\begin{cases} \frac{s - l\tan\theta_{max}}{v_{Veh}} - \frac{l}{v_{Ped,DSR}\cos\theta_{max}} = t_M \\ 0 \leq \theta_{max} < \frac{\pi}{2} \end{cases} \tag{6.7}$$

From which θ_{max} can then be solved. The decision of the pedestrian is $v_{Ped} = v_{Ped,DSR}$ and $\theta = min\{\theta_{max},\ \angle P_1PD,\ \theta_F\}$.

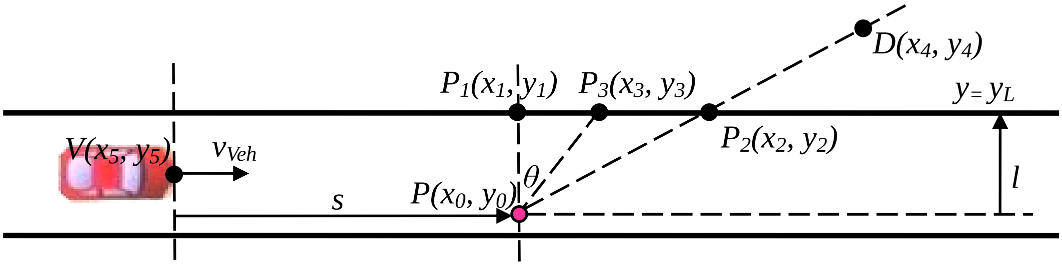
(2) If $\frac{s}{v_{Veh}} - \frac{l}{v_{Ped,DSR}} \leq t_M$, the pedestrian is unable to gain any advantage of journey time with his/her desired speed and thus has to choose P_1 as the target position to escape the lane (i.e. $\theta = 0$). In addition, there exists a minimum speed $v_{Ped,min}$ with which the condition of the pedestrian's crossing margin of t_M can just be fulfilled. This relationship can be expressed by Equation 6.8.

$$\frac{s}{v_{Veh}} - \frac{l}{v_{Ped,min}} = t_M, \quad v_{Ped,min} \geq v_{Ped,DSR} \tag{6.8}$$

Assuming a pedestrian always tries to keep a minimum deviation from its desired speed, the speed decision of the pedestrian can then be solved using Equation 6.9, which is derived from Equation 6.8.

$$v_{Ped} = v_{Ped,min} = \frac{l \cdot v_{Veh}}{s - v_{Veh} t_M} \tag{6.9}$$

2. Scenario 2: (x_4-x_0) (x_5-x_0) < 0, as shown in Figure 6.11



(x_4-x_0) (x_5-x_0) < 0

Figure 6.11 Pedestrian on-road movement (Scenario 2)

The principle to determine the pedestrian's decision is similar to the previous discussion with an exception that when $\frac{s}{v_{Veh}} - \frac{l}{v_{Ped,DSR}} > t_M$, Equation 6.7 is modified to Equation 6.10 to calculate θ_{max}.

$$\begin{cases} \frac{s+l\tan\theta_{max}}{v_{Veh}} - \frac{l}{v_{Ped,DSR}\cos\theta_{max}} = t_M \\ 0 \le \theta_{max} < \frac{\pi}{2} \end{cases} \tag{6.10}$$

3. Scenario 3: (x_4-x_0) (x_5-x_0) = 0 (i.e. $x_4 = x_0$)

In this scenario, the pedestrian has no motivation to cross diagonally. Therefore, the pedestrian's decision on direction is $\theta = 0$. His/her speed is set to $v_{Ped} = \frac{l \cdot v_{Veh}}{s - v_{Veh} t_M}$, as expressed by Equation 6.9.

It can be seen from the above discussion that the two parameters θ_F and t_M need to be calibrated according to local data. The following paragraphs will describe the two experiments conducted to calibrate these two parameters.

1. Calibration of θ_F

θ_F is the pedestrian's maximum deviation angle to the direction perpendicular to the road when there is no conflicting vehicles on the lane. The existence of θ_F represents the fact that even there is no conflicting vehicles on the lane, a pedestrian still feels the road is unsafe and cannot accept a large θ, which may keep the pedestrian on the vehicle lane for too long. The distribution of θ_F within each type of pedestrians is calibrated from field survey described as follows.

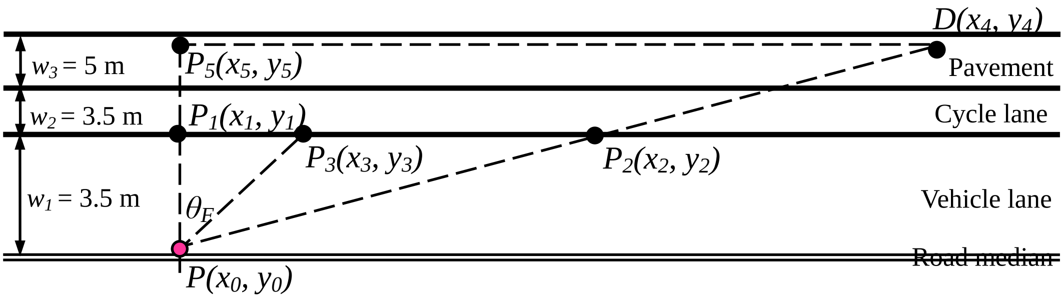


Figure 6.12 Calibration of θ_F from field survey

The data for calibration of θ_F were collected in a typical 2-way-2-lane road section in Beijing, China during off-peak time on a weekend, when the traffic was light and pedestrians were not likely to encounter conflicting vehicles when they were crossing the road. Pedestrian behaviour was recorded non-intrusively using the video camera surveying method discussed in Chapter 3. Only pedestrians fulfilling the following conditions were selected as samples for the calibration, as shown in Figure 6.12.

(1) The pedestrian just left the median of the road and stepped onto the last vehicle lane to be crossed;

(2) The destination position $D\ (x_4, y_4)$ on the pavement of this pedestrian was still far away from his/her current position, to ensure $\angle P_1PD$ was sufficiently large to cover θ_F ($|x_4 - x_5|$was set to be above 100 m, $s.t.\ \angle P_1PD = arctan\frac{|x_4-x_5|}{w_1+w_2+w_3} \geq arctan\frac{100}{3.5+3.5+5} \geq arctan\ 8.33 \approx 1.45$);

(3) There was no interference from vehicle traffic, other pedestrians or obstacles during the crossing process of this pedestrian.

As shown in Figure 6.12, assuming P_3 is the pedestrian's final departing point from the vehicle lane, as there is no vehicle interference, $\angle P_1PP_3$ is an estimation of θ_F. The distribution of θ_F for each type of pedestrians is shown in Table 6.7.

Table 6.7 The distribution of θ_F calibrated from field survey

$F(\theta)$	θ (radius)			
	YM	YF	OM	OF
0.1	0.69	0.57	0.52	0.46
0.2	0.73	0.61	0.55	0.48
0.3	0.76	0.64	0.57	0.50
0.4	0.80	0.68	0.60	0.51
0.5	0.84	0.72	0.63	0.53
0.6	0.86	0.75	0.65	0.55
0.7	0.87	0.78	0.67	0.56
0.8	0.92	0.85	0.70	0.59
0.9	1.01	0.88	0.74	0.64
1.0	1.14	1.12	0.90	0.75
Sample size	100	100	100	100

YM = Younger Male, YF = Younger Female, OM = Older Male, OF = Older Female

$F(\theta) = P(\theta_F \leq \theta)$, which is the cumulative distribution function of θ_F

2. Calibration of t_M

t_M is the minimum projected time-to-collision a pedestrian will accept when he/she just finishes the vehicle lane if the speed and direction of the pedestrian and the conflicting vehicle keep unchanged. The set-up of the experiment for calibration of t_M is illustrated in Figure 6.13.

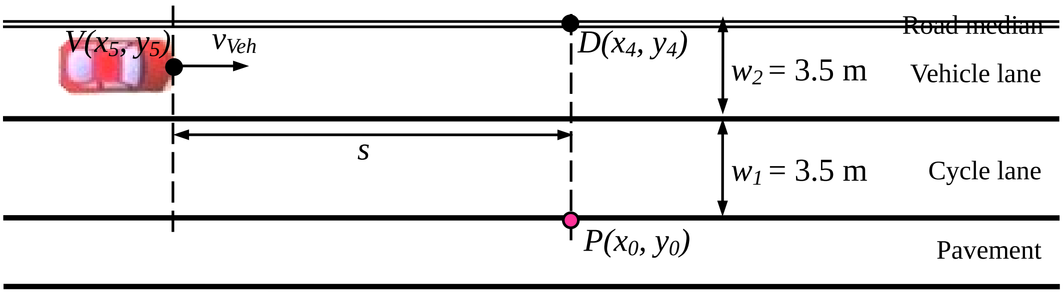


Figure 6.13 Calibration of t_M from field survey

A total of 200 people from Beijing, China were selected as subjects in this experiment. Before the on-road survey, the age and gender of each person were recorded. The desired walking speed ($v_{Ped,DSR}$) of each individual was also surveyed using the method discussed in Chapter 5. Then, a typical 2-way-2-lane street section was selected to calibrate t_M.

Each individual was assigned a crossing task from the kerb to the median of the street with a direction perpendicular to the road. For safety reasons, each subject was accompanied by the surveyor on the kerb and was instructed to report his/her decision instead of actually crossing the road, as shown in Figure 6.13. Each subject was asked to report his/her last decision to begin the crossing action when there was a vehicle approaching. The time when the subject made that decision was recorded by a stop watch carried by the surveyor whilst the scenario was recorded by a series of roof-level video cameras synchronised to the stop watch, in order to calculate the instantaneous speed (v_{Veh}) and distance (s) of the conflicting vehicle to the subject using the method discussed in Chapter 3. The value of t_M for each pedestrian in each observation was then estimated by Equation 6.11.

$$t_M = \frac{s}{v_{Veh}} - \frac{w_1 + w_2}{v_{Ped,DSR}} \tag{6.11}$$

Ten observations were conducted for each subject and the mean value of t_M was used as the crossing margin for that subject. The subjects were divided into four groups according to their gender and age. The distribution of t_M within each group of pedestrians is shown in Table 6.8.

Table 6.8 The distribution of t_M calibrated from field survey

F(t)	*t* (s)			
	YM	YF	OM	OF
0.1	1.61	1.62	2.35	2.61
0.2	1.87	1.74	2.65	2.82
0.3	2.04	1.99	2.96	3.15
0.4	2.22	2.22	3.13	3.41
0.5	2.43	2.47	3.33	3.55
0.6	2.53	2.61	3.46	3.69
0.7	2.62	2.76	3.60	3.73
0.8	2.78	2.89	3.72	3.86
0.9	2.87	3.08	3.94	4.09
1.0	3.06	3.19	4.07	4.13
Sample size	50	50	50	50

YM = Younger Male, YF = Younger Female, OM = Older Male, OF = Older Female

$F(t) = P\ (t_M \leq t)$, which is the cumulative distribution function of t_M

6.6 Pedestrian departing from vehicle lanes

This module concerns pedestrian behaviour from the time when he/she just leaves the last vehicle lane to the time when he/she just reaches his/her destination, i.e. from *P* to *D*, as shown in Figure 6.14. As the vehicle traffic no longer has influences to his/her behaviour and the interference of cyclists is not modelled in this research, the pedestrian movement behaviour in this module can be described with intra pedestrian behaviour model, which has been discussed in Chapter 5. However, when a pedestrian crosses a cycle lane, there still exists a maximum deviation angle (denoted as φ_F,) even when there is no conflicting cyclists on that cycle lane due to the pedestrian's perception of being unsafe to stay in that cycle lane for too long.

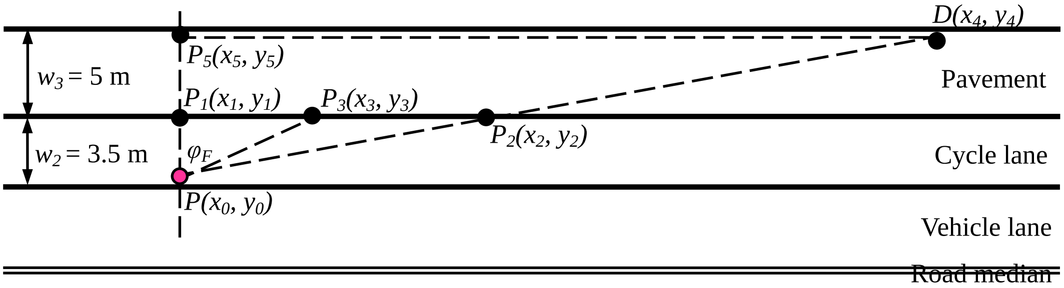


Figure 6.14 Pedestrian departing from vehicle lanes

The method for the calibration of φ_F is similar to the one used to calibrate θ_F, which has been discussed in the previous section. A field survey was carried out in a typical 2-way-2-lane road section in Beijing, China. Pedestrians fulfilling the following conditions were selected as samples for the calibration, as shown in Figure 6.14.

(1) The pedestrian just left the last vehicle lane and stepped onto the subsequent cycle lane;

(2) The destination position $D\ (x_4, y_4)$ on the pavement of this pedestrian was still far away from his/her current position, to ensure $\angle P_1PD$ was sufficiently large to cover φ_F ($|x_4 - x_5|$was set to be above 100 m, $s.t.\ \angle P_1PD = arctan\frac{|x_4-x_5|}{w_2+w_3} \geq arctan\frac{100}{3.5+5} \geq arctan\ 11.76 \approx 1.49$);

(3) There was no interference from cyclists, other pedestrians or obstacles when the pedestrian was crossing that cycle lane.

As shown in Figure 6.14, assuming P_3 is the pedestrian's final departing position from the cycle lane, as there is no interference from other objects, $\angle P_1PP_3$ can be used as an estimation of φ_F. The distribution of φ_F for each type of pedestrians is shown in Table 6.9.

Table 6.9 The distribution of φ_F calibrated from field survey

F(φ)	φ (radius)			
	YM	YF	OM	OF
0.1	0.81	0.70	0.59	0.50
0.2	0.82	0.75	0.64	0.55
0.3	0.90	0.77	0.65	0.57
0.4	0.95	0.80	0.71	0.60
0.5	0.97	0.82	0.72	0.62
0.6	0.99	0.83	0.75	0.64
0.7	1.04	0.88	0.76	0.68
0.8	1.09	0.92	0.81	0.69
0.9	1.14	1.02	0.87	0.77
1.0	1.33	1.21	1.06	0.89
Sample size	100	100	100	100

YM = Younger Male, YF = Younger Female, OM = Older Male, OF = Older Female

$F(\varphi) = P\ (\varphi_F \leq \varphi)$, which is the cumulative distribution function of φ_F

6.7 Vehicle reactions to pedestrians

This module describes how the dynamics of a vehicle can be influenced by a conflicting pedestrian ahead of it on the vehicle lane in an unsignalised situation. As it was extremely difficult to collect data on the microscopic decisions of individual drivers' reactions to pedestrians with current equipment, an assumption analogous to the concept of collision avoidance in the car-following model was used to describe any vehicle's reaction in a conflicting situation.

For any vehicle at its reaction time, there exists a maximum acceleration a_{max} that can be applies due to the conflict with the pedestrian ahead, under which the vehicle is still able to achieve a full stop with a comfortable (free flow) deceleration rate, assuming the pedestrian suddenly stops on the vehicle lane. This assumption is reasonable in that if the conflicting pedestrian is far away from the vehicle, a_{max} is likely to be large and therefore cannot limit the acceleration calculated from the intra vehicle model, representing the vehicle is not influenced by the pedestrian crossing ahead of it. On the contrary, if the conflicting pedestrian is near the vehicle, a_{max} can be small than the acceleration calculated from the intra vehicle model and in this case the vehicle has to consider a_{max} as a constraint in order to avoid a potential collision, representing the vehicle is forced to deviate from its desired decision by the crossing pedestrian ahead of it. The calculation of a_{max} is described as follows.

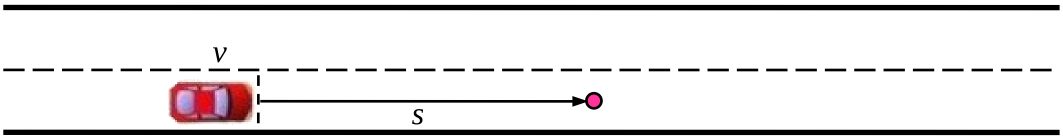


Figure 6.15 Vehicle reaction to pedestrian

As shown in Figure 6.15, assuming the distance between the vehicle and the pedestrian is s, the vehicle's instantaneous speed is v, comfortable (free flow) deceleration rate is d_{FRF} ($d_{FRF} > 0$), maximum allowed acceleration is a_{max} and reaction time is τ. The above discussion can be express with Equation 6.12.

$$v\tau + \frac{1}{2}a_{max}\tau^2 + \frac{(v+a_{max}\tau)^2}{2d_{FRF}} = s \tag{6.12}$$

Equation 6.12 can then be transformed into Equation 6.13.

$$\frac{\tau^2}{2d_{FRF}}a_{max}^2 + \left[\frac{\tau^2}{2} + \frac{v\tau}{d_{FRF}}\right]a_{max} + v\tau + \frac{v^2}{2d_{FRF}} - s = 0 \tag{6.13}$$

It can be noted that:

$$v\tau + \frac{v^2}{2d_{FRF}} - s < 0 \Rightarrow a_{max} > 0 \tag{6.14}$$

$$v\tau + \frac{v^2}{2d_{FRF}} - s = 0 \Rightarrow a_{max} = 0 \tag{6.15}$$

$$v\tau + \frac{v^2}{2d_{FRF}} - s > 0 \Rightarrow a_{max} < 0 \tag{6.16}$$

Therefore, a_{max} can be calculated by Equation 6.17, which is derived from Equation 6.13 to 6.16.

$$a_{max} = \frac{-\left(\frac{\tau^2}{2}+\frac{v\tau}{d_{FRF}}\right)+\sqrt{\left(\frac{\tau^2}{2}+\frac{v\tau}{d_{FRF}}\right)^2-4\cdot\frac{\tau^2}{2d_{FRF}}\cdot\left(v\tau+\frac{v^2}{2d_{FRF}}-s\right)}}{2\left(\frac{\tau^2}{2d_{FRF}}\right)} \quad (6.17)$$

Any vehicle's acceleration calculated from the intra vehicle models has to be filtered by a_{max} expressed by Equation 6.17 in order that there is no collision between the vehicle and any pedestrian ahead of it.

To conclude, the five sub-models described in this chapter, together with the intra vehicle and pedestrian models discussed in the previous two chapters, were integrated and implemented with C++ programming language by the author to form a complete micro-simulation model, PVISIM (Pedestrian-Vehicle Interaction SIMulation). The test of the validity of the complete model is to be presented in the next chapter.

6.8 Conclusion

This chapter described the development, calibration and validation of the pedestrian-vehicle interaction behaviour used in this research. The pedestrian-vehicle interaction process was abstracted into the following five modules: pedestrian gap acceptance, pedestrian approaching vehicle lanes, pedestrian on-road movement, pedestrian departing from vehicle lanes and vehicle reactions to pedestrians.

For pedestrian gap acceptance, behaviour at locations with no-control and with a Zebra crossing were studied. For a typical location with no-control, the pedestrian's probability of accepting a gap in the traffic was found to be able to be predicted with a binary logit model. Three factors including the nearside traffic time gap, pedestrian's age type and the number of people in a crossing group were identified to be sufficiently significant to be included in such a model. For a typical location with a Zebra crossing, similar result was achieved but with slightly different model parameters, showing that the probability of accepting smaller gaps was slightly higher.

Pedestrian approaching vehicle lanes, on-road movement and departing from vehicle lanes concerned pedestrian's path finding and speed choice. Path finding behaviour related to the location where a pedestrian decided to cross the road and whether the pedestrian would use a nearby pedestrian crossing if available. The behaviour was described based on the assumption that pedestrians were likely to organise their crossing location to minimise their delay and whilst also considering safety.

It was extremely difficult to collect data on the microscopic decisions of individual drivers' reactions to pedestrians with current equipment. To describe this process, an assumption analogous to the concept of collision avoidance in the car-following model was used, which was that for any vehicle at its reaction time, there existed a maximum acceleration it could apply due to the conflict with the pedestrian ahead, under which the vehicle was still able to come to a full stop with a comfortable deceleration rate, assuming the pedestrian suddenly stopped on that vehicle lane. These assumptions will be tested in the validation process for the compete model in the next chapter.

These five modules during the interaction process, together with the intra vehicle model and intra pedestrian model discussed in Chapter 4 and Chapter 5 form the core algorithms of the complete pedestrian-vehicle interaction model. The flow chart in Figure 6.16 illustrates the top-level logic of the complete simulation model. The complete model was implemented with C++, a highly object oriented computer programming language. Part of the source codes are attached in Appendix II of this dissertation.

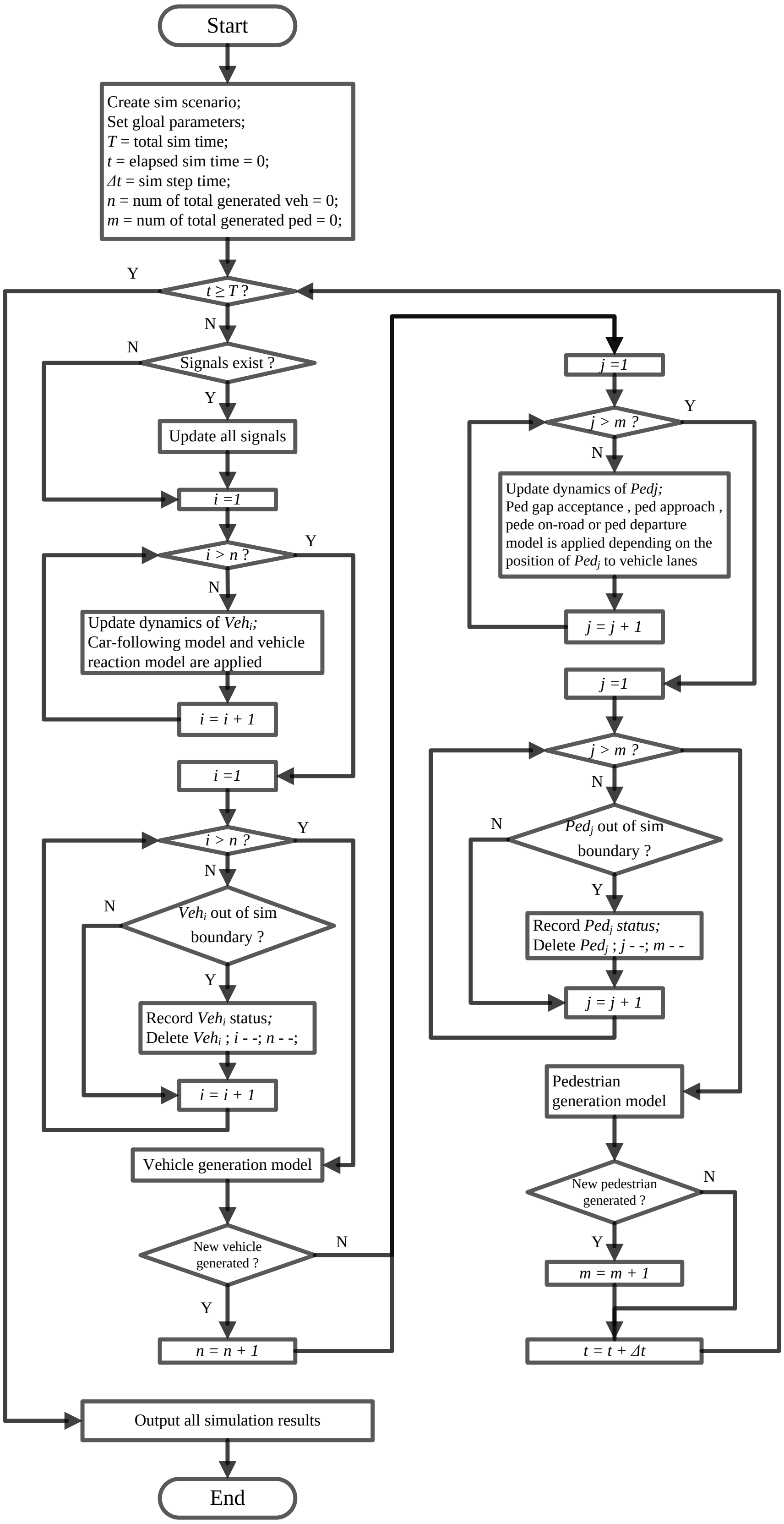


Figure 6.16 The top-level flow chart of the complete microscopic pedestrian-vehicle interaction simulation model

CHAPTER 7
VALIDATION OF THE COMPLETE MODEL

7.1 Introduction

It can be seen that the development of the complete model described in previous chapters involves both behaviour interpretation from the new data and assumptions made based on previous research and qualitative observations. To ensure the credibility of the model based on those interpretation and assumptions, it is necessary to conduct model validation. Model validation is a process that tests the accuracy of the proposed model by comparing relative traffic flow data generated by the model with those collected from field survey (Park and Schneeberger 2003). For a long time, the validation of traffic simulation model is often discussed and practised informally among researchers (Sack et al 2001), until Park and Schneeberger (2003) developed a rather standard procedure and demonstrated it through a case study using VISSIM. This validation procedure can be summarised into the following 4 steps:

1. Determination of measures of effectiveness: the first step is to determine which indicators to be used as measures of effectiveness. It should be noted that it is not practical or necessary to demand a simulation model acts exactly the same in every aspects like the actual system. The goal of model validation is to make the model useful in the sense that the model can address the concerned problem and provide accurate information about the system being modelled. In most research, as the average delays of road users are of particular interests in traffic engineering, it is expected that given the same inputs, the simulation system can generate realistic delay data for both modes compared to the actual system. Therefore, ideally the delay data generated from the simulation system should be tested against the data collected from the field survey. However, it is often not practical to obtain such data in the real world and some researchers usually adopt alternative methods. Park and Schneeberger (2003) pointed that journey time was found to be related to delay and

was usually selected as an alternative for model validation in practice. In their demonstration, the average vehicle travel time between two data collection points in a network was chosen as the performance measure.

<u>2. Data collection from field survey:</u> once the measures of effectiveness have been identified, the next step is to collect necessary data from the field during a particular time period on a typical day. These data include output measures of performance for model validation and also scenario parameter data as simulation inputs, such as road geometry, traffic counts, signal timing and etc. The collection of vehicle or pedestrian travel times between two designated points can be achieved by the video camera data collection method discussed in Chapter 3.

<u>3. Simulation run:</u> next, the surveyed scenario characteristic data during that particular time period on the data collection day should be modelled into the simulation system, which then runs for a period of time to collect a series of journal times from the computer.

<u>4. Comparison of results from field survey and simulation:</u> in this step, field individual travel times will be compared with individual travel times from the simulation model output. Park and Schneeberger (2003) suggested using the t test to compare the means of the two data sets, or more rigorously using the Kolmogorov-Smirnov test to compare the distributions of the two data sets. If there is not enough evidence to reject that the distributions of the two data sets are the same, the model can be regarded accurate in a statistical meaning. Otherwise, the model developer should examine and recalibrate the behavioural parameters inside the simulation model and repeat the validation process until the model is accurate.

In microscopic simulation modelling, this method has been widely accepted and adopted by many researchers. Typical use cases can be found from research work by Li et al (2010), Ishaque and Norland (2009), Du (2008), Min et al (2008) and etc. In this research, the model validation was conducted based on the above method. Both the vehicle journey time and pedestrian journey time between two designated data collection points during a particular time period on a typical day were examined for the validation of the complete model. In addition, to ensure the robustness of the model validation, sensitivity and error analysis were also conducted to test whether the model can behave similarly to the actual system on different days.

Given a road section as shown in Figure 7.1, the vehicle journey time and pedestrian journey time for model validation can be defined as follows.

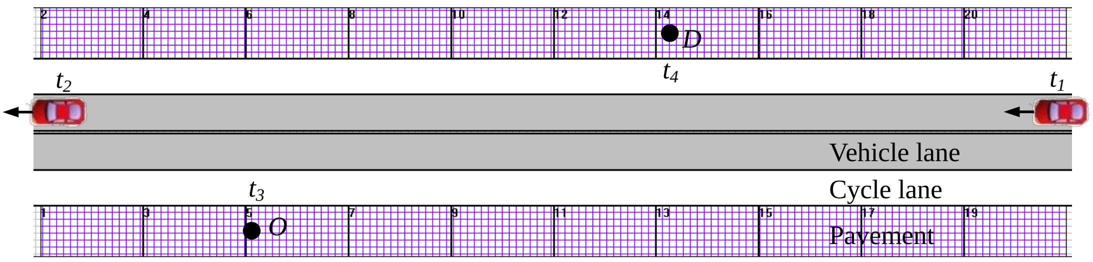


Figure 7.1 Definition of vehicle and pedestrian journey times for model validation

The journey time of any vehicle in a road section (both lanes) is defined as the time difference between the time when the vehicle just leaves the area and the time when the vehicle just enters the area, i.e. t_2-t_1. The journey time of any pedestrian is defined as the time difference between the time the pedestrian just reaches its crossing destination *D* and the time the pedestrian just emerges at its crossing origin *O*, i.e. t_4-t_3. In reality, the journey time of any vehicle or pedestrian can be surveyed using the video camera method discussed in Chapter 3.

As the model is required to function valid in both unsignalised and signalised scenarios, three typical types of scenarios (locations with no-control, a Zebra crossing and a fixed-time signalised crossing, respectively) were selected for model validation. For each scenario, data collected for model validation were independent to those used for model development, either at different locations or the same location but on different days. First, data collection was conducted during a particular time on a typical weekday to check if the distributions of actual and simulated vehicle/pedestrian journey times were the same given the same system inputs. The Kolmogorov-Smirnov (K-S) test was applied for this purpose. Then, data were collected on another 13 days including both weekdays and weekends. The average vehicle/pedestrian journey time were calculated from field survey and simulation for each of all the 14 days. These two series of data were used for sensitivity analysis and error checking to examine if the behaviour (in terms of average vehicle and pedestrian journey times) of the simulation system was still similar to the actual system when inputs changed. For this purpose, the correlation analysis was conducted in SPSS using Pearson's method and the Root Mean Square Percent Error (RMSPE) was calculated using the definition in Equation 7.1.

$$RMSPE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i - y_i}{y_i}\right)^2} \quad (7.1)$$

Where,

i is the ID number of the i^{th} case;

x_i is the simulated measurement value of the i^{th} case;

y_i is the actual measurement value of the i^{th} case;

n is the total number of pairs of cases;

RMSPE is the root mean square percent error. For validation using journey times, Wisconsin DOT (2002) suggested that RMSPE within 15% was usually acceptable.

The validation process for each scenario is detailed in the following sections.

7.2 Scenario 1: a typical location with no-control

7.2.1 Data collection

The layout and description of the site chosen for the field survey is illustrated in Figure 7.2 and Table 7.1.

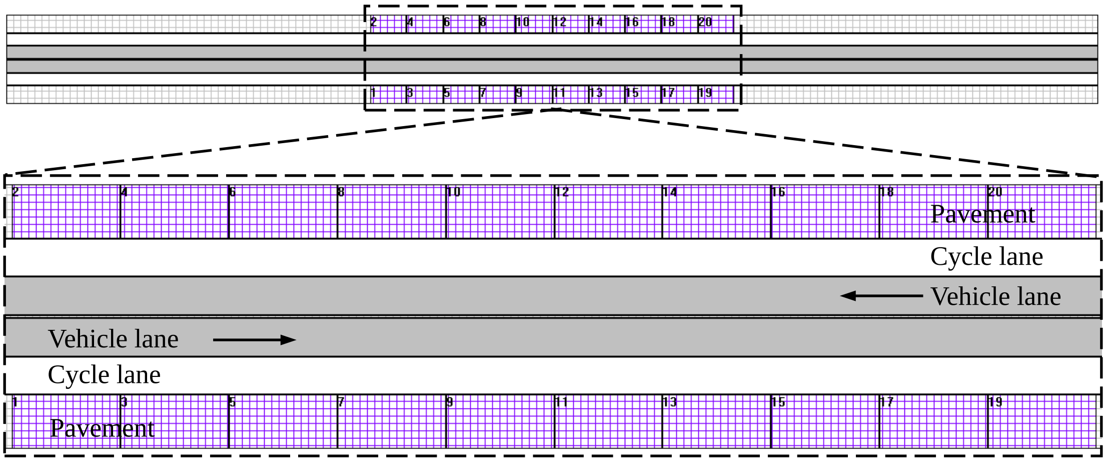


Figure 7.2 Layout of the site for model validation in a typical no-control scenario

Table 7.1 Description of the site for model validation in a typical no-control scenario

Location	Jiaoda East Road, Beijing, China
Time	12:00 to 13:00
Date	15 September 2008
Length of road section	300 m
Width of lanes	vehicle lane = 3.5 m, cycle lane = 3.5 m, pavement = 5.0 m
Width of road median	0.3 m
Speed limit	30 km/h
Type of pedestrian crossing	no-control
Vehicle composition	79% LV, 16% MCV, 1% HCV, 2% BCR and 2% BCA
Vehicle desired speed mean	8.97 m/s
Pedestrian composition	39% YM, 36% YF, 15% OM, and 10% OF

Individual journey times of vehicles passing this section and pedestrians crossing in this area were surveyed using the video camera method discussed in Chapter 3. The distributions of individual vehicle and pedestrian journey times from field survey are illustrated in Figure 7.3 and Figure 7.4 respectively.

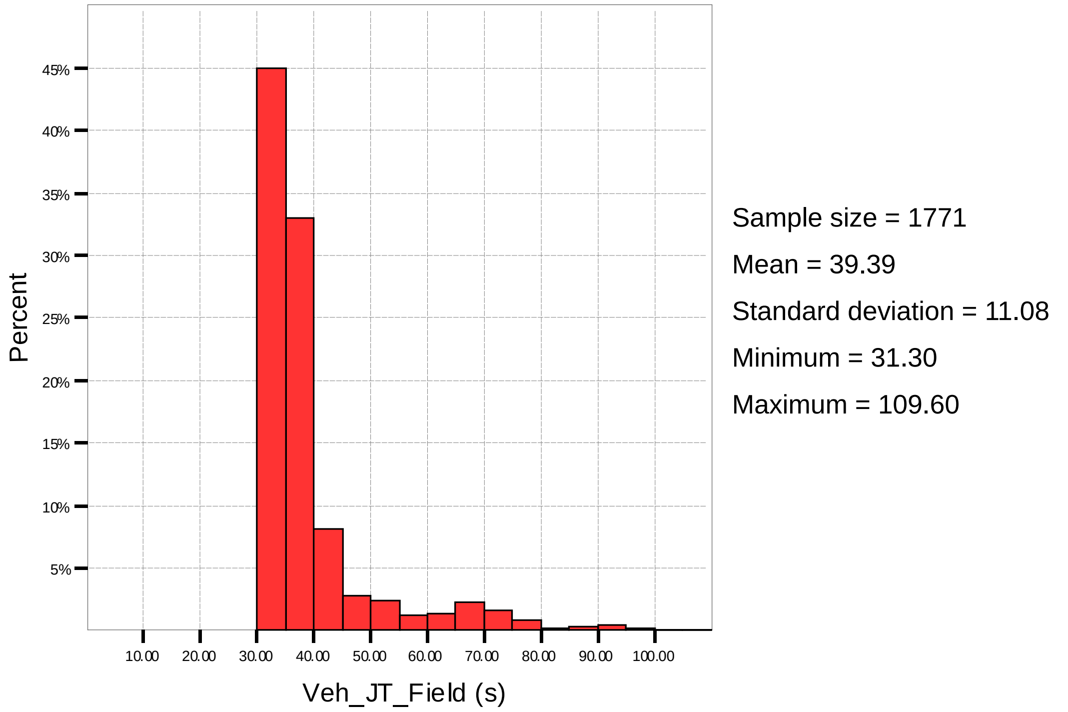


Figure 7.3 The distribution of vehicle journey times from field survey

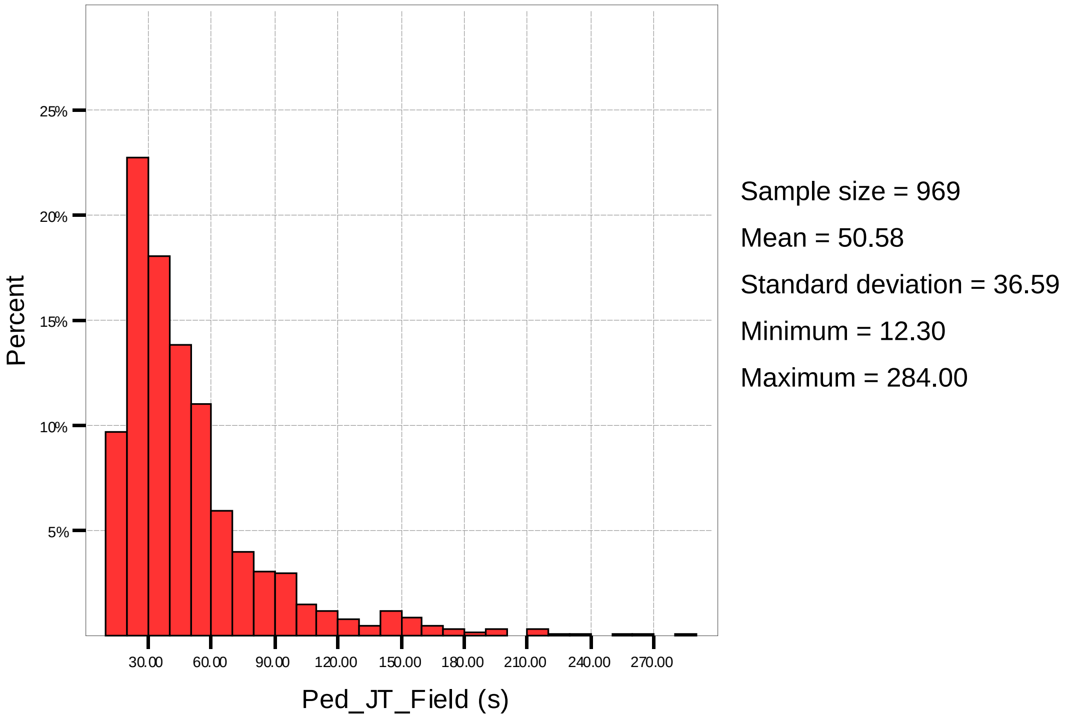


Figure 7.4 The distribution of pedestrian journey times from field survey

The vehicle and pedestrian crossing demand data during the data collection time period on that day were also surveyed, as shown in Table 7.2. These demand data, as well as the site characteristics as shown in Table 7.1, were used as inputs to conduct simulation. The simulation time was set longer than the actual surveying time in order to achieve a larger sample size. The distributions of individual vehicle and pedestrian journey times from simulation are illustrated in Figure 7.5 and Figure 7.6 respectively.

Table 7.2 The vehicle and pedestrian crossing demand during surveying time period on one data collection day at a location with no-control

Pedestrian crossing demand (*O-D*, ped/h)								Vehicle traffic demand (veh/h)	
9-10	9-12	11-10	11-12	10-9	10-11	12-9	12-11	W-E	E-W
102	96	149	107	121	123	130	141	884	887

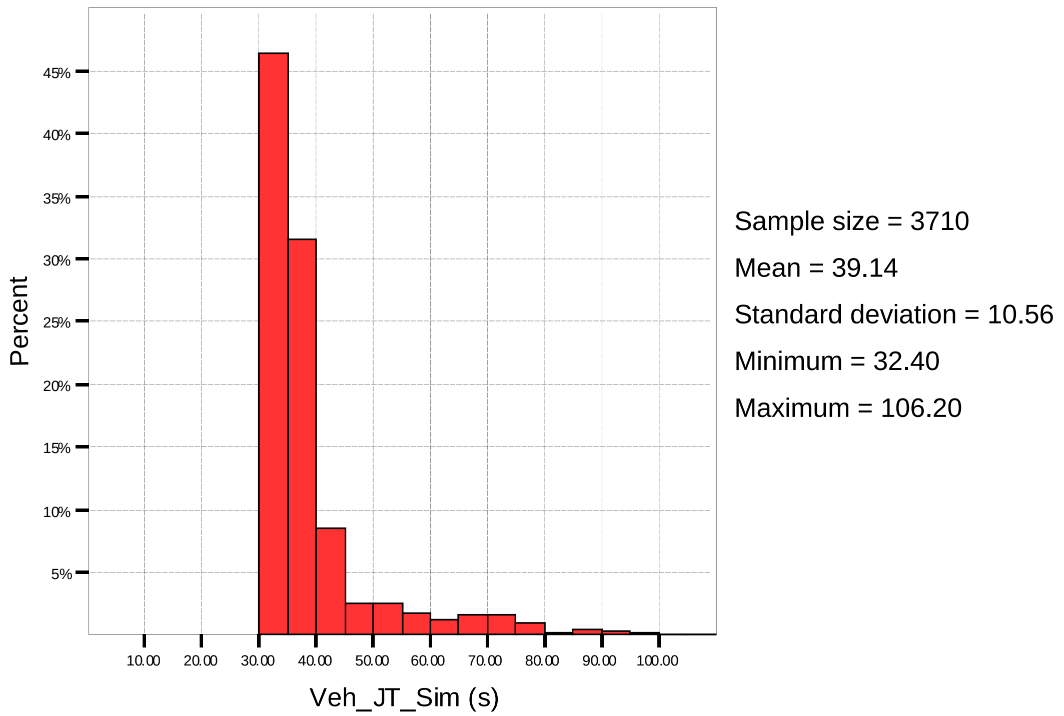


Figure 7.5 The distribution of vehicle journey times from simulation

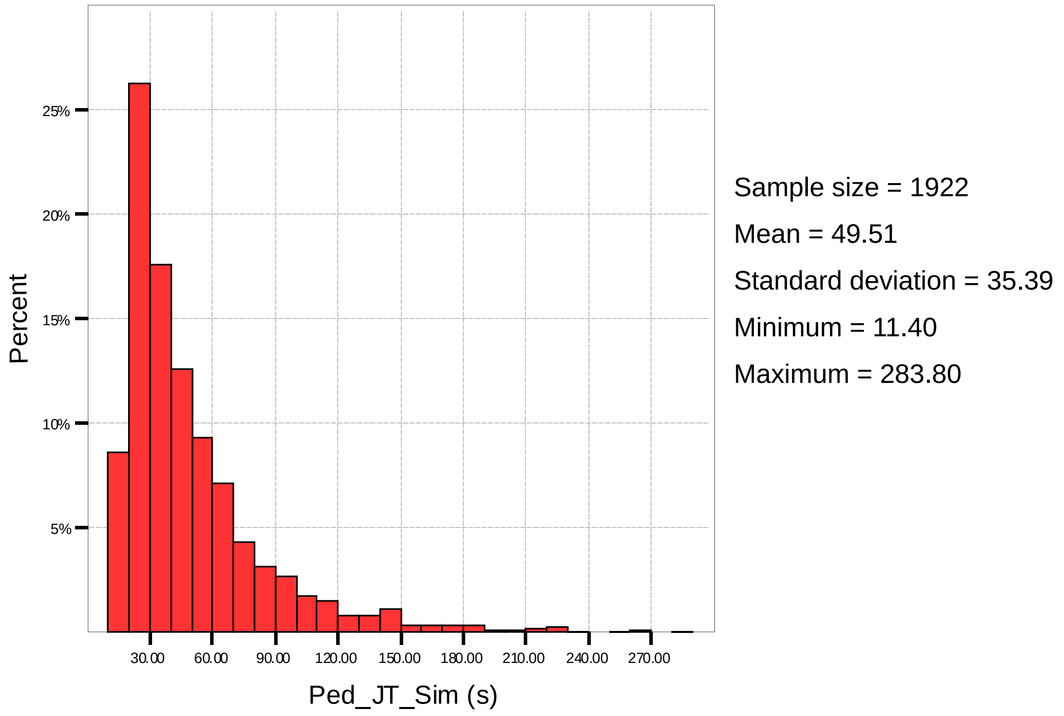


Figure 7.6 The distribution of vehicle journey times from simulation

7.2.2 Comparison of vehicle journey times

It can be seen from Figure 7.3 and Figure 7.5 that the distributions of actual and simulated vehicle journey times appear to be close. The following 2-sample K-S test was conducted to check the similarity of the distributions from the two samples.

Test method: 2-sample K-S test.

Test hypotheses: H_0: the distributions of the vehicle journey times from field survey and simulation are the same; H_1: the distributions of the vehicle journey times from field survey and simulation are not the same;

The result of the test is shown in Table 7.3, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.510). Therefore, it is reasonable to accept H_0. The distributions of vehicle journey times from field survey and simulation have no difference in a statistical meaning.

Table 7.3 The result of 2-sample K-S test for vehicle journey times in a typical no-control scenario

Frequencies

	Type	N
Veh_JT	Field	1771
	Sim	3710
	Total	5481

Veh_JT = Vehicle Journey Time, N = Sample Size

Test Statistics[a]

		Veh_JT
Most Extreme Differences	Absolute	.024
	Positive	.024
	Negative	-.006
Kolmogorov-Smirnov Z		.821
Asymp. Sig. (2-tailed)		.510

a. Grouping Variable: Type

7.2.3 Comparison of pedestrian journey times

Similarly, Figure 7.4 and Figure 7.6 show that the distributions of actual and simulated pedestrian journey times appear to be close. Therefore, the same procedure was used to examine the pedestrian journey times. The following 2-sample K-S test was conducted to check the similarity of the distributions from the two samples.

Test method: 2-sample K-S test.

Test hypotheses: H_0: the distributions of the pedestrian journey times from field survey and simulation are the same; H_1: the distributions of the pedestrian journey times from field survey and simulation are not the same;

The result of the test is shown in Table 7.4, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.280). Therefore, it is reasonable to accept H_0. The distributions of pedestrian journey times from field survey and simulation have no difference in a statistical meaning.

Table 7.4 The result of 2-sample K-S test for pedestrian journey times in a typical no-control scenario

Frequencies

	Type	N
Ped_JT	Field	969
	Sim	1922
	Total	2891

Ped_JT = Pedestrian Journey Time, N = Sample Size

Test Statistics[a]

		Ped_JT
Most Extreme Differences	Absolute	.039
	Positive	.039
	Negative	-.015
Kolmogorov-Smirnov Z		.991
Asymp. Sig. (2-tailed)		.280

a. Grouping Variable: Type

7.2.4 Sensitivity analysis and error checking

The above discussion shows that given the same system inputs at a particular level, the simulation can generate similar results as the actual system in terms of vehicle and pedestrian journey times. Therefore, it is reasonable to use the average vehicle and pedestrian journey time from the simulation to reflect the efficiency of the actual system. In order to ensure this idea is robust, this section conducts the sensitivity and error analysis to check if the model can still behave according to the actual system when levels of key inputs, including vehicle and pedestrian crossing demands, change significantly.

For this purpose, the data collection process described in Section 7.2.1 was repeated on the same site during the same time-of-day but on another 13 days, including 9 weekdays and 4 weekends from 16 to 28 in September 2008.The average vehicle and pedestrian journey times were calculated from field survey and simulation for each of all the 14 surveying days. The levels of vehicle and pedestrian crossing demand for each day, as well as the resulting average vehicle and pedestrian journey times from field survey and simulation are shown in Table 7.5.

Table 7.5 Data collection on multiple days at a location with no-control

Day	Field survey												Simulation	
	Pedestrian crossing demand (*O-D*, ped/h)								Vehicle traffic demand (veh/h)		Average journey time (s)		Average journey time (s)	
	9-10	9-12	11-10	11-12	10-9	10-11	12-9	12-11	W-E	E-W	Veh.	Ped.	Veh.	Ped.
1	102	96	149	107	121	123	130	141	884	887	39.39	50.58	39.14	49.51
2	101	99	136	111	124	113	123	150	872	849	36.52	48.66	36.95	47.55
3	107	100	128	103	114	129	125	131	874	835	36.95	49.54	37.51	48.42
4	122	91	131	107	123	124	142	141	868	808	38.6	48.75	37.60	48.55
5	123	93	148	109	111	112	134	135	846	858	36.4	42.82	37.14	44.39
6	54	47	68	53	57	62	71	66	452	444	33.92	22.94	33.73	22.79
7	48	42	61	47	51	56	64	59	426	417	33.15	22.14	33.54	22.28
8	108	87	148	105	125	126	134	140	875	834	37.05	46.47	37.40	46.21
9	102	85	125	113	135	129	139	147	899	815	40.62	49.97	37.85	49.54
10	105	95	152	102	127	129	124	145	877	810	38.37	46.63	38.24	46.44
11	106	87	141	110	116	114	128	134	840	835	36.54	42.07	38.10	44.04
12	102	95	134	106	119	129	119	136	846	848	36.84	44.3	38.02	43.98
13	61	46	74	54	56	56	67	68	433	429	33.88	22.13	33.61	22.19
14	55	42	66	49	50	51	60	61	398	403	33.04	21.97	33.47	22.36

Day 1-5 and 8-12 are weekdays; Day 6-7 and 13-14 are weekends

For average vehicle journey times, the correlation between values from simulation and field survey computed in SPSS is illustrated in Table 7.6, which shows that these two series of data have a strong correlation. The RMSPE between the simulated and actual values, calculated by Equation 7.1, is 2.60%, which is acceptable according to the criteria suggested by Wisconsin DOT (2002).

Table 7.6 Correlations for average vehicle journey times from field survey and simulation on multiple days in a no-control scenario

Correlations

		Avg_Veh_JT_Field	Avg_Veh_JT_Sim
Avg_Veh_JT_Field	Pearson Correlation	1	.895**
	Sig. (2-tailed)		.000
	N	14	14
Avg_Veh_JT_Sim	Pearson Correlation	.895**	1
	Sig. (2-tailed)	.000	
	N	14	14

**. Correlation is significant at the 0.01 level (2-tailed).
Avg_Veh_JT_Field = Average Vehicle Journey Time from Field Survey
Avg_Veh_JT_Sim = Average Vehicle Journey Time from Simulation

For average pedestrian journey times, the correlation between values from simulation and field survey computed in SPSS is illustrated in Table 7.7, which shows that these two series of data have a strong correlation. The RMSPE between the simulated and actual values, calculated by Equation 7.1, is 2.00%, which is acceptable according to the criteria suggested by Wisconsin DOT (2002).

Table 7.7 Correlations for average pedestrian journey times from field survey and simulation on multiple days in a no-control scenario

Correlations

		Avg_Ped_JT_Field	Avg_Ped_JT_Sim
Avg_Ped_JT_Field	Pearson Correlation	1	.997**
	Sig. (2-tailed)		.000
	N	14	14
Avg_Ped_JT_Sim	Pearson Correlation	.997**	1
	Sig. (2-tailed)	.000	
	N	14	14

**. Correlation is significant at the 0.01 level (2-tailed).
Avg_Ped_JT_Field = Average Pedestrian Journey Time from Field Survey
Avg_Ped_JT_Sim = Average Pedestrian Journey Time from Simulation

In conclusion, the behaviour of the model is reasonable and similar to the actual system at different levels of vehicle and pedestrian crossing demands and the errors are acceptable. The complete model is sufficiently reliable to be used to conduct simulation study for this scenario.

7.3 Scenario 2: a typical location with a Zebra crossing

7.3.1 Data collection

The layout and description of the site chosen for the field survey is illustrated in Figure 7.7 and Table 7.8.

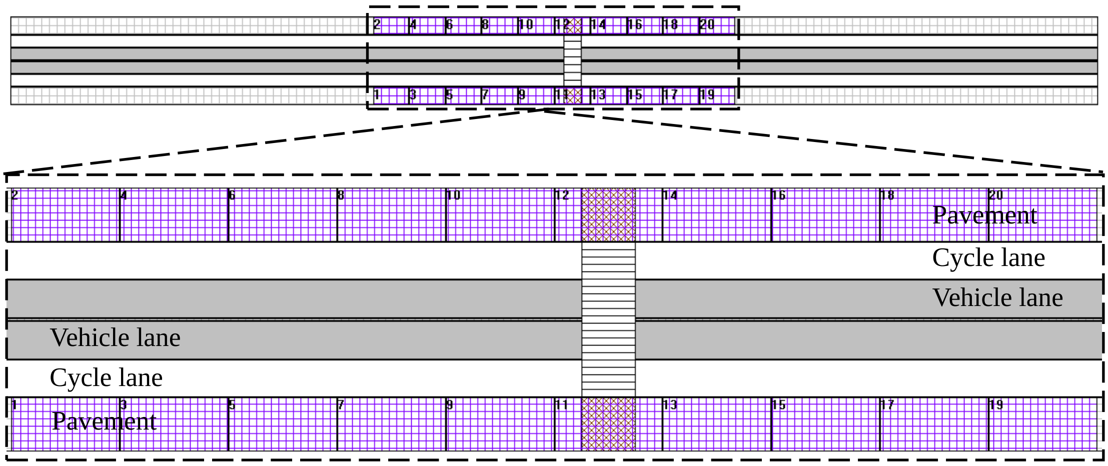
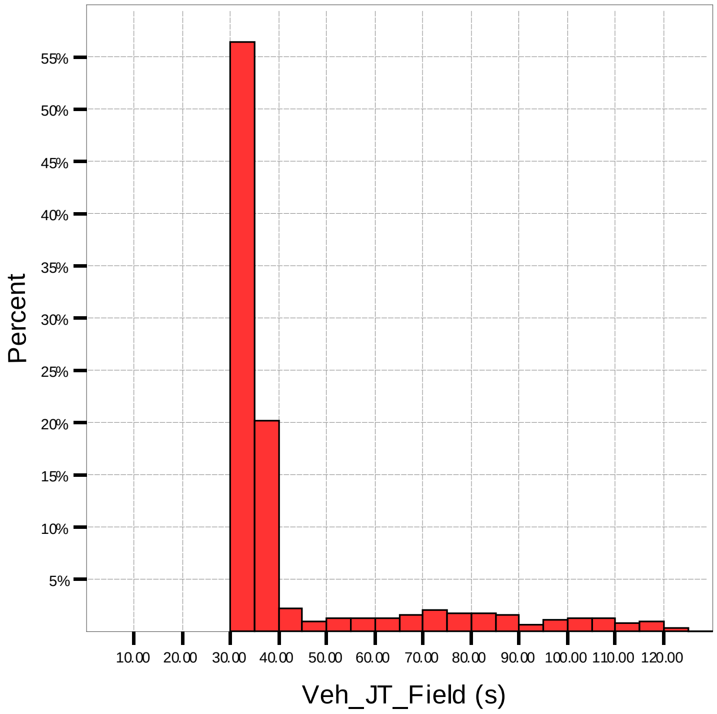


Figure 7.7 Layout of the site for model validation in a Zebra crossing scenario

Table 7.8 Description of the site for model validation in a Zebra crossing scenario

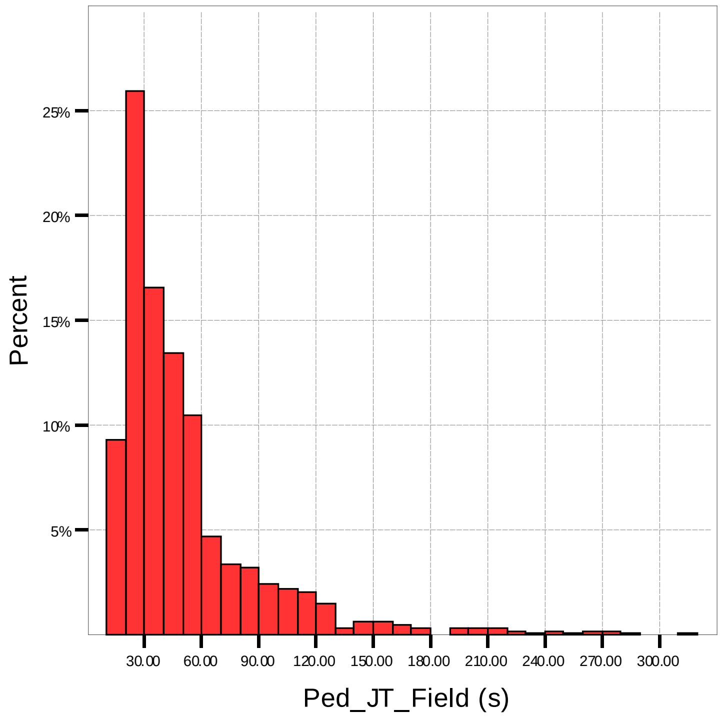
Location	Jianhua South Road, Greater Beijing Area, China
Time	16:30 to 17:30
Date	27 October 2008
Length of road section	300 m
Width of lanes	vehicle lane = 3.5 m, cycle lane = 3.5 m, pavement = 5.0 m
Width of road median	0.3 m
Speed limit	30 km/h
Type of pedestrian crossing	one Zebra crossing in the middle of the road section
Vehicle composition	78% LV, 17% MCV, 1% HCV, 4% BCR and 0% BCA
Vehicle desired speed mean	9.15 m/s
Pedestrian composition	41% YM, 38% YF, 11% OM, and 10% OF

Individual journey times of vehicles passing this section and pedestrians crossing in this area were surveyed using the video camera method discussed in Chapter 3. The distributions of individual vehicle and pedestrian journey times from field survey are illustrated in Figure 7.8 and Figure 7.9 respectively.



Sample size = 1577
Mean = 44.06
Standard deviation = 21.93
Minimum = 30.10
Maximum = 129.80

Figure 7.8 The distribution of vehicle journey times from field survey



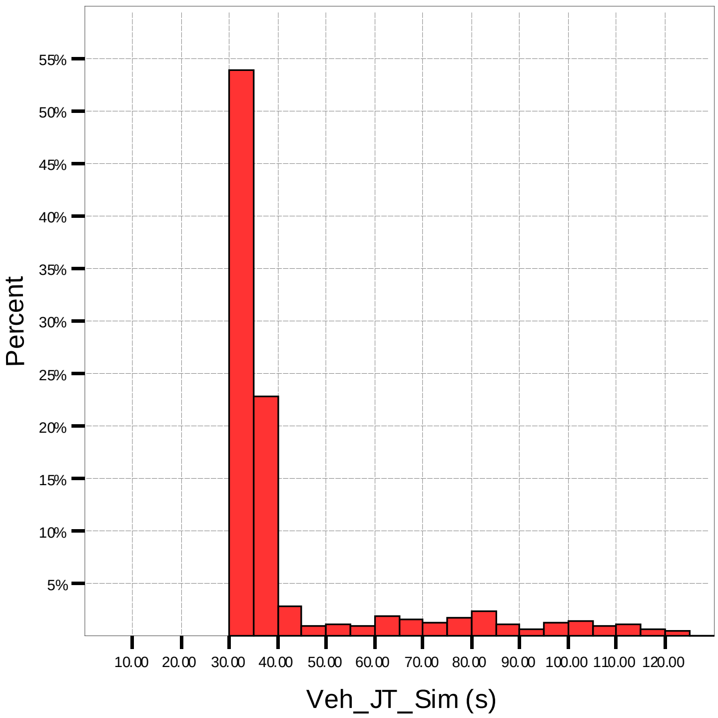
Sample size = 936
Mean = 51.89
Standard deviation = 41.69
Minimum = 12.80
Maximum = 314.70

Figure 7.9 The distribution of pedestrian journey times from field survey

The vehicle and pedestrian crossing demand data during the data collection time period on that day were also surveyed, as shown in Table 7.9. These demand data, as well as the site characteristics as shown in Table 7.8, were used as inputs to conduct simulation. The simulation time was set longer than the actual surveying time in order to achieve a larger sample size. The distributions of individual vehicle and pedestrian journey times from simulation are illustrated in Figure 7.10 and Figure 7.11 respectively.

Table 7.9 The vehicle and pedestrian crossing demand during surveying time period on one data collection day at a location with a Zebra crossing

Pedestrian crossing demand (O-D, ped/h)								Vehicle traffic demand (veh/h)	
11-10	11-12	13-10	13-12	10-11	10-13	12-11	12-13	W-E	E-W
65	243	97	103	90	96	192	50	806	771



Sample size = 3185
Mean = 43.80
Standard deviation = 21.56
Minimum = 31.10
Maximum = 127.30

Figure 7.10 The distribution of vehicle journey times from simulation

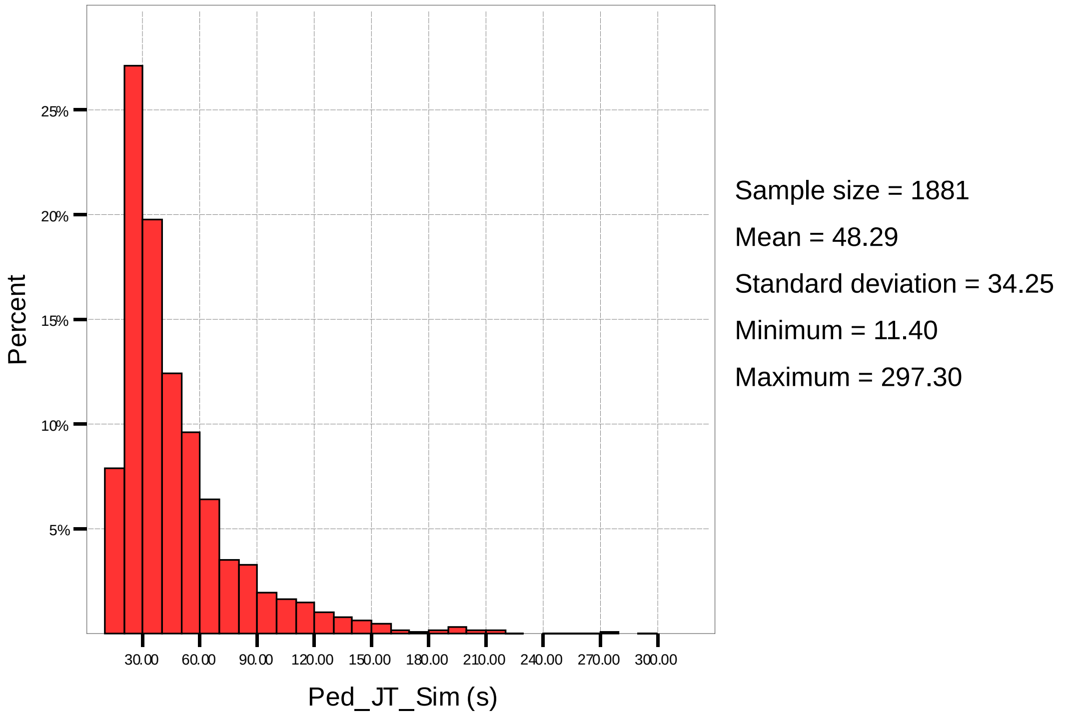


Figure 7.11 The distribution of pedestrian journey times from simulation

7.3.2 Comparison of vehicle journey times

It can be seen from Figure 7.8 and Figure 7.10 that the distributions of actual and simulated vehicle journey times appear to be close. The following 2-sample K-S test was conducted to check the similarity of the distributions from the two samples.

Test method: 2-sample K-S test.

Test hypotheses: H_0: the distributions of the vehicle journey times from field survey and simulation are the same; H_1: the distributions of the vehicle journey times from field survey and simulation are not the same;

The result of the test is shown in Table 7.10, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.478). Therefore, it is reasonable to accept H_0. The distributions of vehicle journey times from field survey and simulation have no difference in a statistical meaning.

Table 7.10 The result of 2-sample K-S test for vehicle journey times in a typical Zebra crossing scenario

Frequencies

	Type	N
Veh_JT	Field	1577
	Sim	3185
	Total	4762

Veh_JT = Vehicle Journey Time, N = Sample Size

Test Statistics[a]

		Veh_JT
Most Extreme Differences	Absolute	.026
	Positive	.012
	Negative	-.026
Kolmogorov-Smirnov Z		.841
Asymp. Sig. (2-tailed)		.478

a. Grouping Variable: Type

7.3.3 Comparison of pedestrian journey times

Similarly, Figure 7.9 and Figure 7.11 show that the distributions of actual and simulated pedestrian journey times appear to be close. Therefore, the same procedure was used to examine the pedestrian journey times. The following 2-sample K-S test was conducted to check the similarity of the distributions from the two samples.

Test method: 2-sample K-S test.

Test hypotheses: H_0: the distributions of the pedestrian journey times from field survey and simulation are the same; H_1: the distributions of the pedestrian journey times from field survey and simulation are not the same;

The result of the test is shown in Table 7.11, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.418). Therefore, it is reasonable to accept H_0. The distributions of pedestrian journey times from field survey and simulation have no difference in a statistical meaning.

Table 7.11 The result of 2-sample K-S test for pedestrian journey times in a typical Zebra crossing scenario

Frequencies

	Type	N
Ped_JT	Field	936
	Sim	1881
	Total	2817

Ped_JT = Pedestrian Journey Time, N = Sample Size

Test Statistics[a]

		Ped_JT
Most Extreme Differences	Absolute	.035
	Positive	.035
	Negative	-.026
Kolmogorov-Smirnov Z		.882
Asymp. Sig. (2-tailed)		.418

a. Grouping Variable: Type

7.3.4 Sensitivity analysis and error checking

The above discussion shows that given the same system inputs at a particular level, the simulation can generate similar results as the actual system in terms of vehicle and pedestrian journey times. Therefore, it is reasonable to use the average vehicle and pedestrian journey time from the simulation to reflect the efficiency of the actual system. In order to ensure this idea is robust, this section conducts the sensitivity and error analysis to check if the model can still behave according to the actual system when levels of key inputs, including vehicle and pedestrian crossing demands, change significantly.

For this purpose, the data collection process described in Section 7.3.1 was repeated on the same site during the same time-of-day but on another 13 days, including 9 weekdays and 4 weekends from 28 October to 9 November, 2008. The average vehicle and pedestrian journey times were calculated from field survey and simulation for each of all the 14 surveying days. The levels of vehicle and pedestrian crossing demand for each day, as well as the resulting average vehicle and pedestrian journey times from field survey and simulation are shown in Table 7.12.

Table 7.12 Data collection on multiple days at a location with a Zebra crossing

Field survey													Simulation	
Day	Pedestrian crossing demand (O-D, ped/h)								Vehicle traffic demand (veh/h)		Average journey time (s)		Average journey time (s)	
	11-10	11-12	13-10	13-12	10-11	10-13	12-11	12-13	W-E	E-W	Veh.	Ped.	Veh.	Ped.
1	65	243	97	103	90	96	192	50	806	771	44.06	51.89	43.80	49.29
2	54	243	88	113	86	91	207	39	834	819	43.22	48.81	41.23	47.88
3	70	234	86	95	81	99	186	48	848	836	45.67	53.27	42.29	52.60
4	58	248	98	109	83	84	185	45	844	823	42.84	54.77	41.94	51.70
5	76	237	81	96	98	97	191	38	863	880	48.41	56.82	46.23	56.85
6	59	131	53	67	79	61	101	37	512	542	33.30	27.07	33.16	26.91
7	56	109	47	64	77	54	94	35	494	458	33.27	24.96	33.24	25.61
8	62	255	97	110	100	98	184	54	858	779	39.21	40.08	38.96	42.19
9	54	235	84	101	98	90	193	39	907	853	42.20	42.79	44.39	46.82
10	51	248	76	93	96	94	192	57	817	770	37.37	43.20	37.41	46.33
11	64	250	79	108	80	96	194	48	846	818	36.23	42.07	38.23	41.06
12	55	247	97	100	96	83	182	55	869	873	49.72	57.22	49.54	56.91
13	53	128	51	69	89	61	103	42	542	515	33.22	26.31	33.25	25.89
14	52	115	45	64	85	55	95	39	509	435	33.17	25.18	33.21	24.99

Day 1-5 and 8-12 are weekdays; Day 6-7 and 13-14 are weekends

For average vehicle journey times, the correlation between values from simulation and field survey computed in SPSS is illustrated in Table 7.13, which shows that these two series of data have a strong correlation. The RMSPE between the simulated and actual values, calculated by Equation 7.1, is 3.37%, which is acceptable according to the criteria suggested by Wisconsin DOT (2002).

Table 7.13 Correlations for average vehicle journey times from field survey and simulation on multiple days in a Zebra crossing scenario

Correlations

		Avg_Veh_JT_Field	Avg_Veh_JT_Sim
Avg_Veh_JT_Field	Pearson Correlation	1	.969**
	Sig. (2-tailed)		.000
	N	14	14
Avg_Veh_JT_Sim	Pearson Correlation	.969**	1
	Sig. (2-tailed)	.000	
	N	14	14

**. Correlation is significant at the 0.01 level (2-tailed).

Avg_Veh_JT_Field = Average Vehicle Journey Time from Field Survey

Avg_Veh_JT_Sim = Average Vehicle Journey Time from Simulation

For average pedestrian journey times, the correlation between values from simulation and field survey computed in SPSS is illustrated in Table 7.14, which shows that these two series of data have a strong correlation. The RMSPE between the simulated and actual values, calculated by Equation 7.1, is 4.20%, which is acceptable according to the criteria suggested by Wisconsin DOT (2002).

Table 7.14 Correlations for average pedestrian journey times from field survey and simulation on multiple days in a Zebra crossing scenario

Correlations

		Avg_Ped_JT_Field	Avg_Ped_JT_Sim
Avg_Ped_JT_Field	Pearson Correlation	1	.987**
	Sig. (2-tailed)		.000
	N	14	14
Avg_Ped_JT_Sim	Pearson Correlation	.987**	1
	Sig. (2-tailed)	.000	
	N	14	14

**. Correlation is significant at the 0.01 level (2-tailed).
Avg_Ped_JT_Field = Average Pedestrian Journey Time from Field Survey
Avg_Ped_JT_Sim = Average Pedestrian Journey Time from Simulation

In conclusion, the behaviour of the model is reasonable and similar to the actual system at different levels of vehicle and pedestrian crossing demands and the errors are acceptable. The complete model is sufficiently reliable to be used to conduct simulation study for this scenario.

7.4 Scenario 3: a typical location with a signalised crossing

7.4.1 Data collection

The layout and description of the site chosen for the field survey is illustrated in Figure 7.12 and Table 7.15.

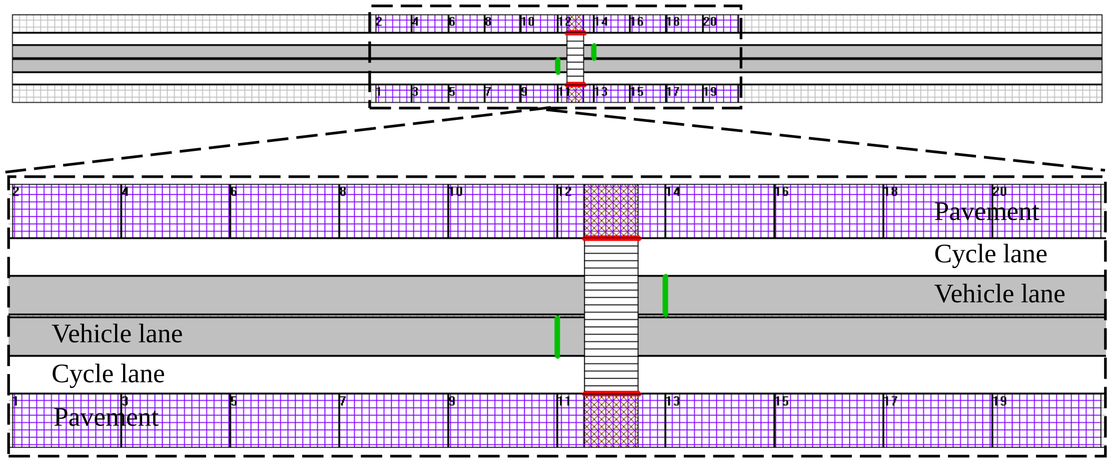


Figure 7.12 Layout of the site for model validation in a signalised crossing scenario

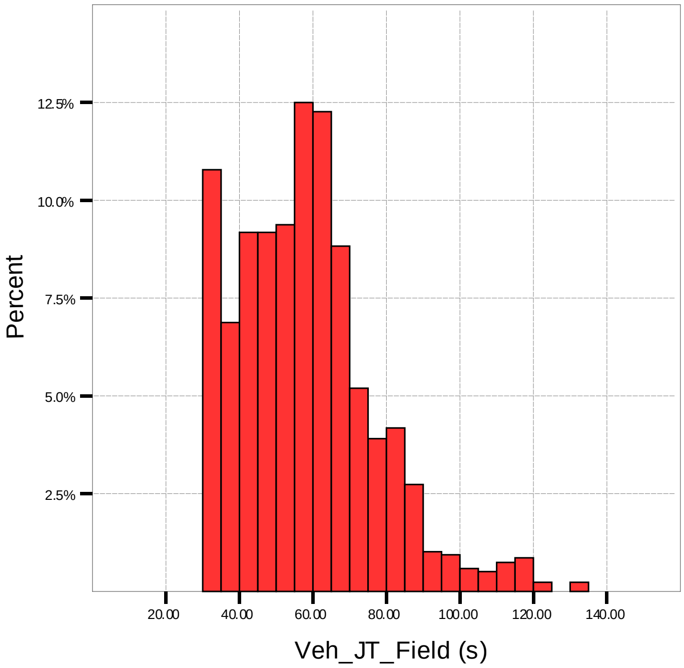
Table 7.15 Description of the site with a signalised crossing for model validation

Location	Weigongcun Road, Beijing, China
Time	7:30 to 8:30
Date	13 October 2008
Length of road section	300 m
Width of lanes	vehicle lane = 3.5 m, cycle lane = 3.5 m, pavement = 5.0 m
Width of road median	0.3 m
Speed limit	30 km/h
Type of pedestrian crossing	one fixed signal crossing in the middle of the road section; the signal timing plan is shown in Figure 7.13
Vehicle composition	89% LV, 6% MCV, 0% HCV, 4% BCR and 1% BCA
Vehicle desired speed mean	9.83 m/s
Pedestrian composition	47% YM, 44% YF, 4% OM, 5% OF

Period	P1 (50 s)	P2 (3 s)	P3 (2 s)	P4 (20 s)	P5 (5 s)
Veh. signal	G	A	R	R	R
Ped. signal	R	R	R	G	R

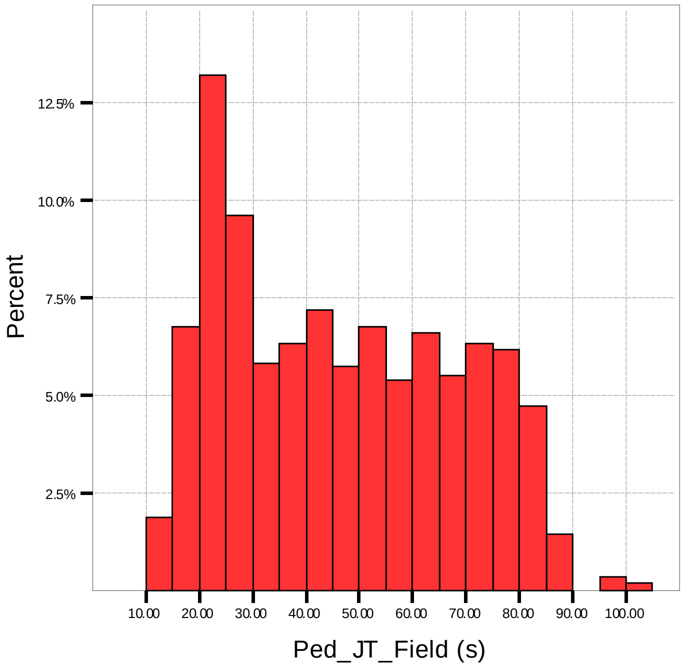
Figure 7.13 Timing plan of the signalised crossing at the site for model validation

Individual journey times of vehicles passing this section and pedestrians crossing in this area were surveyed using the video camera method discussed in Chapter 3. The distributions of individual vehicle and pedestrian journey times from field survey are illustrated in Figure 7.14 and Figure 7.15 respectively.



Sample size = 1385
Mean = 58.06
Standard deviation = 18.53
Minimum = 30.50
Maximum = 131.90

Figure 7.14 The distribution of vehicle journey times from field survey



Sample size = 1167
Mean = 46.70
Standard deviation = 21.40
Minimum = 12.20
Maximum = 102.30

Figure 7.15 The distribution of pedestrian journey times from field survey

The vehicle and pedestrian crossing demand data during the data collection time period on that day were also surveyed, as shown in Table 7.16. These demand data, as well as the site characteristics as shown in Table 7.15, were used as inputs to conduct simulation. The simulation time was set longer than the actual surveying time in order to achieve a larger sample size. The distributions of individual vehicle and pedestrian journey times from simulation are illustrated in Figure 7.16 and Figure 7.17 respectively.

Table 7.16 The vehicle and pedestrian crossing demand during surveying time period on one data collection day at a location with a signalised crossing

Pedestrian crossing demand (O-D, ped/h)																	
9-10	9-12	9-14	11-10	11-12	11-14	13-10	13-12	13-14	10-9	10-11	10-13	12-9	12-11	12-13	14-9	14-11	14-13
63	25	49	33	20	28	94	29	52	31	34	254	77	57	116	55	52	98

Vehicle traffic demand (veh/h)	
W-E	E-W
689	696

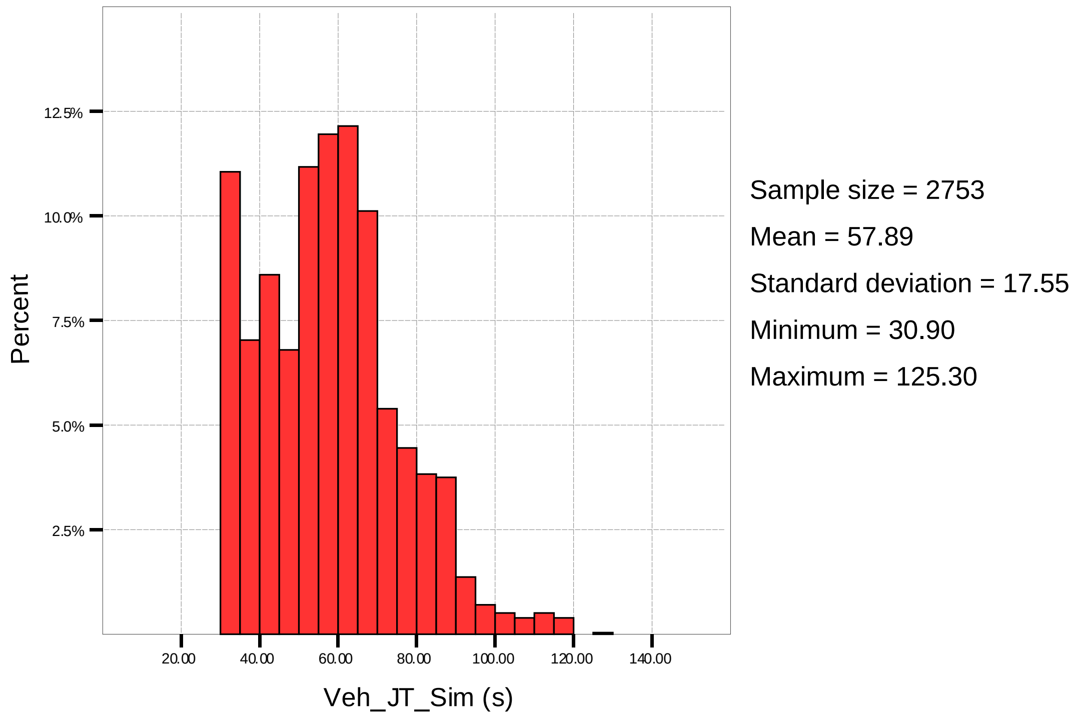
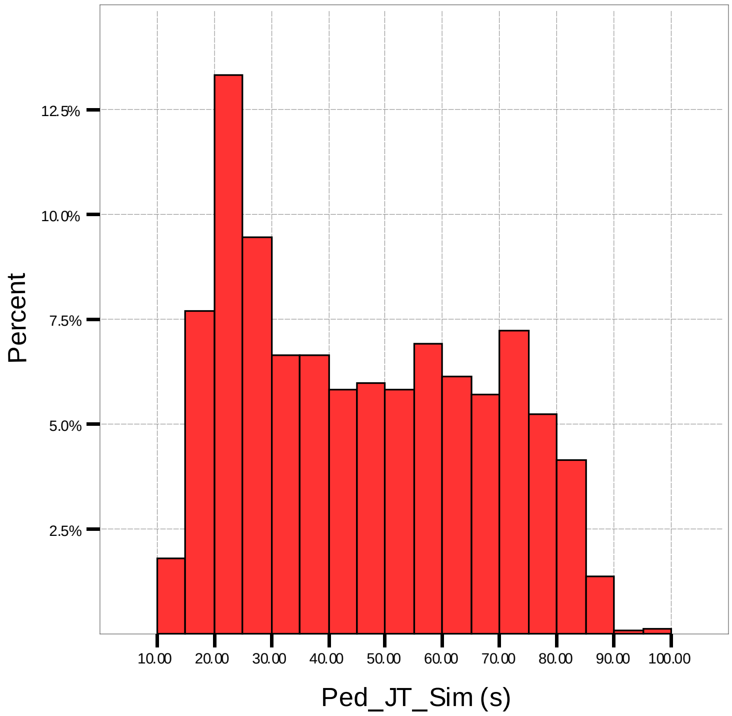


Figure 7.16 The distribution of vehicle journey times from simulation



Sample size = 2422

Mean = 45.97

Standard deviation = 21.03

Minimum = 11.70

Maximum = 98.60

Figure 7.17 The distribution of pedestrian journey times from simulation

7.4.2 Comparison of vehicle journey times

It can be seen from Figure 7.14 and Figure 7.16 that the distributions of actual and simulated vehicle journey times appear to be close. The following 2-sample K-S test was conducted to check the similarity of the distributions from the two samples.

Test method: 2-sample K-S test.

Test hypotheses: H_0: the distributions of the vehicle journey times from field survey and simulation are the same; H_1: the distributions of the vehicle journey times from field survey and simulation are not the same;

The result of the test is shown in Table 7.17, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.337). Therefore, it is reasonable to accept H_0. The distributions of vehicle journey times from field survey and simulation have no difference in a statistical meaning.

Table 7.17 The result of 2-sample K-S test for vehicle journey times in a typical signalised crossing scenario

Frequencies

	Type	N
Veh_JT	Field	1385
	Sim	2753
	Total	4138

Veh_JT = Vehicle Journey Time, N = Sample Size

Test Statistics[a]

		Veh_JT
Most Extreme Differences	Absolute	.031
	Positive	.016
	Negative	-.031
Kolmogorov-Smirnov Z		.942
Asymp. Sig. (2-tailed)		.337

a. Grouping Variable: Type

7.4.3 Comparison of pedestrian journey times

Similarly, Figure 7.15 and Figure 7.17 show that the distributions of actual and simulated pedestrian journey times appear to be close. Therefore, the same procedure was used to examine the pedestrian journey times. The following 2-sample K-S test was conducted to check the similarity of the distributions from the two samples.

Test method: 2-sample K-S test.

Test hypotheses: H_0: the distributions of the pedestrian journey times from field survey and simulation are the same; H_1: the distributions of the pedestrian journey times from field survey and simulation are not the same;

The result of the test is shown in Table 7.18, which indicates that there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.816). Therefore, it is reasonable to accept H_0. The distributions of pedestrian journey times from field survey and simulation have no difference in a statistical meaning.

Table 7.18 The result of 2-sample K-S test for pedestrian journey times in a typical signalised crossing scenario

Frequencies

	Type	N
Ped_JT	Field	1167
	Sim	2422
	Total	3589

Ped_JT = Pedestrian Journey Time, N = Sample Size

Test Statistics[a]

		Ped_JT
Most Extreme Differences	Absolute	.023
	Positive	.023
	Negative	-.003
Kolmogorov-Smirnov Z		.634
Asymp. Sig. (2-tailed)		.816

a. Grouping Variable: Type

7.3.4 Sensitivity analysis and error checking

The above discussion shows that given the same system inputs at a particular level, the simulation can generate similar results as the actual system in terms of vehicle and pedestrian journey times. Therefore, it is reasonable to use the average vehicle and pedestrian journey time from the simulation to reflect the efficiency of the actual system. In order to ensure this idea is robust, this section conducts the sensitivity and error analysis to check if the model can still behave according to the actual system when levels of key inputs, including vehicle and pedestrian crossing demands, change significantly.

For this purpose, the data collection process described in Section 7.4.1 was repeated on the same site during the same time-of-day but on another 13 days, including 9 weekdays and 4 weekends from 14 to 26 October, 2008. The average vehicle and pedestrian journey times were calculated from field survey and simulation for each of all the 14 surveying days. The levels of vehicle and pedestrian crossing demand for each day, as well as the resulting average vehicle and pedestrian journey times from field survey and simulation are shown in Table 7.19.

Table 7.19 Data collection on multiple days at a location with a signalised crossing

Field survey																							Simulation	
Day	Pedestrian crossing demand (O-D, ped/h)																		Vehicle traffic demand (veh/h)		Average journey time (s)		Average journey time (s)	
	9-10	9-12	9-14	11-10	11-12	11-14	13-10	13-12	13-14	10-9	10-11	10-13	12-9	12-11	12-13	14-9	14-11	14-13	W-E	E-W	Veh	Ped	Veh	Ped
1	63	25	49	33	20	28	94	29	52	31	34	254	77	57	116	55	52	98	689	696	58.06	46.70	57.89	45.97
2	58	27	51	31	19	24	102	27	48	29	32	247	79	59	122	56	48	108	661	652	56.34	47.01	53.55	46.33
3	68	23	53	31	21	28	99	31	54	31	36	272	78	50	109	57	51	105	621	770	63.29	46.50	66.32	46.16
4	62	23	50	35	21	27	100	30	51	31	33	252	85	53	116	52	51	104	713	686	61.53	46.47	60.48	45.87
5	66	26	57	30	21	29	96	31	54	30	34	243	86	57	100	50	51	99	756	710	62.77	46.04	58.53	46.37
6	41	15	32	20	12	16	59	17	31	19	20	158	49	35	66	34	31	62	400	409	44.28	44.40	44.06	44.35
7	32	13	26	16	10	14	49	15	26	15	16	126	42	27	55	26	25	51	339	349	43.55	44.53	43.52	44.72
8	61	28	57	34	20	29	83	30	50	30	34	260	80	62	118	50	55	93	665	714	57.82	45.98	58.56	46.59
9	69	28	49	39	19	25	90	33	58	32	33	275	85	54	118	56	49	96	694	636	54.13	46.11	56.71	45.29
10	66	25	56	37	19	27	103	31	53	34	31	277	81	58	97	46	54	98	679	741	63.92	46.64	65.98	46.32
11	69	26	53	35	20	28	84	31	52	34	34	283	74	61	108	49	46	110	700	689	58.65	46.15	56.95	45.45
12	67	26	52	34	18	24	90	30	54	30	34	251	74	54	107	51	50	89	631	672	54.26	45.99	53.79	45.09
13	40	16	34	21	12	16	55	19	34	21	21	168	47	37	67	32	31	59	408	441	44.37	44.24	44.46	44.65
14	34	13	27	17	10	14	48	15	26	17	17	136	42	31	56	26	28	52	352	354	43.16	44.52	43.49	45.01

Day 1-5 and 8-12 are weekdays; Day 6-7 and 13-14 are weekends

For average vehicle journey times, the correlation between values from simulation and field survey computed in SPSS is illustrated in Table 7.20, which shows that these two series of data have a strong correlation. The RMSPE between the simulated and actual values, calculated by Equation 7.1, is 3.17%, which is acceptable according to the criteria suggested by Wisconsin DOT (2002).

Table 7.20 Correlations for average vehicle journey times from field survey and simulation on multiple days in a signalised crossing scenario

Correlations

		Avg_Veh_JT_Field	Avg_Veh_JT_Sim
Avg_Veh_JT_Field	Pearson Correlation	1	.969**
	Sig. (2-tailed)		.000
	N	14	14
Avg_Veh_JT_Sim	Pearson Correlation	.969**	1
	Sig. (2-tailed)	.000	
	N	14	14

**. Correlation is significant at the 0.01 level (2-tailed).

Avg_Veh_JT_Field = Average Vehicle Journey Time from Field Survey

Avg_Veh_JT_Sim = Average Vehicle Journey Time from Simulation

For average pedestrian journey times, the correlation between values from simulation and field survey computed in SPSS is illustrated in Table 7.21, which shows that these two series of data have a strong correlation. The RMSPE between the simulated and actual values, calculated by Equation 7.1, is 1.22%, which is acceptable according to the criteria suggested by Wisconsin DOT (2002).

Table 7.21 Correlations for average pedestrian journey times from field survey and simulation on multiple days in a signalised crossing scenario

Correlations

		Avg_Ped_JT_Field	Avg_Ped_JT_Sim
Avg_Ped_JT_Field	Pearson Correlation	1	.828**
	Sig. (2-tailed)		.000
	N	14	14
Avg_Ped_JT_Sim	Pearson Correlation	.828**	1
	Sig. (2-tailed)	.000	
	N	14	14

**. Correlation is significant at the 0.01 level (2-tailed).

Avg_Ped_JT_Field = Average Pedestrian Journey Time from Field Survey

Avg_Ped_JT_Sim = Average Pedestrian Journey Time from Simulation

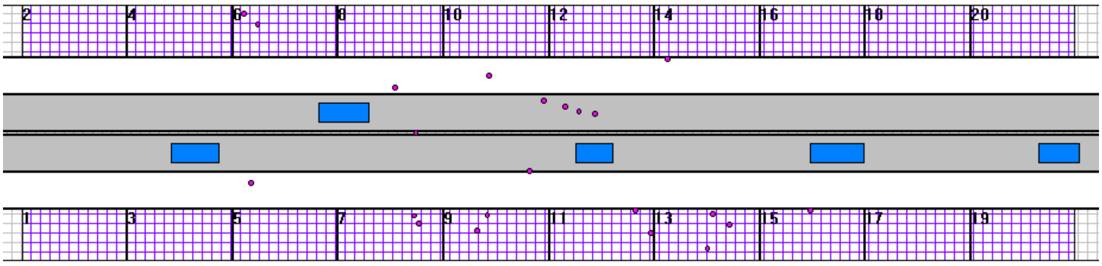
In conclusion, the behaviour of the model is reasonable and similar to the actual system at different levels of vehicle and pedestrian crossing demands and the errors are acceptable. The complete model is sufficiently reliable to be used to conduct simulation study for this scenario.

7.5 Conclusion

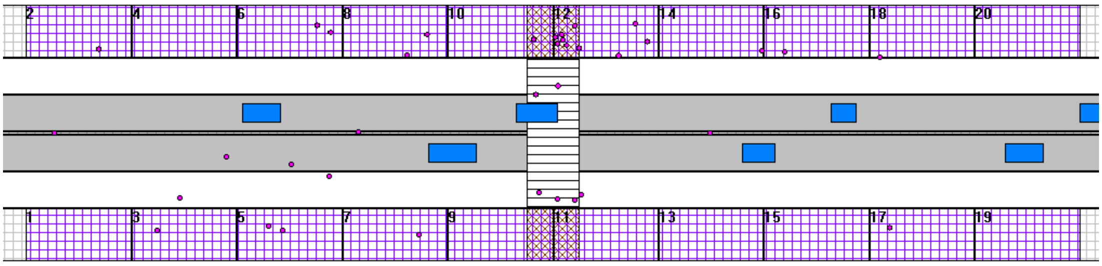
This chapter described the validation process for the complete micro-simulation model. The method firstly proposed by Park and Schneeberger (2003) and then practised by many other researchers (Li et al (2010), Ishaque and Norland (2009), Du (2008), Min et al (2008) and etc) was applied for model validation.

The complete model has been validated in a number of typical interaction scenarios, including both unsignalised and signalised situations. For each scenario, data collected for model validation were independent to those used for model development, either at different locations or the same location but on different days. First, data collection was conducted during a particular time on a typical weekday to check if the distributions of actual and simulated vehicle/pedestrian journey times were the same given the same system inputs. The Kolmogorov-Smirnov (K-S) test was applied for this purpose. Then, data were collected on another 13 days including both weekdays and weekends. The average vehicle/pedestrian journey time were calculated from field survey and simulation for each of all the 14 days. These two series of data were used for sensitivity analysis and error checking to examine if the behaviour (in terms of average vehicle and pedestrian journey times) of the simulation system was still similar to the actual system when inputs changed. The correlation analysis and RMSPE calculation were conducted for this purpose. The validation for the complete model showed that the assumptions made during the model development process were reasonable and the integrated model was sufficiently reliable to be used to conduct simulation studies.

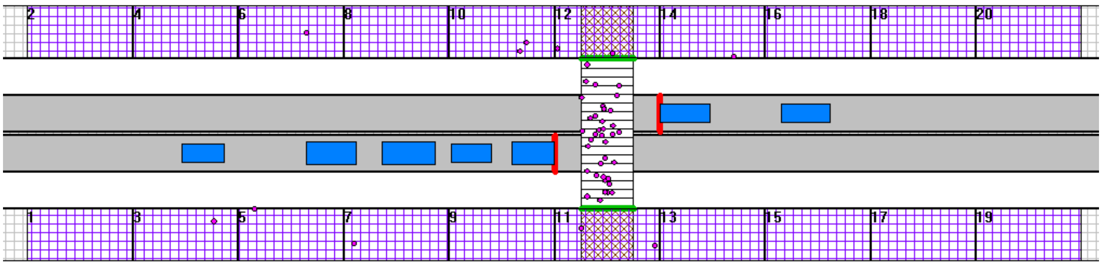
In addition to the quantitative validation process presented above, the simulation program developed in this research also provides a visualisation interface for the user to examine the model performance in a qualitative way. The rationale of the interaction behaviour can also be subjectively examined from the animation generated by the simulation program. Three snapshots from the visualisation are illustrated in Figure 7.18, which shows that the complete simulation program can mimic the pedestrian-vehicle interaction process in a realistic fashion.



(a) A typical location with no-control



(b) A typical location with a Zebra crossing



(c) A typical location with a signalised crossing

Figure 7.18 The visualisation of the simulated pedestrian-vehicle interaction behaviour in three different types of scenarios

CHAPTER 8
MODEL APPLICATION

8.1 Introduction

The frequent interaction between vehicles and crossing pedestrians is one of the main characteristics in the urban street environment in China due to less disciplined behaviour on the streets in a mixed traffic condition. The conflicts between motorised vehicles and pedestrians at mid-block locations contribute much to traffic congestion and accidents. Few studies have been conducted to analyse the operational performances of unsignalised locations or to compare the effects among different treatments. There are few guidelines regarding the set-up of pedestrian crossings either. The inadequate understandings and improper management of the mixed traffic resulted in a number of serious problems such as decreased efficiency, potential accidents and pollutions to the environment. This chapter describes two applications conducted by using the developed and validated micro-simulation model. First, the model is applied to study a typical no-control road section by analysing the system performances with different combinations of vehicular traffic and pedestrian crossing demand, in terms of efficiency, safety and environmental impact. Second, the multi-criteria performances of different treatments including do-nothing, Zebra crossing, fixed-signal crossing and Puffin crossing at two typical types of mid-block locations are compared. Recommendations and interpretations are given at the end of each application.

8.2 Definition of model indicators

The operational performances in different scenarios are assessed in terms of 3 aspects including efficiency, safety and environmental impact for multi-criteria evaluation. Model indicators for these 3 aspects are defined as follows.

1. Efficiency:

System efficiency is indicated by road section capacity (2-way) and average delays including average pedestrian delay, average vehicle delay and average traveller delay. Road section capacity is obtained by summing up the maximum number of vehicles able to pass through the road section under the mixed-traffic condition. Vehicle delay is defined as the time difference between the actual time and desired time when a vehicle passes through the road section. The desired journey time of a vehicle is calculated by dividing the length of the road section by the vehicle's desired speed. Average vehicle journey time can then be obtained by averaging all journey times of vehicles passing through the road section during the surveying period. Pedestrian delay is defined as the time difference between the actual time and desired time when the pedestrian moves from his/her origin position to destination position. The desired journey time of a pedestrian is calculated by dividing the length of the direct link between the origin position and destination position by the pedestrian's desired speed. Average pedestrian journey time can then be obtained by averaging all journey times of pedestrians who finished their crossing activities during the surveying period. Average traveller delay is an integrated index for pedestrian and vehicle delays and can be given by Equation 8.1.

$$D_{trvlr} = \frac{\sum_{i=1}^{n} D_{Veh,i} O_{Veh,i} C_{Veh,i} + \sum_{j=1}^{m} D_{Ped,j} C_{Ped,j}}{\sum_{i=1}^{n} O_{Veh,i} + m} \qquad (8.1)$$

Where,

D_{trvlr} is the average traveller delay, (s);

n is the total number of vehicles completing their journey in the simulation period;

$D_{veh,i}$ is the delay of vehicle i, (s);

$O_{veh,i}$ is the occupancy of vehicle i;

$C_{veh,i}$ is the average relative cost of time of passengers in vehicle i;

m is the total number of pedestrians completing their journey in the simulation period;

$D_{ped,j}$ is the delay of pedestrian j, (s);

$C_{ped,j}$ is the average relative cost of time of pedestrians.

The mean vehicle occupancy is assigned to each vehicle according to previous studies in China and field observation by the author, as shown in Table 8.1.

Table 8.1 Mean occupancy values for various vehicle types used in this research

Vehicle type	Mean occupancy	Source
Private car	1.68	Lin et al (2006)
Taxi	1.27	Lin et al (2006)
Van	2.72	Field observation
HGV	1.20	Field observation
Bus (including regular and articulated)	64.4	Lin et al (2006)

The relative value of time for each transport mode (including pedestrians) is closely related to the social-economic characteristics of a country. As there is currently no such study in China, it is assigned to 1.0 for any mode, representing that all transport modes have equal rights on the road.

2. Safety:

For safety, although it is not possible to directly predict accidents with existing micro-simulation techniques, some researchers suggest the use of initial time-to-collision as a surrogate measure for safety, as lower initial time-to-collision indicates a higher probability of accidents (Gettman and Head 2003). The initial time-to-collision is defined as the projected time-to-collision of the conflicting vehicle to the pedestrian crossing that vehicle lane when the pedestrian just leaves that lane, if their speed and direction remain unchanged with the status at the time when the pedestrian just steps onto that lane (Gettman and Head 2003). Following this suggestion, the level of safety is then primarily indicated by the percentage of pedestrian crossing actions with initial time-to-collision below or equal to 0 s. For the specific 2-way-2-lane scenarios in this research, any pedestrian exhibits two crossing actions to complete crossing the whole road. This percentage can be calculated by Equation 8.2.

$$Pct_{TTC0} = \frac{n_{TTC0}}{2n_{Ped}} \times 100\% \tag{8.2}$$

Where,

Pct_{TTC0} is the percentage of pedestrian crossing actions with initial time-to-collision below or equal to 0 s;

n_{TTC0} is the total number of actions with initial time-to-collision below or equal to 0 s;

n_{ped} is the total number of pedestrians completing their journey in the simulation period.

In addition, some other researchers in China found that pedestrians became more aggressive and were likely to commit risky behaviour when their accumulative waiting time exceeded 20 s (Chen and Qi 2002). Therefore, the percentage of pedestrians with waiting time above 20 s is also selected as a reference indicator for pedestrian safety.

3. Environmental impact:

The environmental impact of the system is evaluated by vehicle emissions generated by linking the micro-simulation model developed in this research with a certain external vehicle emission model. An instantaneous emission model capable of outputting second-by-second emission data with instantaneous vehicle speed and acceleration changes was selected for this purpose. This model was developed by Panis et al (2006) based on the measurement of emissions of 25 instrumented vehicles, including 12 petrol cars, 5 diesel cars, 6 buses and 2 trucks. Emission functions for each type of vehicle was derived with instantaneous vehicle speed and acceleration as parameters using non-linear multiple regression techniques, covering NO_x, HC, PM and CO_2 pollutants. The general function for each pollutant is given by Equation 8.3 (Panis et al 2006).

$$E_n(t) = max\{E_0, f_1 + f_2 v_n(t) + f_3 v_n^2(t) + f_4 a_n(t) + f_5 a_n^2(t) + f_6 v_n(t) a_n(t)\} \qquad (8.3)$$

Where,

$E_n(t)$ is instantaneous emission, specified for each vehicle and pollutant type, (g/s);

E_0 is the lower limit of emission specified for each vehicle and pollutant type, (g/s);

f_1 to f_6 are emission constants specified for each vehicle and pollutant type determined by regression analysis;

$v_n(t)$, $a_n(t)$ are instantaneous speed (m/s), acceleration (m/s^2) of vehicle *n* at time *t*.

It should be noted that the validity of such an emission model itself was highly related to the characteristics and compositions of vehicles according to the local situation and this model was only validated in some European countries. Therefore, the absolute values generated by such a model were not necessarily accurate when it was used in China. However, the relative differences for various scenarios could still be used as a reference to reflect the environmental impact in each scenario.

Although this model is capable of generating various emission data, only CO_2 is selected for evaluation studies in this research as it is one of the major green house gases and also closely related to vehicle fuel consumption. More results are worthy to be mentioned when the validity of the emission model is fully proved for its use in China. The parameters in this model for CO_2 of different types of vehicles are given by Panis et al (2006), as shown in Table 8.2.

Table 8.2 Parameters in the instantaneous emission model for CO_2

Pollutant	Vehicle type	E_0	f_1	f_2	f_3	f_4	f_5	f_6
CO_2	Petrol car	0	5.53e-01	1.61e-01	-2.89e-03	2.66e-01	5.11e-01	1.83e-01
	Diesel car	0	3.24e-01	8.59e-02	4.96e-03	-5.86e-02	4.48e-01	2.30e-01
	Heavy duty vehicle	0	1.52e+00	-6.95e-02	-6.95e-02	4.71e+00	5.88e+00	2.09e+00
	Bus	0	9.04e-01	-4.27e-02	-4.27e-02	2.81e+00	3.45e+00	1.22e+00

(Source: Panis et al 2006)

8.3 System performance at a typical mid-block location with no-control

8.3.1 Scenario description

In this section, a hypothetical no-control mid-block area is created to represent the typical mixed traffic condition involving pedestrian-vehicle interactions in China. Although the scenario is created hypothetically, its main parameters are determined in accordance with the real situation, which were surveyed during the data collection process in this research. The main parameters of this scenario are described in Figure 8.1 and Table 8.3.

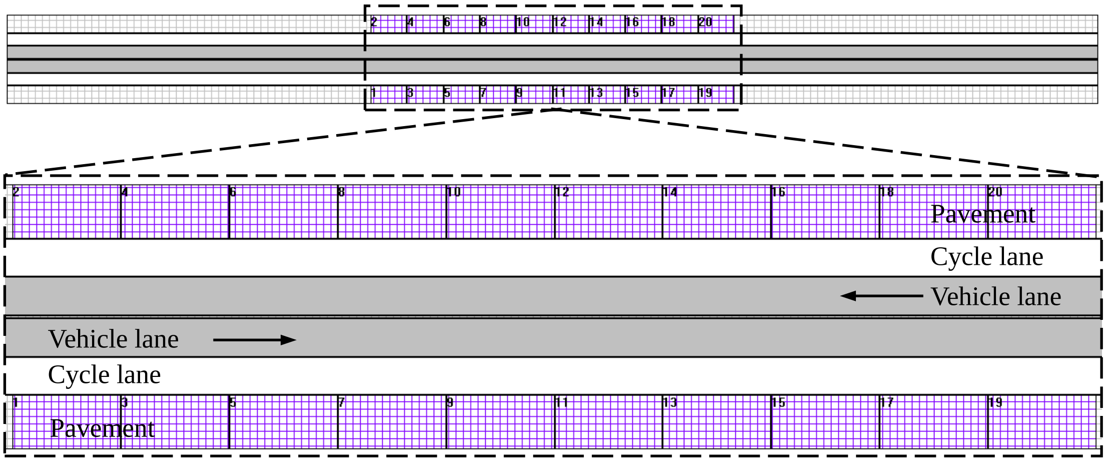


Figure 8.1 Scenario for evaluation study of the performance at a typical mid-block location with no-control

Table 8.3 Scenario for evaluation study at a typical no-control mid-block location

Parameters	Description
Geometry	A typical 2-way-2-lane road section with a length of 300 m, vehicle lane width of 3.5 m, cycle lane width of 3.5 m, pavement width of 5.0 m and road median width of 0.3 m; the 100 m in the middle of the road section is considered as the frequent interaction area
Saturation flow per lane	1400 pcu/h
Total vehicle traffic demand (2-way)	Ranging from 280 to 3080 veh/h (10% to 110% of road saturation flow), with an increment of 280 veh/h (10% of road saturation flow) per each simulation run, with an exception that for analysis of road capacity, an over-saturated flow (5000 veh/h/lane) is assigned to each vehicle lane for each simulation run. The traffic demand is assigned evenly to the two lanes.
Vehicle composition	85% LV (of which 70% are petrol and 30% are diesel), 12% MCV, 1% HCV, 1% BCR and 1% BCA
Vehicle desired speed mean	9.15 m/s
Total pedestrian crossing demand (2-way) along the 100 m frequent interaction area (D_{ped})	Ranging from 240 to 3600 ped/h, with an increment of 240 ped/h per each simulation run, with an exception that for analysis of road capacity, 0-6000 ped/h with an increment of 240 ped/h is used as input for each simulation run. The total pedestrian crossing demand is evenly assigned to the total crossing demand of each direction (total-south-to-north and total-north-to-south).
Pedestrian demand spatial distribution	$x_{ori}, x_{dest} \overset{iid}{\sim} N(x_0, 25)$, where x_{ori} and x_{dest} are the x coordinates of a pedestrian's origin and destination positions respectively and x_0 is the x coordinate of the centroid of the road section, representing that pedestrian crossing demand is centralised in the middle of the frequent interaction area.
Pedestrian demand temporal distribution	The pedestrian appearing time interval for each *O-D* area pair S_i-S_j ($i, j \in \mathbb{N}$ and $i+j=2k+1$, $k \in \mathbb{N}$ *and* $k \leq 19$) is assumed to yield to a exponential distribution $E(\frac{1}{d_{ped,i,j}})$, where $d_{ped,i,j}$ is the crossing demand from S_i to S_j ($d_{ped,i,j} \neq 0$)
Pedestrian composition	41% YM, 38% YF, 11% OM and 10% OF

8.3.2 Analysis of results

The simulation program was run for several times to obtain the performances under different combinations of vehicle traffic demand (2-way) and pedestrian crossing demand (2-way). The results are illustrated as follows.

1. Efficiency:

(1) Road capacity:

In order to obtain levels of road capacity under different mixed traffic conditions, the input of pedestrian crossing demand (2-way) changes from 0 to 6000 ped/h, with an increment of 240 ped/h per step. For each level of pedestrian crossing demand, the simulation is run for 5 times with the input of vehicle traffic demand (2-way) exceeding the saturation flow of the road. The change of levels of the road section capacity as the pedestrian crossing demand increases is illustrated in Figure 8.2.

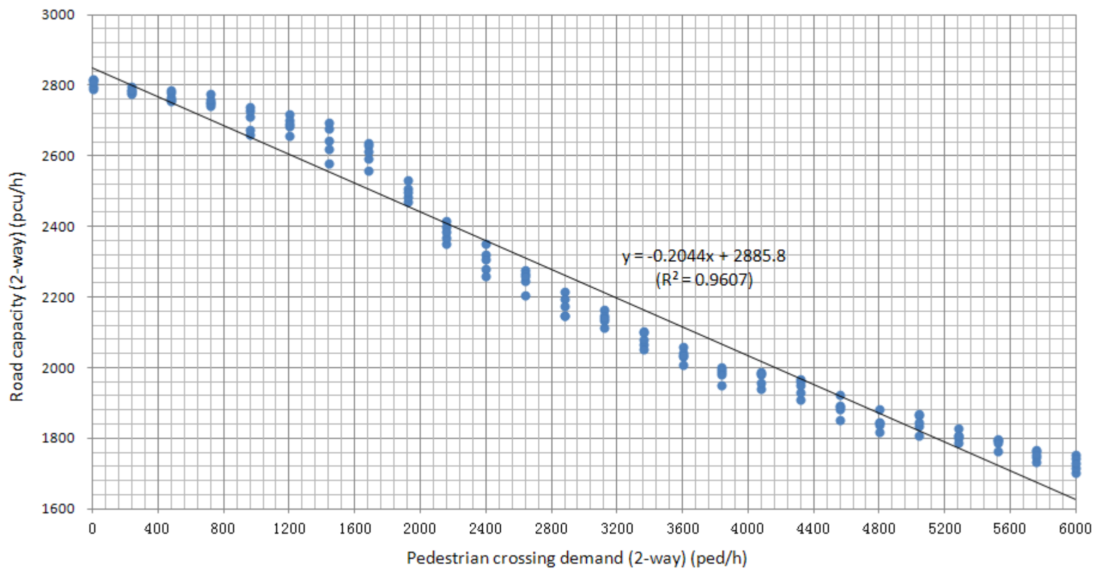


Figure 8.2 The level of road capacity decreases as pedestrian crossing demand increases at a no-control mid-block location

Overall, the level of the road section capacity decreases as the pedestrian crossing demand increases. The drop of the capacity becomes more significant when the pedestrian crossing demand exceeds 1200 ped/h. An explanation is that when the crossing demand increases, pedestrians are more likely to form a larger waiting group, which makes them feel safer and more confident and thus exert more aggressive and risky behaviour such as accepting smaller gaps and forcing the traffic slow down or stop for a longer time. When the crossing demand is above 5000 ped/h (although this is unusual in reality), there is significant chaos on the road, where the capacity drops to below 1800 pcu/h, only approximately 65% of the original capacity

(2800 pcu/h) when there is no interference of pedestrians.

(2) Average delays:

The average pedestrian delay, average vehicle delay and average traveller delay for those who completed their journeys during the simulation time are illustrated in Figure 8.3, Figure 8.4 and Figure 8.5 respectively.

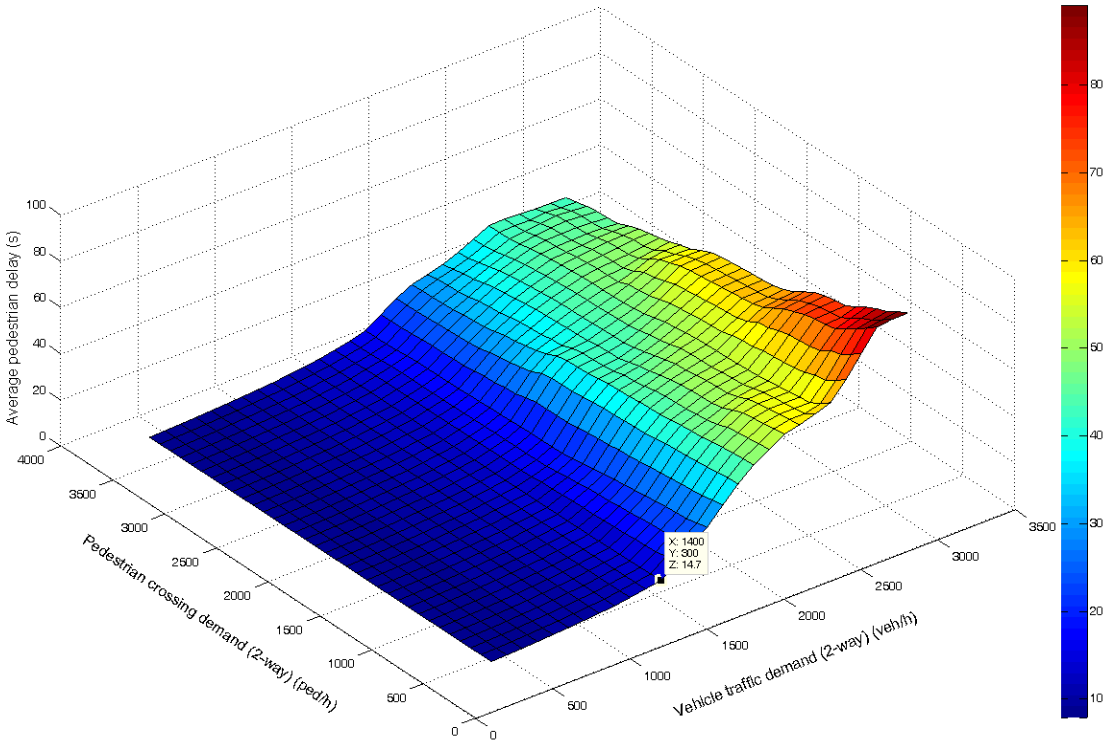


Figure 8.3 Average pedestrian delays under different combinations of vehicle traffic and pedestrian crossing demand at a typical mid-block location with no-control

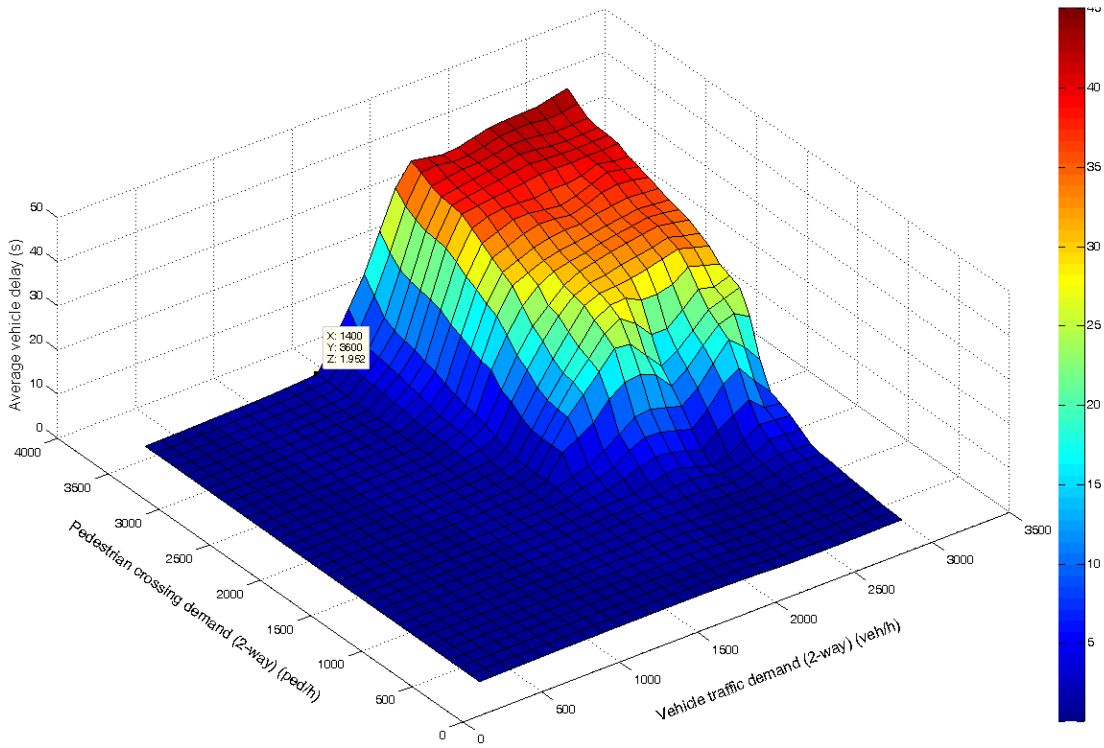


Figure 8.4 Average vehicle delays under different combinations of vehicle traffic and pedestrian crossing demand at a typical mid-block location with no-control

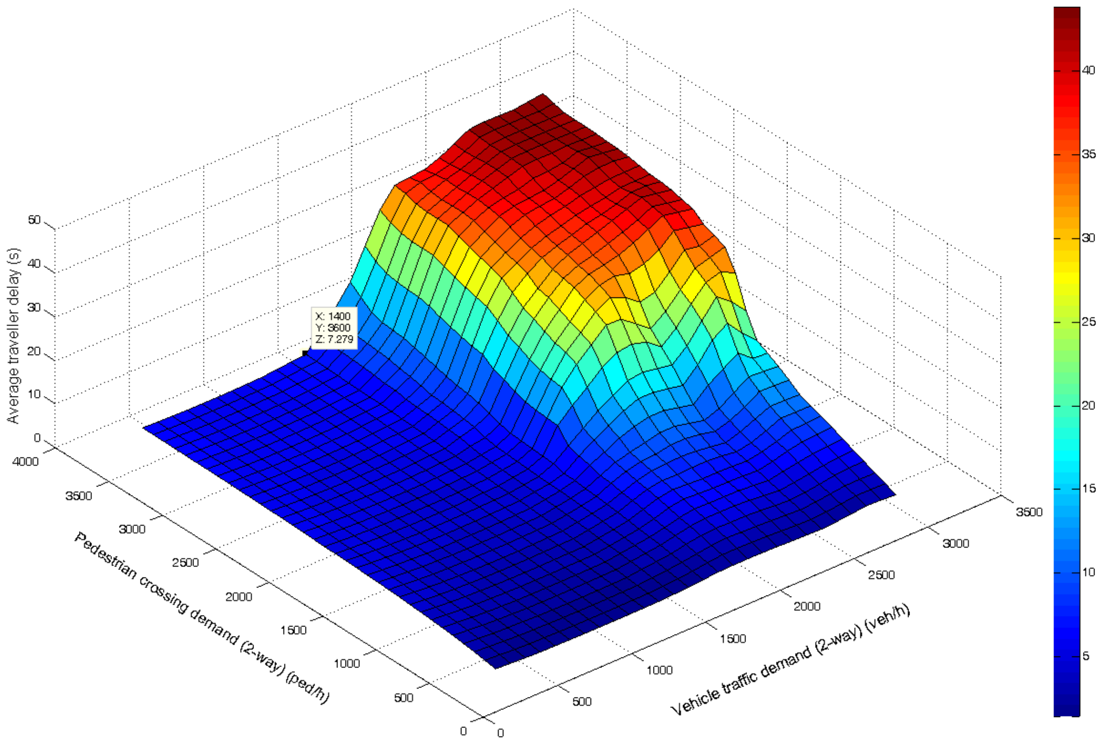


Figure 8.5 Average traveller delays under different combinations of vehicle traffic and pedestrian crossing demand at a typical mid-block location with no-control

Figure 8.3 shows that the average pedestrian delay grows as the vehicle traffic demand increases for each level of pedestrian crossing demand. This trend becomes more dramatic when the vehicle traffic demand (2-way) passes 1400 veh/h, below which the trend is not obvious and for most cases the average pedestrian delay can keep at a lower level of below 15 s. This can be explained by the fact that as the vehicle demand increases, there are fewer available gaps for pedestrians to make use of, resulting in more pedestrian delays. However, the level of the delay becomes stable again (but in a higher level) when the vehicle demand exceeds the road capacity, when the percentage of available gaps stay at a constant level. On the other hand, for each vehicle traffic demand, the average pedestrian delay only experiences a little drop for each increase of pedestrian crossing demand. This is because when the crossing demand increases, pedestrians are more likely to form large waiting groups, which make them behave more aggressively, sometimes force the traffic to a stop and therefore reduce their delays. This trend is more obvious when the traffic demand approaches the road capacity, when the traffic flow is in a saturated and unstable status, and large groups of pedestrians often make use of the fluctuation in traffic to cut off vehicle flow. Further, the highest value of average pedestrian delay appears when the traffic demand is high and meanwhile the pedestrian demand is low. This is because the dense vehicle flow provides fewer available gaps for pedestrians whilst the number of pedestrians is not sufficiently large to form groups to frequently force the traffic to slow down or stop for them.

For average vehicle delay, it can be seen from Figure 8.4 that it stays at a relatively constant level (less than 2 s) when the vehicle demand (2-way) is below 1400 veh/h, where there are sufficient available gaps for pedestrians to utilise and pedestrians are less likely to form large waiting groups. Therefore, the vehicle flow is less likely to be influenced by pedestrian crossing behaviour. For each level of vehicle demand exceeding 1400 veh/h, the average vehicle delay increases as the pedestrian crossing demand goes up. Especially, when the vehicle demand exceeds 2100 veh/h whilst the pedestrian crossing demand is over 2500 ped/h, the average vehicle delay becomes significant, mostly above 30 s. This is because frequent congestion happens when the traffic is in a near saturated condition. If the pedestrian crossing demand is sufficiently high, they are more likely to form large crossing groups, in which they behave more aggressively and sometimes force the conflicting vehicles to yield to them, resulting in higher vehicle delays.

The average traveller delay shown in Figure 8.5 has a similar trend as the average vehicle delay in that vehicles have higher occupancy than pedestrians and thus have more weights when being included in the calculation for average traveller delay. Significant average traveller delay can be observed when vehicle traffic demand is above 2100 veh/h and pedestrian crossing demand exceeds 2300 ped/h, where most of the values are above 30 s.

2. Pedestrian safety:

(1) Percentage of pedestrian crossing actions with initial time-to-collision below or equal to 0 s

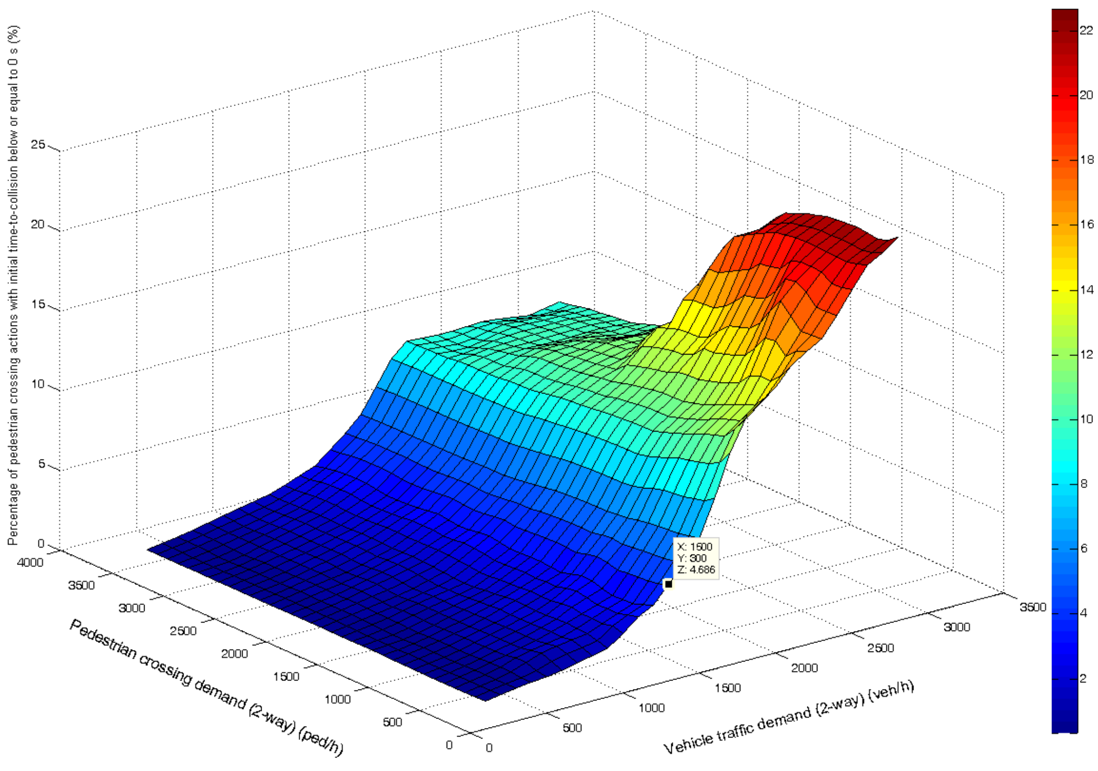


Figure 8.6 Percentage of pedestrian crossing actions with initial time-to-collision below or equal to 0 s under different combinations of vehicle traffic and pedestrian crossing demand at a typical mid-block location with no-control

Overall, the vehicle traffic demand has significant influences on the road safety. When the vehicle traffic demand is less than 1500 veh/h, the percentage of risky behaviour (initial time-to-collision below or equals to 0 s) can be controlled to less than 5% for most cases. For each level of pedestrian crossing demand, the percentage of risky actions increases as the vehicle traffic demand rises. Higher values appear at the region in which vehicle demand exceeds 1800 veh/h and meanwhile pedestrian crossing demand is less than 2000 ped/h, where most values are above 10%. This can be explained that, in this situation, there are fewer proper gaps in the traffic whereas the number of pedestrians is not sufficient for them to make use of the grouping advantage. Thus, the probability of choosing smaller gaps rises. If the pedestrian crossing demand is higher, as discussed previously, the road capacity drops accordingly and traffic is more likely to become unstable. Consequently, some pedestrians can utilise the forced gaps in the traffic. For example, if one pedestrian in a large waiting group accepts a smaller gap and forces the conflicting vehicle to slow down or stop, other people in the same group can then utilise that gap to cross the road. This can lead to a decrease in the percentage of smaller initial time-to-collision.

(2) Percentage of pedestrians with waiting time above 20 s

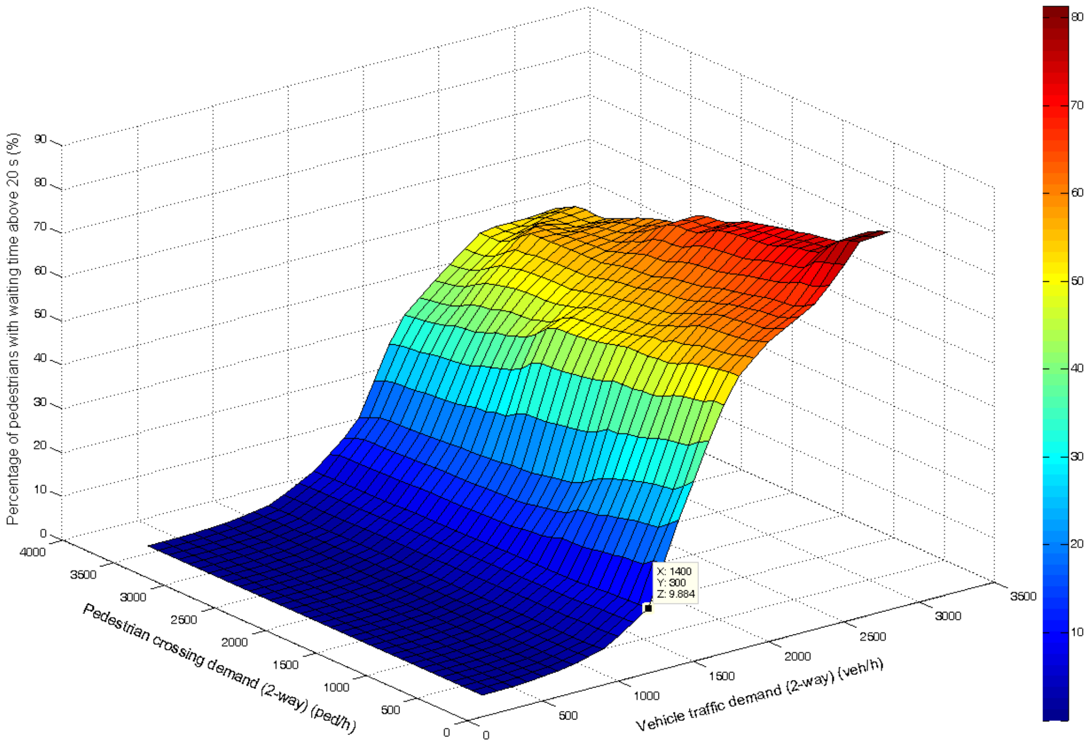


Figure 8.7 Percentage of pedestrians with waiting time above 20 s under different combinations of vehicle traffic and pedestrian crossing demand at a typical mid-block location with no-control

The trend of this indicator has similar pattern with the one of the other safety indicator discussed previously, with an exception that the pedestrian crossing demand has less effect on reducing waiting time. The traffic demand has major impact on pedestrians' waiting time. When the traffic demand is lower than 1400, the percentage of waiting time above 20 s can be controlled to less than 10% for most cases. When the traffic demand is higher than 1900 veh/h, for most cases, over 50% pedestrians have to wait more than 20 s to finish crossing the road, which may lead to potential risky behaviour.

3. Environmental impact:

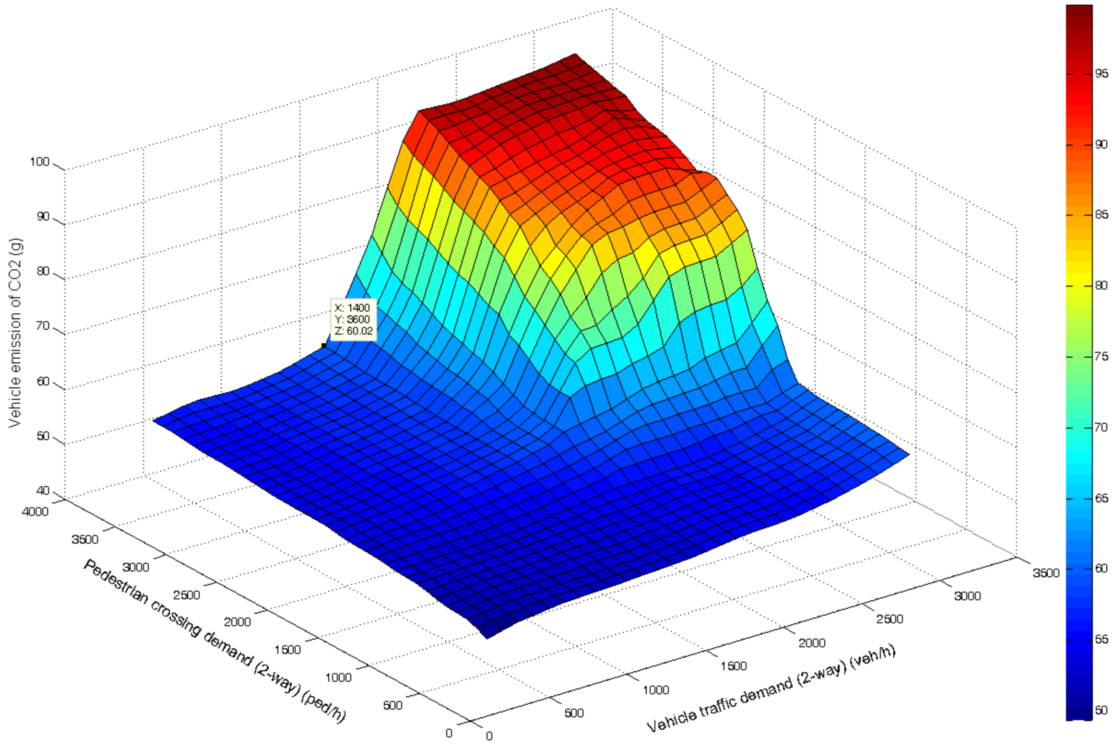


Figure 8.8 Average vehicle emission of CO_2 under different combinations of vehicle traffic and pedestrian crossing demand at a typical mid-block location with no-control

When either the traffic or pedestrian demand is low, the average vehicle CO_2 emission keeps at a normal level of approximately 60g/300m. Higher average emission of CO_2 occurs in the region where traffic demand is above 1600 veh/h and pedestrian crossing demand is above 1700 ped/h, when pedestrian-vehicle interaction becomes more frequent. More stop-and-go phenomena resulted from increasingly unstable traffic flow can be one possible cause for the increase of the CO_2 emission.

8.3.3 Summary and recommendation

Overall, the vehicle demand has major effect on system efficiency, pedestrian safety and environmental impact. When the vehicle demand (2-way) is lower than a level of 1400-1600 veh/h, pedestrians and vehicles can well negotiate their right-of-way and the system performs well in terms of all three aspects. Therefore, generally, there is no need for special treatments. On the contrary, when the level of vehicle traffic demand is above 1600 veh/h, the system is likely to experience dramatic decrease in efficiency and increase in CO_2 emission when the pedestrian crossing demand is also high, or undergo a severe decrease in pedestrian safety when the pedestrian crossing demand is low. Therefore, special treatments should be considered in this situation.

8.4 Comparison of different treatments in typical scenarios

In this section, two hypothetical mid-block areas are created to represent two typical scenarios involving pedestrian-vehicle interactions in China, including a downtown area, where both the vehicle traffic and pedestrian crossing demand are high and a suburban area, where the vehicle traffic is moderate and the pedestrian crossing demand is low. Although the scenarios are created hypothetically, the main parameters of these two scenarios are defined to be similar to the situation in the real world, which was surveyed during the data collection process in this research.

8.4.1 Scenario 1: downtown area

8.4.1.1 Scenario description

The main characteristics of a typical road section in downtown area in Beijing, China are described in Table 8.4. Usually, both the vehicle traffic demand and pedestrian crossing demand are high at such locations.

Table 8.4 Characteristics of a typical road section in downtown area in China

Geometry	2-way-2-lane road section
Length	300 m
Width of lanes	vehicle lane = 3.5 m, cycle lane = 3.5 m, pavement = 5.0 m
Width of road median	0.3 m
Speed limit	30 km/h
Vehicle composition	85% LV, 12% MCV, 1% HCV, 1% BCR and 1% BCA
Vehicle desired speed mean	9.15 m/s
Pedestrian composition	41% YM, 38% YF, 11% OM, and 10% OF
Vehicle traffic demand	900 veh/h for each vehicle lane
Pedestrian crossing demand (2-way)	2400 ped/h, with the same temporal and spatial distribution specified in Table 8.3

The performances of the following four treatments are proposed to be compared using the micro-simulation developed in this research.

1. Do-nothing:

This is the based scenario; no pedestrian crossing is established in the scenario.

2. Zebra crossing:

In this treatment, a Zebra crossing is placed in the middle of the road section.

3. Fixed-time signal crossing:

In this treatment, a fixed-time signal crossing is placed in the middle of the road section. The signal timing is determined according to the local standard specified in MOHURD (1995), which suggests:

(1) The start and end lost time for vehicle green period is 2 s;
(2) The vehicle amber period is 3 s;
(3) All-red period for vehicle clearance is 2 s;
(4) All-red period for pedestrian clearance is 5 s;
(5) The capacity for each vehicle lane should exceed demand;
(6) The pedestrian waiting time should be minimised.

The saturation flow for each vehicle lane in the urban street environment in Beijing, China was previously surveyed by the author (discussed in Chapter 4). The value 1400 pcu/h obtained from the field survey is used to determine signal timing. Based on the above suggestion, the relationship expressed in Equation 8.4 can be obtained.

$$\begin{cases} 1400 \times \frac{G_{Veh}+3-2}{G_{Veh}+3+2+G_{Ped}+5} \geq 900 \\ G_{Ped} \geq \frac{w}{v_{Ped,min}} \end{cases} \tag{8.4}$$

Where,

G_{Veh} is vehicle green time;

G_{Ped} is pedestrian green time;

w is the width of the road, which is 14.3 m in this scenario;

$v_{Ped,min}$ is the minimum speed of pedestrians, which is 0.90 m/s according to the calibration by the author (discussed in Chapter 5).

Equation 8.4 can then be transformed into Equation 8.5.

$$\begin{cases} G_{Veh} \geq \frac{9G_{Ped}+76}{5} \\ G_{Ped} \geq 15.89 \end{cases} \tag{8.5}$$

Therefore, to minimise pedestrian delay and for calculation convenience, G_{Ped} is set to be 16 s and G_{Veh} 44 s, which makes the whole cycle 70 s, as shown in Figure 8.9.

Period	P1 (44 s)	P2 (3 s)	P3 (2 s)	P4 (16 s)	P5 (5 s)
Veh. signal	G	A	R	R	R
Ped. signal	R	R	R	G	R

Figure 8.9 Signal timing for a fixed-time signal crossing at a typical downtown area

<u>4. Puffin crossing:</u>

In this treatment, a Puffin signal crossing is placed in the middle of the road section. The signal timing is determined on the basis of the fixed-time signal discussed above, considering default parameters suggested by Walker et al (2005). The signal timing of the Puffin crossing is illustrated in Figure 8.10.

Period	P1 (min=57 s, extendable)	P2 (3 s)	P3 (2 s)	P4 (16 s)	P5 (2 s)	P6 (max=11 s, may gap off)	P7 (2 s)
Veh. signal	G	A	A	R	R	R	R & A
Ped. signal	R	R	R	R	G	R	R

Figure 8.10 Signal timing for a Puffin crossing at a typical downtown area

8.4.1.2 Comparison of different treatments

The multi-criteria comparison of the performances of different treatments is illustrated in Figure 8.11, Figure 8.12 and Figure 8.13.

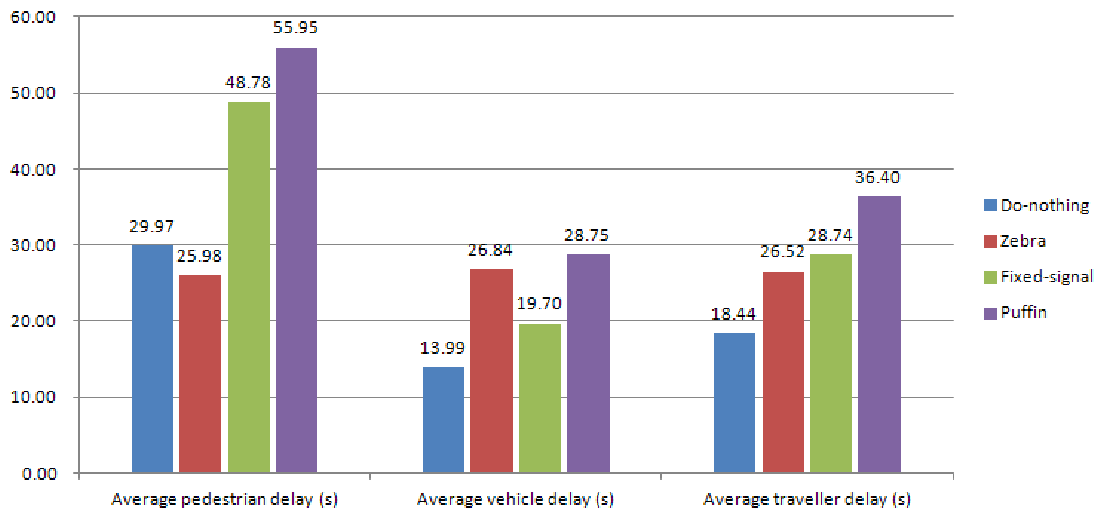


Figure 8.11 Comparison of system efficiency of different treatments in a typical downtown area

It can be seen from Figure 8.11 that the introduction of a Zebra crossing brings a slight drop (13.3%) to average pedestrian delay and a significant increase (91.9%) to average vehicle delay, compared to the base scenario. This is not because more vehicles give way to pedestrians but due to the fact that at a Zebra crossing pedestrians are more likely to form large crossing groups, which sometimes force the traffic to a stop, resulting in more vehicle delays.

The introduction of signalised crossings usually cannot improve the efficiency because of the lost time and all-red periods in the signal timing. However, the fixed-signal has a better performance in terms of average vehicle delay. This is because when the conflicts between pedestrians and vehicles are high, the traffic signal can regulate their behaviour and reduce the number of jaywalking pedestrians to some extent to make the traffic flow smoother and thus offset some disadvantages due to signal timing.

When both the vehicle traffic demand and pedestrian crossing demand are high, the performance of a Puffin crossing is similar to a fix-time signal crossing, but with more average delays for both pedestrians and vehicles. This is because a Puffin crossing focuses more on pedestrian safety and the all-red period is often set to be longer for pedestrians to finish their crossing actions at the end of any pedestrian green time.

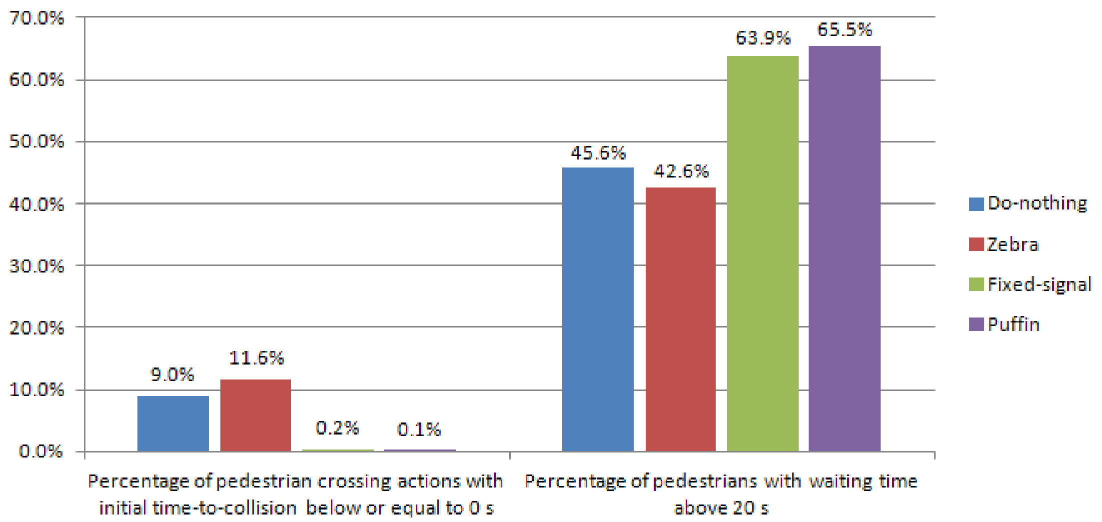


Figure 8.12 Comparison of safety of different treatments in a typical downtown area

For pedestrian safety, a Zebra crossing has a similar performance as the do-nothing scenario, with a slight increase in the probability of risky behaviour because of the increased likelihood of forming large group at such a location, which makes pedestrians feel more confident to accept smaller gaps in the traffic.

The introduction of signalised crossings can almost eliminate all risky crossing behaviour and significantly improve pedestrian safety as they separate the two modes temporally. However, compared to the base scenario, a drawback is that when the traffic is heavy they may result in a moderate increase (40.1%-43.6%) of the percentage of pedestrians with waiting time above 20 s, which may result in risky behaviour such as violating the signal.

When both the vehicle traffic demand and pedestrian crossing demand are high, a Puffin crossing functions like a fixed-time signal crossing and thus the safety performances of the two types of crossings are similar.

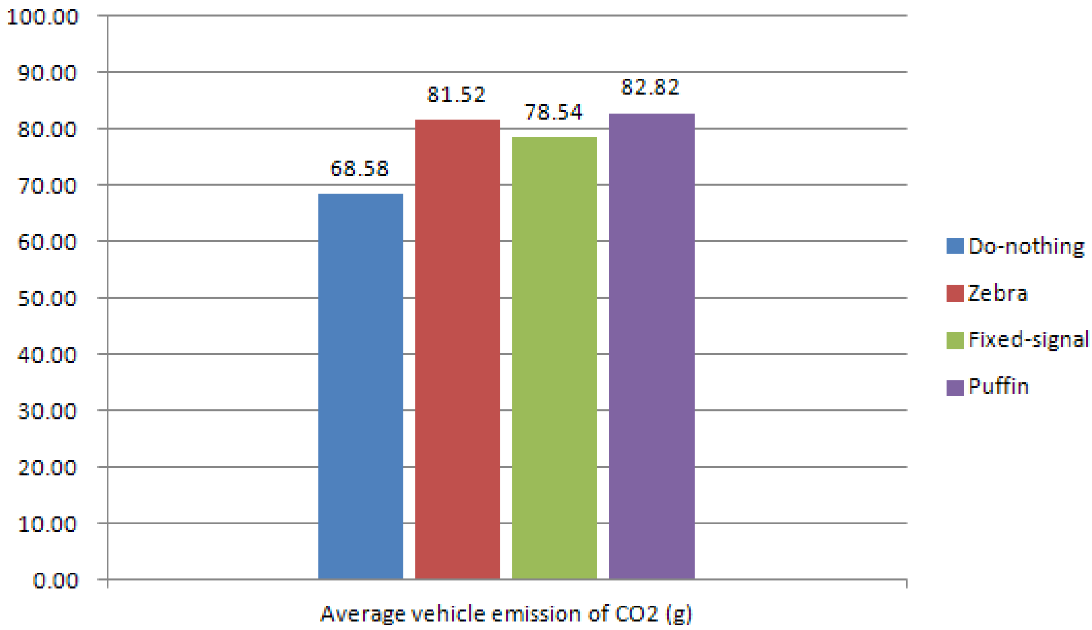


Figure 8.13 Comparison of CO_2 emission of different treatments in a typical downtown area

The CO_2 emissions of the four treatments are all relatively high, with slightly higher values when crossing facilities are introduced. This indicates that in downtown areas where both of the vehicle and pedestrian demand are high, the vehicle traffic has already been experiencing substantial fluctuations due to pedestrians' jaywalking behaviour in large groups and thus in a worse condition in terms of CO_2 emission. The introduction of pedestrian crossings is not like to bring much change to the level of CO_2 emission.

8.4.2 Scenario 2: suburban area

8.4.2.1 Scenario description

The characteristics of a typical road section in suburban area in Beijing, China are described in Table 8.5. Usually, at such locations, the vehicle traffic demand is moderate and pedestrian crossing demand is low.

Table 8.5 Characteristics of a typical road section in downtown area in China

Geometry	2-way-2-lane road section
Length	300 m
Width of lanes	vehicle lane = 3.5 m, cycle lane = 3.5 m, pavement = 5.0 m
Width of road median	0.3 m
Speed limit	30 km/h
Vehicle composition	85% LV, 12% MCV, 1% HCV, 1% BCR and 1% BCA
Vehicle desired speed mean	9.15 m/s
Pedestrian composition	41% YM, 38% YF, 11% OM, and 10% OF
Vehicle traffic demand	500 veh/h for each vehicle lane
Pedestrian crossing demand (2-way)	150 ped/h, with the same temporal and spatial distribution specified in Table 8.3

Similar to the discussion in Section 8.4.1.1, the performances of the following four treatments are proposed to be compared.

<u>1. Do-nothing:</u>

This is the based scenario; no pedestrian crossing is established in the scenario.

<u>2. Zebra crossing:</u>

In this treatment, a Zebra crossing is placed in the middle of the road section.

<u>3. Fixed-time signal crossing:</u>

In this treatment, a fixed-time signal crossing is placed in the middle of the road section. The signal timing is determined based on the principle discussed in Section 8.4.1.1, the signal timing plan is shown in Figure 8.14.

Period	P1 (14 s)	P2 (3 s)	P3 (2 s)	P4 (16 s)	P5 (5 s)
Veh. signal	G	A	R	R	R
Ped. signal	R	R	R	G	R

Figure 8.14 Signal timing for a fixed-time crossing at a typical suburban area

4. <u>Puffin crossing:</u>

In this treatment, a Puffin signal crossing is placed in the middle of the road section. The signal timing is determined on the basis of the above fixed-time signal, considering default parameters suggested by Walker et al (2005). The signal timing of the Puffin crossing is illustrated in Figure 8.15.

Period	P1 (min=7 s, extendable)	P2 (3 s)	P3 (2 s)	P4 (16 s)	P5 (2 s)	P6 (max=11 s, may gap off)	P7 (2 s)
Veh. signal	G	A	A	R	R	R	R & A
Ped. signal	R	R	R	R	G	R	R

Figure 8.15 Signal timing for a Puffin crossing at a typical suburban area

8.4.2.2 Comparison of different treatments

The multi-criteria comparison of the operational performances of different treatments is illustrated in Figure 8.16, Figure 8.17 and Figure 8.18.

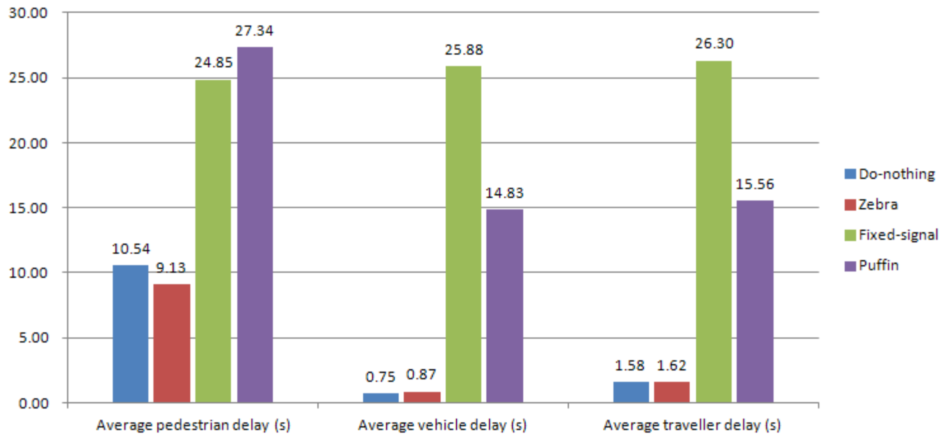


Figure 8.16 Efficiency comparison of different treatments in a typical suburban area

Overall, unsignalised treatments outperform signalised treatments in terms of average delays. For the base scenario, there is almost no delay for vehicle traffic as the vehicle traffic has priority on the road and the number of pedestrians is not sufficiently large to pose impact to vehicle traffic. This indicates that pedestrians and vehicles can well negotiate their right-of-way when both the vehicle and pedestrian demand are in lower levels. In this situation, it is not necessary to introduce signalised treatments, which can result in significant delays for both modes. However, if the safety records of the site suggest that the two modes should be separated, a Puffin crossing should be a much better choice than a fixed-time signal crossing, as the right-of-way for vehicular traffic can be resumed as soon as the crossing facility is clear, effectively avoiding unnecessary delays for vehicles.

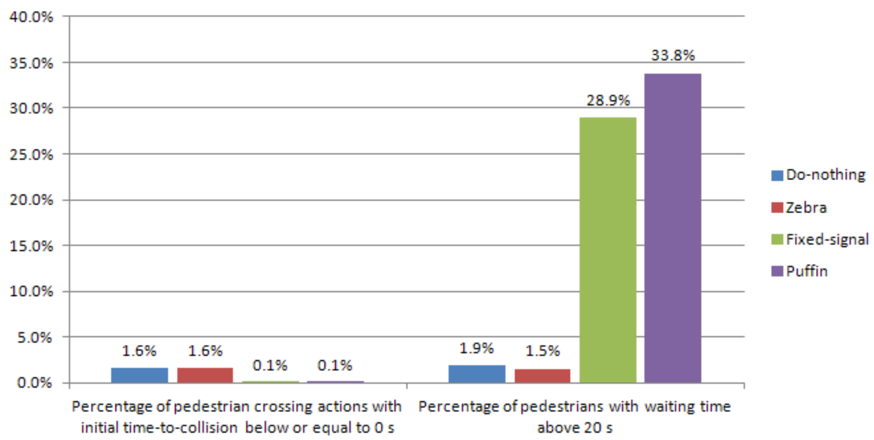


Figure 8.17 Comparison of safety of different treatments in a typical suburban area

For safety, as pedestrians and vehicles can negotiate well when both their demand are in lower levels, the effects of improvement from the signalised crossings are not significant compared in downtown areas. On the contrary, the introduction of signalised crossings can substantially increase the possibility that a pedestrian has to wait a longer time to finish crossing the road, resulting that some pedestrians may exert risky behaviour when their waiting time is too long. This drawback of signalised crossings may offset some advantages they bring.

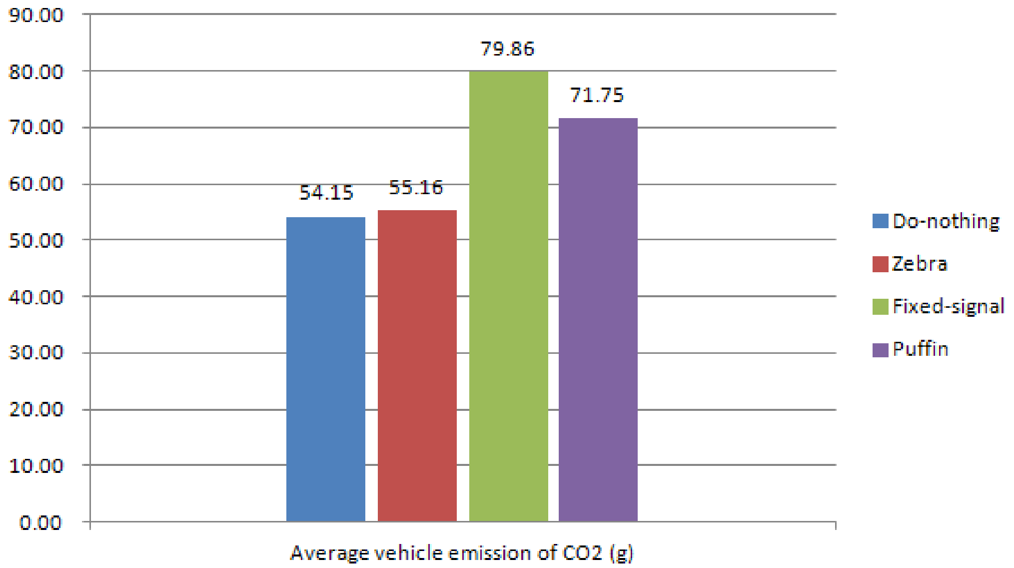


Figure 8.18 Comparison of CO_2 emissions of different treatments in a typical suburban area

In terms of CO_2 emission, unsignalised treatments have better performances than signalised treatments. This can also be explained by the fact that the vehicle traffic can hardly be influenced by pedestrian crossing behaviour and can stay in an optimum smooth status when pedestrian crossing demand is low. When signalised treatments are introduced, there are more stop-and-go phenomena in the traffic, which can result in more emission.

8.4.3 Summary and recommendation

Two typical scenarios representing downtown and suburban areas were chosen to compare the multi-criteria performances of different treatments, including do-nothing, Zebra crossing, fixed-time signal crossing and Puffin crossing. Overall, Puffin crossing is likely to have a high performance and the advantage is more significant when the pedestrian demand is not high. In peak hours in downtown areas where the demand of both vehicles and pedestrians are high, the Puffin crossing functions like a fixed-time signal crossing and its advantage cannot be fully exploited. Therefore, compared to the high costs in construction, maintenance and training for the public of the Puffin crossing, the traditional fixed-time signal crossings still have their position at such locations if there are other priorities with the limited budget.

In suburban areas where the vehicle traffic and pedestrian crossing demand are relatively low, the two modes can well negotiate their right-of-way. Overall, unsignalised treatments outperform signalised treatments. Generally, it is not recommended to introduce signalised crossings in this situation. However, if the safety records of the site guarantee a signalised treatment, a Puffin crossing will have a much better performance than a fixed-time signal crossing.

It can also be noted that the effect of improvement with a Zebra crossings was minor in both scenarios. The safety indicators are slightly improved not because of the pedestrian priority this crossing should provide but due to the fact that at such locations pedestrians are more like to gather in a group, which makes some of them feel safer and sometimes commit risky behaviour such as accepting smaller gaps and forcing the vehicle traffic come to a stop. In such a case, people waiting nearby are likely to make use of the forced gap in the traffic flow to cross the road, resulting in a longer initial time-to-collision. Therefore, the use of Zebra crossing should be considered more carefully as the priority at such places is ambiguous and thus can result in potential safety problems.

8.5 Conclusion

This chapter described the application of the pedestrian-vehicle interaction model to derive new understandings from a multi-criteria evaluation study of a pure unsignalised area and a comparison study of different treatments regarding the interaction of the two modes at some specific locations. Firstly, a hypothetical uncontrolled road section with scenarios settings similar to reality was used to evaluate the system performances, in terms of efficiency, safety and environmental impact, under different combinations of vehicle traffic and pedestrian crossing demand. It was found that the vehicle traffic demand had major influence on the overall system performance. The system efficiency, safety and environmental impact began to worsen and pedestrians were more likely to have unsafe experience and exert risky behaviour when vehicle traffic demand (2-way) exceeded 1400-1600 veh/h, where appropriate treatments were needed. Secondly, two typical scenarios were chosen to compare the multi-criteria performances of different treatments, including do-nothing, Zebra crossing, fixed-time signal crossing and Puffin crossing. One scenario was with both higher vehicle traffic and pedestrian crossing demand, representing most cases in downtown areas and the other was with moderate vehicle traffic demand and lower pedestrian crossing demand, standing for most cases in sub-urban areas. It was found that the Puffin crossing was likely to have a high overall performance and the advantage was more significant when the pedestrian demand was not high. Unsignalised treatments outperformed signalised treatments when both the vehicle traffic and pedestrian crossing demand were relatively low. Also, the improvement that a Zebra crossing brought was minor. The use of Zebra crossings should be considered more carefully as the priority at such places is ambiguous and thus can result in potential safety problems.

CHAPTER 9
CONCLUSIONS

9.1 Introduction

This chapter summarises the activities conducted in this research, revisits the objectives raised at the beginning of this dissertation, indentifies main contributions and discusses directions of potential work in the future.

9.2 Summary of the research

The research described in this dissertation developed a microscopic simulation model to study pedestrian-vehicle interaction behaviour in a range of circumstances. Key contributions relate to the development of the model, an appreciation of the value of the approach and new understandings of pedestrian-vehicle interactions. The first parts of the dissertation provided a review of general issues on the importance of walking transport mode, main barriers to pedestrians existing engineering treatments regarding pedestrian-vehicle interactions and common guidelines and tools for the assessment of various solutions. Significant gaps between the capabilities of existing models and the practical needs were identified when conducting evaluation and comparison studies of different treatments in micro-simulation environments. The review also provided a content for the methodology, especially the data collection methods employed in this research.

A series of studies to measure behaviour based on the data collected in Beijing, China have been detailed. Intra vehicle and pedestrian behaviour models were developed, calibrated and validated separately, incorporating the best available understandings from existing published studies and in accordance with the specific local data. The two modes were integrated by interpreting new findings from the study of microscopic interaction behaviour of the two modes. The complete model was validated against field data independent of those used in model development, covering a number of typical interaction scenarios, including both unsignalised and signalised situations.

The developed and validated model has been applied to study a typical unsignalised location by analysing the system performances with different combinations of vehicular traffic and pedestrian crossing demand, in terms of efficiency, safety and environmental impact. Also, the performances of different treatments including no-control, Zebra crossing, fixed-time signal crossing and Puffin crossing at two typical locations have been compared. Recommendations and interpretations have been given at the end of each application.

9.3 Main findings

9.3.1 Research Objective 1

To develop new understandings of the microscopic interaction behaviour between pedestrians and vehicles in the urban street environment in China

A video camera based data collection method was employed to collect the dynamics of pedestrians and concurrent vehicle traffic during their interaction process. The raw video data was then quantified and extracted in the laboratory with the help of video processing software. A substantial data base was established, from which the microscopic behaviour of the two modes during the interaction process was determined. It should be noted that due to the limitations of existing data collection techniques, the collected data could not cover every aspect of the interaction process and some assumptions were made based on the findings of previous studies. Experiments were carried out to provide support for assumptions made in behaviour studies.

The pedestrian-vehicle interaction process was abstracted into the following five modules: pedestrian gap acceptance, pedestrian approaching vehicle lanes, pedestrian on-road movement, pedestrian departing from vehicle lanes and vehicle reactions to pedestrians.

For pedestrian gap acceptance, behaviour at locations with no-control and with a Zebra crossing were studied. For a typical location with no-control, the pedestrian's probability of accepting a gap in the traffic was found to be able to be predicted with a binary logit model. Three factors including the nearside traffic time gap, pedestrian's age type and the number of people in a crossing group were identified to be sufficiently significant to be included in such a model. For a typical location with a Zebra crossing, similar result was achieved but with slightly different model parameters, showing that the probability of accepting smaller gaps was slightly higher. This could be interpreted by the following two reasons. First, in China, pedestrians were not given full priority at Zebra crossings, and motorists hardly yielded to pedestrians even when they had already established their precedence on the crossing; therefore, the operations of such crossing facilities were more similar to pure unsignalised scenarios in that pedestrians had to wait for proper gaps in the traffic to cross the street. Second, Zebra crossings were usually built at places where pedestrians are more likely to form large groups, in which they probably felt "protected by others", "confident" or "safer" and therefore their behaviour seemed to be somewhat more aggressive than at pure unsignalised locations where they seemed to be more cautious when crossing.

Pedestrian approaching vehicle lanes, on-road movement and departing from vehicle lanes concerned pedestrian's path finding and speed choice. Path finding behaviour related to the location where a pedestrian decided to cross the road and whether the pedestrian would use a nearby pedestrian crossing if available. The behaviour was described based on the assumption that pedestrians were likely to organise their crossing location to minimise their delay and whilst also considering safety.

It was extremely difficult to collect data on the microscopic decisions of individual drivers' reactions to pedestrians with current equipment. To describe this process, an assumption analogous to the concept of collision avoidance in the car-following model was used, which was that for any vehicle at its reaction time, there existed a maximum acceleration it could apply due to the conflict with the pedestrian ahead, under which the vehicle was still able to come to a full stop with a comfortable deceleration rate, assuming the pedestrian suddenly stopped on that vehicle lane. These assumptions proved rational in the further model validation process.

9.3.2 Research Objective 2

To develop a micro-simulation model that can fully simulate the interaction process between the two modes for both signalised and unsignalised scenarios, incorporating the understandings from the new behaviour study and best available research findings published previously

The intra vehicle model was developed initially on the basis of a review of existing common models of car-following behaviour. The concepts of the fuzzy logic model in FLOWSIM, a micro-simulation tool developed at Transportation Research Group, University of Southampton, were employed in this research and series of parameters were calibrated according to the local situation in Beijing, China. The vehicle model was validated in terms of saturation flow and average journey time against field data independent of those used for model development.

The intra pedestrian model was then developed using discrete choice model approaches suggested by Wakim et al (2004) and Antonini et al (2006), with some modification to enable it to be integrated with vehicle models. The pedestrian model was validated in terms of pedestrian crossing capacity. Results showed that the model could generate a reasonable level of pedestrian flow at the bottleneck of pedestrian crossing facility.

Next, the two modes were integrated in the interaction model, which was developed based on the output of the behavioural study proposed in the first objective. The whole model was then validated in terms of both individual and average vehicle/pedestrian journey times, in three typical scenarios including no control, Zebra crossing and signalised crossing, to ensure the model was credible to conduct an evaluation study at unsignalised locations or to carry out comparison studies between unsignalised and signalised locations. Further, the outputs of the model were linked with a commonly used instantaneous vehicle emission model to generate second-by-second emission data resulted from the vehicle dynamics output by the interaction model, in order to provide an indicator for environmental impact. It should be noted that the validity of such an emission model itself was highly related to the characteristics and compositions of vehicles according to the local situation and this emission model was only validated in some European countries. Therefore, the absolute values generated by such a model were not necessarily accurate when it was used in China. However, the relative differences generated by the model could still be used as a reference to reflect the environmental impact in each scenario.

Last, it should also be noted that all the models were implemented with C++, a highly object oriented computer programming language. All the necessary algorithms were designed by the author and were described with equations or pseudo-codes throughout this thesis to facilitate subsequent researcher to make use of the model. Main source codes are attached in Appendix II of this dissertation.

9.3.3 Research Objective 3

To apply the model to derive new understandings from a multi-criteria evaluation study of a pure unsignalised area and a comparison study of different treatments regarding pedestrian-vehicle interactions at some specific locations

For the first application, a hypothetical road section but with scenarios settings similar to reality was used to evaluate the system performances with no control and under different combinations of vehicle traffic and pedestrian crossing demand. The vehicle traffic capacity, average vehicle delay, average pedestrian delay and average traveller delay (considering vehicle occupancy) were selected as indictors for system efficiency; the percentage of pedestrians' waiting time above 20s and the percentage of pedestrians' initial time-to-collision (defined in Chapter 8) below or equal to 0s were selected as surrogate measures to reflect safety; the vehicle emission of CO_2 was used to reflect environmental impact. The following results were obtained through micro-simulation study using the developed model.

It was found that the road capacity decreased as the pedestrian crossing demand increased and the capacity drop was more significant when the pedestrian crossing demand (2-way) exceeded 1200 ped/h. The level of vehicle traffic demand had a major influence on the average vehicle, pedestrian and traveller delays. Major delays of the three aspects happened when traffic demand (2-way) exceeded approximately 1400-1600 veh/h. Especially, when pedestrian crossing demand (2-way) exceeded 1200 ped/h, all the three indicators were considerably high. Meanwhile, the vehicle emissions of CO_2 were also high in this situation due to frequent unstable flow resulted from the stop-and-go phenomena.

For pedestrian safety, longer pedestrian waiting times and smaller time-to-collision values happened more frequently when vehicle traffic demand (2-way) exceeded 1500 veh/h with slight fluctuations for each of the vehicle demand above this level. It was more dangerous when vehicle flow was high and pedestrian flow was low as there were fewer proper gaps for pedestrians to utilise and meanwhile pedestrians were less likely to make use of the grouping advantage to force the traffic to a stop.

In conclusion, the vehicle traffic demand had major influence on the overall system performance. The system efficiency, safety and environmental impact began to worsen when vehicle traffic demand (2-way) exceeded 1400-1600 veh/h; in this situation, pedestrians were more likely to have unsafe experience and exert risky behaviour. Appropriate treatments were needed in this situation.

For the second application, two typical scenarios were chosen to compare the multi-criteria performances of different treatments, including do-nothing, Zebra crossing, fixed-time signal crossing and Puffin crossing. One scenario was with both higher vehicle traffic and pedestrian crossing demand, standing for most cases in downtown areas and the other was with moderate vehicle traffic demand and lower pedestrian crossing demand, standing for most cases in sub-urban areas. It was found that the Puffin crossing was likely to have a high overall performance and the advantage was more significant when the pedestrian demand was not high. In peak hours in downtown areas where the demand of both modes were high, the Puffin crossing acted like fixed-time signal crossing and its advantage was not fully exploited. Therefore, compared to the high costs in construction, maintenance and training for the public of the Puffin crossing, the traditional fixed-time signal crossings still have their position at such locations if there are other priorities with the limited budget.

In suburban areas where both the vehicle traffic and pedestrian crossing demand were relatively low, the two modes could well negotiate their right-of-way. Overall, unsignalised treatments outperformed signalised treatments. Generally, it is not recommended to introduce signalised crossings in this situation. However, if the safety records of the site guarantee a signalised treatment, a Puffin crossing will have a much better performance than a fixed-time signal crossing.

It was also found that the improving effect of Zebra crossings was minor. The safety indicators were slightly improved not because of pedestrian priority this crossing should provide but due to the fact that at such a place pedestrians were likely to gather in groups, resulting that some may feel safer and sometimes commit risky behaviour such as accepting smaller gaps or forcing the vehicle traffic come to a stop. In such cases, the rest of people waiting nearby were likely to make use of the forced gap in the traffic flow to cross the road, resulting in a longer initial time-to-collision.

9.4 Main contributions

The contributions of this research can be summarised from the following aspects.

9.4.1 A substantial database

This research developed a data collection and extraction methodology for researchers to measure a range of variables describing the interaction of pedestrians and vehicles, a substantial data base of the pedestrian-vehicle interaction behaviour in China was established. For data from on-road survey, the data collection methodology proved feasible and non-intrusive. The author was able to obtain a very large sample size of data of interests from a vast amount of raw video recorded during the data collection process in this research. Table 9.1 provides a summary of main data collected in this research. These data are well achieved in the author’s research library for future use.

Table 9.1 A summary table of main data collected in this research

Raw data from field survey				Analysed and quantified data	
Content	**Amount**	**Time and Place**	**Purpose**	**Structure**	**Sample Size**
Video record of random on-road vehicles	1 hour of video record	15 September 2008, Jiaoda East Road, Beijing	To calibrate the distributions of length, width and margin of different types of on-road vehicle in Beijing	Two-dimensional table with each row representing a vehicle and each column representing a combination of (ID, type, length, width and margin)	289 in total, as shown in Table 4.2
Video record of queuing vehicles departing from a signalised crossing, as shown in Figure 4.7	2 hours of video record	16:00-18:00, 29 September 2008, Xueyuan South Road, Beijing	To validate the intra-vehicle model in terms of saturation flow	Saturation flow data, as shown in Table 4.11	15
Video record of vehicles passing through a road section, as shown in Figure 4.8	1 hour of video record	16:30-17:30, 30 September 2008, Xueyuan South Road, Beijing	To validate the intra-vehicle model in terms of journey time	Individual vehicle journey times passing through the surveying road section	1198
Video record of pedestrians walking along a marked and designated path in designed experiments, as shown in Figure 5.7	Several hours of video record	1-10 October 2008, inside Beijing Jiaotong University, Beijing	To calibrate pedestrians' desired and maximum walking speeds in Beijing	Two-dimensional table with each row representing a pedestrian and each column representing a combination of (ID, type of age, gender, desired walking speed and maximum walking speed)	400 in total, as shown in Table 5.1 and Table 5.2

Table 9.1 A summary table of main data collected in this research (continued)

Raw data from field survey				Analysed and quantified data	
Content	**Amount**	**Time and Place**	**Purpose**	**Structure**	**Sample Size**
Video record of pedestrian-vehicle interaction behaviour in a road section with no-control, as shown in Figure 6.1	1 hour of video record	11:30-12:30, 10 September 2008, Jiaoda East Road, Beijing	To inspire a framework in a bottom-up way to model the interaction behaviour of the pedestrians and vehicles	Typical patterns of pedestrian moving trajectories, as shown in Figure 6.2	100 of many were used for analysis of typical patterns of moving trajectories
Video record of pedestrian-vehicle interaction behaviour in a road section with no-control, as shown in Figure 6.6	20 hours of video record	11:30-12:30 and 16:30-17:30, 10-14 September 2008, Jiaoda East Raod, Beijing	To develop the pedestrian gap acceptance model at a typical location with no-control	Two-dimensional table with each row representing a pedestrian's decision case and each column representing a combination of (case ID, pedestrian gap acceptance decision, pedestrian's type of age, gender, accumulative waiting time, the number of pedestrians in the crossing group, the time gap in the nearside vehicle lane and the time gap in the far-side vehicle lane), as discussed in Section 6.3.1	600 near-side gap acceptance decision cases, of which 70% for coefficients estimation and 30% for model validation for nearside gap acceptance model; 500 far-side gap acceptance to validate coefficients for far-side gap acceptance model, as described in Section 6.3.1
Video record of pedestrian-vehicle interaction behaviour in a road section with a Zebra crossing	20 hours of video record	11:30-12:30 and 16:30-17:30, 13-26 October 2008, Jianhua South Road, Greater Beijing Area, China	To calibrate the parameters for the near-side pedestrian gap acceptance model at a typical location with a Zebra crossing	Two-dimensional table with each row representing a pedestrian's decision case and each column representing a combination of (case ID, pedestrian gap acceptance decision, pedestrian's type of age, gender, accumulative waiting time, the number of pedestrians in the crossing group, the time gap in the nearside vehicle lane and the time gap in the far-side vehicle lane), as discussed in Section 6.3.1	500 near-side gap acceptance decision cases, of which 70% for coefficients estimation and 30% for model validation for nearside gap acceptance model; 500 far-side gap acceptance to validate coefficients for far-side gap acceptance model, as described in 6.3.2

Table 9.1 A summary table of main data collected in this research (continued)

Raw data from field survey				Analysed and quantified data	
Content	**Amount**	**Time and Place**	**Purpose**	**Structure**	**Sample Size**
Video data of pedestrian on-road movement, as shown in Figure 6.12	4 hours of video record	14:00-16:00, 13-14 September 2008, Jiaoda East Road, Beijing	To calibrate θ_F, a parameter in pedestrian on-road movement model	Two-dimensional table with each row representing a pedestrian and each column representing a combination of (ID, type of age, gender and θ_F)	400 in total, as shown in Table 6.7
Video data of subject pedestrians' crossing decisions at the edge of the nearside pavement, as shown in Figure 6.13	Several hours of video record	1-10 October 2008, Jiaoda East Road, Beijing	To calibrate t_M, a parameter in pedestrian on-road movement model	Two-dimensional table with each row representing a pedestrian and each column representing a combination of (ID, type of age, gender and t_M)	200 in total, as shown in Table 6.8
Video data of pedestrians just leave the last vehicle lane to the time when he/she just reaches his/her destination, as shown in Figure 6.14	4 hours of video record	14:00-16:00, 13-14 September 2008, Jiaoda East Road, Beijing	To calibrate φ_F, a parameter in pedestrian on-road movement model	Two-dimensional table with each row representing a pedestrian and each column representing a combination of (ID, type of age, gender and φ_F)	400 in total, as shown in Table 6.9
Video data of pedestrian and vehicle interaction in a typical location with no-control, as shown in Figure 7.2	1 hour of video record	12:00-13:00, 15 September 2008, Jiaoda East Road, Beijing	To validate the complete simulation model for no-control scenarios	Individual vehicle journey times passing the road section and individual pedestrian journey times crossing this road, as shown in Figure 7.3 and Figure 7.4	1771 for vehicle and 969 for pedestrian, as shown in Figure 7.3 and Figure 7.4

Table 9.1 A summary table of main data collected in this research (continued)

Raw data from field survey				Analysed and quantified data	
Content	**Amount**	**Time and Place**	**Purpose**	**Structure**	**Sample Size**
The same as above	13 hours of video record	12:00-13:00, 16-28 September 2008, Jiaoda East Road, Beijing	Sensitivity analysis and error checking for the complete simulation model for no-control scenarios	Vehicle and pedestrian crossing demands, as well as average vehicle and pedestrian journey times for each day, as shown in Table 7.5	13 for vehicle and pedestrian average times respectively, as shown in Table 7.5
Video data of pedestrian and vehicle interaction in a typical location with a Zebra crossing, as shown in Figure 7.7	1 hour of video record	16:30-17:30, 27 October 2008, Jianhua South Road, Greater Beijing Area	To validate the complete simulation model for Zebra crossing scenarios	Individual vehicle journey times passing the road section and individual pedestrian journey times crossing this road, as shown in Figure 7.8 and Figure 7.9	1577 for vehicle and 936 for pedestrian, as shown in Figure 7.8 and Figure 7.9
The same as above	13 hours of video record	16:30-17:30, 28 October – 9 November 2008, Jianhua South Road, Greater Beijing Area	Sensitivity analysis and error checking for the complete simulation model for Zebra crossing scenarios	Vehicle and pedestrian crossing demands, as well as average vehicle and pedestrian journey times for each day, as shown in Table 7.12	13 for vehicle and pedestrian average times respectively, as shown in Table 7.12
Video data of pedestrian and vehicle interaction in a typical location with a signalised crossing, as shown in Figure 7.12	1 hour of video record	7:30-8:30, 13 October 2008, Weigongcun Road, Beijing	To validate the complete simulation model for signalised crossing scenarios	Individual vehicle journey times passing the road section and individual pedestrian journey times crossing this road, as shown in Figure 7.14 and Figure 7.15	1385 for vehicle and 1167 for pedestrian, as shown in Figure 7.14 and Figure 7.15
The same as above	13 hours of video record	7:30-8:30, 14-26 October 2008, Weigongcun Road, Beijing	Sensitivity analysis and error checking for the complete simulation model for signalised crossing scenarios	Vehicle and pedestrian crossing demands, as well as average vehicle and pedestrian journey times for each day, as shown in Table 7.19	13 for vehicle and pedestrian average times respectively, as shown in Table 7.19

9.4.2 Behavioural insights

The following main understandings have been obtained through the study of microscopic behaviour of pedestrians, vehicles and their interactions.

First, the interaction model developed in this research introduced the concept of pedestrian crossing *O-D* in the pedestrian generation model, enabling the analysis of pedestrians' route choice behaviour when they cross a road section, which was often over-simplified in existing micro-simulation tools. This is crucial because the route choice behaviour is a link between the intra pedestrian behaviour on pavements and the gap acceptance behaviour on the edge of vehicle lanes, and has impact on pedestrians' overall journey times. The pedestrians' route choice behaviour were categorised into 3 types including approaching vehicle lanes, on-road movement and departing from vehicle lanes, according to their positions to vehicle lanes. It was found that the path finding behaviour related to pedestrian's crossing origin and destination, as well as the relative position of the nearby crossing facility (if there is one) to the pedestrian's *O-D* pair. Pedestrians were likely to organise their crossing location to minimise their delay and whilst also considering safety. For signalised crossings, if the use of signalised crossing did not involve detouring, pedestrians were likely to choose to cross at the nearby signalised crossing. Otherwise, sometimes they might commit risky behaviour such as jaywalking if the perceived decrease in efficiency overweighed the demand for safety. For Zebra crossings, pedestrians were likely to use such a facility only when it did not involve detouring. Otherwise, if an appropriate expected gap appeared on the nearside lane, he/she was likely to accept it and moved towards the vehicle lane instead of heading to the Zebra crossing. This fact implies that Zebra crossings have less attraction to pedestrians in China. Care should be taken when planning a Zebra crossing as pedestrians may not use such a facility as expected. One suggestion is that the location of the proposed Zebra crossing should be carefully chosen in order that most of pedestrian crossing activities will not involve detouring when using such a crossing.

Second, pedestrian gap acceptance was analysed in both pure unsignalised and Zebra crossing scenarios. For pure unsignalised scenarios, three factors including nearside time gap, pedestrian's age type and number of pedestrians in a crossing group were identified most influential to pedestrian gap acceptance decision. One interesting fact was that the expected far-side time gap had no significant effect on such a decision process, meaning that pedestrians were likely to cross the road lane by lane to minimise delay instead of waiting until all lanes were cleared. Data analysis showed that pedestrians hardly paid attention to the far-side incoming vehicles when they started to cross the first lane. Many pedestrians crossed the road regardless of the expected far-side time gap, even when the expected far-side time gap was too small to be utilised, resulting in that such pedestrians stopped and waited at the median of the road for next possible gaps in the next lane to continue the crossing behaviour. Another finding was that older pedestrians were more cautious and waited for longer nearside gaps to start the crossing. Therefore, higher proportion of older pedestrians could lead to more pedestrian delays when there were fewer available gaps on the nearside vehicle lane during busy hours.

Third, from the perspective of drivers, it was found that the sole car-following could not sufficiently describe drivers' behaviour when there were frequent pedestrian jaywalking actions. The drivers' decisions of accelerations were also restricted to the conflicting pedestrian in addition to leading vehicles under mixed traffic conditions. There existed a maximum acceleration under which a driver would not collide with the conflicting pedestrian at his/her next reaction time. This indicated that although vehicles had priorities on roads, they were prone to jaywalking pedestrians due to lack of traffic disciplines. Data showed that as the size of a crossing group increased, pedestrians became more aggressive and thus were more likely to accept smaller gaps. Sometimes they could even force the conflicting vehicles to slow down or stop, resulting a major efficiency loss for vehicular traffic. The collision avoidance model was applied to mimic such friction effect of jaywalking pedestrians to vehicular traffic.

In summary, the interpretation from behavioural studies could be used as basis for developing similar simulation models in the future. The development, calibration and validation of behaviour models related to the pedestrian-vehicle interaction process have been detailed to allow subsequent researchers to make use of those models. The complete micro-simulation program was implemented with C++, a commonly used computer programming language. The simulation program can be integrated with existing simulation tools such as FLOWSIM to form a multimodal simulation platform to study mixed traffic, or can be applied as a standalone tool for evaluation and comparison studies of various pedestrian related treatments.

9.4.3 Policy implications

Policy implications were obtained from simulation studies using the developed model, including a multi-criteria evaluation study at a pure unsignalised area and a comparison study of different treatments regarding pedestrian-vehicle interactions at some specific locations.

For the first application, a typical uncontrolled road section was used to evaluate the system performances for situations where there was no segregation between pedestrians and vehicles, under different combinations of vehicle traffic and pedestrian crossing demand. It was found that the road capacity decreased as the pedestrian crossing demand increased. The drop of the capacity was more significant when the pedestrian crossing demand (2-way) exceeded 1200 ped/h. The level of vehicle traffic demand had a major influence on both vehicle and pedestrian averge delays, as well as overall average traveller delay. Major delays of the three aspects happened when the vehicle traffic demand (2-way) surpassed a level of approximately 1400-1600 veh/h. In this situation, if the pedestrian crossing demand (2-way) also exceeded a certain level (1200 ped/h, approximately), all the three indicators were considerably high. Meanwhile, the vehicle CO_2 emission was also high due to frequent unstable flow resulted from the stop-and-go phenomena.

For pedestrian safety, longer pedestrian waiting times and smaller time-to-collision values happened more frequently when vehicle traffic demand (2-way) exceeded approximately 1500 veh/h and with slight fluctuations for any vehicle traffic demand above this level. It was more dangerous when vehicle flow was high and pedestrian flow was low as there were fewer proper gaps for pedestrians to utilise and meanwhile pedestrians were less likely to make use of the grouping advantage to force the traffic to a stop.

To conclude, the vehicle traffic demand has a major influence on the overall system performance in typical minor streets (generally with a speed limit of 30 km/h and acting like shared spaces for vehicles and pedestrians) in Beijing, China. The system efficiency, safety and environmental impact began to worsen significantly when vehicle traffic demand (2-way) exceeded a level of approximately 1400-1600 veh/h, where pedestrians were more likely to have unsafe experience and exert risky behaviour. In a short-term, appropriate treatments such as temporal or spatial segregation between the two modes were suggested. In a long-term, it is also suggested to regulate the road users' behaviour by legislative and administrative methods. The efficiency, safety and environmental impact of road traffic in China can be improved substantially if the frictions in the mixed traffic can be minimised.

For the second application, two typical scenarios were chosen to compare the multi-criteria performances of different treatments, including do-nothing, Zebra crossing, fixed-time signal crossing and Puffin crossing. One scenario was with both higher vehicle traffic and pedestrian crossing demand, representing most cases in downtown areas and the other was with moderate vehicle traffic demand and lower pedestrian crossing demand, standing for most cases in sub-urban areas. It was found that the Puffin crossing was likely to have a high overall performance and the supiorority was more significant if the concurrent pedestrian demand was not high. In peak hours in downtown areas where the demand of both modes were high, the Puffin crossing acted like fixed-time signal crossing and its advantage was not fully exploited. Therefore, compared to the high costs of construction and maintenance for Puffin crossing, as well as training for the public to adapt the use of such a new crossing, the traditional fixed-time signal crossings still have their positions at such locations in a short-term, if there are other priorities under a limited budget.

In suburban areas where both the vehicle traffic and pedestrian crossing demand were relatively low, the two modes could well negotiate their right-of-way. Overall, unsignalised treatments outperformed signalised treatments. Generally, it is not necessary to introduce signalised crossings in this situation. However, if the safety records of the site guarantee a signalised treatment, a Puffin crossing will have a much better performance than a fixed-time signalised crossing.

It was also found that the improving effect of Zebra crossings was minor in China. The safety indicators were slightly improved not because of pedestrian priority this crossing should provide but due to the fact that at such a place pedestrians were likely to gather in groups, resulting that some may feel safer and sometimes commit risky behaviour such as accepting smaller gaps or forcing the vehicle traffic come to a stop. In such cases, the rest of people waiting nearby were likely to make use of the forced gap in the traffic flow to cross the road, resulting in a longer initial time-to-collision. The use of Zebra crossings should be considered more carefully as the priority at such places is ambiguous and thus can result in potential safety problems.

Overall, the model applications showcased a multi-criteria evaluation methodology proposed for pedestrian related treatments in the micro-simulation environment. The scenarios for model applications were created to reflect the typical mixed traffic situation in China. The direct results from the simulation studies could be used as references by local authorities when planning pedestrian related treatments. In reality, the culture of transport and exact situations might differ from place to place. However, the methodology should not have much difference and could be used as a standard procedure to conduct similar micro-simulation studies in the future.

It should be noted that the behavioural study and model application were focused in China, where such a research will have fundamental value in an area of growing importance. As behaviour will vary by location and due to the cultural differences between China and other countries, it can be expected in the future that extensive data collection and calibration/validation work are required when applying this model somewhere else, especially in developed countries such as UK, where the road discipline is better maintained. However, the methodology developed in this research for data collection, model development and multi-criteria evaluation study in micro-simulation environment are transferable to other locations and offer an opportunity for other researchers to conduct behaviour studies in a detailed microscopic level.

9.5 Directions of future work

As the pedestrians' behaviour is complex in nature, it is difficult to predict their behaviour as well as to collect the microscopic trajectory data, especially mixed with other transport modes in the urban environment. Due to limitations of time and resource, this research is still at an early stage regarding to the multi-modal transport micro-simulation modelling. There are a number of improvements that could be conducted in the future.

1. The model developed in this research can be further extended to include other modes of transport such as cyclists and buses. In some areas where the bicycle flow is high or there are frequent buses arriving and departing activities, pedestrian behaviour can be influenced by these two modes and vice versa.

2. The intra pedestrian model used in this research is only validated in terms of pedestrian crossing capacity. Further validation is needed to examine the full flow-speed-density relationship on pavements. Extensive data of pedestrian walking behaviour at such locations are required for this purpose.

3. The interaction model mainly concerns the road section areas and should be further extended and validated at junctions and a large street network, where pedestrians' gap acceptance behaviour to turning vehicles can be different.

4. The data extraction from raw video pictures is achieved mainly by manual calculations, which is extremely time-consuming. There is also a need of better automatic tools for trajectory tracking and pattern recognition.

5. In this research, a crossing *O-D* pair is assigned to each pedestrian to generate crossing demand. This can only reflects part of the entire trip. For the longer future, a complete multimodal model should consider how to integrate the full trip generation model in a larger road network, not just from one side of a road to the other but from the real origin to the final destination of any pedestrian's entire journey.

APPENDIXES

Appendix I: Statistical tests

1. Test of vehicle instantaneous speed from radar speed gun:

(1) Test method: 1-sample K-S test.

(2) Test hypotheses: H_0: the vehicle instantaneous speed data from radar speed gun yield to a normal distribution; H_1: the vehicle instantaneous speed data from radar speed gun do not yield to a normal distribution.

(3) Test result: as shown in Table Appx.1.

Table Appx.1 The result of 1-sample K-S test for vehicle instantaneous speed from radar speed gun

		Speed
N		50
Normal Parameters[a,b]	Mean	29.9284
	Std. Deviation	.83239
Most Extreme Differences	Absolute	.130
	Positive	.130
	Negative	-.081
Kolmogorov-Smirnov Z		.916
Asymp. Sig. (2-tailed)		.371

a. Test distribution is Normal.

b. Calculated from data.

(4) Conclusion: there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.371). Therefore, it is reasonable to accept H_0 that the vehicle instantaneous speed data from radar speed gun yield to a normal distribution.

2. Test of vehicle instantaneous speed from video survey:

(1) Test method: 1-sample K-S test.

(2) Test hypotheses: H_0: the vehicle instantaneous speed data from video survey yield to a normal distribution; H_1: the vehicle instantaneous speed data from video survey do not yield to a normal distribution.

(3) Test result: as shown in Table Appx.2.

Table Appx.2 The result of 1-sample K-S test for vehicle instantaneous speed from video survey

		Speed
N		50
Normal Parameters[a,b]	Mean	29.9930
	Std. Deviation	1.80777
Most Extreme Differences	Absolute	.111
	Positive	.111
	Negative	-.094
Kolmogorov-Smirnov Z		.784
Asymp. Sig. (2-tailed)		.570

a. Test distribution is Normal.

b. Calculated from data.

(4) Conclusion: there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.570). Therefore, it is reasonable to accept H_0 that the vehicle instantaneous speed data from video survey yield to a normal distribution.

3. Test of saturation flow without pedestrian interference from field survey:

(1) Test method: 1-sample K-S test.

(2) Test hypotheses: H_0: the saturation flow data from field survey yield to a normal distribution; H_1: the saturation flow data from field survey do not yield to a normal distribution.

(3) Test result: as shown in Table Appx.3.

Table Appx.3 The result of 1-sample K-S test for saturation flow from field survey

		Saturation Flow
N		15
Normal Parameters[a,b]	Mean	1405.9887
	Std. Deviation	49.41636
Most Extreme Differences	Absolute	.120
	Positive	.104
	Negative	-.120
Kolmogorov-Smirnov Z		.464
Asymp. Sig. (2-tailed)		.982

a. Test distribution is Normal.

b. Calculated from data.

(4) Conclusion: there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.982). Therefore, it is reasonable to accept H_0 that the saturation flow data from field survey yield to a normal distribution.

4. Test of saturation flow without pedestrian interference from simulation:

(1) Test method: 1-sample K-S test.

(2) Test hypotheses: H_0: the saturation flow data from micro-simulation yield to a normal distribution; H_1: the saturation flow data from micro-simulation do not yield to a normal distribution.

(3) Test result: as shown in Table Appx.4.

Table Appx.4 The result of 1-sample K-S test for saturation flow from simulation

		Saturation Flow
N		30
Normal Parameters[a,b]	Mean	1410.4970
	Std. Deviation	14.17558
Most Extreme Differences	Absolute	.124
	Positive	.124
	Negative	-.087
Kolmogorov-Smirnov Z		.680
Asymp. Sig. (2-tailed)		.745

a. Test distribution is Normal.

b. Calculated from data.

(4) Conclusion: there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.745). Therefore, it is reasonable to accept H_0 that the saturation flow data from simulation yield to a normal distribution.

5. Test of pedestrian crossing capacity data from simulation

(1) Test method: 1-sample K-S test.

(2) Test hypotheses: H_0: the pedestrian crossing capacity data from simulation yield to a normal distribution; H_1: the pedestrian crossing capacity data from simulation do not yield to a normal distribution.

(3) Test result: as shown in Table Appx.5.

Table Appx.5 The result of 1-sample K-S test for pedestrian crossing capacity from simulation

		Ped Crossing Capacity
N		10
Normal Parameters[a,b]	Mean	2373.80
	Std. Deviation	126.992
Most Extreme Differences	Absolute	.162
	Positive	.162
	Negative	-.140
Kolmogorov-Smirnov Z		.512
Asymp. Sig. (2-tailed)		.956

a. Test distribution is Normal.

b. Calculated from data.

(4) Conclusion: there is not enough evidence to reject the null hypothesis at 95% confidence level (p=0.956). Therefore, it is reasonable to accept H_0 that the pedestrian crossing capacity data from simulation yield to a normal distribution.

Appendix II: Main source codes

Part of the source codes of PVISIM (Pedestrian-Vehicle Interaction SIMulation), the micro-simulation program developed in this research is attached in this section. The complete simulation codes are available by request.

```
#include "Global.h"
#include "Position.h"
#include "AddPedCrossing.h"
#include "ScenarioOverall.h"
#include "Edge.h"
#include <string>
#include <sstream>
#include <fstream>
#include <vector>
using namespace std;

class LaneVeh;
class Median;
class LaneCycle;
class Pavement;
class RandomSolver;
class MathSolver;
class FuzzySolver;
class CarFollowingModel_Fuzzy;
class VehicleDynamics;
class VehicleOverall;
class PedCrossing;
class SignalDynamics;
class Vehicle;
class Signal;
class PedODA;
class PedODPair;
class Pedestrian;
class PedOverall;

class CPVIView : public CScrollView
{
protected: // create from serialization only
	CPVIView();
	DECLARE_DYNCREATE(CPVIView)

// Attributes
public:
	vector<LaneVeh*>		vLaneVeh;		// To store all the LaneVeh in the simulation world
	vector<Median*>			vMedian;		// To store all the Median in the simulation world
	vector<LaneCycle*>		vLaneCycle;		// To store all the LaneCycle in the simulation world
	vector<Pavement*>		vPavement;		// To store all the Pavement in the simulation world
	vector<VehicleDynamics*>	vVehicleDynamics;	// To store all vehicles' dynamics records throughout simulation run
	vector<VehicleOverall*>		vVehicleOverall;	// to store vehicle overall stats when it quits the simulation area
	vector<SignalDynamics*>		vSignalDynamics;
	PedCrossing*			pPedCrossing;		// To store the ped crossings
	vector<PedODA*>			vPedODASouth;		// Ped ODAs in South
	vector<PedODA*>			vPedODANorth;		// Ped ODAs in North
	vector<PedODPair*>		vPedODPair;		// Ped Crossing OD Pairs
	vector<Pedestrian*>		vPedestrian;		// Store all pedestrians
	vector<PedOverall*>		vPedOverall;		// Store ped overall status
	vector<vector<Edge*> >		vvEdge;			// Store all edges
	bool				bScenarioSettings;	// Whether all the global simulation parameters have been set
	bool				bCreateLanes;		// Whether all the lanes have been created
	bool				bCreatePedCrossing;
	bool				bCreatePedODPairs;
	float				fScale;			// World distance * scale = Screen distance (Default=5.0)
	float				fSimTimeWhole;		// The whole simulation time, second
	float				fSimTimeStep;		// The simulation scanning time, second
	float				fSimTimeElapse;		// The elapsed simulation time, second
	RandomSolver*			pRandomSolver;
	MathSolver*			pMathSolver;
	FuzzySolver*			pFuzzySolver;
	int				iSimStatus;
	CarFollowingModel_Fuzzy*	pCarFollowingModel_Fuzzy;
	unsigned int			iTimerInterval;
	unsigned int			iNumVehGenerated;	// number of vehicles generated in the simulation
	unsigned int			iNumPedGenerated;	// Total number of peds gened
	DWORD				DRandomSeed;		// miliseconds since the computer starts, uses as random seeds
	int				iScenarioType;		// Scenario type
```

```
    // Simulation Status
    bool                    bSimInfo;               // whether to show SimStatus
    ScenarioOverall*        pSceOverall;

    // Parameter for ID
    Pavement*               pPaveS;
    Pavement*               pPaveN;
    LaneVeh*                pLaneVehS;
    LaneVeh*                pLaneVehN;
    LaneCycle*              pLaneCycleS;
    LaneCycle*              pLaneCycleN;
    Median*                 pMedianCentre;
    int                     iSimID;                 // the number of sim conducted
    float                   fVehDemandPercent;      // initial level
    float                   fPedDemandPercent;
    CPVIDoc*                GetDocument();

// Implementation
public:
    virtual ~CPVIView();

    // Convert a world Position to a Screen Point
    CPoint WorldToScreen(Position& worldPosition);

    // Update vehicle status
    void VehUpdate(vector<LaneVeh*>& vLaneVeh);

    // Delete vehicles out of simulation boundary
    void VehDelete(vector<LaneVeh*>& vLaneVeh);

    // Generate vehicles in this LaneVeh
    void VehGenerate(vector<LaneVeh*>& vLaneVeh);

    // Update signal status
    void SignalUpdate();

    // Record vehicle overall status when it quits simulation
    void RecordVehicleOverall(Vehicle* pVehicle);

    // Update Vehicle Acc considering not to violate Signal

    // Given vehicle initial status v0, a, calculate the new speed after t
    float GetNewSpeed(float v0, float a, float t);

    // Given vehicle initial status v0, a, calculate the travel distance after t
    float GetDistance(float v0, float a, float t);

    // Assign initial status to a new vehicle, pLaneVeh is the Lane the vehicle is in
    void AssignInitStatusToVeh(Vehicle* pVehicle, LaneVeh* pLaneVeh);

    // Update scene at any sim time
    void UpdateScene();

    // Generate pedestrians
    void PedGenerate();

    // Update pedestrian status
    void PedUpdate();

    // Delete pedestrian
    void PedDelete();

    // Assign initial status to a new ped
    void AssignInitStatusToPed(Pedestrian* pPed, PedODPair* pPedODPair);

    // Get pedestrian position type
    int GetPedPosType(Pedestrian* pPed);

    // Get a expected position type
    int GetPedPosTypeExpect(Pedestrian* pPed, Position posExpected);

    // Check if the pedestrian would use the crossing facility, to be modified

    // Record Ped overall status when it quits simulation
    void RecordPedOverall(Pedestrian* pPed);

    // Get which edge
    Edge* GetEdge(int iRow, int iCol);

    // Reset Edge
    void ResetEdge();

    // Get a ped colume id 1-10, left to right
    int GetColID(float xPed, float xStart, float fUnitLength);
```

```
	// Get a ped row id , 1, 2, 3 only; ped must be right in edge, otherwise =-1
	int GetRowID(Pedestrian* pPed);

	// Record the scenario overall status
	void RecordScenarioOverall();

	// reset the sim world and re populate the lanes and ODAs
	void ResetSimWorld();

	// Output the sim world status
	void OutputSimWorld();


#include "stdafx.h"
#include "PVI.h"
#include "PVIDoc.h"
#include "PVIView.h"
#include "LaneVeh.h"
#include "Median.h"
#include "LaneCycle.h"
#include "Pavement.h"
#include "RandomSolver.h"
#include "MathSolver.h"
#include "Position.h"
#include "Vehicle.h"
#include "VehicleDynamics.h"
#include "FuzzySolver.h"
#include "CarFollowingModel_Fuzzy.h"
#include "VehicleDynamics.h"
#include "VehicleOverall.h"
#include "PedCrossing.h"
#include "PedWaitingZone.h"
#include "Signal.h"
#include "SignalDynamics.h"
#include "PedODA.h"
#include "PedODPair.h"
#include "Pedestrian.h"
#include "PedOverall.h"

CPVIView::CPVIView()
{
	fScale			=	SCALE_DEFAULT;
	bScenarioSettings	=	false;
	bCreateLanes		=	false;
	bCreatePedCrossing	=	false;
	iSimStatus		=	SIM_STATUS_EDIT;
	bSimInfo		=	false;
	// fSimTimeWhole	=	SIM_TIME_WHOLE_DEFAULT;
	fSimTimeElapse		=	0.0f;
	pPedCrossing		=	NULL;
}

void CPVIView::OnCreateLanes()
{
	if (bScenarioSettings == true)
	{
		// 0. Create Median
		Median* pMedian = new Median();
		pMedian->iID			= 1;
		pMedian->posStart.x		= LANE_LENGTH / 2 * (-1);
		pMedian->posStart.y		= 0;
		pMedian->posEnd.x		= LANE_LENGTH / 2;
		pMedian->posEnd.y		= 0;
		pMedian->fWidth			= LANE_WIDTH_MEDIAN;
		pMedian->fLength		= pMedian->posStart.GetDistance(pMedian->posEnd);
		vMedian.push_back(pMedian);
		pMedian->fEdgeNorth		= pMedian->posStart.y + pMedian->fWidth / 2;
		pMedian->fEdgeSouth		= pMedian->posStart.y - pMedian->fWidth / 2;
		pMedianCentre			= pMedian;

		// 1. Create LaneVeh
		// 1.1 LaneVeh South
		LaneVeh* pLaneVeh		= new LaneVeh();
		pLaneVeh->iID			= 1;
		pLaneVeh->posStart.x		= LANE_LENGTH / 2 * (-1);
		pLaneVeh->posStart.y		= LANE_WIDTH_MEDIAN /2 *(-1) - LANE_WIDTH_VEHICLE /2;
		pLaneVeh->posEnd.x		= LANE_LENGTH / 2;
		pLaneVeh->posEnd.y		= LANE_WIDTH_MEDIAN /2 *(-1) - LANE_WIDTH_VEHICLE /2;
		pLaneVeh->iVehiclesGenerated	= 0;

		// The following parameters need calibration
		pLaneVeh->fWidth		= LANE_WIDTH_VEHICLE;
		pLaneVeh->fLength		= pLaneVeh->posStart.GetDistance(pLaneVeh->posEnd);
		pLaneVeh->fDemand		= (float)LANE_FLOW_SOUTH * this->fVehDemandPercent;
		pLaneVeh->fDemandInit		= (float)LANE_FLOW_SOUTH;
		pLaneVeh->pSignal		= NULL;
```

```
pLaneVeh->fEdgeNorth                = pLaneVeh->posStart.y + pLaneVeh->fWidth / 2;
pLaneVeh->fEdgeSouth                = pLaneVeh->posStart.y - pLaneVeh->fWidth / 2;
pLaneVehS                           = pLaneVeh;

// Assign initial headways to this lane
float fAccumulativeTimeHeadway      = 0.0f;
while (fAccumulativeTimeHeadway < fSimTimeWhole)
{
        float fTimeHeadway = pRandomSolver->GetInitTimeHeadwayVeh(pLaneVeh->fDemand);
        pLaneVeh->vInitialHeadways.push_back(fTimeHeadway);
        fAccumulativeTimeHeadway = fAccumulativeTimeHeadway + fTimeHeadway;
}
if (fAccumulativeTimeHeadway > fSimTimeWhole)
        pLaneVeh->vInitialHeadways.pop_back();

pLaneVeh->GetGlobalGeneratingTimes();
pLaneVeh->iVehiclesAll = pLaneVeh->vGlobalGeneratingTimes.size();
vLaneVeh.push_back(pLaneVeh);

// 1.2 LaneVeh North
pLaneVeh = new LaneVeh();
pLaneVeh->iID                       = 2;
pLaneVeh->posStart.x                = LANE_LENGTH / 2;
pLaneVeh->posStart.y                = LANE_WIDTH_MEDIAN / 2 + LANE_WIDTH_VEHICLE / 2;
pLaneVeh->posEnd.x                  = LANE_LENGTH / 2 * (-1);
pLaneVeh->posEnd.y                  = LANE_WIDTH_MEDIAN / 2 + LANE_WIDTH_VEHICLE / 2;
pLaneVeh->iVehiclesGenerated        = 0;
// The following parameters need calibration
pLaneVeh->fWidth                    = LANE_WIDTH_VEHICLE;
pLaneVeh->fLength                   = pLaneVeh->posStart.GetDistance(pLaneVeh->posEnd);
pLaneVeh->fDemand                   = (float)LANE_FLOW_NORTH * this->fVehDemandPercent;
pLaneVeh->fDemandInit               = (float)LANE_FLOW_NORTH;
pLaneVeh->pSignal                   = NULL;
pLaneVeh->fEdgeNorth                = pLaneVeh->posStart.y + pLaneVeh->fWidth / 2;
pLaneVeh->fEdgeSouth                = pLaneVeh->posStart.y - pLaneVeh->fWidth / 2;
pLaneVehN                           = pLaneVeh;

// Assign initial headways to this lane
fAccumulativeTimeHeadway = 0.0f;
while (fAccumulativeTimeHeadway < fSimTimeWhole)
{
        float fTimeHeadway = pRandomSolver->GetInitTimeHeadwayVeh(pLaneVeh->fDemand);
        pLaneVeh->vInitialHeadways.push_back(fTimeHeadway);
        fAccumulativeTimeHeadway = fAccumulativeTimeHeadway + fTimeHeadway;
}
if (fAccumulativeTimeHeadway > fSimTimeWhole)
        pLaneVeh->vInitialHeadways.pop_back();

pLaneVeh->GetGlobalGeneratingTimes();
pLaneVeh->iVehiclesAll = pLaneVeh->vGlobalGeneratingTimes.size();
vLaneVeh.push_back(pLaneVeh);

// 2. Create LaneCycle
// SOUTH
LaneCycle* pLaneCycle = new LaneCycle();
pLaneCycle->iID                     = 1;
pLaneCycle->posStart.x              = LANE_LENGTH / 2 * (-1);
pLaneCycle->posStart.y              = LANE_WIDTH_MEDIAN / 2 * (-1)
                                          - LANE_WIDTH_VEHICLE - LANE_WIDTH_CYCLE / 2;
pLaneCycle->posEnd.x                = LANE_LENGTH / 2;
pLaneCycle->posEnd.y                = LANE_WIDTH_MEDIAN / 2 * (-1)
                                          - LANE_WIDTH_VEHICLE - LANE_WIDTH_CYCLE / 2;
pLaneCycle->fWidth                  = LANE_WIDTH_CYCLE;
pLaneCycle->fLength                 = pLaneCycle->posStart.GetDistance(pLaneCycle->posEnd);
vLaneCycle.push_back(pLaneCycle);
pLaneCycle->fEdgeNorth              = pLaneCycle->posStart.y + pLaneCycle->fWidth / 2;
pLaneCycle->fEdgeSouth              = pLaneCycle->posStart.y - pLaneCycle->fWidth / 2;
pLaneCycleS                         = pLaneCycle;

// NORTH
pLaneCycle                          = new LaneCycle();
pLaneCycle->iID                     = 2;
pLaneCycle->posStart.x              = LANE_LENGTH / 2;
pLaneCycle->posStart.y              = LANE_WIDTH_MEDIAN/2
                                          +LANE_WIDTH_VEHICLE+LANE_WIDTH_CYCLE/2;
pLaneCycle->posEnd.x                = LANE_LENGTH / 2 * (-1);
pLaneCycle->posEnd.y                = LANE_WIDTH_MEDIAN/2
                                          +LANE_WIDTH_VEHICLE+LANE_WIDTH_CYCLE/2;
pLaneCycle->fWidth                  = LANE_WIDTH_CYCLE;
pLaneCycle->fLength                 = pLaneCycle->posStart.GetDistance(pLaneCycle->posEnd);
vLaneCycle.push_back(pLaneCycle);
pLaneCycle->fEdgeNorth              = pLaneCycle->posStart.y + pLaneCycle->fWidth / 2;
pLaneCycle->fEdgeSouth              = pLaneCycle->posStart.y - pLaneCycle->fWidth / 2;
pLaneCycleN                         = pLaneCycle;

// 3. Create Pavement
```

```
			// SOUTH
			Pavement* pPavement = new Pavement();
			pPavement->iID			= 1;
			pPavement->posStart.x		= LANE_LENGTH / 2 * (-1);
			pPavement->posStart.y		= LANE_WIDTH_MEDIAN / 2 * (-1) - LANE_WIDTH_VEHICLE
							- LANE_WIDTH_CYCLE - LANE_WIDTH_PAVEMENT / 2;
			pPavement->posEnd.x		= LANE_LENGTH / 2;
			pPavement->posEnd.y		= LANE_WIDTH_MEDIAN / 2 * (-1) - LANE_WIDTH_VEHICLE
							- LANE_WIDTH_CYCLE - LANE_WIDTH_PAVEMENT / 2;
			pPavement->fWidth		= LANE_WIDTH_PAVEMENT;
			pPavement->fLength		= pPavement->posStart.GetDistance(pPavement->posEnd);
			vPavement.push_back(pPavement);
			pPavement->fEdgeNorth		= pPavement->posStart.y + pPavement->fWidth / 2;
			pPavement->fEdgeSouth		= pPavement->posStart.y - pPavement->fWidth / 2;
			pPaveS				= pPavement;

			// NORTH
			pPavement = new Pavement();
			pPavement->iID			= 2;
			pPavement->posStart.x		= LANE_LENGTH / 2;
			pPavement->posStart.y		= LANE_WIDTH_MEDIAN / 2 + LANE_WIDTH_VEHICLE
							+ LANE_WIDTH_CYCLE + LANE_WIDTH_PAVEMENT / 2;
			pPavement->posEnd.x		= LANE_LENGTH / 2 * (-1);
			pPavement->posEnd.y		= LANE_WIDTH_MEDIAN / 2 + LANE_WIDTH_VEHICLE
							+ LANE_WIDTH_CYCLE + LANE_WIDTH_PAVEMENT / 2;
			pPavement->fWidth		= LANE_WIDTH_PAVEMENT;
			pPavement->fLength		= pPavement->posStart.GetDistance(pPavement->posEnd);
			vPavement.push_back(pPavement);
			pPavement->fEdgeNorth		= pPavement->posStart.y + pPavement->fWidth / 2;
			pPavement->fEdgeSouth		= pPavement->posStart.y - pPavement->fWidth / 2;
			pPaveN				= pPavement;

			// 5. Create lanes done
			bCreateLanes = true;
			iSimStatus   = SIM_STATUS_READY;
			Invalidate(false);
		}

		else
			MessageBox("Please set the scenario parameters first!");
}

void CPVIView::OnScenarioSettings()
{
		iSimID				= 1;
		// Set initial flow level
		this->fPedDemandPercent		= 1.0f;
		this->fVehDemandPercent		= 1.0f;

		pSceOverall			= new ScenarioOverall();

		pPaveS				= NULL;
		pPaveN				= NULL;
		pLaneVehS			= NULL;
		pLaneVehN			= NULL;
		pLaneCycleS			= NULL;
		pLaneCycleN			= NULL;
		pMedianCentre			= NULL;

		// Set simulation timer
		iTimerInterval			= TIMER_INTERVAL_DEFAULT;
		SetTimer(1,iTimerInterval,NULL);

		// Create the SimWorld Handler
		pRandomSolver			= new RandomSolver;
		DRandomSeed			 = GetTickCount();
		srand(DRandomSeed);
		pMathSolver			= new MathSolver;
		pFuzzySolver			= new FuzzySolver;
		pCarFollowingModel_Fuzzy	= new CarFollowingModel_Fuzzy(pFuzzySolver);
		iScenarioType			= SCENARIO_TYPE;

		// Set simulation time, etc
		fSimTimeWhole			= 3600.0f * (float)SAMPLE_SIZE / (LANE_FLOW_SOUTH + LANE_FLOW_NORTH);
		fSimTimeStep			= SIM_TIME_STEP_DEFAULT;
		fSimTimeElapse			= 0.0f;
		iNumVehGenerated		= 0;
		iNumPedGenerated		= 0;
		bCreatePedCrossing		= false;
		bCreatePedODPairs		= false;
		pPedCrossing			= NULL;

		// Scenatio setting finished
		bScenarioSettings = true;
}
```

```
void CPVIView::VehUpdate(vector<LaneVeh*>& vLaneVeh)
{
        // Search all lanes
        int iLaneVehSize = vLaneVeh.size();
        for(int i=0; i<iLaneVehSize; i++)
        {
                LaneVeh* pLaneVeh = vLaneVeh[i];
                int iVehicleNum = pLaneVeh->vVehicles.size();

                ////////////////////////////////////////////////////////////////////////////////////////////////////
                // Update all vehicles' DeltaX, Speed, Position, Time-to-react, bOut,
                // Emission Interval Time
                for (int j=0; j<iVehicleNum; j++)
                {
                        // Get the vehicle
                        Vehicle* pVehicle = pLaneVeh->vVehicles[j];

                        // Update DeltaX, Speed
                        pVehicle->fDeltaX = pVehicle->fDeltaX + pVehicle->GetDistance(fSimTimeStep);
                        pVehicle->fSpeed  = pVehicle->GetNewSpeed(fSimTimeStep);
                        pVehicle->FilterDynWhenStopAtSignal();
                        // Speed nearly zero

                        // Update bOut
                        if(pVehicle->fDeltaX > pLaneVeh->fLength)        // Out of boundary
                                pVehicle->bOut = true;
                        // Update Vehicle position (x,y)
                        pVehicle->GetPosition(pVehicle->fDeltaX);
                        // Update Time-to-react
                        pVehicle->fTimeToReact = pVehicle->fTimeToReact - fSimTimeStep;
                        // Update Emission Interval Time
                        pVehicle->fEmissionInterval = pVehicle->fEmissionInterval - fSimTimeStep;
                }
                // Update all vehicles' DeltaX, Speed, Position, Time-to-react, bOut,
                // Emission Interval Time
                ////////////////////////////////////////////////////////////////////////////////////////////////////

                //***************************************************************************************
                // Update all vehicles' Acc
                for (j=0; j<iVehicleNum; j++)
                {
                        Vehicle* pVehicle = pLaneVeh->vVehicles[j];    // Get the vehicle
                        if(pVehicle->fTimeToReact - fSimTimeStep < 0)    // It is time to make reaction
                        {
                                pVehicle->fTimeToReact = pVehicle->fReactionTime;    // Reset the Time-to-react
                                bool bIsCarFollowing  = pVehicle->IsCarFollowing();    // Check if car following
                                if(!bIsCarFollowing)    // Free flow condition
                                {
                                        if(pVehicle->fSpeed >= 0 && pVehicle->fSpeed <= pVehicle->fSpeedDSR)
                                        {
                                                pVehicle->fAcc = pVehicle->fAccFrFInit
                                                * sqrt(1-pVehicle->fSpeed / pVehicle->fSpeedDSR);
                                        }
                                        else
                                        {
                                                // pVehicle->fAcc = -1 * pVehicle->fDecFrF;
                                                pVehicle->fAcc = 0;
                                        }
                                }
                                else    // Car following condition
                                {
                                        float DV = pVehicle->pLeader->fSpeed - pVehicle->fSpeed;
                                        float fDistDSR = pVehicle->fSpeed * pVehicle->fTimeHeadwayDSR;
                                        if(fDistDSR < pVehicle->pLeader->fMargin)
                                        {
                                                fDistDSR = pVehicle->pLeader->fMargin;
                                        }
                                        float DSSD = (pVehicle->pLeader->fDeltaX - pVehicle->pLeader->fLength
                                                                - pVehicle->fDeltaX) / fDistDSR;
                                        pVehicle->fAcc = pCarFollowingModel_Fuzzy->GetAcceleration(DV,DSSD);
                                        // Get Acc from Fuzzy Logic Model
                                }

                                pVehicle->FilterAccFromLeaderVeh();
                                pVehicle->FilterAccFromSignal();
                                pVehicle->FilterAccFromPed();

                                // Decide Final Acc
                                if(pVehicle->fAcc < -1 * pVehicle->fDecMax)
                                        pVehicle->fAcc = -1 * pVehicle->fDecMax;
                                if(pVehicle->fAcc > pVehicle->fAccMax)
                                        pVehicle->fAcc = pVehicle->fAccMax;
                                if(pVehicle->fSpeed<=0 && pVehicle->fAcc<0)
                                        pVehicle->fAcc = 0;
                        }
```

```
				}
				// Update all vehicles' Acc
				//**************************************************************************************


		//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
				// Update vehicle indicators
				for (j=0; j<iVehicleNum; j++)
				{
					// Get the vehicle
					Vehicle* pVehicle = pLaneVeh->vVehicles[j];
					if(pVehicle->fDeltaX >= 0)
					{
						// Update TravelTime and TravelDist
						pVehicle->fTravelTime	= pVehicle->fTravelTime + fSimTimeStep;
						pVehicle->fTravelDist	= pVehicle->fDeltaX;
						// Update Emissions
						if(pVehicle->fEmissionInterval <=0)
						{
							pVehicle->fEmissionInterval = VEH_EMISSION_INTERVAL;
							pVehicle->fEmissionCO2 = pVehicle->fEmissionCO2
										+ pVehicle->GetEmissionCO2();
						}
					}
				}
				// Update vehicle indicators -----

		//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
		}
}

void CPVIView::VehDelete(vector<LaneVeh*>& vLaneVeh)
{
		int iLaneVehSize = vLaneVeh.size();
		for(int i=0; i<iLaneVehSize; i++)
		{
				LaneVeh* pLaneVeh = vLaneVeh[i];
				int iVehicleNum = pLaneVeh->vVehicles.size();
				for (int j=0; j<iVehicleNum; j++)
				{
					Vehicle* pVehicle = pLaneVeh->vVehicles[j];
					if(pVehicle->bOut == true)
					{
						// Modify Leader and Follower
						if(pVehicle->pFollower!=NULL)
							pVehicle->pFollower->pLeader = NULL;

						// Modify conflicting pedestrians
						if(pVehicle->vPedConflict.size()!=0)
							pVehicle->vPedConflict.clear();

						// Export the overall stats of this vehicle to Class VehicleOverall
						RecordVehicleOverall(pVehicle);

						// Delete this vehicle object
						delete pVehicle;
						pVehicle = NULL;
						pLaneVeh->vVehicles.erase(pLaneVeh->vVehicles.begin()+j);			//+j
						j--;
						iVehicleNum--;
					}
				}
		}
}

void CPVIView::VehGenerate(vector<LaneVeh*>& vLaneVeh)
{
		int iLaneVehNum = vLaneVeh.size();			// Get the size of the Vehicle Lane Vector
		for(int i=0; i<iLaneVehNum; i++)
		{
				LaneVeh* pLaneVeh = vLaneVeh[i];
				if(pLaneVeh->iVehiclesGenerated < pLaneVeh->iVehiclesAll)
				// Vehicles exist in this lane and still got vehicles to be generated
				{
					// Get the next time stamp at which to generate a vehicle
					float fTimeStampToGenerateVeh = pLaneVeh->vGlobalGeneratingTimes[pLaneVeh->iVehiclesGenerated];

					if(fSimTimeElapse >= fTimeStampToGenerateVeh)
					{
						// Check if there is a space for new vehicle
						if(pLaneVeh->IsAvailableForNewVehicle())
						{
							Vehicle* pVehicle = new Vehicle;	// Generate a vehicle in this lane
							iNumVehGenerated++;			// Update the global number
							pLaneVeh->iVehiclesGenerated++;	// Update the number of this lane
```

```
                                        // Assign initial status to the new vehicle
                                        AssignInitStatusToVeh(pVehicle,pLaneVeh);

                                        // Add the generated vehicle to this lane
                                        pLaneVeh->vVehicles.push_back(pVehicle);
                                }

                                else    // No space for new vehicle
                                {
                                        Vehicle* pLastVeh = pLaneVeh->vVehicles.back();
                                       if(! ( (pLastVeh->fDeltaX<0 && pLastVeh->fSpeed==0 && pLastVeh->fAcc==0)
                                                    ||(pLaneVeh->vVehicles.size()>50) ) )
                                        // Generate new vehicle
                                        {
                                                Vehicle* pVehicle = new Vehicle;    // Generate a vehicle in this lane
                                                iNumVehGenerated++;                 // Update the global number
                                                pLaneVeh->iVehiclesGenerated++; // Update the number of this lane

                                                // Assign initial status to the new vehicle
                                                AssignInitStatusToVeh(pVehicle,pLaneVeh);

                                                // Add the generated vehicle to this lane
                                                pLaneVeh->vVehicles.push_back(pVehicle);
                                        }
                                }
                        }
                }
        }
}

void CPVIView::UpdateScene()
{
        ResetEdge();
        SignalUpdate();

        // Update vehicles
        VehUpdate(vLaneVeh);
        VehDelete(vLaneVeh);
        VehGenerate(vLaneVeh);

        // Update pedestrians
        PedUpdate();
        PedDelete();
        PedGenerate();

        // Last: Update time
        fSimTimeElapse = fSimTimeElapse + fSimTimeStep;
}

void CPVIView::OnCreatePedCrossing()
{
        AddPedCrossing Dlg;
        if(IDOK == Dlg.DoModal())
        {
                bCreatePedCrossing = true;
                // Create a ped crossing
                pPedCrossing = new PedCrossing;
                pPedCrossing->iType                     = Dlg.m_PedCrossingType;
                pPedCrossing->PosCentroid.x             = ODA_LENGTH/2;                 // For model validation
//              pPedCrossing->PosCentroid.x             = 0.0f;
                pPedCrossing->PosCentroid.y             = 0;
                pPedCrossing->fLength                   = CROSSING_LENGTH;
                pPedCrossing->fWidth                    = LANE_WIDTH_MEDIAN + LANE_WIDTH_VEHICLE * 2
                                                                + LANE_WIDTH_CYCLE * 2;
                pPedCrossing->iPedEntered               = 0;

                // Assign PedCrossing Initial Status
                switch(pPedCrossing->iType)
                {
                        // Zebra Crossing
                        case(CROSSING_TYPE_ZEBRA):
                        {
                                pPedCrossing->iMaxPeriod                = 0;
                                pPedCrossing->iCurrentPeriod            = 0;
                                pPedCrossing->fMarginUseDist            = CROSSING_USE_MARGIN_ZEBRA;
                               pPedCrossing->fUseMarginTimeRatio=CROSSING_USE_MARGIN_TIME_RATIO_ZEBRA;
                                // Do nothing
                                break;
                        }
                        // Fixed-time Crossing
                        case(CROSSING_TYPE_FIXED):
                        {
                                pPedCrossing->iMaxPeriod                = 5;
                                int fPeriodTimes[]                      = {50,3,2,20,5};
                                pPedCrossing->vPeriodTimes.assign
                                (fPeriodTimes, fPeriodTimes + pPedCrossing->iMaxPeriod);
```

```
                pPedCrossing->iCurrentPeriod            = 1;
                pPedCrossing->fCurrentPeriodStartTime           = 0;
                pPedCrossing->fCurrentPeriodElapseTime          = 0;
                pPedCrossing->fMarginUseDist                    = CROSSING_USE_MARGIN_FIXED;
                pPedCrossing->fUseMarginTimeRatio= CROSSING_USE_MARGIN_TIME_RATIO_FIXED;
                break;
        }
        case(CROSSING_TYPE_PUFFIN):
        {
                pPedCrossing->iMaxPeriod                        = 7;
                int fPeriodTimes[]                              = {26,3,2,16,3,12,2};
                pPedCrossing->vPeriodTimes.assign
                (fPeriodTimes, fPeriodTimes + pPedCrossing->iMaxPeriod);

                pPedCrossing->iCurrentPeriod                    = 1;
                pPedCrossing->fCurrentPeriodStartTime           = 0;
                pPedCrossing->fCurrentPeriodElapseTime          = 0;
                pPedCrossing->fMarginUseDist                    = CROSSING_USE_MARGIN_PUFFIN;
                pPedCrossing->fUseMarginTimeRatio               =
                CROSSING_USE_MARGIN_TIME_RATIO_PUFFIN;
                break;
        }
        case(CROSSING_TYPE_ADV):
        {
                break;
        }
}

// Create PedWaitingZone South
PedWaitingZone* pPedWaitingZone = new PedWaitingZone;
pPedWaitingZone->strID = "S";
pPedWaitingZone->fLength = pPedCrossing->fLength;
pPedWaitingZone->fWidth = CROSSING_WAITING_AREA_WIDTH;
pPedWaitingZone->PosCentroid.x = pPedCrossing->PosCentroid.x;
pPedWaitingZone->PosCentroid.y = LANE_WIDTH_MEDIAN / 2 * (-1) - LANE_WIDTH_VEHICLE
                                - LANE_WIDTH_CYCLE - pPedWaitingZone->fWidth/2;
pPedWaitingZone->pPedCrossing = pPedCrossing;
pPedWaitingZone->pPavement = vPavement[0];    // Belongs to South Pavement
pPedCrossing->pPedWaitingZoneS = pPedWaitingZone;

// Create PedWaitingZone North
pPedWaitingZone = new PedWaitingZone;
pPedWaitingZone->strID = "N";
pPedWaitingZone->fLength = pPedCrossing->fLength;
pPedWaitingZone->fWidth = CROSSING_WAITING_AREA_WIDTH;

pPedWaitingZone->PosCentroid.x = pPedCrossing->PosCentroid.x;
pPedWaitingZone->PosCentroid.y = LANE_WIDTH_MEDIAN / 2 + LANE_WIDTH_VEHICLE
                                + LANE_WIDTH_CYCLE + pPedWaitingZone->fWidth/2;
pPedWaitingZone->pPedCrossing = pPedCrossing;
pPedWaitingZone->pPavement = vPavement[1];    // Belongs to North Pavement
pPedCrossing->pPedWaitingZoneN = pPedWaitingZone;

if(pPedCrossing->iType == CROSSING_TYPE_ZEBRA)        // No signal
{
        pPedCrossing->pSignalS = NULL;
        pPedCrossing->pSignalN = NULL;
        pPedCrossing->pSignalW = NULL;
        pPedCrossing->pSignalE = NULL;
}
else        // Create signal
{
        // Create Signal South
        Signal* pSignal = new Signal;
        pPedCrossing->pSignalS = pSignal;
        pSignal->strID = "S";
        pSignal->pPedCrossing = pPedCrossing;
        pSignal->PosCentroid.x = pSignal->pPedCrossing->PosCentroid.x;
        pSignal->PosCentroid.y = pSignal->pPedCrossing->pPedWaitingZoneS->PosCentroid.y
                        + pSignal->pPedCrossing->pPedWaitingZoneS->fWidth / 2;
        pSignal->fLength = pSignal->pPedCrossing->fLength;
        pSignal->iStatus = SIGNAL_STATUS_RED;

        // Create Signal North
        pSignal = new Signal;
        pPedCrossing->pSignalN = pSignal;
        pSignal->strID = "N";
        pSignal->pPedCrossing = pPedCrossing;
        pSignal->PosCentroid.x = pSignal->pPedCrossing->PosCentroid.x;
        pSignal->PosCentroid.y = pSignal->pPedCrossing->pPedWaitingZoneN->PosCentroid.y
                        - pSignal->pPedCrossing->pPedWaitingZoneS->fWidth / 2;
        pSignal->fLength = pSignal->pPedCrossing->fLength;
        pSignal->iStatus = SIGNAL_STATUS_RED;

        // Create Signal West
```

```
			pSignal = new Signal;
			pPedCrossing->pSignalW		= pSignal;
			pSignal->strID			= "W";
			pSignal->pPedCrossing		= pPedCrossing;
			pSignal->PosCentroid.x		= pPedCrossing->PosCentroid.x
						  - pPedCrossing->fLength /2 - CROSSING_STOP_LINE_GAP;
			pSignal->PosCentroid.y		= vLaneVeh[0]->posStart.y;
			pSignal->fLength		= vLaneVeh[0]->fWidth;
			pSignal->iStatus		= SIGNAL_STATUS_GREEN;
			pSignal->pLaneVeh		= vLaneVeh[0];
			pSignal->pLaneVeh->pSignal	= pSignal;
			pSignal->fDeltaX		= fabs(pSignal->PosCentroid.x
						  - pSignal->pLaneVeh->posStart.x);

			// Create Signal East
			pSignal = new Signal;
			pPedCrossing->pSignalE		= pSignal;
			pSignal->strID			= "E";
			pSignal->pPedCrossing		= pPedCrossing;
			pSignal->PosCentroid.x		= pPedCrossing->PosCentroid.x
						  + pPedCrossing->fLength /2 + CROSSING_STOP_LINE_GAP;
			pSignal->PosCentroid.y		= vLaneVeh[1]->posStart.y;
			pSignal->fLength		= vLaneVeh[1]->fWidth;
			pSignal->iStatus		= SIGNAL_STATUS_GREEN;
			pSignal->pLaneVeh		= vLaneVeh[1];
			pSignal->pLaneVeh->pSignal	= pSignal;
			pSignal->fDeltaX		= fabs(pSignal->PosCentroid.x
						  - pSignal->pLaneVeh->posStart.x);
		}

		// Create a Ped Crossing done
		Invalidate(false);
	}
}

void CPVIView::SignalUpdate()
{
	if(bCreatePedCrossing)
	{
		switch(pPedCrossing->iType)
		{
			// Zebra Crossing
			case(CROSSING_TYPE_ZEBRA):
			{
				// Do nothing
				break;
			}

			// Fixed-time Crossing
			case(CROSSING_TYPE_FIXED):
			{
				// Update the ElapseTime of Current Period
				pPedCrossing->fCurrentPeriodElapseTime = fSimTimeElapse
				- pPedCrossing->fCurrentPeriodStartTime;
				// Update Period Number
				if(pPedCrossing->fCurrentPeriodElapseTime >=
				pPedCrossing->vPeriodTimes[pPedCrossing->iCurrentPeriod-1])
				{
					pPedCrossing->iCurrentPeriod++;
					if(pPedCrossing->iCurrentPeriod > pPedCrossing->iMaxPeriod)
					{
						pPedCrossing->iCurrentPeriod = 1;
					}
					pPedCrossing->fCurrentPeriodStartTime		= fSimTimeElapse;
					pPedCrossing->fCurrentPeriodElapseTime		= 0;
				}
				// Update Signal Status
				switch(pPedCrossing->iCurrentPeriod)
				{
					case(1):
					{
						pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_GREEN;
						pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_GREEN;
						pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
						pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
						break;
					}
					case(2):
					{
						pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_AMBER;
						pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_AMBER;
						pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
						pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
						break;
					}
					case(3):
```

```
                {
                        pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
                        break;
                }
                case(4):
                {
                        pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_GREEN;
                        pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_GREEN;
                        break;
                }
                case(5):
                {
                        pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
                        break;
                }
        }
        break;
}

// Puffin Crossing
case(CROSSING_TYPE_PUFFIN):
{
        // Update the ElapseTime of Current Period
        pPedCrossing->fCurrentPeriodElapseTime = fSimTimeElapse
        - pPedCrossing->fCurrentPeriodStartTime;

        // Update Period Number
        // Check if period time is exceeded
        if(pPedCrossing->fCurrentPeriodElapseTime >=
        pPedCrossing->vPeriodTimes[pPedCrossing->iCurrentPeriod-1]) // exceeded
        {
                // Check Period 1
                if(pPedCrossing->iCurrentPeriod==1)
                {
                        // Check pedestrian crossing demand
                        bool bPedCrossDemand = false;
                        int iSize = vPedestrian.size();
                        for(int i=0;i<iSize;i++)
                        {
                                Pedestrian* pPed = vPedestrian[i];
                                Position posPed = pPed->posCentroid;
                                Position posRectangle =
                                pPed->pPedWaitingZoneNear->PosCentroid;
                                float fLength = pPed->pPedWaitingZoneNear->fLength;
                                float fHeight = pPed->pPedWaitingZoneNear->fWidth;
                                bool bPedIn = pMathSolver->IsPosInRectangle
                                        (posPed,posRectangle,fLength,fHeight);
                                if(bPedIn)  // demand exists
                                {
                                        bPedCrossDemand = true;
                                        break;
                                }
                        }
                        if(bPedCrossDemand)  // change period
                        {
                                pPedCrossing->iCurrentPeriod++;
                                if(pPedCrossing->iCurrentPeriod >
                                        pPedCrossing->iMaxPeriod)
                                {
                                        pPedCrossing->iCurrentPeriod = 1;
                                }
                                pPedCrossing->fCurrentPeriodStartTime =
                                        fSimTimeElapse;
                                pPedCrossing->fCurrentPeriodElapseTime = 0;
                        }
                }

                // Other Periods
                else
                {
                        // change Period
                        pPedCrossing->iCurrentPeriod++;
                        if(pPedCrossing->iCurrentPeriod > pPedCrossing->iMaxPeriod)
                        {
                                pPedCrossing->iCurrentPeriod = 1;
                        }
                        pPedCrossing->fCurrentPeriodStartTime =  fSimTimeElapse;
                        pPedCrossing->fCurrentPeriodElapseTime          = 0;
```

```
		}
}

else  // Not exceeded
{
		// Check if gap-out for Period 6
		if(pPedCrossing->iCurrentPeriod==6)
		{
			// Check pedestrians presence in the on-road part of crossing
			bool bPedCrossOnRoad = false;
			Position posRectangle = pPedCrossing->PosCentroid;
			float fLength = pPedCrossing->fLength;
			float fHeight = vMedian[0]->fWidth + vLaneVeh[0]->fWidth*2
						+ vLaneCycle[0]->fWidth*2;
			int iSize = vPedestrian.size();
			for(int i=0;i<iSize;i++)
			{
				Pedestrian* pPed = vPedestrian[i];
				bool bPedOnRoad = pMathSolver->IsPosInRectangle
				(pPed->posCentroid,posRectangle,fLength,fHeight);
				if(bPedOnRoad)
				{
					bPedCrossOnRoad = true;
					break;
				}
			}
			// Decide if Period 6 gap out
			if(!bPedCrossOnRoad)
			{
				// Gap out
				pPedCrossing->iCurrentPeriod++;
				if(pPedCrossing->iCurrentPeriod >
					pPedCrossing->iMaxPeriod)
				{
					pPedCrossing->iCurrentPeriod = 1;
				}
				pPedCrossing->fCurrentPeriodStartTime
						= fSimTimeElapse;
				pPedCrossing->fCurrentPeriodElapseTime = 0;
			}
		}
}

// Update Signal Status
switch(pPedCrossing->iCurrentPeriod)
{
		case(1):
		{
			pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_GREEN;
			pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_GREEN;
			pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
			break;
		}
		case(2):
		{
			pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_AMBER;
			pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_AMBER;
			pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
			break;
		}
		case(3):
		{
			pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
			break;
		}
		case(4):
		{
			pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_GREEN;
			pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_GREEN;
			break;
		}
		case(5):
		{
			pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
			pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
			break;
		}
```

```
                    case(6):
                    {
                        pPedCrossing->pSignalW->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalE->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
                        break;
                    }
                    case(7):
                    {
                        pPedCrossing->pSignalW->iStatus =
                                SIGNAL_STATUS_GREENAMBER;
                        pPedCrossing->pSignalE->iStatus =
                                SIGNAL_STATUS_GREENAMBER;
                        pPedCrossing->pSignalS->iStatus = SIGNAL_STATUS_RED;
                        pPedCrossing->pSignalN->iStatus = SIGNAL_STATUS_RED;
                        break;
                    }
                }
                break;
            }

            // Advanced Crossing
            case(CROSSING_TYPE_ADV):
            {
                break;
            }
        }

        // Record the Peroid Number when it changes
        if(pPedCrossing->fCurrentPeriodElapseTime == 0)
        {
            // Record the Peroid Number
            SignalDynamics* pSignalDynamics          = new SignalDynamics();
            vSignalDynamics.push_back(pSignalDynamics);
            pSignalDynamics->fSimTimeElapse          = fSimTimeElapse;
            pSignalDynamics->iCurrentPeriod          = pPedCrossing->iCurrentPeriod;
            // Record how many peds have entered in this period
            pSignalDynamics->iPedEntered             = pPedCrossing->iPedEntered;
            // Reset pedestrian entered number
            pPedCrossing->iPedEntered                = 0;
        }
    }
}

void CPVIView::OnSimPlayStep()
{
    iSimStatus = SIM_STATUS_PLAYSTEP;
    if(fSimTimeElapse < fSimTimeWhole)
    {
        // Draw the current scene to the screen
        Invalidate(false);
        // Update current scene to the next state
        UpdateScene();
    }
    else
    {
        iSimStatus = SIM_STATUS_STOP;
        Invalidate(false);
    }
}

float CPVIView::GetNewSpeed(float v0, float a, float t)
{
    float vt = v0 + a * t;
    if(vt < 0)
    {
        vt = 0;
    }
    return vt;
}

float CPVIView::GetDistance(float v0, float a, float t)
{
    float s;
    float vt = v0 + a* t;
    if(vt >= 0)
    {
        s = v0 * t + 0.5 * a * t * t;
    }
    else
    {
        s = v0 * v0 / (-2*a);
    }
    return s;
}
```

```
void CPVIView::AssignInitStatusToVeh(Vehicle* pVehicle, LaneVeh* pLaneVeh)
{
	pVehicle->iID		= iNumVehGenerated;			// Vehicle ID
	pVehicle->iType		= pRandomSolver->GetVehType();	// Vehicle type

	// Assign vehicle occupancy
	switch(pVehicle->iType)
	{
		case(VEHICLE_TYPE_LV):
		{
			if(pRandomSolver->GetBinary(0.50)) // being a taxi
			{
				pVehicle->fOccupancy = (float)VEHICLE_OCUP_TAXI;
			}
			else
			{
				pVehicle->fOccupancy = (float)VEHICLE_OCUP_CAR;
			}
			break;
		}
		case(VEHICLE_TYPE_MCV):
		{
			pVehicle->fOccupancy = (float)VEHICLE_OCUP_MCV;
			break;
		}
		case(VEHICLE_TYPE_HCV):
		{
			pVehicle->fOccupancy = (float)VEHICLE_OCUP_HCV;
			break;
		}
		case(VEHICLE_TYPE_BCR):
		case(VEHICLE_TYPE_BCA):
		{
			pVehicle->fOccupancy = (float)VEHICLE_OCUP_BUS;
			break;
		}
	}

	pVehicle->iTypeFuel			= pRandomSolver->GetVehTypeFuel(pVehicle);	// Vehicle fuel type
	pVehicle->fLength			= pRandomSolver->GetVehLength(pVehicle);
	pVehicle->fWidth			= pRandomSolver->GetVehWidth(pVehicle);
	pVehicle->fMargin			= pRandomSolver->GetVehMargin();
	pVehicle->pLaneVeh			= pLaneVeh;					// Vehicle in which Lane
	pVehicle->fAcc				= 0;						// Initial Acc
	pVehicle->fDecMax			= pRandomSolver->GetVehDecMax(pVehicle);
	pVehicle->fDecFrF			= pRandomSolver->GetVehDecFrF(pVehicle);
	pVehicle->fAccMax			= pRandomSolver->GetVehAccMax(pVehicle);
	pVehicle->fTimeHeadwayDSR		= pRandomSolver->GetVehTimeHeadwayDSR();
	pVehicle->fTimeHeadwayFrF		= VEHICLE_TIME_HEADWAY_FRF;
	pVehicle->fEmissionInterval		= VEH_EMISSION_INTERVAL;
	pVehicle->bOut				= false;
	pVehicle->bStopAtSignal			= false;

	// Vehicle indicators
	pVehicle->fTravelDist			= 0;
	pVehicle->fTravelTime			= 0;
	pVehicle->fDelay			= 0;
	pVehicle->fEmissionCO2			= 0;

	// Set free flow initial acc
	if(pVehicle->iType == VEHICLE_TYPE_LV || pVehicle->iType == VEHICLE_TYPE_MCV)
	// Light vehicles
	{
		pVehicle->fAccFrFInit = VEHICLE_FF_ACC_INIT_LIGHT;
	}
	else		// Heavy vehicles
	{
		pVehicle->fAccFrFInit = VEHICLE_FF_ACC_INIT_HEAVY;
	}

	// Set leader
	if(pLaneVeh->vVehicles.size() == 0)   // No leader
		pVehicle->pLeader = NULL;
	else
	{
		pVehicle->pLeader = pLaneVeh->vVehicles.back();			// Last vehicle as leader
		pVehicle->pLeader->pFollower = pVehicle;			// this vehicle as last vehicle's follower
	}

	// Set follower
	pVehicle->pFollower = NULL;

	// Set DeltaX
	if(pVehicle->pLeader == NULL)
		pVehicle->fDeltaX = 0;
```

```
		else if(pVehicle->pLeader->fDeltaX - pVehicle->pLeader->fLength
		- pVehicle->pLeader->fMargin >= 0)
			pVehicle->fDeltaX = 0;
		else
			pVehicle->fDeltaX = pVehicle->pLeader->fDeltaX - pVehicle->pLeader->fLength - pVehicle->pLeader->fMargin;

		// Set reaction time
		pVehicle->fReactionTime			= pRandomSolver->GetVehReactionTime();
		pVehicle->fTimeToReact			= pVehicle->fReactionTime;

		// Set desired speed
		pVehicle->fSpeedDSR			= pRandomSolver->GetVehSpeedDSR(pVehicle);

		// Set initial speed
		if(pVehicle->pLeader == NULL)
			pVehicle->fSpeed = pVehicle->fSpeedDSR;
		else
		{
			// Calculate the maximum allowed speed
			float fDeltaXLeaderRearNew = pVehicle->pLeader->fDeltaX
			- pVehicle->pLeader->fLength - pVehicle->pLeader->fMargin
			+ (pVehicle->pLeader->fSpeed * pVehicle->pLeader->fSpeed)
			/ (2 * pVehicle->pLeader->fDecMax);
			float fSpeedMax = pVehicle->fDecMax
			* (sqrt(pVehicle->fTimeToReact * pVehicle->fTimeToReact
			+ 2 * (fDeltaXLeaderRearNew - pVehicle->fDeltaX) / pVehicle->fDecMax)
			- pVehicle->fTimeToReact);
			if(fSpeedMax >= pVehicle->fSpeedDSR)
				pVehicle->fSpeed = pVehicle->fSpeedDSR;
			else
				pVehicle->fSpeed = fSpeedMax;
		}

		// Calculate positions of front and rear centroids
		pVehicle->GetPosition(pVehicle->fDeltaX);
}

void CPVIView::PedGenerate()
{
		int iSize = vPedODPair.size();
		for(int i=0; i<iSize; i++)
		{
			PedODPair* pPedODPair = vPedODPair[i];
			if(pPedODPair->iPedGened < pPedODPair->iPedTotal)   // Still have peds to be gened
			{
				for(int i=pPedODPair->iPedGened; i<pPedODPair->iPedTotal; i++)
				{
					if(fSimTimeElapse >= pPedODPair->vGlobalGenTimes[i])   // Time to gen a ped
					{
						// Check space availability
						if(pPedODPair->pOri->iPedExisting < pPedODPair->pOri->iPedExitMax)
						{
							// Generate a pedestrian to this OD Pair
							Pedestrian* pPed = new Pedestrian;
							vPedestrian.push_back(pPed);
							pPedODPair->pOri->iPedGened++;
							pPedODPair->pOri->iPedExisting++;
							pPedODPair->iPedGened++;
							iNumPedGenerated++;
							// Assign initial status to this ped
							AssignInitStatusToPed(pPed, pPedODPair);
						}
						else
						{
							break;
						}
					}
					else
					{
						break;
					}
				}
			}
		}
}

void CPVIView::AssignInitStatusToPed(Pedestrian* pPed, PedODPair* pPedODPair)
{
		pPed->iID				= iNumPedGenerated;
		pPed->fBodyRadius			= PED_BODY_RADIUS;
		pPed->iType				= pRandomSolver->GetPedType();
		pPed->fSpeedDSR			= pRandomSolver->GetPedSpeedDSR(pPed->iType);
		pPed->fSpeedMaxWalk			= pRandomSolver->GetPedSpeedMaxWalk(pPed->iType);
		pPed->fSpeedMaxRun			= PED_SPEED_MAX_RUN;
		pPed->fSpeed				= 0.0f;
		pPed->fReactionTime			= pRandomSolver->GetPedReactionTime();
```

```
		pPed->fTimeToReact		= 0.0f;
		pPed->pPedODPair		= pPedODPair;
		pPed->posOri			= pPed->pPedODPair->pOri->GetPosition(pRandomSolver);
		pPed->posDest			= pPed->pPedODPair->pDest->GetPosition(pRandomSolver);
		pPed->posTarget			= pPed->posDest;
		pPed->posCentroid		= pPed->posOri;
		pPed->fWaitTime			= 0.0f;
		pPed->fGroupingDist		= pPed->fSpeedDSR * PED_GROUP_RADIUS_TIME;
		pPed->fTravelTimeDSR		= pMathSolver->GetDistance(pPed->posOri,pPed->posDest) / pPed->fSpeedDSR;
		pPed->fTravelTime		= 0.0f;
		pPed->fTravelDist		= 0.0f;
		pPed->fNearsideTTC		= -1*(float)INT_MAX;
		pPed->fFarsideTTC		= -1*(float)INT_MAX;
		pPed->bTimeOut			= false;
		pPed->fSurvivalTime		= 0.0f;
		pPed->pVehConflict		= NULL;
		pPed->pLaneVehCurrent		= NULL;
		pPed->pVehConflictTemp		= NULL;
		pPed->bVehBlockTemp		= false;
		pPed->bStopDueToFlow		= false;
		pPed->iPosTypePrev		= PED_POS_OUTSIDE_SIM;
		pPed->bMedianAvailable		= true;
		pPed->iColID			= GetColID(pPed->posCentroid.x, vPedODASouth[0]->PosCentroid.x
						- vPedODASouth[0]->fLength/2,  ODA_LENGTH);
		pPed->iColIDPrev		= -1;
		pPed->iRowID			= GetRowID(pPed);
		pPed->iRowIDPrev		= -1;

		// Assign lanes to this pedestrian
		if(pPed->posOri.y < 0)   // S-N
		{
			pPed->pPaveNear		= vPavement[0];
			pPed->pPaveFar		= vPavement[1];
			pPed->pLaneCycleNear	= vLaneCycle[0];
			pPed->pLaneCycleFar	= vLaneCycle[1];
			pPed->pLaneVehNear	= vLaneVeh[0];
			pPed->pLaneVehFar	= vLaneVeh[1];
			pPed->pMedian		= vMedian[0];
			pPed->iDirection	= 1;
		}
		else   // N-S
		{
			pPed->pPaveNear		= vPavement[1];
			pPed->pPaveFar		= vPavement[0];
			pPed->pLaneCycleNear	= vLaneCycle[1];
			pPed->pLaneCycleFar	= vLaneCycle[0];
			pPed->pLaneVehNear	= vLaneVeh[1];
			pPed->pLaneVehFar	= vLaneVeh[0];
			pPed->pMedian		= vMedian[0];
			pPed->iDirection	= -1;
		}

		// Assign initial Position Type
		pPed->iPosType = GetPedPosType(pPed);

		// Link PedCrossing facility and
		// Decide if use the Crossing Facility
		if(this->bCreatePedCrossing==true)
		{
			pPed->pPedCrossing		= this->pPedCrossing;
			pPed->pPedWaitingZoneNear	= (pPed->iDirection>0)?this->pPedCrossing->pPedWaitingZoneS:
							  this->pPedCrossing->pPedWaitingZoneN;
			pPed->posWaitingZoneNear	= pPed->pPedWaitingZoneNear->GetPosition (pRandomSolver,pMathSolver);
			pPed->pPedWaitingZoneFar	= (pPed->iDirection>0)?this->pPedCrossing->pPedWaitingZoneN:
							  this->pPedCrossing->pPedWaitingZoneS;
			pPed->posWaitingZoneFar		= pPed->pPedWaitingZoneFar->GetPosition (pRandomSolver,pMathSolver);
			pPed->bUseCrossing		= pPed->IsUseCrossing(pPedCrossing);
		}
		else
		{
			pPed->pPedCrossing		= NULL;
			pPed->pPedWaitingZoneNear	= NULL;
			pPed->posWaitingZoneNear	= pPed->posDest;
			pPed->pPedWaitingZoneFar	= NULL;
			pPed->posWaitingZoneFar		= pPed->posOri;
			pPed->bUseCrossing		= false;
		}

		// Appears at the edge at the beginning, update the edge
		if(pPed->iPosType==PED_POS_PAVE_NEAR_EDGE)
		{
			// Update the ped row num at the edge(nearside pavement)
			vvEdge[pPed->iColID-1][3-2*pPed->iDirection-1]->iPedExist++;
		}
}
```

```
void CPVIView::PedUpdate()
{
	int iSize = vPedestrian.size();

	/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
	// Update Waiting time,TravelDist,position,position type,Time-to-React,TravelTime,
	// Current Vehicle lane, VehConflict, bUseCrossing
	for(int i=0;i<iSize;i++)
	{
		Pedestrian* pPed = vPedestrian[i];

		// Update Waiting Time and survival time
		if(pPed->fSpeed==0)
		{
			pPed->fWaitTime =  pPed->fWaitTime + fSimTimeStep;
		}
		pPed->fSurvivalTime = pPed->fSurvivalTime + fSimTimeStep;

		// Update bUseCrossing
		if(this->bCreatePedCrossing==true)
		{
			pPed->bUseCrossing = pPed->IsUseCrossing(pPedCrossing);
		}

		// Update Travel Distance
		pPed->fTravelDist = pPed->fTravelDist + pPed->fSpeed * fSimTimeStep;

		// Update position ///////////////////////////////////////////////////////////////////////////////////////////
		// get new position
		Position posNew;
		float lamda = pPed->GetLamda(fSimTimeStep,pMathSolver);
		if(lamda == -1)   // Reached target
		{
			posNew = pPed->posTarget;
		}
		else   // not yet reached target
		{
			// Calculate new position
			posNew=pMathSolver->GetPositionFromLamda(pPed->posCentroid,pPed->posTarget,lamda);
		}

		// Get new position type
		int iPosTypeNew = GetPedPosTypeExpect(pPed,posNew);

		// Get ColID
		int iColID = -1;
		// Find the edge col id
		float fColID = (pPed->posCentroid.x - (vPedODASouth[0]->PosCentroid.x - vPedODASouth[0]->fLength/2))
		                 / ODA_LENGTH;
		if(fColID<0)
			fColID=0;
		if(fColID>9)
			fColID=9;
		iColID = (int)(fColID) + 1;

		// Check if position type will change and record edge row number
		int iRowID = -1;  // No change id=-1
		switch(pPed->iPosType)
		{
			case(PED_POS_PAVE_NEAR):
			case(PED_POS_PAVE_NEAR_EDGE):
			{
				if(iPosTypeNew==PED_POS_CYCLE_NEAR
				   || iPosTypeNew==PED_POS_CYCLE_NEAR_EDGE
				   || iPosTypeNew==PED_POS_VEH_NEAR
				   || iPosTypeNew==PED_POS_MEDIAN
				   || iPosTypeNew==PED_POS_VEH_FAR
				   || iPosTypeNew==PED_POS_CYCLE_FAR
				   || iPosTypeNew==PED_POS_PAVE_FAR)
				{
					if(pPed->iDirection>0)  // S-N
						iRowID = 1;
					else
						iRowID = 5;
				}
				break;
			}
			case(PED_POS_CYCLE_NEAR):
			case(PED_POS_CYCLE_NEAR_EDGE):
			{
				if(iPosTypeNew==PED_POS_VEH_NEAR
				   || iPosTypeNew==PED_POS_MEDIAN
				   || iPosTypeNew==PED_POS_VEH_FAR
				   || iPosTypeNew==PED_POS_CYCLE_FAR
				   || iPosTypeNew==PED_POS_PAVE_FAR)
				{
```

```
                                        if(pPed->iDirection>0)  // S-N
                                                iRowID = 2;
                                        else
                                                iRowID = 4;
                                }
                                break;
                        }
                        case(PED_POS_VEH_NEAR):
                        case(PED_POS_MEDIAN):
                        {
                                if(iPosTypeNew==PED_POS_VEH_FAR
                                  || iPosTypeNew==PED_POS_CYCLE_FAR
                                  || iPosTypeNew==PED_POS_PAVE_FAR)
                                {
                                        if(pPed->iDirection>0)  // S-N
                                                iRowID = 3;
                                        else
                                                iRowID = 3;
                                }
                                break;
                        }
                        case(PED_POS_VEH_FAR):
                        {
                                if(iPosTypeNew==PED_POS_CYCLE_FAR
                                  || iPosTypeNew==PED_POS_PAVE_FAR)
                                {
                                        if(pPed->iDirection>0)  // S-N
                                                iRowID = 4;
                                        else
                                                iRowID = 2;
                                }
                                break;
                        }
                        case(PED_POS_CYCLE_FAR):
                        {
                                if(iPosTypeNew==PED_POS_PAVE_FAR)
                                {
                                        if(pPed->iDirection>0)  // S-N
                                                iRowID = 5;
                                        else
                                                iRowID = 1;
                                }
                                break;
                        }
                        default:
                        {
                                int iRowID = -1;  // No change id = -1
                                break;
                        }
                }
                if(!(iRowID>=1 && iRowID<=5))      // No crossing edge behaviour
                {
                        // Update position
                        pPed->posCentroid = posNew;
                        pPed->bStopDueToFlow = false;
                }

                else  // crossing edge happened, Check edge flow
                {
                        // Pass edge probability
                        float fPassEdge = 0.0f;
                        // Get the Edge
                        Edge* pEdge = vvEdge[iColID-1][iRowID-1];
                        if(pEdge!=NULL)
                        {
                                // Check edge flow
                                if(pEdge->iFlowCurrent < pEdge->iFlowMax)
                                // not exceed max, Allow position change
                                {
                                        // Calculate expected flow if step in
                                        float fExtraFlow = 3600/(pEdge->fLength * fSimTimeStep);
                                        if(pEdge->iFlowCurrent + fExtraFlow <= pEdge->iFlowMax)  // Allow step in
                                        {
                                                fPassEdge = 1.0f;
                                        }
                                        else
                                        {
                                                // Calculate the probability of passing the edge
                                                fPassEdge = (pEdge->iFlowMax - pEdge->iFlowCurrent)
                                                        * pEdge->fLength * fSimTimeStep / 3600;
                                        }

                                        // Decide if pass
                                        bool bPass = pRandomSolver->GetBinary(fPassEdge);
                                        if(bPass)
                                        {
```

```
                                        // Update position
                                        pPed->posCentroid = posNew;
                                        pPed->bStopDueToFlow = false;

                                        // Update edge flow
                                        pEdge->PedPassed++;

                                        pEdge->iFlowCurrent = pEdge->PedPassed  / pEdge->fLength
                                                                / fSimTimeStep  * 3600;
                                        // Update num of peds entered the crossing(if exists)
                                        if(pPedCrossing!=NULL)
                                        {
                                                Position posRectCentroid = pPedCrossing->PosCentroid;
                                                float fLength = pPedCrossing->fLength;
                                                float fHeight = (float)INT_MAX/2;
                                                bool bIn = pMathSolver->IsPosInRectangle
                                                (pPed->posCentroid,posRectCentroid,fLength,fHeight);
                                                if(bIn)
                                                {
                                                        if((pPed->iDirection>0&&iRowID==1)
                                                        ||(pPed->iDirection<0&&iRowID==5))
                                                            pPedCrossing->iPedEntered++;
                                                }
                                        }
                                }
                        }

                        else  // Exceed max flow: no position change, [if in pavement, speed=0;]
                        {
                                pPed->bStopDueToFlow = true;
                                fPassEdge = 0;
                        }

                }
                else
                {
                        // Update position
                        pPed->posCentroid = posNew;
                        pPed->bStopDueToFlow = false;
                }
}
// Update position ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

// Update position type,ColID,RowID and previous position type, ColID, RowID
pPed->iPosTypePrev      = pPed->iPosType;
pPed->iPosType          = GetPedPosType(pPed);
pPed->iColIDPrev        = pPed->iColID;
pPed->iColID            = GetColID(pPed->posCentroid.x, vPedODASouth[0]->PosCentroid.x
                                - vPedODASouth[0]->fLength/2,  ODA_LENGTH);
pPed->iRowIDPrev        = pPed->iRowID;
pPed->iRowID            = GetRowID(pPed);

// Update the edge ped num
pPed->EnterAnEdge(vvEdge);
pPed->LeftAnEdge(vvEdge);
// Update ODA ped num
pPed->EnterAnODA(vPedODASouth,vPedODANorth);
pPed->LeftAnODA(vPedODASouth,vPedODANorth);

// Update current vehicle lane, (NULL, nearside or farside)
pPed->pLaneVehCurrent = pPed->GetVehLaneCurrent();

// Update Time-To-React
pPed->fTimeToReact = pPed->fTimeToReact - fSimTimeStep;

// Update Travel Time
pPed->fTravelTime = pPed->fTravelTime + fSimTimeStep;

// Update vehicle conflict
// Find new conflicting vehicle
Vehicle* pVehicle = pPed->GetConflictVehInThisLane(pPed->pLaneVehCurrent,pMathSolver);
if(pVehicle==NULL)  // No conflicting vehicle
{
        if(pPed->pVehConflict!=NULL)
        {
                // Remove this ped from the conflicting veh's conflicting peds list
                int iSize = pPed->pVehConflict->vPedConflict.size();
                for(int i=0;i<iSize;i++)
                {
                        if(pPed->pVehConflict->vPedConflict[i]->iID == pPed->iID)
                        {
                                pPed->pVehConflict->vPedConflict.erase
                                (pPed->pVehConflict->vPedConflict.begin()+i);
                                break;
                        }
                }
```

```
					// Update the current conflicting vehicle
					pPed->pVehConflict = NULL;
				}
			}
			else  // Found a conflicting vehicle
			{
				if(pPed->pVehConflict==NULL)  // Conflicting veh changed from NULL
				{
					pPed->pVehConflict = pVehicle;
					pPed->pVehConflict->vPedConflict.push_back(pPed);
				}
				else
				{
					if(pVehicle->iID != pPed->pVehConflict->iID)  // Conflicting veh changed
					{
						// Remove this ped from the conflicting veh's conflicting peds list
						int iSize = pPed->pVehConflict->vPedConflict.size();
						for(int i=0;i<iSize;i++)
						{
							if(pPed->pVehConflict->vPedConflict[i]->iID == pPed->iID)
							{
								pPed->pVehConflict->vPedConflict.erase
								(pPed->pVehConflict->vPedConflict.begin()+i);
								break;
							}
						}
						// Update the current conflicting vehicle
						pPed->pVehConflict = pVehicle;
						pPed->pVehConflict->vPedConflict.push_back(pPed);
					}
				}
			}
		}
		// Update Waiting time,TravelDist,position,position type,Time-to-React,TravelTime,
		// Current Vehicle lane, VehConflict
		////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

		//****************************************************************************************************************
		// Update velocity: 1. direction point; 2. speed rate
		for(i=0;i<iSize;i++)
		{
			Pedestrian* pPed = vPedestrian[i];
//			if(!(pPed->bStopDueToFlow&&pPed->fSpeed==0))
//			{
				if(pPed->fTimeToReact - fSimTimeStep < 0)    // It is time to make reaction
				{
					pPed->fTimeToReact = pPed->fReactionTime;    // Reset Time-to-react
					// Update velocity
					// update its median info first
					int iColID = this->GetColID(pPed->posCentroid.x,
							vPedODASouth[0]->PosCentroid.x - vPedODASouth[0]->fLength/2,
									ODA_LENGTH);
					if(vvEdge[iColID-1][2]->iPedExist>=vvEdge[iColID-1][2]->iPedExistMax)
					{
						pPed->bMedianAvailable = false;
					}
					else
					{
						pPed->bMedianAvailable = true;
					}
					// Then update velocity
					pPed->UpdateVelocity(fSimTimeStep,pMathSolver,vPedestrian,pRandomSolver);
				}
//			}
		}
		// Update velocity: 1. direction point; 2. speed rate
		//****************************************************************************************************************
}

void CPVIView::PedDelete()   // To be modified
{
		int iSize = vPedestrian.size();
		for(int i=0;i<iSize;i++)
		{
			Pedestrian* pPed = vPedestrian[i];
			if(  (pPed->posCentroid.x == pPed->posDest.x && pPed->posCentroid.y == pPed->posDest.y)
				|| ((pPed->fSurvivalTime > PED_SURVIVAL_TIME_MAX)&&((pPed->posCentroid.y
					<= pPed->pMedian->posStart.y * pPed->iDirection)))  )

			{
				if(pPed->fSurvivalTime > PED_SURVIVAL_TIME_MAX)
					pPed->bTimeOut = true;

				// Export the overall stats of this Ped to Class PedOverall
				RecordPedOverall(pPed);
```

```
					// Modify conflicting vehicle
					if(pPed->pVehConflict!=NULL)
					{
						// Remove this ped from the conflicting veh's conflicting peds list
						int iSize = pPed->pVehConflict->vPedConflict.size();
						for(int i=0;i<iSize;i++)
						{
							if(pPed->pVehConflict->vPedConflict[i]->iID == pPed->iID)
							{
								pPed->pVehConflict->vPedConflict.erase
								(pPed->pVehConflict->vPedConflict.begin()+i);
								break;
							}
						}
					}

					// Modify ODA existing ped number
					pPed->pPedODPair->pDest->iPedExisting--;

					// Delete this pedestrian object
					delete pPed;
					pPed = NULL;
					vPedestrian.erase(vPedestrian.begin()+i);
					i--;
					iSize--;
				}
		}
}

int CPVIView::GetPedPosTypeExpect(Pedestrian* pPed, Position posExpected)
{
		int iType;
		if(pPed->posOri.y < 0)   // S-N
		{
				if(posExpected.y <= pPed->pPaveNear->fEdgeNorth)
				{
					if(posExpected.y + pPed->fSpeedDSR * pPed->fReactionTime <=
					pPed->pPaveNear->fEdgeNorth)
						iType = PED_POS_PAVE_NEAR;
					else
						iType = PED_POS_PAVE_NEAR_EDGE;
				}
				else if(posExpected.y <= pPed->pLaneCycleNear->fEdgeNorth)
				{
					if(posExpected.y + pPed->fSpeedDSR * pPed->fReactionTime <=
					pPed->pLaneCycleNear->fEdgeNorth)
						iType = PED_POS_CYCLE_NEAR;
					else
						iType = PED_POS_CYCLE_NEAR_EDGE;
				}
				else if(posExpected.y < pPed->pLaneVehNear->fEdgeNorth)
				{
					iType = PED_POS_VEH_NEAR;
				}
				else if(posExpected.y <= pPed->pMedian->fEdgeNorth)
				{
					iType = PED_POS_MEDIAN;
				}
				else if(posExpected.y < pPed->pLaneVehFar->fEdgeNorth)
				{
					iType = PED_POS_VEH_FAR;
				}
				else if(posExpected.y < pPed->pLaneCycleFar->fEdgeNorth)
				{
					iType = PED_POS_CYCLE_FAR;
				}
				else
				{
					iType = PED_POS_PAVE_FAR;
				}
		}
		else   // N-S
		{
				if(posExpected.y >= pPed->pPaveNear->fEdgeSouth)
				{
					if(posExpected.y - pPed->fSpeedDSR * pPed->fReactionTime >=
					pPed->pPaveNear->fEdgeSouth)
						iType = PED_POS_PAVE_NEAR;
					else
						iType = PED_POS_PAVE_NEAR_EDGE;
				}
				else if(posExpected.y >= pPed->pLaneCycleNear->fEdgeSouth)
				{
					if(posExpected.y - pPed->fSpeedDSR * pPed->fReactionTime >=
					pPed->pLaneCycleNear->fEdgeSouth)
						iType = PED_POS_CYCLE_NEAR;
```

```
					else
						iType = PED_POS_CYCLE_NEAR_EDGE;
				}
				else if(posExpected.y > pPed->pLaneVehNear->fEdgeSouth)
				{
					iType = PED_POS_VEH_NEAR;
				}
				else if(posExpected.y >= pPed->pMedian->fEdgeSouth)
				{
					iType = PED_POS_MEDIAN;
				}
				else if(posExpected.y > pPed->pLaneVehFar->fEdgeSouth)
				{
					iType = PED_POS_VEH_FAR;
				}
				else if(posExpected.y > pPed->pLaneCycleFar->fEdgeSouth)
				{
					iType = PED_POS_CYCLE_FAR;
				}
				else
				{
					iType = PED_POS_PAVE_FAR;
				}
		}

		return iType;
}

void CPVIView::ResetEdge()
{

		int iSize = vvEdge.size();
		for(int i=0;i<iSize;i++)
		{
				int iSize2 = vvEdge[i].size();
				for(int j=0;j<iSize2;j++)
				{
					Edge* pEdge = vvEdge[i][j];
					pEdge->PedPassed	= 0;
					pEdge->iFlowCurrent = 0;
				}
		}
}

int CPVIView::GetColID(float xPed, float xStart, float fUnitLength)
{
		// Get ColID
		int iColID = -1;
		// Find the edge col id
		float fColID = (xPed - xStart) / fUnitLength;

		if(fColID<0)
				fColID=0;
		if(fColID>9)
				fColID=9;
		iColID = (int)(fColID) + 1;
		return iColID;
}

int CPVIView::GetRowID(Pedestrian* pPed)
{
		int iRowID = -1;

		switch(pPed->iPosType)
		{
				case(PED_POS_PAVE_NEAR_EDGE):
				{
					iRowID = 3 - pPed->iDirection*2;
					break;
				}
				case(PED_POS_CYCLE_NEAR_EDGE):
				{
					iRowID = 3 - pPed->iDirection;
					break;
				}
				case(PED_POS_MEDIAN):
				{
					iRowID = 3;
					break;
				}
				default:
				{
					iRowID = -1;
				}
		}
```

```
		return iRowID;
}

void CPVIView::ResetSimWorld()
{
		// reset vehicles lanes
		int iSize = vLaneVeh.size();
		for(int i=0;i<iSize;i++)
		{
			vLaneVeh[i]->fDemand = vLaneVeh[i]->fDemandInit * this->fVehDemandPercent;
			vLaneVeh[i]->iVehiclesAll = 0;
			vLaneVeh[i]->iVehiclesGenerated = 0;
			vLaneVeh[i]->vGlobalGeneratingTimes.clear();
			vLaneVeh[i]->vInitialHeadways.clear();
			// Clear vehicles left in this lane
			int iVehNum = vLaneVeh[i]->vVehicles.size();
			for(int j=0;j<iVehNum;j++)
			{
				delete vLaneVeh[i]->vVehicles[j];
			}
			vLaneVeh[i]->vVehicles.clear();
		}

		// reset world time
		this->fSimTimeElapse = 0.0f;
		fSimTimeWhole = (float)SAMPLE_SIZE / (pLaneVehS->fDemand + pLaneVehN->fDemand) * 3600;

		// Reset random seed
		DRandomSeed		= GetTickCount();
		srand(DRandomSeed);

		// Reset number of objects generated
		iNumVehGenerated = 0;
		iNumPedGenerated = 0;

		// re populate headways and global generating times
		iSize = vLaneVeh.size();
		for(i=0;i<iSize;i++)
		{
			float fAccumulativeTimeHeadway = 0.0f;
			while (fAccumulativeTimeHeadway < fSimTimeWhole)
			{
				float fTimeHeadway = pRandomSolver->GetInitTimeHeadwayVeh(vLaneVeh[i]->fDemand);
				vLaneVeh[i]->vInitialHeadways.push_back(fTimeHeadway);
				fAccumulativeTimeHeadway = fAccumulativeTimeHeadway + fTimeHeadway;
			}
			if (fAccumulativeTimeHeadway > fSimTimeWhole)
				vLaneVeh[i]->vInitialHeadways.pop_back();
			vLaneVeh[i]->GetGlobalGeneratingTimes();
			vLaneVeh[i]->iVehiclesAll = vLaneVeh[i]->vGlobalGeneratingTimes.size();
		}

		// Clear pedestrians left
		iSize = vPedestrian.size();
		for(i=0;i<iSize;i++)
		{
			delete vPedestrian[i];
		}
		vPedestrian.clear();

		// Reset Ped ODAs
		// South
		iSize = vPedODASouth.size();
		for(i=0;i<iSize;i++)
		{
			vPedODASouth[i]->iPedExisting = 0;
			vPedODASouth[i]->iPedGened = 0;
		}
		// North
		iSize = vPedODANorth.size();
		for(i=0;i<iSize;i++)
		{
			vPedODANorth[i]->iPedExisting = 0;
			vPedODANorth[i]->iPedGened = 0;
		}

		// Reset Ped ODPairs
		iSize = vPedODPair.size();
		for(i=0;i<iSize;i++)
		{
			vPedODPair[i]->fDemand = vPedODPair[i]->fDemandInit * this->fPedDemandPercent;
			vPedODPair[i]->iPedGened = 0;
			vPedODPair[i]->iPedTotal = 0;
			vPedODPair[i]->vApprIntervals.clear();
			vPedODPair[i]->vGlobalGenTimes.clear();
			// Repopulate
```

```
			vPedODPair[i]->GetApprTimeIntervals(pRandomSolver,vPedODPair[i]->fDemand,fSimTimeWhole);
			vPedODPair[i]->GetGlobalGenTimes();
			vPedODPair[i]->iPedTotal = vPedODPair[i]->vGlobalGenTimes.size();
		}

		// Reset vPedOverall
		iSize = vPedOverall.size();
		for(i=0;i<iSize;i++)
		{
			delete vPedOverall[i];
		}
		vPedOverall.clear();

		// Reset vVehOverall
		iSize = vVehicleOverall.size();
		for(i=0;i<iSize;i++)
		{
			delete vVehicleOverall[i];
		}
		vVehicleOverall.clear();

		// Reset vSignalDynamics
		iSize = vSignalDynamics.size();
		for(i=0;i<iSize;i++)
		{
			delete vSignalDynamics[i];
		}
		vSignalDynamics.clear();

		// Reset vVehicleDynamics
		iSize = vVehicleDynamics.size();
		for(i=0;i<iSize;i++)
		{
			delete vVehicleDynamics[i];
		}
		vVehicleDynamics.clear();

		// Reset vvEdge
		iSize = vvEdge.size();
		for(i=0;i<iSize;i++)
		{
			int iSizeCol = vvEdge[i].size();
			for(int j=0;j<iSizeCol;j++)
			{
				vvEdge[i][j]->iFlowCurrent = 0;
				vvEdge[i][j]->iPedExist = 0;
				vvEdge[i][j]->PedPassed = 0;
			}
		}

		// Reset PedCrossing
		if(pPedCrossing!=NULL)
		{
			pPedCrossing->fCurrentPeriodElapseTime = 0;
			pPedCrossing->fCurrentPeriodStartTime = 0;
			pPedCrossing->iCurrentPeriod = 1;
			pPedCrossing->iPedEntered = 0;
		}
}


#ifndef PVI__CAR_FOLLOWING_MODEL_FUZZY_H
#define PVI__CAR_FOLLOWING_MODEL_FUZZY_H

#define CAR_FOLLOWING_FUZZY_DV_MIN		-8
#define CAR_FOLLOWING_FUZZY_DV_MAX		 8
#define CAR_FOLLOWING_FUZZY_DSSD_MIN		0.35
#define CAR_FOLLOWING_FUZZY_DSSD_MAX		1.65

#include "FuzzySolver.h"
#include <vector>
using namespace std;


class CarFollowingModel_Fuzzy
{
public:
		FIS*			fis;
		double**		fisMatrix;
		double**		dataMatrix;
		double**		outputMatrix;
		char*			fis_file;
		int			fis_row_n;
		int			fis_col_n;
		int			data_row_n;
		int			data_col_n;
```

```
	FuzzySolver*		pFuzzySolver;

public:
	CarFollowingModel_Fuzzy(FuzzySolver* pFuzzy);
	virtual ~CarFollowingModel_Fuzzy();
	double GetAcceleration(double dDV, double dDSSD);

};
#endif


#include "stdafx.h"
#include "PVI.h"
#include "CarFollowingModel_Fuzzy.h"

CarFollowingModel_Fuzzy::CarFollowingModel_Fuzzy(FuzzySolver* pFuzzy)
{
	pFuzzySolver = pFuzzy;

	/* Read the fuzzy model file */
	fis_file	= "Models/dvdssdnor.fis";
	fisMatrix	= pFuzzySolver->returnFismatrix(fis_file, &fis_row_n, &fis_col_n);

	/* build FIS data structure */
	fis = (FIS *)pFuzzySolver->fisCalloc(1, sizeof(FIS));
	pFuzzySolver->fisBuildFisNode(fis, fisMatrix, fis_col_n, MF_POINT_N);

	data_row_n = 1;
	data_col_n = 2;

}

double CarFollowingModel_Fuzzy::GetAcceleration(double dDV, double dDSSD)
{

	// Inputs validity check
	if (dDV < CAR_FOLLOWING_FUZZY_DV_MIN)
		dDV = CAR_FOLLOWING_FUZZY_DV_MIN;
	else if (dDV > CAR_FOLLOWING_FUZZY_DV_MAX)
		dDV = CAR_FOLLOWING_FUZZY_DV_MAX;
	if (dDSSD < CAR_FOLLOWING_FUZZY_DSSD_MIN)
		dDSSD = CAR_FOLLOWING_FUZZY_DSSD_MIN;
	else if (dDSSD > CAR_FOLLOWING_FUZZY_DSSD_MAX)
		dDSSD = CAR_FOLLOWING_FUZZY_DSSD_MAX;

	// Build input data matrix
	vector<double>*		pvFIS_Input = new vector<double>;
	pvFIS_Input->push_back(dDV);
	pvFIS_Input->push_back(dDSSD);
	dataMatrix = pFuzzySolver->returnDataMatrix(pvFIS_Input, &data_row_n, &data_col_n);

	/* create output matrix */
	outputMatrix = (DOUBLE **) pFuzzySolver->fisCreateMatrix(data_row_n, fis->out_n, sizeof(DOUBLE));

	/* evaluate FIS the input vector */
	pFuzzySolver->getFisOutput(dataMatrix[0], fis, outputMatrix[0]);

	return outputMatrix[0][0];
}


#ifndef PVI__PEDESTRIAN_H
#define PVI__PEDESTRIAN_H
#include "Global.h"
#include "Position.h"
#include "Pavement.h"
#include "LaneCycle.h"
#include "Median.h"
#include "PedODPair.h"
#include "MathSolver.h"
#include "RandomSolver.h"
#include "Vehicle.h"
#include "LaneVeh.h"
#include "PedCrossing.h"
#include "Edge.h"
#include "PedODA.h"

class Pedestrian
{
public:
	unsigned int	iID;		// Ped ID, total number of peds gened
	float		fBodyRadius;	// Ped Body Radius, m
	int		iType;		// YM, YF, OM, OF
	Position	posCentroid;	// Current coordinates of the ped
	int		iPosType;	// Position Type;
	Position	posOri;		// The coordinates of the origin point
```

```
        Position            posDest;              // The coordinates of the destination point
        Position            posDestTemp;          // The coordinates of the temp destination point
        Position            posTarget;            // The coordinates of the target point,
                                                  // used to indicate the direction of velocity
        int                 iDirection;           // Direction vector, 1=S->N; -1=N->S
        float               fSpeed;               // Current speed, m/s
        float               fSpeedDSR;            // Desired speed, m/s
        float               fSpeedMaxWalk;        // Maximum walking speed, m/s
        float               fSpeedMaxRun;         // Maximum running speed, m/s
        LaneVeh*            pLaneVehNear;         // The nearside vehicle lane
        LaneVeh*            pLaneVehFar;          // The farside vehicle lane
        Pavement*           pPaveNear;            // The nearside pavement
        Pavement*           pPaveFar;
        LaneCycle*          pLaneCycleNear;       // The nearside cycle lane
        LaneCycle*          pLaneCycleFar;
        Median*             pMedian;              // The median
        float               fReactionTime;        // Reaction time, s
        float               fTimeToReact;         // Time to make next reaction,
        PedODPair*          pPedODPair;           // Belongs to which Ped OD Pair
        bool                bUseCrossing;         // 0=not use; 1=use
        float               fWaitTime;
        float               fGroupingDist;
        float               fTravelTimeDSR;       // = OD distance/desired speed, s
        float               fTravelTime;          // Total time the ped has travelled
        float               fTravelDist;          // Total distance travelled
        float               fNearsideTTC;         // Initial TTC of Nearside lane
        float               fFarsideTTC;          // Initial TTC of Farside lane
        Vehicle*            pVehConflict;         // Current conflicting veh when ped in veh lane
        LaneVeh*            pLaneVehCurrent;      // Currently in which vehicle lane
        Vehicle*            pVehConflictTemp;     // Used for PGA
        bool                bVehBlockTemp;        // Used for PGA
        PedCrossing*        pPedCrossing;
        Position            posWaitingZoneNear;   // the nearside waiting zone coordinates to this ped
        PedWaitingZone*     pPedWaitingZoneNear;  // The nearside waiting zone
        Position            posWaitingZoneFar;    // the farside waiting zone coordinates to this ped
        PedWaitingZone*     pPedWaitingZoneFar ;  // The farside waiting zone
        bool                bStopDueToFlow;
        int                 iPosTypePrev;         // Position type in previous simstep
        bool                bMedianAvailable;     // Is median available
        int                 iColID;               // in which col
        int                 iColIDPrev;           // in which col previously
        int                 iRowID;
        int                 iRowIDPrev;
        bool                bTimeOut;             // Time-out due to exceed max suvival time
        float               fSurvivalTime;        // how much time the ped has already in the world

public:
        Pedestrian();
        virtual ~Pedestrian();

        // GetLamda, lamda=-1 means this pedestrian has reached target point
        float GetLamda(float fTime, MathSolver* pMathSolver);

        // Update velocity: 1. Target Position; 2. Speed Rate
        void UpdateVelocity(float fSimTimeStep, MathSolver* pMathSolver,
        vector<Pedestrian*> vPedestrian, RandomSolver* pRandomSolver);

        // Get a time gap from a vehicle lane, given a current pedestrian position and target lane
        // the pedestrian has not been crossing this lane yet
        // pVehConflict is used to store the conflicting vehicle in this lane
        // bVehBlock is true if the conflicting veh is a blocking vehicle
        float GetTrafficTimeGap(LaneVeh* pLaneVeh);

        // Get minimum needed crossing time
        float GetCrossingTimeMin(LaneVeh* pLaneVeh);

        // Check if the gap is accepted
        bool GapAcceptance(LaneVeh* pLaneVeh, float fTimeGap, vector<Pedestrian*> vPedestrian,
                            RandomSolver* pRandomSolver, MathSolver* pMathSolver);

        // Get number of peds in a group
        int GetGroupNum(vector<Pedestrian*> vPedestrian, MathSolver* pMathSolver);

        // Get the conflicting vehicle in the current lane when the pedestrian is in this veh lane
        Vehicle* GetConflictVehInThisLane(LaneVeh* pLaneVeh, MathSolver* pMathSolver);

        // Get the right crossing speed in order not to collide with conflicting vehicle
        float GetCrossingSpeed(LaneVeh* pLaneVeh, MathSolver* pMathSolver);

        // Get current vehicle lane
        LaneVeh* GetVehLaneCurrent();

        // Get the distance to the nearby crossing facility (to crossing central line)
        float GetDistanceToCrossing(PedCrossing* pPedCrossing);

        // Is use crossing
```

```
        bool IsUseCrossing(PedCrossing* pPedCrossing);

        // Is in nearside waiting area
        bool IsInWaitingZoneNear(PedCrossing* pPedCrossing,MathSolver* pMathSolver);

        // Return the pedestrian destination position towards a line y=y0
        Position GoToLine(float fLineY);

        // Is reached target position
        bool IsReachedTargetPos();

        // entered a new edge, ped num on that edge  +1
        bool EnterAnEdge(vector<vector<Edge*> >  vvEdge);

        // left a edge, ped num on that edge  -1
        bool LeftAnEdge(vector<vector<Edge*> >  vvEdge);

        // entered a new ODA, ped num on that ODA  +1
        bool EnterAnODA(vector<PedODA*> vPedODASouth, vector<PedODA*> vPedODANorth);

        // left a ODA, ped num on that ODA  -1
        bool LeftAnODA(vector<PedODA*> vPedODASouth, vector<PedODA*> vPedODANorth);
};

#endif


bool Pedestrian::GapAcceptance(LaneVeh* pLaneVeh, float fTimeGap, vector<Pedestrian*> vPedestrian,
                               RandomSolver* pRandomSolver, MathSolver* pMathSolver)
{
        bool bAccept;
        // Get the minimum crossing time needed
        float fCrossingTimeMin = this->GetCrossingTimeMin(pLaneVeh);
        if(fTimeGap <= fCrossingTimeMin)  // cannot cross
        {
                bAccept = false;
        }
        else  // use PGA model
        {
                int iAgeGroup = (this->iType==PED_TYPE_YM || this->iType==PED_TYPE_YF)?0:1;
                int iPedNum = this->GetGroupNum(vPedestrian,pMathSolver);
                float fUtility = PED_PGA_NOCONTROL_CONSTANT
                                        + PED_PGA_NOCONTROL_AGE * iAgeGroup
                                        + PED_PGA_NOCONTROL_PEDNUM * iPedNum
                                        + PED_PGA_NOCONTROL_NEARGAP * fTimeGap;
                float fAcceptProb = 1 / (1 + exp(-1*fUtility));
                bAccept = pRandomSolver->GetBinary(fAcceptProb);
        }
        return bAccept;
}

Vehicle* Pedestrian::GetConflictVehInThisLane(LaneVeh* pLaneVeh, MathSolver* pMathSolver)
{
        Vehicle* pVehicleConflict = NULL;

        if(pLaneVeh!=NULL)
        {
                // Get Lane direction vector: 1=W->E; -1=E->W
                int iLaneDirection = pLaneVeh->posEnd.x - pLaneVeh->posStart.x;
                iLaneDirection = abs(iLaneDirection) / iLaneDirection;

                // Get Lane Type: 1=nearside; -1=farside
                int iLaneType;
                if(this->iDirection * iLaneDirection >0)
                        iLaneType = 1;
                else
                        iLaneType = -1;

                // Check all vehicles in this lane
                int iNumVehicle = pLaneVeh->vVehicles.size();
                for(int i=0;i<iNumVehicle;i++)
                {
                        Vehicle* pVehicle = pLaneVeh->vVehicles[i];
                        // Calculate vehicle current position
                        float fDeltaXFront = pVehicle->fDeltaX;
                        float fDeltaXRear  = pVehicle->fDeltaX - pVehicle->fLength;
                        Position posVehFront = pVehicle->GetPosition2(fDeltaXFront);
                        Position posVehRear        = pVehicle->GetPosition2(fDeltaXRear);
                        // Decide the conflicting vehicle
                        // Conflicting
                        if((posVehFront.x - this->posCentroid.x) * this->iDirection * iLaneType <= 0)
                        {
                                pVehicleConflict = pVehicle;
                                break;
                        }
                        // Blocking
```

```
                        else if((posVehRear.x - this->posCentroid.x) * this->iDirection * iLaneType < 0)
                        {
                                pVehicleConflict = pVehicle;
                                break;
                        }
                }
        }
        return pVehicleConflict;
}

float Pedestrian::GetCrossingSpeed(LaneVeh* pLaneVeh, MathSolver* pMathSolver)
{
        float fCrossingSpeed = this->fSpeedDSR;

        if(this->pVehConflict!=NULL)
        {
                // Get Lane direction vector: 1=W->E; -1=E->W
                int iLaneDirection = pLaneVeh->posEnd.x - pLaneVeh->posStart.x;
                iLaneDirection = abs(iLaneDirection) / iLaneDirection;

                // Get Lane Type: 1=nearside; -1=farside
                int iLaneType;
                if(this->iDirection * iLaneDirection >0)
                        iLaneType = 1;
                else
                        iLaneType = -1;

                // Calculate the TTC
                float fTTC;

                // Get front centriod position of the conflicting vehicle
                Position posVehFront = this->pVehConflict->GetPosition2(this->pVehConflict->fDeltaX);

                // Decide the type of the conflicting vehicle
                if((posVehFront.x-this->posCentroid.x) * this->iDirection * iLaneType <= 0) // Conflicting
                {
                        if(this->pVehConflict->fSpeed>0)
                                fTTC = fabs(posVehFront.x - this->posCentroid.x) / this->pVehConflict->fSpeed;
                        else
                                fTTC = FLT_MAX;
                }
                else  // Blocking
                {
                        fTTC = 0;  // means blocking
                }

                // Calculate speed needed for the pedestrian to get out of the lane
                float fDistToGo = pMathSolver->GetDistance(this->posCentroid,this->posTarget);
                float fSpeedNeed;
                if(fTTC > 0)
                {
                        fSpeedNeed = fDistToGo / fTTC;
                }
                else
                {
                        fSpeedNeed = this->fSpeedMaxRun;
                }

                // Update crossing speed
                if(fSpeedNeed > this->fSpeedMaxRun)
                {
                        fCrossingSpeed = this->fSpeedMaxRun;
                }
                else if(fSpeedNeed > this->fSpeedDSR)
                {
                        fCrossingSpeed = fSpeedNeed;
                }
        }

        return fCrossingSpeed;
}

float Pedestrian::GetTrafficTimeGap(LaneVeh* pLaneVeh)
{

        float fTimeGap= (float)INT_MAX;
        // Get Lane direction vector: 1=W->E; -1=E->W
        int iLaneDirection = pLaneVeh->posEnd.x - pLaneVeh->posStart.x;
        iLaneDirection = abs(iLaneDirection) / iLaneDirection;
        int iLaneType;  // 1=nearside lane; -1=farside lane

        if(this->iDirection * iLaneDirection >0)  // This is a nearside vehicle lane
        {
                iLaneType = 1;
                switch(this->iPosType)
                {
```

```
case(PED_POS_PAVE_NEAR):
case(PED_POS_PAVE_NEAR_EDGE):
case(PED_POS_CYCLE_NEAR):
case(PED_POS_CYCLE_NEAR_EDGE):  // Cacculate the expected conflicting vehicle
{
        // Calculate time gap
        // Get the time cost for the pedestrian to reach the lane nearside edge
        float fDist = fabs( (pLaneVeh->posStart.y - pLaneVeh->fWidth/2*this->iDirection)
                                        - this->posCentroid.y );
        float fTime = fDist / this->fSpeedDSR;

        // Check all vehicles in this lane
        int iNumVehicle = pLaneVeh->vVehicles.size();
        for(int i=0;i<iNumVehicle;i++)
        {
                Vehicle* pVehicle = pLaneVeh->vVehicles[i];
                // Calculate vehicle position after fTime with current instant speed
                float fDeltaXFront = pVehicle->fDeltaX + pVehicle->fSpeed * fTime;
                float fDeltaXRear = pVehicle->fDeltaX + pVehicle->fSpeed * fTime
                                        - pVehicle->fLength;
                Position posVehFront = pVehicle->GetPosition2(fDeltaXFront);
                Position posVehRear = pVehicle->GetPosition2(fDeltaXRear);

                // Find the conflicting vehicle
                if( (this->posCentroid.x - posVehFront.x) * this->iDirection >  this->fBodyRadius)
                // Conflicting
                {
                        // Record the conflicting vehicle
                        this->pVehConflictTemp  = pVehicle;
                        this->bVehBlockTemp     = false;

                        // Calculate the time gap
                        if(pVehicle->fSpeed > 0)
                        {
                                fTimeGap = ( (this->posCentroid.x - posVehFront.x) *
                                this->iDirection- this->fBodyRadius)/ pVehicle-fSpeed;
                        }
                        else if(pVehicle->fAcc > 0)
                        {
                                fTimeGap = sqrt( 2 * ((this->posCentroid.x -
                                posVehFront.x) * this->iDirection - this->fBodyRadius)
                                / pVehicle->fAcc );
                        }
                        else
                        {
                                fTimeGap = (float)INT_MAX;
                        }
                        break;
                }
                else if( (this->posCentroid.x - posVehRear.x) * this->iDirection >
                        (-1) * this->fBodyRadius )
                // Blocking
                {
                        this->pVehConflictTemp  = pVehicle;
                        this->bVehBlockTemp     = true;
                        fTimeGap = 0.0f;
                        break;
                }
        }
        break;
}
case(PED_POS_VEH_NEAR):  // Caculate the direct conflicting vehicle
{
        // Calculate time gap
        // Get the time cost for the pedestrian to reach the lane nearside edge
        float fTime = 0.0f;

        // Check all vehicles in this lane
        int iNumVehicle = pLaneVeh->vVehicles.size();
        for(int i=0;i<iNumVehicle;i++)
        {
                Vehicle* pVehicle = pLaneVeh->vVehicles[i];
                // Calculate vehicle position after fTime with current instant speed
                float fDeltaXFront       = pVehicle->fDeltaX + pVehicle->fSpeed * fTime;
                float fDeltaXRear        =pVehicle->fDeltaX+pVehicle->fSpeed
                                           *fTime-pVehicle->fLength;
                Position posVehFront     = pVehicle->GetPosition2(fDeltaXFront);
                Position posVehRear      = pVehicle->GetPosition2(fDeltaXRear);

                // Find the conflicting vehicle
                if((this->posCentroid.x - posVehFront.x) * this->iDirection >
                        this->fBodyRadius)
                // Conflicting
                {
                        // Record the conflicting vehicle
                        this->pVehConflictTemp  = pVehicle;
```

```
                                        this->bVehBlockTemp = false;

                                        // Calculate the time gap
                                        if(pVehicle->fSpeed > 0)
                                        {
                                                fTimeGap = ((this->posCentroid.x - posVehFront.x)
                                                * this->iDirection - this->fBodyRadius)
                                                   / pVehicle->fSpeed;
                                        }
                                        else if(pVehicle->fAcc > 0)
                                        {
                                                fTimeGap = sqrt( 2 * ((this->posCentroid.x
                                                 - posVehFront.x)  * this->iDirection
                                                - this->fBodyRadius)  / pVehicle->fAcc );
                                        }
                                        else
                                        {
                                                fTimeGap = (float)INT_MAX;
                                        }
                                        break;
                                }
                                else if( (this->posCentroid.x - posVehRear.x) * this->iDirection >
                                         (-1) * this->fBodyRadius )
                                // Blocking
                                {
                                        // Record the blocking vehicle
                                        this->pVehConflictTemp = pVehicle;
                                        this->bVehBlockTemp                     = true;
                                        fTimeGap = 0.0f;
                                        break;
                                }
                        }
                        break;
                }
                default:  // N/A
                {
                        break;
                }
        }
}
else  // This is a farside vehicle lane
{
        iLaneType = -1;
        switch(this->iPosType)
        {
                case(PED_POS_PAVE_NEAR):
                case(PED_POS_PAVE_NEAR_EDGE):
                case(PED_POS_CYCLE_NEAR):
                case(PED_POS_CYCLE_NEAR_EDGE):
                case(PED_POS_VEH_NEAR):
                case(PED_POS_MEDIAN):  // Cacculate the expected conflicting vehicle
                {
                        // Calculate time gap
                        // Get the time cost for the pedestrian to reach the lane nearside edge
                        float fDist = fabs( (pLaneVeh->posStart.y - pLaneVeh->fWidth/2*this->iDirection)
                                                        - this->posCentroid.y );
                        float fTime = fDist / this->fSpeedDSR;

                        // Check all vehicles in this lane
                        int iNumVehicle = pLaneVeh->vVehicles.size();
                        for(int i=0;i<iNumVehicle;i++)
                        {
                                Vehicle* pVehicle = pLaneVeh->vVehicles[i];
                                // Calculate vehicle position after fTime with current instant speed
                                float fDeltaXFront = pVehicle->fDeltaX + pVehicle->fSpeed * fTime;
                                float fDeltaXRear = pVehicle->fDeltaX + pVehicle->fSpeed * fTime
                                                        -pVehicle->fLength;
                                Position posVehFront = pVehicle->GetPosition2(fDeltaXFront);
                                Position posVehRear = pVehicle->GetPosition2(fDeltaXRear);

                                // Find the conflicting vehicle
                                if( (this->posCentroid.x - posVehFront.x) * this->iDirection <
                                        (-1) * this->fBodyRadius)
                                // Conflicting
                                {
                                        // Record the conflicting vehicle
                                        this->pVehConflictTemp = pVehicle;
                                        this->bVehBlockTemp     = false;

                                        // Calculate the time gap
                                        if(pVehicle->fSpeed > 0)
                                        {
                                                fTimeGap = ((this->posCentroid.x - posVehFront.x)
                                                * this->iDirection* (-1) - this->fBodyRadius )
                                                / pVehicle->fSpeed;
                                        }
```

```
                else if(pVehicle->fAcc > 0)
                {
                        fTimeGap = sqrt( 2 * ((this->posCentroid.x
                        - posVehFront.x) * this->iDirection * (-1)
                        - this->fBodyRadius) / pVehicle->fAcc );
                }
                else
                {
                        fTimeGap = (float)INT_MAX;
                }
                break;
        }
        else if( (this->posCentroid.x - posVehRear.x) * this->iDirection <
          this->fBodyRadius )
        // Blocking
        {
                // Record the blocking vehicle
                this->pVehConflictTemp  = pVehicle;
                this->bVehBlockTemp     = true;

                fTimeGap = 0.0f;
                break;
        }
    }
    break;
}
case(PED_POS_VEH_FAR):  // Caculate the direct conflicting vehicle
{
    // Calculate time gap
    // Get the time cost for the pedestrian to reach the lane nearside edge
    float fTime = 0.0f;

    // Check all vehicles in this lane
    int iNumVehicle = pLaneVeh->vVehicles.size();
    for(int i=0;i<iNumVehicle;i++)
    {
        Vehicle* pVehicle = pLaneVeh->vVehicles[i];
        // Calculate vehicle position after fTime with current instant speed
        float fDeltaXFront = pVehicle->fDeltaX + pVehicle->fSpeed * fTime;
        float fDeltaXRear = pVehicle->fDeltaX + pVehicle->fSpeed * fTime
                                - pVehicle->fLength;
        Position posVehFront      = pVehicle->GetPosition2(fDeltaXFront);
        Position posVehRear       = pVehicle->GetPosition2(fDeltaXRear);

        // Find the conflicting vehicle
        if( (this->posCentroid.x - posVehFront.x) * this->iDirection <
                (-1) * this->fBodyRadius)
        // Conflicting
        {
                // Record the conflicting vehicle
                this->pVehConflictTemp  = pVehicle;
                this->bVehBlockTemp     = false;

                // Calculate the time gap
                if(pVehicle->fSpeed > 0)
                {
                        fTimeGap = ((this->posCentroid.x - posVehFront.x)
                        * this->iDirection * (-1) - this->fBodyRadius)
                          / pVehicle->fSpeed;
                }
                else if(pVehicle->fAcc > 0)
                {
                        fTimeGap = sqrt( 2 * ((this->posCentroid.x
                        - posVehFront.x) * this->iDirection * (-1)
                        - this->fBodyRadius) / pVehicle->fAcc );
                }
                else
                {
                        fTimeGap = (float)INT_MAX;
                }
                break;
        }
        else if( (this->posCentroid.x - posVehRear.x) * this->iDirection <
          this->fBodyRadius )
        // Blocking
        {
                // Record the blocking vehicle
                this->pVehConflictTemp  = pVehicle;
                this->bVehBlockTemp     = true;

                fTimeGap = 0.0f;
                break;
        }
    }
    break;
}
```

```
					default:  // N/A
					{
						break;
					}
				}
		}

		return fTimeGap;
}

void Pedestrian::UpdateVelocity(float fSimTimeStep, MathSolver* pMathSolver,
				vector<Pedestrian*> vPedestrian, RandomSolver* pRandomSolver)
{

		if(this->bUseCrossing==true)   // Use Crossing
		{
				switch(this->pPedCrossing->iType)
				{
					case(CROSSING_TYPE_ZEBRA):
					{
						switch(this->iPosType)
						{
							// 1. Pavement Near
							case(PED_POS_PAVE_NEAR):
							{
								// Nearside PGA
								float fTimeGap = this->GetTrafficTimeGap(this->pLaneVehNear);
								bool bPGA = this->GapAcceptance(this->pLaneVehNear, fTimeGap,
									vPedestrian, pRandomSolver, pMathSolver);
								// Decide velocity
								if(bPGA)
								{
									// To PavementNear Edge
									float fLineY = (this->iDirection>0)?
									  this->pLaneCycleNear->fEdgeSouth:
									  this->pLaneCycleNear->fEdgeNorth;
									this->posDestTemp = this->GoToLine(fLineY);
									// Record Nearside TTC
									this->fNearsideTTC = fTimeGap - this->
									pLaneVehNear->fWidth/this->fSpeedDSR;
								}
								else
								{
									// To crossing nearside waiting zone
									if((this->posCentroid.y-this->posWaitingZoneNear.y)
										*this->iDirection <= 0)
									{
										this->posDestTemp =
										this->posWaitingZoneNear;
									}
									else
									{
										this->posDestTemp = this->posCentroid;
										this->posDestTemp.x =
										this->posWaitingZoneNear.x;
									}

								}
								// Calculate target position and speed rate
								this->posTarget = this->posDestTemp;
								// Check if reached the target position
								bool bReachTarget = this->IsReachedTargetPos();
								// Assign speed
								this->fSpeed = bReachTarget ? 0.0f : this->fSpeedDSR;
								break;
							}

							// 2. Nearside pavement edge
							case(PED_POS_PAVE_NEAR_EDGE):
							{
								// Nearside PGA
								float fTimeGap = this->GetTrafficTimeGap(this->pLaneVehNear);
								bool bPGA = this->GapAcceptance(this->pLaneVehNear, fTimeGap,
									vPedestrian, pRandomSolver, pMathSolver);
								// Decide velocity
								if(bPGA)
								{
									// To nearside cylce lane edge
									float fLineY = (this->iDirection>0)?
									  this->pLaneVehNear->fEdgeSouth:
									  this->pLaneVehNear->fEdgeNorth;
									this->posDestTemp= this->GoToLine(fLineY);
									// Record Nearside TTC
									this->fNearsideTTC = fTimeGap - this->
									pLaneVehNear->fWidth/this->fSpeedDSR;
								}
```

```
        else
        {
                // To Zebra crossing nearside waiting zone
                if((this->posCentroid.y-this->posWaitingZoneNear.y)
                        *this->iDirection <= 0)
                {
                        this->posDestTemp =
                         this->posWaitingZoneNear;
                }
                else
                {
                        this->posDestTemp = this->posCentroid;
                        this->posDestTemp.x =
                        this->posWaitingZoneNear.x;
                }
        }
        // Calculate target position and speed rate
        this->posTarget = this->posDestTemp;
        // Check if reached the target position
        bool bReachTarget = this->IsReachedTargetPos();
        // Assign speed
        this->fSpeed = bReachTarget ? 0.0f : this->fSpeedDSR;
        break;
}

// 3. Nearside cycle lane
case(PED_POS_CYCLE_NEAR):
{
        // To nearside cylce lane edge
        float fLineY = (this->iDirection>0)?
          this->pLaneVehNear->fEdgeSouth:
          this->pLaneVehNear->fEdgeNorth;
        this->posDestTemp = this->GoToLine(fLineY);
        // Check if in crossing
        Position posRectangle = this->pPedCrossing->PosCentroid;
        float fLength = this->pPedCrossing->fLength;
        float fHeight = this->pPedCrossing->fWidth;
        bool bInCrossing =
        pMathSolver->IsPosInRectangle(this->
                posCentroid,posRectangle,fLength,fHeight);
        // Update if use crossing
        this->bUseCrossing = bInCrossing ? true : false;
        // Calculate target position and speed rate
        this->posTarget = this->posDestTemp;
        // Check if reached the target position
        bool bReachTarget = this->IsReachedTargetPos();
        // Assign speed
        this->fSpeed = bReachTarget ? 0.0f : this->fSpeedDSR;
        break;
}

// 4. Nearside cycle lane edge
case(PED_POS_CYCLE_NEAR_EDGE):
{
        // Check if in crossing
        Position posRectangle = this->pPedCrossing->PosCentroid;
        float fLength = this->pPedCrossing->fLength;
        float fHeight = this->pPedCrossing->fWidth;
        bool bInCrossing =
        pMathSolver->IsPosInRectangle(this->
        posCentroid,posRectangle,fLength,fHeight);
        // Nearside PGA
        float fTimeGap = this->GetTrafficTimeGap(this->pLaneVehNear);
        bool bPGA = this->GapAcceptance(this->pLaneVehNear, fTimeGap,
        vPedestrian, pRandomSolver, pMathSolver);
        if(bPGA)
        {
                // Check if the median available
                if(this->bMedianAvailable)
                {
                        // To Median
                        float fLineY = this->pMedian->posStart.y;
                        this->posDestTemp= this->
                        GoToLine(fLineY);
                        // Record Nearside TTC
                        this->fNearsideTTC = fTimeGap - this->
                        pLaneVehNear->fWidth/this->fSpeedDSR;
                }
                else
                {
                        this->posDestTemp = this->posCentroid;
                }
        }
        else if(!bInCrossing)
        {
                // To crossing
```

```
                this->posDestTemp.y = this->posCentroid.y;
                this->posDestTemp.x = this->posWaitingZoneNear.x;
        }
        else
        {
                // Wait
                this->posDestTemp = this->posCentroid;
        }
        // Calculate target position and speed rate
        this->posTarget = this->posDestTemp;
        // Check if reached the target position
        bool bReachTarget = this->IsReachedTargetPos();
        // Assign speed
        this->fSpeed = bReachTarget ? 0.0f : this->fSpeedDSR;
        break;
}

// 5. Nearside vehicle lane
case(PED_POS_VEH_NEAR):
{
        // farside PGA
        float fTimeGap = this->GetTrafficTimeGap(this->pLaneVehFar);
        bool bPGA = this->GapAcceptance(this->pLaneVehFar, fTimeGap,
        vPedestrian, pRandomSolver,
        pMathSolver);
        if(bPGA)
        {
                // To far-side cycle lane
                float fLineY = (this->iDirection>0)?
                  this->pLaneCycleFar->fEdgeSouth:
                  this->pLaneCycleFar->fEdgeNorth;
                this->posDestTemp        = this->GoToLine(fLineY);
                // Record Farside TTC
                this->fFarsideTTC = fTimeGap - this->
                pLaneVehFar->fWidth/this->fSpeedDSR;
        }
        else
        {
                // To Median
                float fLineY = this->pMedian->posStart.y;
                this->posDestTemp= this->GoToLine(fLineY);
        }
        // Check if in crossing
        Position posRectangle = this->pPedCrossing->PosCentroid;
        float fLength = this->pPedCrossing->fLength;
        float fHeight = this->pPedCrossing->fWidth;
        bool bInCrossing =
        pMathSolver->IsPosInRectangle(this->
        posCentroid,posRectangle,fLength,fHeight);
        // Update if use crossing
        this->bUseCrossing = bInCrossing ? true : false;
        // Calculate target position and speed rate
        this->posTarget = this->posDestTemp;
        // Check if reached the target position
        bool bReachTarget = this->IsReachedTargetPos();
        if(bReachTarget)
        {
                this->fSpeed = 0.0f;
        }
        else
        {
                // Get the right speed in order not to collide
                        with conflicting vehicle
                this->fSpeed = this->GetCrossingSpeed(this->
                pLaneVehNear,pMathSolver);
        }
        break;
}

// 6. Median
case(PED_POS_MEDIAN):
{
        // Check if in crossing
        Position posRectangle = this->pPedCrossing->PosCentroid;
        float fLength = this->pPedCrossing->fLength;
        float fHeight = this->pPedCrossing->fWidth;
        bool bInCrossing =
        pMathSolver->IsPosInRectangle(this->
                posCentroid,posRectangle,fLength,fHeight);
        // Far PGA
        float fTimeGap = this->GetTrafficTimeGap(this->pLaneVehFar);
        bool bPGA = this->GapAcceptance(this->pLaneVehFar, fTimeGap,
        vPedestrian, pRandomSolver, pMathSolver);
        if(bPGA)
        {
                // To far-side cycle lane
```

```
                float fLineY = (this->iDirection>0)?
                  this->pLaneCycleFar->fEdgeSouth:
                  this->pLaneCycleFar->fEdgeNorth;
                this->posDestTemp = this->GoToLine(fLineY);
                // Record Farside TTC
                this->fFarsideTTC = fTimeGap - this->
                pLaneVehFar->fWidth/this->fSpeedDSR;
        }
        else if(!bInCrossing)
        {
                // To crossing
                this->posDestTemp.y = this->posCentroid.y;
                this->posDestTemp.x = this->posWaitingZoneNear.x;
        }
        else
        {
                // Wait
                this->posDestTemp       = this->posCentroid;
        }
        // Calculate target position and speed rate
        this->posTarget = this->posDestTemp;
        // Check if reached the target position
        bool bReachTarget = this->IsReachedTargetPos();
        // Assign speed
        this->fSpeed = bReachTarget ? 0.0f : this->fSpeedDSR;
        break;
}

// 7. Far-side vehicle lane
case(PED_POS_VEH_FAR):
{
        // To far-side cycle lane
        float fLineY = (this->iDirection>0)?
          this->pLaneCycleFar->fEdgeSouth:
          this->pLaneCycleFar->fEdgeNorth;
        this->posDestTemp       = this->GoToLine(fLineY);
        // Calculate target position and speed rate
        this->posTarget = this->posDestTemp;
        // Check if reached the target position
        bool bReachTarget = this->IsReachedTargetPos();
        if(bReachTarget)
        {
                this->fSpeed = 0.0f;
        }
        else
        {
                // Get the right speed in order not to collide
                        with conflicting vehicle
                this->fSpeed = this->GetCrossingSpeed(this->
                        pLaneVehFar,pMathSolver);
        }
        break;
}

// 8. Far-side cycle lane
case(PED_POS_CYCLE_FAR):
{
        // To far-side pavement
        this->posDestTemp.x = this->posCentroid.x;
        this->posDestTemp.y = this->posDest.y;
        // Calculate target position and speed rate
        this->posTarget = this->posDestTemp;
        // Check if reached the target position
        bool bReachTarget = this->IsReachedTargetPos();
        if(bReachTarget)
        {
                this->fSpeed = 0.0f;
        }
        else
        {
                this->fSpeed = this->fSpeedDSR;
        }
        break;
}

// 9. far-side pavement
case(PED_POS_PAVE_FAR):
{
        // To Destination
        this->posDestTemp = this->posDest;
        // Calculate target position and speed rate
        this->posTarget = this->posDestTemp;
        // Check if reached the target position
        bool bReachTarget = this->IsReachedTargetPos();
        if(bReachTarget)
        {
```

```
                                        this->fSpeed = 0.0f;
                                }
                                else
                                {
                                        this->fSpeed = this->fSpeedDSR;
                                }
                                break;
                        }
                }
                break;
        }

        case(CROSSING_TYPE_FIXED):
        case(CROSSING_TYPE_PUFFIN):
        {
                switch(this->iPosType)
                {
                        // Zone 1 and 2
                        case(PED_POS_PAVE_NEAR):
                        case(PED_POS_PAVE_NEAR_EDGE):
                        {
                                // Check if in Nearside Wait zone
                                bool bIsInWaitZoneNear =
                                this->IsInWaitingZoneNear(this->pPedCrossing,pMathSolver);
                                if(bIsInWaitZoneNear)
                                {
                                        // Check ped signal
                                        int iPedSignal = this->pPedCrossing->
                                                pSignalS->iStatus;
                                        if(iPedSignal==SIGNAL_STATUS_GREEN)
                                        {
                                                // To Far-side WaitZonePos
                                                this->posDestTemp =
                                                this->posWaitingZoneFar;
                                        }
                                        else
                                        {
                                                // To nearside WaitZone
                                                this->posDestTemp.x =
                                                this->posWaitingZoneNear.x;
                                                this->posDestTemp.y =
                                                ( (this->posWaitingZoneNear.y
                                                - this->posCentroid.y)
                                                  * this->iDirection >=0 )?
                                                  this->posWaitingZoneNear.y :
                                                 this->posCentroid.y;
                                        }
                                }
                                else
                                {
                                        // To nearside WaitZone
                                        this->posDestTemp.x = this->posWaitingZoneNear.x;
                                        this->posDestTemp.y =
                                        ( (this->posWaitingZoneNear.y - this->posCentroid.y)
                                          * this->iDirection >=0 )?
                                          this->posWaitingZoneNear.y : this->posCentroid.y;
                                }

                                // Calculate target position and speed rate
                                this->posTarget = this->posDestTemp;
                                // Check if reached the target position
                                bool bReachTarget = this->IsReachedTargetPos();
                                if(bReachTarget)
                                {
                                        this->fSpeed = 0.0f;
                                }
                                else
                                {
                                        this->fSpeed = this->fSpeedDSR;
                                }
                                break;
                        }

                        // Zone 3
                        case(PED_POS_CYCLE_NEAR):
                        {
                                // Check if in crossing
                                Position posRectangle = this->pPedCrossing->PosCentroid;
                                float fLength = this->pPedCrossing->fLength;
                                float fHeight = this->pPedCrossing->fWidth;
                                bool bInCrossing =
                                pMathSolver->IsPosInRectangle(this->
                                        posCentroid,posRectangle,fLength,fHeight);

                                if(bInCrossing)
                                {
```

```
				// To Far-side WaitZonePos
				this->posDestTemp = this->posWaitingZoneFar;
			}
			else
			{
				// To nearside cylce lane edge
				float fLineY			= (this->iDirection>0)?
				  this->pLaneCycleNear->fEdgeSouth:
				  this->pLaneCycleNear->fEdgeNorth;
				this->posDestTemp		= this->GoToLine(fLineY);
			}

			// Calculate target position and speed rate
			this->posTarget = this->posDestTemp;
			// Check if reached the target position
			bool bReachTarget = this->IsReachedTargetPos();
			// Assign speed
			this->fSpeed = bReachTarget ? 0.0f : this->fSpeedDSR;
			break;
		}

		// Zone 4
		case(PED_POS_CYCLE_NEAR_EDGE):
		{
			// Check if in crossing
			Position posRectangle = this->pPedCrossing->PosCentroid;
			float fLength = this->pPedCrossing->fLength;
			float fHeight = this->pPedCrossing->fWidth;
			bool bInCrossing =
			pMathSolver->IsPosInRectangle(this->
				posCentroid,posRectangle,fLength,fHeight);

			if(bInCrossing)
			{
				this->posDestTemp = this->posWaitingZoneFar;
			}
			else
			{
				// Nearside PGA
				float fTimeGap = this->GetTrafficTimeGap(this->
					pLaneVehNear);
				bool bPGA = this->GapAcceptance(this->
					pLaneVehNear, fTimeGap,
					vPedestrian, pRandomSolver,
					pMathSolver);
				if(bPGA)
				{
					if(this->bMedianAvailable)
					{
						// To Median
						float fLineY=  this->pMedian->
						posStart.y;   this->
						posDestTemp= this->
						GoToLine(fLineY); // Record
						Nearside TTC
						this->fNearsideTTC =
						fTimeGap - this-
						>pLaneVehNear->fWidth/this-
						>fSpeedDSR;
					}
				}
				else
				{
					// To crossing
					this->posDestTemp.y	= this->
					posCentroid.y; this->posDestTemp.x
					= this->posWaitingZoneNear.x;
				}
			}

			// Calculate target position and speed rate
			this->posTarget = this->posDestTemp;
			// Check if reached the target position
			bool bReachTarget = this->IsReachedTargetPos();
			// Assign speed
			this->fSpeed = bReachTarget ? 0.0f : this->fSpeedDSR;
			break;
		}

		// Zone 5
		case(PED_POS_VEH_NEAR):
		{
			// Check if in crossing
			Position posRectangle = this->pPedCrossing->PosCentroid;
			float fLength = this->pPedCrossing->fLength;
			float fHeight = this->pPedCrossing->fWidth;
```

```
		bool bInCrossing =
		pMathSolver->IsPosInRectangle(this->
			posCentroid,posRectangle,fLength,fHeight);

		if(bInCrossing)
		{
			this->posDestTemp = this->posWaitingZoneFar;
		}
		else
		{
			// To Median
			float fLineY			= this->pMedian->posStart.y;
			this->posDestTemp		= this->GoToLine(fLineY);
		}

		// Calculate target position and speed rate
		this->posTarget = this->posDestTemp;
		// Check if reached the target position
		bool bReachTarget = this->IsReachedTargetPos();
		if(bReachTarget)
		{
			this->fSpeed = 0.0f;
		}
		else
		{
			// Get the right speed in order not to collide
				with conflicting vehicle
			this->fSpeed = this->GetCrossingSpeed(this->
			pLaneVehNear,pMathSolver);
		}
		break;
	}

	// Zone 6
	case(PED_POS_MEDIAN):
	{
		// Check if in crossing
		Position posRectangle = this->pPedCrossing->PosCentroid;
		float fLength = this->pPedCrossing->fLength;
		float fHeight = this->pPedCrossing->fWidth;
		bool bInCrossing =
		pMathSolver->IsPosInRectangle(this->
		posCentroid,posRectangle,fLength,fHeight);

		if(bInCrossing)
		{
			this->posDestTemp = this->posWaitingZoneFar;
		}
		else
		{
			// Farside PGA
			float fTimeGap = this->GetTrafficTimeGap(this->
			pLaneVehFar);bool bPGA = this->
			GapAcceptance(this->pLaneVehFar, fTimeGap,
			vPedestrian, pRandomSolver, pMathSolver);
			if(bPGA)
			{
				// To far-side cycle lane
				float fLineY = (this->iDirection>0)?
				  this->pLaneCycleFar->fEdgeSouth:
				  this->pLaneCycleFar->fEdgeNorth;
				this->posDestTemp= this->
				GoToLine(fLineY);
				// Record Farside TTC
				this->fFarsideTTC = fTimeGap - this->
				pLaneVehFar->fWidth/this->fSpeedDSR;
			}
			else
			{
				// To crossing
				this->posDestTemp.y	= this->
				posCentroid.y;
				this->posDestTemp.x = this->
				posWaitingZoneNear.x;
			}
		}

		// Calculate target position and speed rate
		this->posTarget = this->posDestTemp;
		// Check if reached the target position
		bool bReachTarget = this->IsReachedTargetPos();
		// Assign speed
		this->fSpeed = bReachTarget ? 0.0f : this->fSpeedDSR;
		break;
	}
```

```
						// Zone 7
						case(PED_POS_VEH_FAR):
						{
							// Check if in crossing
							Position posRectangle = this->pPedCrossing->PosCentroid;
							float fLength = this->pPedCrossing->fLength;
							float fHeight = this->pPedCrossing->fWidth;
							bool bInCrossing =
							pMathSolver->IsPosInRectangle(this->
							posCentroid,posRectangle,fLength,fHeight);

							if(bInCrossing)
							{
								this->posDestTemp = this->posWaitingZoneFar;
							}
							else
							{
								// To far-side cycle lane
								float fLineY		= (this->iDirection>0)?
								  this->pLaneCycleFar->fEdgeSouth:
								  this->pLaneCycleFar->fEdgeNorth;
								this->posDestTemp	= this->GoToLine(fLineY);
							}

							// Calculate target position and speed rate
							this->posTarget = this->posDestTemp;
							// Check if reached the target position
							bool bReachTarget = this->IsReachedTargetPos();
							if(bReachTarget)
							{
								this->fSpeed = 0.0f;
							}
							else
							{
								// Get the right speed in order not to collide
									with conflicting vehicle
								this->fSpeed = this->GetCrossingSpeed(this->
								pLaneVehFar,pMathSolver);
							}
							break;
						}

						// Zone 8
						case(PED_POS_CYCLE_FAR):
						{
							// To far-side pavement
							this->posDestTemp.x = this->posCentroid.x;
							this->posDestTemp.y = this->posDest.y;
							// Calculate target position and speed rate
							this->posTarget = this->posDestTemp;
							// Check if reached the target position
							bool bReachTarget = this->IsReachedTargetPos();
							if(bReachTarget)
							{
								this->fSpeed = 0.0f;
							}
							else
							{
								this->fSpeed = this->fSpeedDSR;
							}
							break;
						}

						// Zone 9
						case(PED_POS_PAVE_FAR):
						{
							// To Destination
							this->posDestTemp = this->posDest;
							// Calculate target position and speed rate
							this->posTarget = this->posDestTemp;
							// Check if reached the target position
							bool bReachTarget = this->IsReachedTargetPos();
							if(bReachTarget)
							{
								this->fSpeed = 0.0f;
							}
							else
							{
								this->fSpeed = this->fSpeedDSR;
							}
							break;
						}
					}
					break;
				}
			}
```

```
}

else   // Not use Crossing
{
        // Update velocity according to different Position Type
        switch(this->iPosType)
        {
                // Zone 1
                case(PED_POS_PAVE_NEAR):
                {
                        float fTimeGap = (float)INT_MAX;
                        // Check nearside traffic gap
                        fTimeGap = this->GetTrafficTimeGap(this->pLaneVehNear);
                        // Check if the pedestrian accepts this gap
                        bool bGapAcceptance = this->GapAcceptance(this->pLaneVehNear, fTimeGap,
                                                vPedestrian, pRandomSolver,   pMathSolver);
                        if(bGapAcceptance)  // Gap accepted
                        {
                                this->posDestTemp.x = this->posCentroid.x;
                                this->posDestTemp.y = (this->iDirection>0)?
                                  this->pLaneCycleNear->fEdgeSouth:
                                  this->pLaneCycleNear->fEdgeNorth;
                                // Record Nearside TTC
                                this->fNearsideTTC = fTimeGap - this->pLaneVehNear->
                                fWidth/this->fSpeedDSR;
                        }
                        else  // Gap refused
                        {
                                this->posDestTemp.x = this->posDest.x;
                                this->posDestTemp.y = (this->iDirection>0)?
                                  this->pLaneCycleNear->fEdgeSouth:
                                  this->pLaneCycleNear->fEdgeNorth;
                        }
                        // Calculate target position and speed rate
                        this->posTarget = this->posDestTemp;
                        // Check if reached the target position
                        if(this->posCentroid.x==this->posTarget.x
                        && this->posCentroid.y==this->posTarget.y)  // Reached
                        {
                                this->fSpeed = 0.0f;
                        }
                        else
                        {
                                this->fSpeed = this->fSpeedDSR;
                        }
                        break;
                }


                // Zone 2
                case(PED_POS_PAVE_NEAR_EDGE):
                {
                        float fTimeGap = (float)INT_MAX;
                        // Get the conflicting vehicle and its type in the nearside veh lane
                        // Check nearside traffic gap
                        fTimeGap = this->GetTrafficTimeGap(this->pLaneVehNear);

                        // Decide temporary destination according to the conflicting vehicle
                        if(fTimeGap > 0)  // Use PGA model
                        {
                                // Check if the pedestrian accepts this gap
                                bool bGapAcceptance = this->GapAcceptance(this->pLaneVehNear, fTimeGap,
                                  vPedestrian, pRandomSolver,   pMathSolver);
                                if(bGapAcceptance)  // Gap accepted
                                {
                                        this->posDestTemp.x = this->posCentroid.x;
                                        this->posDestTemp.y = (this->iDirection>0)?
                                           this->pLaneVehNear->fEdgeSouth:
                                           this->pLaneVehNear->fEdgeNorth;
                                        // Record Nearside TTC
                                        this->fNearsideTTC = fTimeGap - this->
                                        pLaneVehNear->fWidth/this->fSpeedDSR;
                                }
                                else  // Gap refused
                                {
                                        this->posDestTemp.x = this->posDest.x;
                                        this->posDestTemp.y = (this->iDirection>0)?
                                          this->pLaneCycleNear->fEdgeSouth:
                                          this->pLaneCycleNear->fEdgeNorth;
                                }
                        }
                        else  // Use blocking model, to be modified
                        {
                                if(this->pVehConflictTemp->fSpeed > 0)
                                {
                                        this->posDestTemp.x = this->posDest.x;
```

```
                    this->posDestTemp.y = (this->iDirection>0)?
                      this->pLaneCycleNear->fEdgeSouth:
                      this->pLaneCycleNear->fEdgeNorth;
                }
                else
                {
                    // Get x of VehRear
                    Position posVehRear = this->pVehConflictTemp->GetPosition2
                      (this->pVehConflictTemp->fDeltaX
                      - this->pVehConflictTemp->fLength
                      - this->pVehConflictTemp->fMargin / 2);
                    // Get new tempdest
                    this->posDestTemp.x = posVehRear.x;
                    this->posDestTemp.y = (this->iDirection>0)?
                      this->pLaneCycleNear->fEdgeSouth:
                      this->pLaneCycleNear->fEdgeNorth;
                }
            }
            // Calculate target position and speed rate
            this->posTarget = this->posDestTemp;
            // Check if reached the target position
            if(this->posCentroid.x==this->posTarget.x
            && this->posCentroid.y==this->posTarget.y)  // Reached
            {
                this->fSpeed = 0.0f;
            }
            else
            {
                this->fSpeed = this->fSpeedDSR;
            }
            break;
        }


        // Zone 3
        case(PED_POS_CYCLE_NEAR):
        {
            // Update DestTemp
            this->posDestTemp.x = this->posCentroid.x;
            this->posDestTemp.y = (this->iDirection>0)?
              this->pLaneVehNear->fEdgeSouth:
              this->pLaneVehNear->fEdgeNorth;
            // Calculate target position and speed rate
            this->posTarget = this->posDestTemp;
            // Check if reach the target position in next reaction time
            if(this->posCentroid.x==this->posTarget.x
            && this->posCentroid.y==this->posTarget.y)  // Reached
            {
                this->fSpeed = 0.0f;
            }
            else
            {
                this->fSpeed = this->fSpeedDSR;
            }
            break;
        }


        // Zone 4
        case(PED_POS_CYCLE_NEAR_EDGE):
        {
            float fTimeGap = (float)INT_MAX;

            // Get the conflicting vehicle and its type in the nearside veh lane
            // Check nearside traffic gap
            fTimeGap = this->GetTrafficTimeGap(this->pLaneVehNear);
            // Decide temporary destination according to the conflicting vehicle
            if(fTimeGap > 0)  // Use PGA model
            {
                // Check if the pedestrian accepts this gap
                bool bGapAcceptance = this->GapAcceptance(this->pLaneVehNear, fTimeGap,

              vPedestrian, pRandomSolver,

              pMathSolver);
                if(bGapAcceptance)  // Gap accepted
                {
                    // Check if the median available
                    if(this->bMedianAvailable)
                    {
                        this->posDestTemp.x = this->posCentroid.x;
                        this->posDestTemp.y = this->pMedian->posStart.y;
                        // Record Nearside TTC
                        this->fNearsideTTC = fTimeGap - this->
                        pLaneVehNear->fWidth/this->fSpeedDSR;
                    }
```

```
                        else
                        {
                                this->posDestTemp = this->posCentroid;
                        }
                }
                else  // Gap refused
                {
                        this->posDestTemp.x = this->posDest.x;
                        this->posDestTemp.y = (this->iDirection>0)?
                          this->pLaneVehNear->fEdgeSouth:
                          this->pLaneVehNear->fEdgeNorth;
                }
        }
        else  // Use blocking model, to be modified
        {
                if(this->pVehConflictTemp->fSpeed > 0)
                {
                        this->posDestTemp.x = this->posDest.x;
                        this->posDestTemp.y = (this->iDirection>0)?
                          this->pLaneVehNear->fEdgeSouth:
                          this->pLaneVehNear->fEdgeNorth;
                }
                else
                {
                        // Get x of VehRear
                        Position posVehRear = this->pVehConflictTemp->GetPosition2
                          (this->pVehConflictTemp->fDeltaX
                          - this->pVehConflictTemp->fLength
                          - this->pVehConflictTemp->fMargin / 2);
                        // Get new tempdest
                        this->posDestTemp.x = posVehRear.x;
                        this->posDestTemp.y = (this->iDirection>0)?
                          this->pLaneVehNear->fEdgeSouth:
                          this->pLaneVehNear->fEdgeNorth;
                }
        }
        // Calculate target position and speed rate
        this->posTarget = this->posDestTemp;
        // Check if reach the target position in next reaction time
        if(this->posCentroid.x==this->posTarget.x
        && this->posCentroid.y==this->posTarget.y)  // Reached
        {
                this->fSpeed = 0.0f;
        }
        else
        {
                this->fSpeed = this->fSpeedDSR;
        }
        break;
}


// Zone 5
case(PED_POS_VEH_NEAR):
{
        float fTimeGap = (float)INT_MAX;
        // Get the conflicting vehicle and its type in the farside veh lane
        // Check farside traffic gap
        fTimeGap = this->GetTrafficTimeGap(this->pLaneVehFar);
        // Decide temporary destination according to the conflicting vehicle
        if(fTimeGap > 0)  // Use PGA model
        {
                // Check if the pedestrian accepts this gap
                bool bGapAcceptance = this->GapAcceptance(this->pLaneVehFar, fTimeGap,
                  vPedestrian, pRandomSolver,  pMathSolver);
                if(bGapAcceptance)  // Gap accepted
                {
                        this->posDestTemp.x = this->posCentroid.x;
                        this->posDestTemp.y = (this->iDirection>0)?
                          this->pLaneCycleFar->fEdgeSouth:
                          this->pLaneCycleFar->fEdgeNorth;
                        // Record Farside TTC
                        this->fFarsideTTC = fTimeGap - this->pLaneVehFar->
                        fWidth/this->fSpeedDSR;
                }
                else  // Gap refused
                {
                        this->posDestTemp.x = this->posCentroid.x;
                        this->posDestTemp.y = this->pMedian->posStart.y;
                }
        }
        else  // Use blocking model, to be modified
        {
                this->posDestTemp.x = this->posCentroid.x;
                this->posDestTemp.y = this->pMedian->posStart.y;
        }
```

```
		// Calculate target position and speed rate
		this->posTarget = this->posDestTemp;
		// Check if reach the target position
		if(this->posCentroid.x==this->posTarget.x
		&& this->posCentroid.y==this->posTarget.y)  // Reached
		{
			this->fSpeed = 0.0f;
		}
		else
		{
			// Get the right speed in order not to collide with conflicting vehicle
			this->fSpeed = this->GetCrossingSpeed(this->pLaneVehNear,pMathSolver);
		}
		break;
	}

	// Zone 6
	case(PED_POS_MEDIAN):
	{
		float fTimeGap = (float)INT_MAX;
		// Get the conflicting vehicle and its type in the farside veh lane
		// Check farside traffic gap
		fTimeGap = this->GetTrafficTimeGap(this->pLaneVehFar);

		// Decide temporary destination according to the conflicting vehicle
		if(fTimeGap > 0)  // Use PGA model
		{
			// Check if the pedestrian accepts this gap
			bool bGapAcceptance = this->GapAcceptance(this->pLaneVehFar, fTimeGap,
			  vPedestrian, pRandomSolver,   pMathSolver);
			if(bGapAcceptance)  // Gap accepted
			{
				this->posDestTemp.x = this->posCentroid.x;
				this->posDestTemp.y = (this->iDirection>0)?
				  this->pLaneCycleFar->fEdgeSouth:
				  this->pLaneCycleFar->fEdgeNorth;
				// Record Farside TTC
				this->fFarsideTTC = fTimeGap - this->pLaneVehFar->
					fWidth/this->fSpeedDSR;
			}
			else  // Gap refused
			{
				this->posDestTemp = this->posCentroid;
			}
		}
		else  // Use blocking model, to be modified
		{
			if(this->pVehConflictTemp->fSpeed > 0)
			{
				this->posDestTemp = this->posCentroid;
			}
			else
			{
				// Get x of VehRear
				Position posVehRear = this->pVehConflictTemp->GetPosition2
				  (this->pVehConflictTemp->fDeltaX
				  - this->pVehConflictTemp->fLength
				  - this->pVehConflictTemp->fMargin / 2);
				// Get new tempdest
				this->posDestTemp.x = posVehRear.x;
				this->posDestTemp.y = this->pMedian->posStart.y;
			}
		}
		// Calculate target position and speed rate
		this->posTarget = this->posDestTemp;
		// Check if reach the target position
		if(this->posCentroid.x==this->posTarget.x
		&& this->posCentroid.y==this->posTarget.y)  // Reached
		{
			this->fSpeed = 0.0f;
		}
		else
		{
			this->fSpeed = this->fSpeedDSR;
		}
		break;
	}


	// Zone 7
	case(PED_POS_VEH_FAR):
	{
		// Recalculate posDestTemp
		this->posDestTemp.x = this->posCentroid.x;
		this->posDestTemp.y = (this->iDirection>0)?
		  this->pLaneCycleFar->fEdgeSouth:
```

```
                          this->pLaneCycleFar->fEdgeNorth;
                        // Calculate target position and speed rate
                        this->posTarget = this->posDestTemp;
                        // Check if reach the target position
                        if(this->posCentroid.x==this->posTarget.x
                        && this->posCentroid.y==this->posTarget.y)  // Reached
                        {
                                this->fSpeed = 0.0f;
                        }
                        else
                        {
                                // Get the right speed in order not to collide with conflicting vehicle
                                this->fSpeed = this->GetCrossingSpeed(this->pLaneVehFar,pMathSolver);
                        }
                        break;
                }

                // Zone 8
                case(PED_POS_CYCLE_FAR):
                {
                        // Recalculate posDestTemp
                        this->posDestTemp.x = this->posCentroid.x;
                        this->posDestTemp.y = this->posDest.y;
                        // Calculate target position and speed rate
                        this->posTarget = this->posDestTemp;
                        // Check if reach the target position
                        if(this->posCentroid.x==this->posTarget.x
                        && this->posCentroid.y==this->posTarget.y)  // Reached
                        {
                                this->fSpeed = 0.0f;
                        }
                        else
                        {
                                this->fSpeed = this->fSpeedDSR;
                        }
                        break;
                }

                // Zone 9
                case(PED_POS_PAVE_FAR):
                {
                        // Recalculate posDestTemp
                        this->posDestTemp = this->posDest;

                        // Calculate target position and speed rate
                        this->posTarget = this->posDestTemp;

                        // Check if reach the target position
                        if(this->posCentroid.x==this->posTarget.x
                        && this->posCentroid.y==this->posTarget.y)  // Reached
                        {
                                this->fSpeed = 0.0f;
                        }
                        else
                        {
                                this->fSpeed = this->fSpeedDSR;
                        }
                        break;
                }

                default:
                {
                        break;
                }
        }
    }
}
```

Appendix III: Publications during candidature

[1] Wang, T., Wu, J. and McDonald, M. (2012) A Micro-Simulation Model of Pedestrian-Vehicle Interaction Behaviour at Unsignalised Mid-Block Locations. *Proceedings of 15th International IEEE Conference on Intelligent Transportation Systems (ITSC 2012),* Anchorage, Alaska, United States, 16-19, 2012.

[2] Wang, T., Wu, J., Zheng, P. and McDonald, M. (2010) Study of Pedestrians' Gap Acceptance Behavior when They Jaywalk outside Crossing Facilities. *Proceedings of 13th International IEEE Conference on Intelligent Transportation Systems (ITSC 2010),* Madeira Island, Portugal, September 19-22, 2010.

[3] Wang, T., Wu, J., Zheng, P. and McDonald, M. (2010) A Framework for Analysing and Modelling Pedestrian-Vehicle Interaction Behaviour in a Micro-simulation Environment. *Proceedings (CD) of 3rd Transport Research Arena Conference (TRA 2010),* Brussels, Belgium, 7-10, June, 2010.

[4] Yang, Y., Zhao, X. and Wang, T. (2009) Design of Arbitrarily Controlled Multi-Beam Antennas via Optical Transformation. *Journal of Infrared, Millimeter and Terahertz Waves,* ISSN: 1866-6892 (Print) 1866-6906 (Online), 30 (4), 337-348.

[5] Wu, J., Sui, Y. and Wang, T. (2009) Intelligent Transport Systems in China. *Proceedings of Institution of Civil Engineers, Municipal Engineer,* ISSN: 0965-0903, 162 (1), 25-32.

[6] Yang, Y., Zhao, X. and Wang, T. (2008) Design of multi-beam antennas via optical transformation for anti-collision radar system. *Proceedings of 2008 International Workshop on Metamaterials,* Nanjing, China, 9-12 November 2008, Proceedings ISBN: 978-1-4244-2608-9, 240-243.

[7] Wu, J., Yin, X., Wang, J. and Wang, T. (2008) A Real-time Traffic Information System (RTIS) based on GPS floating vehicle technology in Hangzhou. *Proceedings (CD) of 15th World Congress on ITS (WCITS 2008),* New York, USA, 16-20, November, 2008.

REFERNCES

[1] Adams, W. F. (1936) Road traffic considered as a random series. *Journal of Institution of Civil Engineers,* 4, 121-130.

[2] Airault, V., Espié, S., Lattaud, C. and Auberlet, J. M. (2004) Interaction between pedestrians and their environment when road-crossing: A behavioural approach. In E. Fendel, & M. Rumor (Eds.), *Proceedings of the 24th urban data management symposium,* Italy, 2004.

[3] Algers, S., Bernauer, E., Boero, M., Breheret, L. Di Taranto, C., Dougherty, M., Fox, K. and Gabard, J. (1997) *Review of microsimulation models*. SMARTEST Project Deliverable D3, Institute for Transport Studies, University of Leeds, Leeds, UK. Available from: http://www.its.leeds.ac.uk/projects/smartest/deliv3.html#a2 [Accessed 25 June 2008].

[4] Anderson, T. E, Kidd, E. and Laughery, K. R. (1968) *TEXA Model: A computer simulation of driver behaviour at intersection.* Federal Highway Association.

[5] Antonini, G., Bierlaire, M. and Weber, M. (2006) Discrete choice models of pedestrian walking behavior. *Transportation Research Part B,* 40 (8), 667-687.

[6] Aycin, M. F. (2001) *Development of a Car-following Model to Simulate Driver and Autonomous Intelligent Cruise Controlled Vehicular Traffic Flow.* PhD Dissertation, University of Illinois.

[7] Bierlaire, M. and Robin, T. (2009) Pedestrians Choices. In Timmermans, H. (ed.) *Pedestrian Behavior: Models, Data Collection and Applications (ISBN: 978-1-84855-750-5),* pp. 1-26, Emerald Group Publishing Limited, UK.

[8] Blue, V. J. and Adler, J. L. Cellular Automata Model of Emergent Collective Bi-Directional Pedestrian Dynamics. *Artificial Life.* VII Bedau, M. A., McCaskill, J. S. Packard, N. H. and Rasmussen, S. MIT Press, 2000.

[9] Bohannon, R. (1997) Comfortable and maximum walking speed of adults aged 20-79 years: reference values and determinants. *Age and ageing,* 1997 (26), 15-19.

[10] Bouwman, M. (2000) *Tracking transport systems, an environmental perspective on passenger transport modes*. Geo Press, Groningen.

[11] Brackstone, M. and McDonald, M. (1999) Car-following: a historical review. *Transportation Research,* 2, 181-196.

[12] Buckley, D. J. (1962) Road traffic headway distribution. *Proceedings of Australian Road Research Board Conference,* 1, 153-183.

[13] Burstedde, C., Klauck, K., Schadschneider, A. and Zittartz, J. (2001) Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physics A,* 295, 507-525.

[14] Bönisch, C. and Kretz, T. (2009) *Simulation of Pedestrians Crossing a Street.* Availabe from: http://arxiv.org/pdf/0911.2902v1.pdf [Accessed on 10 September 2012].

[15] Cassini, M. (2006) Rip them out. *Telegraph (Newspaper in United Kingdom),* 14 October 2006. Available from: http://www.telegraph.co.uk/motoring/safety/2743688/Rip-them-out.html [Accessed on 12 August 2009].

[16] Chandler, R. E. Herman, R. and Montroll, E. W. (1958) Traffic dynamics: studies in car following. *Operations Research,* 6, 165-184.

[17] Chen, B. and Qi, T. (2002) Study of signal timing at mid-block pedestrian crossing facilities (in Chinese). *Urban Transport of China*, 2002, 1, 11-16.

[18] Chen, X., Shao, C. and Hao, Y. (2008) Influence of Pedestrian Traffic on Capacity of Right-Turning Movements at Signalized Intersections. *Transportation Research Record: Journal of the Transportation Research Board.* No. 2073, TRB, National Research Council, Washington D.C., 2008, 114-124.

[19] Chin, H. C. (1983) *A Computer Simulation Model of Traffic Operation at Roundabouts.* Dissertation (PhD Civil Engineering), University of Southampton, United Kingdom.

[20] Chu, X. and Baltes, M. R. (2003) Measuring Pedestrian Level of Service for Midblock Street Crossing: Selection of Potential Determinants. *Transportation Research Record: Journal of the Transportation Research Board.* No. 1828, TRB, National Research Council, Washington D.C., 2003, 89-97.

[21] Daamen, W. (2008) *SimPed.* Available at: http://www.citg.tudelft.nl/live/pagina.jsp?id=20f496a1-bf11-48e1-a34b-165b6a1caf94&lang=en [Accessed on 02 June 2008].

[22] Daff, R., Cramphorn, B., Wilson, C. J. and Neylan, J. (1991) Pedestrian behaviour near signalised crossings (Sydney). *Proceedings of 16th ARRB Conference, Part 4.*

[23] Dell'Orco, M. (2007) Competitive egress behaviour: a fuzzy logic-inspired microscopic model. *Int. J. Critical Infrastructure*, 3(3/4), 408-429.

[24] Department for Transport (1995a) *Local Transport Note 1/95 – The Assessment of Pedestrian Crossings.* London: TSO.

[25] Department for Transport (1995b) *Local Transport Note 2/95 – The Design of Pedestrian Crossings.* London: TSO.

[26] Department for Transport (2006) *The Puffin Good Practice Guide.* Queen's Printer and Controller of HMSO, 2006, UK.

[27] Dore, C. J., Murrells, T. P., Passant, N. R., Hobson, M. M., Baggott, S. L., Thistlethwaite, G., Goodwin, J. W. L., King, K. R., Adams, M., Walker, C., Downes, M. K., Coleman, P. J., Stewart, R. A., Wagner, A., Sturman, J., Conolly, C., Lawrence, H. and Cumine, P. R. (2005) *UK Emissions of Air Pollutants 1970 to 2003, 17th annual report from the UK National Atmospheric Emissions Inventory (NAEI).* Available from: http://www.airquality.co.uk/archive/reports/cat07/0602221510_1_main03_pt1_v5.doc#_Toc126986766. [Accessed 9 October 2011].

[28] Drew, D. R. (1968) *Traffic Flow Theory and Control*. New York: McGraw-Hill.

[29] Du, Y. (2008) *Simulation Study of Different Operations of Mixed Traffic with Motor Vehicles and Bicycles at AT-Grade Intersections in Beijing, China.* Dissertation (PhD Civil Engineering), University of Southampton, United Kingdom.

[30] Faghri, A. and Egyhaziova, E. (1999) Development of a Computer Simulation Model of Mixed Motor Vehicle and Bicycle Traffic on an Urban Road Network. *Transportation Research Record*, 1674, 86-93.

[31] FHWA (Federal HighWay Administration of the United States) (1998) *CORSIM User Manual (1.04 ed.).* Federal Highway Administration, Department of Transportation, the United States, McLean, Virginia.

[32] FHWA (Federal HighWay Administration of the United States) (2003) *Manual on Uniform Traffic Control Devices (MUTCD).* FHWA, Department of Transportation, the United States. Available from: http://mutcd.fhwa.dot.gov [Accessed 20 January 2009].

[33] Frank, L. D., Anderson, M. A. and Schmid, T. L. (2004) Obesity relationships with community design, physical activity, and time spent in cars. *American Journal of Preventive Medicine*, 27, 87-96.

[34] Gettman, D. and Head, L. (2003) *Surrogate Safety Measures From Traffic Simulation Models: Final Report*, US: Federal Highway Administration.

[35] Gipps, P. G. (1981) A Behavioural Car-Following Model for Computer Simulation. *Transportation Research B*, 15B, 105-111.

[36] Granovskii, M., Dincer, I. and Rosen, M. A. (2006) Economic and environmental comparison of conventional, hybrid, electric and hydrogen fuel cell vehicles. *Journal of Power Sources*, 159, 1186-1193.

[37] Green, M. (2000) "How Long Does It Take to Stop?" Methodological Analysis of Driver Perception-Brake Times. *Transportation Human Factors*, 2 (3), 195-216.

[38] Griffiths, J. D. (1991) Vehicle headways in urban areas. *Traffic Engineering and control*, 32, 458, 462

[39] Halcrow Group (2008) *Paxport Pedestrian Planning Software.* Available at: http://www.halcrow.com/software/solutions/paxport.asp. [Accessed on 5 June 2008]

[40] Hamilton-Baillie, B. (2001) *Home Zones - Reconciling People, Places and Transport.* August 2000, Publisher: Study Tour of Denmark, Germany, Holland, and Sweden.

[41] Hamiltion-Baillie, B. (2008) Shared Space: Reconciling People, Places and Traffic. *Built environment*, 34 (2), 161-181.

[42] Harney, D. (2002) Pedestrian modelling: Current methods and future directions. *Road & Transport Research: Journal of Australian and New Zealand Research and Practice*, 11 (4), 38-48.

[43] Hayashi, T. Tsumura, K., Suematsu, C., Okada, K., Fujii, S. and Endo, G. (1999) Walking to work and the risk for hypertension in men: The Osaka Health Survey. *Annals of Internal Medicine*, 131, 21-26.

[44] Helbing, D. and Molnár, P. (1995) Social force model for pedestrian dynamics. *Physical Review E*, 51 (5), 4282-4286.

[45] Herman, R. and Potts, R. B. (1959) Single Lane Traffic Theory and Experiment. In *Proceedings of the Symposium on Theory of Traffic Flow*, Research Labs, General Motors (pp. 147-157). New York: Elsevier.

[46] Highways Agency (2010) *Air quality, Design Manual for Roads and Bridges.* Vol. 11, Sec. 3, Pt. 1. Available from: http://www.dft.gov.uk/ha/standards/dmrb/vol11/section3/ha20707.pdf [Accessed 12 October 2010].

[47] Hine, J. (1996) Pedestrian travel experiences: assessing the impact of traffic on behaviour and perceptions of safety using an in-depth interview technique. *Journal of Transport Geography*, 4, 179-199.

[48] Hirsh, M. (1986) Disaggregate analysis of speeding behaviour of drivers. *Transportation Research Record*, 1059, 13-16.

[49] Hoefs, D. H. (1972) *Entwicklung einer Messmethode uber den Bewegungsablauf des Kolonnenverrkehrs.* Universitat (TH) Karlsruhe, Germany.

[50] Hoogendoorn, S. P. (2004) Pedestrian flow modeling by adaptive control. *In: Proceedings of the TRB 2004 annual meeting*, Washington DC.

[51] Ishaque, M. M. (2006) *Policies for Pedestrian Access: a Multi-modal Trade-off Analysis using Micro-simulation Techniques.* Dissertation (PhD Civil Engineering). University of London, United Kingdom.

[52] Ishaque, M. M. and Noland, R.B. (2007) Behavioural Issues in Pedestrian Speed Choice and Street Crossing Behaviour: A review. *Transport Reviews* 28(1): 61-85.

[53] Ishaque, M. M. and Norland, R. B. (2009) Pedestrian and Vehicle Flow Calibration in Multimodal Traffic Microsimulation. *Journal of Transportation Engineering – ASCE,* 135 (6), 338-348.

[54] Kahn Ribeiro, S., Kobayashi, S., Beuthe, M., Gasca, J., Greene, D., Lee, D. S., Muromachi, Y., Newton, P. J., Plotkin, S., Sperling, D., Wit, R. and Zhou, P. J. (2007) Transport and its infrastructure. *In Climate Change 2007: Mitigation. Contribution of Working Group III to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change* [Metz, B., Davidson, O. R., Bosch, P. R., Dave, R. and Meyer, L. A. (eds)], ambridge University Press, Cambridge, United Kingdom and New York, NY, USA.

[55] Knuth, D. (1997) Chapter 3 – Random Numbers. *The Art of Computer Programming. Vol. 2: Seminumerical algorithms* (3 ed.). Publisher: Addison-Wesley, United States.

[56] Kukla, R., Kerridge, J., Willis, A. and Hine, J. (2001) PEDFLOW: Development of an Autonomous Agent Model of Pedestrian Flow. *Journal of the Transportation Research Board*, 1774, 11-17.

[57] Legion (2012) *Aimsun 6.0.1 Released: Legion and Aimsun Get Closer.* Available from: http://www.legion.com/news/aimsun-601-released-legion-aimsun-get-closer [Accessed on 4 July 2012].

[58] Leutzbach, W. and Wiedemann, R. (1986) Development and applications of traffic simulation models at the Karlsruhe Institute für Verkehrwesen. *Traffic Engineering and Control,* 27 (May), 270-278.

[59] Li, D., Han, B. and Zhang, Q. (2007) Modeling and simulation of Microscopic Pedestrian Using Game Theory. *Journal of System Simulation,* 19 (11), 2590-2593.

[60] Li, J., Yang, L. and Zhao, D. (2005) Simulation of bi-directional pedestrian movement in corridor. *Physica A,* 354, 619-628.

[61] Li, Z., Liu, H. and Li, J. (2010) A Calibration and Validation Procedure for Microscopic Simulation Model. *Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems,* Madeira Island, Portugal, 19-22 September, 2010.

[62] Liang, K. and Zou, Z. (2006) Opinions of planning of pedestrian traffic system in urban traffic planning (in Chinese). Urban Roads, Bridge and Flood Control (Bimonthly), 2006 (2), 12-15.

[63] Lin, L., Mao, B., Ding, Y., Chen, Z. and Li, H. (2006) A Preliminary Analysis on Rational Development of Urban Taxi Traffic (in Chinese). *Urban Transport of China,* 4 (5), 69-72.

[64] Liu, Y., Shi, J. and Xiong, H. (2002) *The simulation technology in traffic systems (in Chinese).* Beijing: China Communications Press.

[65] Lu, H., Zhang, Y. and Liu, Q. (2009) A Methodology for Urban Pedestrian System Planning (in Chinese). *Urban Transport of China,* 7 (6), 53-58.

[66] Lyons, G., Hunt, J. and McLeod, F. (2001) Theory and methodology – A neural network model for enhanced operation of midblock signalled pedestrian crossings. *European Journal of Operational Research,* 129, 346-354.

[67] MathWorks (2000) Fuzzy Logic Toolbox Users' Guide. The MathWorks, Inc.

[68] McLean, J. R. (1982) *Two lane road traffic flow and capacity,* Australian Road Research Board.

[69] McLeod, F. N., Hounsell, N. B. and Rajbhandari, B. (2004) Improving traffic signal control for pedestrians. *In, 12th IEE International Conference on Road Transport Information and Control (RTIC 2004),* IEEE, 268-277.

[70] Mei, M. and Bullen, A. G. R. (1993) Lognormal distribution for high traffic flows. *Transportation Research Record,* 1398, 125-128.

[71] Michaels, R. M. (1963) Perceptual factors in car following. *Proceedings of the 2nd international symposium on the theory of road traffic flow,* 44-59. Paris: OECD.

[72] Min, G., Du, Y., Wu, J. and Yan, S. (2008) Simulation Study of Mixed Traffic in China – a Practice in Beijing. *Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems,* Beijing, China, 12-15 October, 2008.

[73] Mintsis, G. (1982) *Driver Behaviour on Curves.* Dissertation (PhD Civil Engineering), University of Southampton, United Kingdom.

[74] MOHURD (Ministry Of Housing and Urban-Rural Development of the People's Republic of China) (1995) *Code for transport planning on urban road,* GB 50220-95 (in Chinese).

[75] Montufar, J., Arango, J., Porter, M. and Nakagawa, S. (2007) Pedestrians' Normal Walking Speed and Speed When Crossing a Street. Transportation Research Record, 2002, 90-97.

[76] New Straits Times (2002) Too many traffic lights can hinder smooth traffic flow. *New Straits Times (Newspaper in United States),* 9 September 2002. Available from: http://www.highbeam.com/doc/1P1-82695203.html [Accessed on 12 August 2009].

[77] New Zealand Transport Agency (2007) *Pedestrian planning and design guide.* Land Transport New Zealand/ New Zealand Transport Agency, 2007, ISBN: 978-0-478-30945-4.

[78] Ozaki, H. (1993) Reaction and anticipation in the car following behaviour. In *Proceedings of the 13th International Symposium on Traffic and Transportation Theory,* 349-366.

[79] Pan, Z., Li, X., Chen, Y. and Li, W. (2010) Improvement Measures for Beijing Pedestrian and Bicycle Transportation (in Chinese). *Urban Transport of China,* 8 (1), 53-59, 73.

[80] Panis, L., Broekx, S. and Liu, R. (2006) Modelling instantaneous traffic emission and the influence of traffic speed limits. *Science of the Total Environment,* 371 (2006), 270-285.

[81] Papadimitriou, E., Yannis, G. and Golias, J. (2009) A critical assessment of pedestrian behaviour models. *Transportation Research Part F,* 12 (2009), 242-255.

[82] Park, B. and Schneeberger, J. D. (2003) Microscopic Simulation Model Calibration and Validation: Case Study of VISSIM Simulation Model for a Coordinated Actuated Signal System. *Transportation Research Record 1856,* TRB, National Research Council, Washington, D.C., 2003, 185-192.

[83] PTV AG (2008) *PTV Compass – The PTV Customer Magazine for Mobile People. 1st Edition 2008,* available from: http://www.ptvgroup.com/fileadmin/files_ptvag.com/download/news/compass/ptv_compass_0108.pdf [Accessed on 10 September 2012].

[84] PTV AG (2010) *VISSIM Pedestrian Simulation.* Available from: http://www.ptvgroup.com/fileadmin/files_ptvag.com/download/pageflip/brochures/vissim_pedestrian_sim/vissim_pedestrian_sim_files/vissim-ped_e_2010_lowres.pdf [Accessed on 10 September 2012].

[85] PTV AG (2012) *PTV Compass – The PTV Customer Magazine for Mobile People. 1st Edition 2012,* available from: http://www.ptvgroup.com/fileadmin/files_ptvgroup/Downloads/3_News_und_Presse/2_News/3_Kundenmagazin/Compass_International/PTV-Compass_international_2012-01.pdf [Accessed on 10 September 2012].

[86] Pucher, J. and Dijkstra, L. (2003) Promoting safe walking and cycling to improve public health: lessons from the Netherlands and Germany. *American Journal of Public Health,* 93, 1509-1516.

[87] Quadstone Paramics (2012a) *Pedestrians in Street Space.* Available at: http://www.paramics-online.com/pedestrian-simulation.php [Accessed on 5 July 2012].

[88] Quadstone Paramics (2012b) *Case study - customer success stories from the paramics community.* Available from: http://www.paramics-online.com/downloads/casestudies/interaction_newcastle.pdf [Accessed on 5 July 2012].

[89] Rietveld, P. (2001) *Biking and walking; the position of non-motorised transport modes in transport systems.* Tinbergen Institute Discussion Paper, Vrije Universiteit, Amsterdam, the Netherlands.

[90] Ronald, N., Sterling, L. and Kirkley, M. (2005). A conceptual framework for specifying and developing pedestrian models. *The Proceedings of the 16th Biennial Conference of the Modelling and Simulation Society of Australia and New Zealand (MODSIM 2005).*

[91] Rosen, R. (1985) *Anticipatory systems: Philosophical, Mathematical, and Methodological Foundations.* Pergamon Press, Oxford.

[92] Sack, J., Rouphail, N., Park, B. and Thakuriah, P. (2001) Statistically based validation of computer simulation models in traffic operations and management. *Journal of Transportation and Statistics,* Dec, 2001.

[93] Saelens, B. E., Sallis, J. F., Black, J. B. and Chen, D. (2003) Neighbourhood-based differences in physical activity: an environmental scale evaluation. *American Journal of Public Health,* 93, 1552-1558.

[94] Salter, R. J. (1985) *Highway Traffic analysis and Design.* MacMillan Education LTD, Basingstoke, United Kingdom.

[95] Sample, I. (2008) Health: Walking for 45 minutes helps to control diabetes. *The Guardian (web edition),* Monday 28 July 2008. Available at: http://www.guardian.co.uk/science/2008/jul/28/medicalresearch.health1 [Accessed on 22 January 2012].

[96] Schadschneider, A. and Seyfried, A. (2009) Empirical Results for Pedestrian Dynamics and their Implications for Cellular Automata Models. In: *Pedestrian Behavior* (Chapter 2) [Ed: Timmermans H.], Emerald Group Publishing Limited, UK.

[97] Schroeder, B. J. (2008) *A Behaviour-Based Methodology for Evaluating Pedestrian-Vehicle Interaction at Crosswalks.* Dissertation (PhD Civil Engineering), North Carolina State University, United States.

[98] Schroeder, B. J. and Rouphail, N. M. (2007) A Framework for Evaluating Pedestrian-Vehicle Interactions at Unsignalized Crossing Facilities in a Microscopic Modeling Environment. In: *86th Annual Meeting of the Transportation Research Board,* Washington DC, United States, 2007.

[99] Schull, A. (1955) *The probability theory applied to distribution of vehicle on two-lane highways.* Poisson and Traffic, ENO foundation for transportation.

[100] Shared Space Project (2008) *Shared Space.* Available from: http://en.wikipedia.org/w/index.php?title=Shared_space&oldid=307232690 [Accessed 10 June 2008].

[101] SimWalk (2012) *Simwalk – the simulation for pedestrian flows.* Available at: http://www.simwalk.com/simwalk_transport/index.html [Accessed on 6 August 2012].

[102] Sisiopiku, V. and Akin, D. (2003) Pedestrian behaviours at and perceptions towards various pedestrian facilities: an examination based on observation and survey data. *Transportation Research Part F* (6), 249-274.

[103] Sui, Y., Guo, M., Wu, J., Du, Y. and Song, Y. (2008) A microscopic simulation assessment of urban traffic operation optimization. In *3rd China International Road & Traffic Safety Products Exhibition,* Beijing, China, 24-26 November 2008.

[104] Sun, D., Ukkusuri, S. V. S. K., Benekohal, R. F. and Waller, S. T. (2003) Modeling of Motorist-Pedestrian Interaction at Uncontrolled Mid-block Crosswalks. *In: Proceedings (CD) of the Transportation Research Board annual meeting,* 2003, Washington DC.

[105] Taylor, M. A. P. (1976) *Traffic flow theory.* Department of Civil Engineering, Monash University, Australia.

[106] Toledo, T. (2003) *Integrated Driving Behaviour Modelling.* PhD Dissertation, Massachusetts Institute of Technology.

[107] Traffic Management Bureau, Ministry of Public Security of China (2007) *2007 Annual Statistical Report on Road Traffic Accidents (in Chinese).* Traffic Management Bureau, Ministry of Public Security, Beijing, China.

[108] TRB (Transportation Research Board) (2000) *Highway Capacity Manual (HCM).* Transportation Research Board (TRB), Washington, DC, 2000.

[109] Treiterer, J. and Myers, J. A. (1974) The hysteresis phenomenon in traffic flow. In *Proceedings of the Sixth International Symposium on Transportation and Traffic Theory,* Sydney, 13-38.

[110] TSS (2009) *Microsimulator and Mesosimulator in Aimsun 6.1 User's Manual*. TSS-Transport Simulation Systems, 2009.

[111] TSS (2012) *Technical specifications of AIMSUN*. Available from: http://www.aimsun.com/wp/?page_id=27 [Accessed on 4 July 2012].

[112] US EPA (United States Environmental Protection Agency) (2011) *Air Emission Source.* Available from: http://www.epa.gov/air/emissions/multi.htm#multinat [Accessed on 22 November 2011].

[113] Wakim, C., Capperon, S. and Oksman, J. (2004) A Markovian model of pedestrian behaviour. *In: 2004 IEEE Interactional Conference on Systems, Man and Cybernetics.*

[114] Walker, R., Winnett, M., Martin, A. and Kennedy, J. (2005) *Puffin crossing operation and behavior study.* TRL Published Project Report, PPR 239.

[115] Wang, T., Wu, J., Zheng, P. and McDonald, M. (2010a) A Framework for Analysing and Modelling Pedestrian-Vehicle Interaction Behaviour in a Micro-simulation Environment. *Proceedings (CD) of 3rd Transport Research Arena Conference (TRA 2010),* Brussels, Belgium, 7-10, June, 2010.

[116] Wang, T., Wu, J., Zheng, P. and McDonald, M. (2010b) Study of Pedestrians' Gap Acceptance Behavior when They Jaywalk outside Crossing Facilities. *Proceedings of 13th International IEEE Conference on Intelligent Transportation Systems (ITSC 2010),* Madeira Island, Portugal, September 19-22, 2010.

[117] Wiedemann, R. (1974) *Simulation des Straßenverkehrsflusses. Schriftenreihe des IfV, 8, 1974.* Institut für Verkehrswesen. Universität Karlsruhe (in German).

[118] Wisconsin DOT (2002) Freeway System Operational Assessment, Technical Report I-33, Paramics Calibration & Validation Guidelines, DRAFT, June 2002.

[119] World Energy Council (2007) *Transport Technologies and Policy Scenarios to 2050.* London, Kingdom: World Energy Council. ISBN: 0 946121 28 1.

[120] Wu, J. (1994) *Simulation Study of AT-Grade LRT at Signalised Intersections.* Dissertation (PhD Civil Engineering), University of Southampton, United Kingdom.

[121] Wu, J., Brackstone, M. and McDonald, M. (2000) Fuzzy Sets and Systems for a Motorway Microscopic Simulation Model. *Fuzzy Sets and Systems,* 116 (1), 16, 65-76.

[122] Wu, J. (2001) The driving behaviour and traffic flow microscopic simulations study and application. *In 4th National Young Scientists and Research Congress*, Nanjing, China, 2001.

[123] You, X. (2004) *Modern Road Survey and Design (in Chinese).* Beijing: Beijing Jiaotong University Press.

[124] Zheng, P. (2003) *A Microscopic Simulation model of Merging Operation at Motorway On-ramps.* Dissertation (PhD (Civil Engineering)). University of Southampton, United Kingdom.

[125] Zhou, J. and Chen, S. (2004) Study on car-following simulation in microscopic traffic flow. *Journal of Southeast University (Natural Science Edition),* 2004, 34 (4), 545-548.