A MERGING MODEL FOR MOTORWAY TRAFFIC

Jiao Wang

Submitted in accordance with the requirements for the degree of PhD

The University of Leeds
Institute for Transport Studies

February 2006

The candidate confirms that the work submitted is her own and that the appropriate credit has been given where reference has been made to the work of others.

Acknowledgements

I take this opportunity to thank my supervisors, Dr. Ronghui Liu and Mr. Frank Montgomery for their invaluable advice and guidance throughout my PhD study. Their constant support, encouragement and great friendship made my stay at ITS an enjoyable experience. I would like to thank Professor David Watling, the first year supervisor of my PhD study, for his advice and feedback during my doctoral study. Thanks to Dr. Susan Grant-Muller, the PhD research tutor, for her constant help throughout this research.

Special thanks go to my close friends, PhD students Hui Lu and Hazel Baslington who have made my living and studying at ITS an wonderful experience. Thanks also go to Shaopeng Sun, Chandrasekha Balijepalli, Steve Robinson and Nick Stockman for their advice on my PhD thesis.

I would like to thank the staff of NADICS (*National Driver Information and Control System*), Glasgow, UK for their continuous cooperation during the research on driving behaviour in motorway merging sections. I would also like to extend thanks to Mr. Tim Rees of TRL (*Transport Research Laboratory*), UK for providing the M25 motorway data for this study. I deeply appreciate the financial support to my PhD study by a scholarship from the Overseas Research Students Award Scheme, UK, and Institute for Transport Studies, University of Leeds.

Finally, I wish to express my heartfelt thanks to my parents and my husband Feng Qi for their endless support and encouragement.

Abstract

Motorway merging has long been regarded as a major source of conflicts and congestion on motorways. Traditional studies of merging behaviour are based on gap acceptance models developed mainly for urban intersections, which tend to over-simplify the very complex dynamic interactive merging behaviour involved.

It is believed that this research represents the first comprehensive investigation and modelling of dynamic merging interactions at motorway on-ramps. Emphasis has been given to improving the modelling of merging behaviour and in particular to capture the cooperation between the merging and motorway traffic. This research has developed a feasible integrated microscopic simulation framework to model the interactions among traffic in motorway merging sections. This has been achieved by developing an integrated model (MergeSim) consisting of two sub-models working in tandem: a car-following and a merging model.

By assuming different reaction times for different driver states (alert, non-alert and close-following), the new car-following model is shown to be able to capture traffic breakdown, hysteresis, shockwave propagations and close-following situations.

The merging model is developed to capture both the acceleration and gap-acceptance behaviour of the merging traffic, and the cooperative behaviour of the motorway traffic. The merging model is composed of several sub models: for the traffic in the motorway nearside lane, there is a cooperation model to simulate the cooperative lane-changing and courtesy yielding behaviour and the interactions with the merging traffic; for the merging traffic in the acceleration lane, there are models such as acceleration model, gap selection model, gap acceptance model and a merge model.

Sensitivity tests have shown that the integrated model can reasonably replicate all relevant behaviour of individual drivers in merging areas such as normal car-following, close-following, cooperative lane-changing, courtesy yielding and gap acceptance. The sensitivity tests on the different merging lengths showed that increased length might reduce merging failures (i.e. the occurrence that the merging driver fails to move into the motorway before reaching the end of the acceleration lane). It can be explained that more merging traffic can successfully take the following gaps with increased merging lengths, which has implications for the geometric configuration of the acceleration lane.

The study also established a general calibration and validation framework designed for real-world applications in highway networks using the most readily available traffic surveillance data, the loop detector data. Currently no commonly agreed bench-marking procedure exists (Brockfeld et al., 2005), and this framework has the advantage that the concept and the proposed methodology are suitable for general application to other micro-simulation models using detector data sets.

In conclusion, the integrated simulation model (MergeSim) can reliably be used as a tool for further studies and investigations into the effectiveness of techniques related to motorway merging operations.

Declaration

Some parts of the work presented in the thesis have been published or submitted in the following articles and conferences:

Wang, J., Liu, R. & Montgomery, F. O. (2005a) A car-following model for motorway traffic, Transportation Research Record **1934**, pp33-42.

Wang, J., Liu, R. & Montgomery, F. O. (2005b) A simulation model for motorway merging behaviour, Presented and Published in the *Proceedings of the 16th International Symposium on Transportation and Traffic Theory*, Elsevier, U.S.A., pp281-301.

Liu, R. and Wang, J (2005) A general framework for the calibration and validation of car-Following models along an uninterrupted open highway, *4th IMA International Conference on Mathematics in Transport*, London, U.K., September 2005.

Wang, J. (2005) A simulation model for motorway merging behaviour, *37th UTSG Annual Conference*, Bristol, U.K., January 2005.

Wang, J. (2004) Car following models for motorway traffic, *Workshop on Modelling Link Flows and Travel Times for Dynamic Traffic Assignment*, Queen's University, Belfast, U.K., September 2004.

Wang, J. (2003) Car-following models for motorway traffic, *35th UTSG Annual Conference*, Loughborough, U.K., January 2003.

CONTENTS

FIGURES

TABLES

CHAPTER 1
INTRODUCTION

1.1. BACKGROUND

As one of the most complicated and important aspects of motorway operation, motorway merging has been regarded as being closely related to motorway flow breakdown and traffic congestion (e.g. Hounsell and McDonald, 1992; Zheng, 2002). Conflicts often occur between vehicles at the motorway merging areas. Usually, in order to avoid collision and disturbance from the ramp traffic, the nearside motorway traffic is often observed shifting to the offside lanes or slowing down to create gaps for the merging traffic.

Conflicts of traffic at motorway merging areas have already attracted the attention of many researchers (e.g. Evans et al., 2001; Kita et al., 2002). However, one particular problem that the existing motorway traffic models do not adequately represent, is the interactive behaviours between the two traffic streams involved in the merging area: the merging traffic and the motorway traffic. Traditional studies of merging behaviour have been based on the gap acceptance models developed mainly for urban give-way intersections, which over-simplified the very complex dynamic interactive merging behaviour involved. The assumption of the gap acceptance model is that merging manoeuvres have no influence on the motorway traffic (e.g. Yang and Koutsopoulos, 1996). But it is often observed empirically that motorway traffic exhibit cooperation to facilitate the merging traffic manoeuvre (e.g. Elefteriadou et al., 1995; Evans et al., 2001) and that merging vehicles continuously adjust their speed and position to fit into their target gaps on the motorway (Michaels and Fazio, 1989). Failure to model the interactions between merging and motorway traffic may lead to inaccuracy in the traffic analysis on a merging section (Kita et al., 2002).

Micro-simulation is a promising approach for achieving a better understanding of motorway driving behaviours (Brackstone and McDonald, 1995). In the past several years, the use of micro-simulation has been widespread in the traffic engineering community (Bachem et al., 1996). There are several micro-simulation software developed specifically for motorway traffic, e.g. VISSIM (PTV, 2003), FRESIM (TRB, 1997) and SISTM (1993), etc. "*Most models only use some simple logic to describe very complex merging behaviour, typically gap-acceptance logic.*

These models often demonstrate a poor ability to reproduce realistic merging operations." (Zheng, 2002). For example, according to Webster and Elefteriadou (1999), "*FRESIM can reasonably predict real world traffic behaviour. Exceptions were in merging and weaving operations*". Based on the theoretical analyses and numerical simulations, this research will focus on setting up more realistic microscopic motorway traffic flow models for merging areas, and help to better understand the relationships among merging operation performance, geometric configurations and cooperative motorway driving, all of which have profound implications for motorway traffic management.

Two models are developed for this study: a car-following model and a merging model. The car-following model aims to capture some of the key motorway flow characteristics, namely traffic breakdown, hysteresis, shockwave propagation as well as close-following behaviour. The merging model is developed to represent the acceleration and gap acceptance behaviour of the merging traffic, and the cooperative behaviour of the motorway traffic. Instead of modelling different behaviours separately as in existing studies, an integrated simulation framework is applied so as to capture the inter-dependencies between different behaviours. "*In order to model a more sophisticated driving behaviour it is necessary to account for inter-dependencies between behaviours.*" (Toledo, 2003). In this study, the new car-following model and merging model are calibrated separately with M25 and M27 motorways data at first. This is then followed by a calibration of the integrated simulation of the car-following and merging behaviour through a case study of a motorway merging site at Junction 27, M8 motorway. The next section introduces the relevant terms and definitions applied throughout the rest of the thesis. The specific objectives of the research are given in section 1.3 and followed by the outline of this thesis.

1.2. TERMS AND DEFINITIONS

Merging Manoeuvre

Before the description of the merging manoeuvre, a typical merging area of UK motorway is illustrated in Figure 1-1. This is the area where a merging vehicle on the acceleration lane directly interacts with the motorway through traffic on the nearside motorway lane. The merging area starts from the point where the right edge of the

ramp intersects the left edge of the motorway and ends at the point where the left edge of the acceleration lane intersects the left edge of the motorway.

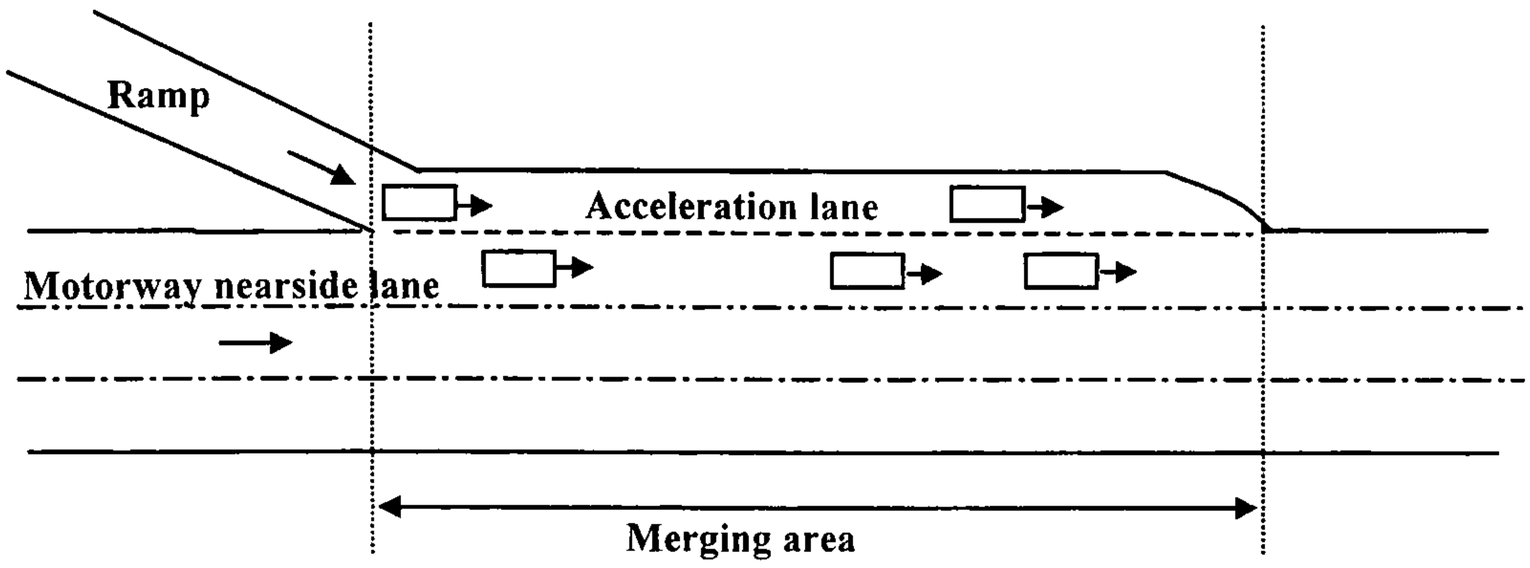


Figure 1-1 A motorway on-ramp merging area.

Merging Manoeuvre is executed by the merging vehicles to enter into the mainstream traffic. It requires the driver to perform an iterative task of speed adjustments and gap searching while running on the acceleration lane (Michaels and Fazio, 1989). Merging failure is the occurrence that the merging driver fails to move into the motorway before reaching the end of the acceleration lane.

Traffic Breakdown

The occurrence of traffic breakdown on a section of motorway can be identified with reference to the speed and flow profiles. It is taken to correspond to a rapid reduction in speeds and flows, which may result in a standing queue (Hounsell et al., 1992).

Traffic Hysteresis

Traffic hysteresis is a phenomenon characterised by the loop structure observed from empirical flow-occupancy plots, where the capacity of the traffic flow recovering from a traffic breakdown does not reach the capacity level before the breakdown (Zhang, 1999; Zhang and Kim, 2001).

Shockwave

"*Flow- speed- density states change over space and time. When the distinct discontinuity in traffic flow and density occur, a boundary is established. The boundary is referred to as a shockwave. It can occur when a platoon of high-speed vehicles approaches a queue of slow-moving vehicles, and vice versa*" (May, 1990). Shockwave can be easily identified from the plot of individual vehicle trajectories of traffic with significant increase/reduction of traffic flow, velocity and density.

Close-following Behaviour

Close-following behaviour is often observed on motorways (before traffic congestion) where a group of vehicles is driving with high speed and very small gaps (such as time gap intervals below 0.8 seconds) (Brackstone et al., 2002).

Micro-simulation Approach

This is an approach to study the flow of traffic by modelling the motion of each interacting individual driver-vehicle unit in the traffic. Each driver-vehicle unit has specific properties that are determined stochastically, such as vehicle length, driver aggressiveness, etc. The control of the vehicle's movement is based on the individual driver's decision/action relative to other vehicles and motorway geometry (May, 1990; Brackstone and McDonald, 1995).

Macro-simulation Approach

This is an approach to study traffic by modelling it as a fluid flowing along the carriageway. Viewing the collections of vehicles as an entity, one considers global quantities like speed, density and flow on certain road segments. It assumes identical vehicles, and ignores lane-changing and overtaking. Estimation of the traffic speed or flow is based on certain assumptions, for instance that the speed of traffic adjusts instantaneously to traffic density, etc (May, 1990).

1.3. OBJECTIVES AND SCOPE OF THIS STUDY

This study will focus on the modelling of traffic behaviour at a motorway merging area using a micro-simulation approach. Development of a feasible microscopic simulation framework is proposed by considering the interactions between the two traffic streams involved, namely the merging traffic and the motorway traffic. Two models are developed here: a car-following model and a merging model. The objective of this study is to improve modelling of merging driving behaviour in the motorway merging sections and in particular capture the interactions between the merging traffic and motorway traffic. The study has sought to contribute to the state-of-the-art in modelling driving behaviour in the following aspects:

- A framework for integrated simulation of driving behaviour in motorway merging sections is proposed. This framework captures both car-following behaviour and merging behaviour on UK motorways.
- The development of the car-following model aims to capture the traffic breakdown, hysteresis, shockwave propagation and close-following phenomena as often observed in real traffic.
- The merging model is proposed to explicitly simulate interactions such as the acceleration and gap-acceptance behaviour of the merging traffic and the cooperative behaviour of the motorway traffic. The merging vehicles' movements are controlled by several sub-models: the gap selection model, the acceleration model, the gap acceptance model and the merge model. The acceleration model applies when interacting with the motorway traffic and the other models consider all involved vehicles' actions in the merging process (e.g. cooperative lane-changing or yielding from the nearside motorway traffic).
- The calibration of the models involves two steps: firstly, the two models are separately calibrated with data from the M25 and M27 motorways. The integrated simulation model, which includes both of the two models, is then calibrated and validated through the case study of the merging section on M8 motorway.

1.4. THESIS OUTLINE

Chapter 2 introduces the methodology of this study. It includes an introduction to micro-simulation approach at the beginning of this chapter and briefly describes a conceptual simulation framework. It also introduces vehicle generation method, data collection, and method of model verification, calibration and validation.

Chapter 3 begins with a review of motorway traffic flow characteristics and observed motorway driving behaviour. A review of the existing car-following models is also included and followed by a detailed examination of several existing car-following models. Based on the problems highlighted from the existing models, a new car-following model is proposed and analysed through both theoretical analysis and simulation tests. The calibration and validation of the model with M25 motorway data is also presented.

In Chapter 4, a description of the interactions in a merging process is described. This is followed by a literature review of the existing merging studies. By considering the interactions between the merging traffic and motorway traffic, a new merging model is developed. The new model explicitly simulates the courtesy yielding and cooperative lane-changing of the motorway traffic and the acceleration and gap selection, acceptance behaviour of the merging traffic. From numerical tests, the model performance is evaluated. Finally the merging model is calibrated against the data collected by Zheng (2002) on M27 motorway.

Chapter 5 describes the calibration, validation and application of the integrated simulation framework of driver's car-following and merging behaviour using data collected from a section of the M8 Motorway, Glasgow.

Finally, conclusions and direction for future research are presented in Chapter 6.

CHAPTER 2
METHODOLOGY

As discussed in Chapter 1, there is a need for an integrated simulation framework for modelling the interactions among the drivers in motorway merging sections so as to accurately represent the merging operations. The available mechanisms within the framework (car-following and merging models), which will be discussed in detail in the following chapters, enable the simulation to capture the inter-dependencies between the two models involved. An interactive microscopic traffic simulation approach has been formulated featuring realistic merging operations. A self-contained program "MergeSim" (source codes of main sub-routines given in Appendix A) is implemented for this study (available from ITS, University of Leeds), which was developed in software Visual C++.

This chapter is organised as follows: first, it introduces the micro-simulation approach and a conceptual framework for simulating the driving behaviours in motorway merging sections. The method of vehicle generation is then discussed, which aims to represent the "driver-vehicle unit" with its relevant attributes. Vehicle movements will be briefly introduced here with the detailed logic and algorithm illustrated in the following chapters. Simulated data collection is explained with respect to the data collected at two levels, namely micro-level and macro-level. Finally, the method for model verification, calibration and validation is proposed.

2.1. MICRO-SIMULATION APPROACH AND INTEGRATED SIMULATION FRAMEWORK

For many years, the modelling of road traffic flow was dominated by the macroscopic approach, which usually involved describing the traffic as a fluid flowing along the carriageway (Addison and Low, 1998). In recent years dramatic improvements in computer power have made it possible to study the flow of traffic by modelling the motion of each interacting individual vehicle in the traffic (Bloomberg and Dale, 2000), i.e. microscopic simulation approach. This approach involves the modelling of individual vehicle behaviour including the interaction of

vehicles with their environment (Middelham, 2001). The efficiency of microscopic simulation models nowadays allows for traffic flow to be reproduced in real time. By defining carefully the details of car motions, the simulation can yield the correct single-vehicle data in various situations, e.g. bottlenecks, junctions (Schreckenberg et al., 2001). The analysis of such detailed data gives important information about the driving behaviour of vehicles in the various traffic states (Knospe et al., 2002).

There are many advantages of using a micro-simulation approach to study the properties of motorway traffic flow, especially for the motorway merging sections. First, it is clear that phenomena such as the interactions among the merging traffic and motorway traffic are much more influenced by individual drivers, implying that an approach using microscopic simulation is necessary. On the motorway, the subtle changes in individual behaviour may cause a significant change to the 'bulk' behaviour. Macro-level analysis, however, uses massive simplifications such as: the assumption of identical vehicles, drivers' reaction time equal to zero, ignorance of fast and slow lanes, ignorance of overtaking (Papageorgiou, 1998). Instead of the assumption of identical vehicles, the microscopic simulation is stochastic in that each vehicle has specific properties that are determined stochastically (Hansen et al., 2000). "*It is plain that the domain in which the microscopic models are most prone to problems is that of motorway.*" (Brackstone et al., 1995). These "stochastic" vehicle properties may include vehicle length, driver aggressiveness, reaction time, acceleration rate, etc. Important advantages of the microscopic approach over a macroscopic are, very detailed output data, comprehensible processes, good visualisation thus simple error handling, subtle effects to be observed and interpreted, and in-depth analysis of parts of the motorway.

This research is focused on the modelling of motorway traffic at a merging area with a micro-simulation approach, which requires the development of a microscopic model by considering traffic flows interactions on merge.

Object-oriented languages (e.g. C++) have become popular platforms since they support the concept of reusable software by defining objects to solve a programming task.

The framework of the simulation program is shown in Figure 2-1, which includes user interface, core algorithms, visual simulation and data outputs:

- **User Interface** has two components: simulation input settings and network settings. The simulation input settings enable the user to manage simulation parameters, driver-vehicle unit properties (vehicle length, acceleration and desired speed, etc.), and traffic input (i.e. the traffic flow, speed and composition profiles at the periphery of the analysis network). The network

settings allow the user to set up, amend the simulated network and locate loop detectors for simulated data collection according to the empirical test site.

- **Core Algorithms** include vehicle generation and vehicle movement control. The vehicle generation algorithm generates vehicles according to driver-vehicle unit properties given in the *User Interface* as discussed earlier. Vehicle movement in this study is based on the combination of car-following rule and merging rule. When a vehicle is moved, its position on the network and its relationship to other vehicles nearby are re-calculated, as are speed, acceleration, and distance/time gap between individual vehicles, etc. by computer simulation.

- **Visual Animation** is also implemented in the framework to provide real-time visualisation of the simulation output. It helps the user to review coding accuracy of the core algorithms and the reasonableness of the simulation input settings, network settings etc. It is also useful for the user to examine the consistency of the on-screen animation results under different test conditions.

- **Data Collection** is another important part of the program where the relevant simulation results can be exported from simulation runs for further analysis. It is composed of two types of outputs: single-vehicle data (speed, position, acceleration, etc.) and detector measurements (speed, flow and occupancy). The detailed data collection is discussed later in section 2.4.

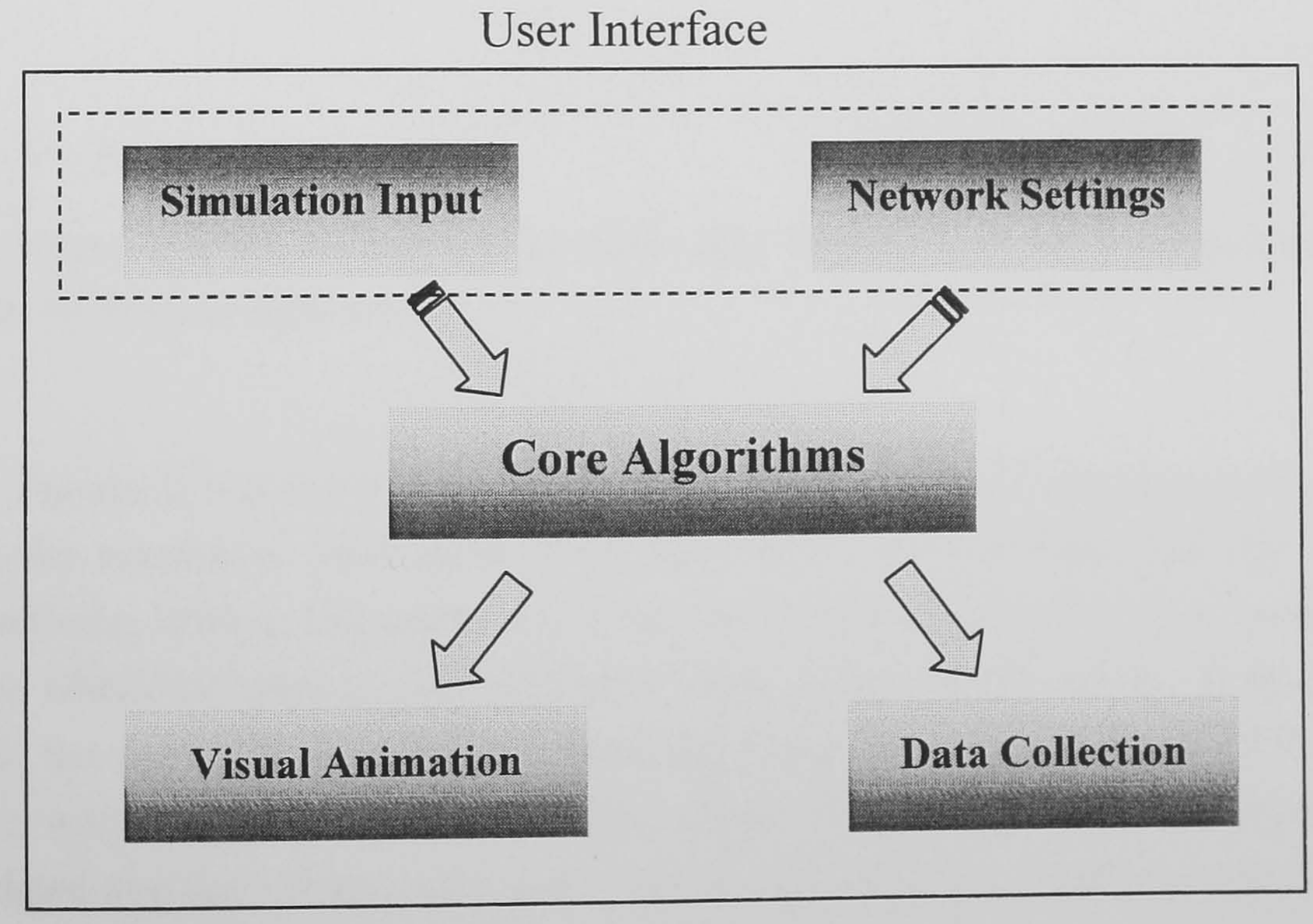


Figure 2-1 Simulation Framework.

2.2. VEHICLE GENERATION

"*At the outset of a simulation run, vehicles are generated at origins, usually at the periphery of the analysed network, according to some headway distribution based on specified volumes.*" (Zheng, 2002). In this study, the traffic arrives at the simulated section randomly with the speeds and flows available from the user settings. A "driver-vehicle unit" can be defined in terms of its relevant vehicle attributes (e.g. vehicle length) and driver attributes (e.g. driver's desired speed, aggressiveness).

2.2.1. Traffic Arrival

Traffic arrival is a random occurrence of discrete events. Early traffic engineers investigated distributions as a means of describing the occurrence of vehicle arrivals during a certain interval. The Poisson distribution has been applied to describe traffic arrivals since 1933 as it is "*the appropriate distribution for describing the truly random occurrence of discrete events*" (TRB, 1996). In this study, vehicle counts are generated from a Poisson distributed traffic flow, which can be denoted as:

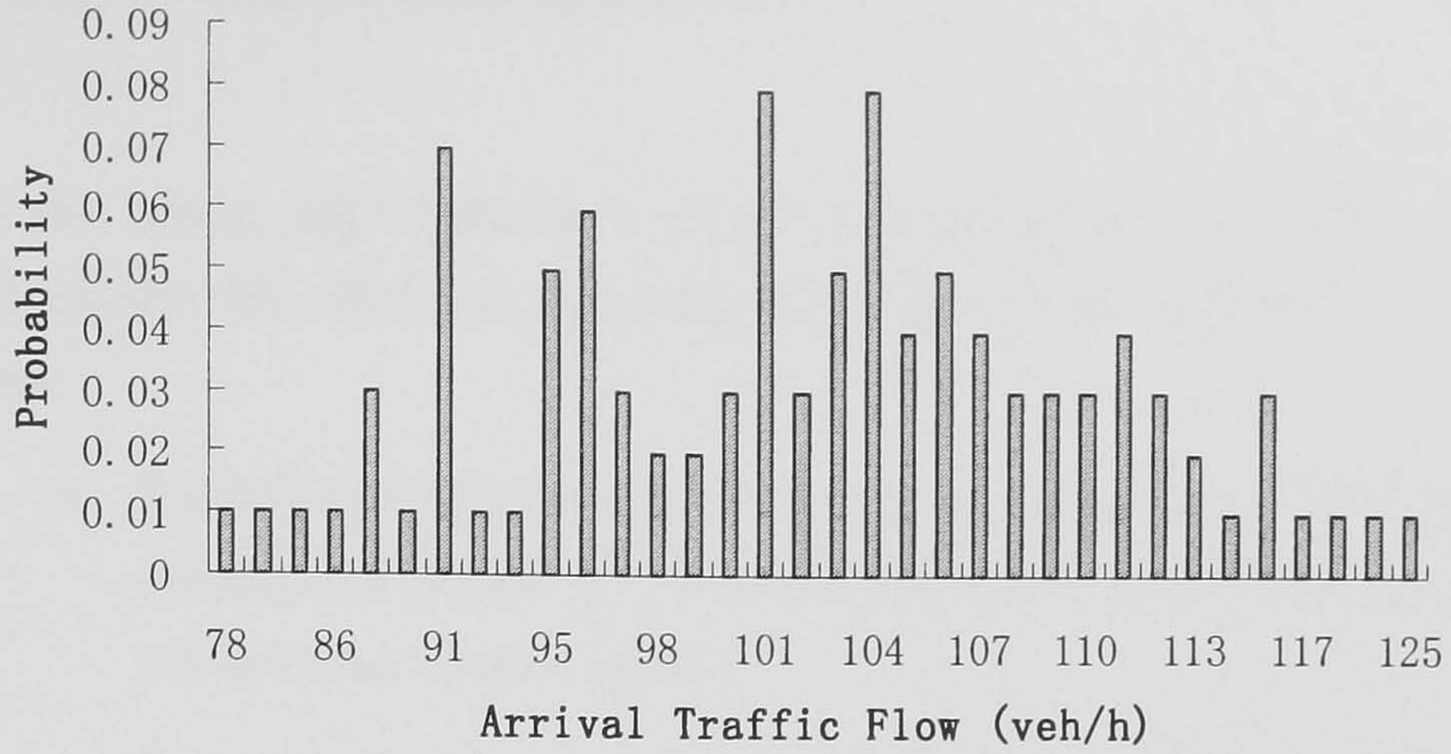
$$N = Poisson\ (q) \tag{2-1}$$

where, q is the average arrival traffic flow (veh/h); N is the generated vehicle counts by Poisson distribution[1].

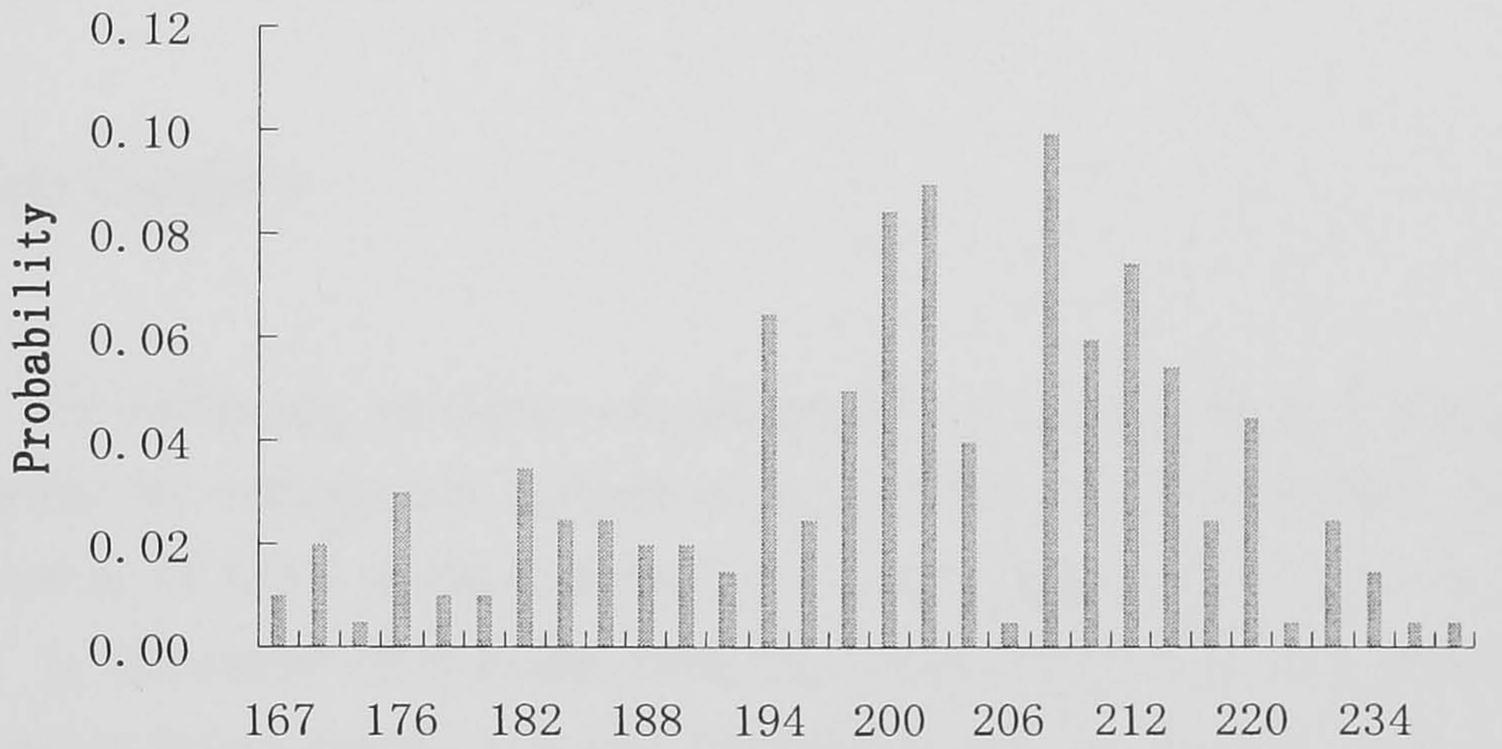
Figures 2-2(a) and (b) give two examples of the Poisson distributed traffic flow from the simulation, with input of average value q of 100 veh/h and 200 veh/h respectively. With q=100 veh/h, it is found that the generated arrival flow values are varied within the range of (78, 125) veh/h, with average of 101 veh/hr. With q=200 veh/h, the generated arrival flow values are varied within the range of (167, 240) veh/h, with average of 201 veh/h. These slight differences between the input and simulated average values (with only 1 veh/h difference) are reasonable due to the random effects (Press et al., 1992).

---

1 The C language coding of the Poisson distribution in the program is according to Press et al. (1992).



(a) Generated from the average arrival flow 100 veh/h



(b) Generated from the average arrival flow 200 veh/h

Figure 2-2 Examples of vehicle generation from different arrival traffic flow.

Thus, vehicle is generated with the interval, denoted as Δt_{gen} (unit: second), simply described as:

$$\Delta t_{gen}=3600/N \tag{2-2}$$

2.2.2. Driver-vehicle Unit Attributes

In this study, each simulated vehicle is represented as a "driver-vehicle unit" which includes the physical attributes of the vehicle together with its driver's attributes:

- Vehicle attributes: category (i.e. HGV or passenger car), length and the mechanical limits of acceleration/deceleration corresponding to the current speed and category;
- Driver attributes: driver aggressiveness and desired speed.

2.2.2.1. Vehicle Attributes Generation

Vehicle Category

The following attributes are assigned to a vehicle when it is generated. Two categories of vehicles are defined in the simulation: Car and HGV. With a certain proportion of HGV in the vehicle composition (denoted as x, $0<x<1$), a generated HGV is Binomial distributed with the proportion of x. An example of HGV generation from a sample test (one-hour simulation) is given in Table 2-1.

Table 2-1 Vehicle composition of a 1-hour simulation test with arrival flow 500veh/h and HGV proportion x=0.15.

Vehicle Category	Generated vehicle counts in one hour period	Vehicle Composition	
		Input	Simulation
Car	420	0.85	0.83
HGV	84	0.15	0.17
Total	504	1	1

As illustrated in Table 2-1, 504 vehicles are generated in a one hour period with a mere difference of 4 vehicles due to the random effects of Poisson distribution. From the simulation results of vehicle generation, the simulated proportion of HGV composition is 0.17, which approximates the input composition of 0.15. It shows that the generated vehicle composition represents the vehicle

generation very well with only 0.02 differences in proportion due to the random effects of Binomial distribution.

Vehicle Lengths

For each vehicle generated within a defined category (sampled from the corresponding vehicle composition), a corresponding length attribute needs to be specified. From the empirical observations on UK motorways (El-Hanna, 1974), the vehicle lengths are Normally distributed according to the category, i.e. HGV or Car (Table 2-2).

Table 2-2 The Normal distributed vehicle lengths on UK motorways (El-Hanna, 1974).

Vehicle category	Mean (m)	Std. Deviation (m)
Car	4.2	0.4
HGV	11.2	2.4

Vehicle Mechanical Limits on Acceleration/Deceleration

For each vehicle, the mechanical limits on its acceleration/ deceleration capabilities are applied in the acceleration calculation in the simulation as a functional relationship (associated with its current speed and vehicle category). Table 2-3 lists the mechanical limits for cars under different speed levels. For HGVs, their values are three-quarters those of a car (Goodman, 2001).

Table 2-3 The mechanical limits on acceleration/deceleration for passenger car under different speed levels (ITE, 1999).

Speed (km/h)	0-32	32-48	48-64	64-80	80-96	>96
Acceleration Limit a_{max}^{+} (m/s^2)	2.4	2.0	1.8	1.6	1.4	1.4
Deceleration Limit a_{max}^{-} (m/s^2)	4.9					

2.2.2.2. Driver Attributes Generation

Two attributes, namely, the desired speed and aggressiveness are assigned to a driver in each simulation.

Desired Speed

A driver's desired speed is the speed that he/she wants to attain under free-flow situation. In this study, the "desired speed" is referred to the observed speed in the data to be replicated by the simulation. From the observed detector data, the desired speed can be estimated from the speed-flow relationship at the flow of 300veh/h according to Duncan (1976). From empirical loop detector data, the desired speed of HGV and car on the nearside lane can be extracted from the traffic information (speed, flow and vehicle composition) with flow below 300 veh/h. It is found that HGV has a lower desired speed than Car, which might be due to the lower speed limit for HGV (60 mile/hour) compared to car (70 mile/hour) on UK motorways generally (www.highwaycode.gov.uk). It is also found that the desired speeds may vary among different sites, which might be caused by the local driving conditions. Thus, to test the simulation model on an empirical site, the user should firstly calibrate the desired speeds with respect to the local area's driving conditions.

Table 2-4 compares the desired speeds of motorway nearside lane extracted from two UK motorway sites data, and the local speed limits in the corresponding sites. The desired speeds of **Site 1** are extracted from the loop detectors data collected between Junction 11 and 12 on M25 motorway, on 28 February 2001. The desired speeds of **Site 2** are extracted from loop detectors data collected on Junction 27, M8 motorway, Glasgow, on 25 July 2002. As shown in Table 2-4, it is found that generally, the desired speeds extracted from Site 1 are around 6-10 km/h higher than that of Site 2, which might be due to the higher speed limits on Site 1. In the simulation of an empirical motorway section, the desired speeds of each vehicle are assumed being normally distributed (Gipps, 1981) corresponding to the vehicle category and the sites.

Table 2-4 Desired speeds from empirical detector data.

Category	Site 1		Site 2	
	(Junction 11 and 12, M25)	Speed Limit	(Junction 27, M8)	Speed Limit
Car (km/h)	Mean: 109.2	113 km/h (70mile/h)	Mean: 100	97 km/h (60mile/h)
	Variance: 9.3^2		Variance: 9.8^2	
HGV (km/h)	Mean: 91.8	97 km/h (60mile/h)	Mean: 85.3	
	Variance: 14.5^2		Variance: 8.7^2	

Driver Aggressiveness

Driver aggressiveness is applied in the simulation of merging vehicles' speed control abilities based on the assumption that the more aggressive drivers may be able to adjust their speeds and positions more rapidly than the more timid drivers, when they are looking for gaps on the motorway. Driver aggressiveness is defined as a factor (uniformly distributed between 0 and 1) applied on the driver's acceleration, according to five different uniformly distributed driver types (Very timid, Timid, Average, Aggressive and Very aggressive). More details about the application of the driver aggressiveness can be found later in Chapter 4.

2.3. VEHICLE MOVEMENTS

As discussed earlier, an integrated logic is needed for modelling the interactions among the drivers in motorway merging sections so as to accurately represent the related driving behaviours during merging operations. The available mechanisms of the car-following model and merging model within the simulation framework, which will be discussed in detail in the following chapters, enable the simulation to capture the inter-dependencies between the two components, i.e. car-following model and merging model.

As shown in Figure 2-3, the basic control of individual vehicle motions is the combination of car-following and merging models, which is similar to Zheng (2002). In each simulation step, firstly, new traffic is generated at the periphery of the

analysed network, i.e. at the start point of simulated section on the motorway road and the on-ramp slip road. The speed and position of each of the individual vehicles are updated according to their current situations in the simulated network: for the traffic running in the merging area, the merging model is applied to simulate all interactive behaviours involved such as motorway traffic cooperation, merging traffic gap searching/ acceptance behaviours, etc; for the motorway traffic not in the vicinity of merging area, the car-following model is simply applied to simulate the "follow-the-leader" behaviour. After all vehicles are processed in this simulation step, the model records the counts of passing vehicles and their speeds when passing through the assigned detectors, and the simulation moves towards the next time step.

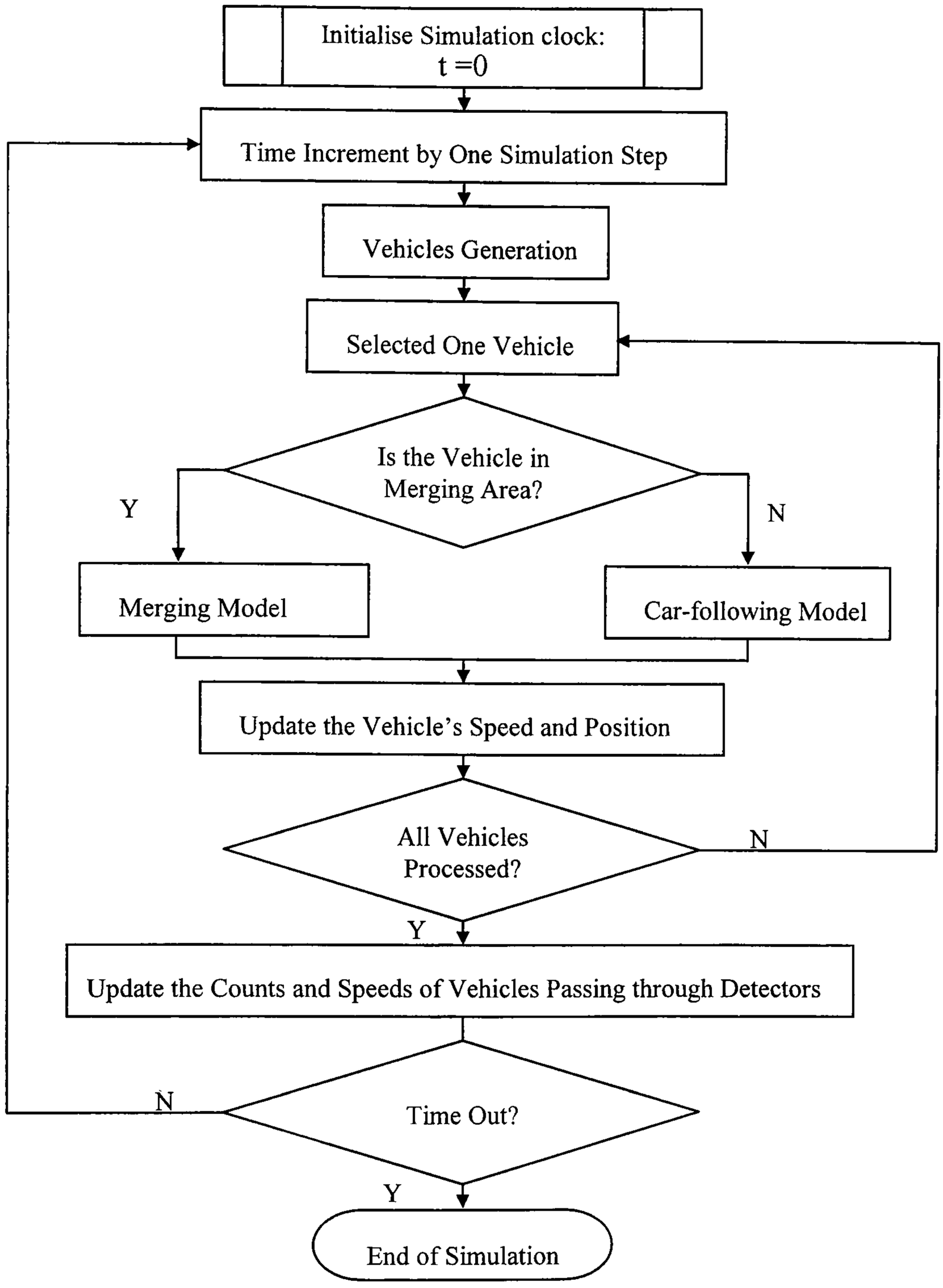


Figure 2-3 The simulation step of vehicle's movements.

2.4. SIMULATION DATA COLLECTION

Data collection is important for simulation model development. Data are collected and compared with the graphs of the corresponding variables of the empirical data to investigate characteristics such as similarity in pattern, magnitude of values, deviation and trend (Zhang et al., 1998). To capture initial conditions, in this study, the simulation begins with a period of "warm-up" before the data collection. The investigated simulation output variables include:

- At micro-level, individual vehicle position, speed and gap profiles over time are collected. Especially, for the analysis of merging manoeuvres, this can provide detailed information of the gap selection, acceleration/deceleration and merging position properties during merging process.
- At macro-level, the aggregated detector data such as speed, flow and occupancy are collected with respect to the empirical aggregated measurements from loop detectors.

2.4.1. Data Collected at Micro-level

For the data collected at micro-level, the related information is recorded during each simulation step. In general, each individual vehicle record includes the basic characteristics such as the vehicle ID (generated when the vehicle arrives at the simulation section), Time (in the simulation run), Position and Speed. Some other characteristics i.e. Acceleration and Gap (with its current leader) profiles over time can be derived based on the above-mentioned individual position and speed information. For the analysis of the merging behaviour, some other information also needs to be recorded as given below. The use of the data will be elaborated later in Chapter 4.

- the accepted lead and lag gaps measured in distance (m);
- the value of driver aggressiveness, speed relative to motorway vehicles, current position (relative to the end of merging section), and the cooperation from motorway traffic (i.e. courtesy or lane-changing cooperation); and
- the selected gap structure (i.e. taking the original gap or other gaps).

2.4.2. Data Collected at Macro-level

Usually the available data in empirical observation are aggregate measurements, e.g. flows, speeds and occupancies at loop detector locations. These data are the emergent results of the interactions between various behaviours of individual vehicles (Toledo, 2003). For model calibration and later validation in the simulation, the detector data are collected in order to achieve the direct comparison between simulated and observed traffic conditions. Each of the detectors in the simulation is two metres long in accordance with the empirical detector length.

2.4.2.1. Speed (V)

Traditionally, there are two principal average speeds, the time mean speed (spot speed) and the space mean speed. There are two other ways to gather speed, i.e. aerial photographs at different instants and moving observers (Daganzo, 1997). In this study, similar to aerial photographs' speed collection, the speed collected by the detector over a constant interval (ΔT) is the average of all speeds of passing vehicles being recorded by the detector (L in length) during each simulation step (such as 0.2 second). Figure 2-4 and equation 2-3 illustrate these speed measurements.

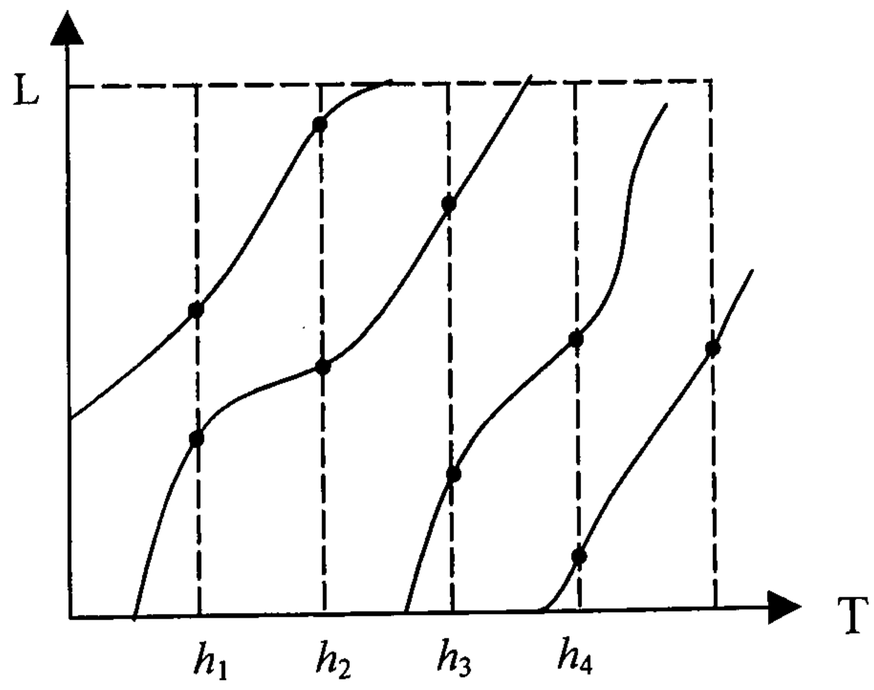


Figure 2-4 Speed collection.

$$V(t,t+\Delta T)=\frac{\sum_{h=t}^{t+\Delta T}\sum_{i=1}^{M} v_i(h)}{M} \qquad (2\text{-}3)$$

where, $V(t, t+\Delta T)$ is the speed collected by detector over ΔT; $v_i(h)$ is the speed of vehicle i that passed the detector at time h; M is the total number of speeds measurements being recorded during the interval (t, t+ΔT) (i.e. number of points in Figure 2.4).

2.4.2.2. Flow (q) and Occupancy (Occ)

The data used to calculate the average flow is collected through point measurement as defined in TRB (1996). The total number (N) of vehicles passing the start point of each detector over the time period ΔT is recorded. The aggregated flow $q(t, t+\Delta T)$is calculated as:

$$q(t,t+\Delta t)=\frac{N}{\Delta T} \qquad (2\text{-}4)$$

Occupancy is the fraction of time that vehicles are over the detector. It is calculated as the sum of the time that vehicles are over the detector, divided by the interval ΔT. For each individual vehicle, the time spent over the detector is determined by the vehicle's speed, $v_i(t)$, and its length, L_i, plus the length of the detector d itself.

$$Occ(t,t+\Delta T)=\frac{\sum_{i=1}^{K}\frac{(L_i+d)}{v_i(t)}}{\Delta T} \qquad (2\text{-}5)$$

where, K is the number of vehicles over the detector during the interval $(t, t+\Delta T)$.

2.4.2.3. Density (ρ)

Density (ρ) is the number of vehicles per unit length, i.e. a measurement along a distance. It is the inverse of the average of all space headways of passing vehicles. According to TRB (1996), assuming a uniform vehicle length, L, a simplified relationship between occupancy and density can be described as following:

$$Occ = (L + d)\rho \tag{2-6}$$

Practical motorway traffic management (e.g. loop detector measurement) makes extensive use of occupancy instead of density. At present practical work tends to favour the use of occupancy because "*density, as vehicles per length of road, ignores the effects of vehicle length and traffic composition.*" (TRB, 1996). In this study, both occupancy and density are collected. The occupancy collected in the simulation can bring a direct comparison to the real loop detector observations. The density variables collected in this study, are used for the shockwave speeds' calculation (explained later in Chapter 3). In this simulation, the so called "collection area" of 40 meters (centred at the middle of the detector's location) is applied for density collection according to the test design by Zhang and Kim (2001). The recorded density is based on the inverse of the average space headways when the platoon is passing through the above-mentioned collection area. Under different situations, the density will be collected as follows (Figure 2-5):

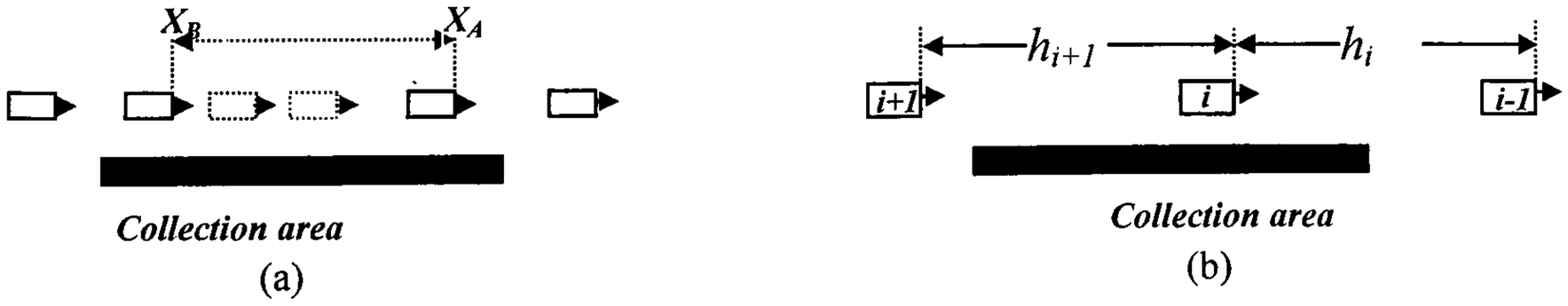


Figure 2-5 Density collection for different situations. (a) More than one vehicle on the collection area. X_A denotes the front position of the first vehicle on the collection area at time t, X_B denotes the front position of last vehicle on the collection area at time t; (b) One vehicle on the collection area. h_i denotes the headway of vehicle i at time t, h_{i+1} denotes the headway of vehicle $i+1$ at time t.

***N* vehicles (> 1) on the collection area** (as shown in Figure 2-5 (a))**:**

The density is calculated as $\rho(t)$ (veh/m):

$$\rho(t) = (N-1)/(X_A - X_B) \tag{2-7a}$$

One vehicle on the collection area (as shown in Figure 2-5 (b))**:**

- Short headway of vehicle *i* (h_i <5s, assumed to be in the following state (May, 1990):

$$\rho(t) = \frac{1}{h_i} \tag{2-7b}$$

- Short headway of vehicle *i*'s follower, vehicle *i+1* (h_{i+1}<5s):

$$\rho(t) = \frac{1}{h_{i+1}} \tag{2-7c}$$

- Both h_i and h_{i+1} are below 5 seconds:

$$\rho(t) = \frac{2}{h_i + h_{i+1}}$$

When there is no vehicle on the collection area, there would be no record of density at this moment.

2.5. PROPOSED METHOD FOR MODEL VERIFICATION, CALIBRATION AND VALIDATION

2.5.1. Basic Definitions

Generally, before the application of a simulation model, three tasks should be performed: verification, calibration and validation in order to achieve the best reproducibility of field conditions. The relationship of these three tasks is depicted graphically in Figure 2-6 given by May (1990).

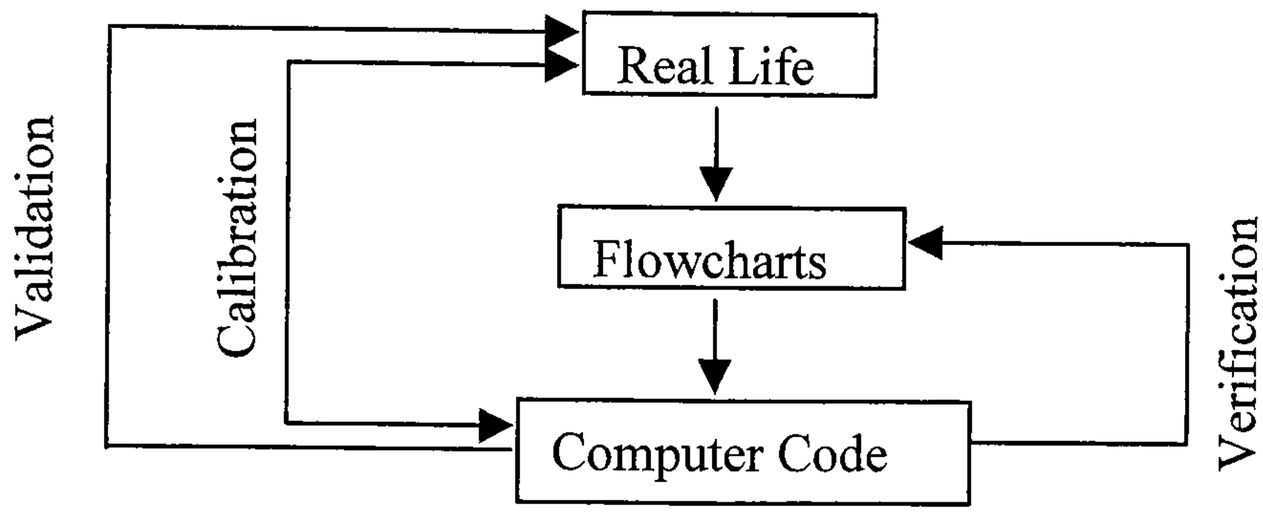


Figure 2-6 Verification, Calibration and Validation Process (May, 1990).

Verification is concerned with determining whether the model assumption has been correctly translated in a computer "program", i.e. debugging the simulation computer program (May, 1990; Law & Kelton, 2000). This is achieved by running the simulation under a variety of settings of input parameters, and checking to see that the output is reasonable (e.g. by observing an animation of the simulation output). Sensitivity analysis can provide evidence whether the results are consistent and reasonable or not. At this point no attention needs to be given to the real-life situation.

The next task is calibration. Calibration is the process of quantifying model parameters using empirical data. Reasonable adjustments are made to match the model's output with real-life observations/ data (Zheng, 2002; May, 1990). For some of the parameters that can not be obtained directly from the field data, there is an alternative approach – to perform simulation runs with several parameter settings and to compare the simulated measures with the field measures (Kosonen, 1999).

The final task is validation. Validation is a process of determining whether a simulation model is an accurate representation of real-life observations/ data under the conditions tested. No adjustments are made to the model and simulation results

are compared with those measured from the field. If the simulated measures match sufficiently with the field data then validation was successful, otherwise further calibration and validation are required (May, 1990; Kosonen, 1999).

Based on these definitions, the detailed methods for model verification, calibration and validation are proposed in the following sections.

2.5.2. Proposed Method of Model Verification, Calibration and Validation

Although verification is simple in concept, debugging a large-scale simulation program is arduous due to the complicated logical paths (Law and Kelton, 2000). A valid simulation model should be able to produce reasonable vehicle actions/ movements. Unrealistic behaviour (for example, rejecting very large gaps when ramp vehicle trying to join to merging traffic) can be easily identified from the animation of the simulation. For this research, the verification is carried out by reviewing the animation of the simulation output for reasonableness and coding accuracy. Some features of merging operation observed in the real world should also be simulated:

- In most cases merging vehicles can enter into the nearside traffic on the motorway without running out of the acceleration lane or stopping at the end;
- No collision occurs;
- Very large gaps are not rejected when the merging vehicle are looking for gaps in the motorway;
- Traffic build-up in the merging section can be observed, with shockwave propagated backward on the main motorway when there is heavy merging traffic.

Plots of simulation model outputs are also be made to check that the values are realistic. The investigated simulation output variables include speed plots, acceleration/deceleration properties:

- The acceleration phase of the merging vehicle- gradually increasing its speed until it achieves its target speed (motorway traffic speed);

- The two traffic streams (motorway nearside traffic and the merging traffic) tend to reach a similar speed at merging section.

The verification through on-screen animation is examined at the beginning stage so as to eliminate coding/logic errors. Sensitivity testing is also involved for the model verification in this study. "*Sensitivity analysis is a valuable part of model verification*" (Bloomberg and Dale, 2000). Some parameters are systematically varied and the outputs are examined at micro- and macro- levels and the conclusion is used for later model calibration.

For the model calibration, basically, two types of parameters need to be considered: *general parameters* and *specific parameters*. General parameters are the inputs that are independent of the particular simulation location, i.e. they are related to general driving dynamics such as the mechanical limits of acceleration/deceleration and vehicle length distribution. Specific parameters depend on the simulated location and hence their values must be adjusted if location is changed. Examples are the section layout, desired speed of vehicles under free-flow situation, vehicle composition, arrival traffic flow and speed, driver's reaction time, etc.

Some of the general parameters are not actually calibrated; however reliable values can be obtained from literature. Regarding the specific parameters, some can be found from direct observation of selected sites such as the geometry of the survey site. Some can be calibrated from statistical analysis of real-traffic observation of some behaviour during a certain study period, e.g. the desired speeds of HGV and car (explained in section 2.2). Some parameters can not or are difficult to be obtained or extracted directly from field data, e.g. drivers' reaction times. For those parameters, the alternative approach used here is to perform numerical tests (simulation runs) with several parameter settings and to compare the output with the real observations, which include the outputs at both micro (such as individual vehicle's trajectories) and macro levels (aggregated detector measurements). According to the discrepancies, an optimisation approach is applied to achieve the best values of the parameters to replicate the field data. With the best fit parameters identified, the validity of the model will be investigated. If the discrepancy between the simulated data and field data is unacceptable, further calibration and validation will be carried out.

As introduced earlier in Chapter 1, two models are developed for this study: a car-following model and a merging model. It should be noted that both of the two models can be either used separately for single purpose (i.e. the car-following model

to simulate a motorway section without any on/off ramps and, the merging model to simulate a on-ramp merging area), or two models working in tandem to simulate motorway section in connected with a ramp road.

In this study, it firstly illustrates how to calibrate each of the two models for single purpose use. The reason for doing so was to facilitate implementation of the individual model into other study or simulation package. It also illustrates the model calibration method when using the integrated model for simulation (i.e. the combination of the two models). The calibration at this stage is aimed to assist the application of the integrated model into other similar study.

The detailed process and results of the first step calibration will be reported in Chapter 3 with respect to the car-following model in comparison with M25 motorway loop detector data; and in Chapter 4, the merging model will be calibrated with M27 motorway data as reported by Zheng (2002). The process and results of the second step, the calibration and validation of the integrated simulation model will be reported in Chapter 5 through a case study of a merging site of Junction 27, on M8 motorway, Glasgow.

CHAPTER 3
A SIMULATION MODEL FOR MOTORWAY CAR-FOLLOWING BEHAVIOUR

In recent years, the phenomena of flow breakdown, traffic hysteresis and close following of motorway traffic have received greater attention (e.g. Zhang and Kim, 2001; Hounsell et al., 1992; Zhang, 1999). When traffic flow breaks down, it generates a traffic jam together with what is termed a *shockwave*, by analogy with compression waves in fluids (Kerner and Rehborn, 1996; Daganzo et al., 1999). Considerably closer following is found when traffic is near capacity (but before the breakdown) and at high speed, that is, smaller gaps are accepted among passenger cars in non-congested than in congested flow conditions (Postans and Wilson, 1983; Dijker et al., 1998). Traffic hysteresis is a phenomenon characterised by a loop structure seen on the empirical flow-occupancy diagrams, where the capacity of a traffic flow recovering from a flow breakdown does not reach the capacity before the breakdown (Treiterer and Mayers, 1974; Daganzo, 2002).

Car-following models are concerned with the response of the driver-vehicle system to inter-vehicular dynamics in a single stream of traffic. They describe the movements of a following vehicle in response to the actions of the lead vehicle(s). There has been considerable interest in the development of car-following models to better understand the traffic flow concept in general (e.g. Forbes and Simpson, 1968; May, 1990) and the observed phenomena in motorway traffic in particular (Ozaki, 1993).

Generally, the car-following models developed can be divided into three categories: stimulus-based models, safety distance models and action point models (Brackstone and McDonald, 1999). The main idea of a stimulus-based model is that the acceleration of a following vehicle is determined by the driver's reaction to the speed and position differences to the vehicle in front (May, 1990). The General Motors (GM) models are some of the best known stimulus-based models, which have been developed since the late 1950s with one of their latest modifications proposed by Ozaki (1993). The safety distance models are based on the idea that the driver of a following vehicle would adopt such a speed and keep at such a distance that he/she would bring his vehicle to a safe stop should the vehicle in front brake to a sudden stop. Gipps car-following model (1981) is one based on the safety-distance idea. Action point models are based on the idea that a driver's driving behaviour

would vary depends on the traffic state he/she is in: whether he/she is in free driving, approaching to the vehicle in front, following the vehicle in front or braking. The boundary conditions defining the different states are usually expressed as a combination of speed difference and relative distance to the vehicle ahead (e.g. Zhang and Kim, 2001; Leutzbach and Wiedemann, 1986).

However, most of the existing models (such as the GM models) are based on empirical investigations carried out on test tracks with driving speeds considerably lower than those on motorways, hence they may not reflect more general car-following behaviour (Brackstone and McDonald, 1999). Even so, these models are found to fail to reproduce close-following – the prominent phenomena of motorway traffic flow (Wilson, 2001). Some models have attempted to rectify these problems for motorway traffic. For example, Brackstone et al. (2002) calibrated the a*ction points* model for the close-following state based on the four thresholds proposed by Leutzbach and Wiedemann (1986), but the model lacks the continuity with other normal following states, namely the free-flow following state.

Gipps' *safety-distance based* car-following model (1981) is widely adopted in micro-simulation packages (e.g. Barcelo, 1996; Axhausen and Pendyala, 1997; SISTM, 1993). His model has the advantage that all its parameters have realistic physical meanings, which makes it desirable without "*resorting to elaborate calibration procedures*" (Gipps, 1981). A detailed analysis of Gipps' model is presented later in this chapter. Zhang and Kim (2001) have shown the effectiveness of their model in presenting traffic breakdown and hysteresis at macro-level by applying large changes in reaction times among different phases (i.e. acceleration, deceleration and cruising). For comparison, the analysis of the model proposed by Zhang and Kim (2001) is also included here. The close-following model calibrated by Brackstone et al. (2002) is reviewed here for the study of close following situation.

Based on Gipps' model for normal following states and the action point model for the close-following situation, a new car-following model is developed here which attempts to capture the full range of characteristics in motorway traffic, including free-flow following, traffic breakdown, traffic hysteresis, as well as close-following driving behaviour (Wang et al., 2005a). This chapter presents the formulation of the new model and discusses its macroscopic and microscopic properties. A generic validation procedure is developed and the model is validated against the observed travel behaviour on a busy UK motorway, the M25 orbital motorway around London.

3.1. MOTORWAY TRAFFIC FLOW CHARACTERISTICS AND BEHAVIOURS

In this section, the review of motorway traffic flow properties is at both the macroscopic level with regard to speed breakdown and traffic hysteresis, and the microscopic level with regard to gap distribution and shockwave propagation. Several observed characteristics of motorway traffic flow have been found at macro level from flow-density-speed relationships, traffic breakdowns, traffic jams, etc. Recently, the phenomena of speed breakdown and traffic hysteresis (Figure 3-1, 3-2) of motorway traffic flow have caught much attention (e.g. Hounsell et al., 1992; Zhang, 1999; Zhang and Kim, 2001). As for traffic breakdowns, these happen when the traffic volume reaches capacity, such that cramming still more vehicles on to the motorway would lead to slow down everyone sharply, and cause a sudden drop of speed and shockwaves (Holmes, 1994).

Figure 3-1 illustrates the speed breakdown on lane 1 (left most) collected by detector 4806A, located between junction 11 and 12 of motorway M25 (eastbound) on Feb 6th 2001. One apparent speed drop can be found between 7:00 and 7:30: the drop started at 7:00 (57 km/h) and finished at 7:07 (11 km/h). Immediately after the sudden speed breakdown, a flow breakdown[2] started at 7:03 as shown in Figure 3-2: there is a sudden drop of flow immediately after the traffic flow reaches its maximum 2100 veh/h (7:03). After recovery from this flow drop (7:06), the flow does not reach the previous maximum level but follows a lower curve back (7:11). This loop structure found from the flow-occupancy diagram is termed traffic hysteresis. As found from the real traffic observation, the extents of speed drop and traffic hysteresis may vary at different level (Appendix B). However, a sudden drop of speed from speed time plot and a loop structure from flow-occupancy plot can always be identified from real data profiles (Hounsell *et al.*, 1992).

---

2 The occurrence and timing of flow breakdown can be identified with reference to the minute-by-minute speed flow plots (Hounsell et al., 1992).

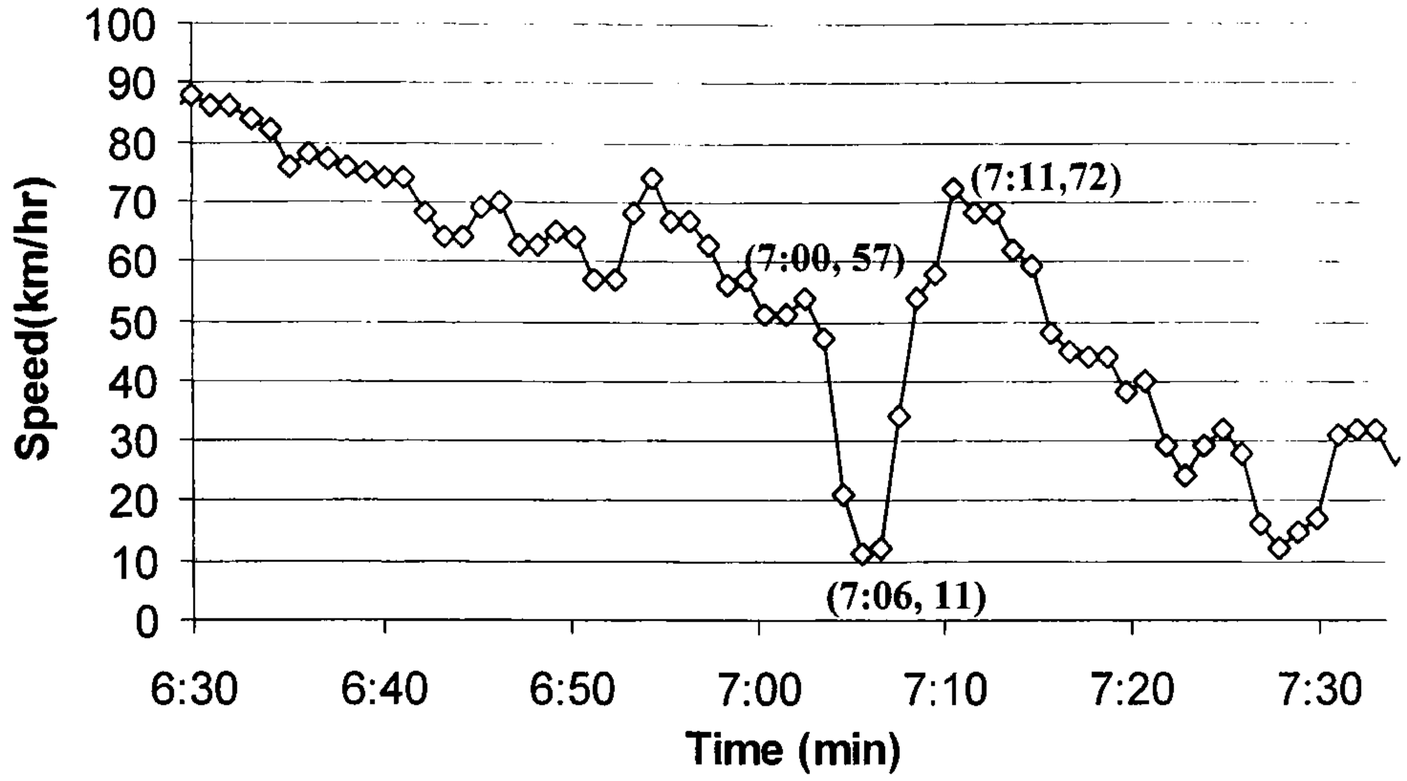


Figure 3-1 Illustration of speed breakdown on lane 1 collected by detector 4806A on M25 between Junction 11 and 12 (eastbound), 6th Feb. 2002.

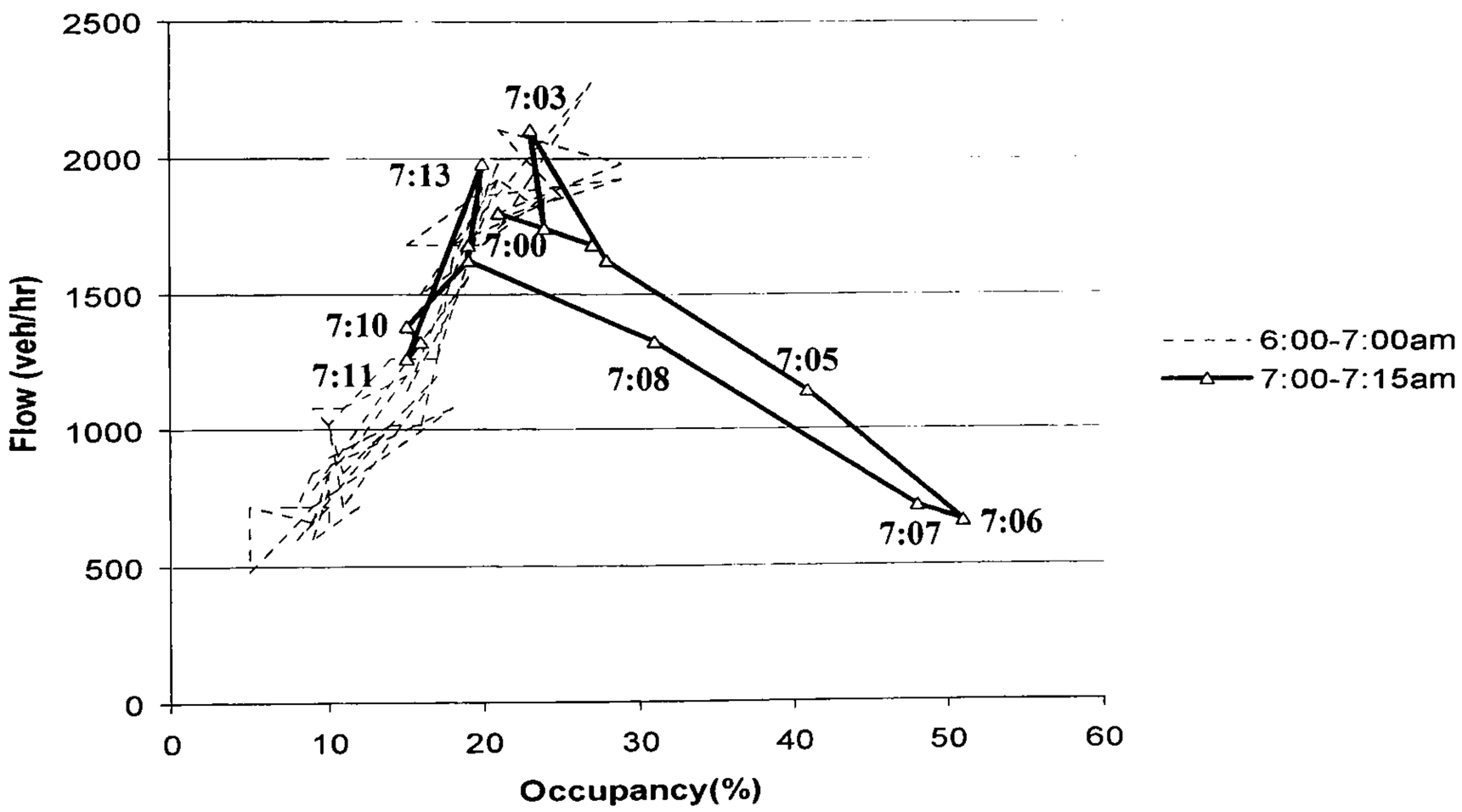


Figure 3-2 Illustration of traffic hysteresis of lane 1 collected by detector 4806A on M25 between Junction 11 and 12 (eastbound), 6th Feb. 2002.

Figure 3-3[3] shows a grey-scale map of the traffic speeds collected from the detectors between Junction 10 and 11 on the M25 motorway (courtesy of TRL-Transport Research Laboratory, UK). The traffic speeds are mapped onto a space-time plane with the x-axis showing the time of the day and y-axis the locations of the detectors. A number of low traffic speeds can be seen propagating backwards in space; the tangent of such propagation measures the shockwave speed. From Figure 3-3, the measured shockwave speeds range from -8 to -18 m/s.

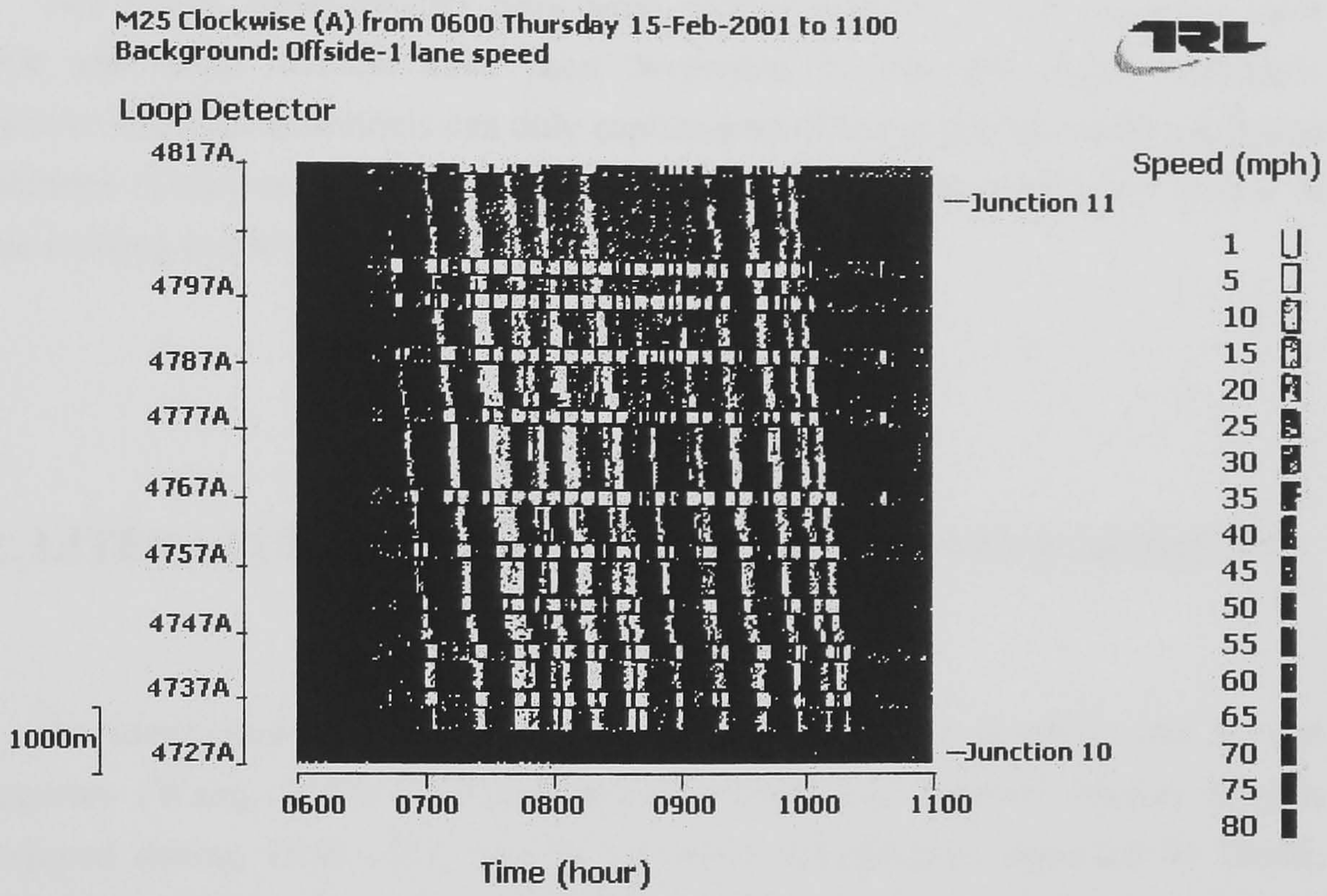


Figure 3-3 A grey-scale map of the traffic speeds collected from the detectors between Junction 10 and 11 on the M25 motorway (courtesy of TRL, UK). (The traffic speeds are mapped onto a space-time plane with the x-axis showing the time of the day and y-axis the locations of the detectors with the detector identification marked. The lower traffic speeds are shown in lighter colours. Not all detectors are shown; between the detector 4727A and 4797A, only alternate detectors are shown at a spacing of 1000m.)

When traffic is near capacity, the flow may be much denser, and many drivers are driving in a following state. Postans and Wilson (1983) reported that some car drivers accept much shorter headways than are necessary to avoid collision- 23% of gaps recorded between leading and tailgating vehicles were less than 0.5 second. There are several explanations for such behaviour. First, the drivers may be able to see several vehicles downstream in the platoon and can take that information into

---

3 This diagram was from a paper copy provided by TRL (produced with MTV program, TRL). It is used here as an example to illustrate M25 shockwave.

account when making their following decisions. Second, the drivers may drive with a higher degree of alertness than under free flow driving conditions, and so have a shorter reaction time. A close following driver may not need to brake very hard to keep a safe distance if none of the vehicles downstream in the platoon (say, the two front leading vehicles) is braking; when one or more of the leading vehicles are braking, the following driver will notice and thus the reaction time should be reduced (Neubert et al., 1999).

The above characteristics have been widely reported in the literature since 1950s and many models have been developed to represent these properties. However, most of the models can only capture part of the properties under particular conditions (Daganzo, 2002). The following section will summarise the review of these existing models.

3.2. LITERATURE REVIEW OF CAR-FOLLOWING MODELS

As mentioned earlier, car-following models can be divided into several categories (Wang, 2003): Stimulus-based models (e.g. General Motors models developed during 1958-1963, one of the latest modification executed by Ozaki, 1993), Safety Distance models (e.g. Gipps, 1981) and Action Point models (e.g. Leutzbach and Wiedemann, 1986; Zhang and Kim, 2001).

Car-following models study the movements of individual vehicles in the following state and also examine the interactions between individual vehicles. Figure 3-4 shows the positions of two vehicles in a platoon on a section of road at the time t, measured from the front bumper of each vehicle. Vehicle *n* is the leading vehicle and *n+1* the following vehicle. The frequently used symbols are given in Table 3-1.

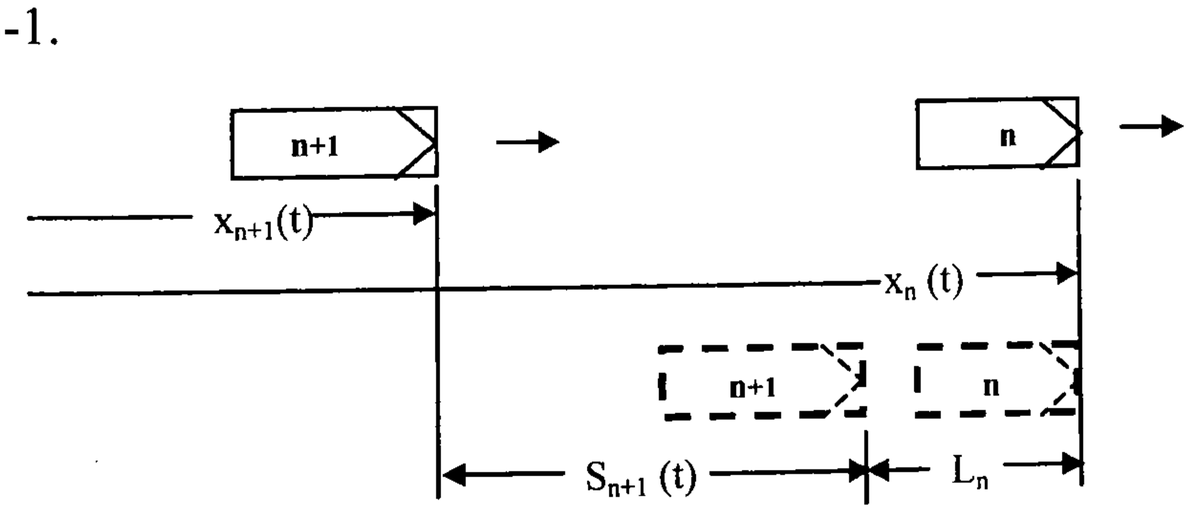


Figure 3-4 Positions of leading and following vehicles on a section of road.

Table 3-1 Frequently used symbols.

$a_n(t)$ (m/s^2)	the instantaneous acceleration at time t of vehicle n
$b_n(t)$ (m/s^2)	the instantaneous deceleration at time t of vehicle n, negative value
$v_n(t)$ (m/s)	the speed of vehicle n at time t
$x_n(t)$ (m)	the position of vehicle n at time t
L_n (m)	the effective size of vehicle n, including the physical length plus a safe margin
$S_n(t)$	the spacing between vehicles n and n+1 at time t, equals to xn+1(t)-xn(t) - Ln
τ (s)	the vehicle's reaction time
a_n^{max} (m/s^2)	the maximum acceleration that vehicle n wishes to undertake
b_n^{max} (m/s^2)	the maximum deceleration that vehicle n wishes to undertake, negative value
b' (m/s^2)	the driver estimated value of the most severe braking that his/her leader wishes to undertake, negative value
V_n (m/s)	the speed at which the driver n wishes to travel

3.2.1. Stimulus-Based Models

The main idea for this type of model is the assumption that the response (usually refers to acceleration) of any vehicle is determined by the driver's sensitivity and stimuli (e.g. speed difference and relative distance), i.e. Response = Function (sensitivity, stimuli).

The GM Models perhaps are the most well known car-following models and date from the late fifties and early sixties (Chandler et al., 1958). The main idea is that the acceleration of a vehicle is determined by the driver's reaction to speed differences and position differences (May, 1990). The general formulation is

$$a_{n+1}(t+\tau) = \frac{\alpha_{l,m}[v_{n+1}(t+\tau)]^m}{[x_n(t)-x_{n+1}(t)]^l}[v_n(t)-v_{n+1}(t)] \tag{3-1}$$

where m, l, α are the constants to be determined.

The first prototype GM model was put forward in the late 50s by Chandler, Herman and Montroll (1958) at the General Motors research labs in Detroit. Rapid developments and modifications to this type of model subsequently followed. This type of model works well when parameters are carefully chosen (Miyahara, 1994). It can represent disturbance propagation[4] and also provide valuable analysis on traffic instability (May, 1990). A great deal of work has been performed on the calibration and validation of the GM models. Table 3-2 is a list of optimal parameter combinations for the GM model reviewed by Brackstone and McDonald (1999). However, it is now being used less frequently, mainly because of the large number of contradictory findings as to the correct values of the parameters (Brackstone and McDonald, 1999).

Table 3-2 Summary of optimal parameter combinations of m and l for the 'GM' model (Brackstone and McDonald, 1999).

Source	*m*	*l*
Chandler et al. (1958)	0	0
Gazis, Herman and Potts (1959)	0	1
Herman and Potts (1959)	0	1
Helly (1959)	1	1
Gazis et al. (1961)	0-2	1-2
May and Keller (1967)	0.8	2.8
Heyes and Ashworth (1972)	-0.8	1.2
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
Ceder and May(1976) Single Regime	0.6	2.4
Ceder and May (1976) (uncgd/cgd)	0/0	3/0-1
Aron (1988) (dcn/ss/acn)	2.5/2.7/2.5	0.7/0.3/0.1
Ozaki (1993) (dcn/acn)	0.9/-0.2	1/0.2

4 Disturbance propagation is that for a group of vehicles in the car-following state, when some disturbance is introduced into the group, e.g. sudden deceleration of the leading vehicle, it leads to the decelerations of the rest of the group.

Note: dcn/acn: deceleration/acceleration; brk/no brk: deceleration with and without the use of the brakes; uncgd/cgd: uncongested/congested; ss: steady state

3.2.2. Safety Distance Models

For this category of car-following models, one of the common assumptions in modelling vehicle following is that the drivers maintain the same driving behaviour regardless of the traffic conditions they are in. The Gipps model assumes that the drivers apply the same reaction time and acceleration/deceleration throughout the different traffic states. The main idea is that the driver of the following vehicle can select a safe speed to ensure that he/she can bring his vehicle to a safe stop, if the vehicle ahead comes to a sudden stop. Gipps (1981) provides a detailed description of the development and use of a behavioural car-following model that mimics observed driver behaviour. Gipps constructs a safety distance car-following model for the response of the following vehicle based on the assumption that each driver sets limits to his/her desired braking and acceleration rates. Constant reaction time (2/3 second) is used over all traffic states (congested and free-flow conditions) and drivers. The model's algorithm will be given in detail in section 3.3.3.

This type of model has an advantage over the GM models in that all its parameters have identifiable physical meanings that represent characteristics of the drivers or vehicles (Sultan, 1998). It is shown to have the advantage of representing very well the individual vehicle's speed control abilities (i.e. with respect to the mechanical capabilities) and shockwave propagation (Brackstone and McDonald, 1999; Wilson, 2001). However, the use of a constant reaction time for all traffic states is not reasonable according to some empirical findings such as Fambro et al. (1998). It is found to be unable to reproduce speed drops or traffic hysteresis as examined later through numerical simulations in section 3.3.3. The modelled headway between vehicles is also found to be longer than empirical data (Wilson, 2001).

3.2.3. Action Point Models

The main idea of this type of model is that a driver can be in one of four driving modes: free driving, approaching, following, braking. The driver switches from one mode to another as soon as he/she reaches a certain threshold usually expressed as a combination of speed difference and relative distance between interacted vehicles.

One of the first constructions of these models was given by Michaels and Gordon (1963), who proposed that the drivers would initially be able to tell they are approaching a vehicle in front, primarily due to changes in the visual angle subtended by the vehicle ahead. The equation is given below:

$$\Delta\theta(t) = \frac{K(v_n(t) - v_{n+1}(t))}{(x_n(t) - x_{n+1}(t))^2} \qquad (3\text{-}2)$$

where, K is the width of the front vehicle (m), $\Delta\theta$ (t) is the changing rate of visual angle (rad/sec).

A series of perception-based experiments and empirical studies have been conducted in order to investigate the perception thresholds. Leutzbach and Wiedemann (1986) proposed and applied this idea in a simulation model *MISSION*. Recently, Brackstone et al. (2002) calibrated the thresholds for close-following situation. Another new car-following model has been proposed by Zhang and Kim (2001), which is aimed at reproducing traffic hysteresis and capacity drop. This model is based on a new interpretation of Pipes Model (1953)[5]: the follower adopts a speed $v_n(t)$ given by the vehicle spacing divided by reaction time:

$$v_n(t) = S_n(t) / \tau_n(t) \qquad (3\text{-}3)$$

where $\tau_n(t)$ denotes the follower- n^{th} vehicle's reaction time at time t.

---

5 Pipes car-following model is a linear model that describes traffic behaviour if drivers observe the driving rules suggested in California Motor Vehicle Code- leave one car length in front every ten miles per hour of speed increment and has the following form: Xn-1=Xn+ τ vn+L. Zhang and Kim's interpretation of Pipe's model: Let Sn=(Xn-1-Xn-L) Then Sn=τ vn

Zhang and Kim (2001) applied the idea that drivers can have different reaction times for different phases: acceleration, deceleration, and cruising. "Varying Reaction Time" is also advocated by many other researchers. "*The use of varying reaction time is more realistic than using a constant value for all densities and more reasonable than using a constant value for all drivers*" (Benekohal and Treiterer, 1988; Toledo, 2003). Zhang and Kim have shown the effectiveness of their model in representing traffic breakdown and hysteresis at macroscopic level by applying large changes in reaction times among different phases: 1.0 second for cruising, 1.2 seconds for deceleration and 1.8 seconds for accelerating.

The action points models can reproduce realistic traffic flow under different real-world conditions, e.g. traffic breakdown. But according to Brackstone and McDonald (1999), "*it is difficult to come to a conclusion as to the validity of these models, as calibration of thresholds and individual elements has been less successful, although the entire system would seem to simulate behaviour acceptably.*" It is found that such sudden change in reaction times for different phases causes discontinuities of the movements of vehicles and unrealistic acceleration and decelerations considering the real world vehicle's mechanical capability, e.g. Zhang and Kim's model, which will be further examined in section 3.4.

In section 3.3, 3.4 and 3.5, the simulation results of Gipps' car-following model, Zhang and Kim's car-following model and Brackstone et al. close-following model will be discussed. The performance of the models will be analysed and compared with real observations.

3.3. REPRODUCTION OF GIPPS' CAR-FOLLOWING MODEL

To reproduce Gipps' car-following model, an experimental test design is firstly proposed in section 3.3.1 to simulate gradually increasing and decreasing flows on a closed ring road. This test design will also be applied to the reproduction of Zhang and Kim's car-following model (section 3.4), close-following model (section 3.5) and the new proposed car-following model (section 3.6) (Appendix C).

3.3.1. Experimental Test design

The simulation is conducted on a single lane, flat and one-way circular road (without consideration of the curvature of the road) similar to that proposed by Zhang and Kim (2001) (Figure 3-5). This experimental test used here is aimed to examine whether the model being tested is able to realistically capture the speed drop, traffic hysteresis and shockwave propagation as well as close-following behaviours. Such circular roads are widely used in numerical experiments of car-following models (Holland, 1998; Wilson, 2001; Mason and Woods, 1997). It should be mentioned that the system performance can be different when open boundary conditions (such as a stretch of road) are applied, compared to the closed periodic boundary conditions (i.e. a ring road). According to Gibson and McCartney (2005), introducing a disturbance into behaviours of a vehicle in the ring road might have a much greater impact on the behaviour of the following vehicles than if a similar disturbance is introduced into the motion of the lead vehicle in the open road. However, ring roads are believed to have wider applicability to real traffic situations such as traffic congestion and disturbance propagation through the importing of boundary conditions (Wilson, 2001; Mason and Woods, 1997).

The length of the ring road is 1080m. The average effective vehicle length is assumed to be 6.5 metres (Gipps, 1981) hence 166 vehicles can be jammed on the ring road. In the simulation, a maximum of 85 vehicles are entered on to the ring road at any one time according to the test design by Zhang and Kim (2001). The simulation is updated every 0.2 second. Vehicles' desired speed (V_n) is normally distributed with free-flow speed (v_f) as the mean value and σ^2 as the variance:

$$V_n \sim N(v_f, \sigma^2) \tag{3-4}$$

The simulation (screenshots given in Appendix C) starts with an empty ring road. It then runs through three demand stages. The *increased demand* stage, which lasts for 1700 seconds, is when the vehicles are entered onto the ring. A maximum 85 vehicles are gradually entered into the ring road, which enables the traffic to build-up from the free-flow to congested states. During the *constant demand* stage, which lasts for 200 seconds in the simulation, no vehicles are entered or exit from the ring road. At this stage, the traffic properties under unstable congested (high density, low speed) situation can be examined. Finally, the *decreased demand* stage, which lasts for 1700 seconds, is the stage where vehicles are gradually removed

from the ring road, which enables the traffic to recover from congestion to a free-flow state. During the increased and decreased demand stages, vehicles enter and exit the ring road one by one every 20 seconds. A new vehicle always joins the tail of the platoon, while an exiting vehicle can exit from any location when it finishes running for a period of 1900 seconds. This enables the test track to experience three different traffic flow stages: traffic build-up from free-flow to congestion, unstable congested traffic, and traffic recovery from congestion to free-flow state.

Three detectors are located along the ring road with centre positions of 270 m, 540 m, and 810 m to collect speeds, occupancies and flows of crossing vehicles. Data collection from simulated detectors is explained earlier in Chapter 2. The test results are analysed and evaluated both at macro- and micro-level.

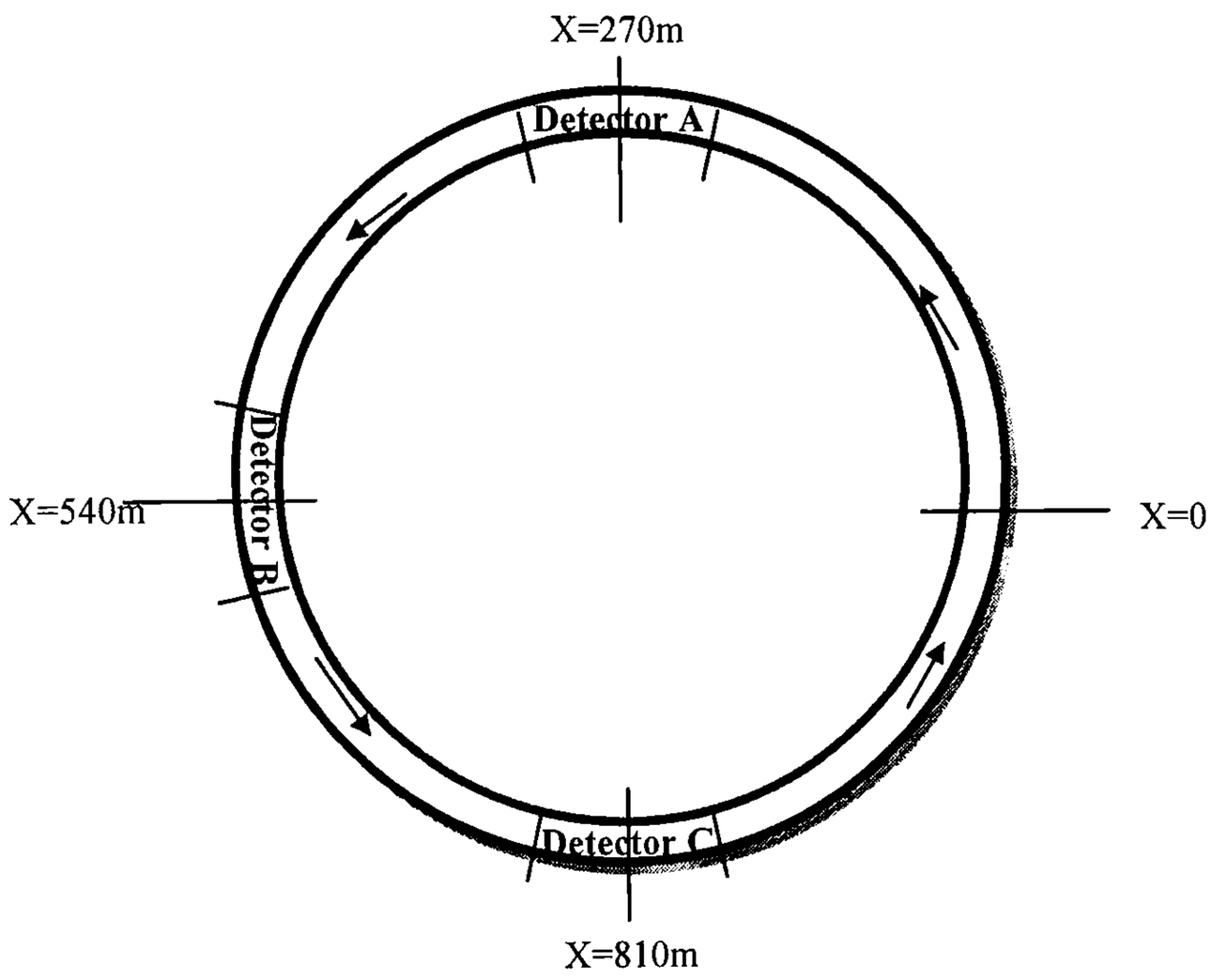


Figure 3-5 Test configuration in the simulation.

3.3.2. Model Description

Gipps (1981) constructs a safety distance car-following model for the response of the following vehicle based on the assumption that each driver sets limits to his desired braking and acceleration rates and constant reaction time (2/3 second) is used over all traffic states (congested and free-flow conditions) and drivers. The model's algorithm is as follows,

$$v_n^a(t+\tau) = v_n(t) + 2.5a_n{}^{\max}\tau[1 - \frac{v_n(t)}{V_n}]\sqrt{0.025 + \frac{v_n(t)}{V_n}} \quad (3\text{-}5)$$

$$v_n^b(t+\tau) = b_n{}^{\max}\tau + \sqrt{(b_n{}^{\max})^2\tau^2 - b_n{}^{\max}\{2[x_{n-1}(t) - L_{n-1} - x_n(t)] - v_n(t)\tau - \frac{v_{n-1}{}^2(t)}{b'}\}} \quad (3\text{-}6)$$

$$v_n(t+\tau) = \min\{ v_n^a(t+\tau), v_n^b(t+\tau)\} \quad (3\text{-}7)$$

where, v_n^a represents the acceleration speeds of vehicle n under free-flow situation and v_n^b represents the deceleration speeds of vehicle n under collision-avoidance situation.

According to Gipps (1981), the parameters of vehicle length and desired speed are described in Table 3-3, and they are also applied to the rest of this chapter. Other parameters of Gipps' model are summarised in Table 3-4.

Table 3-3 Vehicle length and desired speed (Gipps, 1981).

Parameter	Mean	Standard Deviation
L_n (m)	6.5	0.3
V_n (m/s)	20	3.2

Table 3-4 Other parameters used in Gipps's model simulation test (Gipps, 1981)

Parameter	Value
$a_n{}^{max}$ (m/s^2)	Sampled from Normal distribution, N(1.7,0.3^2)
$b_n{}^{max}$ (m/s^2)	Equated to $-2.0a_n$
τ (s)	0.667 ($\approx 2/3$)
b' (m/s^2)	Minimum of –3.0 and (b_n-3.0)/2

3.3.3. Simulation of Gipps' car-following Models

3.3.3.1. Analysis of simulation outputs

The simulation results at the macro level are first examined. Figures 3-6 (a) and (b) show the speed, flow and occupancy profiles collected by detector B (located alongside the test ring road, centred at 270m, refer to section 3.3.1). Figure 3-6(a) shows that in Gipps' model speed drops smoothly when traffic builds up (i.e. *the process of increased demand* by gradually adding vehicles into the ring road), and rises smoothly during traffic recovery (i.e. *the process of decreased demand* by gradually releasing vehicles from the ring road). Figure 3-6(b) shows the flow-occupancy diagram simulated by Gipps' model. No loop structure can be identified from the flow-occupancy diagram, that is, it fails to represent traffic breakdown and traffic hysteresis as discussed earlier in section 3.1.

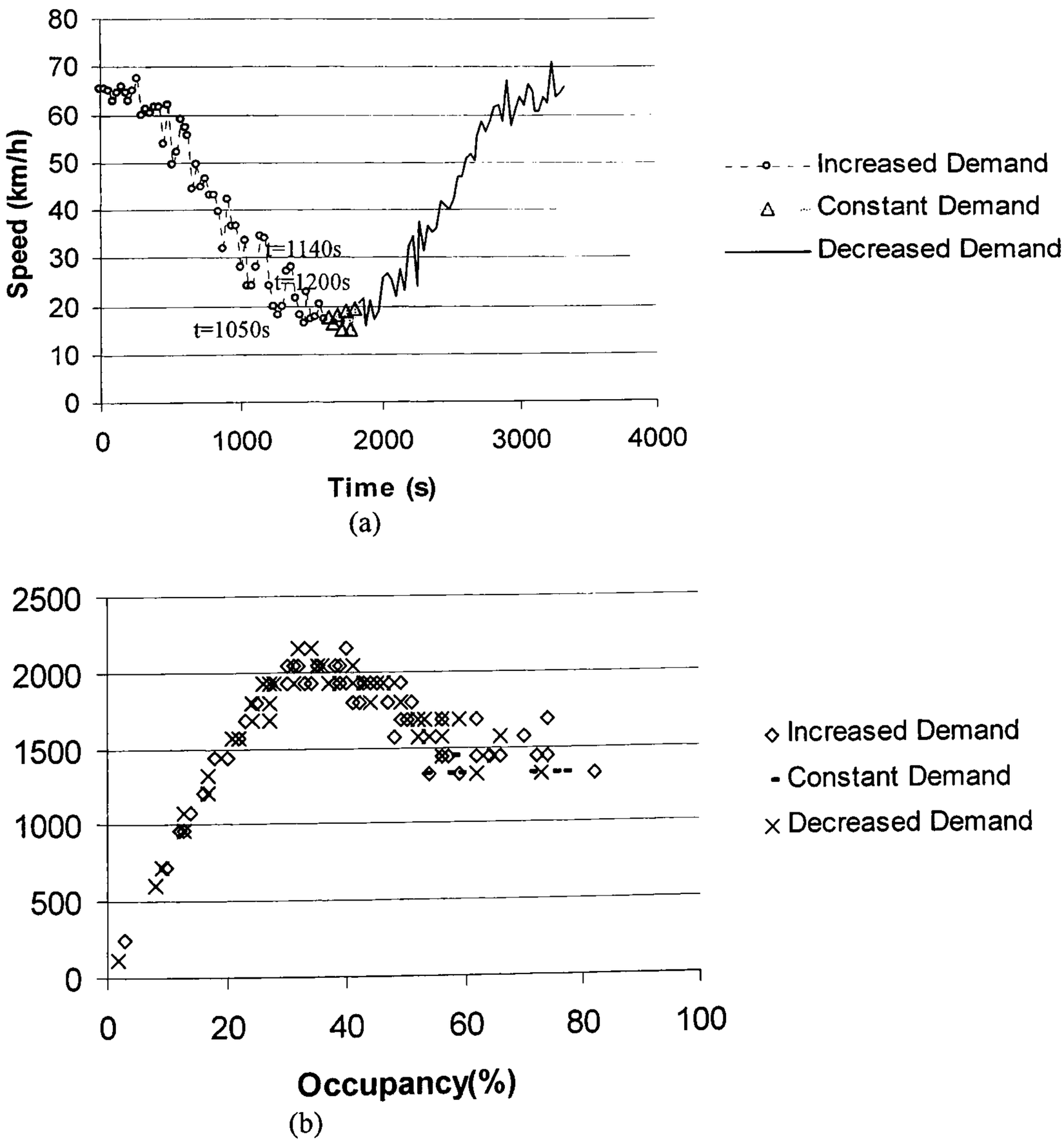


Figure 3-6 Speed-time (a) flow-occupancy and (b) profiles simulated by Gipps' Model.

The simulated traffic is examined during the simulation time 1050~1200 seconds where traffic started to get congested and unstable according to Figure 3-6(a), i.e. 1150~1140s, the traffic speeds increased and, 1140~1200s the traffic speeds dropped. By examining the slope of vehicle trajectories when passing the area where the corresponding detector B was located (i.e. 270m given in section 3.3.1), it shows a similar trend of changes to Figure 3-6(a), i.e. generally, during the period of 1050~1140s the slopes of vehicle trajectories increased slightly and, during 1140~1200s the slopes of vehicle trajectories dropped. The shockwave propagation is examined from the plot of individual vehicle trajectories shown in Figure 3-7. Figure 3-7 shows that Some simulated shockwaves shown in Figure 3-7 can be easily identified with a reduction of traffic flow and velocity, where the trajectories are less condensed denoted as (1), (2) and (3).

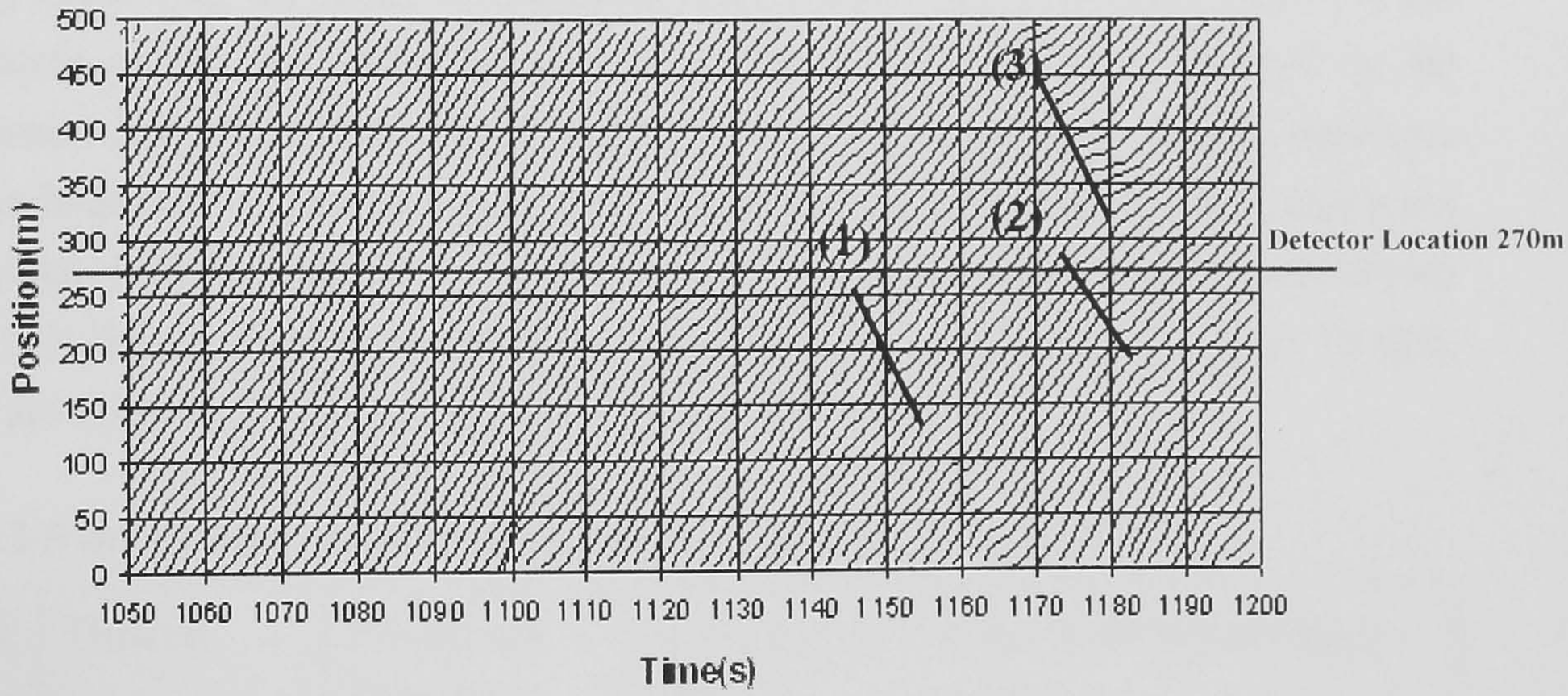


Figure 3-7 Plots of individual vehicle trajectories simulated by Gipps' Model.

The speed of shockwave v_{shock} can be measured from the position of the wave front as it propagates over time from the trajectory plots. Alternatively it can be approximately estimated from the flow and density difference collected from the detectors as:

$$v_{shock} = \Delta q / \Delta \rho \tag{3-8}$$

where, Δq is the change in flow and $\Delta \rho$ is the change in density.

According to Lighthill and Whitham (1955), if the shockwaves involve a reduction of the traffic flow and velocity, they are backward propagated shockwaves. May (1990) gives a further description of this category of shockwaves, which is categorised as backward forming shockwave: the shockwave is propagating upstream in the opposite direction of the moving traffic and results in the increase of the congestion. Thus the simulated shockwaves (Figure 3-7), are backward propagated as the shockwaves move upstream over time and involved a reduction of flow and velocity.

From the experimental test design, at the section from 0 to 500 m, one of the detectors- detector A is located with the centre position of 270 m (Figure 3-5). Thus, the shockwave occurred during 1170 to 1180 s, position 200 m to 300 m (the lower bold line to the right in Figure 3-7), i.e. shockwave (2) can be calculated from eq. (3-8) detector measurements. The other two shockwave speeds can be worked out by directly measuring the slope of the shockwaves from Figure 3-7. In Table 3-5, the shockwave speed calculated from above-mentioned equation is compared to the shockwave slope measured from Figure 3-7. The space time plot of M25 detectors data (as illustrated earlier in Figure 3-3), shows that the shockwaves were backward propagated with the measured shockwave speeds ranging from 8 to 18 m/s. From Table 3-5, the shockwave speeds simulated by Gipps vary from 10 m/s to 14 m/s, which are comparable to the real observations from M25.

Table 3-5 Shockwave speed of the Gipps' Model shown in Figure 3-7.

Shock waves	Time(s)	*Flow from Detector* q (veh/s)	*Density from Detector* ρ (veh/m)	*Shockwave speed* v_{shock} (m/s)	
				Calculated	Measured
(1)	1140~1160	-	-	-	-15±2
(2)	1170	0.60	5.1×10^{-2}	-9.88	-12±2
	1180	0.50	6.1×10^{-2}		
(3)	1170~1180	-	-	-	-15±2
Note: Negative speed denotes the shockwave is backward propagated.					

The modelled gap distribution is examined against those observed. Due to the apparent difficulty in directly calibrating the car-following model, the time gap distribution (Brackstone et al. (2002)) is used here as an indicator of car-following

model's performance. The simulated gap is collected from the time gap among the individual vehicles; but those below 5 seconds were collected. This is because for vehicles in a car-following situation, only those with gaps less than 5s are considered as "following" according to May (1990). The results are compared to the results simulated by Gipps' model and the real data by Brackstone et al. (2002), which were collected by using an instrumented vehicle on M27 motorway, UK. As shown in Figure 3-8, it is found that Gipps' model is under-predicting the distribution of smaller gaps (i.e. gaps < 1s or less) compared to the observed data. It is possible that this model could improve the gap distribution performance if better calibrated. However, as the models do not include direct simulation of close-following situation, the capability of representing the smaller gaps distribution (e.g. less than 0.8 second) might be doubtful.

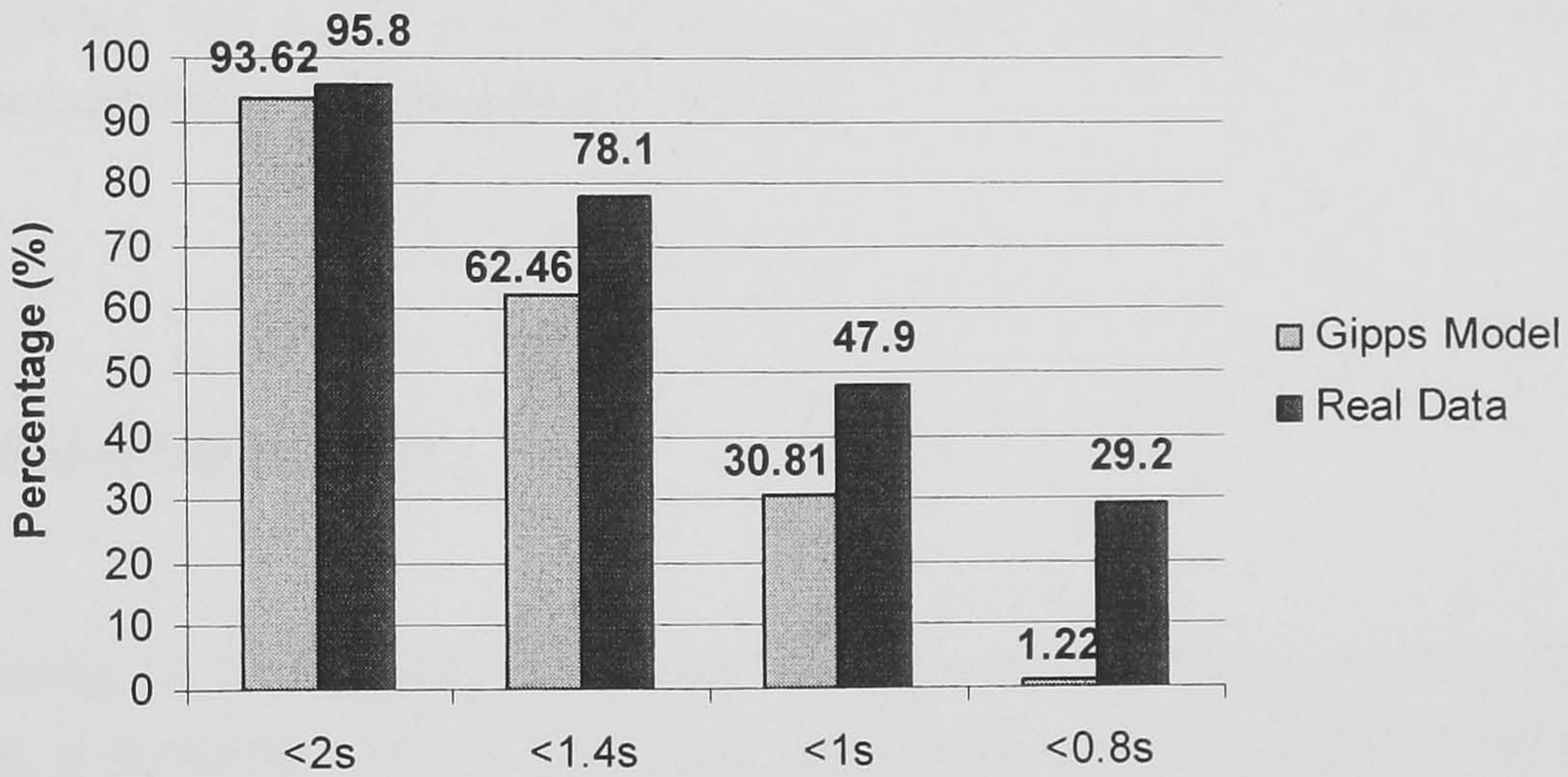


Figure 3-8 Gap distribution comparison between Gipps' model and real data.

3.3.3.2. Summary of the model performance

From the simulation of Gipps' car-following model, the results are examined both at macro- and micro- levels. It is found that the Gipps' model can not reproduce traffic breakdown and hysteresis. However, the reproduced shockwave is backward propagated with a realistic speed. The gaps among individual vehicles simulated by Gipps' model are comparatively longer than those found in real traffic, Gipps' model is under-predicting the smaller gaps.

3.4. REPRODUCTION OF ZHANG AND KIM'S CAR-FOLLOWING MODEL

Gipps model assumes that drivers have the same reaction time and acceleration/deceleration throughout the different traffic states: non-congested and congested traffic. In section 3.3, it has been shown that Gipps model can not represent speed breakdown or traffic hysteresis. As mentioned in section 3.2, Zhang and Kim's car-following Model is based on the new interpretation of Pipes model (1953). Rather than using a constant reaction time as in Gipps' model, Zhang and Kim proposed that driver's reaction time $\tau_n(t)$ varies according to traffic phases, namely acceleration, deceleration and cruising.

3.4.1. Model Description

Among the existing action point car-following models, Zhang and Kim (2001) proposed a powerful one in that it can capture the two prominent features of multiphase vehicular traffic flow/speed drop and traffic hysteresis which most other models failed to reproduce. As given earlier in eq. (3-3), vehicle n adopts a speed $v_n(t)$ with the spacing $S_n(t)$ divided by the reaction time $\tau_n(t)$. The reaction time $\tau_n(t)$ varies according to both spacing and traffic phases $P_n(t)$, namely acceleration ($P_n(t)$=A), deceleration ($P_n(t)$=D) and cruising ($P_n(t)$=C).

The function of reaction time and the logic of the model are shown schematically in Figure 3-9 and Figure 3-10. In Figure 3-9, the lowest line (τ_1) is the deceleration reaction time at free-flow situation with S_1 as the closing threshold[6] for spacing, i.e. at such threshold, the vehicle would shift its traffic phase $P_n(t)$ from free-flow situation to deceleration situation so to avoid collision; the middle line (τ_2)

---

6 The closing process denotes the situation where the following vehicle is driving faster than its preceding vehicle and the gap between the two vehicles is decreasing; the closing threshold is the distance value at which the vehicle would decelerate so to avoid collision with its proceeding vehicle.

is the deceleration reaction time with S_2 as the closing threshold at non-free flow traffic situation; the upper line (τ_3) is the acceleration reaction time with S_3 as the opening threshold at non-free flow traffic situation; the slanted line is the cruise reaction time as a function of spacing and speed. Figure 3-10 illustrates the logic of this model when applied into simulation. With vehicle desired speed and length given in Table 3-3, other specified parameters in Zhang and Kim's model are listed in Table 3-6.

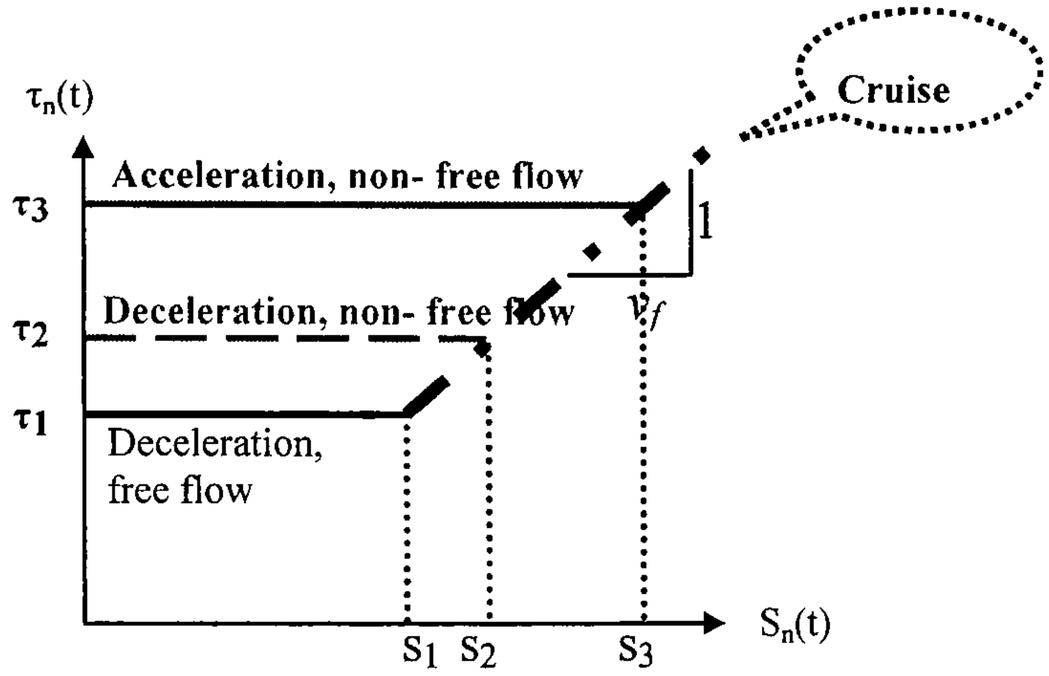


Figure 3-9 Reaction time and spacing relationship proposed by Zhang and Kim (2001).

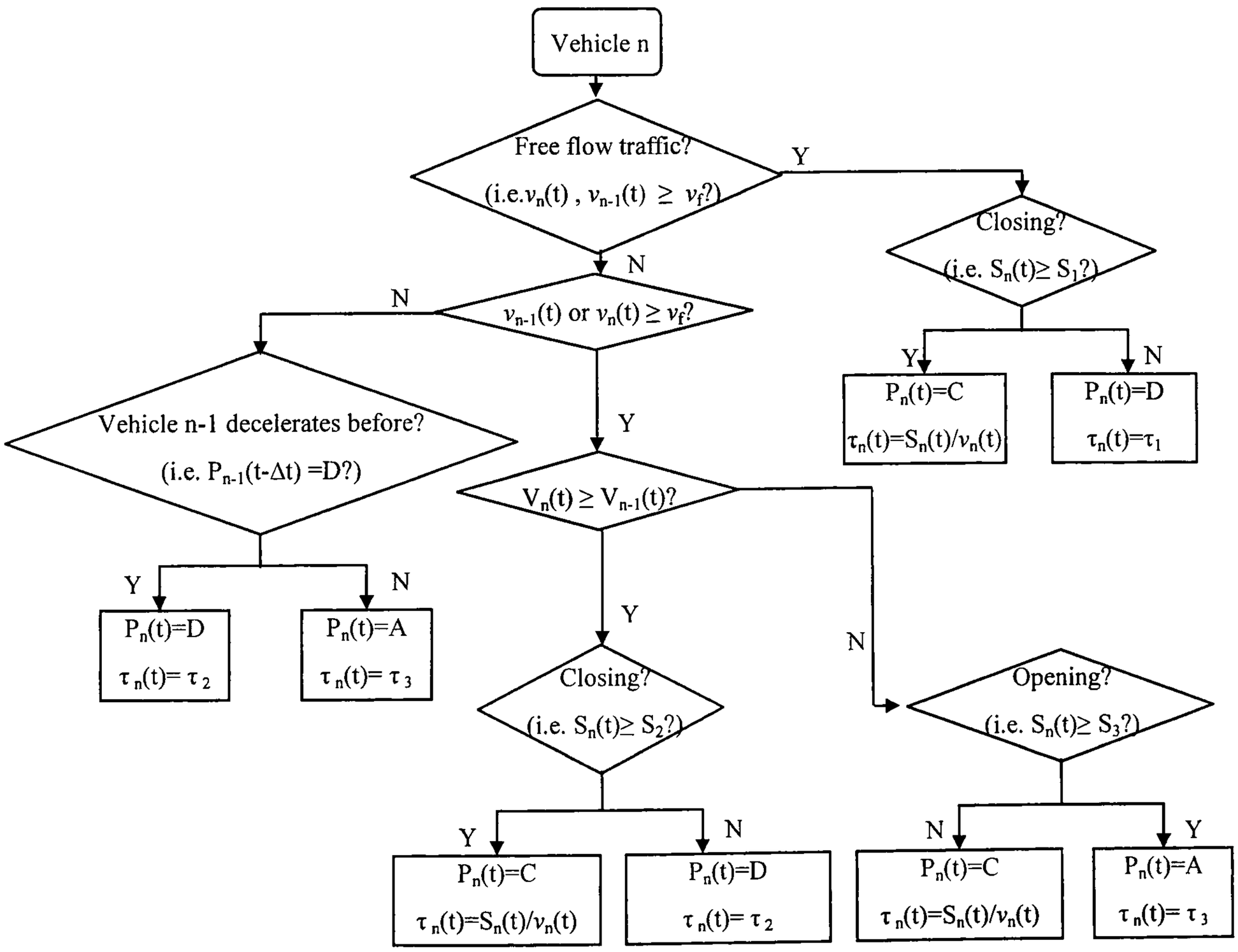


Figure 3-10 Flowchart of Zhang and Kim's car-following model.

Table 3-6 Parameters in Zhang and Kim's model.

Parameter	Denotation	Value (s)	Related Spacing Threshold (m)
τ_1	Deceleration reaction time for free flow traffic	1.0	Closing: S_1=30
τ_2	Deceleration reaction time for non-free flow traffic	1.2	Closing : S_2=36
τ_3	Acceleration reaction time for non-free flow traffic	1.8	Opening: S_3=54

3.4.2. Simulation of Zhang and Kim's car-following Model

3.4.2.1. Analysis of simulation outputs

The simulation results are examined at macro level. Figure 3-11 shows that in Zhang and Kim's model speed drops suddenly during traffic build up: at time 810s the speed is 61km/h and after two minutes the speed is 31km/h, i.e. in 2 minutes speed drops by 30km/h. As earlier illustrated in Figure 3-1 (Section 3.1), a speed drop in the real traffic is found such that within 3 minutes, speed dropped by 30km/h.

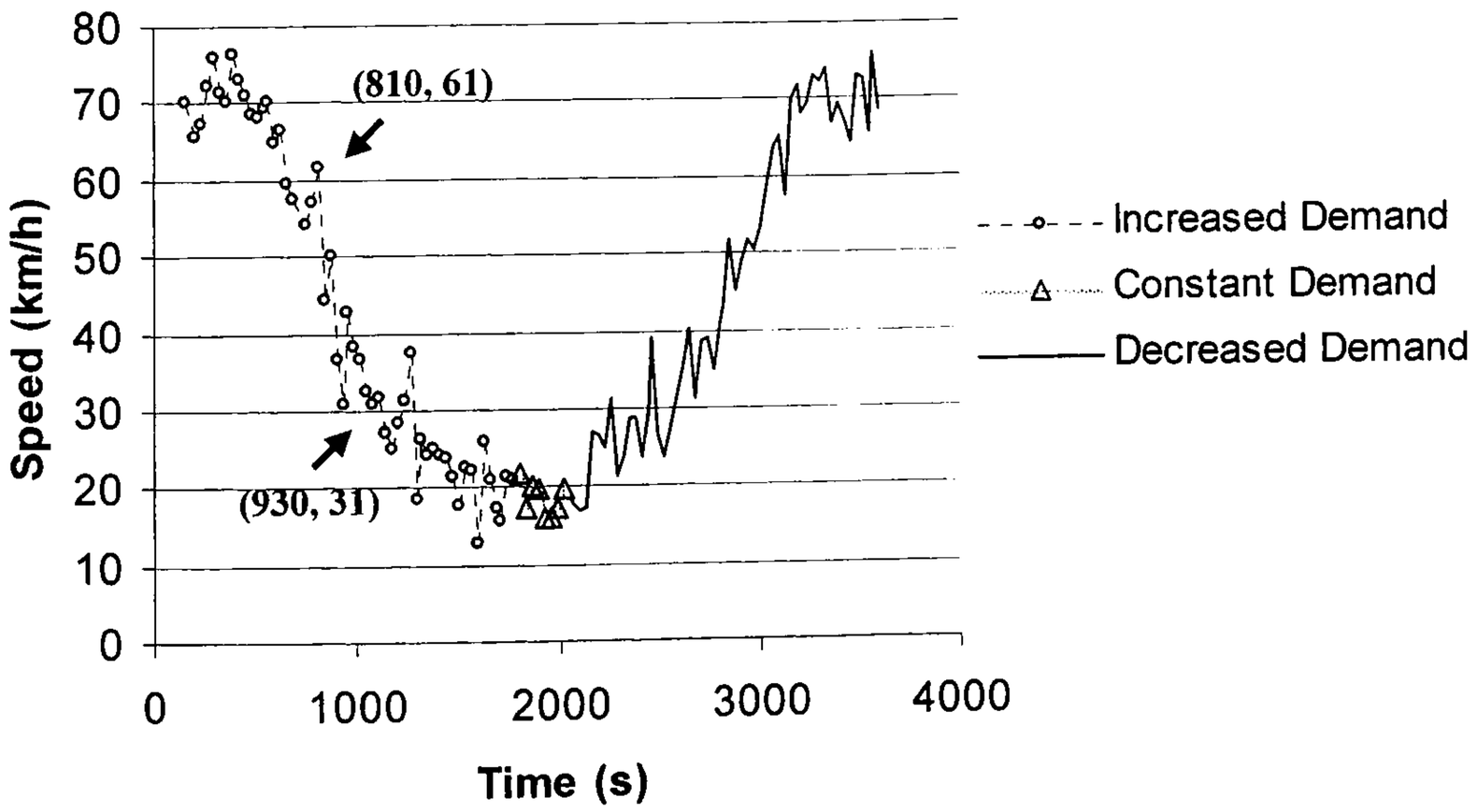


Figure 3-11 Speed-time diagram simulated by Zhang and Kim's Model.

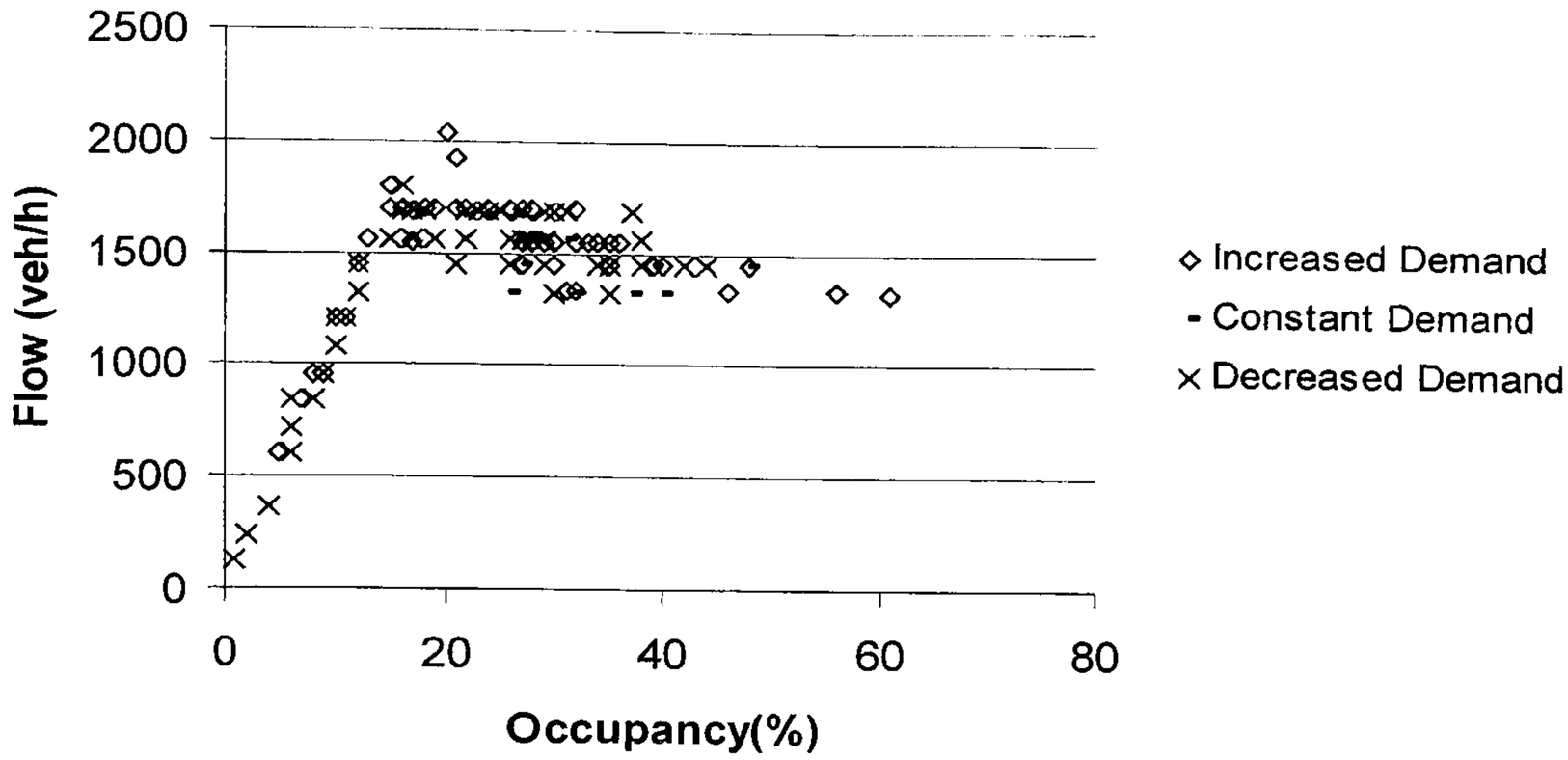


Figure 3-12 Flow-occupancy diagram simulated by Zhang and Kim's Model.

Thus, Zhang and Kim's model can reasonably represent speed breakdown. Figure 3-12 shows that traffic hysteresis can be reproduced by Zhang and Kim's model in the flow-occupancy diagram. A loop structure can be identified, that is, during traffic build up, the flow reaches its maximum, whereas during the traffic recovery, the maximum flow can never be reached again.

The shockwave propagation is examined from the plot of individual vehicle trajectories. As shown in Figure 3-13, the traffic is examined during the simulation time 800~1000 seconds (traffic speed breakdown according to Figure 3-11). It shows that the vehicle trajectories simulated by Zhang and Kim are not very realistic esp. in the circled areas: with discontinuities of the movements of vehicles and sudden changes of the individual speeds (the slopes of the trajectories) , i.e. the speeds change abruptly from quite a high level (approx. 50 km/h) to almost 0 in less than 2 seconds. This is quite contradictory to the car's acceleration/deceleration ability in the real world (mechanical capability of braking is 4.9 m/s^2 according to ITE, 1999). It is also found that although some disturbances can be found in Figure 3-13, only one backward propagated shockwave (bold slanted line) can be identified between position 50 and 150 m.

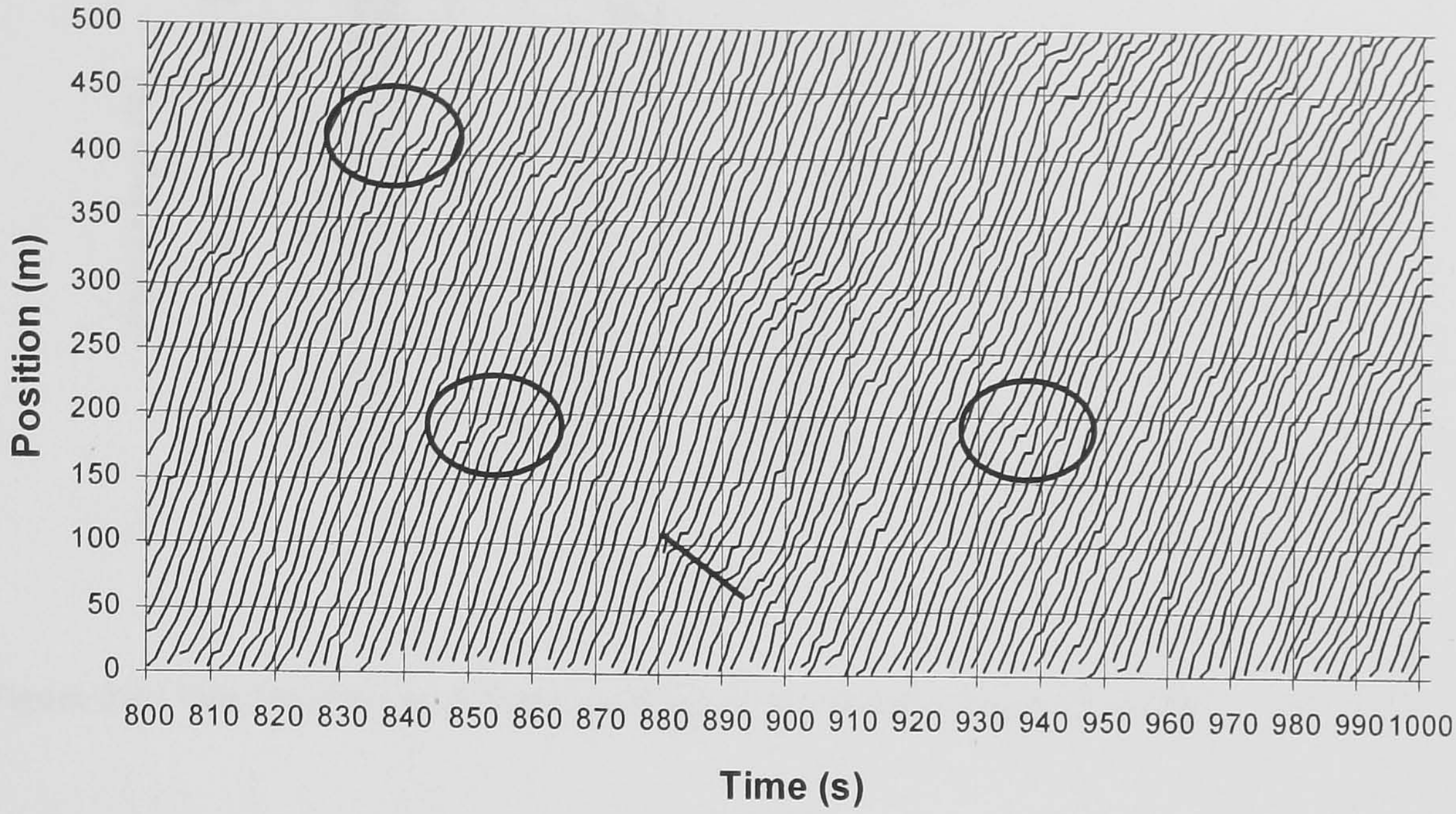


Figure 3-13 Plots of individual vehicle trajectories simulated by Zhang and Kim Model.

As there is no detector located between 50~100 m on the test road, the shockwave speed can only be measured directly from the tangent of such shockwave propagation from Figure 3-13 (Table 3-7). In agreement with Lighthill and Whitham (1955) and May (1990), the shockwave is backward propagated. The shockwave speed simulated by Zhang and Kim's model is 4.7m/s, which is less than those simulated by Gipps (12~15 m/s) and real observation from M25 motorway (8~18 m/s).

Table 3-7 Shockwave speed of the Zhang and Kim's model shown in Figure 3-13.

Time(s)	*Flow from Detector* q (veh/s)	*Density from Detector* ρ (veh/m)	*Shockwave speed* v_{shock} (m/s)	
			Calculated	Measured
880~890	-	-	-	-4.7

Gap distribution is also examined by collecting the time gap among the individual vehicles (only collected those below 5 seconds) in the simulation. The results are compared to the results simulated by Gipps and the real data mentioned in section 3.3. As shown in Figure 3-14, although Zhang and Kim's model shows a higher proportion of the close-following situation (gap <0.8s) than Gipps' model, it is still far below the real data.

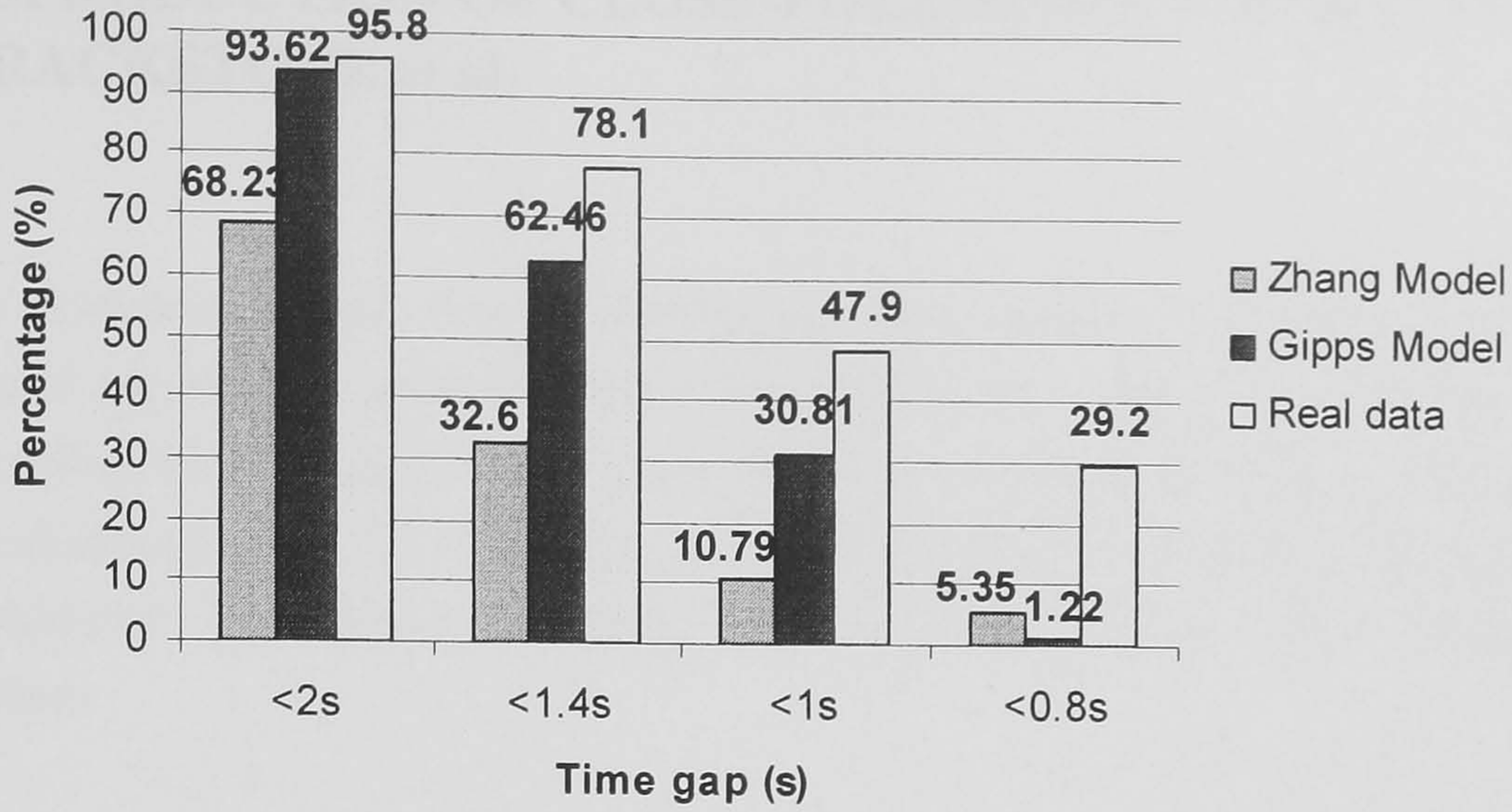


Figure 3-14 Gap distribution of Zhang and Kim model, Gipps' model and real data.

3.4.2.2. Summary of the model performance

From the simulation of Zhang and Kim's car-following model, the results are examined both at macro- and micro- levels. It is found that although Zhang and Kim's model can reproduce the traffic breakdown and hysteresis, it can not reproduce the shockwave speed as realistically as Gipps' model does. It is also found that the sudden change in reaction times for different phases causes discontinuities of the movements of vehicles and unrealistic acceleration and decelerations compared to the real world vehicle's mechanical capability. Compared to Gipps model, Zhang and Kim's model has better performance in representing real world smaller gaps distribution (e.g. less than 0.8s), but it is also under-predicting the smaller gaps.

3.5. REPRODUCTION OF CLOSE-FOLLOWING MODEL BY BRACKSTONE et al.

As discussed earlier, close-following has been widely observed in practical driving and reported in some literature (Hounsell et al., 1992). Brackstone et al. (2002) calibrated the *Action Points* model for close-following situation based on four thresholds proposed by Leutzbach and Wiedemann (1986). In this section, this new calibrated close-following model is examined through the simulation performance.

3.5.1. Model Description

Brackstone et al. (2002) calibrated the Action Point Model using time series data acquired from field tests with instrumented vehicles to describe the close following behaviour of UK drivers. As stated above, the model is based on four thresholds given by Leutzbach and Widedmann (1986):

(a) Minimum desired following distance, d_{min};

(b) Maximum desired following distance, d_{max};

(c) A threshold for recognizing small negative (closing) relative speeds, ΔV_a;

(d) A threshold of recognizing small positive (opening) relative speeds, ΔV_b.

Table 3-8 Values applied in the close-following model (Brackstone et al. 2002).

Notations	Algorithms / Values
d_{min} (m)	$d_{\min} = L_n + C_1\sqrt{v_n(t)}$
d_{max} (m)	$d_{\max} = L_n + C_1\sqrt{C_2 \times v_n(t)}$
ΔV_a (m/s)	-2
ΔV_b (m/s)	2
A_3 (m/s^2)	0.6
*Note: τ (s)=0.6s is applied in the simulation according to the review of Toledo (2003)	

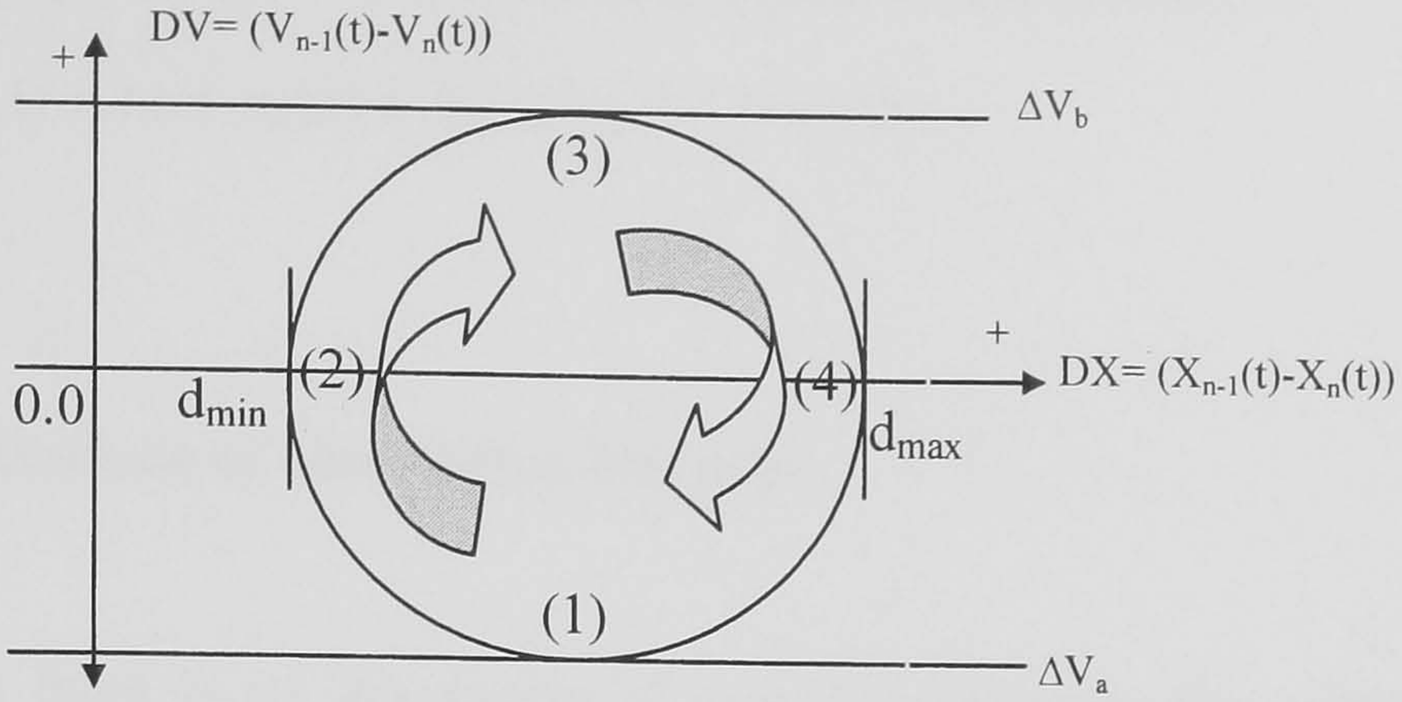


Figure 3-15 The close-following spiral in a plane of relative speed (DV) and space gap (DX) according to Brackstone et al. (2002).

The model is illustrated by Brackstone et al. (2002) as a bounded spiral in the relative speed and space gap diagram reproduced in Figure 3-15. The traces of the close-following are divided into four quarter cycles. First, the *closing process* (the bottom left hand quarter of the spiral in Figure 3-15), where the following vehicle is driving faster than its preceding vehicle and the gap between the two vehicles is decreasing, the follower would want to keep a larger space gap than his current one and begin to decelerate, while the relative distance with its leader is still decreasing. In the second, *continued decelerating process* (the top left hand quarter in Figure 3-15), the follower's speed is less than that of its preceding vehicle but the follower would still want to increase the space gap thus would keep decelerating. In the third, *opening process* (the top right hand quarter in Figure 3-15) the following driver will want to catch up with its leader and start to accelerate whilst the relative distance to its leader is still increasing. Finally, in the *continue acceleration process* (the bottom right hand quarter in Figure 3-15), the follower vehicle's speed exceeds that of the vehicle in front, their relative distance is getting smaller, and their state is moving towards the first, closing process.

Depending on whether a vehicle is in the deceleration or acceleration model, a constant acceleration or deceleration rate A_3 is applied. The speed of the vehicle is simply updated according to the Newtonian equations of motion:

$$v_n(t+\tau) = v_n(t) + A_3\tau \qquad \text{for acceleration} \qquad (3\text{-}9a)$$

$$v_n(t+\tau) = v_n(t) - A_3\tau \qquad \text{for deceleration} \qquad (3\text{-}9b)$$

The position of the vehicle, in all states, is then updated as:

$$x_n(t+\tau) = x_n(t) + \frac{1}{2}(v_n(t) + v_n(t+\tau))\tau \qquad (3\text{-}10)$$

3.5.2. Analysis of Simulation Outputs

As there is no description of continuity between the normal car-following states and the close-following state, the close-following model calibrated by Brackstone et al. (2002) is simply combined with Gipps' car-following model (1981) for the simulation results analysis. The simulation results are examined at the macro level first. Figure 3-16 shows that in the simulation speed drops suddenly during traffic build up: at time 780 s the speed is 64 km/h and after 2 minutes the speed is 51 km/h, i.e. in 2 minutes speed dropped by 13 km/h. As earlier illustrated in Figure 3-1 (section 3.1), a speed drop in the real traffic is found such that within 3 minutes, speed dropped by 43 km/h. Thus, this model can represent speed breakdown, however, with a smaller drop scale. Figure 3-17 shows that traffic hysteresis can not be reproduced in the flow-occupancy diagram, i.e. no loop structure can be identified from the flow-occupancy diagram. Therefore, by only applying close-following modelling in the simulation, it can represent speed breakdown, but fails to represent the traffic hysteresis.

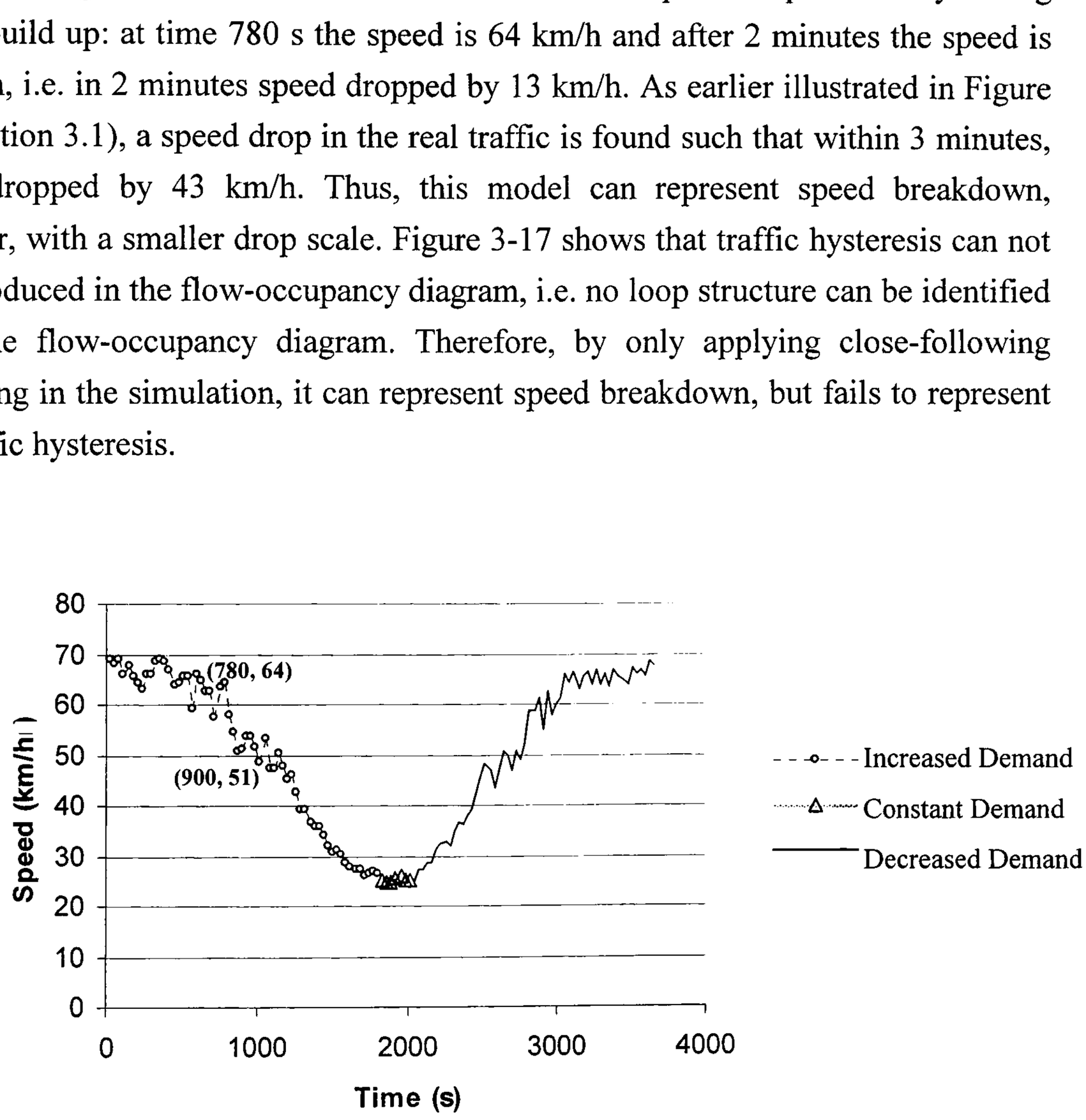


Figure 3-16 Speed-time diagram simulated with application of close-following situation.

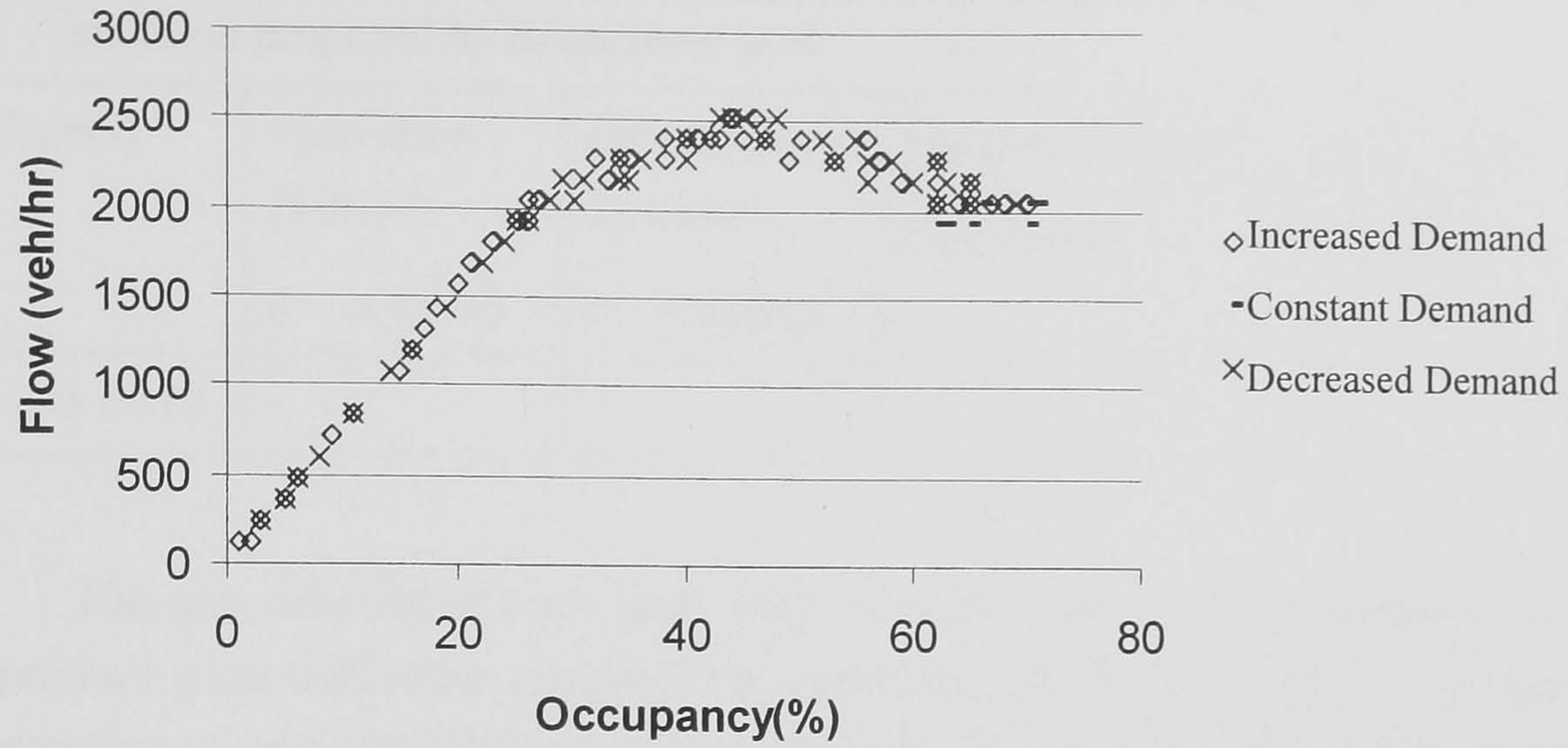


Figure 3-17 Flow-occupancy diagram simulated with application of close-following situation.

The shockwave propagation is examined from the plot of individual vehicle trajectories. As shown in Figure 3-18, the traffic is examined during the simulation time 1000~1200 seconds (traffic getting congested according to Figure 3-16). It shows that only one backward propagated shockwave (bold slanted line) can be identified from Figure 3-18. As this shockwave happened at the position from 700~750 m, according to the experimental test design (section 3.3.1) there is no detector located within this area. Therefore, the speed of this shockwave can be obtained by directly measuring the slope of such shockwave from Figure 3-18. The shockwave (backward forming shockwave) speed is around 20 m/s, which is higher than the real observations (8~18 m/s).

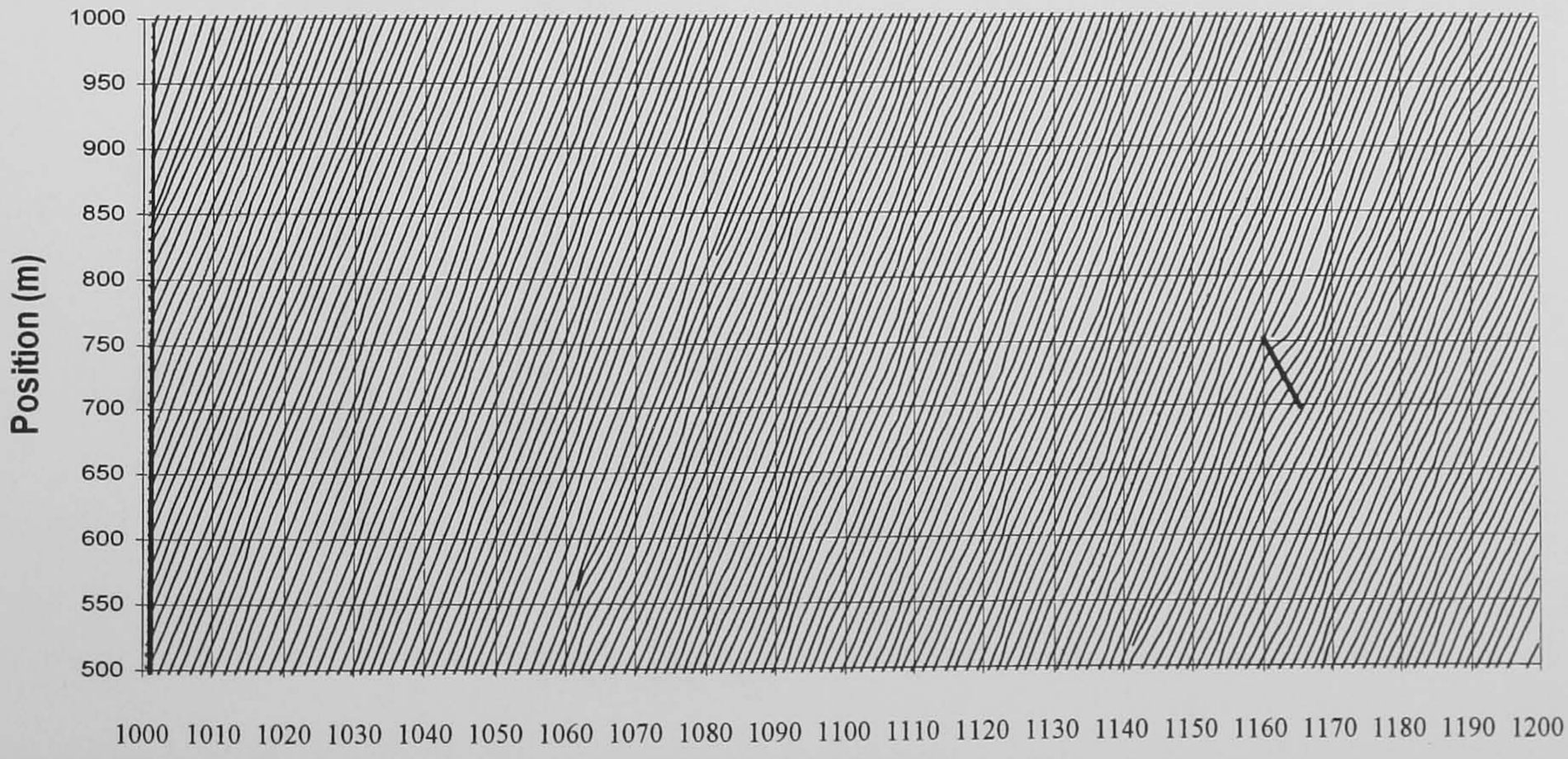


Figure 3-18 Plots of individual vehicle trajectories simulated with application of close-following situation.

Table 3-9 Shockwave speed of the simulation considering the close-following situation proposed by Brackstone et al.

Time(s)	*Flow from Detector* q (veh/s)	*Density from Detector* ρ (veh/m)	*Shockwave speed* v_{shock} (m/s)	
			Calculated	Measured
1160~1170	-	-	-	-20

The gap distributions are used here to reflect the model's performance in reproduce close-following situation (as explained earlier in Chapter 1, a certain percentage of vehicles (29%) can drive with small time-gap (less than 0.8 s) when in close-following situation). By examining the time gap among the individual vehicles (excluding those above 5 seconds) in the simulation, the results show that the gap distribution is a much better approximation to the real world observation than that simulated by Zhang and Kim's model and Gipps' model (Figure 3-19). It could be explained that by applying the close-following model, the Brackstone et al.'s model can reproduce vehicles driving with small gaps (i.e. < 0.8 s) at certain level. However, the proportion of smaller gaps is 8.1%, still a bit lower than the real data 29.2%.

This might be because in the simulation, the close-following vehicles do not "observe" several vehicles downstream ahead, as discussed earlier in section 3.1. Thus it results in the difficulty to adjust the speeds and positions appropriately and accordingly to the leaders' actions so to stay in the close-following state.

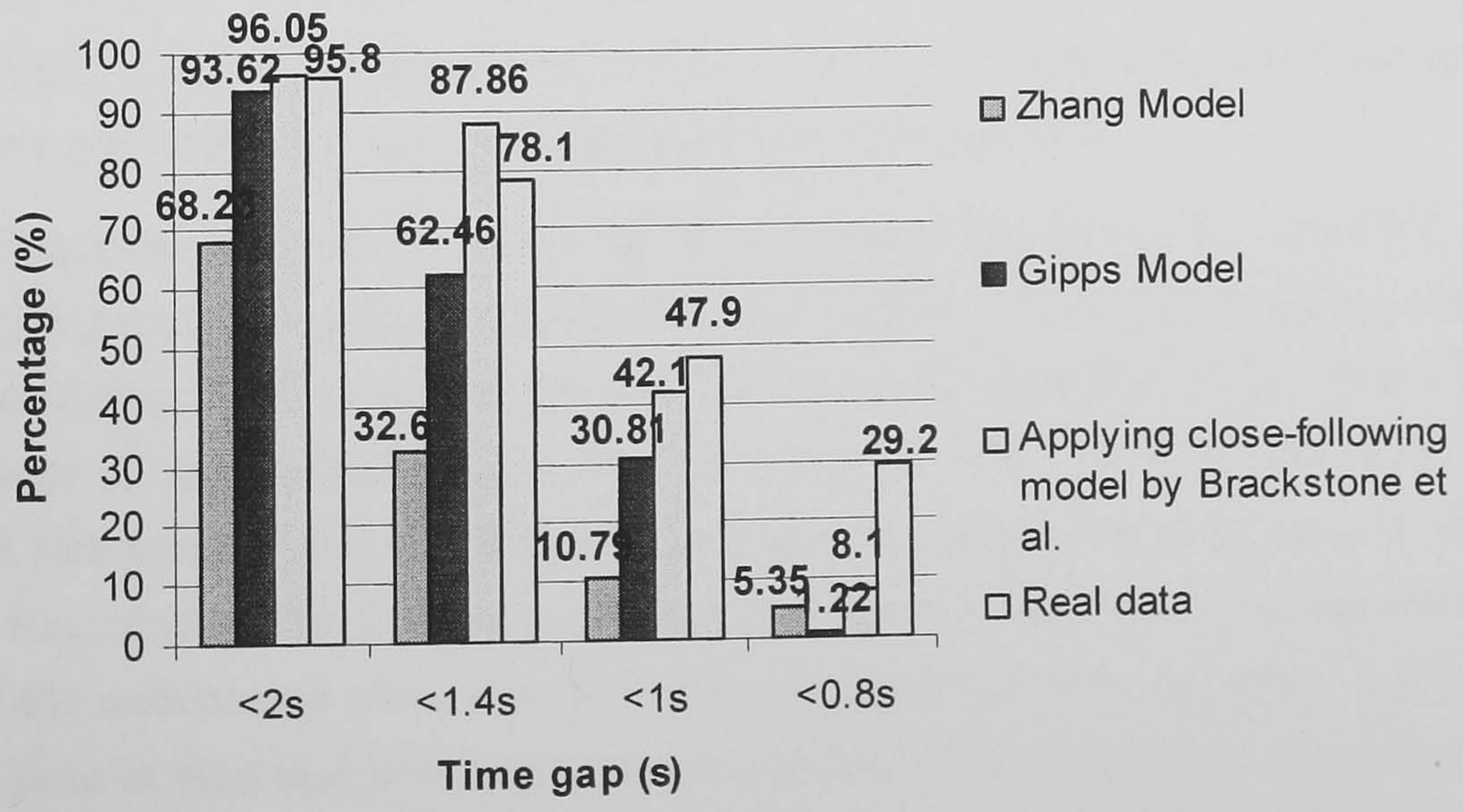


Figure 3-19 Gap distribution in the simulation by applying the close-following model by Brackstone et al. and the comparison with other models.

3.5.3. Summary of the Model Performance

From the simulation with the application of the close-following model by Brackstone et al., the results are examined both at macro- and micro- levels. It is found from this simulation that it can reproduce the speed breakdown but fails to represent traffic hysteresis. It can reproduce the backward propagated shockwave with a speed slightly higher than the real data. It is also found that the traffic is very stable with less frequent shockwave propagations compared to Gipps' model. The simulated gaps among individual vehicles have a higher percentage of close-following (8.1% of gap less than 0.8 s) than those simulated by Gipps' and Zhang and Kim's model; however, compared to the real world 29.2%, it is still low.

3.6. THE NEW CAR-FOLLOWING MODEL

Reaction time, which has been considered in many car-following models, is the time lag between the detection of a stimulus and application of the response (Toledo, 2003). Hereafter the term reaction time refers to a driver-vehicle unit. One of the common assumptions in car-following models is that drivers have the same reaction time and the same acceleration and deceleration throughout the different traffic states, i.e. whether they are in non-congested or congested traffic situations (e.g. Gipps, 1981). It has been shown that models of this kind cannot represent the speed breakdown and traffic hysteresis as discussed in earlier sections.

Empirical experiments have shown that reaction times to expected and unexpected stimuli are different (Johansson and Rumar, 1971; Fambro et al., 1998). The experiments made by Johansson and Rumar (1971) recorded the reaction times (the timings of vehicle brake lights after the klaxon in their experiments) of drivers who had forewarning that an incident was about to happen and compared to those without forewarning. They found a ratio of 1.35 between the mean braking reaction times of the unexpected stimuli to that of the expected stimuli. The idea of varying reaction time in alert and non-alert states has recently been applied to car-following studies (e.g. Toledo, 2003; Benekohal and Treiterer, 1988), which have successfully reproduced the hysteresis loops in the modelled flow-occupancy diagrams. Similarly, Zhang and Kim (2001) applied the idea for driving at different modes; they assumed a reaction time of 1.0 second when driving in cruising mode, 1.2 seconds for

deceleration and 1.8 seconds for acceleration. They have shown the effectiveness of the model in representing traffic breakdown and hysteresis. A detailed examination of Zhang and Kim's model (presented earlier) suggests that the relative large changes in reaction times could result in discontinuities in the movements of vehicles and in unrealistically high acceleration and deceleration. None of the models, which applied the varied reaction times, have been shown to reproduce the close-following phenomena of the motorway traffic.

The Gipps model has the advantage of representing realistically the individual vehicle's speed control (i.e. with respect to their mechanical capabilities) and shockwave propagation (Brackstone and McDonald, 1999; Wilson, 2001). It is also desirable for model calibration. It is believed that the close-following situation is the cause of traffic instability (Hounsell et al., 1992). Brackstone et al. (2002) calibrated the action point model and showed that it can reproduce small following gaps. The proposed new car-following model is developed based on a combination of the Gipps' safety-distance model and the action point model, and is aimed to represent the full range of motorway flow characteristics. Table 3-10 lists the key motorway flow characteristics aimed to be captured and the performance of the existing models in these aspects. Table 3-11 summarises the shockwave properties (absolute values) simulated by the models reviewed earlier: only Gipps' model can reproduce the shockwave realistically compared to the real observation.

Table 3-10 The key motorway flow characteristics aimed to be captured in the new model and the performance of the existing models.

Characteristics / Models	Speed Breakdown	Traffic Hysteresis	Shockwave Propagation	Close-following
Gipps (1981)	NO	NO	YES	NO
Zhang and Kim (2001)	YES	YES	YES (but unrealistic deceleration, such as -7 m/s^2)	NO
Brackstone et al. (2002)	YES	NO	Too stable to produce shockwaves	YES

Table 3-11 The shockwave properties of the existing models.

Characteristics / Models	Backward Propagated Shockwave	
	Shockwave speed (m/s)	Reasonability
Gipps	10-15	Yes
Zhang and Kim	5	No. (Some unrealistic of individual trajectories, i.e. sudden change of speed)
Brackstone et al.	20	No. (Traffic too stable to produce shockwaves)
Observation on M25	8-18	-

3.6.1. The Proposed Logic

The logic of the new car-following model is shown in Figure 3-20. The model is built on the concept that drivers in different traffic conditions (called states here) may have different acceleration and reaction time. It defines three states: non-alert, close-following and alert states.

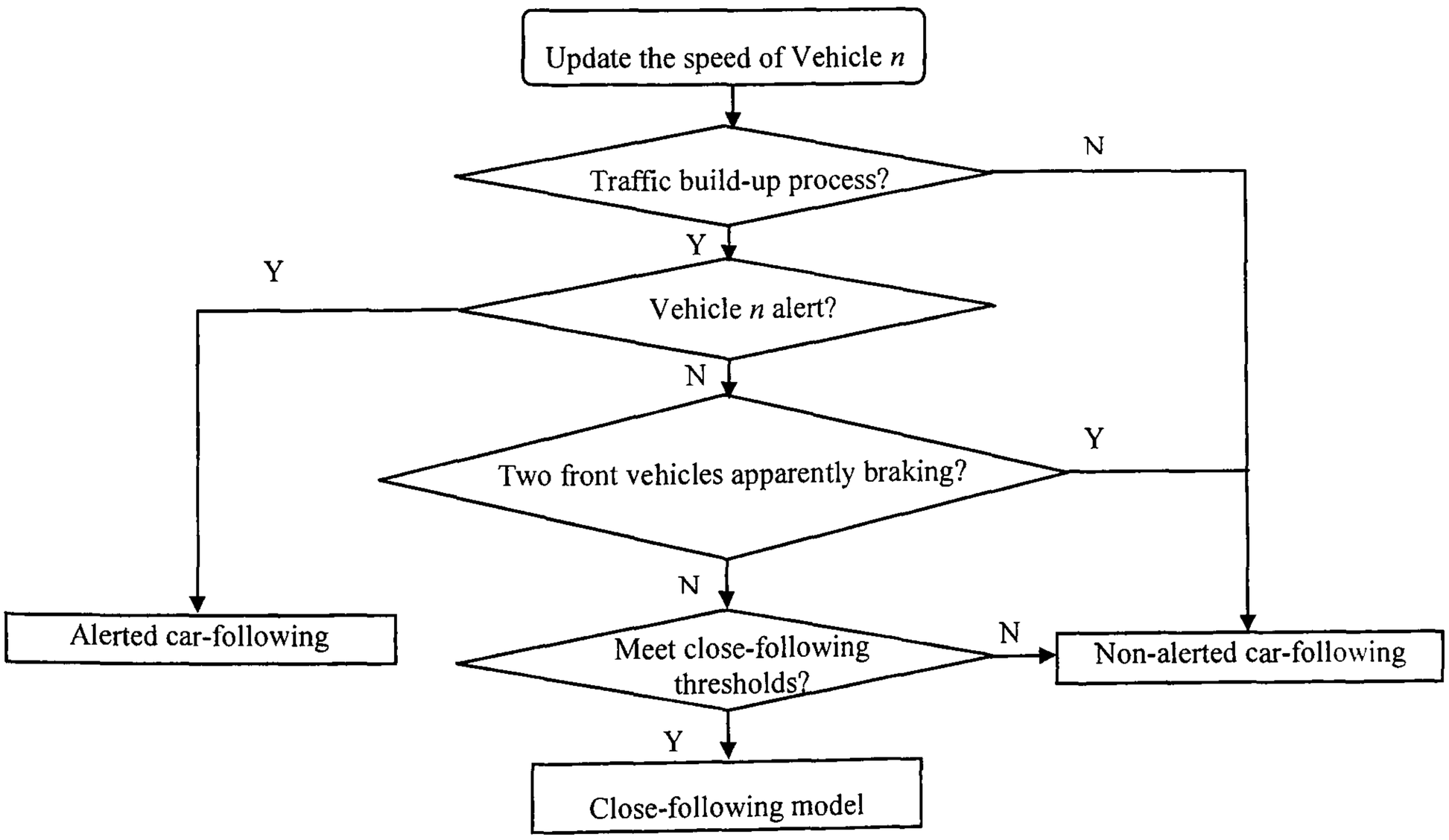


Figure 3-20 Flow chart of the new car-following model.

The model assumes that during the traffic build-up process (i.e. from non-congested to congested traffic), drivers shift from non-alert to alert state and this shift depends on the individual speeds (Dijker et al., 1998). Below this speed threshold, drivers are considered to be alert (shorter reaction time and higher acceleration/ braking). Above this speed, drivers are considered to be either not alert or to be close-following subject to the satisfaction of close-following thresholds defined later. The speed threshold (v_C) of 50 km/h is obtained from the observation of speed breakdown during traffic build-up process in the real traffic (Hounsell et al., 1992; Dijker et al., 1998).

During traffic recovery to free-flow states, however, the drivers are generally more relaxed and gradually increase their speeds (i.e. an overall acceleration of the traffic can be perceived). According to Zhang and Kim's hypothesis, the reaction time for acceleration is longer than other phases (2001). Thus during the process of traffic recovery when vehicles start to regain their desired speed, it is assumed the drivers are in the non-alert state with longer reaction times.

Close-following is defined in the model as that situation where none of the vehicles downstream in the platoon is apparently braking (i.e. the braking can not be noticed by the following vehicles) and the following vehicle will not brake very hard even when keeping a safe distance (Holland, 1998). In the model, the decelerations of only two front vehicles during the last reaction time interval in the platoon are checked. The action point model calibrated by Brackstone et al. (2002) (discussed in section 3.5) is used in the new model to represent the vehicle-following behaviour in the close-following situation, which gives the boundary conditions for close-following. The driver can adjust acceleration/deceleration to avoid collision and maintain a close-following spiral (described earlier in Figure 3-15).

The different states of the new car-following model and the mathematical formulation of the car-following behaviour at each state are summarised in Table 3-12 with the following Table 3-13 given the default values obtained from literature.

Table 3-12 The algorithms included in the new car-following model.

Conditions			Situation	Equations	Parameter
Traffic Build-up	$v_n(t) < v_c$		Alert Situation	eq. (3-5) - (3-7) and (3-10)	$\tau = \tau_1$, $a_n^{\max} = A_1$ $b_n^{\max} = -2a_n^{\max}$
	$v_n(t) \geq v_c$	$b_{n-1}(t-\tau) > D_C$, and $b_{n-2}(t-\tau) > D_C$, and $d_{\min} \leq S_n(t) \leq d_{\max}$, and $\Delta V_a \leq (v_n(t) - v_{n-1}(t)) \leq \Delta V_b$	Close-following Situation	eq.(3-9) and (3-10)	$\tau = \tau_3$, $a_n(t) = A_3$ $b_n(t) = -A_3$
		else	Non-alert Situation	eq. (3-5) - (3-7) and (3-10)	$\tau = \tau_1$, $a_n^{\max} = A_2$ $b_n^{\max} = -2a_n^{\max}$
Traffic Recovery	-		Non-alert Situation	eq. (3-5) - (3-7) and (3-10)	$\tau = \tau_1$, $a_n^{\max} = A_2$ $b_n^{\max} = -2a_n^{\max}$
Note: Due to the delay of drivers' reaction, $b_{n-1}(t-\tau), b_{n-2}(t-\tau)$ instead of $b_{n-1}(t), b_{n-2}(t)$ are checked at time t for the decision of close-following acceptance at the next moment $(t+\tau)$.					

Table 3-13 Parameters in the new car-following model.

Denotation	Default Values	Study
v_C Critical speed from non-congested to congested traffic (km/h)	50	Dijker et al. (1998), Hounsell et al. (1992)
τ_1 Reaction time in alert state(s)	0.6	Review, Toledo, (2003)
τ_2 Reaction time in non-alert state(s)	0.8	
τ_3 Reaction time in close-following state(s)	0.6	
A_1 Maximum acceleration in alert state(m/s^2)	2.18	Benekohal and Treiterer (1988)
A_2 Maximum acceleration in non-alert state (m/s^2)	1.7	Gipps, (1981)
A_3 Acceleration/deceleration in close following state (m/s^2)	0.6	Brackstone et al. (2002)
D_c Perceivable deceleration (i.e. braking lights are lit) (m/s^2)	-1.48	Benekohal and Treiterer (1988)

3.6.2. Theoretical Explanations

The theoretical analysis of the proposed car-following model is performed through the *uniform flow solutions* (Bando et al., 1995; Berg et al., 2000; Wilson, 2001) to derive the macroscopic flow-density functions. Firstly, section 3.6.2.1 explains the transitions between the different driving states in the new model. The detailed mathematical derivations of the Gipps' model and the new model for the different driving states are presented in later sections.

3.6.2.1. The transitions between the different driving states of the new model

The assumptions of the proposed model suggest that during traffic build-up (i.e. the vehicle's speed is more than the critical speed v_C), most drivers will be in the non-alert state (except a few who are close-following the leader, as described before, subject to the satisfaction to the close-following thresholds); when the traffic is congested (i.e. overall speed is less than the critical speed v_C), the driver will be in the alert state. During the traffic recovery, as explained earlier, it is assumed that drivers are quite relaxed thus in non-alert situation.

When the traffic builds up from the non-alert free-flow conditions, the vehicles may enter one of the two possible states. Some drivers may choose to drive in the close-following state subject to the satisfaction of the close-following thresholds. Such traffic will transfer from the non-alert state (along lines OA or AB in Figure 3-21) to the close-following state (on curve DE in Figure 3-21). Whilst their speed is above the critical speed of v_C, they will remain at the close-following state, or return back to the non-alert state if the close-following criteria can no longer be met. When the speed of the close-following traffic drops down to the level of v_C, the drivers start to transfer from the close-following state to the alert state (along path ECJ in Figure 3-21).

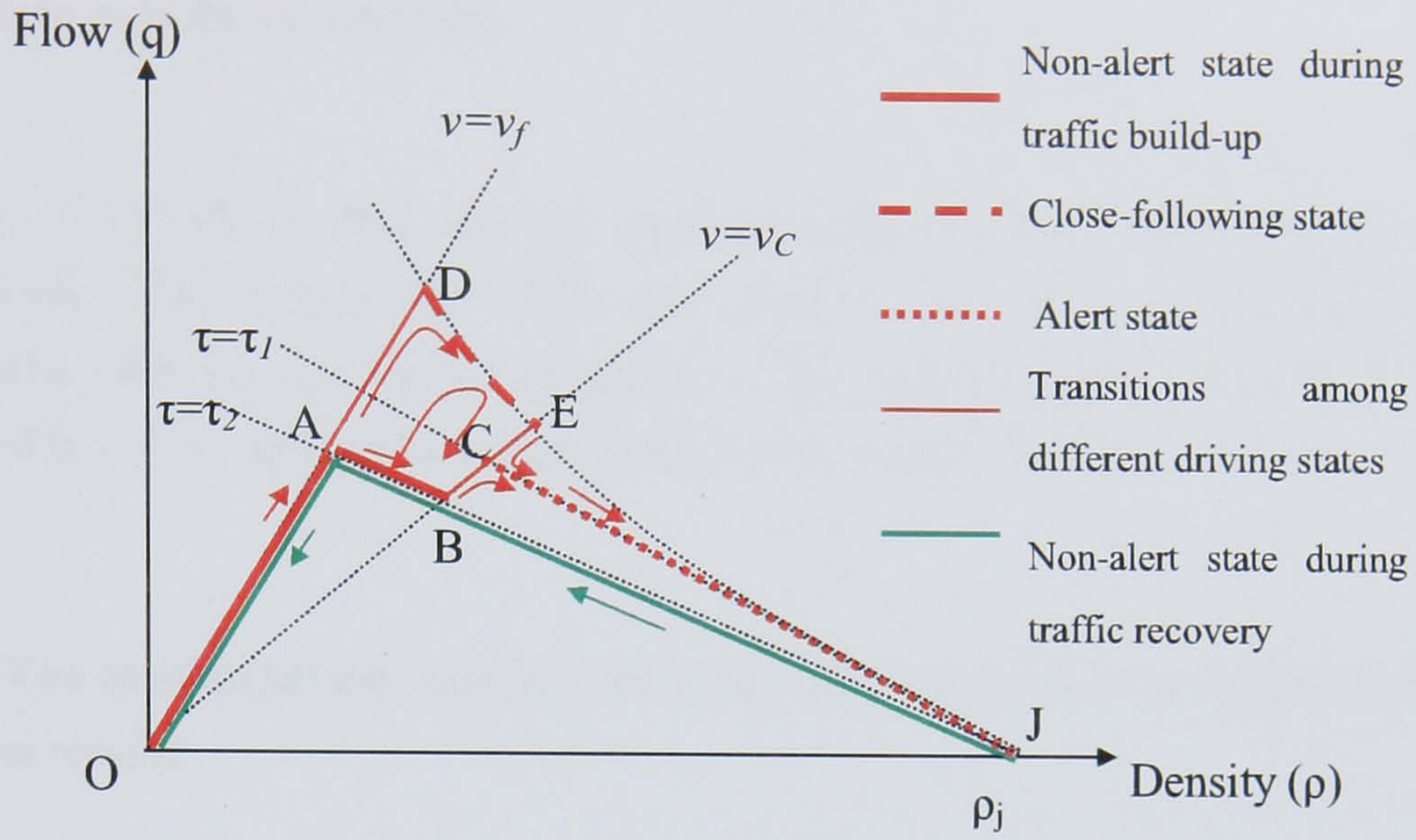


Figure 3-21 The transitions between different driving states.

Coming from the non-alert free-flow state, some drivers may never get into the close-following state. Instead, they stay in the non-alert state (along path OAB in Figure 3-21) and may be transferred to the alert state (along path BC in Figure 3-21) when their speeds reduce to the level of v_C.

When the speed of the traffic is below v_C, all drivers are assumed to be in the alert state with a smaller reaction time (τ_1). They may stay at that state till the traffic becomes completely jammed (point J in Figure 3-21). During traffic recovery to free-flow states, however, the drivers are generally more relaxed and gradually increase their speeds (i.e. an overall acceleration of the traffic can be perceived). According to Zhang and Kim's hypothesis, the reaction time for acceleration is longer than other phases (2001). During traffic recovery when vehicles start to regain their desired speed, it is assumed that drivers are thus in non-alert situations with a longer reaction time (τ_2) (along line JB in Figure 3-21).

The relationships between flow (q) and density (ρ) of the non-alert states (ABJ), alert (CJ), and close-following (DE) are given as follows (the derivations of these functions are given in detail in sections 3.6.2.2 to 3.6.2.4):

Line ABJ: $$q = \frac{2}{3\tau_2}(1-\frac{\rho}{\rho_j}) \quad (3\text{-}11a)$$

Line CJ: $$q = \frac{2}{3\tau_1}(1-\frac{\rho}{\rho_j}) \quad (3\text{-}11b)$$

Curve DE: $$q = \frac{4}{C_1^2(1+\sqrt{C_2})^2}(\frac{1}{\rho} - \frac{1}{2\rho_j} + \frac{\rho}{\rho_j^2}) \quad (3\text{-}12)$$

where ρ_j is the jam density.

Eq. (3-11) shows that that the reaction times of the drivers affect the traffic flow levels. The bigger the difference between τ_1 and τ_2, the more clearly represented will be the traffic hysteresis. This property of the model is further examined later through numerical simulations in section 3.6.3.

3.6.2.2. The mathematical derivations of the alert and non-alert states in the new model

In the proposed car-following model, during the traffic build-up, two different driving states become possible: the *close-following state* (the theoretical analysis will be explained in 3.6.2.3) and the *non-alert state*. When the traffic flow and density increase further, the speed of the traffic may drop and the traffic may enter into an *alert state* where the drivers are more alert by adopting a smaller reaction time. The car-following behaviour for the alert and non-alert states are modelled based on the Gipps' model (Figure 3-22, the mathematical derivations refer to Appendix D), but with different behaviour characteristics.

Four processes are considered here and they are illustrated in a flow-density diagram in Figure 3-23:

(a) *the non-alert state of free-flow traffic* (point O-A) *and build-up traffic* (point A-B): drivers travel at a constant free-flow speed of v_f between points O and A (eq. (3-13) when the traffic is very light; and they follow the Gipps safety distance car-following model with reaction time τ_2 from A to B when the traffic is getting denser until the overall speed is at the level of v_C.

(b) *the transition from the non-alert state to alert state*: when the speed of the traffic reduces to v_C, drivers shift into the alert state along the trajectory of points B-C in Figure 3-23.

(c) *the alert state*: when the speed of the traffic reduces further (less than v_C), vehicles follow the Gipps' safety distance car-following model with a small reaction time τ_1 following the trajectory from point C to J in Figure 3-23.

(d) *the non-alert state during flow recovery and free-flow*: drivers travel following Gipps' car-following model with a larger reaction time τ_2 ($>\tau_1$) from point J to

point in Figure 3-23. When the traffic flow becomes light, vehicles return to the free-flow state along trajectory from point A to O.

As the flow density relationship for the non-alert free-flow state is represented in eq. (3-13), during the transition state, the flow-density relationship is simply:

$$q = v_c \rho \tag{3-13a}$$

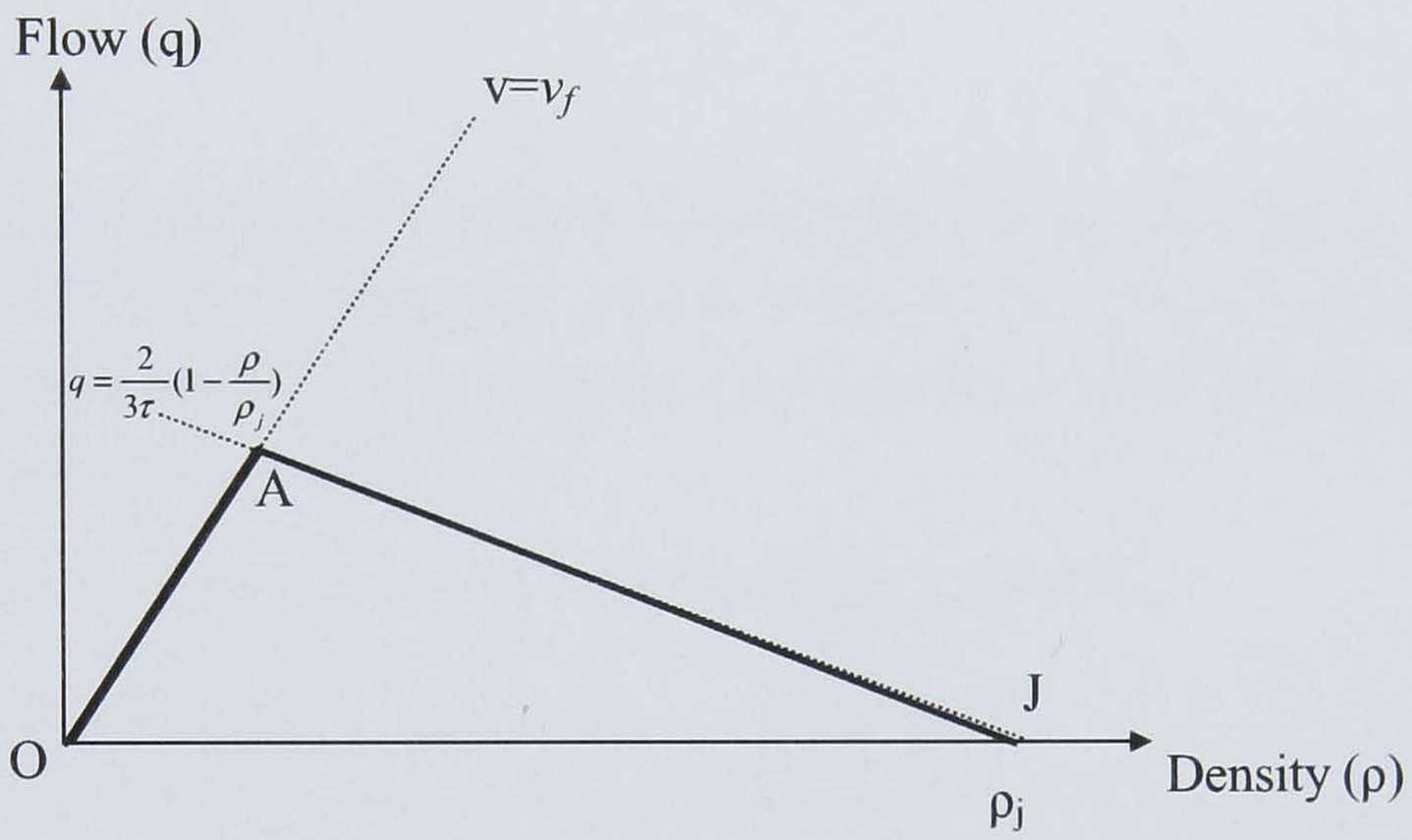


Figure 3-22 Fundamental diagram of the Gipps' car-following model.

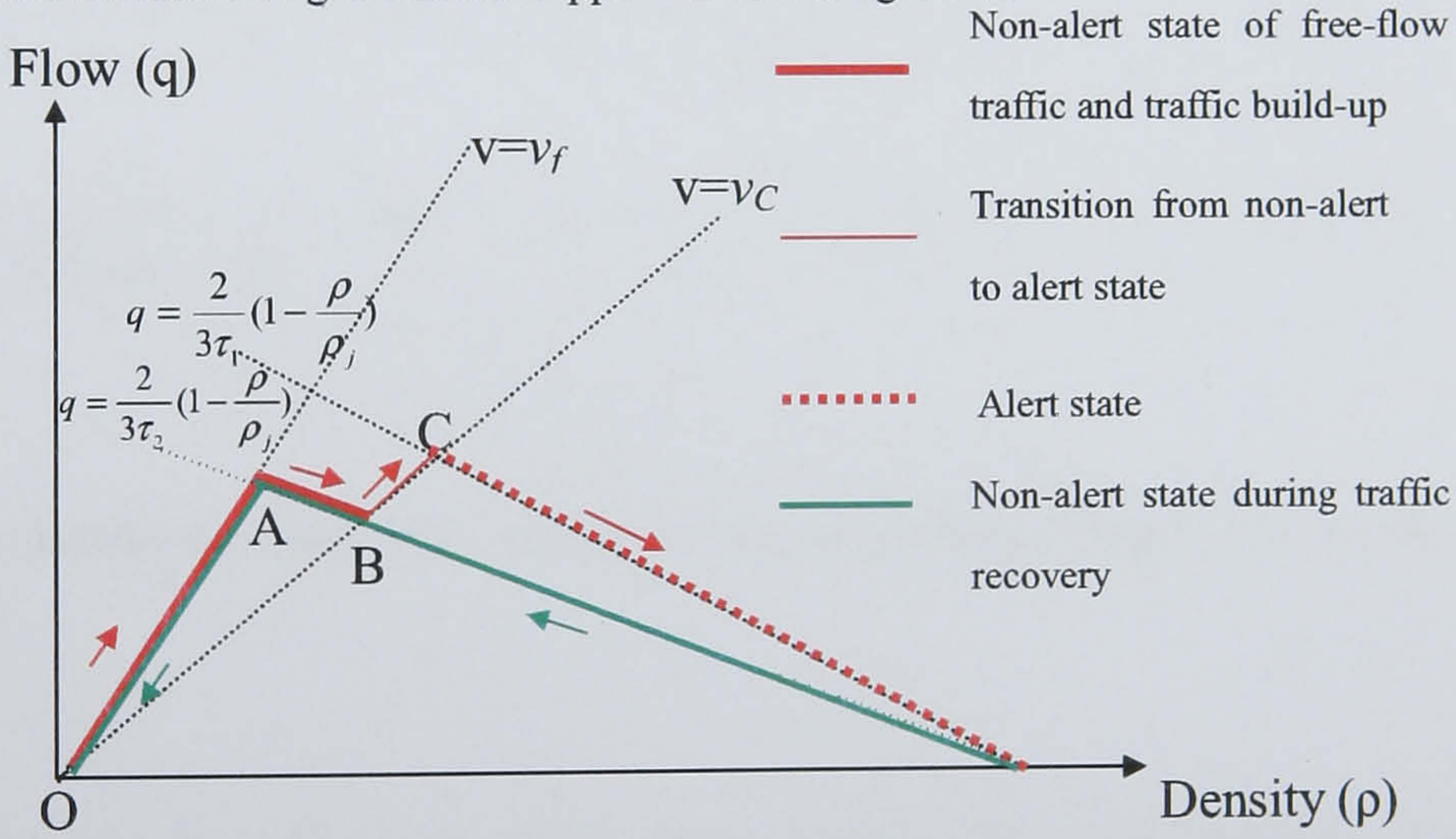


Figure 3-23 Fundamental diagram of the alert and non-alert states.

The flow-density relationships of the alert and non-alert recovery states follow the flow-density relationship of Gipps' model (mathematical derivations

given in Appendix D) with the appropriate reaction times. Therefore, for the non-alert states, as given earlier in section 3.6.2.1, it is:

$$q = \frac{2}{3\tau_2}(1 - \frac{\rho}{\rho_j}) \tag{3-13b}$$

And for the alert states, it is:

$$q = \frac{2}{3\tau_1}(1 - \frac{\rho}{\rho_j}) \tag{3-13c}$$

Given the assumption that the reaction time in the alert state (τ_1) is smaller than that in the non-alert state (τ_2), it is clear from eq. (3-13b,c) that the traffic flow in the alert state would be higher than that during the non-alert recovery state, hence the resulting traffic hysteresis.

The states at point A, B and C can be readily derived as:

$$\rho_A = \frac{2\rho_j}{3\tau_2 v_f \rho_j + 2} \quad \text{and} \quad q_A = \frac{2\rho_j v_f}{3\tau_2 v_f \rho_j + 2}$$

$$\rho_B = \frac{2\rho_j}{3\tau_2 v_C \rho_j + 2} \quad \text{and} \quad q_B = \frac{2\rho_j v_C}{3\tau_2 v_C \rho_j + 2}$$

$$\rho_C = \frac{2\rho_j}{3\tau_1 v_C \rho_j + 2} \quad \text{and} \quad q_c = \frac{2\rho_j v_C}{3\tau_1 v_C \rho_j + 2}$$

3.6.2.3. The mathematical derivations of the close-following states in the new model

During the *non-alert free-flow state,* traffic flow and density increase and vehicles get closer to each other. Some of these vehicles move into the *close-following state* subject to the satisfaction of the close-following thresholds as described in the main text.

For steady state macroscopic traffic flow, it is assumed all drivers want to drive stably with the headway as the average value between d_{min} and d_{max}. The headway of the close-following model can be written as:

$$x_{n-1}(t) - x_x(t) = 1/2(d_{\min} + d_{\max}) \tag{3-14}$$

$$= 1/2(L_n + C_1\sqrt{v_n} + L_n + C_1\sqrt{v_n C_2})$$

Replacing L_n with $1/\rho_j$ and $[x_{n-1}(t) - x_{n-1}(t)]$ with $1/\rho$, eq. (3-14) can be rewritten as:

$$q = \frac{4\rho}{C_1^{\,2}(1+\sqrt{C_2})^2}(\frac{1}{\rho} - \frac{1}{\rho_j})^2 \tag{3-15}$$

$$= \frac{4}{C_1^{\,2}(1+\sqrt{C_2})^2}(\frac{1}{\rho} - \frac{1}{2\rho_j} + \frac{\rho}{\rho_j^{\,2}})$$

Let $C = \frac{4}{C_1^{\,2}(1+\sqrt{C_2})^2}$, eq. (3-15) can be rewritten as:

$$q = C(\frac{1}{\rho} - \frac{1}{2\rho_j} + \frac{\rho}{\rho_j^{\,2}}) \tag{3-16}$$

Note that C is a positive value, the first order derivative of the flow shown in eq. (3-17) will always be negative. This means that the flow decreases with the increase of density with a minimum where $\rho = \rho_j$.

$$q'(\rho) = C(\frac{1}{\rho_j^{\,2}} - \frac{1}{\rho^2}) < 0 \tag{3-17}$$

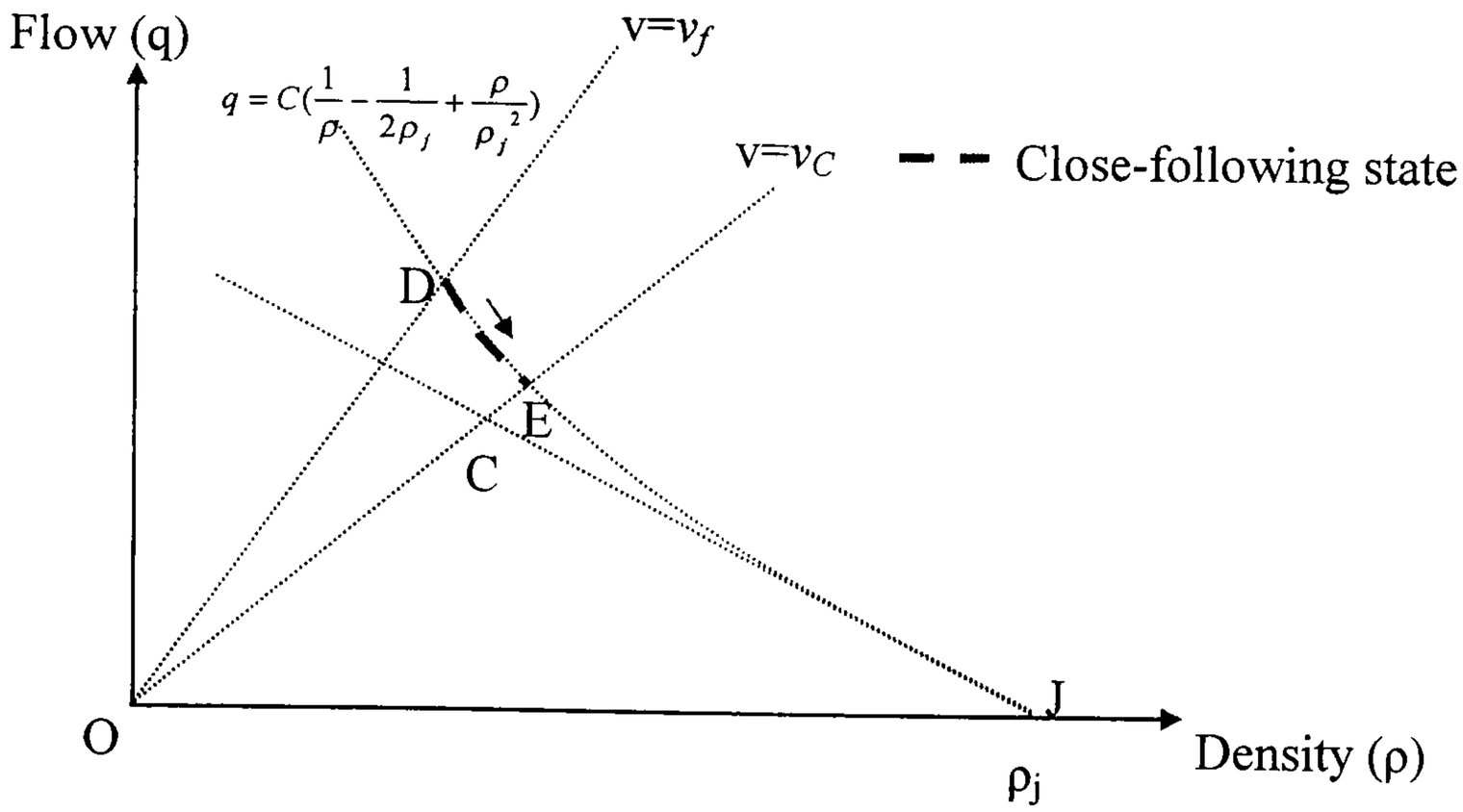


Figure 3-24 Fundamental diagram of the close-following state.

From eqs. (3-16) and (3-13a), the states at points D and E can be derived as:

$$\rho_D = (\frac{1}{\rho_j} + \frac{C_1(1+\sqrt{C_2})}{2}\sqrt{v_f})^{-1} \text{ and } q_D = v_f(\frac{1}{\rho_j} + \frac{C_1(1+\sqrt{C_2})}{2}\sqrt{v_f})^{-1}$$

$$\rho_E = (\frac{1}{\rho_j} + \frac{C_1(1+\sqrt{C_2})}{2}\sqrt{v_C})^{-1} \text{ and } q_E = v_C(\frac{1}{\rho_j} + \frac{C_1(1+\sqrt{C_2})}{2}\sqrt{v_C})^{-1}$$

Comparing state at point C (section 3.6.2.2) with that at point E, we get:

$$q_E > q_C \quad \text{for } v_C > [\frac{C_1(1+\sqrt{C_2})}{3\tau_1}]^2$$

Taking the values of C_1=2.96 and C_2=2.5 as calibrated by Brackstone et al. (2002) and the reaction time τ_1=1 second, it can be seen that:

$$q_E > q_C \quad \text{for } v_C > 6.5 \text{ m/s.}$$

In the model, the speed threshold v_C is assumed to be 14m/s (50km/h), therefore, the above inequality would always be satisfied, i.e. point C should be lower than E. In the field data, point D can be readily observed as the maximum overall traffic flow, point J the jam density, and point A the maximum flow during traffic recovery which can be identified from the speed- and/or flow-time profiles. Points B, C and E can not be directly observed from the field data; however, the analysis presented in the next section shows that the boundary condition defined by these points does not have a significant effect on the modelled results.

3.6.3. Model Verification and Validation

Model verification is concerned with determining whether the model outputs are reasonable and consistent through sensitivity analysis tests. Sensitivity studies of the key model parameters are carried out, and the results are examined against the fundamental diagram of flow-density distributions. The validation of the new car-following model is executed later in the section with respect to its macroscopic properties (such as speed drop and traffic hysteresis) as well as its microscopic properties (such as shockwave propagation). It aims to investigate the traffic properties between the simulation and the observations from UK motorways with respect to the similarity in pattern, magnitude of values and trend.

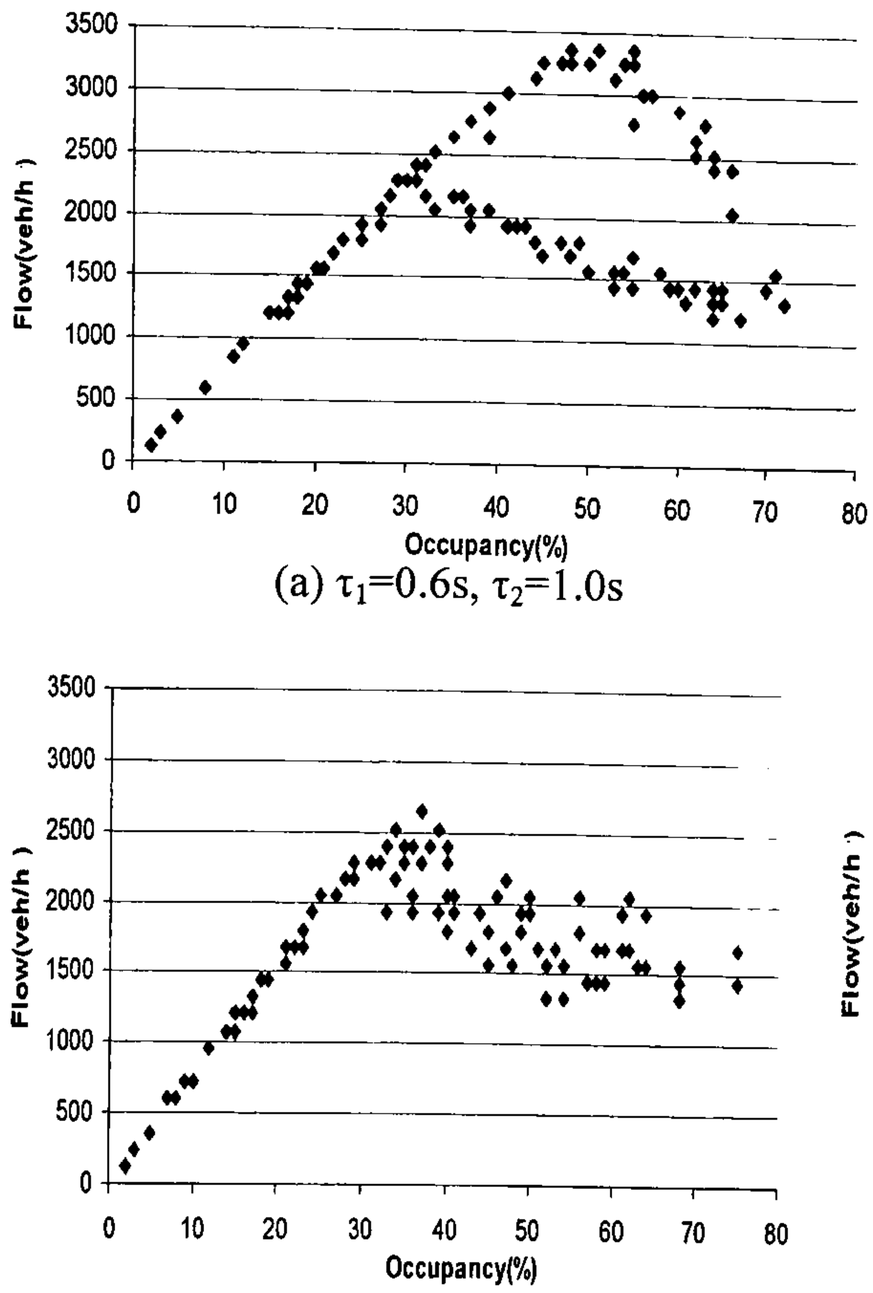
From the theoretical analysis described in Section 3.6.2, it has been shown that the reaction times affect the traffic flow reached during the traffic build-up and recovery process. The sensitivity tests are carried out on five model parameters: reaction time (τ_1) and acceleration (A_1) for the alert state, the reaction time for the non-alert state (τ_2) and acceleration (A_2) for the non-alert state, and the speed threshold (v_C) applied to distinguish different states. The tested range of the values is listed in Table 3-14.

Table 3-14 Parameters for sensitivity tests.

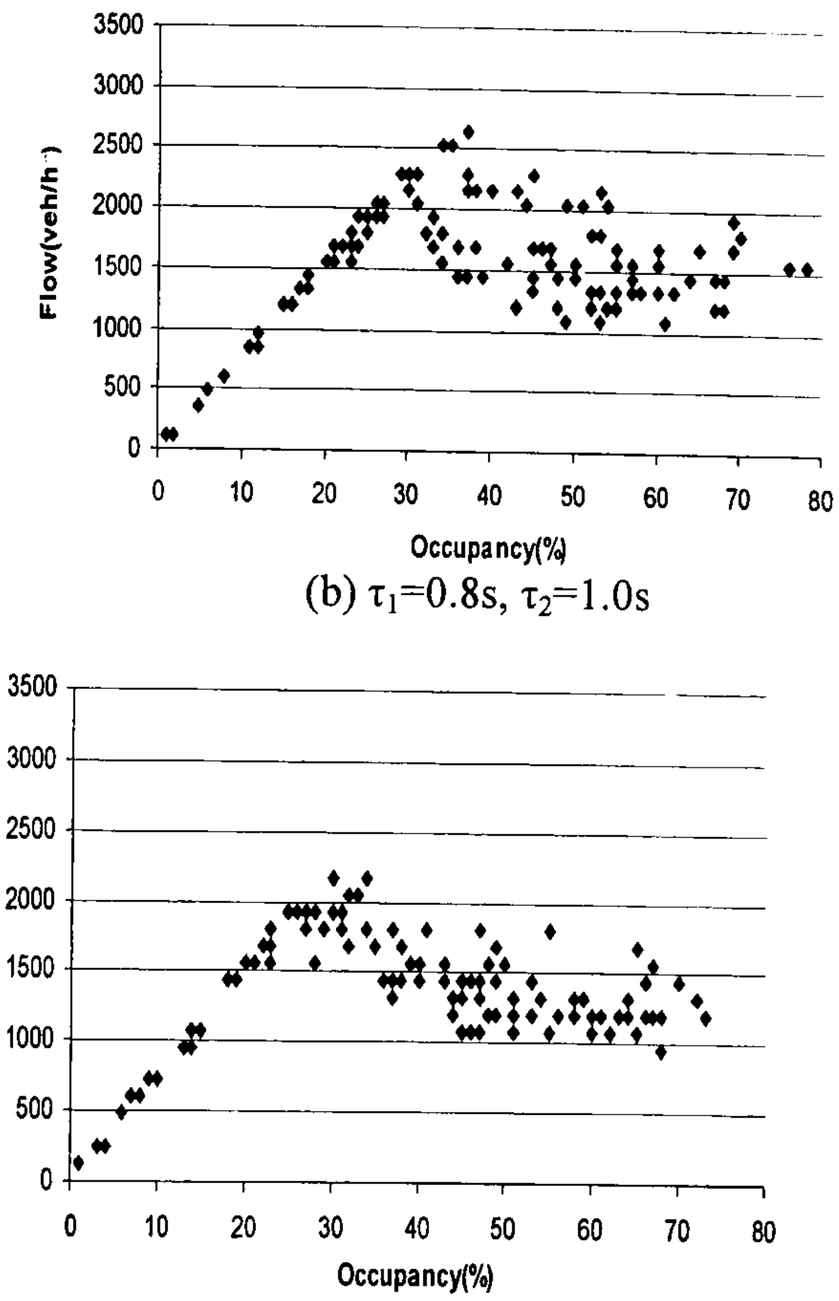
Parameters	Default Values	Reference	Test Range	Test Step
τ_1 (s)	0.6	Toledo (2003)	0.6-1.0	0.2
τ_2 (s)	0.8	Toledo (2003)	0.8-1.2	0.2
A_1 (m/s^2)	2.18	Benekohal and Treiterer (1988)	1.7-2.18	varies
A_2 (m/s^2)	1.7	Gipps (1981)	1.5-1.7	0.2
v_C (km/h)	50	Hounsell et al., (1992); Dijker et al., (1998)	40-60	10

Figure 3-25 shows the sensitivity of the simulation results to the reaction times. It can be seen that the bigger the difference between τ_1 and τ_2, the more clearly does the modelled traffic display hysteresis. From the results, we can also see that the model's capacity during traffic build-up is most sensitive to the alert reaction time τ_1 (Table 3-15): the lower the value of τ_1, the higher the capacity and its corresponding

occupancy. The results also show that the maximum flow during the traffic recovery process is sensitive to τ_2.



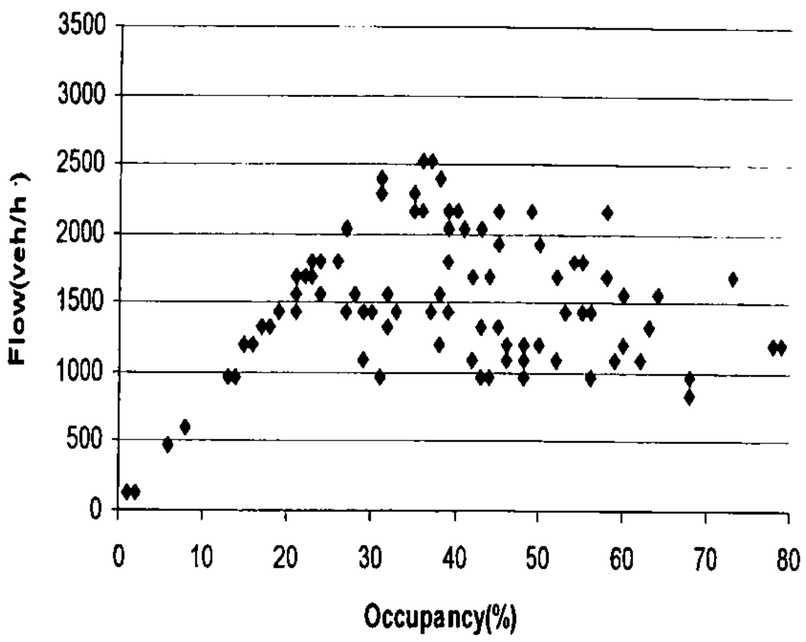
(a) τ_1=0.6s, τ_2=1.0s



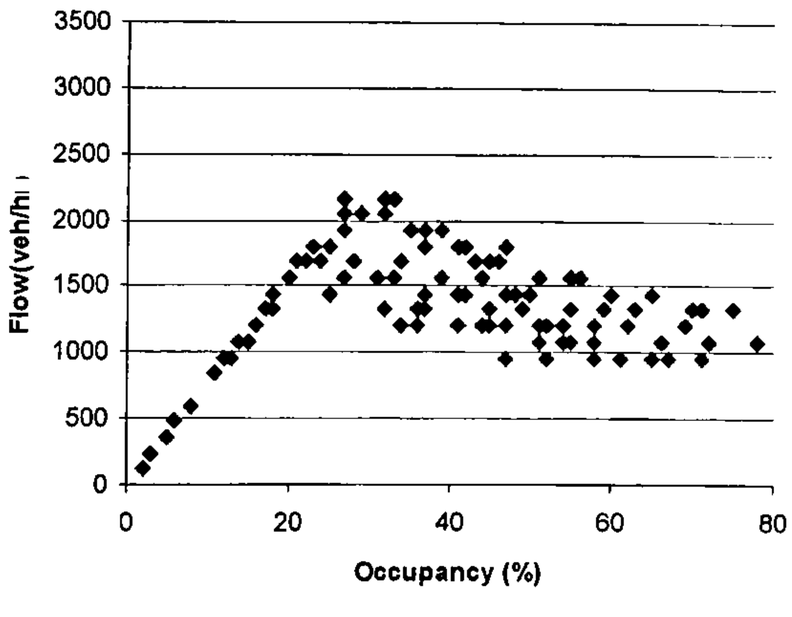
(b) τ_1=0.8s, τ_2=1.0s

(c) τ_1=0.8s, τ_2=0.8s

(d) τ_1=1.0s, τ_2=1.0s



(e) τ_1=0.8s, τ_2=1.2s



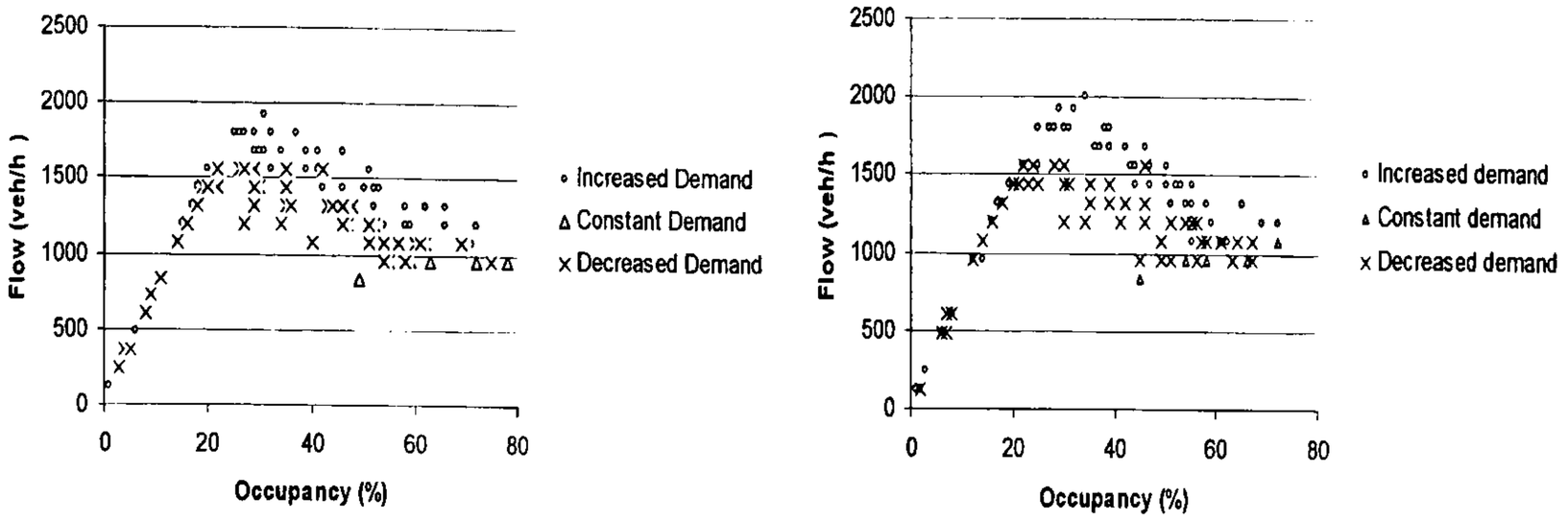
(f) τ_1=1.0s, τ_2=1.2s

Figure 3-25 Sensitivity analysis of the effect of reaction times τ_1 and τ_2 on the macroscopic flow-occupancy relationships.

Table 3-15 The capacity and corresponding occupancy under different τ_1 and τ_2.

Reaction Time (seconds)	Traffic Build-up		Traffic Recovery		Hysteresis
(τ_1, τ_2)	Occ (%)	Q_{max}(veh/h)	Occ (%)	Q_{max}(veh/h)	
(0.6, 1.0)	55	3560	31	2280	Yes
(0.8,1.0)	37	2640	29	2160	Yes
(0.8,0.8)	37	2640	-	-	No
(1.0, 1.0)	34	2160	-	-	No
(0.8, 1.2)	37	2520	23	1680	Yes
(1.0, 1.2)	32	2160	25	1800	Yes
Note: Q_{max} is the maximum flow the traffic can reach before the traffic break down.					

Figure 3-26 shows the sensitivity of the simulation results to the accelerations A_1 and A_2. It can be seen that the changes to the accelerations do not affect the hysteresis properties: e.g. Figure 3-26(b) shows that with $A_1 = A_2 = 1.7$ m/s^2, the model can still reproduce the hysteresis loop. From the results, it can also be found that the model's capacity during traffic build-up is slightly sensitive to the A_1 (Table 3-16): the lower the value of A_1, the lower the capacity; however the effects of A_2 on the model's maximum flow during traffic recovery are not clear. It is found that the effect of speed threshold v_C however, is less significant on the traffic capacity and its corresponding occupancy (Figure 3-27).



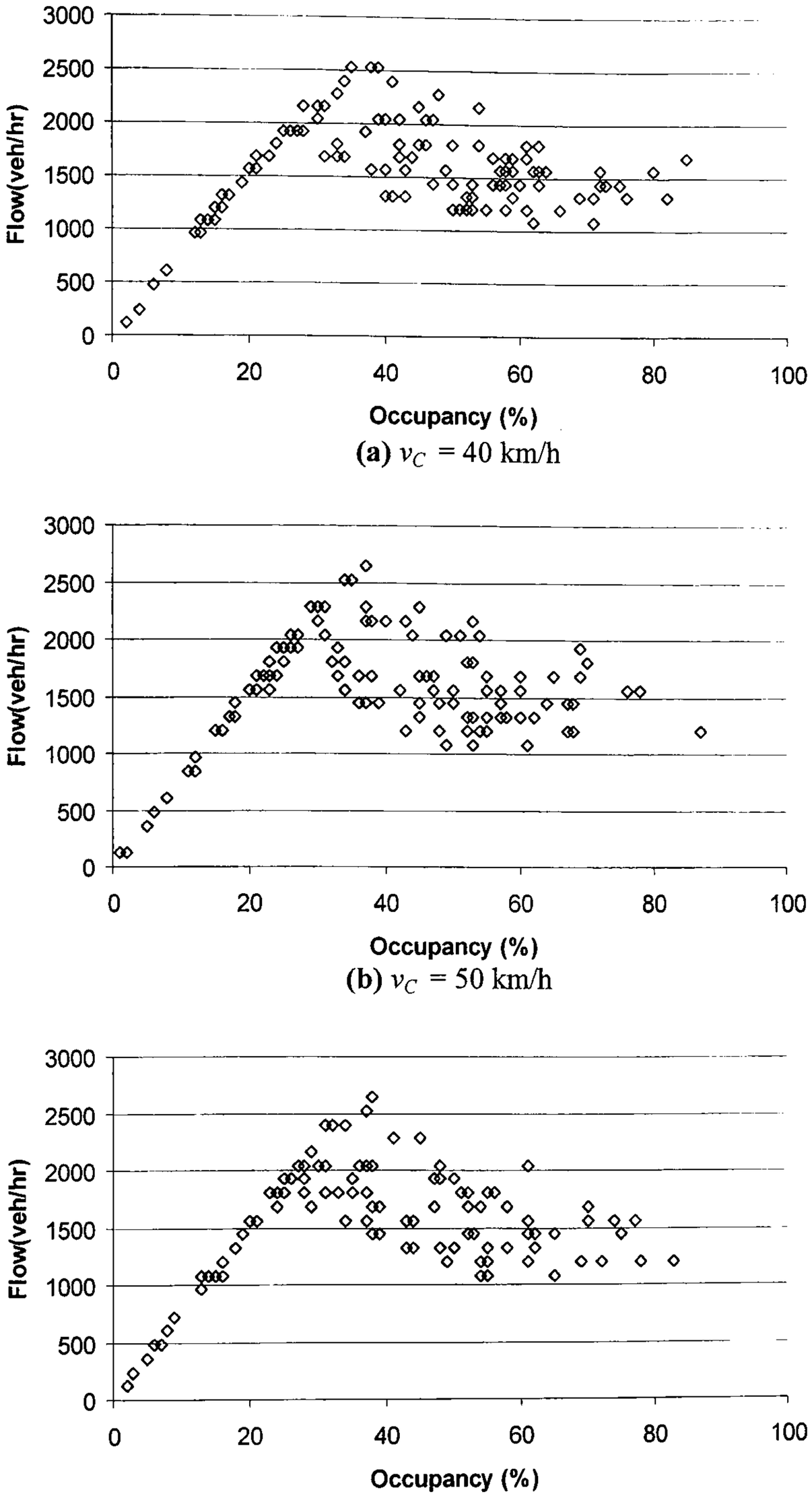
(a) A_1=1.7m/s^2, A_2=1.5m/s^2 (b) A_1=1.7 m/s^2, A_2=1.7 m/s^2

(c) A_1=1.9 m/s^2, A_2=1.7 m/s^2 (d) A_1=2.18 m/s^2, A_2=1.7 m/s^2

Figure 3-26 Sensitivity analysis of the effect of accelerations A_1 and A_2 on the macroscopic flow-occupancy relationships.

Table 3-16 The capacity and corresponding occupancy under different τ_1 and τ_2.

Acceleration (m/s^2)	Traffic Build-up		Traffic Recovery		Hysteresis
(A_1, A_2)	Occ (%)	Q_{max}(veh/h)	Occ (%)	Q_{max}(veh/h)	
(1.7,1.5)	35	1920	42	1560	Yes
(1.7,1.7)	34	2000	30	1560	Yes
(1.9,1.7)	34	2160	30	1680	Yes
(2.18,1.7)	36	2160	31	1800	Yes



(a) v_C = 40 km/h

(b) v_C = 50 km/h

(C) v_C = 60 km/h

Figure 3-27 The effects of different speed criteria v_C on the fundamental diagram.

Observations made on the M25 motorway outside London show that the capacity of a nearside lane is around 2000veh/h (Figure 3-2, section 3.1). From the tests conducted (Table 3-15), it is found that the test with τ_1=1 s and τ_2=1.2 s reproduces the observed capacity whilst also capturing some degree of traffic hysteresis. Thus results from this test are examined further for validation.

Figures 3-28 and 3-29 show the observations made from the M25 motorway in London. The M25 motorway encircles London and has 31 junctions. It is a vital component in Britain's motorway network (http://www.highways.gov.uk). The busiest, western section regularly carries up to 200,000 vehicles per day (http://www.highways.gov.uk; http://www.atkinsglobal.com). Most of the sections are dual 4 lanes except those that are connected to the ramps with only 3 lanes constructed. The data source for observation analysis is the morning one-minute aggregated detector data of the nearside lane (slow lane) collected on 6th February 2002, located between Junction 11 and 12 on the western section of M25.

The simulated data are overlaid on the observed data in Figures 3-28 and 3-29. Although the simulated data from a ring road can not be directly compared with the observation made from an open stretch of motorway, the aim here is to illustrate the scale of the modelled speed breakdown and traffic hysteresis compared to the scale of those observed. Two speed drops can be found in Figure 3-27 during the period 7:00 to 7:30 in the morning: one drop started at 7:00 (56 km/h) and finished at 7:07 (25 km/h); another sudden speed drop in the real traffic lasted for 4 minutes (starting at 7:13, 68 km/h to 7:17, 48 km/h), when the speed dropped by 20km/h. From 7:30 to 9:40 in the morning, the traffic was unstable with the speed oscillating around a low value less than 50 km/h. During this period, the speeds from 7:30 to 7:45 only are illustrated in Figure 3-28 due to the time scale difference between the simulation and the real observation. The traffic started recovering from congestion at 9:40 and reached the free-flow state at 9:50.

The plots superimposed on M25 data in Figures 3-28 and 3-29 are the simulated speed-time and flow-occupancy results respectively. It can be seen in Figure 3-28 that at time 870s into the simulation the model reproduced a speed drop from 66km/h to 45 km/h within 1.5 minutes, a drop of 21 km/h. The simulated level of the speed drop is comparable to that observed during period 7:00 to 7:30.

Figure 3-29 shows that during traffic build-up, the simulated flow reaches its maximum at 2160 veh/h. During traffic recovery, the maximum flow level is a mere 1800 veh/h. The maximum traffic flow reached during the traffic build-up process can not be achieved during traffic recovery; the two peaks differing by 360 veh/h. The observed data (during period between 6:00 and 7:20) shows that during traffic build-up the maximum flow was around 1900 veh/h; after the traffic breakdown the maximum was around 1680 veh/h, a difference of 220 veh/h. It is found that although the simulated data has slightly higher occupancy compared to the real data, it can reasonably capture the loop structure of the traffic hysteresis.

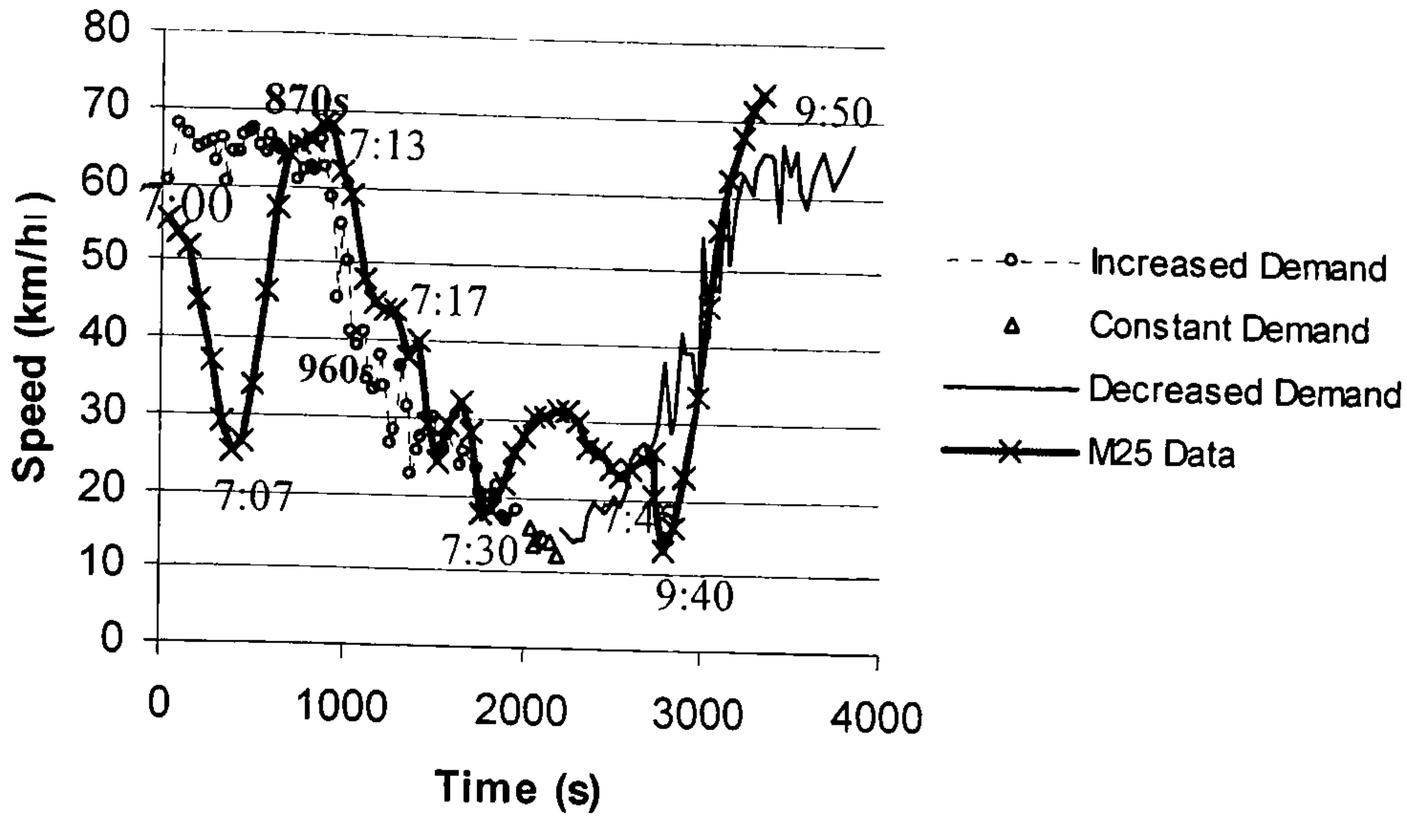


Figure 3-28 Speed-time diagram simulated by the new model.

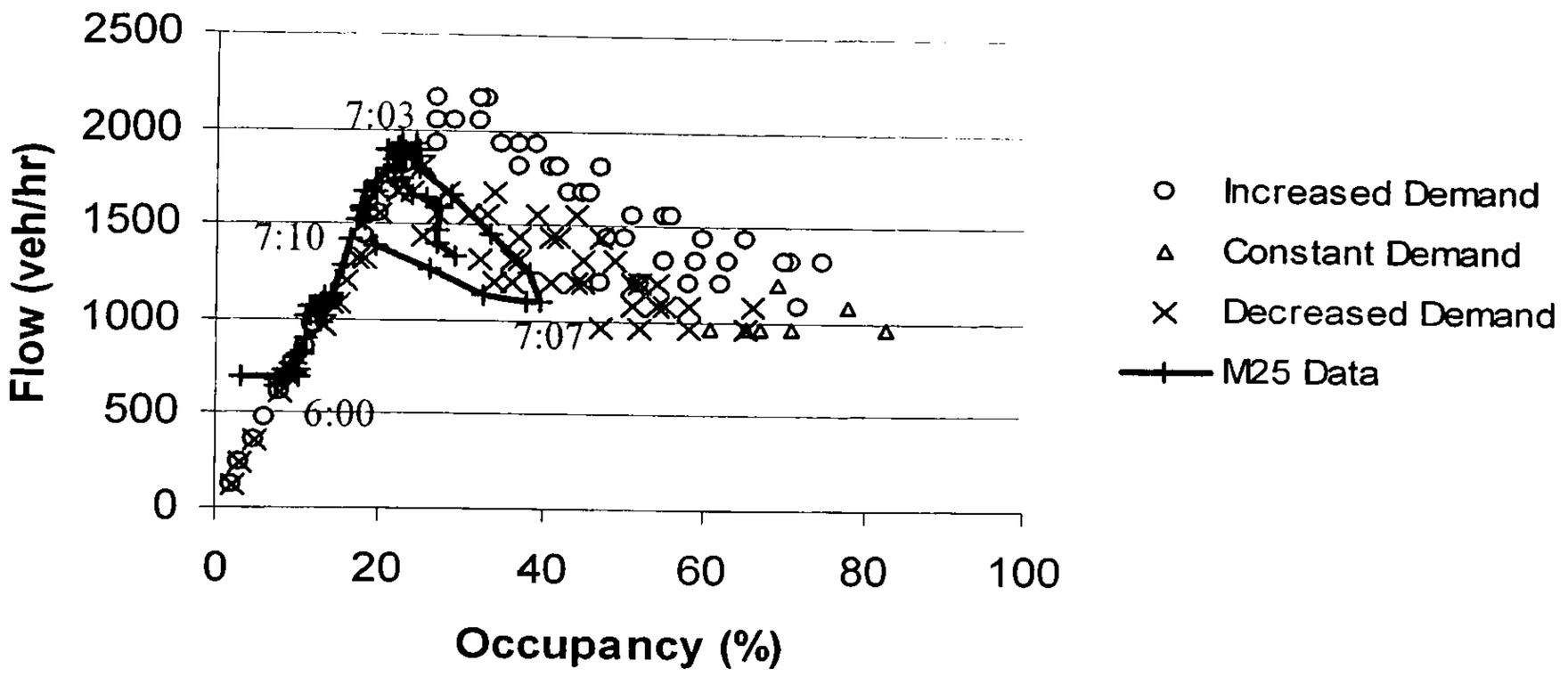


Figure 3-29 Flow-occupancy diagram simulated by the new model.

The shockwave propagation is examined from the plot of individual vehicle trajectories shown in Figure 3-30. The simulated traffic is examined during the period between 1050 and 1150 seconds where traffic started to get congested and unstable according to Figure 3-28 given earlier. Some shockwaves shown in Figure 3-30 can be easily identified with a reduction of traffic flow and velocity (i.e. the trajectories are less condensed). The shockwaves shown in Figure 3-30 are backward propagated shockwaves in accord with Lighthill and Whitham (1955) and May (1990) as the shock waves move upstream over time. Table 3-17 lists the shockwave speeds calculated using eq. (3-8) from detector data and measured from the

trajectory plots[7]. The shockwave speeds vary from -10 to -24m/s which are reasonable compared to M25 motorway where the backward propagated shockwave speeds were observed from -8 to -18 m/s. Therefore the simulated shockwave speeds are comparable with the observed ones.

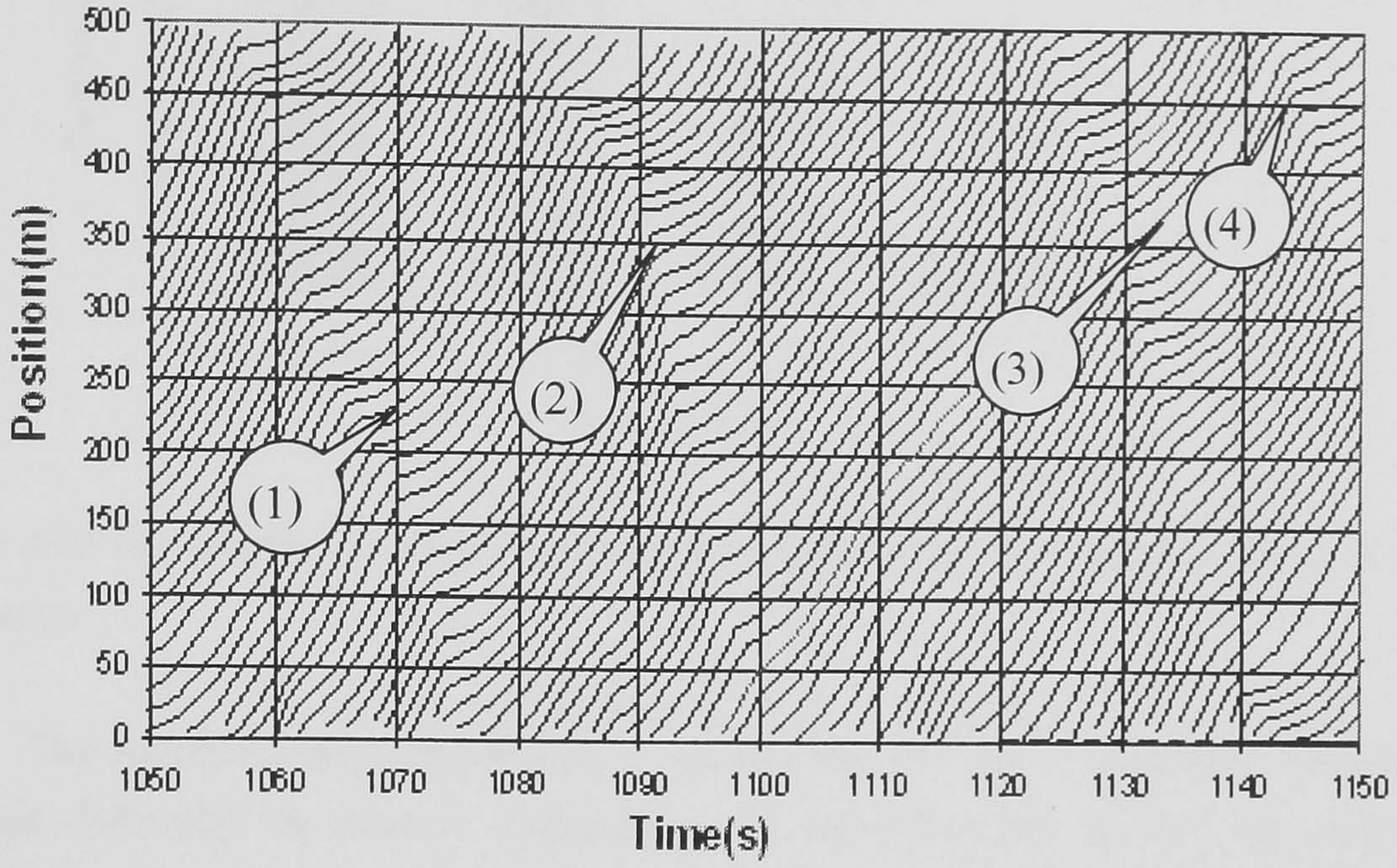


Figure 3-30 Plots of individual vehicle trajectories simulated by the new model.

Table 3-17 Backward propagated shockwave speeds.

Shockwave	Time(s)	q (veh/s)	ρ (veh/m)	v_{shock} (m/s)	
				Calculated	Measured
(1)	1060	0.80	4.87×10^{-2}	-24.85	-22±2
	1070	0.60	5.76×10^{-2}		
(2)	1090	0.80	5.37×10^{-2}	-22.52	-21±1
	1100	0.50	6.70×10^{-2}		
(3)	1130	0.80	4.90×10^{-2}	-17.56	-14±3
	1140	0.50	6.61×10^{-2}		
(4)	1140-50	-	-	-	-10±2

7 It should be noted that Table 3-17 shows some instant high flow values, e.g. 0.8 veh/h (2880 veh/h) which does not match the maximum flow at 2160 veh/h as given in Figure 3-29. This is because that the shockwaves occurred within a very short period as observed in Figure 3-30 (e.g. 10 seconds). However, in Figure 3-29 the speed and flow are the aggregated traffic profiles collection over 1 minute interval, which was not be able to show some instant high flow records.

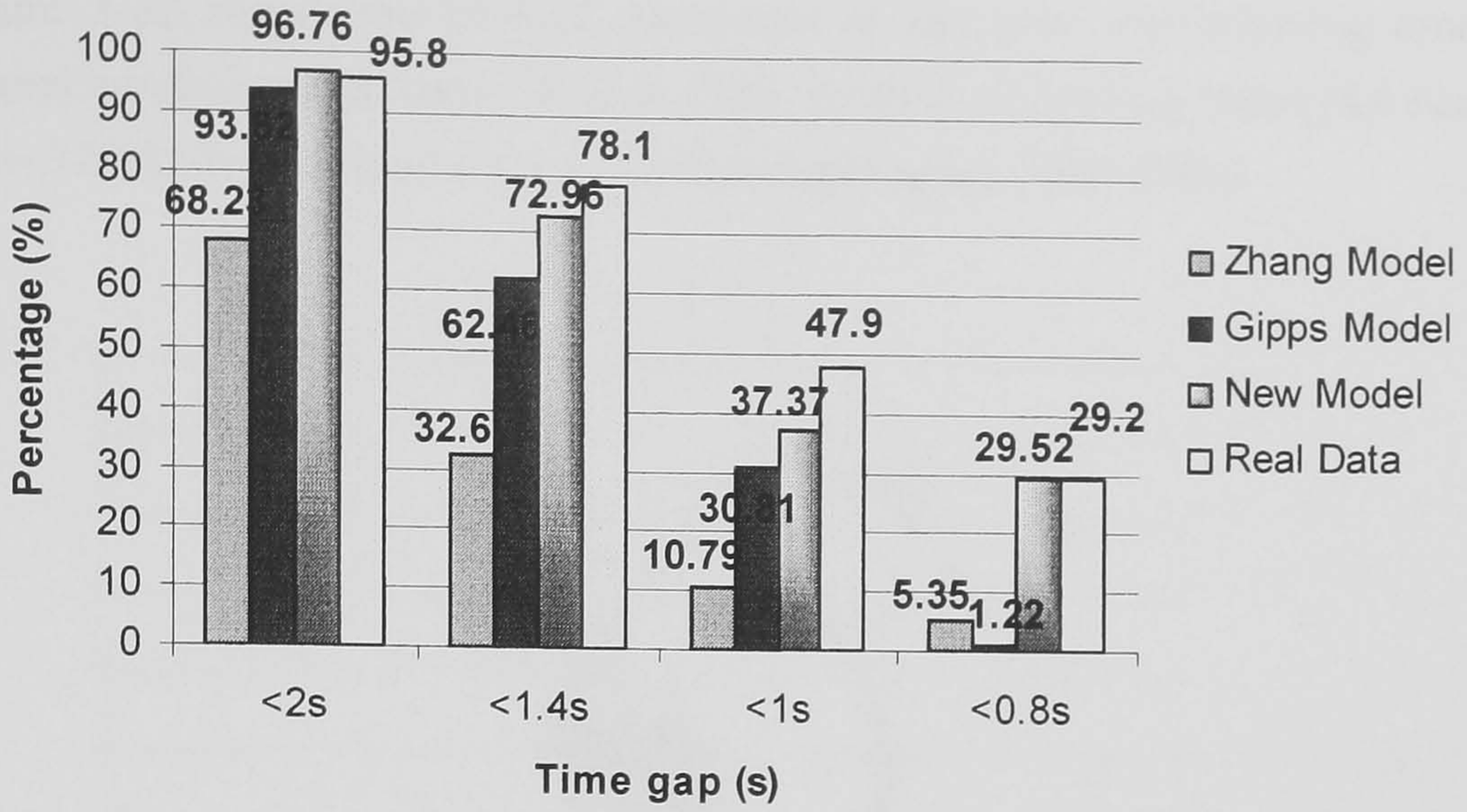


Figure 3-31 Gap distribution simulated by the new model compared to other models and real observation.

The modelled gap distribution is compared with those observed. Due to the apparent difficulty in directly calibrating the car-following model, the time gap distribution (Brackstone et al., 2002) is used here as an indicator of car-following model's performance. The simulated gap is collected from the time gap among the individual vehicles with only those below 5 seconds were collected (same as the earlier sections e.g. section 3.3.3). It should be mentioned that the gaps between 2s and 5s are not actually illustrated here (although they are collected) in Figure 3-31. This is due to the fact that very few drivers (4.2%) taking a gap greater than 2s when they are in following situation (Brackstone et al., 2002). The results are compared to the results simulated by Gipps' model (1981), Zhang and Kim's model (2001) (under the same experimental test design) and the real observation by Brackstone et al. (2002). As shown in Figure 3-31, it is found that generally, the distribution simulated by the new model is closer to the real data than those simulated by Gipps' and Zhang and Kim's model. It is possible that these models could improve their gap distribution performances if better calibrated. However, as neither of the models includes direct simulation of close-following situation, their capability of representing the smaller gaps distribution (e.g. less than 0.8 second) might be doubtful.

Figure 3-32 shows the smooth transitions in the new car-following model: from normal car-following states (700s to 720s) to close-following states (for nearly 2 minutes- 720-850s) and later back to normal states again (after 850s).

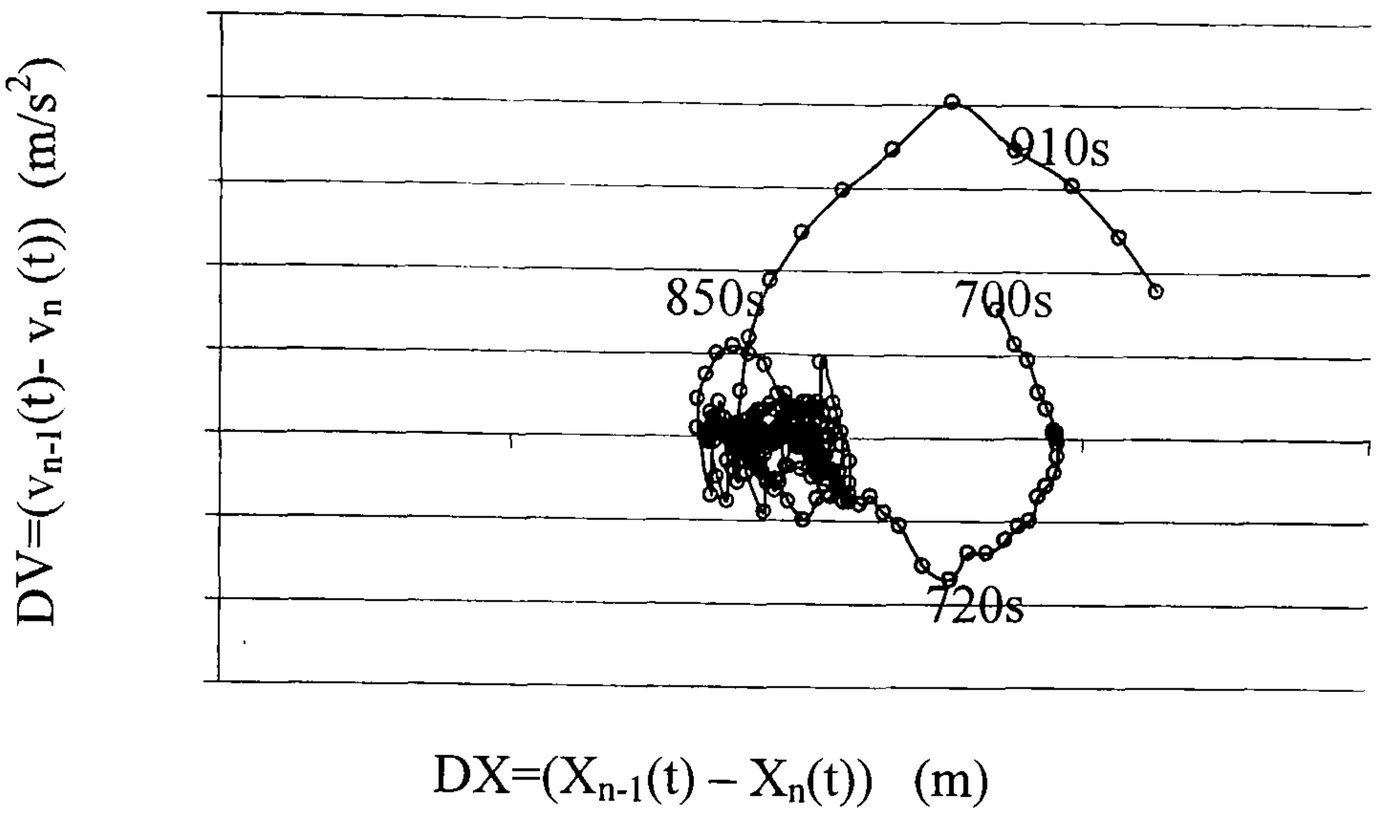


Figure 3-32 Transitions between normal car-following state to close-following state.

The sensitivity tests of the model parameters suggest that the model responds well at macro-level to the changes in the parameters of the alert reaction time and non-alert reaction time. The explanation of this may be that the smaller the alert reaction time, the smaller the acceptable spacings among drivers thus resulting in a higher capacity. The results indicate that the model can reproduce speed drop, traffic hysteresis and shockwave propagation as well as close-following behaviour.

3.6.4. Summary of the Model Performance

The phenomena of traffic breakdown, hysteresis, shockwave propagation and close-following are some of the key characteristics of motorway traffic flow. In order to better manage the motorway traffic, it is important that mechanism of these phenomena are fully understood, and that are able to be accurately represented. Microscopic simulation of vehicle following provides a flexible framework whereby the dynamical inter-vehicular interactions can be represented. This section presents a new car-following model which combines the idea of safe vehicle-following (Gipps, 1981) and that the drivers may vary their driving behaviour in different traffic states

(Brackstone et al., 2002). The model defines three driving states: non-alert, alert and close-following states, and applies to the driver different reaction times and different acceleration and deceleration values under the different states.

Simulation tests have shown that the model is able to realistically capture the speed drop, traffic hysteresis and shockwave propagation as well as close-following behaviour. Further sensitivity studies of the key model parameters suggest that the drivers' reaction times have a significant effect on the modelled capacity and occupancy, whilst the effect of the speed threshold, which distinguishes a congested from a non-congested traffic flow, is less significant.

The next section will focus on applying the model to an open stretch of UK motorway. A simulation study of the proposed car-following model will be applied to a 1.1 km stretch of the M25 motorway road that does not have any on-/off ramps in the vicinity of the site. There are three loop detectors placed along this stretch of the road. The upstream detector data will be used to generate the vehicles and their initial speeds into the simulation. Data from the detector located at the end of the section will be used to constrain the traffic movements downstream of the section. Calibration and validation will be carried out using data collected from the loop detector in the middle.

3.7. MODEL CALIBRATION AND VALIDATION

Calibration and validation process is a very important process in developing and applying micro-simulation models. This process is to ensure the models accurately replicate the observed traffic condition and driving behaviour. However, "*data availability often dictates what steps of calibration are feasible....usually available data is aggregate measurements of traffic characteristics (e.g. flow, speed and occupancy from loop detector) which are the results of the interactions between individual vehicles*" (Balakrishna et al., 2004). Besides, there is a lack of a commonly agreed bench-marking procedure for the calibration and validation of micro-simulation models (Brockfeld et al., 2005).

Model calibration is a process whereby the values of model parameters are adjusted so as to match the simulated model outputs with observations from the study site. It is usually formulated as an optimisation problem to determine the best set of model parameter values in order to minimise the discrepancies between the

observed and simulated values (Toledo, 2003; Balakrishna et al., 2004). Data from a different time period or from a different site can be used in the validation process, using the calibrated parameter values, in which measures of goodness-of-fit are used to quantify the similarity between observed and simulated data (Liu and Wang, 2005).

This section presents a general calibration and validation framework that attempts to rectify the model, using the most readily available traffic surveillance data, the loop detector data. The framework is designed for use in real-world applications upon an open-stretched highway network, rather than on artificial ring-type network (section 3.3.3). Selection criteria for the study site and how the detector data can be used in the framework are described first. The optimisation formulation of the model calibration process and a solution algorithm are then presented. Reference to the UK MIDAS data set (http://www.trlsoftware.co.uk) has been used here to illustrate the calibration and validation processes and how they can be used in real-life applications, it is worth mentioning that the concept and the proposed methodology is applicable to other detector data sets giving similar traffic measures.

Two states of car-following model, namely non-alert and alert states are calibrated and validated using aggregated data collected through loop detectors on M25 motorway. With reliable values calibrated by Brackstone et al. (2002) for close-following state, there is no further calibration work on that state here.

3.7.1. Data

3.7.1.1. Site description

When studying the car-following behaviour on motorways, the site selection should be free of constraints (such as road works, traffic accidents), traffic conflicts and geometric design deficiencies. On this basis the following criteria are required:

- No slip roads (i.e. no on-ramps and off-ramps) should exist in the vicinity of the site;
- No road works are happening in the vicinity of the site during the study period;
- Traffic on site experiences light to heavy traffic conditions during a normal working day.

It should be noted that weather can have considerable effects on driving behaviour. For example, driving behaviour may be different during foggy, rainy and windy weather according to Systems (2002). Also the normal working days' traffic flow may be quite different from that of weekends and holidays (DfT, 2003). The actual calibration and validation processes should take such variations into account. For general application and time limitation, the scope of this study includes only normal working days.

The data considered in this analysis are taken from the detectors installed along western part of the M25 motorway between Junction 11 and 12 (Figure 3-33). In this analysis, the section of 1100m of northbound motorway is selected with 3 detectors 4811A, 4817A and 4822A located across the lanes. Data for the analysis were collected on 28 February 2001 (Wednesday). In this study, only the nearside lane's data (lane 1) were selected for this calibration. The sketch of the locations of detectors is shown schematically in Figure 3-34.

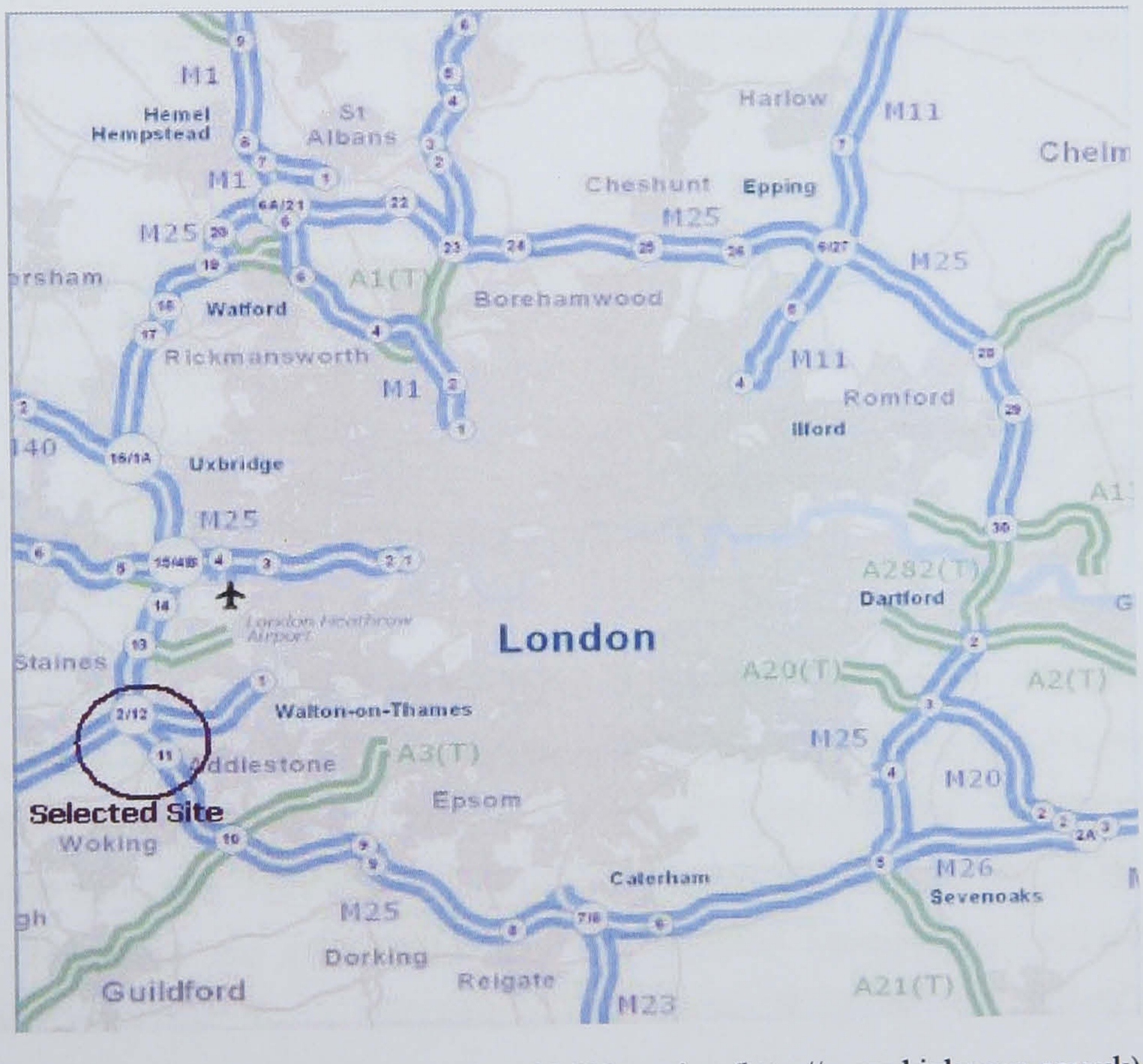


Figure 3-33 Location of the analysed M25 section (http://www.highways.gov.uk)

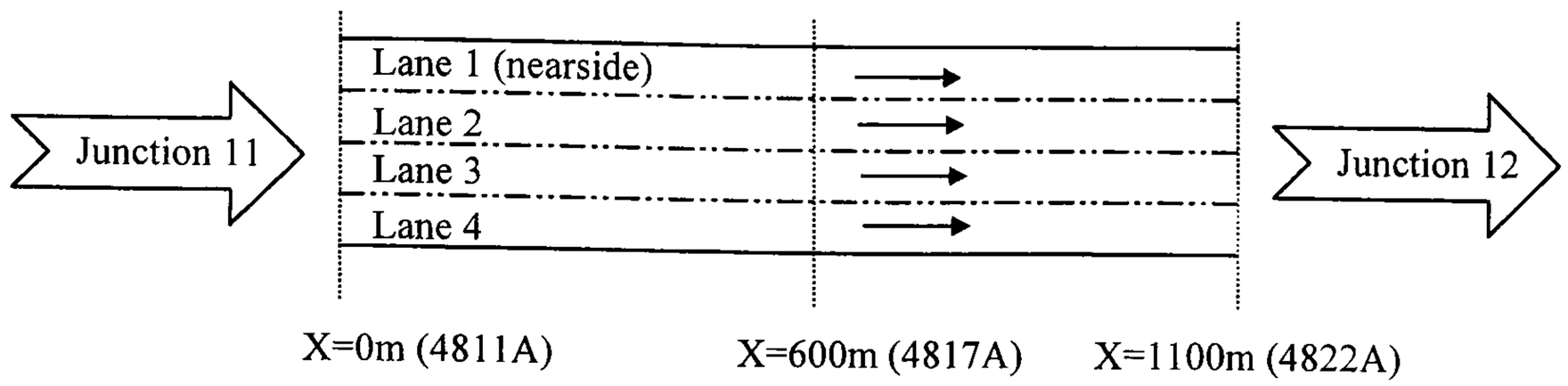


Figure 3-34 A schematic of the selected site on the motorway M25. The studied section starts at the site where detector 4811A is located- denoted X=0, and ends at where detector 4822A is located. All positions are measured relative to the point of X=0.

3.7.1.2. Loop detector data description

In the MIDAS data (http://www.trlsoftware.co.uk), the averaging time periods for data collection are one minute. For many traffic surveillance systems like the MIDAS system in the UK, the following basic data can be provided from each detector:

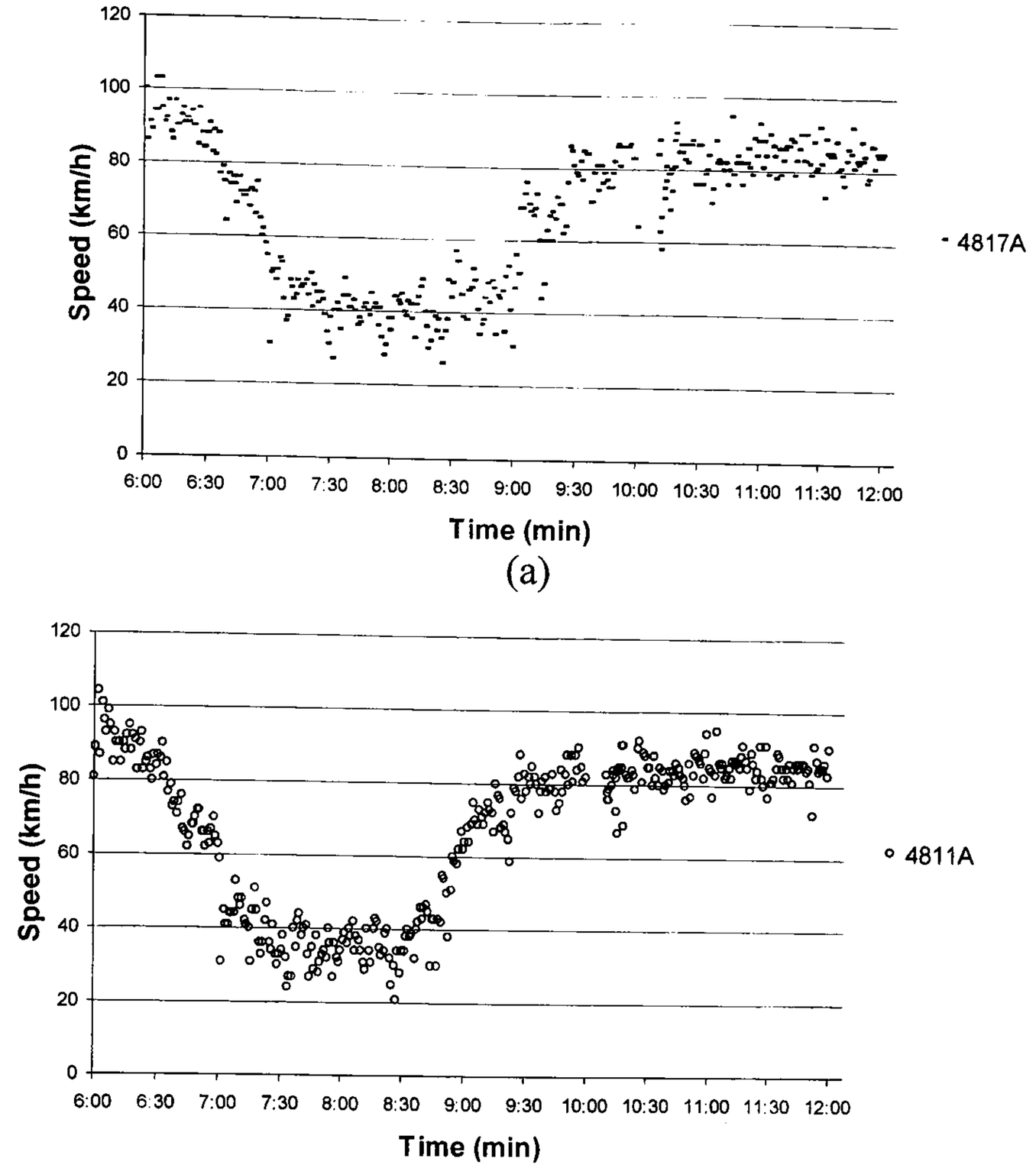
- Time-averaged speed by lane (km/h);
- Time-averaged flow by lane (veh/h);
- Time-averaged occupancy by lane (%);
- Time-averaged vehicle composition data, which categorise the traffic into four groups according to their length: Category 1: < 4.7 m; Category 2: 4.8-7m; Category 3: 7-11m; Category 4: >11m.

However, as discussed in Chapter 2, in the simulation only two types of vehicles, i.e. Car and HGV, are defined with the corresponding length attributes given earlier in Table 2-2. In order to put the above-mentioned vehicle composition information into simulation, the "category 1" vehicles are considered as "Car", and the rest categories are considered as "HGV". This treatment is according to the statistical theory that for a standard normal curve N (μ, σ^2), 99.7% of the area is contained within $\pm 3\sigma$ from the μ.

A typical example of the data collected by MIDAS system is shown in Figure 3-35. It shows the averaged speeds over time on a normal week day of 28 February, 2001, from three consecutive detectors located of the selected site on lane 1. It can be seen that generally the speeds measured from detector 4822A (the most downstream detector of the three) are higher than the speeds measured from other two upstream detectors, i.e. 4811A and 4817 A during 7:00 to 9:00 am. It might due

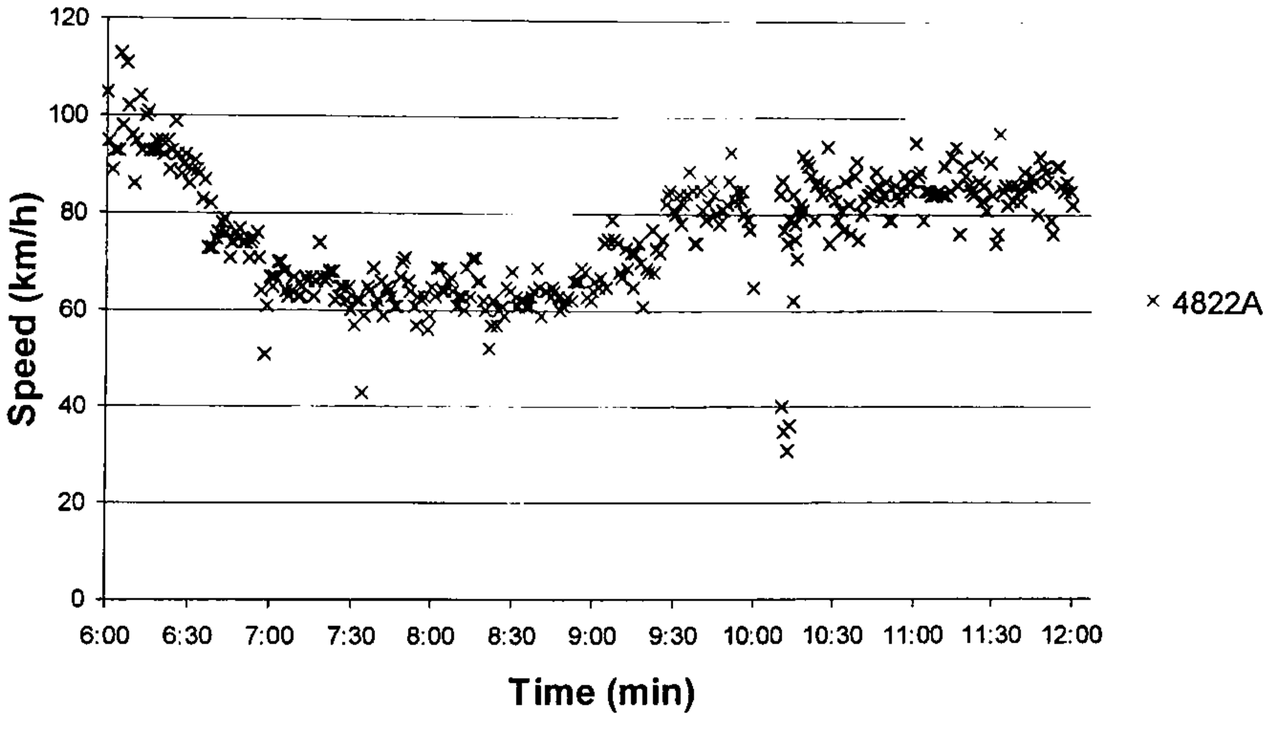
to the disturbance from on-ramp merging traffic at the upstream of the selected motorway section, and this disturbance affects more to the traffic nearby, i.e. where detector 4811A and 4817A are located. During 6:00 to 7:00 in the morning, the traffic started to build up but still with speeds greater than 50km/h. Between 7:00 to 8:00am, traffic at the two upstream sections began to break down and the speeds dropped below 50km/h; although the traffic speeds at 4822A have maintained at a higher speed at around 50km/h. This suggests traffic breakdown is likely to happen in the vicinity of merging areas, which has also been widely observed by Hounsell et al. (1992).

According to the model description discussed in previous section (section 3.6.1), the traffic was in the non-alert state during the morning period between 6:00 and 7:00. In another set of data collected from 7:00 to 8:00 in the morning, the speeds collected at 4811A and 4817A are generally lower than 50km/h; the speeds collected at 4822A are higher than 50km/h. Thus the traffic in the nearside lane on the selected site from 7:00 to 8:00 is used for the calibration of the combined two states, i.e. alert and non-alert car-following state.



(a)

(b)



(c)

Figure 3-35 The speed-time profile collected by detectors 4811A (a), 4817A (b) and 4822A (c) on 28 February, 2001.

Although the MIDAS system provides one-minute averaged data, the data used in the following sections are the three-minutes aggregated values so to reduce the "noise" of the original data.

3.7.2. Calibration Method

The calibration of the car-following model is carried out in a sequential procedure with two steps:

Step 1: Calibration of the non-alert state parameters with the data set collected between 6:00 and 7:00 in the morning,

Step 2: With the calibrated non-alert state parameters, the alert state parameters are further calibrated with the data set collected between 7:00 and 8:00am, during which it is believed to be in the combined states of alert and non-alert states.

3.7.2.1. Parameters extracted from field data

Desired Speed and Vehicle Composition

As discussed earlier in Chapter 2, the desired speed of this site can be extracted directly from the field data according to the speed, flow and vehicle composition information (section 2.2.2.2). Thus, the desired speed of car for this selected site is given as being normally distributed as N(109.2, 9.3^2) km/h, and the desired speed of HGV is normally distributed as N(91.8, 14.5^2) km/h (Table 2-4). The composition of HGV can also be extracted from the field data: HGV percentage 13% between 6:00 and 7:00, 7% between 7:00 and 8:00, 8% between 8:00 and 9:00.

Effects of Lane-changing

As the selected site has no connection to slip roads and there was no roadwork in the vicinity during the observation period, it is assumed that there were no *mandatory lane-changing* behaviours due to traffic leaving/entering the motorway. *Discretionary lane-changes,* which are not considered here, are the lane-changing behaviours where vehicles change lanes in order to obtain speed advantage (TRB, 1997). According to Yousif and Hunt (1995), there are few discretionary lane-changes under free-flow conditions due to the limited demand for lane-changes. As the traffic flow increases, there is an increase in the number of lane-changes since fast moving vehicles try to avoid travelling behind slow-moving vehicles. Yousif &

Hunt (1995) also found that with further flow demand for lane changing increases, but the opportunities decrease, resulting in a lower number of actual lane-changes.

As shown in Figure 3-36, it is found that during the study period between 7:00 and 8:00 in the morning, the speed of lane1 and lane 2 are almost at the same level (during the period, the traffic was quite congested). Thus, it is assumed here with no speed advantage to change to offside lanes (from lane 1 to lane 2) during this period. As for the period between 6:00 and 7:00, the traffic speeds are generally above 60 km/h (Figure 3-35), and it takes no more than 1 minute to drive through the 1100 m section of motorway. Therefore, the lane-changing behaviour is assumed to be rare and is not considered in the model calibration here.

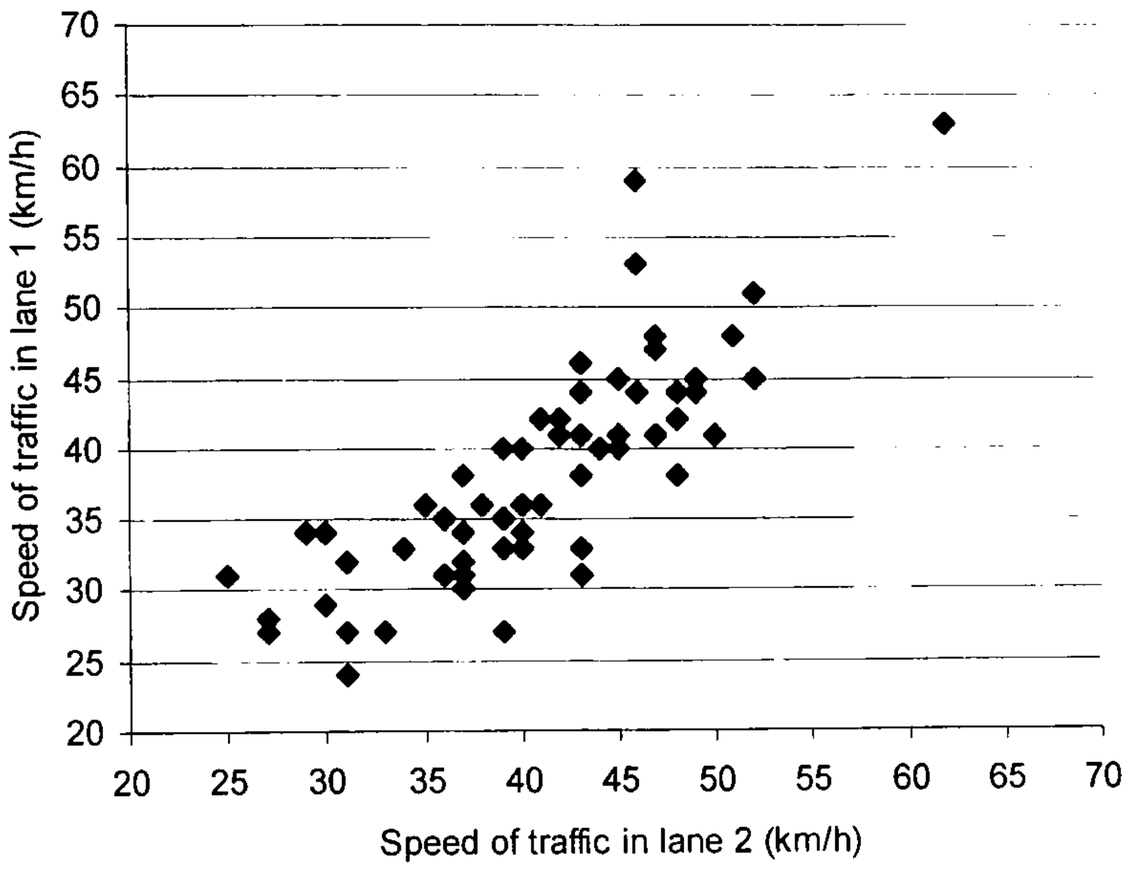


Figure 3-36 Lane1 speed versus Lane 2 speed collected during 7:00 to 8:00 in the morning from detector 4811A, 28 February 2001.

3.7.2.2. Calibration method

As introduced earlier (Section 3.6.4), to calibrate a car-following model on an open highway, at least three detectors are required along the study section; a schematic drawing of the locations of the three detectors from up- to down-stream is shown as detector A, B and C in Figure 3-37. The data from the upstream detector (A in Figure 3-37) is used for generating input traffic; data from the detector located at the end of the section (C) will be used to constrain the out-flow traffic; whilst data collected from the loop detector in the middle is used for model calibration and validation.

The simulated network will then include the section of the highway from A to C, plus a "buffer area" shown by the area bounded by the dashed lines. In the

simulation model, traffic arrives at the upstream end of the section randomly based on the dynamic speed v_T^A and flow q_T^A profiles collected from detector A. Traffic is also generated to start at point C based on the dynamic speed and flow profiles from detector C (v_T^C and q_T^C) and travel through the buffer area before exiting the network. The rationale for having a buffer area is to ensure the vehicles driving at the end of the section still have leaders in front, so as to create a constraint for the "follow-the-lead" rule for the traffic entering from A. Without such constraint, traffic in the section will have an "open-running" when they will all drive their desired speeds by the time they reach the end of the section. The flow and speed collected at the middle detection B will then be used to compare to the real data collected from the corresponding detector. This design for the model calibration is also employed by Brockfeld et al. (2005).

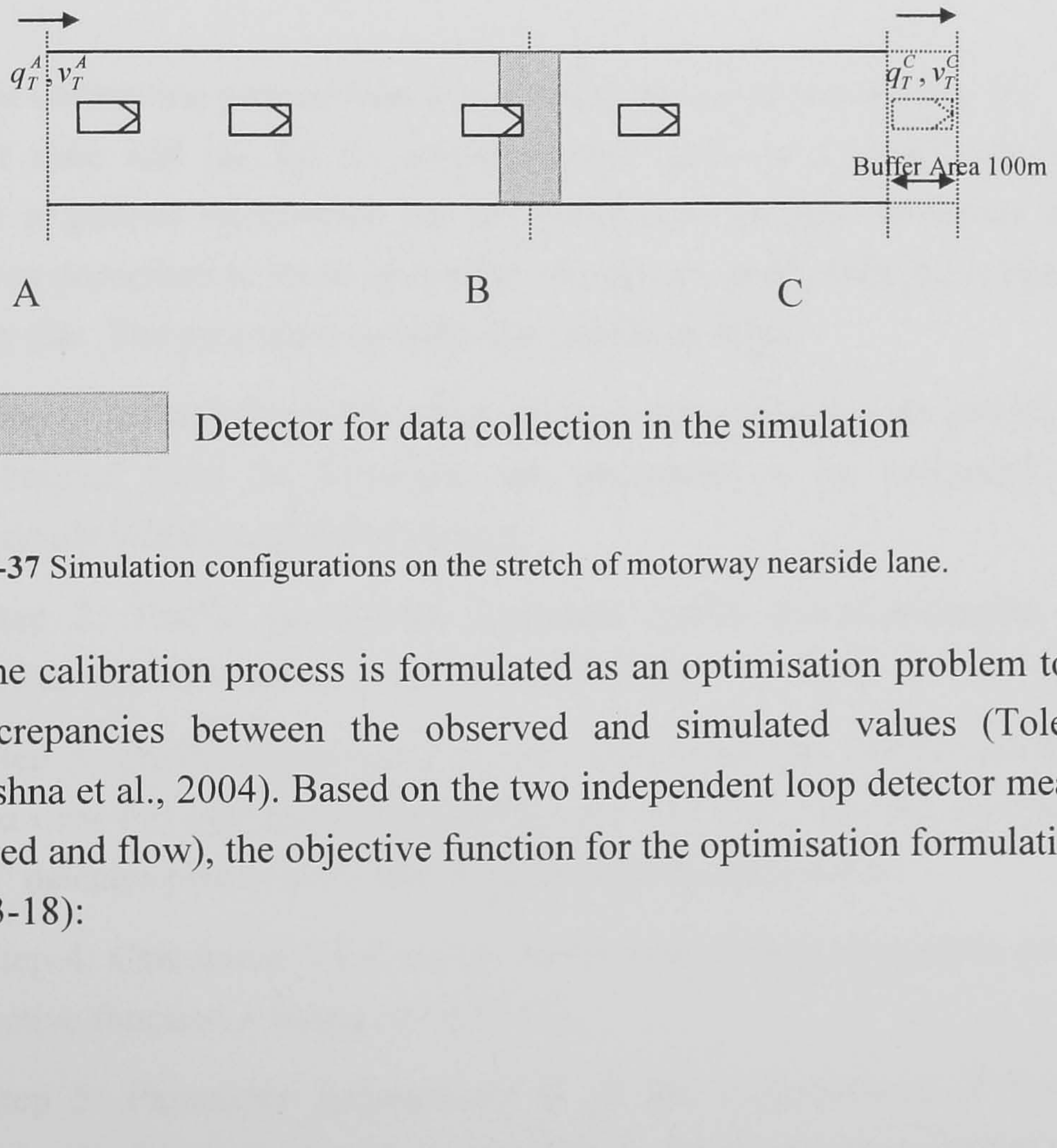


Figure 3-37 Simulation configurations on the stretch of motorway nearside lane.

The calibration process is formulated as an optimisation problem to minimise the discrepancies between the observed and simulated values (Toledo, 2003; Balakrishna et al., 2004). Based on the two independent loop detector measurements (i.e. speed and flow), the objective function for the optimisation formulation is given in eq. (3-18):

$$F_{\{\beta\}}(v,q) = \sum_{t=1}^{T}[(\frac{v_t^{sim} - v_t^{obs}}{v_t^{obs}})^2 + (\frac{q_t^{sim} - q_t^{obs}}{q_t^{obs}})^2] \quad (3\text{-}18)$$

where:

v_t^{sim} and v_t^{obs} the simulated and observed speeds, respectively during time period t;

q_t^{sim} and q_t^{obs} the simulated and observed flows, respectively during time period *t*;

$\{\beta\}$ the set of model parameters to be calibrated;

T the calibration period (1 hour in this study);

t the aggregate time interval (3 minutes in this study).

The calibration process here is to modify the model parameters $\{\beta\}$, i.e. (τ_1, A_1) for alert state and (τ_2, A_2) for non-alert state (refer to section 3.6). Figure 3-38 presents a general framework for the calibration process, which is an iterative simulation procedure to try to match the simulation results with those observed from the study site. The procedure includes the following steps:

[Step 1: Initialisation] The range of the tested values can be initially limited to those obtained from the literature; the parameters to be calibrated are varied systematically within such tested ranges;

[Step 2: Traffic generation] Upstream traffic and downstream traffic are generated as input data during each simulation run;

[Step 3: Traffic Simulation] In each simulation run, traffic speed and flows averaged over the aggregate time interval are collected from the detector from the "virtual" detector placed in middle of the modelled road section;

[Step 4: Calculation of F value] At the end of each simulation run, calculate the objective function F based on eq.(3-18).

[Step 5: Parameter Adjustment] If all the combinations of the calibrated parameter values have been tested, go to Step 6, else adjust the test parameter values, and go back to Step 1.

[Step 6: End of calibration] The best fit of the parameters is the set of values that has the minimum value of F, i.e. the minimum discrepancies between the simulated and observed data.

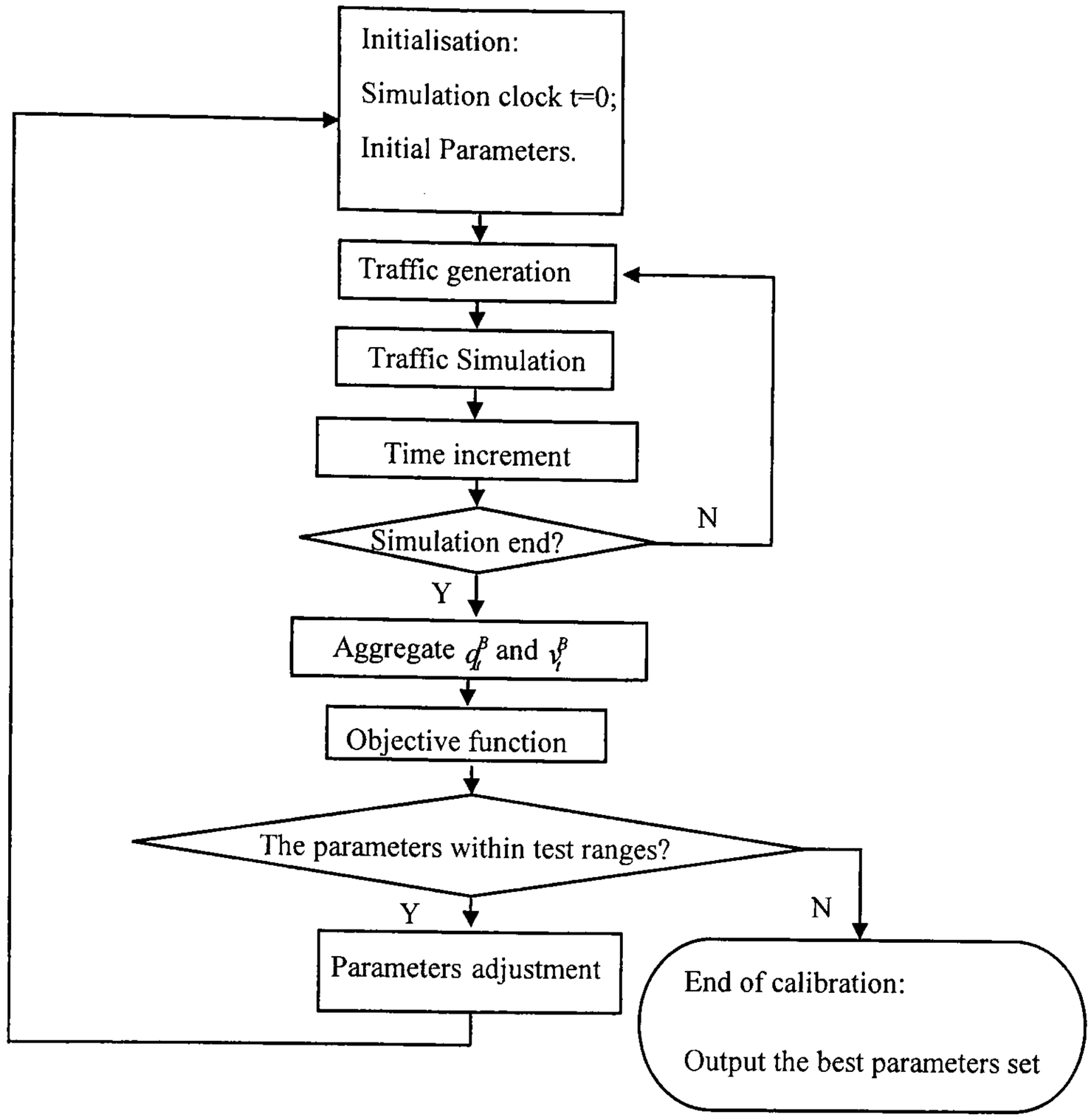


Figure 3-38 General calibration framework.

3.7.3. Calibration Results

The data considered in this analysis is the 3-minute aggregated loop detector data from the MIDAS system over a section of the M25 motorway in the UK. The same traffic speeds data (shown in Figure 3-35) and the aggregated traffic flows for the same section over the same time period are considered. Two sets of model parameters $\{\beta_1\}=\{A_1, \tau_1\}$ and$\{\beta_2\}=\{ A_2, \tau_2\}$, representing driving behaviour of the alert and non-alert states, will then be calibrated separately using traffic data representing the two situations.

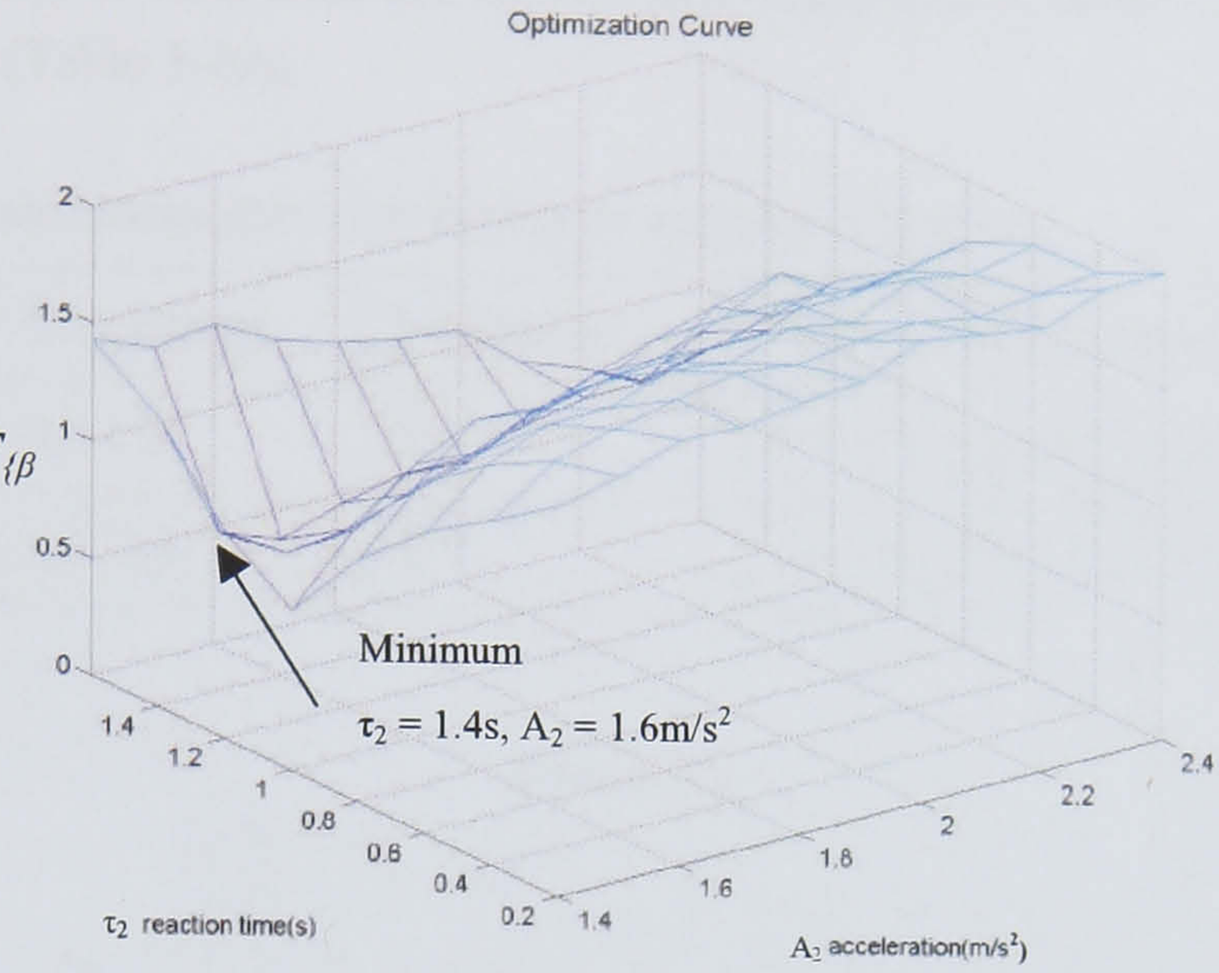
During traffic build-up period of 06:00-07:00, the travel speeds were above the critical threshold 50 km/h. Data from this period is used to calibrate the parameter set $\{\beta_2\}=\{ A_2, \tau_2\}$ for the non-alert state (refer to section 3.6.1).

Following the procedure described earlier, the model parameters to be calibrated are systematically varied over a pre-defined range of values obtained from literature. In the test, a wider range of values are studied in order to further examine the trend of the model's performance in the optimisation process (Table 3-18).

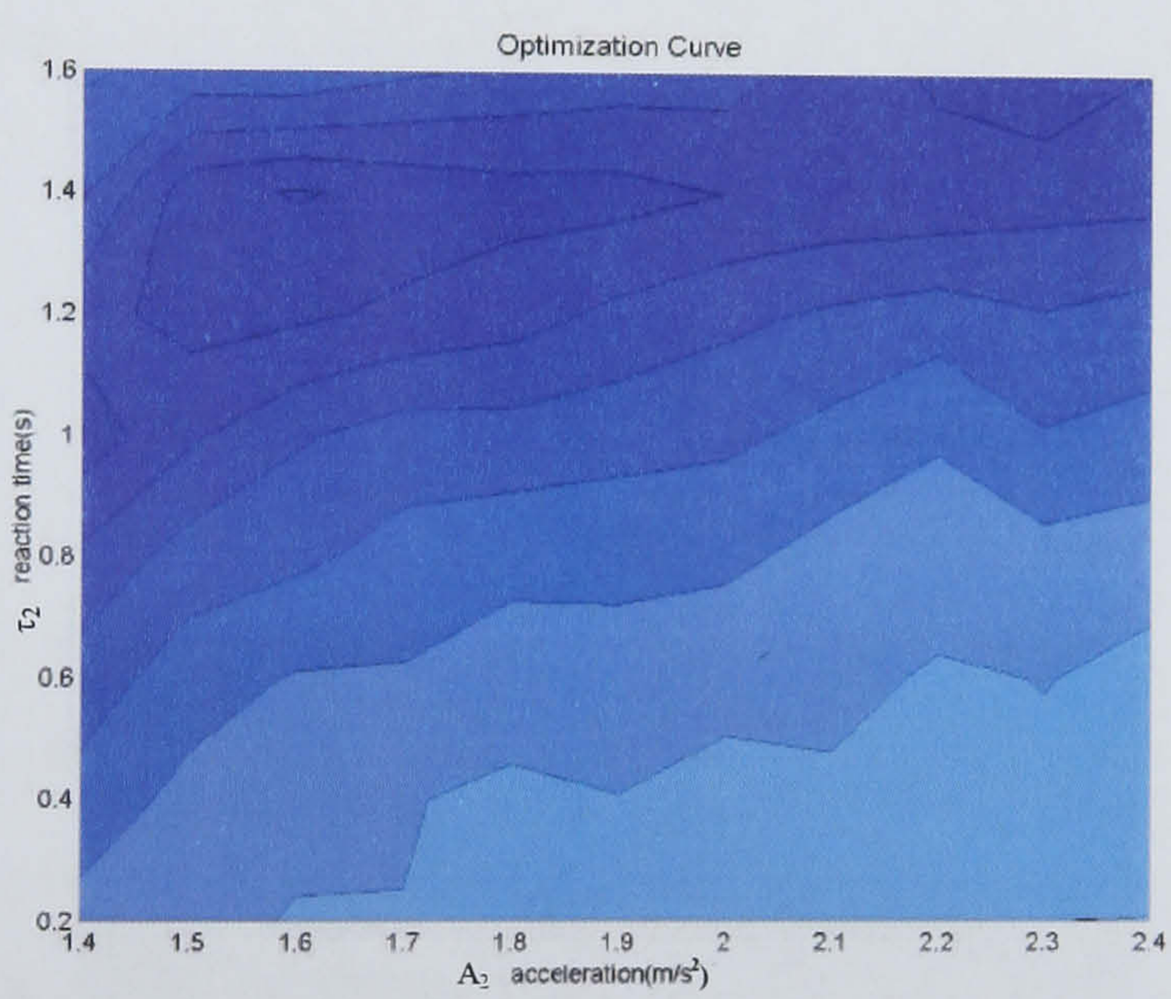
Table 3-18 Model parameters for non-alert state model calibration.

Parameters	Test Range	Increment	Values from literature
τ_2 (s)	0.2-1.6	0.2	0.6-1.4 (Toledo, 2003)
A_2 (m/s^2)	1.4-2.4	0.1	1.5-1.9 (ITE, 1999)

From iterative simulations within the selected test range (Table 3-18), the values calculated from the objective function eq. (3-23) are obtained and displayed in Figure 3-39. The optimisation curves are shown in both 3D (Figure 3-39 (a)) and contour plots (Figure 3-39 (b)). It is found (Figure 3-39 (a)) that with τ_2=1.4s, most of the $F_{\{\beta\}}$ values are varied within the range of 0.9- 1.2 (Figure 3-39 (a)), which is generally lower than those with other values of τ_2 (Figure 3-39 (b)). The optimal $F_{\{\beta\}}$ is 0.581 when τ_2 and A_2 are 1.4s and 1.6m/s^2 respectively.



(a)



(b)

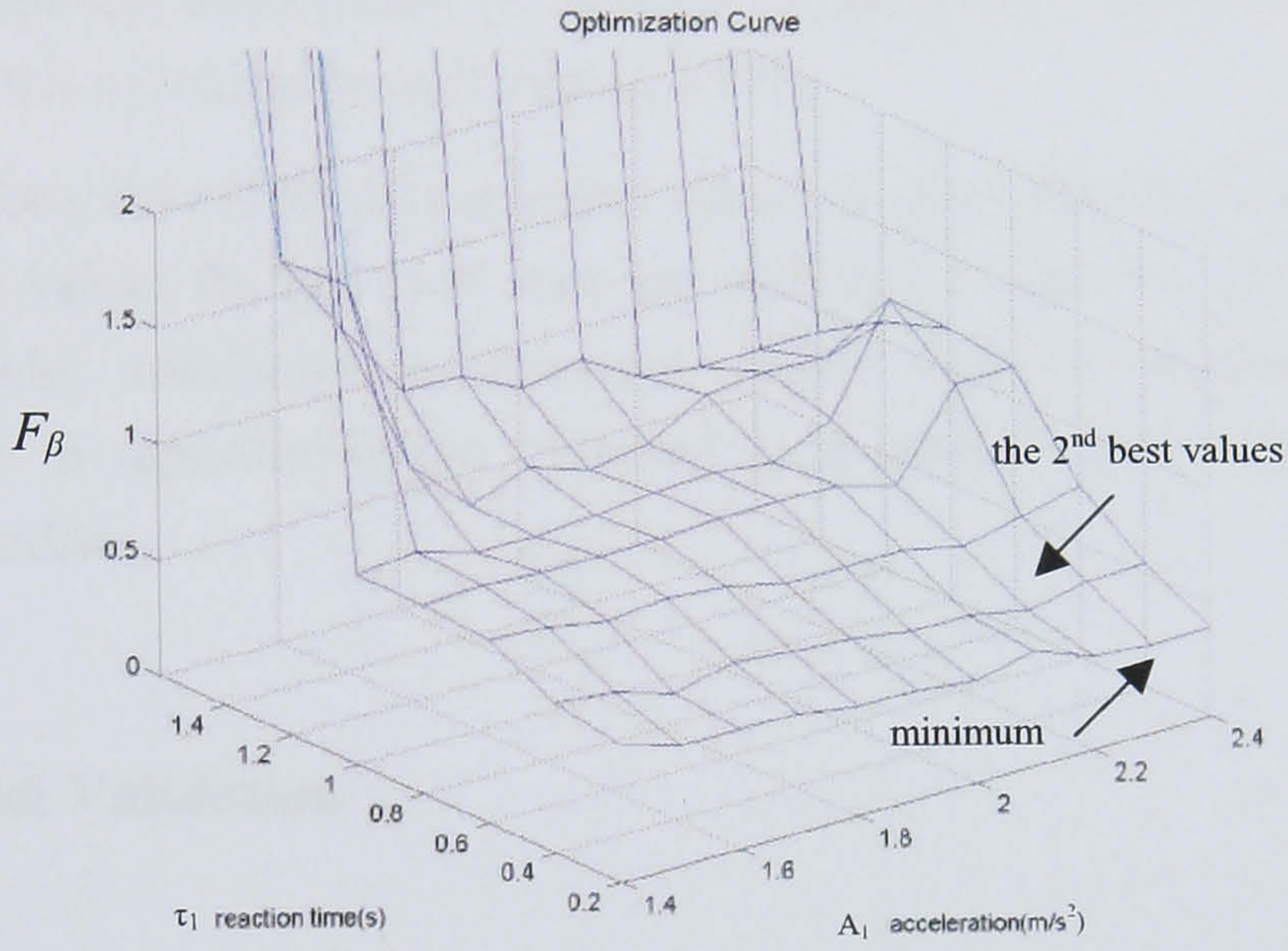
Figure 3-39 The 3D (a) and contour (b) plots of the optimisation curve with respect to the non-alert state parameters of τ_2 and A_2.

The observed data during 07:00-08:00 shows that the traffic speeds over the three detectors range from below to well above the critical speed (Figure 3-35), suggesting the site during this time period consists of both the alert and non-alert states. Given the above calibrated non-alert parameter values, data in this time period can then used to calibrate the parameters for the alert state $\{\beta_1\}=\{A_1, \tau_1\}$.

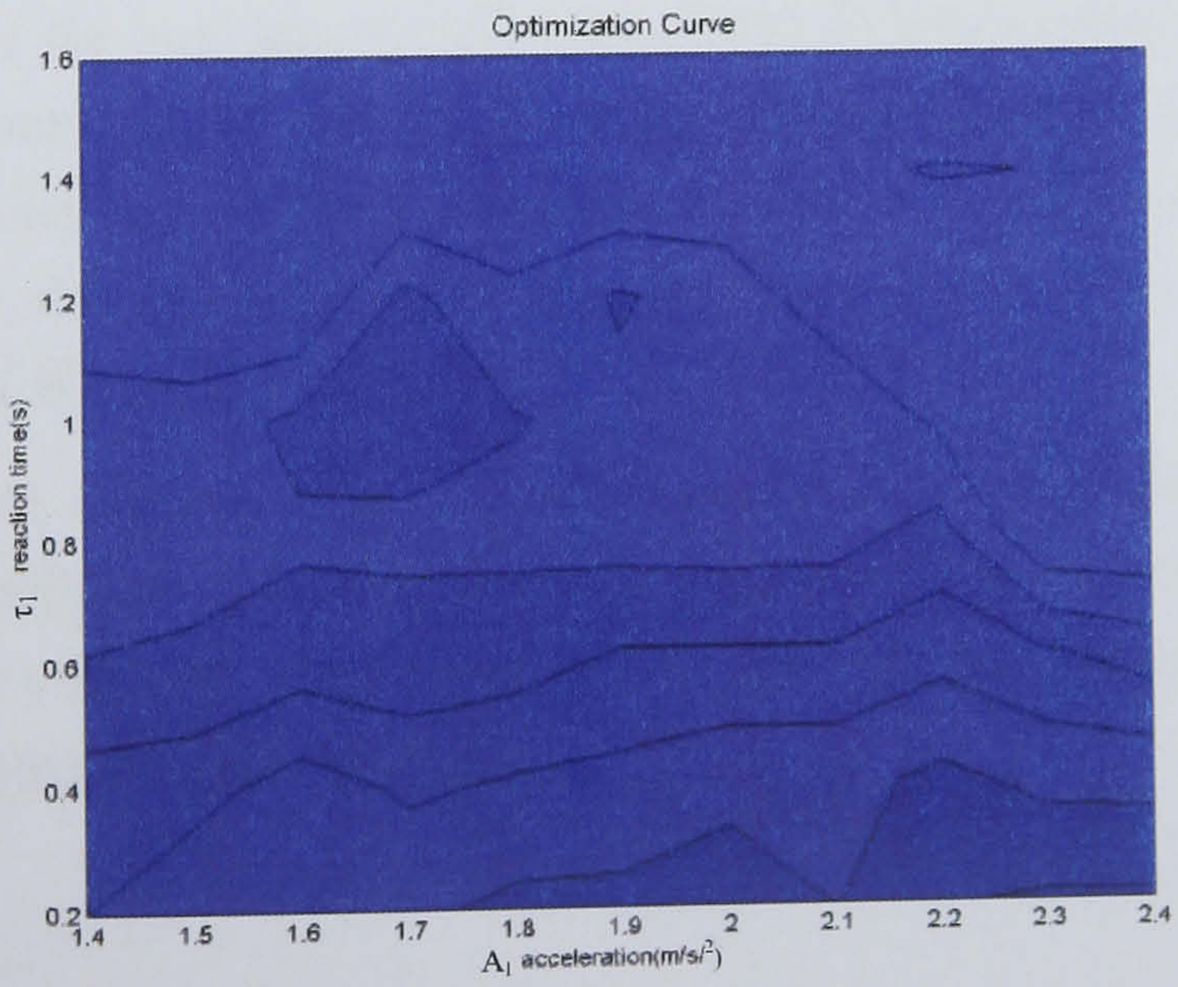
Similar to the calibration for non-alert state's parameters values, wider test ranges of parameters values in alert state are studied here compared to those values obtained from literature (Table 3-19).

Table 3-19 Model parameters for alert state model calibration.

Parameters	Test Range	Increment	Values from literature
τ_1 (s)	0.2-1.6	0.2	0.4-0.8 (Toledo, 2003)
A_1 (m/s^2)	1.4-2.4	0.1	1.9-2.4 (ITE, 1999)



(a)



(b)

Figure 3-40 The 3D (a) and contour (b) plots of the optimisation curve with respect to the alert state parameters of τ_1 and A_1.

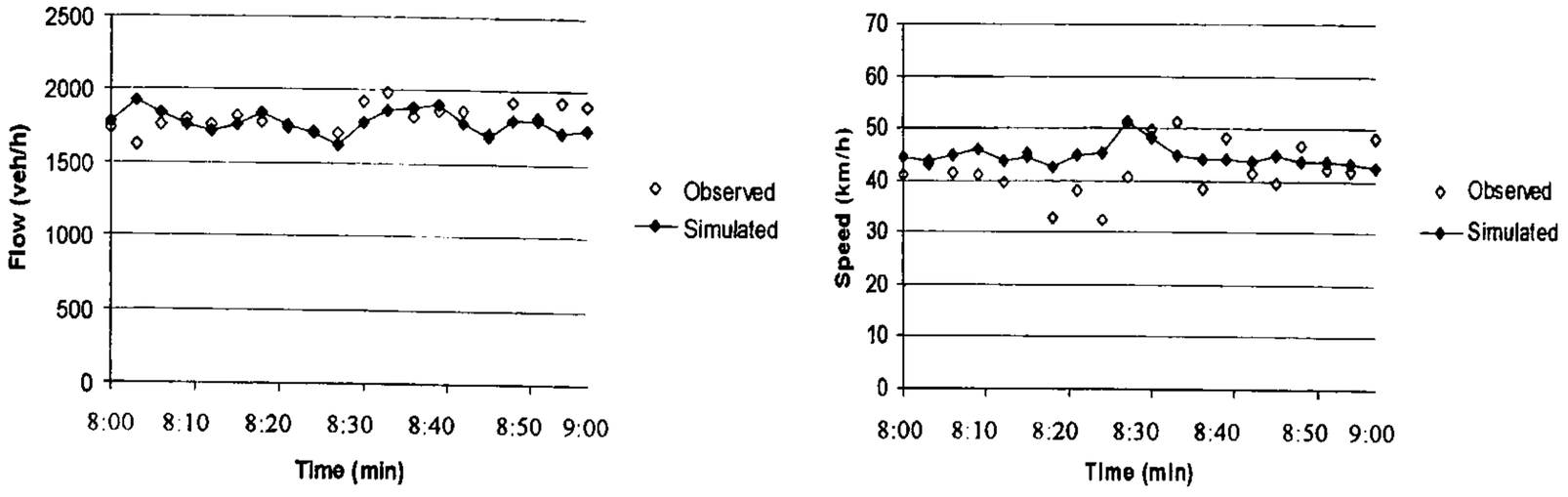
From iterative simulations within the selected test range (Table 3-19), it is found (Figure 3-40 (a) and (b)) that generally the smaller the reaction time of τ_1, the smaller the discrepancies between the simulated and the real data. For example, with τ_1=0.2s and 0.4s, the values calculated from the objective function eq. (3-23) vary from 0.4 to 0.6; with τ_1 as high as 1.6s, most of such values vary from 4 to 6 which are much higher than those with smaller reaction time of τ_1. Figure 3-40(a) and (b) truncated those curves with such values higher than 2 for the better display of other curves. It can be seen in Figure 3-40(a) that the minimum value calculated from eq. (3-23) is 0.388 which can be obtained with τ_1=0.2 s and A_1=2.3 m/s^2; the "second-best" value is 0.478 with τ_1=0.4 s and A_1=2.2 m/s^2. However the set τ_1=0.4 s and A_1=2.2 m/s^2 is selected as the calibrated parameters values because it fits in better with the empirical observation of driver's reaction time, which was found to be greater than 0.4 s (Johansson and Rumar, 1971).

Therefore, the calibrated parameter values for alert state are with τ_1 0.4s and A_1 2.2m/s^2; the values for non-alert state are with τ_2 1.4s and A_2 1.6m/s^2. In the next section, model validation is discussed which aims to examine the model's performance in reproducing the observed loop detector data with the calibrated parameters values.

3.7.4. Model Validation

"*The purpose of validation is to determine the extent to which the simulation model replicates the real system.*" (Toledo, 2003). In this section, it is performed by taking the optimal set of parameter values obtained from the model calibration (section 3.7.3) into the simulation of traffic conditions in another time period on the same site. Here, the data between the morning time period 8:00 and 9:00, collected on 28 Feb, 2001 is used for validation.

Figure 3-41 illustrates the compared speed and flow profiles of the simulated and observed variables. It is found that the simulation can reasonably reproduce the observed traffic with slight under-prediction in flow (Figure 3-41(a)) and slight over-prediction in speed (Figure 3-41(b)).



(a) (b)

Figure 3-41 The simulated (a) flow and (b) speed profiles versus observed measurements.

The extent of the validation result can be quantified using statistical goodness-of-fit measurements. Five measures have been found in the literature: root mean square percent error (RMSPE) (Toledo, 2003; Dowling et al., 2004)), mean percent error (MPE) (Chu et al., 2004), Theil's inequality coefficient (U), Theil's bias proportion (U^M), Theil's variance proportion (U^S) (Theil, 1961). Their equations are given in eqs. (3-19).

RMSPE measure penalises large errors at a higher rate than small errors, and MPE indicates the existence of systematic under- or over-prediction in the simulated measurements (Toledo, 2003). Recently, the measure of Theil's inequality coefficient (U) has been widely applied in the model calibration/validation in the transport area (e.g. Hourdakis et al., 2003; Barcelo and Casas, 2004). It should be mentioned that U (Inequality coefficient) methods have the advantage over traditional correlation coefficients for analysing the accuracy of forecasts. Traditional correlation coefficient is the correlation between a series of predictions and actual outcomes. The disadvantage is that "*perfect correction does not imply perfect forecasting. …… Whereas perfect forecasting requires, in addition to this, intercept=0 and correlation=1.*" (Theil, 1961). U value, however, U=0 implies a perfect fit (eq. 3-19(c)). Based on the inequality analysis, Theil (1961) proposed three Theil's error proportions- the bias (U^M), the variance (U^S) and the covariance (U^C) proportions.

$$RMSPE(\%) = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left(\frac{Y_n^{sim} - Y_n^{obs}}{Y_n^{obs}}\right)^2} \times 100 \qquad (3\text{-}19(a))$$

$$MPE(\%) = \frac{1}{N}\sum_{n=1}^{N}\left(\frac{Y_n^{sim} - Y_n^{obs}}{Y_n^{obs}}\right) \times 100 \qquad (3\text{-}19(b))$$

$$U = \frac{\sqrt{\frac{1}{N}\sum_{n=1}^{N}(Y_n^{sim} - Y_n^{obs})^2}}{\sqrt{\frac{1}{N}\sum_{n=1}^{N}(Y_n^{sim})^2} + \sqrt{\frac{1}{N}\sum_{n=1}^{N}(Y_n^{obs})^2}} \qquad \text{(3-19(c))}$$

$$U^M = \frac{(\overline{Y}^{sim} - \overline{Y}^{obs})^2}{\frac{1}{N}\sum_{n=1}^{N}(Y_n^{sim} - Y_n^{obs})^2} \qquad \text{(3-19(d))}$$

$$U^S = \frac{(S^{sim} - S^{obs})^2}{\frac{1}{N}\sum_{n=1}^{N}(Y_n^{sim} - Y_n^{obs})^2} \qquad \text{(3-19(e))}$$

$$U^C = \frac{2(1-\gamma)S^{sim}S^{obs}}{\frac{1}{N}\sum_{n=1}^{N}(Y_n^{sim} - Y_n^{obs})^2} \qquad \text{(3-19(f))}$$

where,

Y_n^{sim} and Y_n^{obs} are the simulated and observed 3-min aggregated measurements respectively,

N is the number of measurements during the 1-hour study period,

$\overline{Y}_n^{sim}$, $\overline{Y}_n^{obs}$, S^{sim} S^{obs} are the means and standard deviations of the series simulated and observed data respectively,

γ is the correlation coefficient between simulation and observed data.

For Theil's inequality coefficient (U), U=0 implies a perfect fit (i.e. $Y_n^{sim} = Y_n^{obs}$ for all *n*); U=1 implies the worst possible fit (i.e. there is either a negative proportionality, or one of the variables is identically zero). By definition, the three proportions U^M, U^S and U^C are all bounded between 0 and 1 and sum to 1 ($U^M + U^S + U^C = 1$) [8]. The bias proportion (U^M) reflects the systematic error ($U^M = 0$ implies no systematic error in the simulation); the variance proportion U^S indicates

---

8 Theil's inference (1961): $\frac{1}{n}\sum(Y^{sim} - Y^{obs})^2 = (\overline{Y}^{sim} - \overline{Y}^{obs})^2 + (S^{sim} - S^{obs})^2 + 2(1-\gamma)S^{sim}S^{obs}$. Hence, from eqs.3-19 (d), (e), (f) and Theil's inference, $U^M + U^S + U^C = 1$

the capability of the simulation model in replicating the variability in the observed data (U^S =0 implies a perfect prediction of the observed variability); the covariance proportion U^C measures the remaining error. "*It seems that we must draw the conclusion that, if the forecaster's ability does not allow him to attain perfection, the desirable distribution of inequality over the three sources is* $U^M = U^S = 0$, $U^C = 1$. (Theil, 1961)".

Table 3-20 lists the validation results of the new car-following model. The validation of this new car-following model gives a U-value of 0.031 for flow and 0.066 for speed. It should be noted that for flow and speed, both of the two U-values are under 0.1, indicating that the calibrated model can reasonably describe the observed data. "*It seems almost impossible to get better calibration results than 0.01 (of U- value)……,something around 0.14 to 0.16 is much more often found*" (Brockfeld et al., 2005). The bias (U^M) values of the new model are small (< 0.2) and comparable to the similar work by Toledo that was between 0.1 and 0.3 (2003). The variance U^S value of the new model for speed is 0.275 and for flow is 0.022 (<0.3, still close to 0), suggesting a good fit.

On average, the validation error MPE and RMSPE of the new car-following model is within the range of -1% to 15%, which is comparable to similar work by Toledo's results (varying between -3% to 12%). MPE values of the new car-following model (Table 3-20) indicates that flow is slightly under-predicted in the simulation with mean percent error around 1%; and the speed is over-predicted with mean percent error around 8%.

Table 3-20 Statistics for the flow and speed validation results.

Validation results	Flow	Speed
U	0.031	0.066
U^M	0.052	0.125
U^S	0.022	0.275
MPE (%)	-1.170	7.550
RMSPE (%)	6.200	15.400

3.7.5. Summary

The calibration of the car-following model for alert and non-alert states shows that the best fit of alert state parameters are τ_1=0.4s and A_1=2.2m/s^2, and of non-alert state are τ_2=1.4s and A_2=1.6m/s^2. Using the calibrated parameters, the model is validated against the observed data from a different time period to examine the similarity between the observed and simulated data. Five goodness-of-fit measures are used for the validation: Theil's inequality coefficient (U, U^M, U^S), root mean square percent error (RMSPE), mean percent error (MPE). The results show that the simulation can reasonably reproduce the observed measurements from loop detectors with small U values (<0.1).

3.8. CONCLUSION

The phenomena of traffic breakdown, hysteresis, shockwave propagation and close-following are some of the key motorway flow characteristics. To mimic these characteristics, a new car-following model is developed which makes a distinction between three different states of driving behaviour, i.e. non-alert, alert and close-following states. The model combines and modifies two existing models of vehicle following: the safety distance car-following model (Gipps, 1981) and the action-points model for close-following situation (Brackstone et al., 2002). The mathematical formulation and the theoretical analysis of the new model are presented. Micro- and macro-scopic properties of the model are analysed and compared with some of the existing models and with real observations on a UK motorway.

Simulation tests are conducted and the results suggest that the model responds well to the changes in the parameter values of the alert reaction time and non-alert reaction time. The results show that the smaller alert reaction time results in smaller acceptable spacings among drivers and a higher capacity achieved. By using different alert and non-alert reaction times, the model can reproduce traffic hysteresis. Microscopic analysis of the modelled results indicates that the model can reproduce speed drop, shockwave propagation as well as close-following behaviour.

The advantage of this model is that it tries to represent the different driver responses in different traffic conditions. Whilst it presents a challenge in terms of model calibration, it has relatively few parameters to calibrate and it represents a

mechanism to reproduce some of the key characteristics of motorway traffic. For example, it has been shown that traffic hysteresis can be readily replicated by using different reaction times for the different states in the model.

A methodology for a general calibration and validation framework is proposed which attempts to rectify the model with the most commonly available traffic surveillance data, the loop detector data. Using the loop detector data, selected parameters of a car-following model can be calibrated and validated with aggregate speed and flow measurements (aggregated over a time interval of 3 minutes). The framework has been demonstrated through an example to calibrate a newly developed car-following model as given earlier in this chapter. After calibration, the model can reasonably reproduce the observed measurements from loop detectors with small U values (<0.1), which indicate a good fit between the simulated and observed data.

With modern data collection methods such as digital video analysis and driving simulator experiments, it is feasible to directly calibrate the model parameters. Serious progress has recently been made in this area, gathering field data for testing alternative theories (e.g. Brackstone et al., 2002; Chakroborty and Kikuchi, 1999; Rakha and Crowther, 2003; Wu et al., 2003; Bham and Benekohal, 2004). This will also be one of the important priority areas for future research.

CHAPTER 4
A SIMULATION MODEL FOR MOTORWAY MERGING BEHAVIOUR

The most difficult sections of motorway to analyse satisfactorily are the merging sections, because these are where the majority of conflicts and interactions occur. The situation is more complicated than simple gap acceptance because the ramp traffic is not only seeking gaps in the main motorway traffic; its presence also influences the behaviour of drivers on the main motorway. The latter may choose to move to the inner lanes on the motorway or to slow down to create gaps for the merging traffic. Traditional modelling is unable to represent such complex behavioural responses and interactions.

In this chapter, a description of the interactions in a merging process is firstly discussed. It is then followed by a literature review of the existing studies in representing merging behaviours. A new model of motorway traffic merging behaviour is proposed here by explicitly simulating the interactions between the gap-acceptance behaviour of the merging traffic and the cooperative behaviour of the motorway traffic. The chapter also presents the model formulation, sensitivity analysis of some of the key model parameters, initial simulation tests on a standard merging section, and their comparison with observations.

4.1. BACKGROUND

Motorway merging has long been regarded as a major source of conflicts and congestion on motorways (Hounsell and McDonald, 1992; Evans et al., 2001). Traditionally, merging behaviour has been modelled based on the analysis of the headway distributions of the motorway traffic and gap acceptance by the merging traffic (e.g. Drew, 1967; Darzentas, 1981). However, for motorway merging, there is usually an acceleration lane running in parallel with the motorway along which vehicles can merge into the motorway traffic, i.e. there is continuous gap-acceptance and acceleration behaviour over the entire length of the acceleration lane (Michaels and Fazio, 1989; Zheng, 2002).

Obviously, a merging operation involves two interactive traffic streams: the ramp traffic and the motorway traffic. Most existing models simplify the resulting complex dynamic interactive behaviour by assuming that merging traffic has no influence on the motorway traffic (e.g. Michaels and Fazio, 1989; Yang and Koutsopoulos, 1996). However motorway traffic has in fact been seen to exhibit a kind of cooperative behaviour by changing to the inner lanes upon approaching the merge section or by yielding to create gaps for the merging traffic (Hounsell and McDonald, 1992; Sarvi et al., 2002; Elefteriadou et al. 1995; Evans et al., 2001; Troutbeck, 2002; Bunker and Troutbeck, 2003). Such a cooperative process, which is also called "limited priority system", has recently received some attention (Wang et al., 2005b). It may seem to help the joining traffic to merge, therefore reducing conflicts and increasing capacities.

4.2. INTERACTIONS IN A MERGING PROCESS

The behaviour at motorway merges referred to in the previous section is summarised in Figure 4-1. The driver of a merging vehicle takes into consideration two *surrounding factors*- the nearside motorway traffic actions and the remaining distance to the end of the merging lane and decides to accelerate or decelerate to maximise the chances of available gaps.

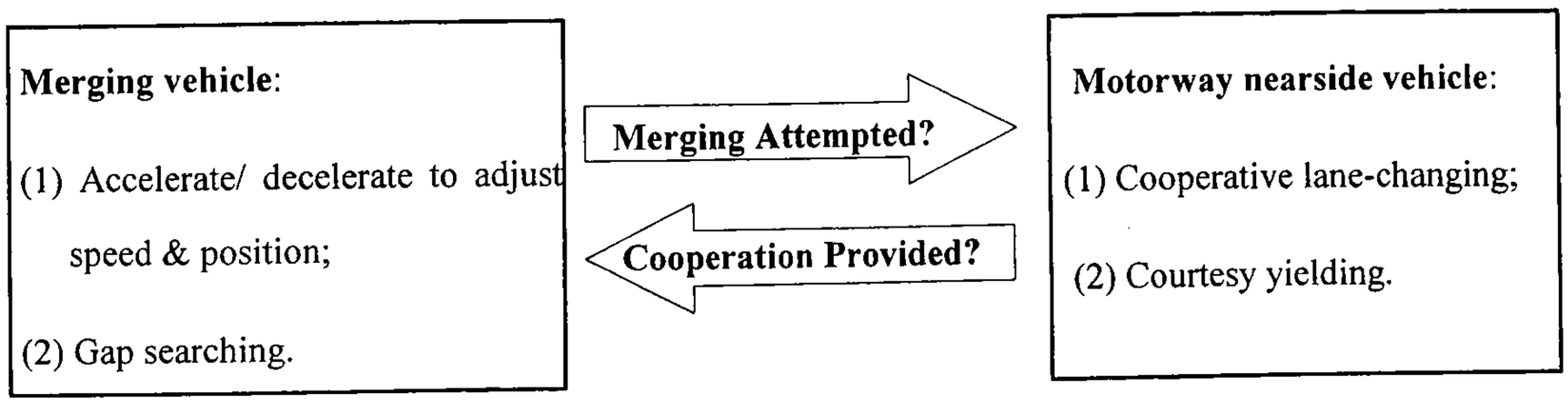


Figure 4-1 The interactive behaviours between the merging traffic and the nearside motorway traffic.

Aware of the arriving merging traffic, the drivers in the motorway nearside lane may choose to ignore the merging traffic by continuing with their desired movements, or they may slow down to create gaps for the merging vehicle, or they may even change lane on the motorway to allow the merging traffic to join. The

latter two behaviours are termed *courtesy yielding* and *cooperative lane-changing* respectively. It is clear that both courtesy yielding and cooperative lane-changing are reactions to the ramp traffic; ignoring these reactions will lead to an inaccurate description of the traffic phenomena and poor evaluation of the merging operations (Kita et al., 2002; Zheng, 2002). A new, simulation-based model of motorway merging behaviour has therefore been developed, aimed at capturing all the merging behaviours involved.

Before the description of the merging process, a typical merging area is illustrated in Figure 4-1, where a merging vehicle (**C**) interacts with its putative leader (**PL**) and the putative follower (**PF**) on the nearside motorway lane. The merging vehicle will examine the *original gap* between PL and PF (the first motorway gap to be faced by C when it arrives at the acceleration lane as shown in Figure 4-2), the *previous gap* in front of PL, and the *following gap* behind PF. A PL (or PF) exists if the lead (or the lag) gap is less than 5 seconds (May, 1990).

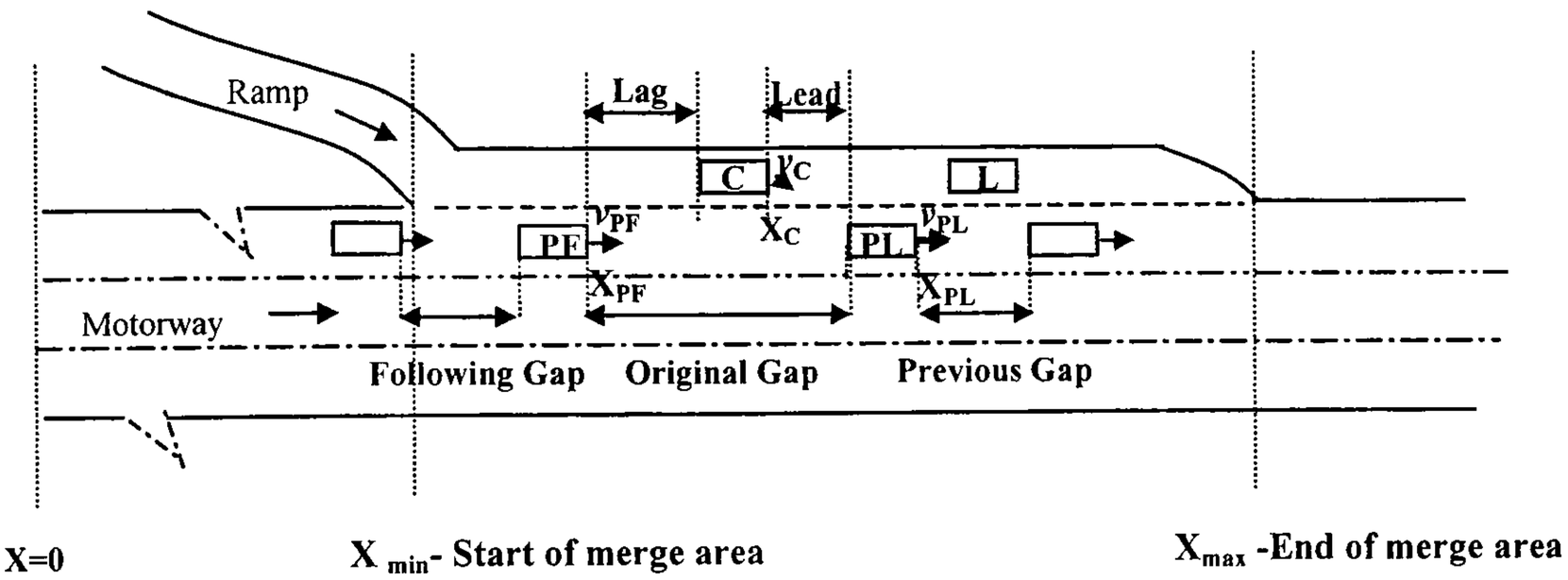


Figure 4-2 A schematic of a merge area. The merge starts from an arbitrary point upstream of the merge area, denoted as X=0; all positions are measured relative to this point.

4.3. LITERATURE REVIEW OF THE EXISTING STUDIES

A detailed understanding of previous related research is important as it provides opportunities for finding and later solving problems in the study of merging sections. The existing studies regarded motorway merging as one kind of mandatory lane-changing, thus few studies concentrate specially on the motorway merging section. However, as earlier discussed in 4.1 and 4.2, it should be noted that the unique acceleration properties on the acceleration lane of the merging traffic are

different from those of other mandatory lane-changing behaviours such as caused by lane-blockage or accident, e.g. the studies by Hunt and Yousif, (1990). Some studies analyse the cooperative behaviours provided from the nearside motorway traffic, e.g. Bunker and Troutbeck (2003) and Elefteriadou et al. (1995). However, their major interests are to analyse such behaviours effects on traffic operation at macro- level, i.e. traffic delays, merging capacity and traffic breakdowns. Few studies dealt with modelling the fundamental mechanism of the interactions between the ramp merging traffic and the nearside motorway traffic in the merging sections (Kita et al., 2002), which may cause an inaccurate analysis of the traffic phenomenon on a merging section.

The existing studies in modelling the motorway on-ramp merging process can be divided into the following categories: Braking Risk model, Probability Function model, Perceptual model, Fuzzy Logic approach and Game Theoretic approach. The following subsections describe the characteristics of each model category and Table 4-1 summarises the model characteristics with the corresponding algorithms.

4.3.1. Braking Risk Model

The main idea is that the urgency of the mandatory lane-changing manoeuvre is considered in terms of the distance to the intended turn of the driver, such as the position of the blockage of a lane. That is, a driver is prepared to accept a higher risk of deceleration (i.e. greater chance of deceleration) for cutting into smaller gaps when he is getting close to the intended turn. The first prototype of this category was proposed by Gipps (1986) and this category of models is widely used in traffic micro-simulation packages, such as FRESIM (TRB, 1997).

Gipps (1986) proposed a framework for the structure of lane-changing decision in urban driving situations including the influence of traffic signals, obstructions, etc. Gipps assumed that a lane-changing manoeuvre takes place without interference with vehicles in the destination lane. The decision of mandatory lane-changing is based on two factors in consideration of urgency effect : (1) Whether it is physically possible and safe to change lanes without an unacceptable risk of collision; (2)The location of permanent obstruction. The braking risk is defined as follows:

$$b_C = [2 - (x_{\max} - x_C)/10v_f]b_C^* \tag{4-1}$$

where, b_C is the maximum acceptable braking rate (braking risk) at current position (m/s^2) (negative value); v_f is the desired speed of the vehicle preparing to

cut in (m/s); b_C* is the average deceleration a vehicle is willing to accept in lane changing (m/s^2).

The model of lane-changing process was designed to be used in conjunction with a car-following model (Gipps, 1981) to calculate a safe speed with respect to the putative leader running on the nearside motorway, given in eq.(4-2):

$$v_c(t+T) = b_c T + \sqrt{b_c^2 T^2 - b_c\{2[x_{PL}(t) - L_{PL} - x_c(t)] - v_c(t)T - v_{PL}^2(t)/b'\}} \qquad (4\text{-}2)$$

where, b' is the most severe braking that the driver of putative leader wishes to undertake, estimated by the merging vehicle's driver (m/s^2); T is the reaction time (s).

The judgement as to whether lane-changing is feasible or not is based on the comparison: if the required deceleration (i.e. $(v_c(t+T)-v_c(t))/T$) is unacceptable to the merging vehicle ($(|v_c(t+T)-v_c(t))/T| > |b_c|$), the merging is not feasible. In this literature no information was found dealing with "cooperative lane-changing" or "courtesy yielding" situations.

Subsequently, Hunt and Yousif (1990) developed a micro-simulation model for the merging behaviour at roadworks rather than motorway on-ramp merging situation. The rules were based on similar logic to that described by Gipps (1986); no detailed equations were listed in this work. For the lane-changing behaviour, it considers the urgency effects of mandatory lane-changing: "*As the driver feels more psychological pressure to move towards a closed lane he is willing to modify his threshold gap size requirement, accepting greater risk and using more severe deceleration rates*." It introduced a random generated parameter of courtesy yielding for accepting merging vehicle that cut in.

Hidas (2002) provided a multi-agent simulation system SITRAS in which the logic and algorithms are based on Gipps model with added consideration of "forced merging" and "courtesy yielding". In order to simulate aggressive drivers, Hidas introduced the 'driver aggression parameter' to Gipps' urgency algorithm that was given earlier in eq. (4-1). This brings an arguable aspect to this model, that is, for aggressive drivers, the calculated most severe braking could be far too high (up to 99 times the value calculated from Gipps model) to be technically feasible.

Probably the best freeway simulation program available, FRESIM (TRB, 1997) provides the logic of mandatory lane changing and pre-emptive lane-changing in the

merging area. In mandatory lane changing, it considers the urgency effects similar to Gipps in terms of the distance to the end of the merging point. In FRESIM pre-emptive lane-changing refers to the lane changes that are performed by motorway nearside moving vehicles near a merging area to avoid potential disturbances from the merging traffic.

All these above-mentioned models consider the urgency effects on the driver to execute the merging manoeuvre, i.e. accept higher braking risk with the resultant smaller acceptable gaps. Hunt and Yousif (1990) and Hidas (2002) consider the cooperation of "forced merging" and/or "courtesy yielding" in the modelling, however without mentioning the effect of "cooperative lane-changing" from the nearside motorway traffic during the merging process. There was no mention of courtesy yielding behaviour on the motorway in FRESIM.

In the above literature on braking risk modes, no detailed descriptions were found of the modelling of acceleration/deceleration and gap searching process.

4.3.2. Probability Function Model

An example of such a model is MITSIM, which describes mandatory lane-changing in a probabilistic manner in combination with a gap acceptance model (Yang and Koutsopoulos, 1996).

A driver will start the lane change at a distance from the downstream node (e.g. lane drop or incident) with probability (using symbols of Figure 4-1):

$$p_c = \begin{cases} \exp\left[-(x_c - x_{\max})^2 / \sigma^2\right], & x_c < x_{\max} \\ 1, & x_c \geq x_{\max} \end{cases} \qquad (4\text{-}3)$$

where, P_c is the probability that merging vehicle starts a mandatory lane change manoeuvre, and σ^2 is a variable.

In the case of mandatory lane-changing, Yang and Koutsopoulos (1996) assumed that drivers tend to accept smaller gaps as they get closer to the last location where the lane change has to take place, as given in the following form (the varied critical gap function is also shown schematically in Figure 4-3),

$$\tilde{g}_c^i = \tilde{\varepsilon}_c^i + \begin{cases} \tilde{g}_{\max}^i & x_c \le x_{\min} \\ \tilde{g}_{\min}^i + (\tilde{g}_{\max}^i - \tilde{g}_{\min}^i)\dfrac{(x_c - x_{\min})}{(x_{\max} - x_{\min})} & x_{\min} < x_c < x_{\max} \\ \tilde{g}_{\min}^i & x_c \ge x_{\max} \end{cases} \qquad (4\text{-}4)$$

where, i is an index indicating whether the parameter is for the lead or lag gap; $\tilde{g}_c^i$ is the minimum gap a merging vehicle accepts for a mandatory lane change; $\tilde{g}_{\min}^i$ and $\tilde{g}_{\max}^i$ are respectively lower and upper bounds; $\tilde{\varepsilon}_c^i$ is an error term.

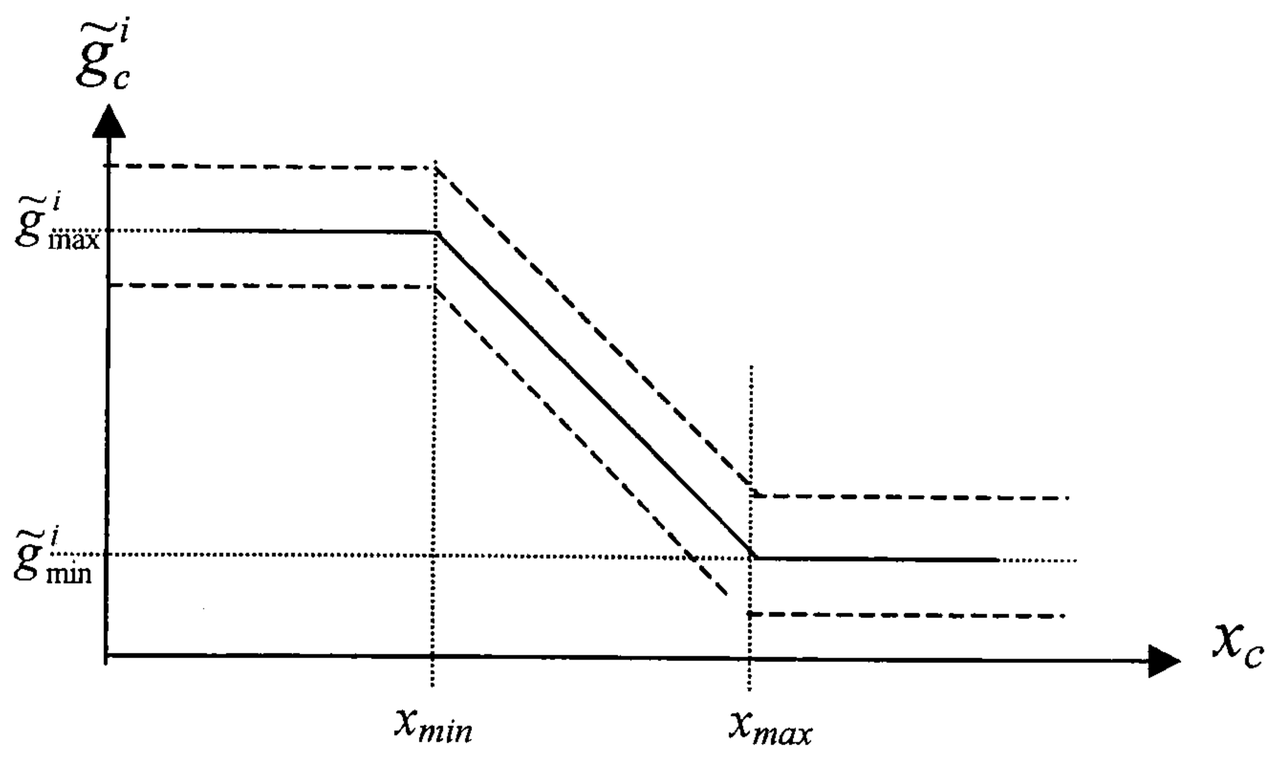


Figure 4-3 Critical gap function proposed by Yang and Loutsopoulos (1996). The broken lines denote the scope of the variations of the critical gaps caused by the error term $\tilde{\varepsilon}_c^i$.

Yang and Koutsopoulos (1996) classified the merging from motorway ramps as "priority-based merging", i.e. merging traffic can execute the merge only if the motorway gaps are acceptable with no consideration of the influence of the nearside motorway traffic's cooperation during the merging process. This model determines the decision to merge, is based solely on the combination of the gap acceptance model and the distance to end point of the merging section.

Ahmed (1999) modelled the merging behaviour in the situation of merging in heavily congested traffic- "*the probability of finding acceptable gaps is very low and in order to merge gaps have to be created*". "*Drivers must create gaps either through force or through courtesy yielding*". In his work, the forced merging situation is modelled by using a binary logit model, which is based on the relative speed to PL, PF, remaining distance to the merging end, gap size, etc. However, this literature did not deal with cooperative lane-changing and the acceleration/deceleration process during the merging manoeuvre.

Kita (1993) described merging behaviour as a binary choice. A binary logit model was proposed involving several variables that would have influence on the driver's judgement of merging (two choice alternatives of the model "accept" and "reject"). The explanatory variables included in the model are gap length, remaining distance of the acceleration lane and the relative speed difference between the merging vehicle and motorway traffic. This literature did not include the modelling of the acceleration control process or the cooperation effects on the gap acceptance behaviour.

4.3.3. Perceptual Model

Michaels and Fazio (1989) proposed a concept of 'angular velocity' based on the relative distance and speed between the merging and upcoming motorway nearside vehicles. Their paper briefly described the rule as acting in the merging model as a threshold for a ramp merging when a driver is considering whether merge into the motorway. The formulation is given as follows (using the symbols of Figure 4-2):

$$w_1 = k(v_c - v_{PL})/(x_{PL} - x_c - L_{PL})^2 \quad (4\text{-}5a)$$

$$w_2 = k(v_{PF} - v_c)/(x_c - x_{PF} - L_c)^2 \quad (4\text{-}5b)$$

where, w_1 *and* w_2 are the values of the angular velocity (rad/sec) related to its PL and PF.

The value could have three properties (these conditions are illustrated in Figure 4-4),

- Positive value- a closing situation
- Negative value- an opening situation
- Null value- free to merge (angular velocity below thresholds)

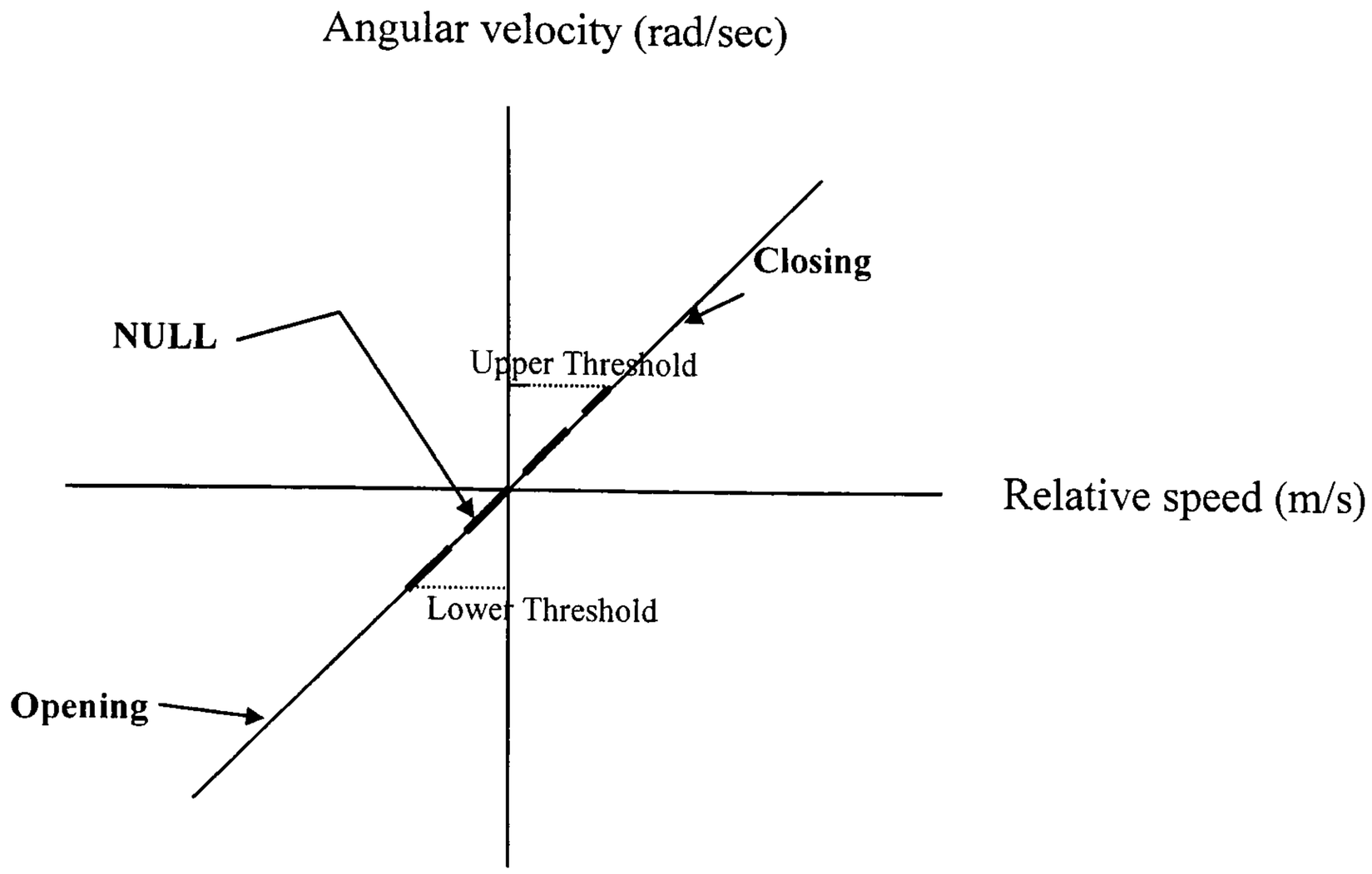


Figure 4-4 Illustration of angular velocity threshold (Michaels and Fazio, 1989).

Michaels and Fazio (1989) observed that in most cases the merging was executed below the threshold with the nominal value 0.004 rad/sec. Similar observations (Zheng, 2002) were found: the average threshold to the PL was 0.0085 rad/sec and the average threshold to the PF was 0.0045 rad/sec. This model is mainly focused on the safety threshold for a merging motion, that is, only safety effects of the motorway traffic on the merging vehicle are considered. There are no findings on the urgency effects, cooperative yielding or lane-changing in this literature. No description about the acceleration and gap searching process on the acceleration lane are mentioned either.

VISSIM (PTV, 2003), the microscopic/stochastic traffic simulator applies a similar perceptual model based on the work of Leutzbach and Wiedemann (1986). In VISSIM considers the forced lane-changing behaviour from ramp drives, however it does not include cooperative lane-changing, courtesy yielding from the nearside motorway and active gap searching process on the acceleration lane. The on-ramp merges only "*work well with small or moderate flows... It fails in representing heavy traffic merges from on-ramp, where it produces a large queue*" (Gomes et al., 2004).

4.3.4. Other Approaches

Zheng (2002) proposed a fuzzy logic model to represent the acceleration control behaviour of a merging vehicle running on the acceleration lane (eq. (4-6)) and the calibrated gap acceptance model (eq. (4-7)) with critical gap thresholds based on the speed difference to the motorway vehicles PL and PF. However, courtesy yielding was not considered in the modelling of the interactions in the merging process. "*A motorway gap follower may signal a merging vehicle to let him/her into the stream of traffic, but this is not considered here*" (Zheng, 2002).

$$v_c(t+T)=FUZZY\ [(v_{PL}(t)-v_c(t)/x_{PL}(t)-x_c(t),v_{PF}(t)-v_c(t)/x_{PF}(t)-x_c(t)] \quad (4\text{-}6a)$$

$$v_{PF}(t+T)=FUZZY\ [(v_{PL}(t)-v_{PF}(t)/x_{PL}(t)-x_{PF}(t),v_c(t)-v_{PF}(t)/x_c(t)-x_{PF}(t)] \quad (4\text{-}6b)$$

where, FUZZY denotes a fuzzy logic mapping expressed as a fuzzy inference system, other variables are the same as in Figure 4-1.

For the gap acceptance behaviour in motion, a merging driver must keep a safe distance to both PL and PF if he/she wants to accept the gap. The thresholds based on the distance separation (m) which are the functions of the speed of merging vehicle and the relative speeds (m/s) to PL and PF:

$$\bar{g}_{lead} = 6.7553\ e^{-0.2684\ (v_{PL}-v_c)} \qquad v_{PL}\text{-}v_c < 0 \qquad (4\text{-}7a)$$

$$\bar{g}_{lag} = 5.3472\ e^{0.5104\ (v_{PF}-v_c)} \qquad v_{PF}\text{-}v_c > 0 \qquad (4\text{-}7b)$$

where, $\bar{g}_i$ is the minimum lead and/or lag thresholds for merging vehicle and i is an index indicating whether the parameter is for the lead or lag gap.

Kita et al. (2002) proposed a game theoretic approach mainly dealing with the interactions during the merging process. In this model, the merging situation was described as a "game", in which each of the drivers chooses their best action by considering his/her forecast of the other drivers' action. The game included two "players"- merging car (two alternative actions: "*merge*" or "*pass*") and PF (two alternative actions- "*go with giveway*" or "*go without giveway*"). The strategy of a driver is defined as a set of probabilities assigned to each action. Thus "*the best response of a driver is the probability that maximises his/her expected payoff under*

the probability chosen by another driver." This model was seen to be capable of representing the drivers' revealed preferences and interactions in the merging process (Kita et al., 2002). However, this model did not include the acceleration control process on the acceleration lane when the merging traffic is interacting with the nearside motorway traffic, which should also be considered as an important kind of interaction during the merging process.

4.3.5. Implications for the Study

The literature review reveals the empirical observation of motorway on-ramp merging process and several limitations of existing merging behaviour models. It is found that most of the existing models assume that merging drivers are purely passive when responding to the situation: drivers evaluate the existing gaps and decide whether to accept or reject them but make no effort to adapt their position (by changing speed and acceleration) so as to fit in the gaps. No detailed description or discussion of the modelling of the acceleration/deceleration process of the merging vehicle between gap searching are included in the existing studies. There was little consideration of the merging vehicle's understanding of its surroundings in the exisiting studies, thus it was effectively "blind" to the courtesy yielding or cooperative lane-changing provided by the nearside motorway traffic.

In the next section, a new model of motorway traffic merging behaviour is presented which explicitly simulates the interactions between the gap-acceptance behaviour of the merging traffic and the cooperative behaviour of the motorway traffic.

Table 4-1 Summary of the motorway merging models.

Category	Main Idea	Research Work	Model	Advantage	Problems
Braking Risk Model	A driver is prepared to accept high risk of deceleration with the left distance to the end of merging area.	Gipps (1986)	Step 1- Urgency effect on acceptable braking risk (eq. (4-1)); Step 2- Safe merging speed (eq. (4-2)); Step 3- Feasibility of lane-changing: If $\| v_c(t+T)-v_c(t))/T \| < \| b_c \|$, it is not feasible.	(1)	(2), (3) and (4).
		Hunt & Yousif (1990)	Logic is similar to Gipps (1986) Note: It is mainly for the merging behaviour at road works instead of motorway acceleration lane situation.	(1) and (3)	(2) and (4)
		Hidas (2002)	Adjust Gipps' urgency effect on acceptable braking risk with the application of driver aggressiveness parameter θ (integer from 0 to 99) $b_C = [2-(x_{\max}-x_C)/10v_f] \times b_C^* \times \theta$	(1) and (3)	(2) and (4). Unrealistic severe braking.
		TRB (1997)	Urgency effect on acceptable braking risk: $b_c = b_{\min} + (b_{\max} - b_{\min}) \sqrt{\frac{(x_c - x_{\min})}{(x_{\max} - x_{\min})}}$ where, b_{max}, b_{min} are the limits of acceptable braking.	(1) and (4)	(2) and (3)

Probability Function Model	Driver merges with a certain probability in conjunction with gap acceptance model.	Yang & Koutsopoulos (1996)	Probability function related to the current position (eq. (4-3)) and gap acceptance model (urgency effect) (eq. (4-4)).	(1)	"Priority-based merging" (2), (3) and (4).
	Drivers force to merging in the situation of heavily congested traffic.	Ahmed (1999)	Using a binary logic model for the force-merging situation: $P\{Force-merging\} = \frac{1}{1+EXP[-(U_{Force-merge}-U_{no})]}$ where, the explanatory variables of $(U_{Force-merge}-U_{no})$ include the relative speed to PF, PL, remaining distance to the merging end, existing gap size and delay.	(1) and (3)	(2) and (4)
	The gap acceptance behaviour is a binary choice ("accept" and "reject")	Kita (1993)	Using a binary logic model for the gap acceptance behaviour: $P\{accept\} = \frac{1}{1+EXP[-(U_{accept}-U_{reject})]}$ where, the explanatory variables of $(U_{accept}-U_{reject})$ include the relative speed to PF, PL, remaining distance to the merge end, existing gap size and delay.	(1)	(2), (3) and (4).

Perceptual Model	Perception thresholds are used for a ramp driver whether to merge into the motorway or not.	Michaels & Fazio (1989) PTV (2003)	Perception threshold for merging (e.g. eq. (4-5)). If w_1 and w_2 below the threshold (0.004 rad/sec), merging is acceptable.	Perception thresholds' application into the merging behaviour.	(1), (3) and (4); No description about (2).
Fuzzy Logic Approach	A fuzzy logic model controls the acceleration behaviour.	Zheng (2002)	Merging vehicle acceleration control model (eq. (4-6)) and Gap acceptance model (eq. (4-7)).	(2) and (4)	(1) and (3)
Game Theoretic Approach	Merging situation is game between the merging car and motorway through car.	Kita et al. (2002)	The best response of a driver is the probability maximises his/her expected payoff under the probability chosen by another driver.	(3) and (4).	(1) and (2).

Note: (1)~(4) in Table 4-1 represents the properties in real traffic which are captured or not in the existing models:

(1) Urgency effects on merging vehicles as they get closer to the end of acceleration lane; (2) Merging drivers are positive instead of being passive so to adjust their speeds and acceleration rates to maximise the probability that they would be able to accept gaps; (3) Motorway nearside traffic's courtesy yielding's effect on the execution of merging decision; (4) Motorway nearside traffic's cooperative lane-changing's effect on the execution of merging decision.

4.4. A SIMULATION FRAMEWORK OF THE NEW MERGING MODEL

The new merging model tries to capture the interactions between the gap-acceptance behaviour of the merging traffic and the cooperative behaviour of the motorway traffic through a number of sub-models described as follows:

(a) Cooperation Model. Cooperative yielding behaviour and cooperative lane-changing are both modelled as a random decision made by the PF as to whether or not to reduce its speed to create gaps for C or to move to the offside lane(s).

(b) An acceleration model. This models the acceleration or deceleration of C towards its target gap while maintaining a safe distance away from the vehicle in front in the acceleration lane.

(c) A gap selection model. Based on its speed and location relative to its PF and PL, the driver of vehicle C will select a target gap to merge.

(d) A gap-acceptance model. Here the acceptable lead and lag gaps are calculated as a function of the speed, merging driver's reaction time and maximum deceleration of vehicle C, PF and PL by considering the forecast of their actions in the merging process.

(e) A merge model. When an acceptable gap is found, vehicle C merges into the motorway traffic. However, if the vehicle has not found an acceptable gap before reaching the end of the acceleration lane, a merge failure will be registered.

Using the symbols in Figure 4-2, the following notations are introduced which will be used in this section:

$a_i(t)$ the acceleration of vehicle i at time t (m/s^2);

b_i the maximum deceleration of vehicle i, negative value (m/s^2);

$b'_{i,j}$ the most severe braking of vehicle i as perceived by vehicle j, a negative value (m/s^2);

$v_i(t)$ the speed of vehicle i at time t (m/s);

$v'_{i,j}(t)$ the speed of vehicle i following its current traffic condition as perceived by vehicle j at time t (m/s);

$x_i(t)$ the position of vehicle i at time t (m);

L_i the length of vehicle i (m);

τ_i the reaction time of the driver of vehicle i (s);

θ a flag indicating whether courtesy yielding is provided by PF, where $\theta=1$ indicates a positive yielding by PF and $\theta=0$ otherwise;

g_z the lead ($z=Lead$) or lag ($z=Lag$) distance gap (m);

G_z the lead ($z=Lead$) or lag ($z=Lag$) time gap (s).

The framework of the simulated merging process is displayed in Figure 4-5, and the model formulation of each of the sub-models is described in the following sub-sections.

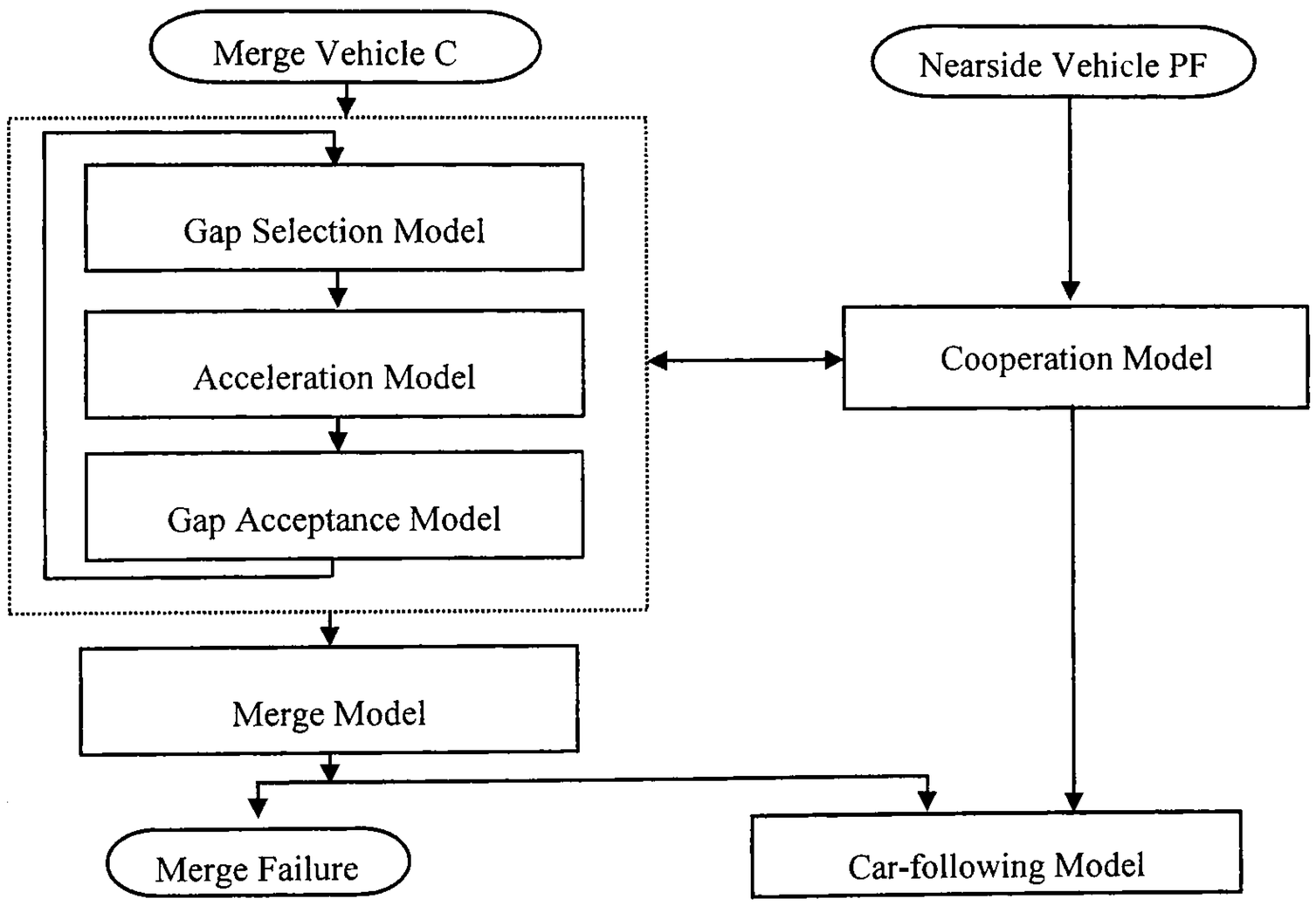


Figure 4-5 Simulation framework of the merging interaction process.

4.4.1. Model of Cooperation by the Motorway Traffic

This model attempts to capture the cooperative behaviour of the PF, which includes courtesy yielding and cooperative lane changing behaviour. These cooperation effects can facilitate the merging manoeuvres by creating gaps for the merging vehicles.

The decision on whether or not to provide cooperative lane-changing is drawn randomly from a binomial distribution with a given probability of α_1, the value of which can be obtained from direct video observation of the apparent cooperative lane-changing movements. In the current model, such a decision is made at every instant of time by the motorway drivers in the merging area where there is a merging attempt in front. A binomial distribution is applied here as it is able to represent these two possible decisions. In the simulation, the cooperative lane-changing can be perceived by C by means of a flag indicating whether lane-changing is occurring or not (in the real world, the direction indicators of the lane-changing vehicle) and will re-set its new *current gap* to that between the new PL and PF.

Similarly, the decision on courtesy yielding is a constrained binomial distribution of a rate α_2, subject to the current G_{Lag} being within the range $G_{\min} \le G_{Lag} \le G_{\max}$. The two limits, G_{min} and G_{max} are applied to represent the circumstances where no courtesy yielding is necessary: these are when the current G_{Lag} is either too small (below 0.25 second) or too large (above 4.0 seconds) based on the observation of Junction 11 of the M27 motorway, UK (Zheng, 2002). Hence the courtesy yielding value indicator (θ) can be generated as:

$$\theta = \begin{cases} Bin\ (\alpha_2) & for \quad G_{\min} \le G_{Lag} \le G_{\max} \\ 0 & otherwise \end{cases} \tag{4-8}$$

In the simulation, once a driver decides to show courtesy yielding, he/she will begin to follow vehicle C instead of his/her current leader on the motorway, and the car-following rule will be applied accordingly. Courtesy yielding is perceived by driver C by noticing an apparent change in the relative speed to vehicle PF. A constant deceleration is assumed by vehicle C in its estimation of the speed of PF. Hence the speed of vehicle PF, as estimated by C will be:

$$v'_{PF,C}(t) = \begin{cases} v_{PF}(t) & for \quad \theta = 0 \\ v_{PF}(t) + b_{PF}\tau_C & for \quad \theta = 1 \end{cases} \tag{4-9}$$

4.4.2. Model of Acceleration of the Merging Vehicle

The driver of the merging vehicle in the acceleration lane needs to conduct multiple tasks: to look for potential gaps on the motorway; to keep a safe distance away from the vehicle in front on the acceleration lane to avoid collision; and to keep an eye on the remaining length of the acceleration lane to avoid running into the end. Thus, three factors influence C's acceleration behaviour: (1) target gap on the motorway; (2) current leader on the acceleration lane and (3) the remaining distance to the end of the acceleration lane.

4.4.2.1. The influence of the target gap

Generally, the speed of the merging traffic is slower than that of the through motorway traffic (Michaels and Fazio, 1989; Kita, 1993). When it arrives at the acceleration lane, the merging vehicle will first accelerate to increase its speed and aim to reach the speed of the motorway traffic. If its speed is greater than that of the motorway traffic, the driver of vehicle C will decelerate to adjust its speed and position so as to fit in the gaps. Eq. (4-10a) represents the acceleration of vehicle C needed to reach the speed of the PL; while eq (4-10b) gives the deceleration of vehicle C at a rate controlled by the relative speed and gap with its PL.

$$a_C(t) = \begin{cases} \min(a_C^+(t), a_{\max}^+), a_C^+ = K_X \dfrac{v_{PL}(t) - v_C(t)}{\tau_C} & for \quad v_C(t) < v_{PL}(t) \quad (4\text{-}10a) \\ \min(a_C^-(t), a_{\max}^-), a_C^- = K_X \dfrac{(v_{PL}(t) - v_C(t))^2}{2\,|x_{PL}(t) - x_C(t) - L_{PL}|} & for \quad v_C(t) \geq v_{PL}(t) \quad (4\text{-}10b) \end{cases}$$

where $a_{\max}^+$ and $a_{\max}^-$ are the maximum acceleration and deceleration of a passenger car (listed in Table 2-3, given earlier in chapter 2); the values for HGVs being three-quarters of those of a passenger car (Goodman, 2001). K_X is a driver aggression factor which is uniformly distributed in the range of (0,1). The driver types are broadly clarified according to K_X values as described in Table 4-2. This parameter is used to model the behaviour whereby more aggressive drivers may be able to adjust their speeds and positions more rapidly than more timid drivers.

Table 4-2 The applied values of driver factor K_X uniformly distributed related to driver type X.

Driver type X	Very Timid	Timid	Average	Aggressive	Very Aggressive
K_X	0-0.2	0.2-0.4	0.4-0.6	0.6-0.8	0.8-1

When the speeds and longitudinal positions of the merging vehicle and the motorway nearside vehicle(s) become very close (i.e. the relative speeds within certain threshold ΔV and, the spacings within certain threshold ΔS), a situation termed as closing situation, eq. (4-10) will yield a very small acceleration of the merging vehicle which may prevent it reaching its desired gap. To overcome such a problem in the model, the acceleration of the merging vehicle will be such as to create a larger lag (eq. (4-11a)) if it is in closing situation with PF, or larger lead gap (eq. (4-11b)) if with PL. If vehicle C is in closing situation with respect to both PL and PF, eq. (4-11b) will be applied.

$$a_C(t) = \begin{cases} \min(a_C^+(t), a_{\max}^+) & a_C^+(t) = K_X \dfrac{2[\Delta S + L_C - (X_C(t) - X_{PF}(t))]}{\tau_C^2} \quad Closing \quad PF, \quad create \quad lag \quad (4\text{-}11a) \\ \min(a_C^-(t), a_{\max}^-) & a_C^-(t) = K_X \dfrac{2[\Delta S + L_{PL} - (X_{PL}(t) - X_C(t))]}{\tau_C^2} \quad Closing \quad PL, \quad create \quad lead \quad (4\text{-}11b) \end{cases}$$

The speed of the merging vehicle C as a result of interaction with PF and PL will be:

$$v_C{}^a(t + \tau_C) = v_c(t) + a_c(t)\tau_C \quad (4\text{-}12)$$

where v_c^a represents the speed of merging vehicle when interacting with its target gap.

It should be noted that if PF and PL do not exist (according to the definition given previously), then vehicle C will merge into the nearside traffic as soon as it arrives at the merging area and will move according to the normal car-following rule used for the motorway traffic. The logic of the acceleration control (when C is interacting with PF and PL) is summarised in Figure 4-6.

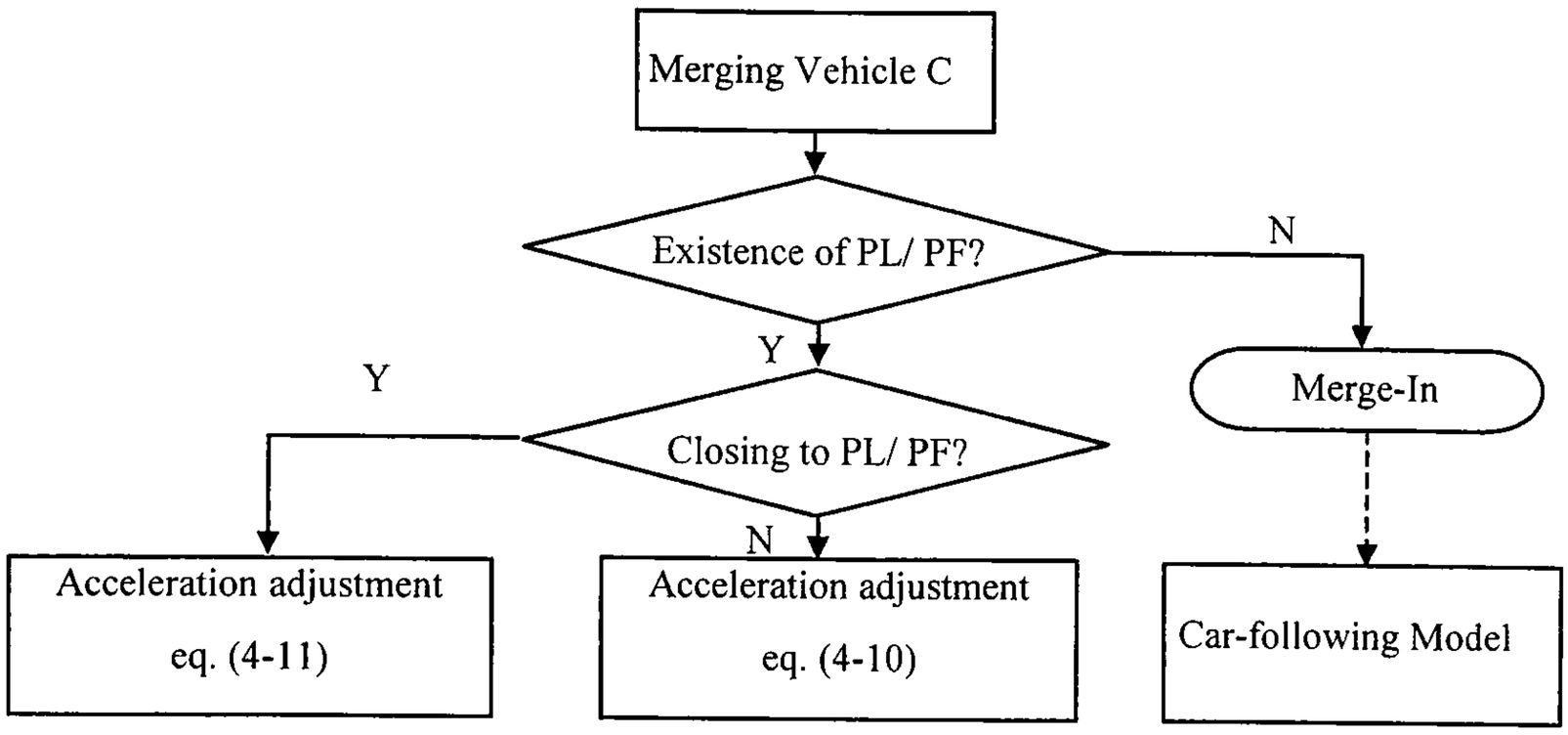


Figure 4-6 Acceleration control when C interacting with PF and PL.

4.4.2.2. The effect of the traffic in and the remaining length of the acceleration lane

If there is a vehicle (L) in front of vehicle C in the acceleration lane, C will try not to collide with L. In addition, vehicle C will continuously monitor its distance to the end of the acceleration lane so as not to overrun it. In the model, a maximum acceptable braking (b_c) is applied as an urgency effect on C to prevent it from running into either the vehicle in front or the end of the acceleration lane. The value of b_C is related to the distance to the end of the acceleration lane, subject to the vehicle's maximum deceleration $a_{\max}^{-}$. The driver aggressiveness factor is also applied here, hence the proposed equation of b_C is as follows:

$$b_c = \max(-K_X \frac{v_C^2}{2(x_{\max} - x_c)}, -a_{\max}^{-}) \quad (4\text{-}13)$$

where x_{max} is the location of the end of the acceleration lane. The maximum deceleration b_C is used as a pressure for the merging vehicle to accept a smaller gap to the preceding vehicle on the acceleration lane and to the PL on the motorway.

Applying b_C to the Gipps collision avoidance model (1981), the speed of vehicle C following its leader in the acceleration lane is calculated as:

$$v_C^b(t+\tau_C)=b_c\tau_C+\sqrt{b_c^{\,2}\tau_C^{\,2}-b_c\{2[x_L(t)-L_L-x_c(t)]-v_c(t)\tau_C-v_L^{\,2}(t)/b_{L,C}^{'}\}} \quad (4\text{-}14)$$

where v_c^b represents the speed of merging vehicle when interacting with the vehicle in front on the acceleration lane by considering the urgency effect.

Then the actual speed of the vehicle C on the acceleration lane which is constrained by both the traffic on the motorway (eq. (4-12)) and that in front in the acceleration lane (eq. (4-14)) is given by:

$$v_C(t+\tau_C)=\min\{v_C^a(t+\tau_C),v_C^b(t+\tau_C)\} \quad (4\text{-}15)$$

This speed is also used as the forecast speed of C in its selection of acceptable gaps as described in the next sub-section.

4.4.3. The Model of Gap Selection and Acceptance

By default, the target gap is assumed to be the original gap according to Zheng's observation (2002). However, the merging driver may choose to select a different gap as the target gap in different situations: a fast moving merging vehicle may overtake PL on the acceleration lane and choose to take the previous gap as its target gap; conversely, a slow moving merging vehicle may then choose to target the following gap; alternatively, if neither lead nor lag gap is acceptable, the merging vehicle will have to wait for other gaps.

The gap-acceptance model developed here has adopted the game theory idea proposed by Kita et al. (2002), where the merging vehicle C chooses its best action by considering its forecast of the other drivers' actions (in terms of their speed changes and cooperation) and its own actions (in terms of its speed and acceptable braking). By applying b_C as a pressure for vehicle C to accept smaller gaps, and with the application of a gap acceptance sensitivity parameter β, the acceptable lead and

lag gaps can be derived from an interpretation of the Gipps car-following model (1981) as follows:

$$\overline{g}_{Lead}(t) = \frac{1}{2}\beta\left[\frac{v_{PL}{}^{2}(t)}{b'_{PL,C}} - \frac{v'_{C}(t)^{2}}{b_{C}} + 2\tau_{C}v'_{C}(t) + v_{C}(t)\tau_{C}\right] \quad (4\text{-}16a)$$

$$\overline{g}_{Lag}(t) = \frac{1}{2}\beta\left[\frac{v_{C}{}^{2}(t)}{b'_{C,PF}} - \frac{v'_{PF,C}(t)^{2}}{b_{PF}} + 2\tau_{C}v'_{PF,C}(t) + v_{PF}(t)\tau_{C}\right] \quad (4\text{-}16b)$$

The gap acceptance sensitivity parameter β (≤ 1) which directly affects the lead and lag thresholds (eqs. (4-16)), is applied based on the finding (Wilson, 2001) that the gaps modelled by Gipps' car-following model are generally longer than observed. The effects of this parameter on the merging process will be discussed in detail in the later section on sensitivity tests.

To account for human errors and variability in driving proficiency, a normal distribution is applied to the acceptable lead and lag (eq. (4-17)) with $\overline{g}_{Lead}(t)$ and $\overline{g}_{Lag}(t)$ being the mean values and σ^2 the variance, limited by a lower boundary g^{min} at 4.5m (Zheng, 2002). It should be emphasised that this variability refers not to that across driver population, but only to the inability of the driver to maintain a constant gap due to driving proficiency or human error.

$$g_{lead}(t) = \max\{N(\overline{g}_{Lead}(t), \sigma^2), g^{\min}\} \quad (4\text{-}17a)$$

$$g_{Lag}(t) = \max\{N(\overline{g}_{Lag}(t), \sigma^2), g^{\min}\} \quad (4\text{-}17b)$$

Note from equations (4-16) and (4-17) that the higher the absolute value of b_C (when vehicle C is closer to the end of the acceleration lane), the smaller the lead threshold, and that the lower the perceived speed of PF (for example when courtesy yielding is provided by PF), the smaller the lag threshold. It can also be seen that the longer the reaction times, the higher the value of lead and lag thresholds and that the lower the gap acceptance sensitivity parameter, the smaller the lead and lag thresholds.

4.4.4. The Merging Model

When an acceptable gap is found, vehicle C will merge into the motorway traffic and be placed behind PF; otherwise, C will adjust its speed and look for another gap. If however, on reaching the end of the acceleration lane, the vehicle C has not found an acceptable gap, it will be removed from the acceleration lane and a merge failure will be registered denoting the time and location of the removal. Such treatment of merging failures is also applied in VISSIM (PTV, 2003). This treatment, though subjective, is consistent with the observation that merging drivers very rarely stop at the end of the acceleration lane (Zheng (2002) and video observations of the merging traffic on the UK M8 motorway for this PhD research).

4.5. SIMULATION AND SENSITIVITY TESTS OF THE NEW MERGE MODEL

4.5.1. Experimental Design and Model Performance

Simulation tests were conducted on a 500-metre long merge section with a multi-lane motorway and one parallel acceleration lane (Figure 4-7) (screenshots given in Appendix E). Both the motorway traffic and the ramp traffic enter the test section at point X=0, 100 metres before the start of the actual merge area and the acceleration lane (point X_{min} in Figure 4-7).

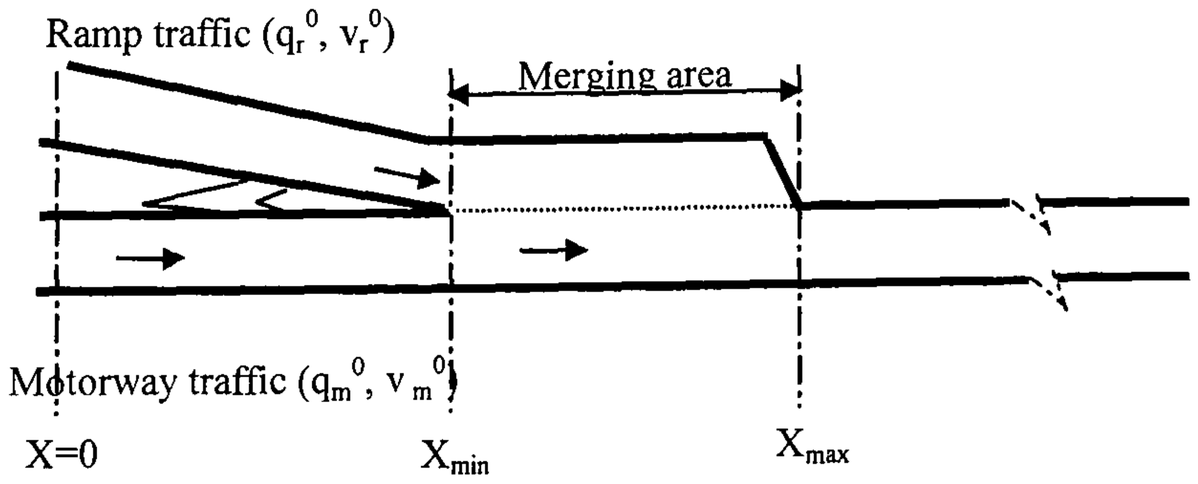


Figure 4-7 Motorway merging configuration used in simulation. The simulated section starts from an arbitrary point upstream of the merge area, denoted as X=0; all positions are measured relative to this point.

In the tests, only the merging traffic on the acceleration lane and the traffic on the nearside lane of the motorway are simulated. It is assumed that the traffic on the other lanes of the motorway will not change to the nearside lane during the merge section, and hence will not have direct interaction with the merging traffic. Any nearside vehicle which chooses to carry out cooperative lane-changing will simply be removed from the simulation. This treatment, though subjective, is consistent with the definition of the parameter- probability of cooperative lane-changing (α_l). As an input parameter, α_l is referred as an observed value which should be obtained from direct observation on the proportion of the cooperative lane-changing vehicles. It should be mentioned that those observed vehicles when providing cooperative lane-changing should have already taken into account the traffic on other lanes.

Both the motorway and the ramp traffic arrive at the simulated section randomly as descibed earlier in Chapter 2. The nearside motorway traffic arrives with an average flow q_m^0 and an entrance speed v_m^0. The ramp traffic is generated from an average flow q_r^0 and enters the section with an initial speed v_r^0. The simulation is updated every 0.2 seconds, and the default simulation parameter values are listed in Table 4-3.

Table 4-3 The default values of the model parameters.

Denotation	Parameters	Default value	Reference
$b'_{i,j}$	Braking of vehicle i estimated by vehicle j (m/s^2)	$\min[-3.0, \frac{-2.0\times N(1.7,0.3^2)-3.0}{2}]$	Gipps (1981)
b_{PF}	The maximum deceleration of vehicle PF (m/s^2)	-4	Gipps (1986)
(ΔS, ΔV)	Closing thresholds (m, m/s)	(4.5, 2)	Zheng (2002); Brackstone et al (2002)
g^{min}	Lower boundary of gap acceptance thresholds (m)	4.5	Zheng (2002)
τ_C	Reaction time (s)	0.4	Toledo (2003)
L_{acc}	Acceleration lane length (m)	147	Detector information of Junction 27, M8, UK
(q_r^0, v_r^0)	Arrival ramp merging traffic flow and speed respectively (veh/h, km/h)	(600, 55)	Detector information of Junction 27, M8, UK
(q_m^0,v_m^0)	Arrival motorway traffic flow and speed respectively (veh/h, km/h)	(1500, 90)	
α_1	Probability of cooperative lane-changing	0.07	Video observation of Junction 27, M8, UK
α_2	Probability of courtesy yielding	0.2	
β	Gap acceptance factor	0.5	Assumption

The simulation outputs a selection of the different gaps taken by the merging vehicle to illustrate the performance of the gap acceptance behaviour modelled, and how smooth the merge is. Figure 4-8 presents examples illustrating merges in which an original, a following and a previous gap were accepted. From the space-time diagrams of the merge, it can be seen that the appropriate gaps with its leading and following vehicles have been sought and maintained by the merge vehicle.

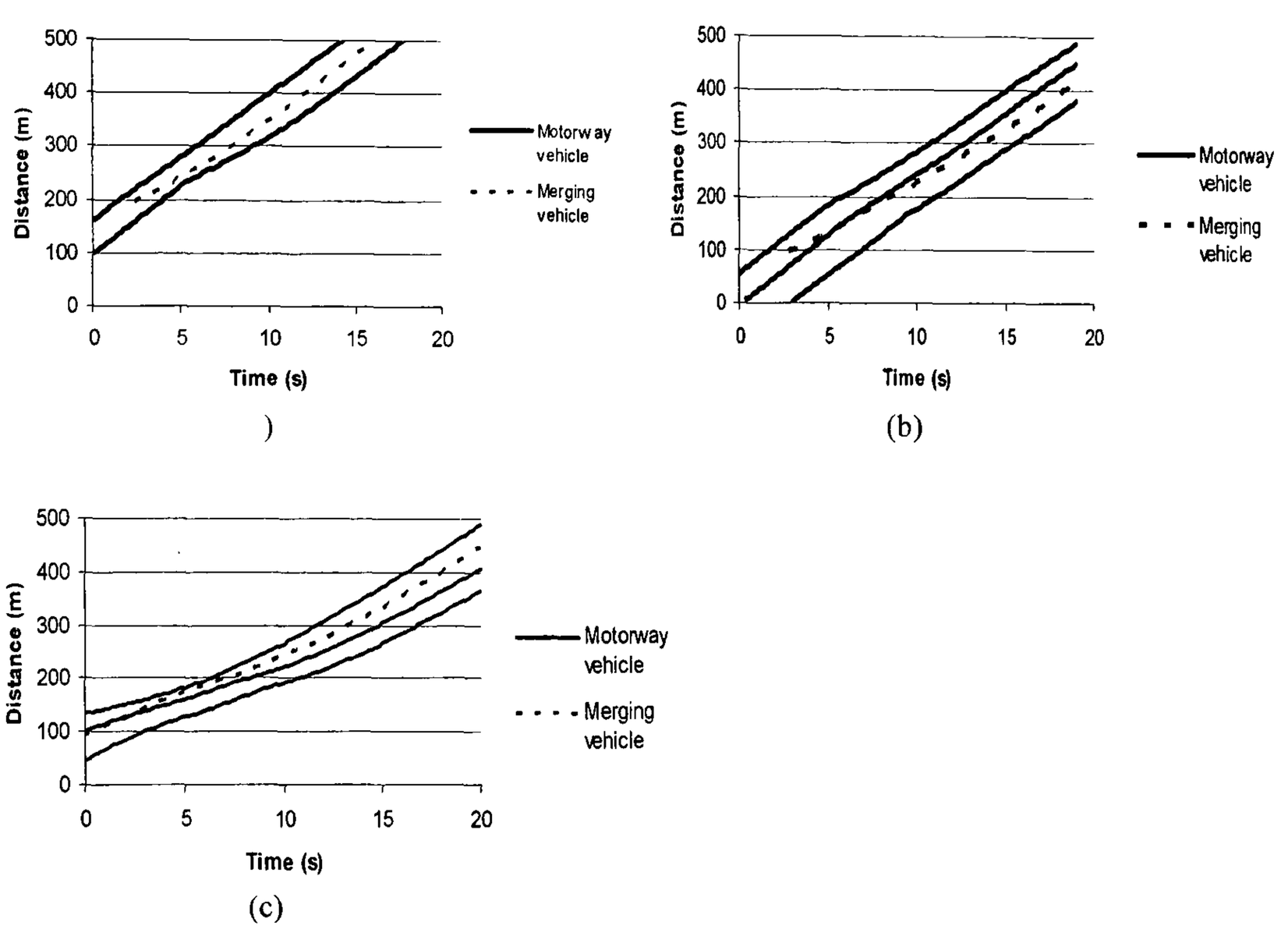


Figure 4-8 The space-time trajectories of a merging vehicle (in broken lines) and of its PL and PF vehicles on the nearside motorway (solid lines) where the merging vehicle accepted (a) the *original gap*, (b) the *following gap* and (c) the *previous gap*.

The interactions between the merging vehicle and the vehicles on the motorway can be better illustrated through detailed analysis of the speed profiles and the distance-time trajectories of the vehicles during a merging operation. Figures 4-9, 4-10, 4-11 show the speed-distance-time profiles of three different merging operations.

Figure 4-9 shows a typical merge whereby the motorway traffic did not give any cooperation and the merging vehicle had to *force* its way into the motorway traffic. It can be seen (Figure 4-9a) that when the merging vehicle arrived at the merging area, it first accelerated to try to catch up the speed of its PL. As it gets closer to the end of the acceleration lane (at a distance of 247m from the origin), the pressure for vehicle C to merge gets higher. Then at time 11.2 second and location

209 m, vehicle C executed a forced merge with a speed of 22m/s (Figure 4-9(a)), a lead gap of 42m and a lag gap of 15m (Figure 4-9(b)). Vehicle PF reacted to the forced merger by decelerating (at a rate between 1.3-2.9m/s^2) from time 11.4s (point A in Figure 4-9) to time 13.2s (point B) at which time PF reached a relatively safer time gap (20m) with its new leader, vehicle C. Thereafter, vehicle PF accelerated to match his speed to that of C and PL and maintain a safe distance gap with vehicle C (Figure 4-9 (b),(c)).

Figure 4-10 shows the speed-distance-time profiles of the vehicles during a merge with courtesy yielding. The courtesy yielding of PF started at time 7.6 s, position 115m with a speed of 23m/s (point A on Figure 4-10). Meanwhile, the merging vehicle C accelerated after it arrived at the acceleration lane with its original lag of 14m and original lead 44m (Figure 4-10 (b)). Vehicle C executed the merge at time 11.4s and location 208m at a speed of 23m/s, comparable to that of its PL with an accepted lead of 53m and lag 23m (Figure 4-10 (b)). Due to the high acceleration of vehicle C and the courtesy yielding, the PF followed with a continuous deceleration after its initial yielding with a deceleration at 3.2m/s^2 and began to accelerate again at around time 9.8 s (point B, Figure 4-10) by which time the safe lead and lag gaps had been reached.

Figure 4-11 shows the situation when the original PF provided cooperative lane-changing to facilitate the merging. When vehicle C first arrived at the merging section, its speed (at about 16 m/s) was much lower than that of the motorway traffic (which is well over 20 m/s) in Figure 4-11(a), and it already had a close gap to the original PF (the *original lag* indicated in Figure 4-11(b) is 28 m). When C accelerated to reach the speed of the motorway traffic, the gap between it and the original PF got even smaller (at time 9.6 s, it decreased to 4m between C and original PF, point A in Figure 4-11 (b)). The original PF made a cooperative lane-changing manoeuvre at time 9.6 s and location 173 m (Figure 4-11(b)-(c)). This instantly created a new and larger lag (63 m as indicated in Figure 4-11(b)) for C which waited for the new PF to respond to the situation (e.g. to react to the presence of a merging vehicle in front) before taking an easy merge at time 10.0 s and location 185 m.

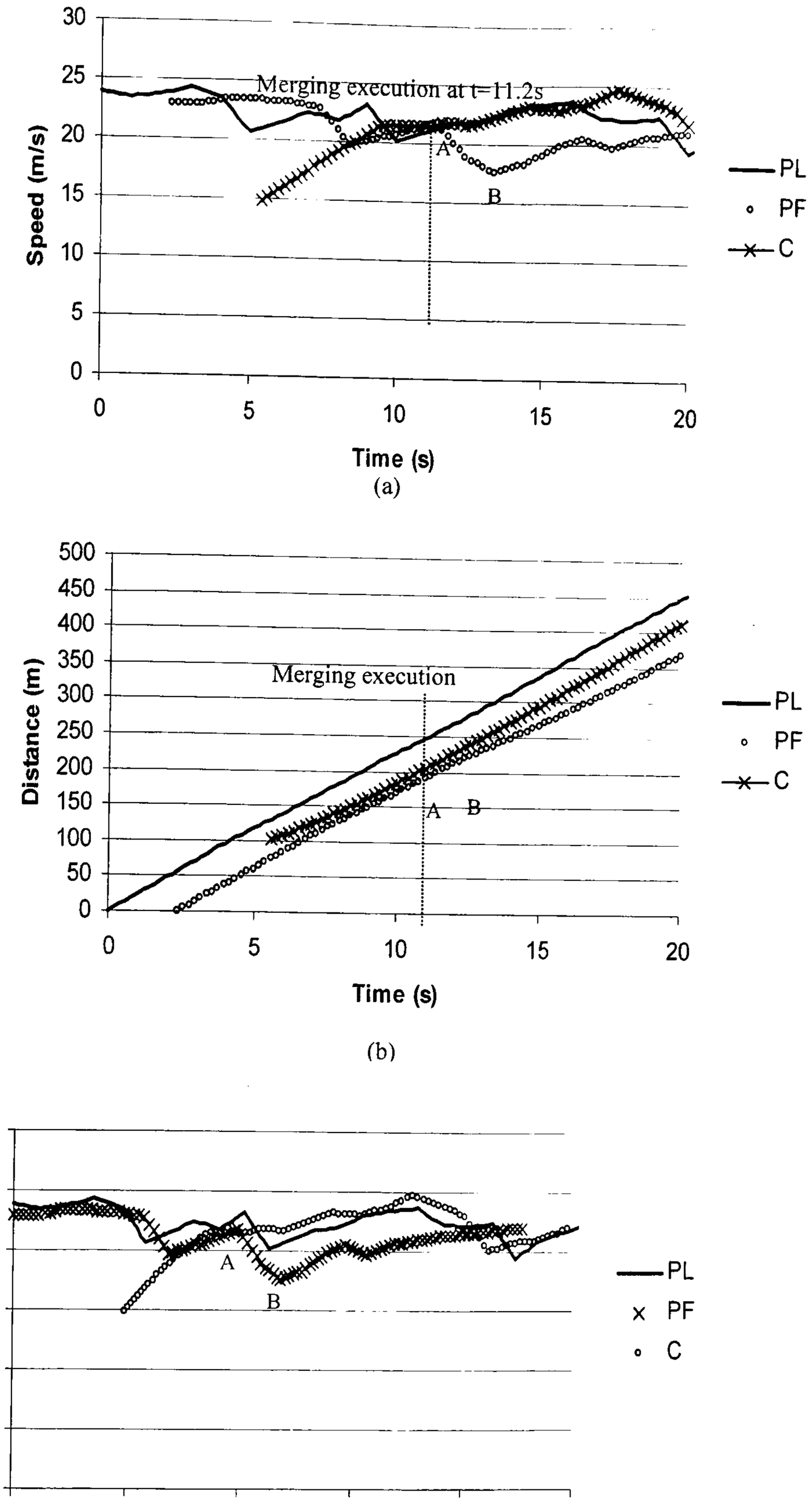
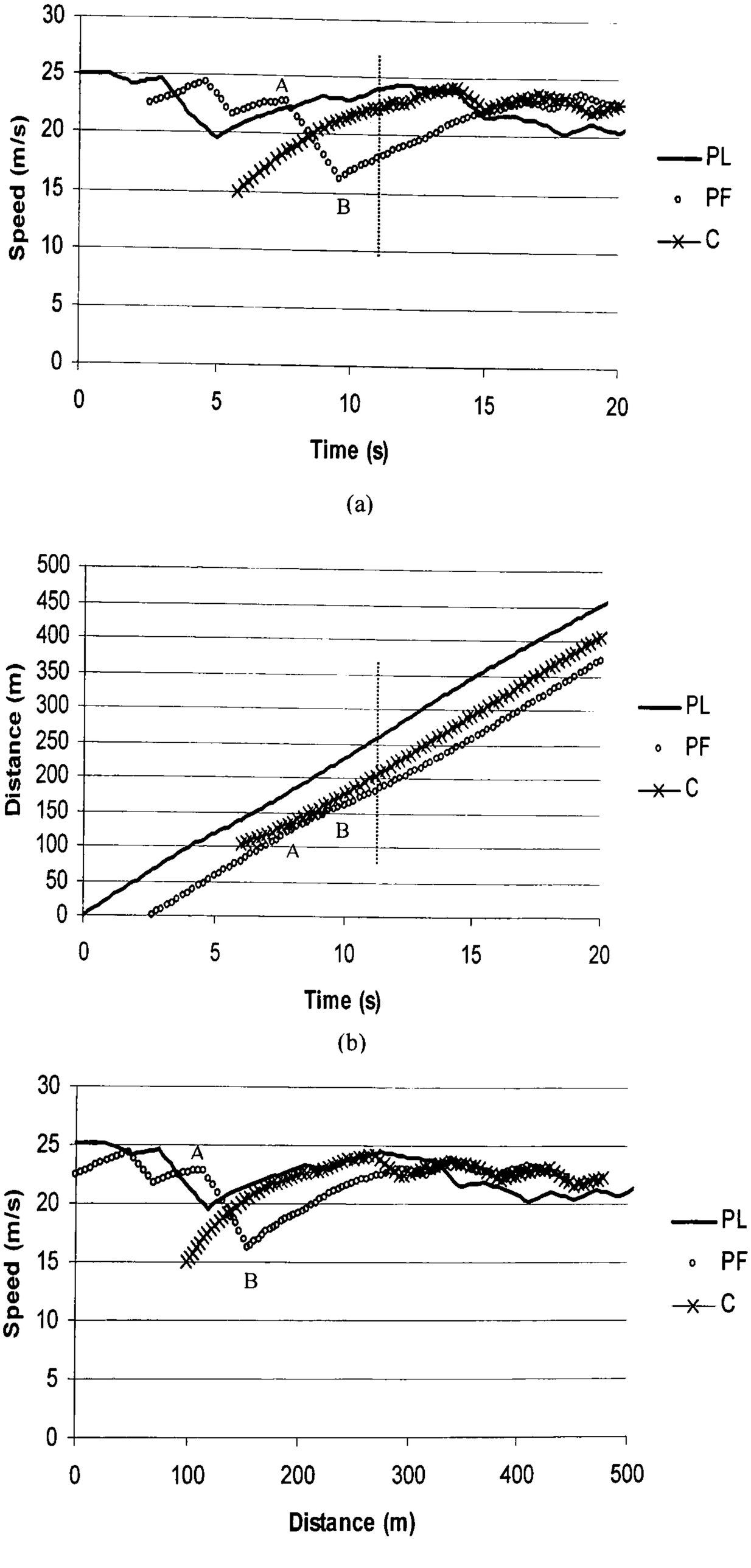


Figure 4-9 The profiles of (a) speed-time, (b) distance-time and (c) speed-distance of the merging vehicle and those of its PL and PFs during a forced merge. Points A and B are the corresponding points where a forced yielding was executed.



(a)

(b)

(c)

Figure 4-10 The profiles of (a) speed-time, (b) distance-time and (c) speed-distance of the merging vehicle and those of its PL and PFs with courtesy yielding provided. Points A and B are the corresponding points where a courtesy yielding was executed.

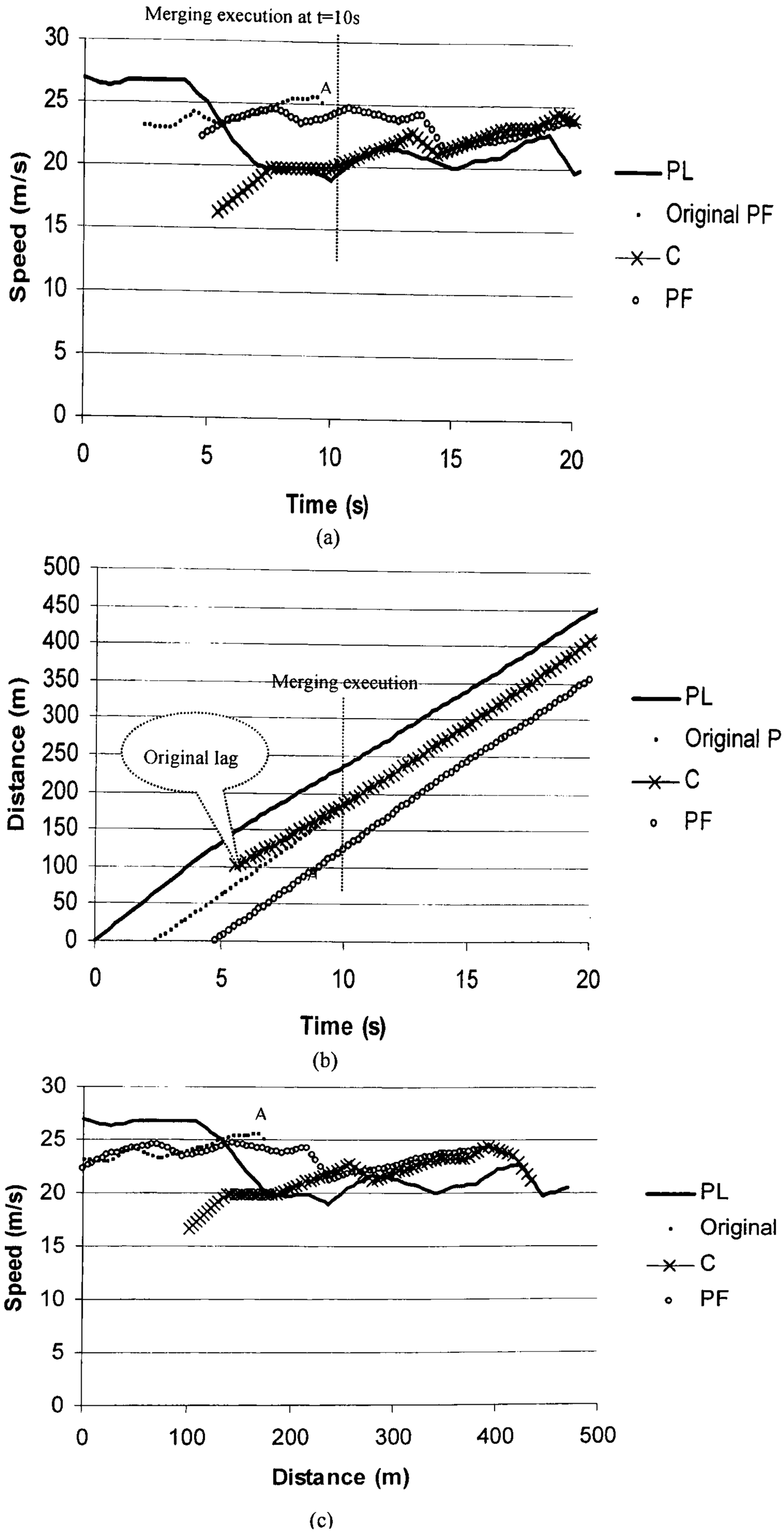


Figure 4-11 The profiles of (a) speed-time, (b) distance-time and (c) speed-distance of the merging vehicle and those of its PL and PFs with cooperative lane-changing. Points A are the corresponding points where the cooperative lane-changing was provided.

The above three examples demonstrate the ability of the model to represent different merging behaviours including forced merging (Figure 4-9), merging under courtesy yielding (Figure 4-10) and cooperative lane-changing (Figure 4-11). In the next section, sensitivity tests are carried out on some of the key model parameters to investigate the model's response to the changes in parameter values.

4.5.2. Sensitivity Analysis

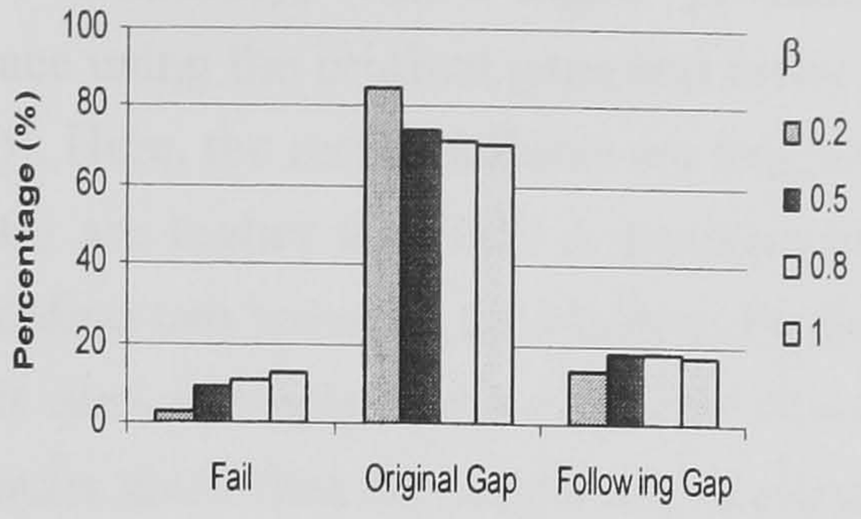
This section reports a sensitivity analysis of some of the key model parameters as listed in Table 4-4. The parameter values have been systematically varied in the simulation tests. The indicators used in the sensitivity analysis are the numbers of successful merges which took the original gap, the following gap and the number of failed merges. Figures 4-12 to 4-14 show the average results of the simulation tests, presented as percentages of the total number of merges.

Table 4-4 Parameters for sensitivity tests.

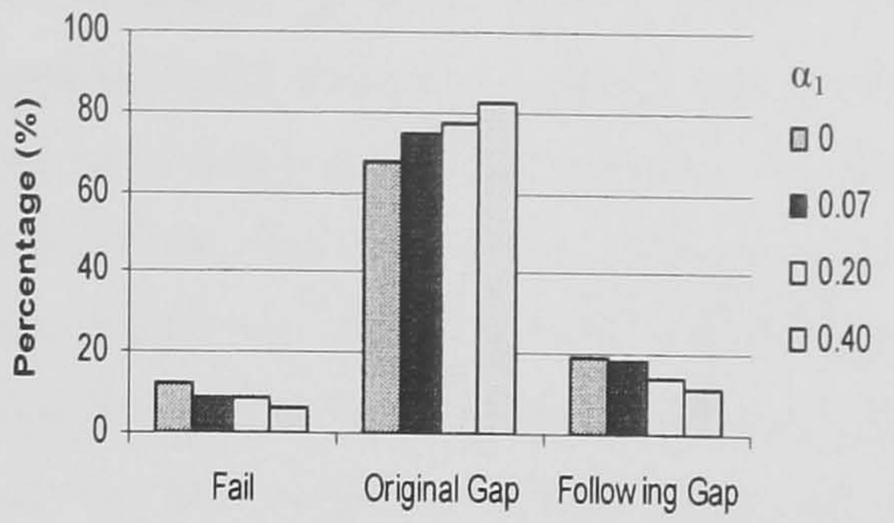
Parameter	Default Value	Test Range	Test Increment
β	0.5	0.2 ~ 1.0	varies
α_1	0.07	0.0 ~ 0.4	varies
α_2	0.2	0.0 ~ 0.8	0.2
L_{acc} (m)	147	50 ~ 250	50
τ_C (s)	0.4	0.4 ~ 1.0	0.2
q_r^0 (veh/h)	600	200 ~ 800	300
v_m^0 (km/h)	90	80 ~ 120	varies
q_m^0(veh/h)	1500	900 ~ 1800	300
HGV (%)	5	0 ~ 20	5

In the following sensitivity tests of the proposed merge model suggest that the model can reasonable replicate the complicated merging process and, reflect the effects of some specific parameters (which is related to empirical site properties) on

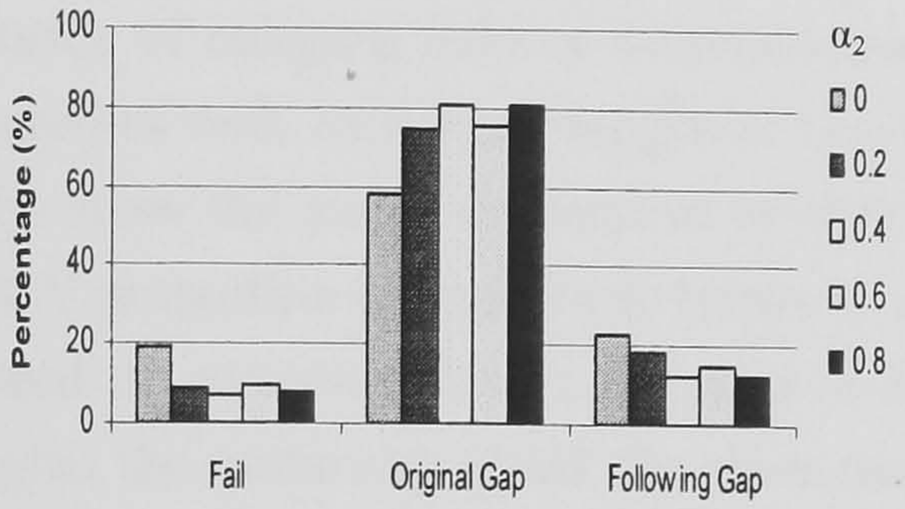
merging manoeuvre, i.e. acceleration lane lengths, motorway drivers' cooperation (i.e. yielding, lane-changing), motorway HGV proportions and traffic speeds and flows. The results show that lower values of the gap-acceptance factor (β) leads to more original gaps being taken, fewer failed merges and fewer merges taking the following gap (Figure 4-12(a)). This is intuitively correct as the parameter β directly affects the gaps acceptable for the merging vehicles (via equations (4-10) and (4-11)). However, the effect of β becomes less significant when its value is higher than 0.8. The analysis shows that the model is sensitive to the cooperative lane-changing rate α_1 (Figure 4-12(b)). When cooperative lane-changing is more common, merging vehicles are more likely to take the original gap and less likely to take the following gap or fail to merge.



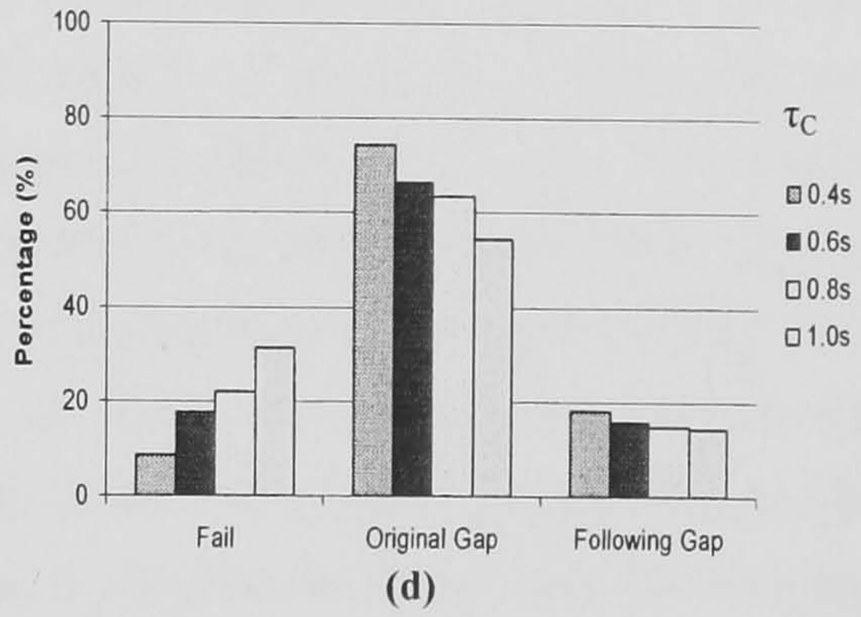
(a)



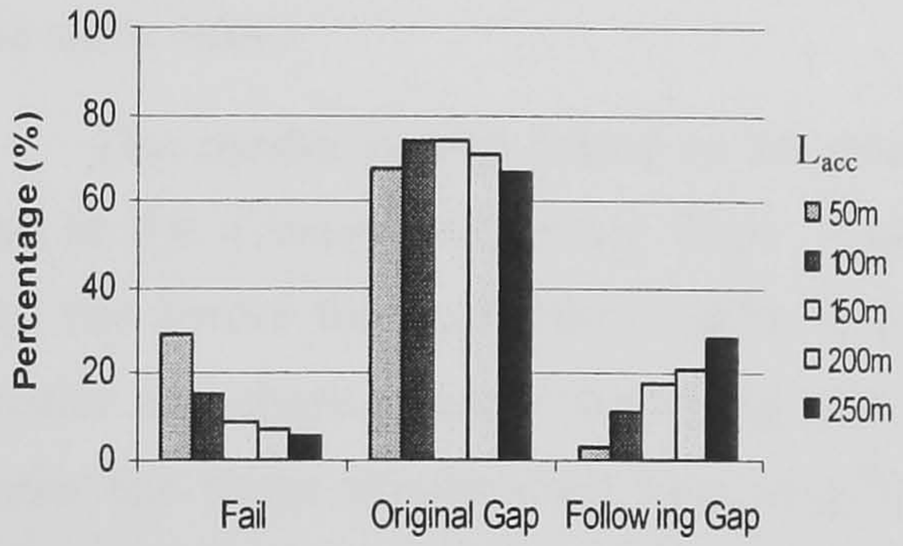
(b)



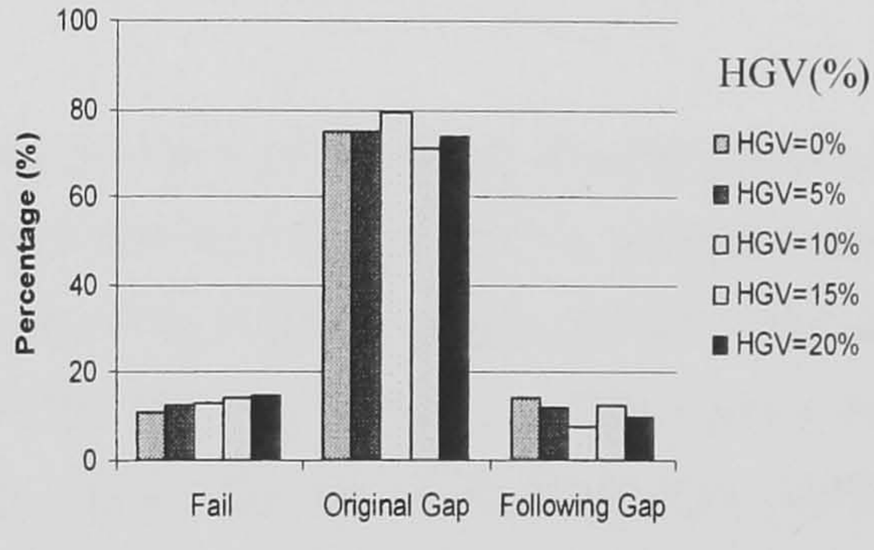
(c)



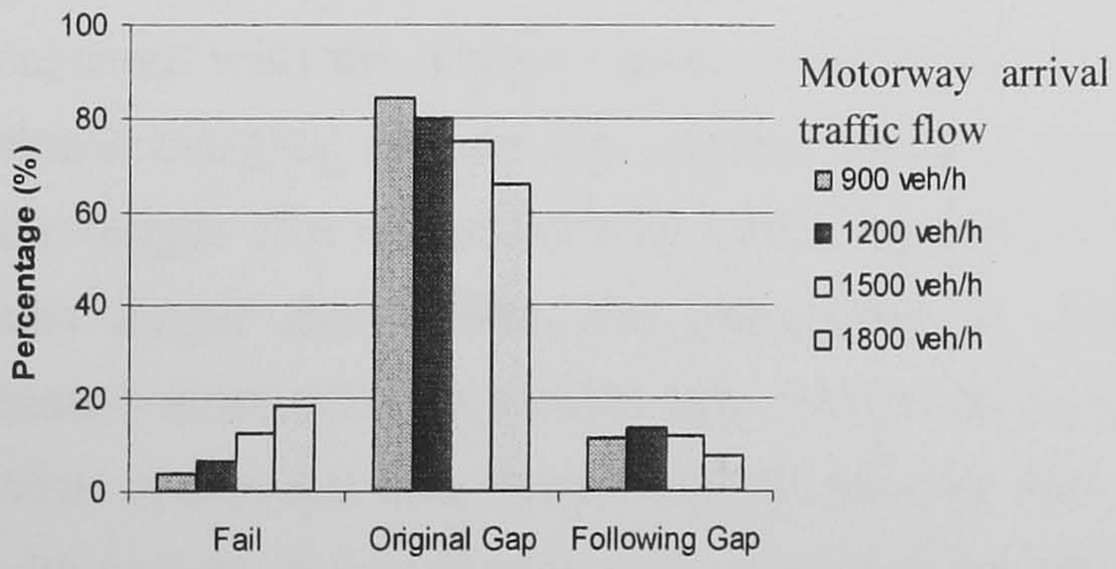
(d)



(e)



(f)

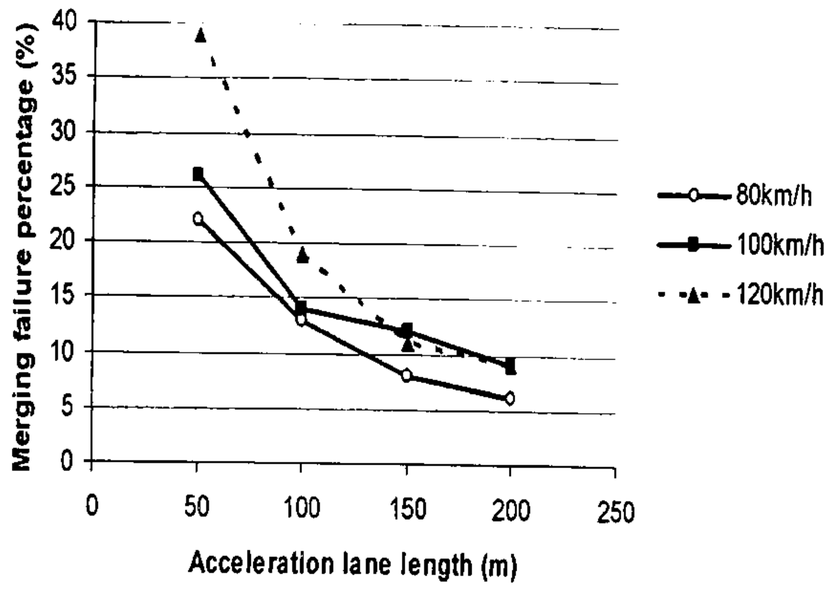


(g)

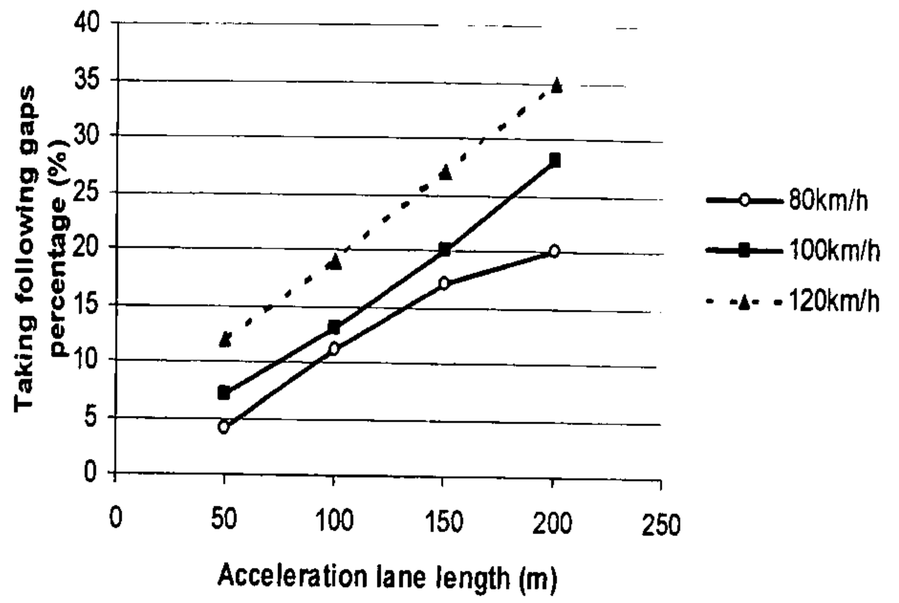
Figure 4-12 Distribution of the percentages of merging failure, successful merging into the original gap and into the following gap in the sensitivity tests on (a) gap acceptance sensitivity value β, (b) cooperative lane-changing fraction α_1, (c) courtesy yielding fraction α_2, (d) reaction time τ_C,(e) the length of acceleration lane, (f) HGV (%) and (g) motorway arrival traffic flow.

Similarly, with a higher probability of courtesy yielding, more merges take place using the original gaps and fewer merges with the following gaps (Figure 4-12 (c)). Here, the merge failures are less sensitive to courtesy yielding when the yielding rates are higher than 0.2. A possible explanation for this is that excessive courtesy yielding can improve the chances for the merging drivers to accept the original gaps but does not help those unskilled drivers who have poor acceleration control. The results show that the longer the reaction time, the lower the percentage successfully merging (Figure 4-12 (d)). The analysis shows that the chance of taking an original gap does not vary significantly with the length of the acceleration lane; however, the chance of merging failures decreases and the chance of accepting a following gap increases with increasing length of acceleration lane (Figure 4-12 (e)). The results also show the model is sensitive to HGV proportion on nearside motorway. Higher HGV proportion on motorway brings to more merging failures (Figure 4-13(f)). The speed of motorway traffic is found to have an effect on merging behaviours: the higher the motorway speed, the more merging failures (Figure 4-13(g)). This might be explained that with higher motorway speed, it bring more difficulties for merging vehicles to fit in motorway gaps and catch up with the motorway vehicle's speed at the same time.

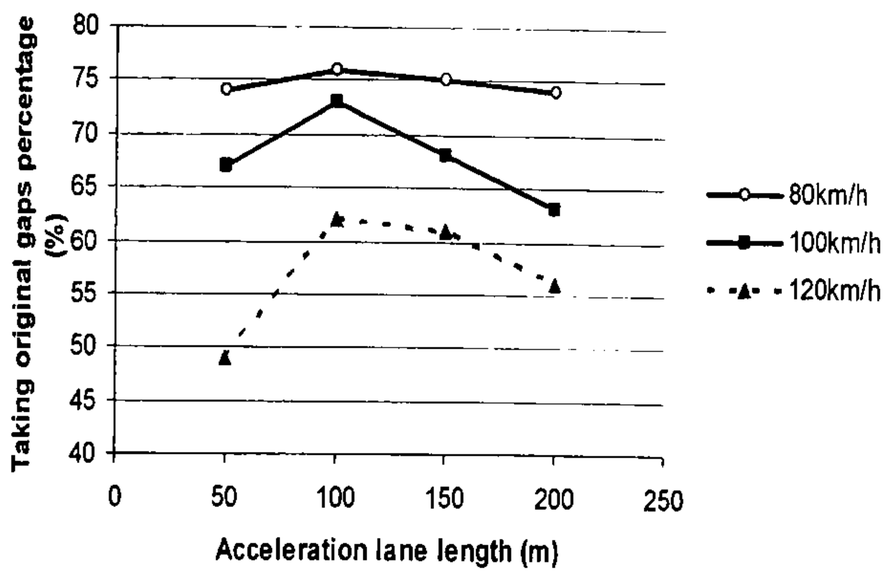
The model is also found to be sensitive to the length of the acceleration lane and to the average motorway flow. It appears that the longer the acceleration lane and the lower the motorway traffic speed, the fewer the merging failures and the greater the chance that a following gap will be chosen. Figure 4-13(a) shows that under the same acceleration lane length, the higher the speed of motorway traffic, and the more the merging failures. This results imply that with higher motorway design speed, generally the longer the acceleration lane length is needed which is consistent with the design standard of DMRB (given Table 4-5). In order to achieve smooth merging (taking the original gaps), it appears that the optimal acceleration lane length (for all motorway traffic speeds), is approx 100m. With acceleration lanes longer than 100m, the percentage of vehicles taking original gaps drops, because more vehicles tend to take the following gaps (as shown in Figure 4-13 (c)). These results are consistent with simulation tests of the standard motorway design (with typical design speeds and associated acceleration lane lengths as applied in the UK, Table 4-5).



(a)



(b)



(c)

Figure 4-13 The effects of acceleration lane length and motorway arrival speed on the distribution of (a)merging failure, (b) successful merging into the following gap, and (c) merging into the original gap.

Table 4-5 Design speed and the associate length of the acceleration lane (DMRB, 2001a).

Design Speed (km/h)	Merging Length (m)
85	90
100	110
120	130

Simulation tests using the design speeds and merge lengths of Table 4-5, suggest that, with the design merge length, merging failures are generally below 15% of the total merging attempts and that with higher design speeds and longer merging lengths, more vehicles tend to take the following gaps and fewer take the original gaps (Figure 4-14). This result may have some practical implications for motorway merging design: taking smooth merging (i.e. taking original gaps) as the main criterion, the optimum length of acceleration lane is approx 100m; taking merging

failure as the main criterion, a longer acceleration length such as 200m is required. That is, the design may be different depending on which criterion is chosen.

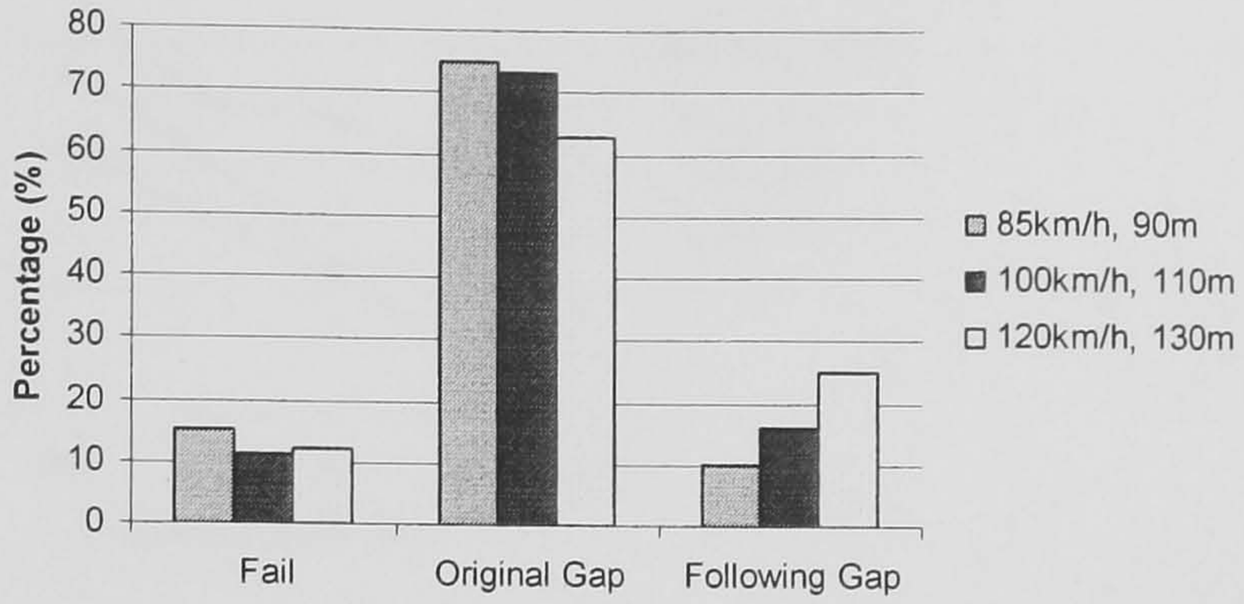


Figure 4-14 Merging at various design speeds and merging lengths, according to DMRB(2001a).

Figure 4-15 shows that the accepted lead and lag gaps decrease as the flows on the motorway increase. However, they do not seem to vary with increasing merging traffic flows. Both of these results are in agreement with real-world observations (Zheng, 2002).

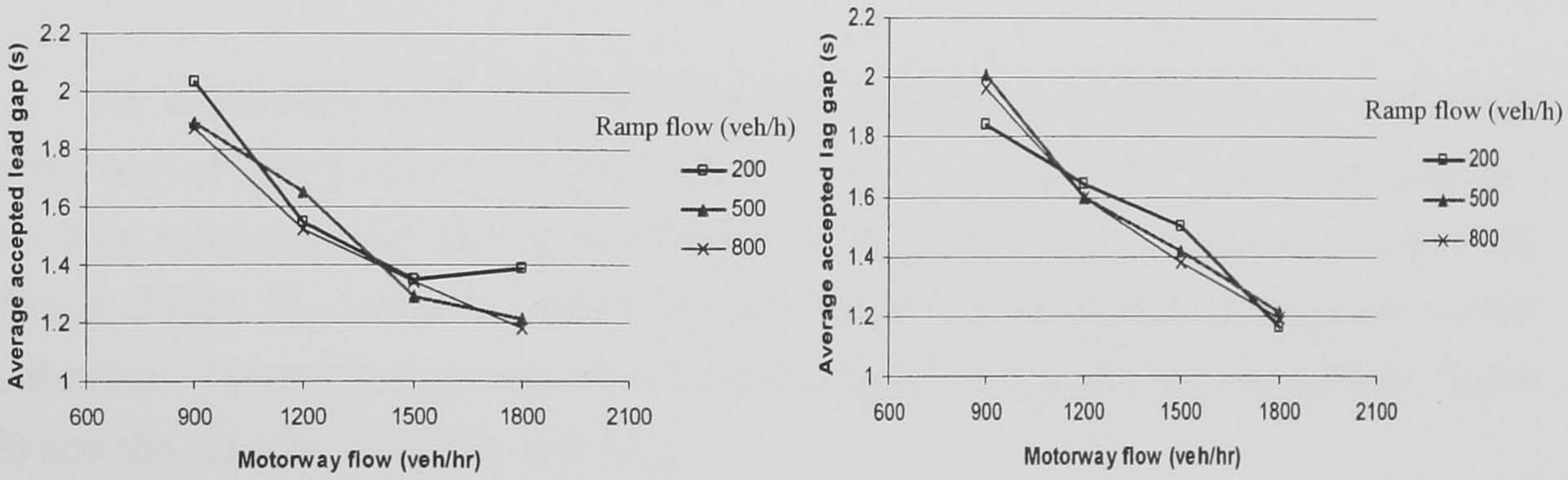


Figure 4-15 Variation of the accepted lead and lag gaps with motorway and ramp flows.

The effects of the ramp merging flow and the motorway traffic flow on successful merging are shown in Figure 4-16. In contrast with Figure 4-15, these results suggest that the percentage of successful merges decreases with increasing nearside motorway traffic flow and also with increasing merging traffic flow. This is because the greater the motorway traffic flow, the smaller are the gaps for the merging vehicles, hence the lower the merging success rate. To summarise, although the ramp flow traffic has an insignificant effect on the accepted gap values (Figure 4-15), it does affect merging success because the higher the ramp flow, the lower the merging opportunities, hence the lower the percentage of merging success.

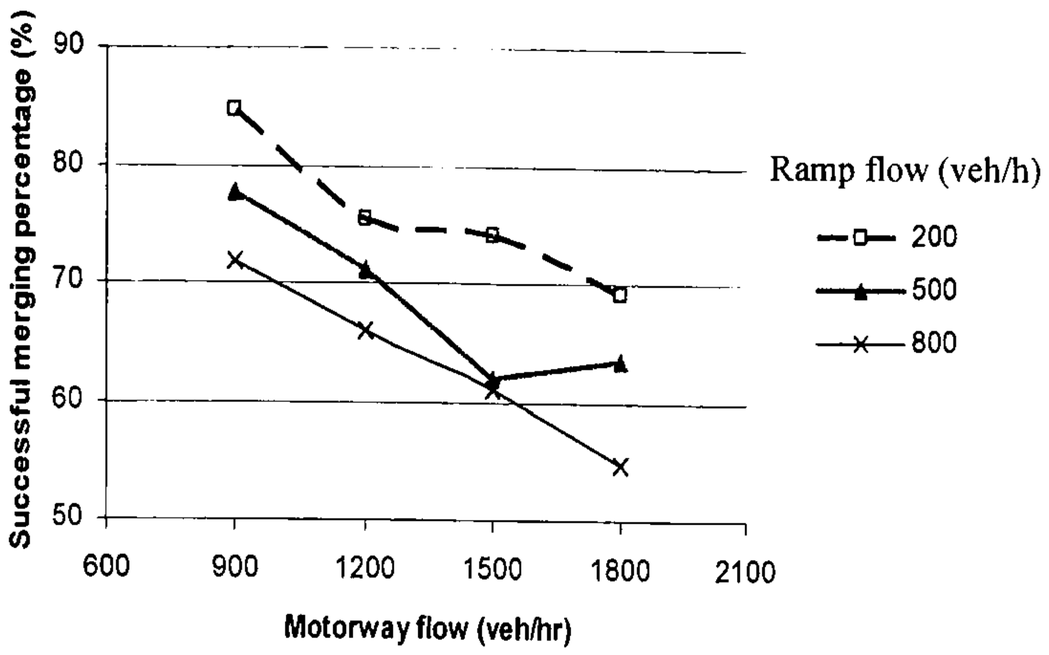


Figure 4-16 Successful merging with motorway and ramp flows.

4.6. CALIBRATION OF THE MERGING MODEL

The calibration of the proposed merging model was conducted on a real-world merge section using data collected from Junction 11 on the M27 motorway during a two-hour morning peak period (07:00-09:00) between 21st May and 17th July 2001 (Zheng, 2002). The observed data listed in Table 4-6 were used as input to the model calibration. The model parameters to be calibrated here are the gap acceptance factor (β) and the drivers' reaction time (τ_C).

Table 4-6 Observed data from Junction 11, M27 motorway (Zheng, 2002).

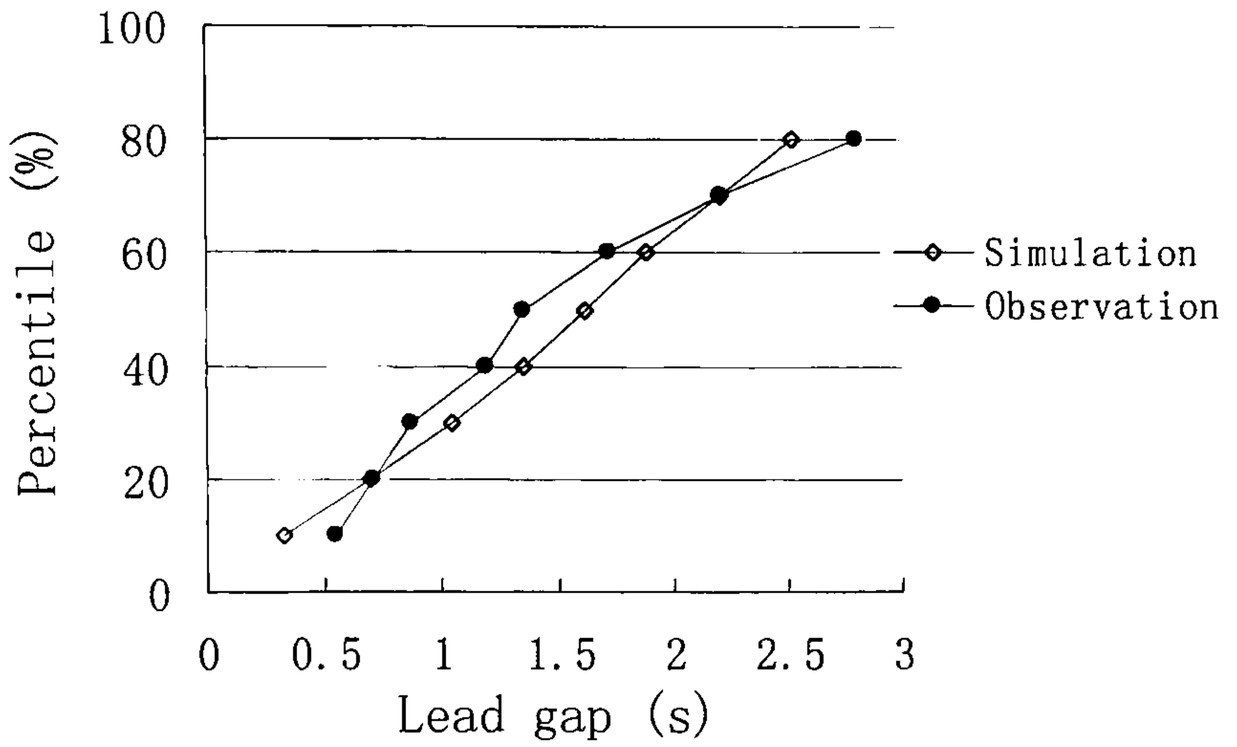
Parameters	Parameter Values
L_{acc} (m)	182
v_r^0 (km/h)	72
v_m^0 (km/h)	86
q_r^0 (veh/h)	932
q_m^0 (veh/h)	1000
α_1	6.63%
α_2	0 (courtesy yielding was not considered in Zheng's study)
HGV (%)	5

Varying the values of the reaction time and the gap acceptance factor, the simulated percentages of those merging vehicles successfully taking the original gaps (denoted as P_i^{sim}) have been obtained for each test run *i*. Thus, the calibration step can be formulated as an optimisation problem which seeks to minimise the difference between the observed percentage of successful merges using the original gap and that simulated. A range of values of β and τ_C have been tested. The set of parameter values β=0.4 and τ_C =0.4s produced P_i^{sim} = 84% which was found to be the closest to that observed (at 87%).

Based on the above calibrated parameter values and those fixed as the inputs, the modelled accepted lead and lag gaps were compared with those observed. The modelled lead and lag gaps were within 0.3 and 0.4 seconds accuracy respectively compared to the observed by Zheng (2002). Table 4-7(a) lists the observed and simulated distributions of the accepted gaps[9]; the comparison is shown in Figure 4-17. Here, the gap structures are the lead and lag gap values (measured in seconds) in cumulative percentile. Table 4-7(a) shows that the average simulated lead gap is 1.75 seconds, a mere 0.2 seconds higher than that observed, whilst the average simulated lag gap is the same as the observed one. Table 4-7(b) shows the error of the model in reproducing the real observed gap structures. It shows that, the model can reasonably capture the real observation with the mean error of 0.3s for both lead

9 The gap structure with gap values in cumulative percentile is used here for direct comparison between this study and the observed data (which was provided in percentile by Zheng (2002)).

and lag gap structures; the mean values of the absolute error are 0.16s for lead gap structure and, 0.26s for lag gap structure.



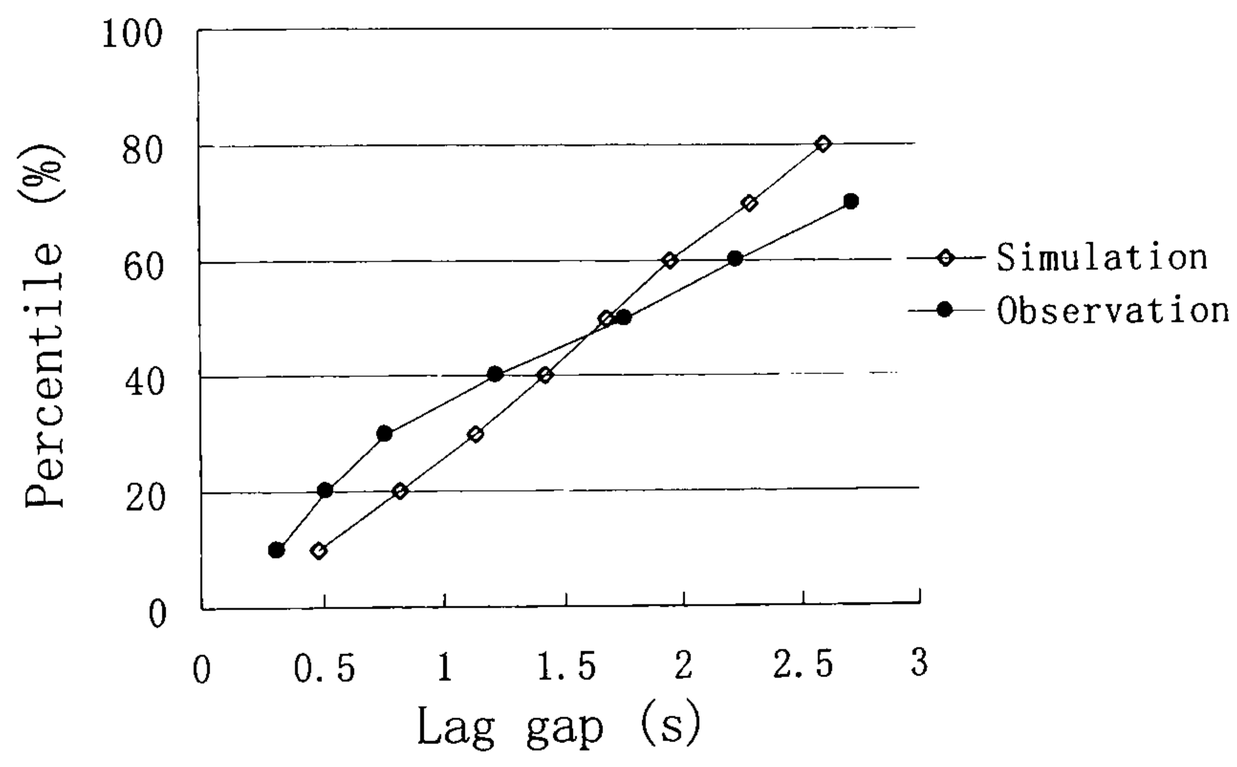


Figure 4-17 Cumulative distributions of simulated lead and lag gaps versus the real observation.

Table 4-7(a) The statistics of the simulated gap structure (<4s) versus real observation by Zheng (2002).

Time Gap		Lead (s)		Lag (s)	
		Simulation	Observation	Simulation	Observation
Observation No.		903	79	903	79
Percentiles	10	0.32	0.55	0.48	0.32
	20	0.69	0.71	0.82	0.52
	30	1.04	0.87	1.14	0.76
	40	1.36	1.19	1.43	1.23
	50	1.62	1.36	1.69	1.76
	60	1.89	1.73	1.95	2.24
	70	2.21	2.21	2.29	2.73
	80	2.53	2.80	2.61	—
	90	3.14	—	3.08	—
Average value		1.75	1.52	1.81	1.81
Std. Deviation		1.17	—	1.10	—

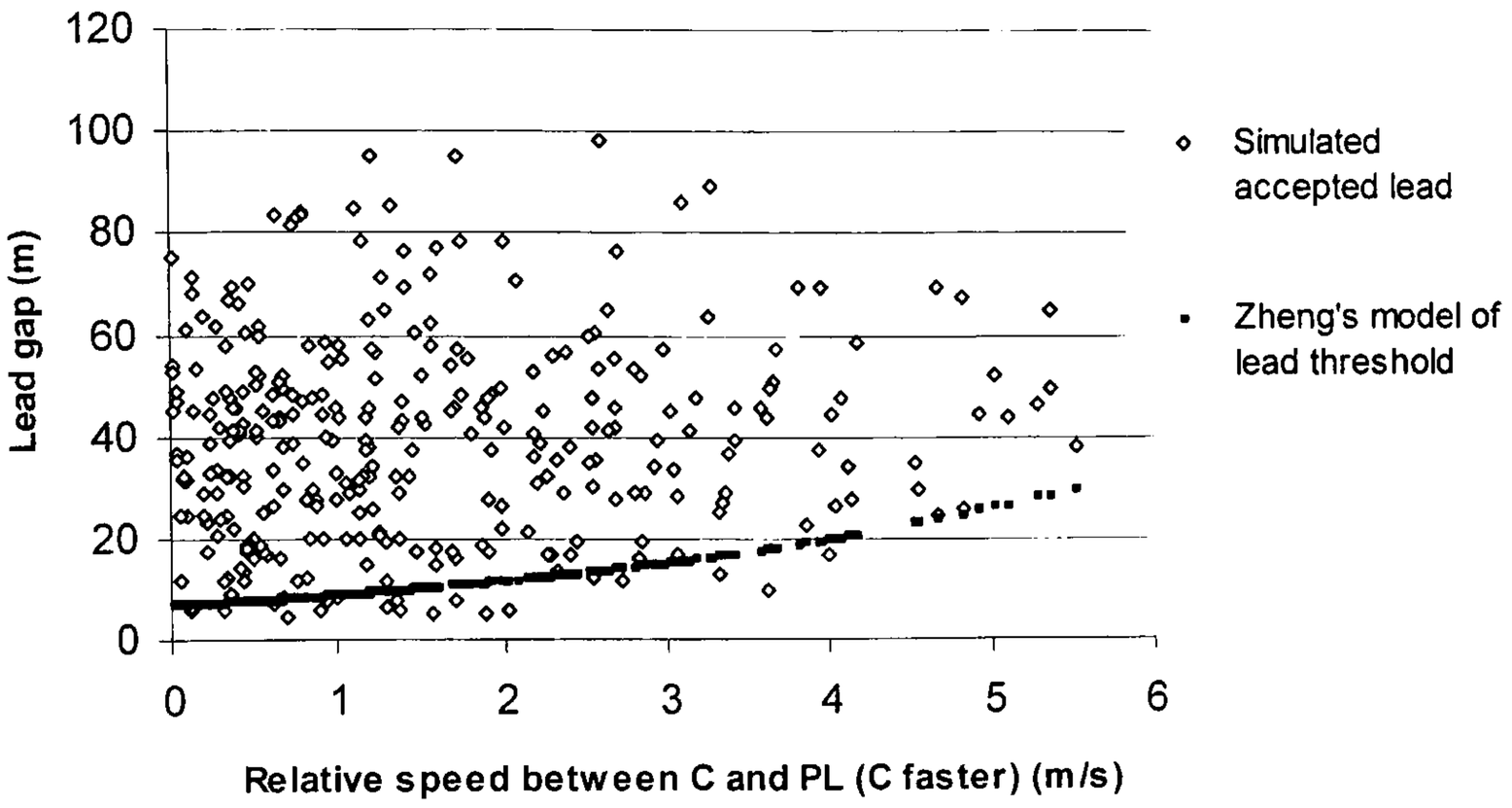
Table 4-7(b) The statistics of the error of simulated gap structure with data from Table 4-7(a).

Error in simulated gap structure (s)	Lead Gap structure(s)		Lag Gap Structure(s)	
	Error	Absolute Error	Error	Absolute Error
Mean	0.03	0.16	0.03	0.26
Std. Dev	0.19	0.10	0.31	0.12
Minimum Absolute Error	0		0.2	
Maximum Absolute Error	0.27		0.44	

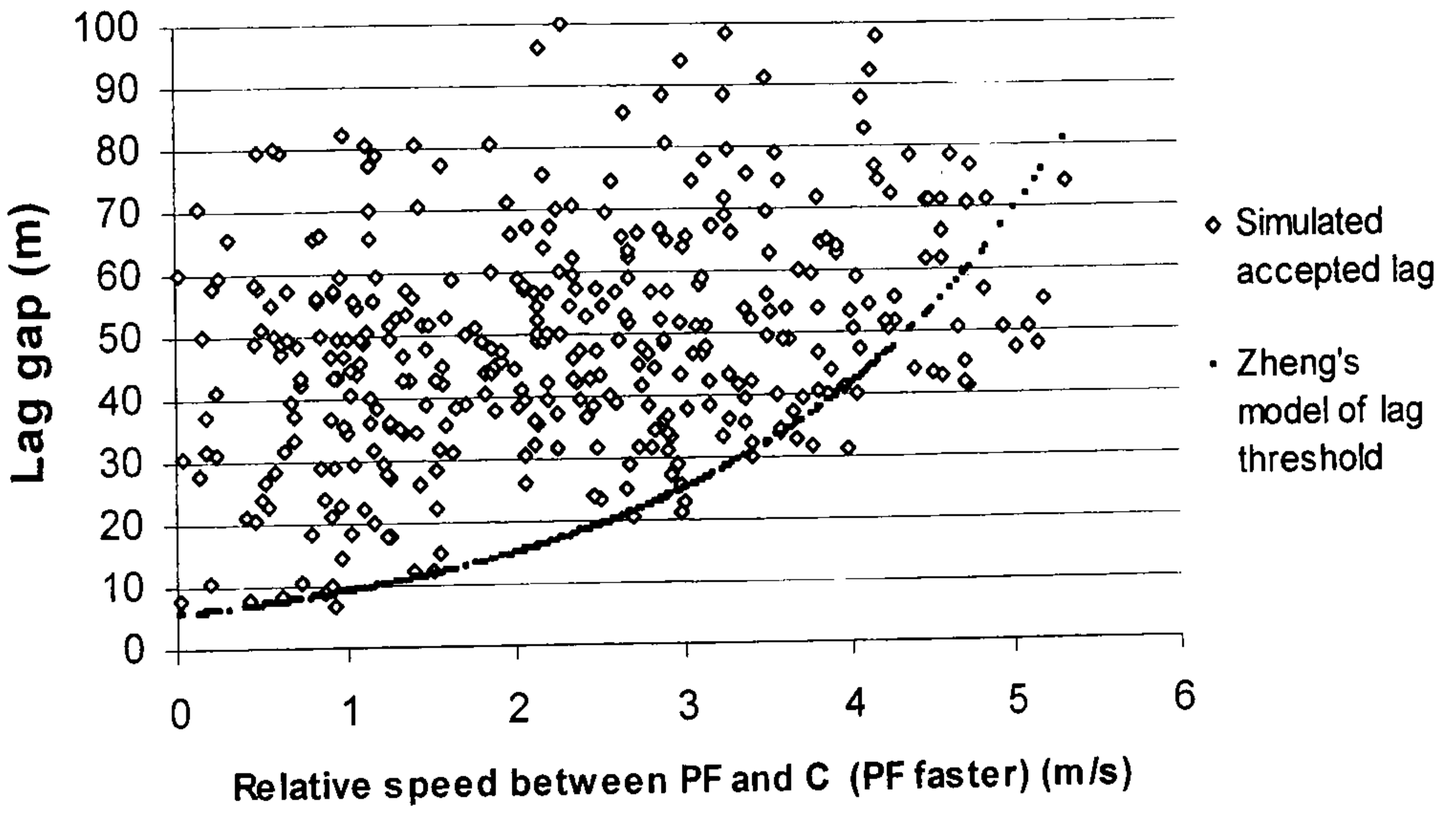
Note: The error here refers to the difference between the simulated gap values and the observed gap values in percentile given in Table 4-7 (a)

Figure 4-18 displays the simulated individual accepted lead and lag gaps as a function of the relative speeds between the merging vehicle and its PF and PL on the motorway compared to the thresholds derived by Zheng (2002) (eqs. (4-7a, b)). Most of the accepted gaps from the proposed merge model are above the thresholds

calculated by Zheng's model with only a few shorter gaps. This is because this new merging model considers factors such as the merging conditions (i.e. forecasts of the actions of vehicle PF and PL and C, the urgency effects) as well as the relative speeds considered in Zheng's gap acceptance model (eqs. (4-7a, b)).



(a)



(b)

Figure 4-18 The simulated acceptable lead gaps (a) and lag gaps (b) versus Zheng's gap acceptance model.

4.7. CONCLUSIONS

A micro-simulation model of motorway merging has been developed based on the behaviour of not only the drivers of the merging vehicles but also those on the motorway. To mimic the dynamic interactions between the motorway and merging vehicles, the model includes the gap selection, acceleration and deceleration behaviour of the merging traffic on the acceleration lanes, as well as the cooperative behaviour of the motorway traffic in terms of gap creation and lane changing.

Sensitivity tests have been carried out on key model parameters. The results suggest that the model responds well to changes in these parameter values:

Sensitivity analysis of the model suggests that cooperative lane-changing and yielding have a significant effect on the proportion of merges that are made easily. Similarly, the more alert the drivers are, the higher the proportion that successfully merge into their first choice of gaps. A longer acceleration lane does not appear to increase the proportion that successfully merges into the driver's first choice of gaps; it does however help when merging into the second or third choices.

Simulation tests on a standard UK merge section were carried out and the results compared with those observed. A preliminary calibration of the model suggests that the modelled gap distribution fits well with those observed.

In the next chapter, the calibration and validation of this merge model combined with the new car-following model (chapter 3), will be further discussed through a case study of Junction 27, Eastbound of motorway M8, UK.

CHAPTER 5
CASE STUDY

This case study provides the calibration and validation of the integrated simulation model in the developed MergeSim program using loop detector data and video observations extracted from Junction 27, eastbound of M8 Motorway, Glasgow. The integrated model, including the car-following and merging models, is applied to simulate the interactive behaviours between the on-ramp merging traffic and nearside motorway traffic. The model calibration is carried out through multiple simulation runs as an optimisation process. Model validation is executed which is aimed to examine the validity of the calibrated model.

5.1. DATA SOURCE

5.1.1. Site Description

When studying the on-ramp merging behaviour, the site selection is critical. The section downstream from the merging area should be free of constraints such as roadworks, traffic accidents, and geometric design deficiencies, etc. As this study does not include the operational effect of ramp metering, there should be no metering control on the selected site. The M8 motorway through Glasgow carries both long-distance traffic within Scotland and local traffic between the eastern and western sectors of the city of Glasgow (Figure 5-1).

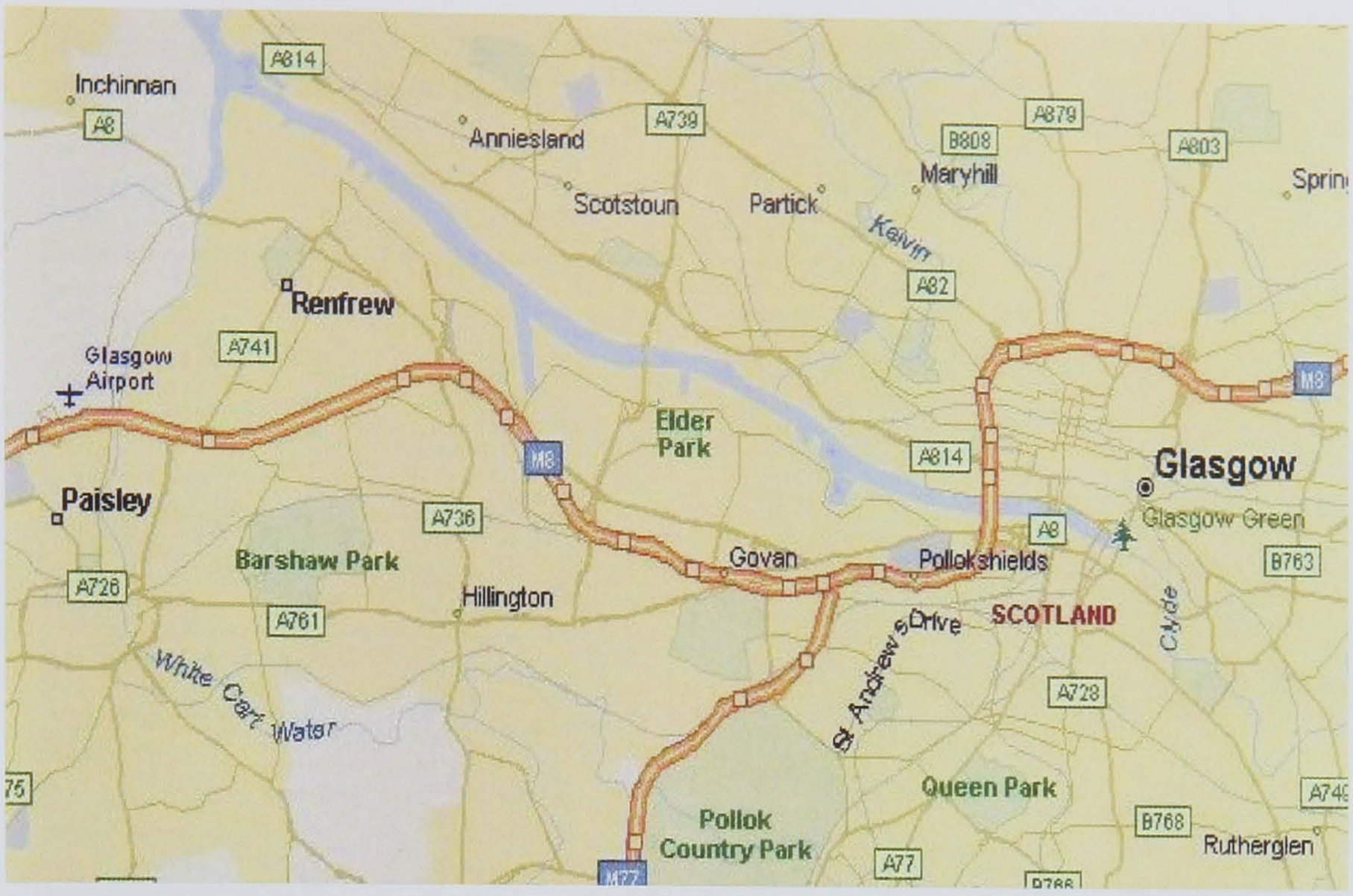


Figure 5-1 Glasgow's motorway M8 (map produced with Microsoft MapPoint Europe) .

Junction 27 on M8 motorway is selected here for the case study. This selected site is near Arkleston Road, with Glasgow airport not far away. It has been reported by Diakaki et al. (2000) that "*during peak hours, local traffic interacting with through motorway traffic at the junction causes major delays to motorway traffic.*" The available data source from M8, provided by NADICS (*National Driver Information and Control System, Scotland*), includes video and loop detector information. At this site, the one-lane merging traffic joins the three-lane main carriageway through a 147 metre long taper merge which is comparatively longer than DMRB (2001a) designed merging lengths between 90 m and 130 m. A section of 1450m of eastbound motorway (Figure 5-2) is selected with one detector located on the on-ramp slip road (detector 8498EI) and another three detectors covering motorway sections upstream (detector 8507E) and through the downstream area (detector 8464E and 8362E). There was no metering control on this site. A schematic of the selected site is also given in Figure 5-3. The speed restrictions are set on the NADICS gantries (Figure 5-2, e.g. gantries ID: 01880, 01890) across the motorway carriageways during peak hours.

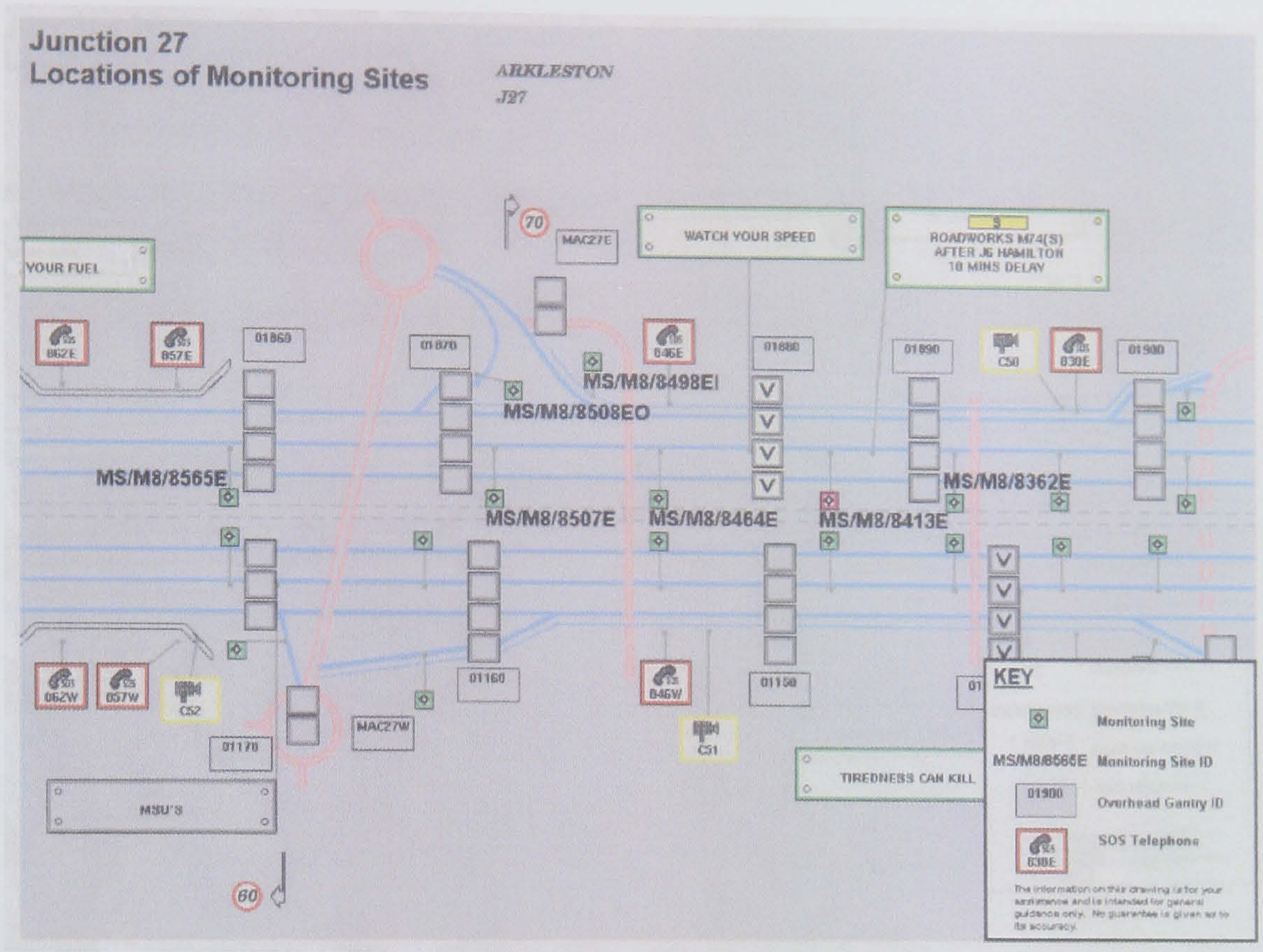


Figure 5-2 The site layout Junction 27, M8 motorway , Glasgow (not to scale) (courtesy of NADICS, Scotland).

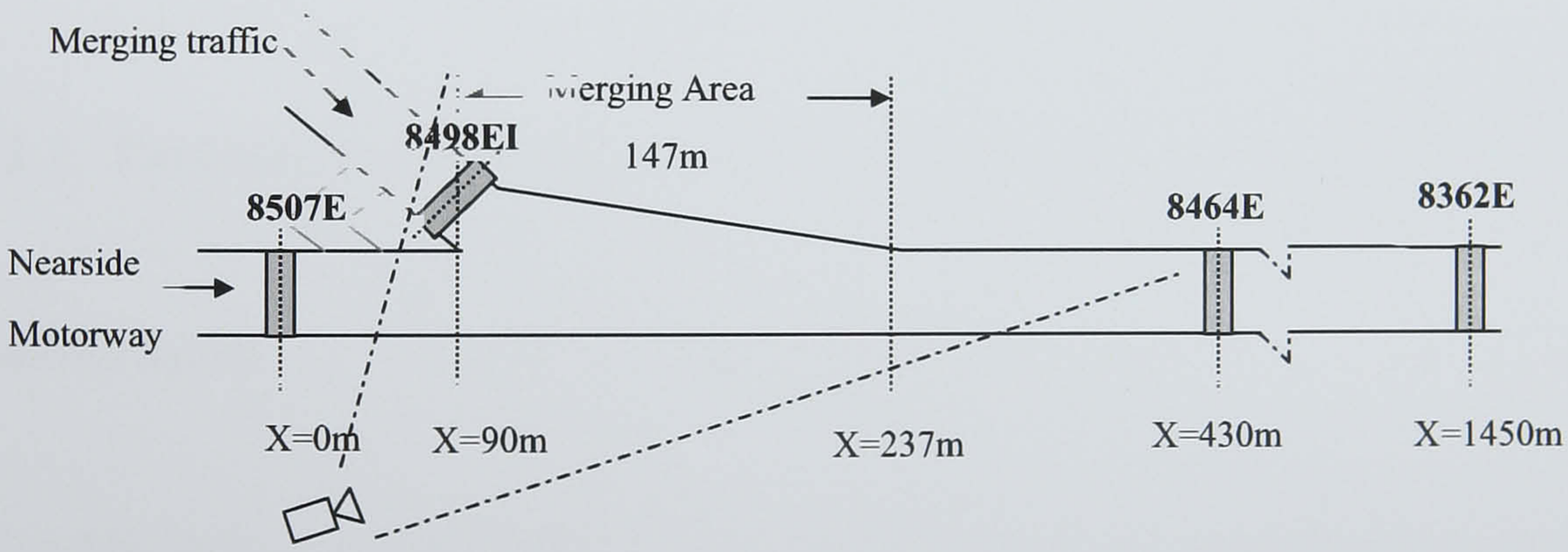


Figure 5-3 A schematic of Junction 27, M8 motorway. The section starts from an arbitrary upstream point of the site where detector 8507E is located, denoted X=0; all positions are measured relative to this point.

5.1.2. Data

The available information provided by NADICS for this study, was collected at the eastbound carriageway of Junction 27, M8 motorway, Glasgow, on Thursday 25th July 2002. It includes:

- Junction 27 layout with detector and video camera locations;
- 3-minute detector data including speed, flow, occupancy and vehicle compositions (i.e. Car and HGV).
- Video recordings covering upstream, through the merge and the downstream area over a length of around 200 meters, between 08:00 and 09:00 in the morning and between 15:00 and 16:00 in the afternoon;
- Speed restrictions information: between 8:00 and 9:00 with the advisory speed of 50 mile/h (80 km/h). However, no speed restrictions were implemented between 15:00 and 16:00, therefore, the national speed limit of 70 mile/h would be in place.

5.2. MODEL CALIBRATION

5.2.1. Parameters Calibrated from the Field Data

Desired Speed

The data collected during 8:00-9:00 was used for model calibration. During this studied period, speed restrictions were implemented with the advisory speed of 50mile/h (i.e. about 80 km/h) which had effects on the drivers' desired speed. Figure 5-4(a) shows the speed-time profiles collected during 8:00 to 9:00. It can be found that all the collected speeds lie between 78 to 88 km/h. The speeds collected at detector 8362E (further downstream of the merging section) are slightly higher than those from detectors upstream (8507E) and immediately downstream (8464E) of the merge (Figure 5-3). This may be because the merging traffic causes disturbance to the nearside motorway traffic in the vicinity of merging area.

The driver's desired speed here, i.e. under the implementation of speed restriction during the study period, was calibrated from the data collected from detector 8362E, which was not disrupted by the merging traffic.

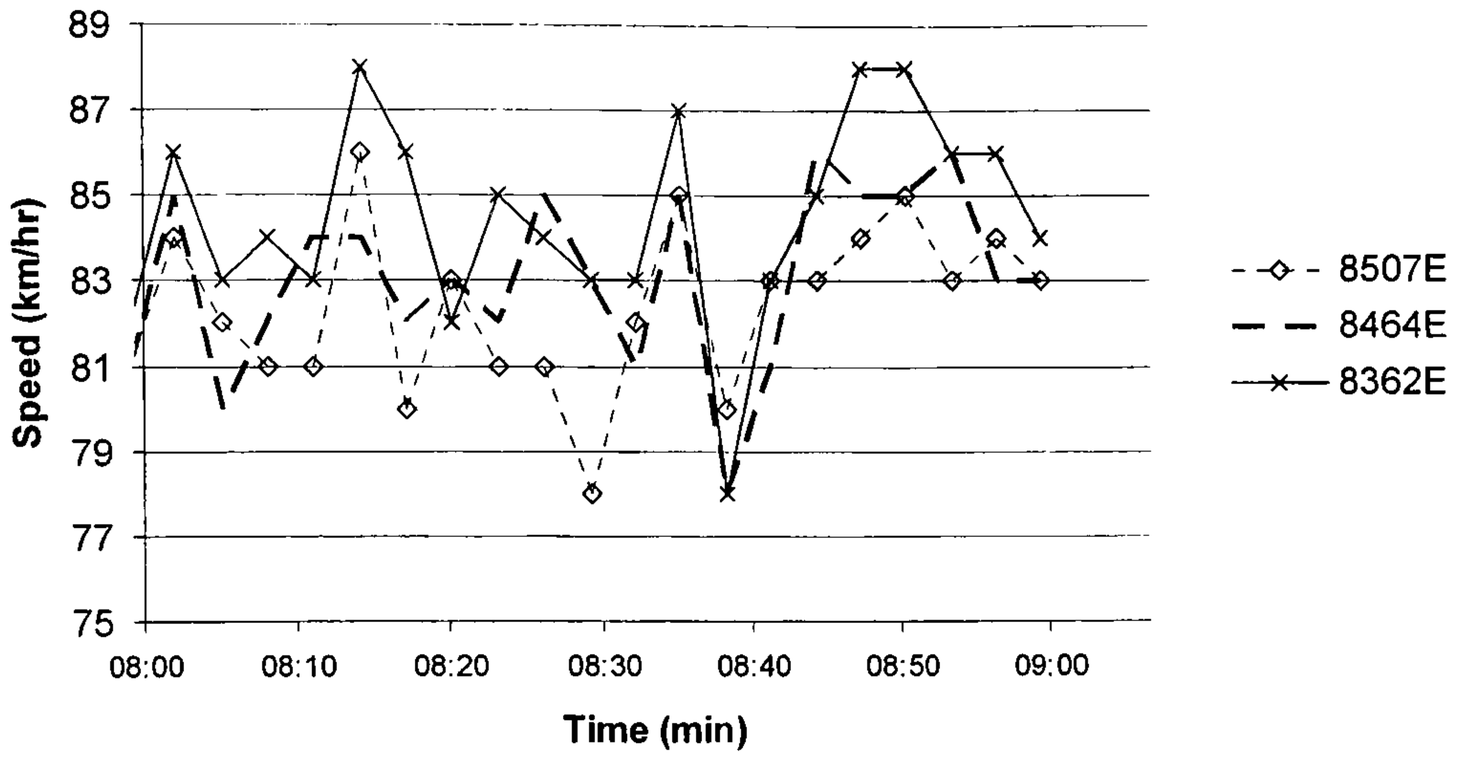


Figure 5-4 The speed profiles collected by detector 8362E between 08:00 and 09:00 on 25th July 2002.

Figure 5-5 shows the speed profile by 15-minute intervals collected by detector 8362E between 8:00 and 9:00. It is found that the average speeds vary between 82.8 and 86.2 km/h, i.e. the average speeds vary within a range of 3 km/h (0.8 m/s) (Table 5-1). Given that there are only about 5 observations per 15 minute interval, the apparent speed differences between the intervals were assumed to be random and normally distributed during the whole studied period as $N(84.5, 2.4^2)$ km/h (Table 5-1).

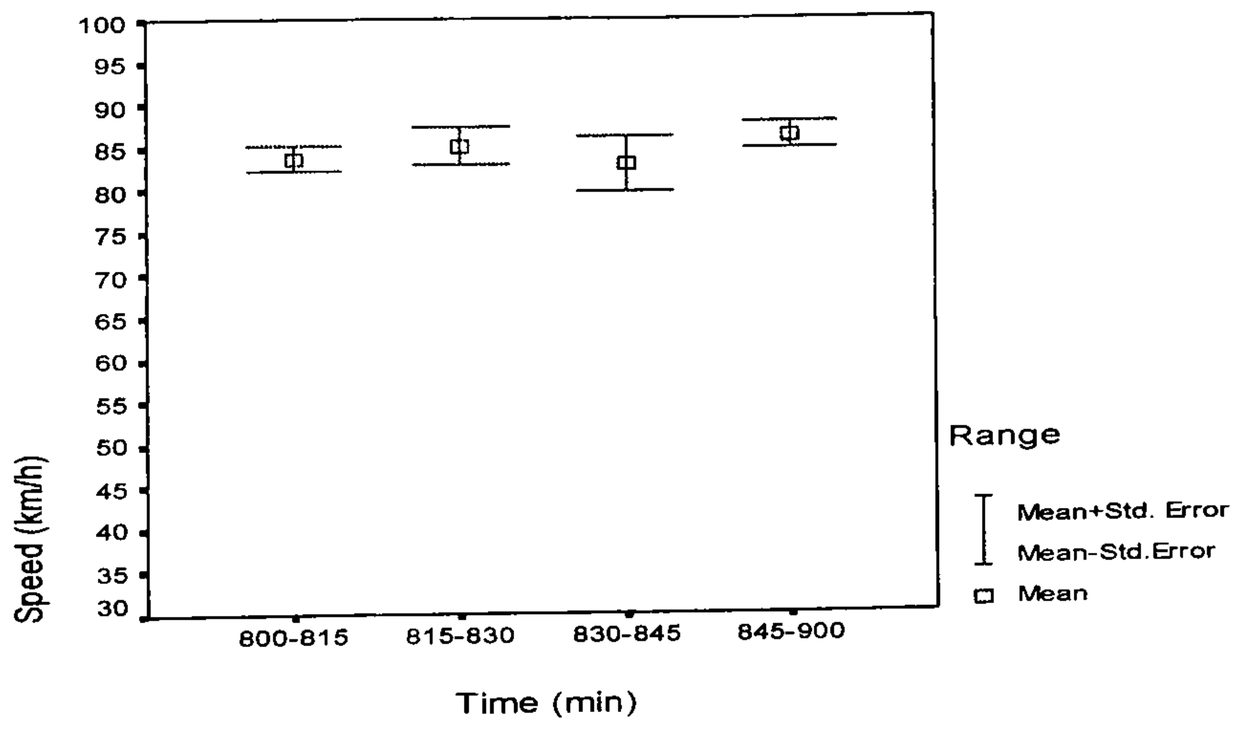


Figure 5-5 speed of nearside motorway traffic during 8:00 to 9:00.

Table 5-1 The statistics of the desired speeds (km/h) between 8:00 and 9:00.

Time Interval	Mean	Std.Error
8:00-8:15	83.6	1.5
8:15-8:30	85.0	2.2
8:30-8:45	82.8	3.2
8:45-9:00	86.2	1.6
8:00-9:00*	84.5	2.4
* The values applied in the simulation.		

Gap Selections of the Merging Traffic and Cooperations of the Motorway Traffic

A comprehensive observation covering both the merging and motorway traffic is useful to analyse the interactive behaviour between the two streams. Using the terms given in Chapter 4, a manual counts from the video observations of Junction 27, M8 motorway was carried out. The results revealed that most drivers (88%) were able to merge into the original gap, while the remainder (12%) were overtaken by the motorway drivers (thus necessitating acceptance of the following gap). No observations of the merging traffic were observed of merging drivers overtaking the motorway drivers to catch up the previous gap. These results are consistent with the observations made on M27 motorway, UK, by Zheng (2002) who found most drivers (87%) were able to merge into the original gaps. Cooperative behaviour provided by the motorway drivers was also observed. From the video tapes, courtesy yielding from the motorway drivers can be identified from the perceivable headway increase accompanied by signalling [10] (i.e. high beam flash) to the merging vehicle. Cooperative lane-changing can be easily identified from videos with those vehicles changing to the inside lanes to facilitate the merging vehicles to enter into the motorway. It was found that out of 302 observed merges, 12% involved courtesy yielding and 7% cooperative lane-changing. Table 5-2 summarises the information

---

10 Here, the signal for courtesy refers to the high beam flash to ramp drivers. In the UK, the high beam flash is widely used to signal to drivers ahead to merge or to cross the road. Also this flash is used when a car stops to let pedestrian cross the road (www.slowtrav.com/uk/instructions/driving.htm, www.motorcycle.co.uk/articles/visibility/twowheels2.html).

extracted from the video observation between 8:00 and 9:00 compared to Zheng's observations (2002).

In the case study, the observed percentage of courtesy yielding and cooperative lane-changing will be used as input parameter values for model calibration. The observed percentage of taking the original gap together with the loop detector measurements (speed and flow) will be used as the indicators for model calibration.

Table 5-2 The information extracted from the video observation during 8:00-9:00am, Junction 27, M8, Glasgow on 25th July, 2002.

Extracted Information		M8 observation	Zheng's observation (2002)
Merging Traffic	Original Gap	88%	87%
	Previous Gap	0	4%
	Following Gap	12%	9%
Motorway Traffic	Courtesy yielding	12%	0 (Not considered)
	Cooperative lane-changing	7%	6.63%

5.2.2. Calibration Method

The intention here is to calibrate the integrated simulation model which includes the two combined models: car-following model and merging model. It is conducted on the merge section with the simulation configuration (Figure 5-6) according to the site layout properties given earlier in Figure 5-2. Only the merging traffic on the acceleration lane and the traffic on the nearside lane of the motorway are simulated in the study. It is assumed that the traffic on the other lanes of the motorway will not change to the nearside lane in the merging area.

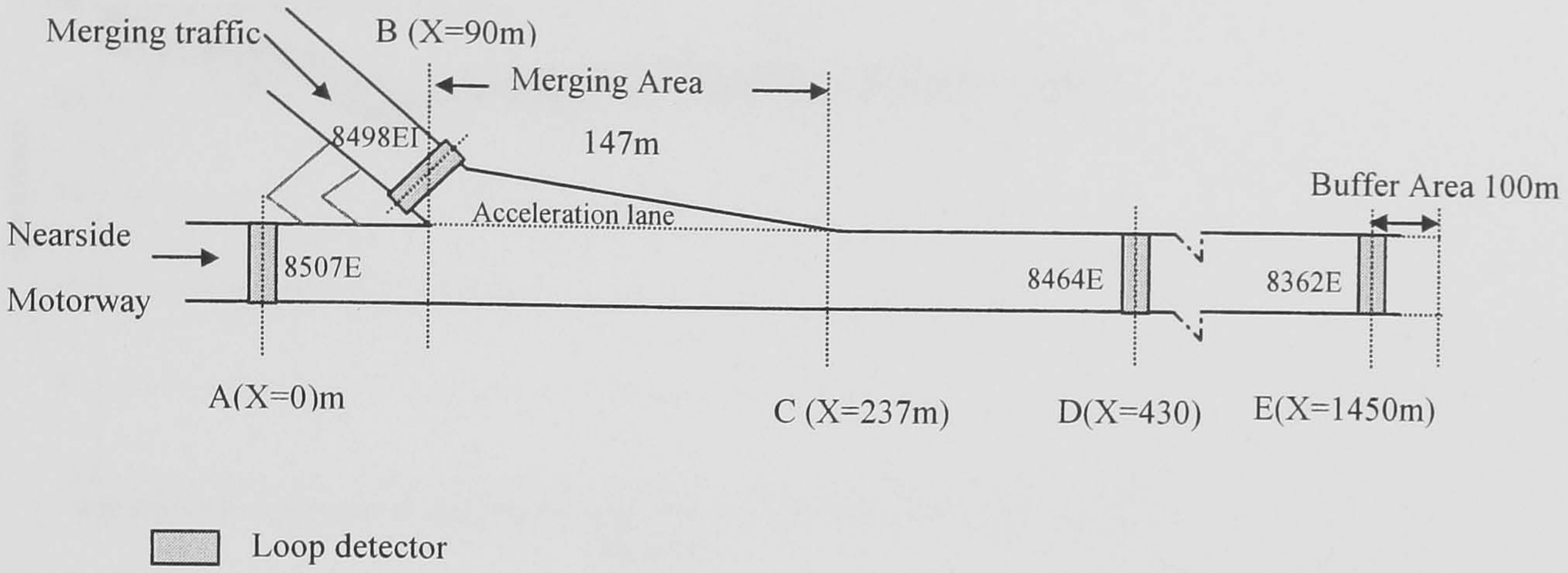


Figure 5-6 Simulation configuration of the selected site of Junction 27, selected site on M8.

The calibration is conducted over a 1450 metre long motorway merging section illustrated in Figure 5-6. The simulation starts from an arbitrary point A with X=0 where detector 8507E is located. Motorway traffic arrives randomly at the simulation section (point A) according to the vehicle generation algorithm discussed in Chapter 2, based on the dynamic speed v_T^A and flow q_T^A profiles collected from detector 8507E. Similarly, merging traffic arrives randomly at the acceleration lane at point B based on the dynamic speed v_T^B and flow q_T^B profiles collected from detector 8498EI. Considering the constraints of the "follow-the-leader" rule, an extended 100 metre long buffer area (explained earlier in Chapter 3) is attached to the end of the simulation section, i.e. the buffer area starts from 1450 metre and ends at 1550metre. At X=1450m, motorway traffic is generated based on the data from detector 8362E (point E) with the dynamic flow profile q_T^E and speed profile v_T^E. In the simulation, the flow and speed are collected at point D and compared to the real observation from the corresponding location of detector 8464E.

The speeds collected by the three motorway detectors between 6:00 and 12:00 noon are all above 50km/h (Figure 5-7). According to the car-following model developed in Chapter 3, the traffic of the motorway nearside lane are thus in the non-alert car-following state during the studied period. Therefore, this dictates that with respect to the car-following behaviour the motorway traffic can only be calibrated for the non-alert state.

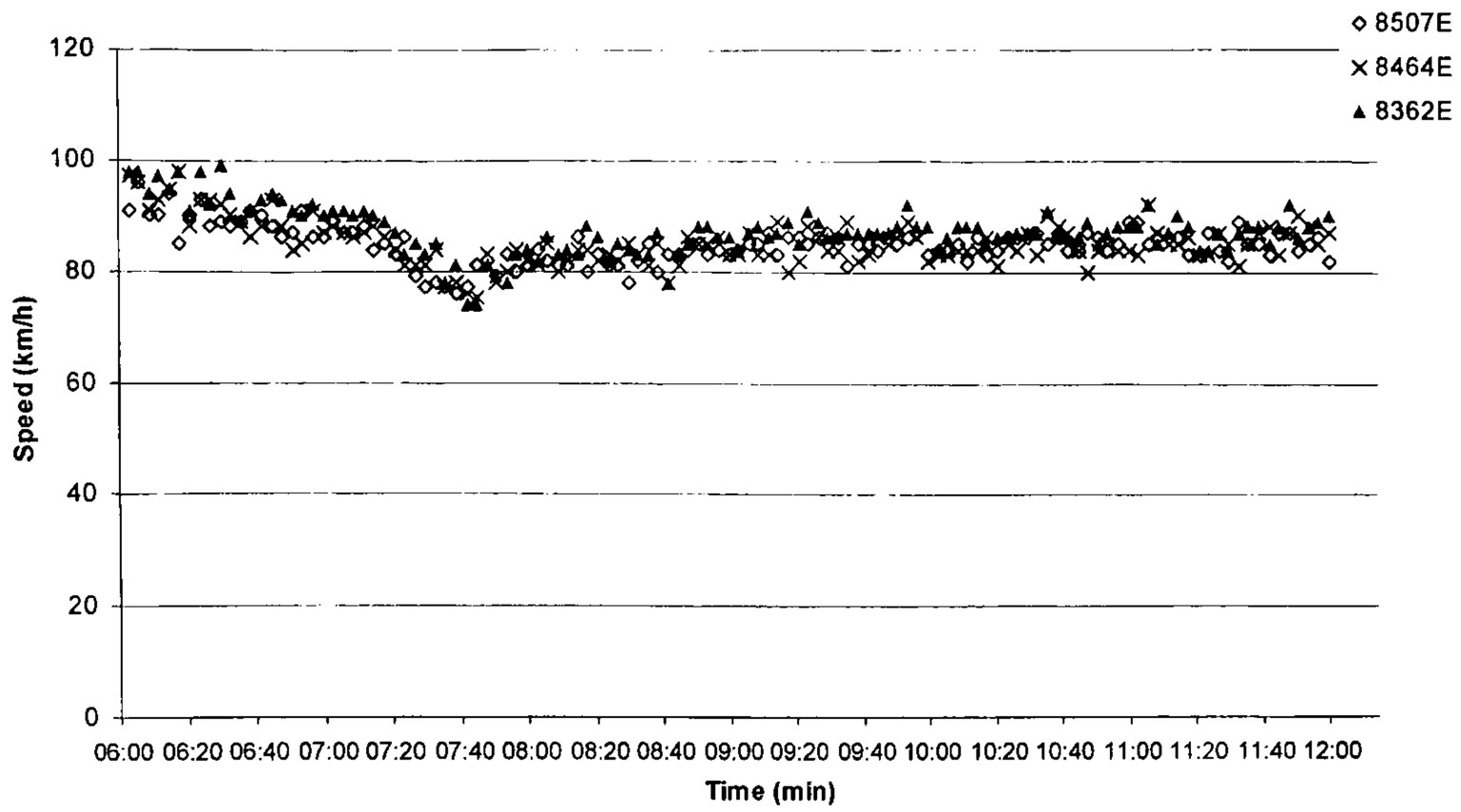


Figure 5-7 The speed-time profile collected by 8507E, 8464E and 8362E on 25th July 2002.

With data collected in the morning period between 8:00 and 9:00, on 25th July 2002, the following parameters of the integrated simulation are calibrated:

- The reaction time τ_2, and acceleration A_2 for the non-alert state in the car-following model; and
- The merging reaction time τ_C and gap acceptance factor β in the merge model.

An example of the optimisation dialogue is shown in Figure 5-8. Data between 15:00 and 16:00 of the same day are used later for model validation. The validation aims to compare the simulated results with observed data on traffic speed, flow at detector 8464E and gap selections.

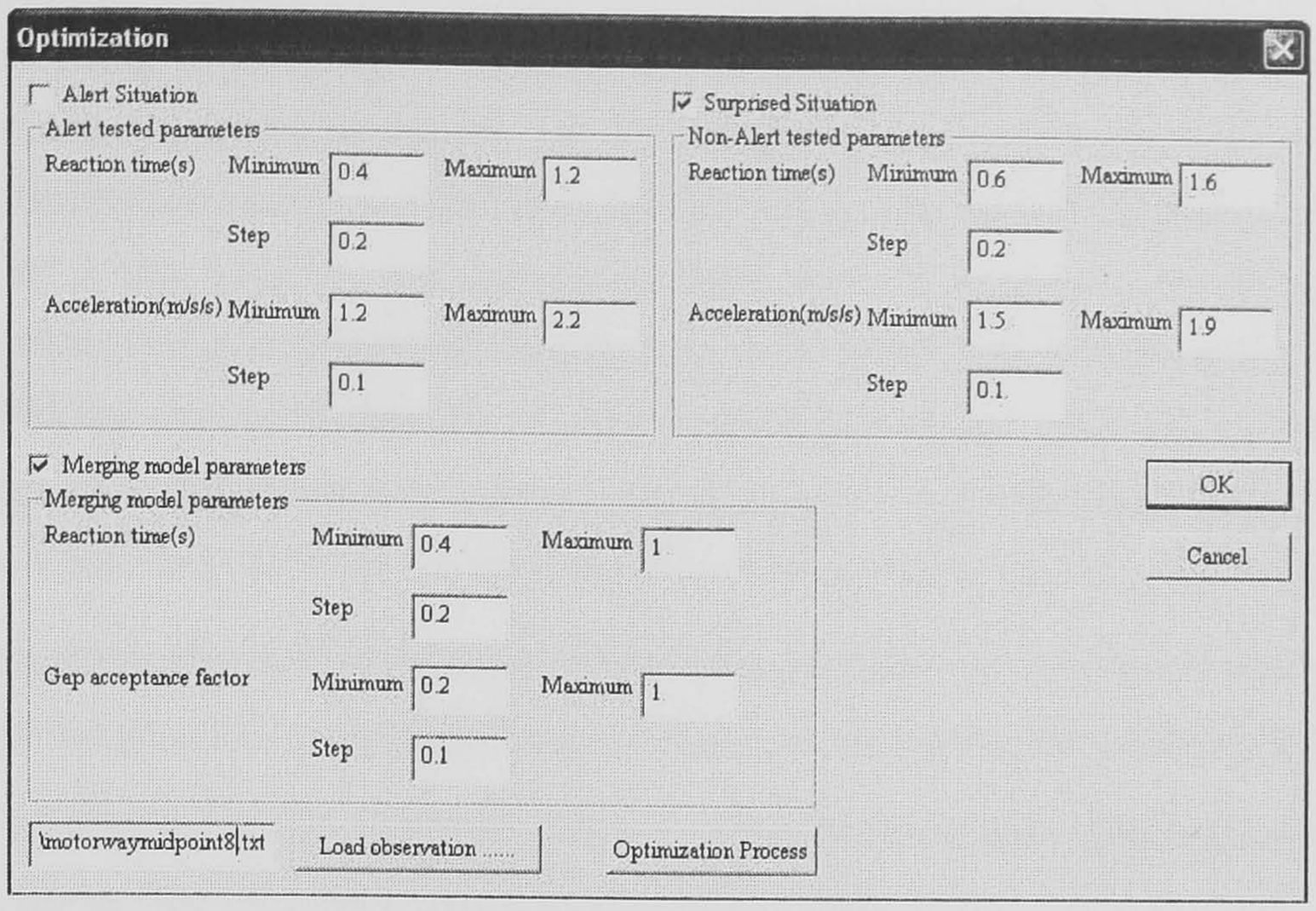


Figure 5-8 The optimisation dialogue in MergeSim program.

5.2.3. Model Calibration

As explained earlier in section 5.2.2, with respect to the car-following model, only the parameters of non-alert following state are calibrated here. Table 5-3 lists the range of values for the calibration of the integrated model of car-following and merging models.

Table 5-3 Model parameters for the calibration of integrated simulation.

Model	Parameters		Notation	Test Range	Increment
	Set	Parameters included within each set			
Car-following model	*I*	Reaction (s)	τ_2	0.6-1.6	0.2
		Acceleration (m/s^2)	A_2	1.5-1.9	0.1
Merging model	*J*	Merge reaction time (s)	τ_C	0.4-1.0	0.2
		Gap acceptance factor	β	0.2-1.0	0.1
Note: The literature for these test ranges are summarised earlier in Tables 3-19 and 4-4.					

According to the range of values tested (Table 5-3), a total of 1080 different combinations of parameter values, i.e. 1080 independent simulation runs are needed in the optimisation process. Similar to the car-following model calibration discussed in Chapter 3 (eq. (3-23)), the calibration here aims to minimise the discrepancies between observed and simulated measurements on speed and flow profiles. However, in considering the performance of the merging operation, another indicator (the percentage of merging drivers taking original gaps) is also included here. As given earlier in Table 5-1, 88% of the merging drivers were observed take the original gaps.

Therefore, the optimisation problem here includes two aspects and Figure 5-9 illustrates the general calibration framework for the integrated simulation model.:

1 Minimise the discrepancies between the observed and simulated speeds and flows (the objective function eq. (5-1a)), and

2 It is acceptable if the differences between modelled and observed smooth merging percentages, i.e. those merging vehicles taking the original gap are within the range of 5%, as given in the Inequality (5-1b).

$$F_{I,J}(v,q) = \sum_{t=1}^{T} [(\frac{v_t^{sim} - v_t^{obs}}{v_t^{obs}})^2 + (\frac{q_t^{sim} - q_t^{obs}}{q_t^{obs}})^2] \qquad (5\text{-}1a)$$

$$F_{I,J}(P) < 5\%, \qquad F_{I,J}(P) = \left| P_{I,J}^{sim} - p^{obs} \right| \qquad (5\text{-}1b)$$

where:

v_t^{sim} and v_t^{obs} the simulated and observed speeds, respectively during time period t;

q_t^{sim} and q_t^{obs} the simulated and observed flows, respectively during time period t;

T the calibration period (1 hour in this study);

t the aggregate time interval (3 minutes in this study).

$P_{I,J}^{sim}$ the simulated percentages of those merging vehicles successfully taking the original gaps;

P^{obs} the observed percentages of those merging vehicles successfully taking the original gaps.

I car-following parameter set to be calibrated

J merge parameter set to be calibrated

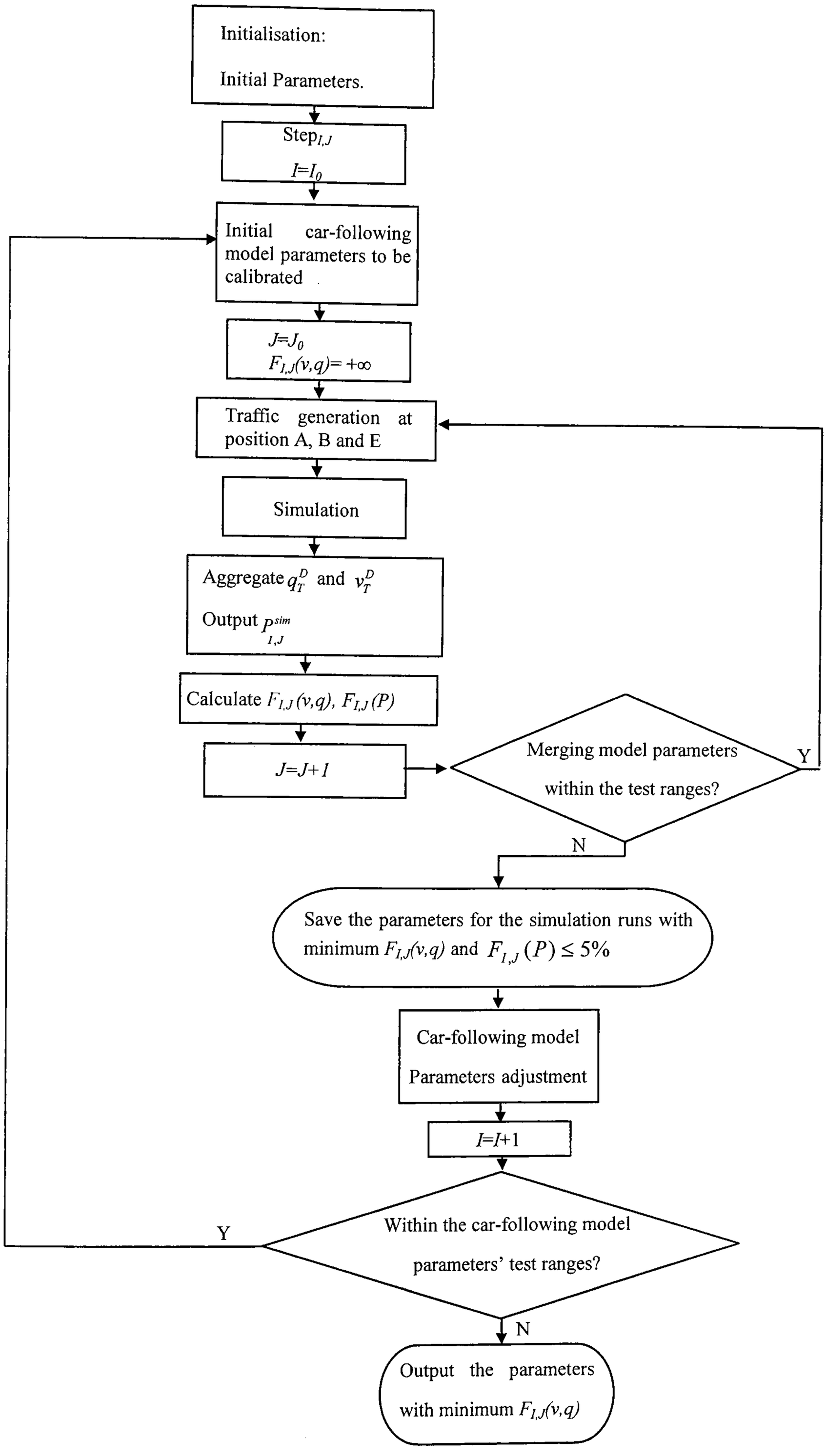


Figure 5-9 General calibration framework.

As shown in Figure 5-9, by initialising the car-following parameters ($I = I_0$) the merge model parameters set of J are firstly tested within the ranges, as given in the Table 5-3. During each simulation run with J, $F_{I_0,J}(v,q)$ and $F_{I_0,J}(P)$ are calculated and only those with $F_{I_0,J}(P) \leq 5\%$ are used for later comparisons. When the full range of the set J of merging parameters have been tested with fixed set I of car-following parameters, the calibration process moves to the next simulation loop with adjusted car-following parameters set, i.e. $I=I+1$.

Table 5-4 lists the ten best simulation outputs from the optimisation process. It shows that with merging reaction time τ_C=0.4s, β=1 and, car-following non-alert state reaction time τ_2=1.6s, A_2=1.9m/s^2, the minimum $F_{I,J}(v,q)$ was 0.1615 and with $P^{sim}_{I,J}$ within the level of 5%. Thus the calibrated parameters set for the integrated simulation model is τ_C=0.4s, β=1, τ_2=1.6s and A_2=1.9m/s^2.

Table 5-4 Ten best simulation outputs of the optimisation process.

Parameters to be calibrated				Optimisation indicators	
τ_C (s)	β	τ_2 (s)	A_2 (m/s^2)	$F_{I,J}(v,q)$	P^{sim} (%)
0.4	1.0	1.6	1.9	0.1615	83
0.4	0.9	1.2	1.8	0.1731	84
0.4	0.8	1.0	1.5	0.1798	84
0.4	0.7	1.2	1.7	0.1947	84
0.4	0.5	0.6	1.7	0.2045	83
0.8	0.2	1.6	1.9	0.2145	87
0.4	0.4	1.6	1.5	0.2177	87
0.6	0.3	1.2	1.6	0.2359	85
0.6	0.2	1.0	1.8	0.2409	88
0.4	0.3	1.2	1.6	0.2521	87

5.2.4. Model Validation

Video and loop detector data from 15:00 to 16:00 on 25th July, 2002 are used for the validation. From the video observation, out of 279 observations of merging process during this period, 3% involved courtesy yielding and 13% cooperative lane-changing. The amount of courtesy yielding and cooperative lane-changing are slightly different from those observed in the morning peak; both sets of results are listed in Table 5-5. The results show that during the afternoon off-peak hour, the motorway drivers are more likely to change into inner lanes rather than yield to create gaps, compared to morning peak hour's observation. This may be because in general most cooperative motorway drivers are more inclined to change lanes rather than yielding (as the former does not involve a loss of speed). However, during the morning peak hour, the opportunities to change into inner lanes are limited due to an overall increase of the traffic flow (Yousif and Hunt, 1995).

During this validation period, no speed restriction (i.e. advisory speeds) was implemented according to NADICS. The desired speed applied here is thus obtained from the extracted free-flow speed given earlier in Chapter 2 (Table 2-4).

Table 5-5 The motorway drivers' cooperation statistics during different observed period.

Observation Period	Courtesy Yielding	Cooperative Lane-changing
15:00-16:00	3%	13%
08:00-09:00	12%	7%

It was also found that most drivers (84%) were able to merge into the original gap while few drivers (16%) took the following gap. Out of 279 observations of the merging process, in only one did the merging drivers overtake to catch up the previous gap. According to the calibration with the data from 08:00 to 09:00, the best fit parameter set is with the parameters of τ_C=0.4s, β=1, τ_2=1.6s, A_2=1.9 m/s^2 (section 5.2.3). By applying these values, and the courtesy yielding and cooperative lane-changing percentages during 15:00 and 16:00 (given in Table 5-4), the validation is to examine the simulated and the observed measurements of speed and flow at detector 8464E and gap selections (the percentage of merging drivers taking original gaps).

The simulation outputs show that the percentage taking original gaps is 82% (comparing to 84% observed). Figure 5-10 compares speed and flow profiles between the simulation and observation. It is found that the simulation slightly over-estimates the flow and speed. The simulated average flow is at 1060 veh/h, at 9% above the observed average flow (968 veh/h). The simulated average speed is at 88 km/h, 2% above the observed average speed (86 km/h).

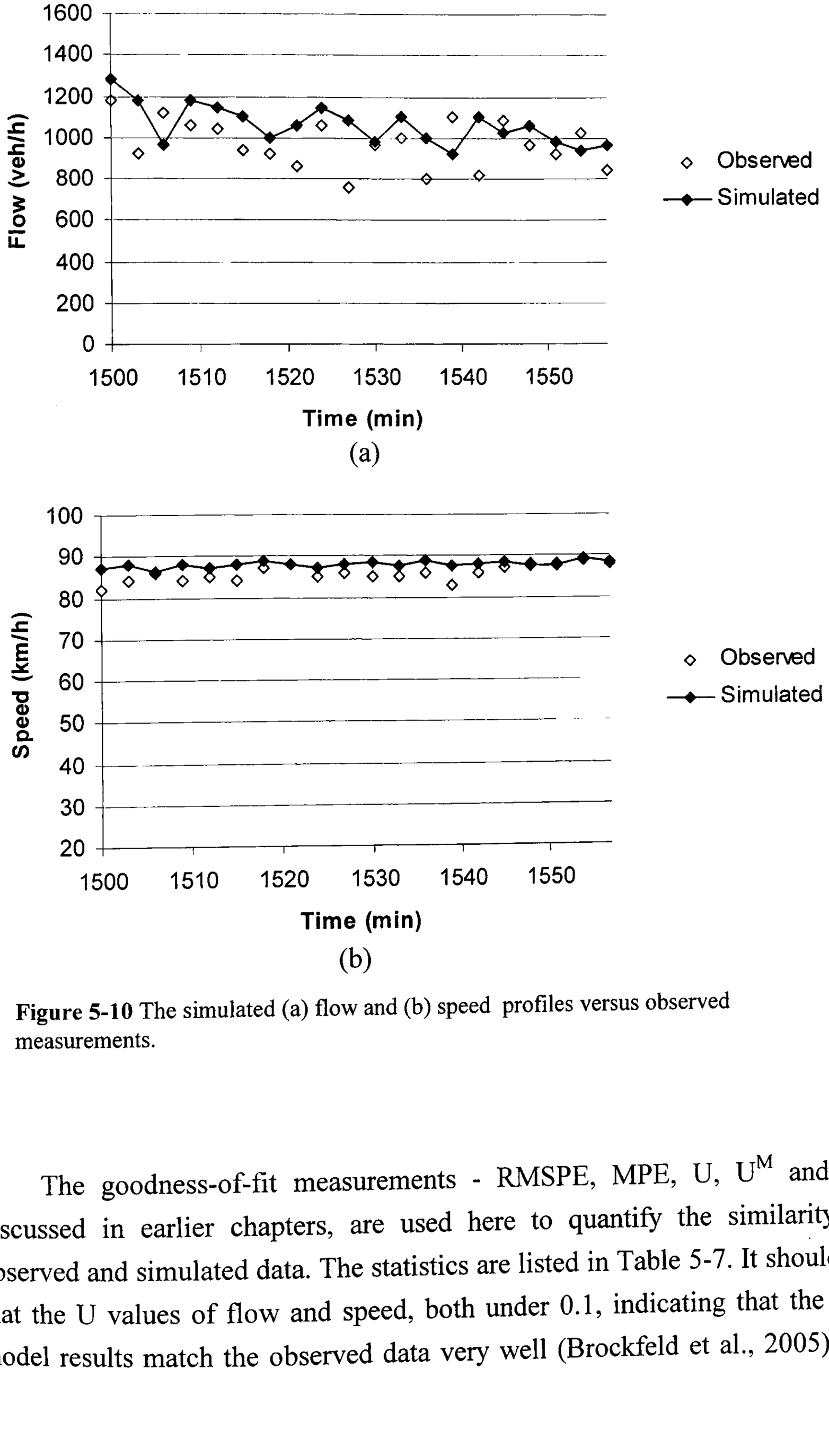


Figure 5-10 The simulated (a) flow and (b) speed profiles versus observed measurements.

The goodness-of-fit measurements - RMSPE, MPE, U, U^M and U^S – as discussed in earlier chapters, are used here to quantify the similarity between observed and simulated data. The statistics are listed in Table 5-7. It should be noted that the U values of flow and speed, both under 0.1, indicating that the calibrated model results match the observed data very well (Brockfeld et al., 2005). The bias

(U^M) values are small (< 0.3) and comparable to the results from a similar work by Toledo (2003) which has U^M values between 0.1 and 0.3. The variance U^S value of the new model for speed and flow are both below 0.2, suggesting a good fit. For the measure of MPE, it is found that flow and speed are slightly over-predicted with mean percent error of 10.73% and 2.52% respectively. The RMSPE values for speed and flow are comparable to similar work by Toledo (2003) which was 14%.

According to DMRB flow acceptability criteria for a single link, for the observed flow between 700 and 2700 veh/h, the validation is acceptable if the difference between modelled and observed flow is within 15% of the observed flow (DMRB, 2001b). As the simulated flow in this study is only 9% above the observation, this validation meets the DMRB flow criteria.

Table 5-6 Statistics for the simulated and observed data.

Goodness of Fit Measure	Flow	Speed
U	0.08	0.02
U^M	0.25	0.30
U^S	0.02	0.19
MPE (%)	10.73	2.52
RMSPE (%)	18.22	3.23

The goodness-of-fit statistics (Table 5-7) show that the model can reasonably simulate flow and speed. It is also found that the modelled percentage taking original gaps is 82% comparing to 84% from observation.

5.3 Further Investigations of the Selected Site

As discussed in above sections of this chapter, it shows that the integrated model if well calibrated and reasonably replicate the studied site conditions (traffic speed, flow and gap acceptance structure, etc.). Therefore, in this section, the calibrated integrated model is used for further investigations into the effectiveness of the ramp geometry layout and its potential in coping with varied traffic flows of this selected site (i.e. Junction 27, M8).

Figure 5-11 illustrates the arrival flow of the ramp road and motorway nearside lane of this studied site. It can be found that generally the merging traffic is heavier (around 200veh/h higher) than the nearside motorway traffic during the studied period (i.e. 8:00 to 9:00 am).

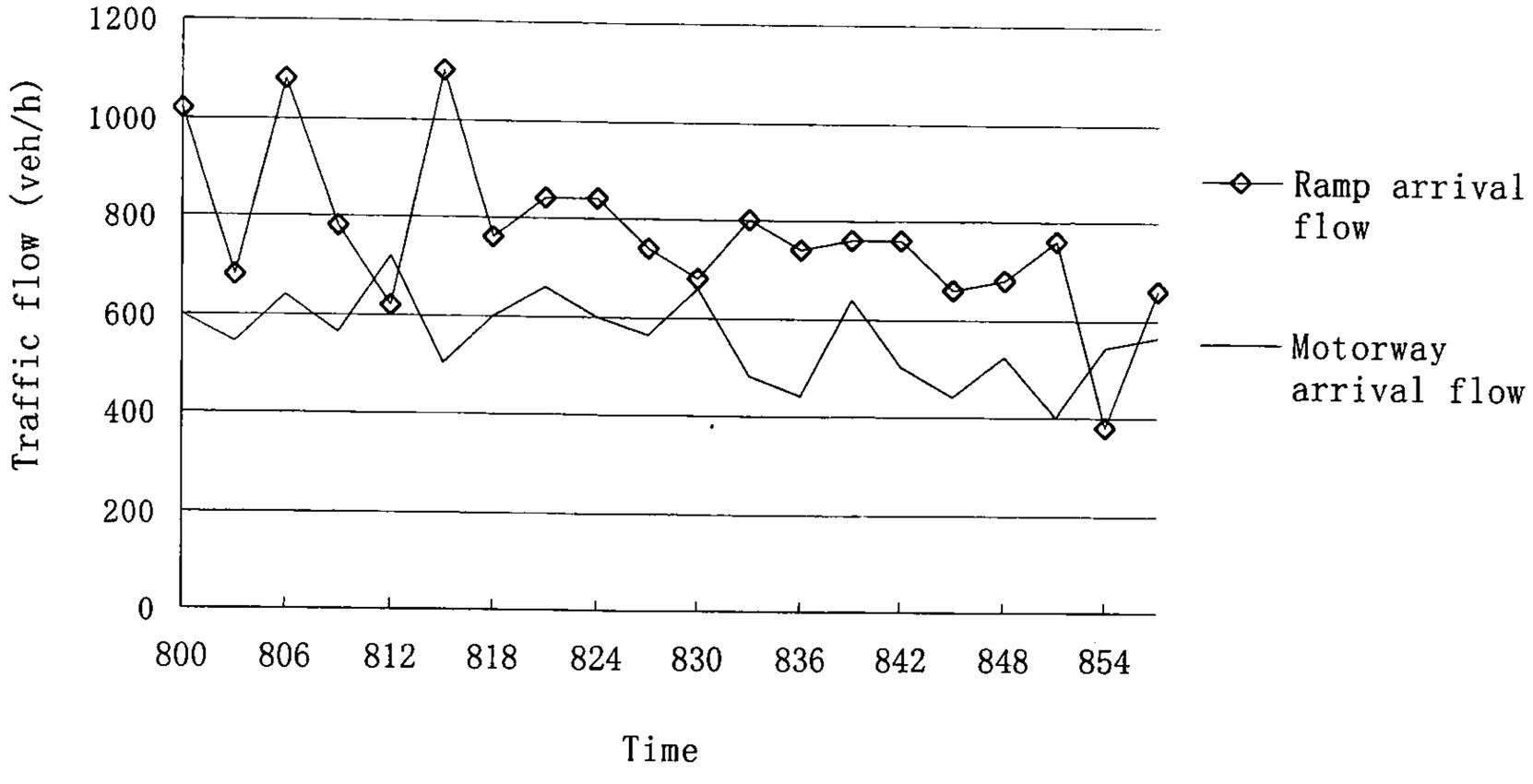


Figure 5-11 The arrival flow rate of ramp traffic and nearside motorway traffic of Juction 27, motorway M8, Glasgow with data collected on 25th July 2002.

From site configuration as provided earlier in the chapter, the acceleration lane length is 147m. In order to test the effectiveness of such length, different values of acceleration lane are tested and analysed from the aspect of merging operations. Figure 5-12 shows that according to current layout (i.e. 147m), most vehicles can merge in successfully. By increasing length of the acceleration lane (170m), the merging failure improves slightly, at around 2%; however, reducing the length will results in the increasing of merging failures. For example, with 90m-long acceleration lane, the merging failure could reach the level of 10%. It also appears that the longer the acceleration lane, the greater the chance that a following gap will be chosen. From the aspect of smooth merging (taking the original gaps), it appears that with merging failure under 5%, the current design length (147m) is the optimal length (i.e. more vehicles tend to take the original gaps rather than the following gap under current design, compared to the results with the length of 170m).

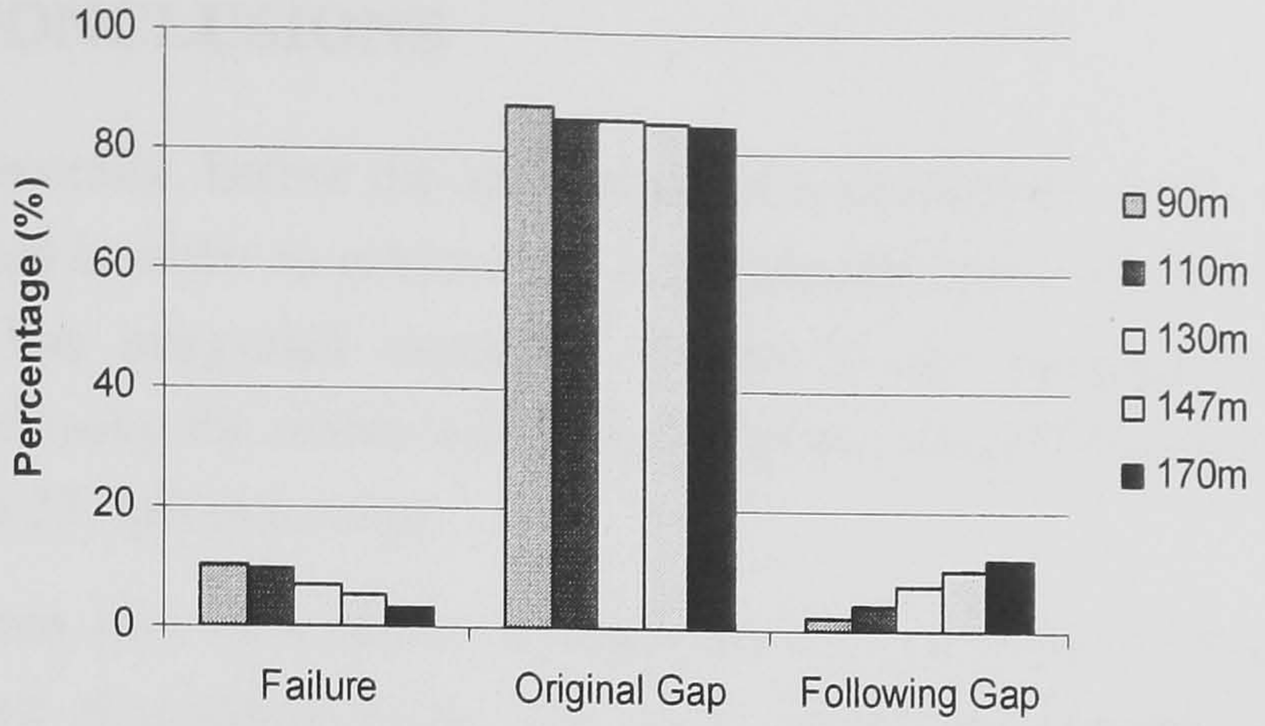


Figure 5-12 The tests of the selected site on hypothetical lengths of acceleration lane.

As discussed earlier in this section, the motorway arrival traffic flow is on the light side when it is compared to ramp traffic (Figure 5-11). From above analysis, the current geometry design has shown its effectiveness in motorway merging operations. The following tests are aimed to examine the site potential in dealing with more arrival traffic flow without interrupting merging operations. Figure 5-13 shows that with an excess of 200 veh/h to the current motorway traffic flow, merging failures increase by 3%. The merging failures are less sensitive when the motorway traffic flows are higher than 800 veh/h. This results imply that this site if the desirable merging failure controlled under 5%, is not capable of coping with motorway traffic higher than 800 veh/h. However, this result does not suggest a geometry design imperfection. As given earlier in the chapter in Figure 5-2, there is an off-ramp (monitoring site denoted as MS/M8/8508EO) on the motorway immediately to the upstream of the studied on-ramp section. Therefore, there current design might have already consider the fact of the motorway traffic is always on the light side.

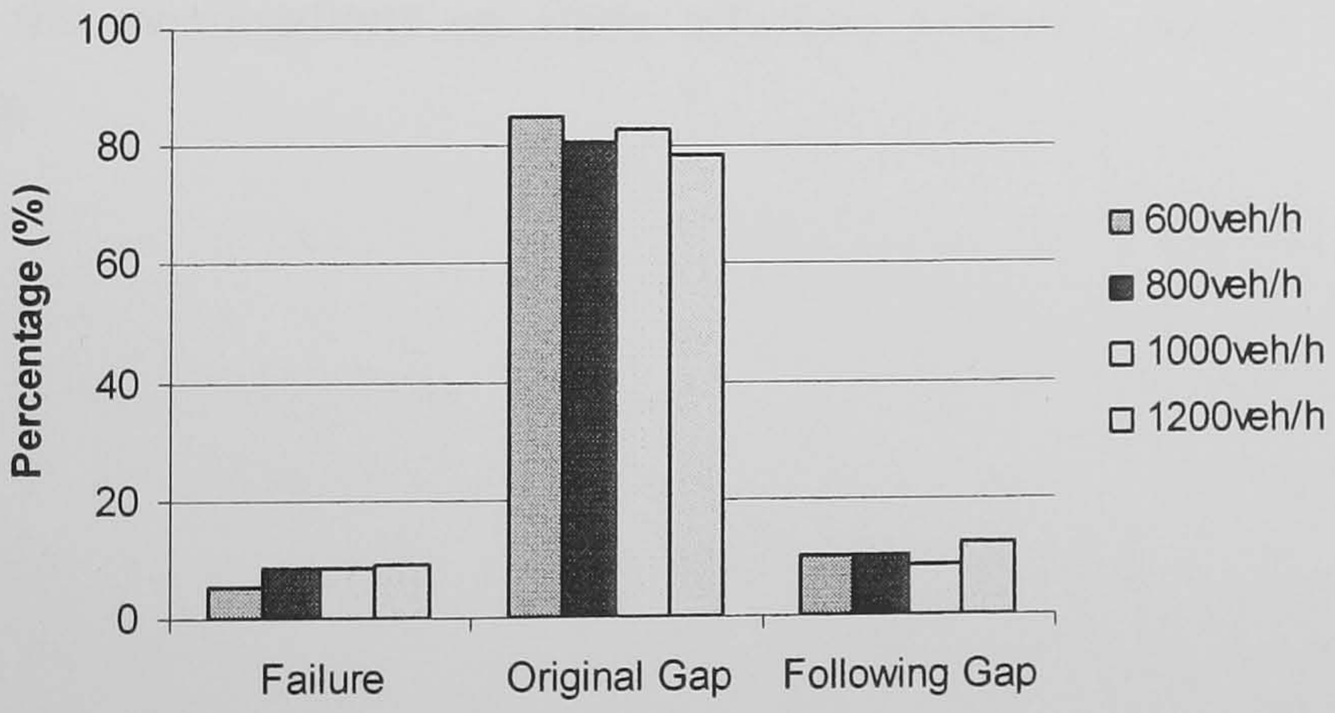


Figure 5-13 The tests of the selected site on hypothetical arrival flow of motorway.

5.4. CONCLUSIONS

Generally, before the application of a simulation model, calibration should be performed in order to achieve the best reproducibility of field conditions (Kosonen, 1999). The integrated simulation model in the merging area is calibrated and validated using the micro-simulation program MergeSim based on the case study of Junction 27, M8 motorway.

From this case study, it illustrates the calibration work when applying the integrated simulation model in a UK motorway merging section. Rather than resorting to elaborate video analysis to extract detailed data (e.g. accepted gaps, individual vehicle speeds), a general framework is proposed by using detector data. It shows the advantage of this integrated model that only a few parameters need to be calibrated through simulation runs (i.e. merging reaction time, gap acceptance factor and motorway drivers' accelerations and reaction times) which make it desirable for the implementation into other related studies.

The goodness-of-fit statistics show that the model can reasonably simulate flow, speed and reasonably replicate the decisions and movement of individual drivers in merging areas. With its reliability in representing the traffic operations, further investigations of the selected site are discussed. It shows that the current geometry design on the aspect of acceleration lane length brings the optimal smooth merging operations with merging failure under 5%. It might also imply that the current design might not be able to deal with more motorway arrival flow. However, this could be explained that the current design might have already considered that the motorway is always on the light side (due to an off-ramp located immediately to the upstream of the ramp-merging area). The further investigation of the study site could be used as an example when applying this model as a tool for further research, such as the examinations on some effective solutions related with the merging operation.

CHAPTER 6
CONCLUSIONS

Referring to the research objectives given in Chapter 1, this research has developed a feasible integrated microscopic simulation framework to model the interactions between traffic in motorway merging sections. This has been achieved by developing two sub-models working in tandem: a car-following and a merging model.

It is believed that this research represents the first comprehensive investigation and modelling of dynamic merging interactions at motorway on-ramps. Emphasis has been given to improving the modelling of merging behaviour and in particular to capture the cooperation between the merging and motorway traffic.

Considering the lack of a commonly agreed bench-marking procedure for the calibration and validation of micro-simulation models (Brockfeld et al., 2005), this study also established a general calibration and validation framework designed for real-world applications in highway networks using the most readily available traffic surveillance data, the loop detector data.

6.1. PROJECT SUMMARY

A framework for integrated driving behaviour is proposed in this research which captures both car-following and merging behaviours on UK motorways.

A newly developed car-following model has been shown to be able to capture some of the key motorway flow characteristics, namely traffic breakdown, hysteresis, shockwave propagation as well as close-following behaviour. The model proposes three different driving states: non-alert, alert and close-following. Under these different driving states, drivers apply different reaction times and accelerations. The new model has the advantage of having relatively few parameters to calibrate; and by using different reaction times and accelerations for different driving states, it can successfully replicate traffic hysteresis and speed drop at varied levels.

The car-following model is calibrated and validated with motorway data from M25 between Junction 11 and 12, with no on/off ramps in the vicinity of the site.

The calibration is carried out with respect to two traffic states, namely, alert and non-alert. Calibration involves modifying the parameters to ensure the simulation correctly reproduces the observations and is formulated as an optimisation problem. Model validation involves examining the similarity between the observed and simulated data with goodness-of-fit measures such as Theil's inequality coefficient, RMSPE and MPE. It is shown that the simulation can reasonably reproduce the observed measurements.

A merging model is also proposed which tries to capture both the acceleration and gap-acceptance behaviour of the merging traffic, and the cooperative behaviour of the motorway traffic. This new merging model is composed of several sub models such as a cooperation model to simulate the cooperative lane-changing and courtesy yielding behaviour and the acceleration, gap selection, gap acceptance and merge models for the merging traffic when interacting with motorway traffic.

It is found that the model is able to reproduce different merging behaviours, i.e. forced merging, merging under courtesy yielding and cooperative lane-changing from motorway traffic. It is also able to simulate different gap selections, namely taking the original gaps, previous gaps and following gaps.

Sensitivity tests suggest that the model responds well to changes in the model parameters including: gap acceptance sensitivity value, the rates of cooperative lane-changing and courtesy yielding, arrival traffic profiles, acceleration lane length as well as merging driver's reaction time. To examine its validity, the merging model is also calibrated independently with M27 motorway data available from literature (Zheng, 2002). The results indicate that the model can well reproduce the observed gap distributions.

The integrated simulation model MergeSim (which includes interdependent car-following and merging models) is calibrated and validated using video observations and aggregated loop detector data collected from Junction 27, eastbound of M8 Motorway, Glasgow. The goodness-of-fit statistics from the model validation. show that the model realistically reproduced the observed data.

6.2. CONTRIBUTIONS TO RESEARCH

It is often empirically observed that in motorway merging sections, drivers in the motorway nearside lane may choose to ignore the merging traffic by continuing with their desired movements. Alternatively they may slow down to create gaps for the merging vehicles, or they may even change lane on the motorway to allow the merging traffic to join. It is clear that both courtesy yielding and cooperative lane-changing are reactions to the ramp traffic. However, most existing studies simplify the complex dynamic interactive behaviours by ignoring these reactions, which will lead to an inaccurate description of the traffic phenomena and poor evaluation of the merging operations (Kita et al., 2002; Zheng, 2002). To address this problem, FHWA (US Federal Highway Administration) has recently commissioned research to investigate the effect of cooperative behaviour on the stability of motorway traffic flow (Halkias, J. 2005, private communication).

It is believed that the present study constitutes the first comprehensive study of the interactions in motorway merging sections by explicitly simulating the interactions between the merging traffic and the motorway traffic, and in particular has been able to capture the cooperative behaviours which have been ignored in most previous studies (Kita et al. 2002). A self-contained dynamic simulation program "MergeSim" has been developed for this study and has been shown to successfully replicate the traffic behaviour in the merging section.

This thesis also presents a general calibration and validation framework using loop detector data, the most readily available traffic surveillance data (Balakrishna et al., 2004). Using UK motorway M25 and M8 detector data, the study illustrated the calibration and validation processes and it is worth mentioning that the concept and the proposed methodology is applicable to other micro-simulation models.

The main contributions to research are:

1. Establishment of an in-depth understanding and modelling of the dynamic interactive behaviours between the two traffic streams at a motorway merging section: the merging traffic and the motorway nearside traffic. The study explicitly modelled the actions of all vehicles involved in the merging process (such as cooperative lane-changing and yielding from the nearside motorway traffic, and the

acceleration/deceleration and gap acceptance behaviours from the merging traffic on the acceleration lane.

2. Development of a general calibration and validation framework which has been shown to function successfully with loop detector data, the most readily available traffic data. This framework has the advantage that the concept and the proposed methodology are suitable for general application to other micro-simulation models.

3. The development of a new car-following model, as part of the study of motorway car following behaviour. This model is relatively easy to calibrate due to the few parameters involved, and has been shown to reasonably reproduce traffic breakdown, hysteresis, shockwave propagation and close-following phenomena.

4. The implications for the geometric configuration of the acceleration lane. The sensitivity tests on the different merging lengths indicated that increased length might reduce merging failures due to more merging traffic successfully taking the following gaps. For motorway merging design, the optimum length of acceleration lane could be different depending on which criterion is chosen, for example:

   - To maximise smooth merging (taking original gaps), the optimum length is approx. 100 m;
   - To minimise merging failure, a longer length such as 200 m is required.

Thus the developed simulation program might be used as a tool to assist in the evaluation and design of motorway merging sites.

In this thesis, an integrated simulation model of the dynamic interactive driving behaviour in merging sections is described. It is shown that the simulation model can reasonably replicate the decisions and movements of individual drivers in merging areas. Hence the integrated simulation model (MergeSim) can reliably be used as a tool for further studies and investigations into the effectiveness of techniques related to motorway merging operations.

6.3. DIRECTIONS FOR FURTHER RESEARCH

Some of the directions in which further research is needed are presented below:

The case study of Junction 27, M8 motorway, showed that the motorway traffic is relatively light due to an off-ramp slip road connected to the upstream of the merging area. Therefore, the integrated simulation model was only calibrated under non-alert situation (speed above 50 km/h, refer to chapter 3). It would be useful to test the model with another motorway merging site, which includes both the non-congested and congested traffic states, and the occurrence of traffic breakdowns.

The research simulated two traffic streams: merging traffic and nearside motorway traffic. As only direct interactive behaviour was being considered, offside motorway lanes were not modelled. Hence lane-changing logic is not included at the current stage. Future research should consider extending the single-lane motorway traffic to multi-lane with appropriate lane-changing logic, weaving driving behaviour could also be studied and included in the simulation.

According to DMRB (2001a), several layouts of ramp-merging design are applied in the UK (e.g. taper merge, parallel lane merge and ghost island merge, etc.). MergeSim is designed to simulate typical merging area, i.e. parallel lane merge. However, it could be used to test a taper merge layout by considering such layout as a short parallel lane merge. MergeSim can also be applied to analyse ghost island merge by representing such layout with two consecutive on-ramp merges. It would be useful to test the model with different merge layouts and examine the effectiveness of these designs in merging operations. It should be mentioned that on the acceleration road, at current stage only one lane is considered and simulated in this study. Further study should extend the single-lane acceleration road into two-lane (or multi-lane) with appropriate lane-changing logic included.

Traffic interactions such as merging and cooperative behaviour have a very great impact on motorway traffic stability, and may lead to flow breakdown especially when the traffic is near capacity (Zheng, 2002). Further research could focus on examination of the effects of cooperative behaviour on congested motorway traffic.

In this study, merging failure is considered as an indicator of merging operation performance. At the current stage, when merging failure occurs, the occurrence is registered and the vehicle is removed from simulation. This merging

failure although is quite rare for motorway ramp sections, could still happen on some very busy motorway sections. Further research could examine this motorway merging failure behaviour and include a related model (such as developing stop merging model) into this integrated model.

This study evaluated existing published parameter values and suggested a range of feasible values. However, the study did not set out specifically to directly estimate the proposed model parameters against field data. Considerable advances have been made recently both in the development of sophisticated estimation models and in testing alternative theories against field data (e.g. Brackstone et al., 2002; Brilon et al., 1999; Hagring, 1998). The next step in this research area could be to apply some of the advanced estimation methods in the calibration of the model using data collected at inter-urban and urban motorway merges.

REFERENCES

Addison, P. S. and Low, D. J. (1998) A Novel nonlinear car-following model, Chaos, December 1998.

Ahmed, K. (1999) *Modelling drivers' acceleration and lane changing behaviour*, PhD thesis, Massachusetts Institute of Technology, USA.

Aron, M. (1988) Car-following in an urban network: simulation and experiments. In *Proceedings of Seminar D*, 16th PTRC Meeting, Bath, UK, pp27-39.

Axhausen K.W. and Pendyala, R. (1997) Microsimulation, *8th IATBR Conference*, Austin, USA.

Bachem, A., Gawron, C., Moll, C., Ricket, M. and Wagner, p. (1996) Microscopic traffic simulations of road network using high-performance computer, *Proceedings of HPCN'98*, Brussels, Belgium, April 1996.

Balakrishna, R., Ben-Akiva, M. E., Koutsopoulos, H. N. and Toledo, T. (2004) Traffic simulation model calibration framework using aggregate data, *5th Triennial Symposium on Transportation Analysis*, Le Gosier, Guadeloupe..

Barcelo, J. (1996) The parallelisation of AIMSUN2 microscopic traffic simulator for ITS applications, *3rd World Conference on Intelligent Transport Systems*, Orlando, USA.

Barcelo, J. and Casas, J. (2004) Methodological notes on the calibration and validation of microscopic traffic simulation models, *Proceedings of the 83rd TRB annual meeting,* Washington, D.C.

Bando, M., Hasebe, K., Nakayama, A., Shibata, A. and Sugiyama, Y. (1995) Dynamical model of congestion and numerical simulation, Physical Review **E**, Vol. 51, pp. 1035-1042.

Benekohal, R.F. and Treiterer, J. (1988) CARSIM: Car-following model for simulation of traffic in normal and stop-and-go conditions, Transportation Research Record **1194,** pp99-111.

Berg, P., Mason, A. and Woods, A. (2000) Continuum approach to car-following models, Physical Review **E**, Vol. 61(2), pp. 1056-1066.

Bham, G. H. and Benekohal, R. F. (2004) A high fidelity traffic simulation model based on cellular automata and car-following concepts, Transportation Research **C**, Vol. 12, pp. 1-32.

Bloomberg, L. and Dale, J. (2000) Comparison of VISSIM and CORSIM traffic simulation models on a congested network, Transportation Research Record **1727**, pp52-60.

Brackstone, M. and McDonald, M. (1995) A Microscopic modelling of traffic flow: weakness and potential developments, *Proceedings of 'Workshop on Traffic and Granular Flow'*, HLRZ, Julich, Germany, October 1995.

Brackstone, M. and McDonald, M. (1999) Car-following: a historical review, Transportation Research **F** (2), pp181-196.

Brackstone, M., Sultan, B. and McDonald, M. (2002) Motorway driver behaviour: studies on car following, Transport Research **F**(5), pp31-46.

Brilon, W., Koenig, R., Troutback, R.J. (1999) Useful estimation procedures for critical gaps, Transportation Research A **(33)**, pp161-186.

Brockfeld, E., Kühne, R. and Wagner, P. (2005) Calibration and Validation of Microscopic Traffic Flow Models, *84th TRB annual meeting*, Washington, D.C.

Bunker, J. and Troutbeck, R. (2003) Prediction of minor stream delays at a limited priority freeway merge, Transportation Research **B (37)**, pp719-735.

Ceder, A. and May, Jr., A. D. (1976) Further evaluation of single and two regime traffic flow models. Transport Research Record **567**, pp1-30.

Chakroborty, P. and Kikuchi, S. (1999) Evaluation of the General Motors based car-following models and a proposed fuzzy inference model, Transportation Research **C**, Vol. 7, pp.209-235.

Chandler, R.E., Herman, R., and Montroll, E.W. (1958) Traffic dynamics: studies in car following, Operations Research **6**, pp165-184.

Chu, L., Liu, H. X., Oh, J. S. and Recker, W. (2004) A calibration procedure for microscopic traffic simulation. *Proceedings of the 83rd TRB annual meeting, Washington*, D.C.

Daganzo, C.F. (1997) *Fundamentals of Transportation and Traffic Operations*, Oxford; New York: Pergamon.

Daganzo, C.F., Cassidy, M.J. and Bertini, R.L. (1999) Some traffic features at freeway bottlenecks. Transportation Research **B**, Vol. 33, pp. 25-42.

Daganzo, C.F. (2002) A behavioral theory of multi-lane traffic flow. Part I: Long homogeneous freeway sections, Transportation Research **B,** Vol. 36(2), pp131-158.

Darzentas, J. (1981) Gap acceptance: myth and reality. In *Proceedings of the 8th International Symposium on Transportation and Traffic Theory*, Toronto, Canada, pp175-192.

DfT (2003) *Transport statistics bulletin- road traffic statistics: 2002*, National Statistics, London

Diakaki, C., Papageorgiou, M. and McLean, T. (2000) Integrated traffic-responsive urban corridor control strategy in Glasgow, Scotland Application and Evaluation, Transportation Research Record **1727**, pp101-111.

Dijker, T., Bovy, P.H. and Vermijs, R. (1998) Car-following under congested conditions empirical findings, Transportation Research Record **1644**, pp20-28.

DMRB (2001a) *Design manual for road and bridges*. Vol 6(2), Part 6, Highways Agency, London, pp 7/15.

DMRB (2001b) *Design manual for road and bridges*. Vol 12(2), Part 1, Highways Agency, London, pp 4/26.

Dowling, R. Skabardonis, A. Halkias, J. McHale, G. and Zammit, G. (2004) Guidelines for calibration of microsimulation models: framework and application, *Proceedings of the 83rd TRB annual meeting*, Washington, D.C.

Drew, D. R. (1967). Gap acceptance characteristics for ramp-freeway surveillance and control. Highway Research Record **157**, 108-134.

Duncan, N. C. (1976) *Rural speed-flow relations*, Transportation and Road Research Laboratory, LR 705.

Elefteriadou, L., Roess, R. and Mcshane, W. (1995) Probabilistic nature of breakdown at freeway merge junctions, Transportation Research Record **1484**, pp 80-89.

El-Hanna, F.I.H. (1974) *An evaluation of highway intersection gap acceptance metering system*, Ph.D. Thesis, University of Bradford, UK.

Evans, J., Elefteriadou, L. and Gautam, N. (2001) Probability of breakdown at freeway merges using Markov chains, Transportation Research **B,** Vol. 35. pp237-254.

Fambro, D.B., Koppa, R.J., Picha, D.L. and Fitzpatrick K. (1998) Driver perception-brake response in stopping sight distance situations, Transportation Research Record **1631**, pp. 13-19.

Forbes, T.W. and Simpson, M.E. (1968) Driver and vehicle response in freeway deceleration waves, Transportation Science, Vol. 2, No. 1, pp. 77-104.

Gazis, D. C., Herman, R. and Potts, R.B. (1959) Car following theory of steady state traffic flow, Operations Research 7, pp499-505.

Gazis, D. C., Herman, R. and Rothery, R.W. (1961) Nonlinear follow the leader models of traffic flow, Operations Research 9, pp545-567.

Gibson S. and McCartney, M. (2005) Ring car-following models, 37th *UTSG Annual Conference*, Bristol.

Gipps, P.G. (1981) A behavioral car-following model for computer simulation, Transportation Research **B**, Vol. 15, pp105-111.

Gipps, P.G. (1986) A model for the structure of lane-changing decisions, Transportation Research **B,** Vol. 20**,** pp403-414.

Gomes, G., May, A. and Horowitz, R. (2004) A microsimulation model of a congested freeway using VISSIM, *83rd TRB Annual Conference,* Washington, D.C.

Goodman, P. (2001) *The prediction of road traffic noise in urban areas*, PhD Thesis, University of Leeds, UK.

Hagring, O. (1998) A further generalisation of Tanner's formula, Transportation Research B (**32)**, pp423-429.

Hansen, P., Martin,T. and Perrin, J. (2000) Scoot real-time adaptive control in a CORSIM simulation environment, Transportation Research Record **1727**, pp27-31.

Helly, W. (1959) Simulation of bottlenecks in single lane traffic flow, In *Proceedings of the Symposium of Theory of Traffic flow*, New York, USA, pp 207-238.

Herman, R. and Potts, R.B. (1959) Single lane traffic theory and experiment, In *Proceedings of the Symposium on Theory of Traffic Flow*, New York, USA, pp147-157.

Heyes, M. P. and Ashworth, R. (1972) Further research on car following models, Transportation Research **6**, pp287-291.

Hidas, P. (2002) Modelling lane-changing and merging in microscopic traffic simulation, Transportation Research **C 10**, pp351-371.

Hoefs, D.H. (1972) Entwicklung einer messmethode uber den bewegungsablauf des kolonnenverrkehrs, Universitat (TH) Karlsruhe, Germany.

Holland, E. N. (1998) A generalised stability criterion for motorway traffic, Transportation Research **B**, Vol. 32(2), pp.141-151.

Holmes, B. (1994) When shockwaves hit traffic, New Scientist, Vol. 142, pp36.

Hounsell, N.B., Barnard, S.R. and McDonald, M. (1992) *An investigation of flow breakdown and merge capacity on motorways*, *Contractor Report* 338.

Hourdakis, J., Michalopoulos, P.G., and Kottommannil, J. (2003) Practical procedure for calibrating microscopic traffic simulation models, Transportation Research Record **1852**, pp130-139.

Hunt, J.G. and Yousif, S. Y. (1990) Merging behaviour at roadworks, *PTRC 18th Summer Annual Meeting*, University of Sussex, UK.

ITE (1999) *Transportation and traffic engineering handbook*, ITE Publications, pp 63-65.

Jiang, R., Wu, Q. and Zhu, Z. (2001) Full velocity difference model for a car following theory, Physical Review **E**, Vol. 64, paper No. 017101.

Johansson, G. and Rumar, K. (1971) Drivers' brake reaction time, Human Factors, Vol. 13, pp23-27.

Kerner, B. S. and Rehborn, H. (1996) Experimental properties and characteristics of traffic jams. Physical Review **E**., Vol. 53, R1297-R1300.

Kita, H. (1993) Effects of merging lane length on the merging behaviour at expressway on-ramps, *Proceedings of the 12th International Symposium on the Theory of Traffic Flow and Transportation*, Berkeley, California, USA, pp37-51.

Kita, H., Tanimoto, K. and Fukuyama, K. (2002) A game theoretic analysis of merging-giveway interaction: a joint estimation model, *Proceedings of the 15th International Symposium on Transportation and Traffic Theory,* Adelaide, Australia, 2002, pp503-518.

Knospe, w., Santen, L., Schadschneider, A. and Schrokenberg, M. (2002) Single-vehicle data of highway traffic: microscopic description of traffic phases, Physical Review **E(65)**, No. 056133.

Kosonen, I. (1999) *HUTSIM- urban traffic simulation and control model: principles and applications*, Helsinki University of Technology Transportation Engineering Publication 100.

Law, A. M. and Kelton, W. D. (2000) *Simulation modeling and analysis*, McGraw-Hill, USA.

Leutzbach, W. and Wiedemann, R. (1986) Development and applications of traffic simulation models at the Karlsruhe Institut Fur Verkehrwesen, Traffic Engineering and Control **27**, pp270-278.

Lighthill, M. J. and Whitham, G. B. (1955) On kinematic wave's Π A theory of traffic flow on long crowded roads, *Proceedings of the Royal Society, London*, Series A 229, pp 317-345.

Liu, R. and Wang, J (2005) A general framework for the calibration and validation of car-following models along an uninterrupted open highway, 4th IMA International Conference on Mathematics in Transport, London, U.K., September 2005.

Mason, A.D. and Woods, A.W. (1997) Car-following model of multispecies systems of road traffic, Physical Review **E**, Vol. 55(3), pp. 2203-2214.

May, A.D. (1990) *Traffic flow fundamentals*, Prentice Hall, New Jersey, USA.

May, Jr., A.D. and Keller, H.E.M. (1967) Non integer car following models, Highway Research Record **199**, pp13-32.

Michaels, R. M. and Gordon, D.A. (1963) Perceptual and field factors causing lateral displacement, Highway Research Record **84**.

Michaels, R. M. and Fazio, J. (1989) Driver behaviour model of merging, Transportation Research Record **1213**, pp4-10.

Middelham, F. (2001) Predictability: Some thoughts on Modelling, Future Generation Computer Systems, Vol. 17, pp627-636

Miyahara, T. (1994) The modelling of motorway traffic flow, 26th *UTSG Annual Conference*, Leeds.

Neubert, L., Santon, L., Schadschneider, A. and Schreckenber, M. (1999) Single-vehicle data of highway traffic: a statistical analysis, Physical Review **E(60)**, pp6480-6491.

Ozaki, H. (1993) Reaction and anticipation in the car following, *In Proceedings of the 13th International Symposium on the Theory of Road Traffic Flow*, Paris, pp349-366.

Papageorgiou, M. (1998) Some remarks on macroscopic traffic flow modelling, Transportation Research **A**, Vol. (32), No.5, pp323-329.

Pipes, L.A. (1953) An operational analysis of traffic dynamics, Journal of Applied Physics **24**, pp274-281.

Postans, R.L. and Wilson, W.T. (1983) Close-following on the motorway, Ergonomics, Vol. 26, No. 4, pp317-27.

Press, W.H., Flammery, B.P., Teukolsey, S.A., and Vetterling, W.T. (1992) *Numerical Recipes in C*, Cambridge University, UK.

PTV (2003) *VISSIM user manual*, Larlsruhe, Germany.

Rakha, H. and Crowther, B. (2003) Comparison and calibration of FRESIM and INTEGRATION steady-state car-following behaviour, Transportation Research **A**, Vol. 37, pp. 1-27.

Sarvi, M., Ceder, A. and Kuwahara, M. (2002) Modelling of freeway ramp merging process observed during traffic congestion. In *Proceedings of the 15th International Symposium on Transportation and Traffic Theory*, Adelaide, Australia, pp483-502.

Schreckenberg, M., Neubert, L. and Wahle, J. (2001) Simulation of traffic in large road networks, Future Generation Computer Systems, Vol. 17, pp649-657.

SISTM (1993) *A Motorway Simulation Model*, Leaflet LF2061, Transport Research Laboratory.

Sultan, B. (1998) Modelling driver behaviour on motorways, *30th UTSG Annual Conference,* Dublin.

Systems, M. (2002) Analysis of weather impacts on traffic flow in Metropolitan Washington DC, *2003 American Meteorology Society Annual Meeting*, California.

Theil, H. (1961) *Economic forecasts and policy*, North-holland Publishing Company, Amsterdam, Netherlands.

Treiterer, J. and Myers, J. A. (1974)The hysteresis phenomenon in traffic flow, *Proceedings of the 6th International Symposium on Transportation and Traffic Theory,* Sydney, Australia, pp. 13-38.

TRB (1996) *Special Report **165:** Traffic Flow Theory*, Transportation Research Board, National Research Council, Washington, D.C., USA.

TRB (1997) *Comparison of the 1994 highway capacity manual's ramp analysis procedures and the FRESIM model*, NCHRP Report 385.

Troutbeck, R. (2002). The performance of uncontrolled merges using a limited priority process, *Proceedings of the 15th International Symposium on Transportation and Traffic Theory* (ed. M.A.P. Taylor), Adelaide, pp463-482.

Toledo, T. (2003) *Integrated driving behaviour modelling*, PhD thesis, Massachusetts Institute of Technology, USA.

Wang, J. (2003) Car-Following models for motorway traffic, *35th UTSG Annual Conference*, Loughborough, U.K., January 2003.

Wang, J., Liu, R. & Montgomery, F. O. (2005a) A car-following model for motorway traffic, Transportation Research Record, in press, proof corrected.

Wang, J., Liu, R. & Montgomery, F. O. (2005b) A simulation model for motorway merging behaviour, Presented and Published in the *Proceedings of the 16th International Symposium on Transportation and Traffic Theory*, Elsevier, U.S.A. , pp281-301.

Webster N. and Elefteriadou, L. (1999) A simulation study of truck passenger car equivalents (PCE) on basic freeway sections, Transportation Research **B**, Vol. 33, pp323-336.

Wilson, R. E. (2001) An analysis of Gipps's car-following model of highway traffic, IMA Journal of Applied Mathematics, Vol. 66, pp509-537.

Wu, J., Brackstone, M. and McDonald, M. (2003) The validation of a microscopic simulation model: a methodological case study, Transportation Research **C**, Vol. 11, pp. 463-479.

Yang, Q. and Koutsopoulos, H. N. (1996) A microscopic traffic simulator for evaluation of dynamic traffic management systems, Transportation Research **C 4(3),** pp113-129.

Yousif, S. and Hunt, J. (1995) Modelling lane utilisation on British dual-carriageway roads: effects on lane-changing, Traffic Engineering and Control **36 (2)**, pp 680-687.

Zhang, Y. L., Owen, L. E., and Clark, J. E. (1998) Multi-regime Approach for Microscopic Traffic Simulation, Transportation Research Record **1644**, pp103-115.

Zhang, H.M. (1999) A mathematical theory of traffic hysteresis, Transportation Research **B**, Vol. 33 (1), pp1-23.

Zhang, H.M. and Kim, T. (2001) A Car-following theory for multiphase vehicular traffic flow, *80th TRB Annual Conference*, Washington D.C., U.S.A.

Zheng, P. (2002) *A Microscopic simulation model of merging operation at motorway on-Ramps*, PhD thesis, University of Southampton, UK.

Appendix A Source codes of Main Sub-routines

The source codes of some important sub-routines are given below. These sub-routines are used to apply the car-following model and merging model as discussed in the thesis. The purpose for doing so is to facilitate implementation of these developed behavioral models of this study into other simulation research.

A.1 MainModel() Source Codes

```
void CMergeModel::MainModel()        //Mainmodel is used for Initialisation of vehicles' data and call subroutines: "motorwaymodel" and "accelerationlanemodel"
{   ……
//Define Parameters
    //Set simulation clock to 0
    //Initialise the New Motorway/Ramp Vehicle and Add Vehicle to Vehicle List
    ……
    while(cur_time < (g_simu_period+g_warmup_period)/computet)  // In simulation period
    {
            //vehicle generation on the motorway and ramp; load arrival flow and speed information
    ……
            g_lane[i-10].Lane_Flow = motor_arrival_flow[tmp_count];
            g_lane[i-10].Lane_Speed_Mean = motor_arrival_speed[tmp_count]/3.6;
            g_lane[i-10].Lane_Flow = ramp_arrival_flow[tmp_count];
            g_lane[i-10].Lane_Speed_Mean = ramp_arrival_speed[tmp_count]/3.6;
……
    GetAddVehicleTime(…); // return vehicle generation time as a function of arrival flow
    ……
                    arrival_count ++;
            }

            cur_time ++; //Time increment
    ……
    MotorwayModel(); // Call sub-routine to calculate motorway vehicle data
            AccelerationLaneModel(); // Call sub-routine to calculate ramp vehicle data
    }
    ……
}
```

A.2 MotorwayModel() Source Codes

```
void CMergeModel:: MotorwayModel()
{
    VehicleStru Vehicle1,Vehicle2;
    int i;
    int CurStepAddHGVCount;
    int CurAddHGVBase;          //
    bool cur_vehicle_is_HGV = false;
    float tempt;
    float an,bn,sn,vn,b_severebra;
    float headwaygap1,headwaygap2,tempacc1,tempacc2;
    float tempDV,tempDX,tempABX,tempSDX,tempAX,tempBX,tempEX;
    int list_vehicle_count;
    int tmp_count;
    bool InMergeArea;

    VehicleStru FrontVehicle;     // Leading vehicle of current vehicle
    VehicleStru Front2Vehicle;    // Leading vehicle of current vehicle's lead
    VehicleStru PreVehicle1;
    VehicleStru PreGipVehicle;
   VehicleStru FrontGipVehicle;
   VehicleStru Front2GipVehicle;
    VehicleStru LastVehicle;
    VehicleStru AddVehicle,AddVehicle2;

    POSITION PosIndex, PrePosIndex, tmpPrePosIndex, PosLastIndex;
    int k;

    for(i=10;i<g_lane_count+10;i++)
    {
            tmp_count = 0;
            list_vehicle_count = 0;
            k = 0;

            if(g_lane[i-10].Lane_Flow_Out > 0)
            {
                    int tt2;
                    tt2 = (int)(g_lane[i-10].AddVehicleTime_Out/computet+.5);
                    if(cur_time == tt2)
                    {
                            //HGV
                            if(g_lane[i-10].AddedVehicleCount_Out+1 == g_lane[i-
10].CurStepAddHGVCount_Out)
                            {
                                    g_lane[i-10].AddedHGVCount_Out ++;
        g_lane[i-10].CurStepAddHGVCount_Out= GetAddHGVCount(g_lane[i-10].CurAddHGVBase_Out,
g_lane[i-10].AddHGVStep_Out);
        AddVehicle2.Type.VehicleType_Length_Mean= g_vehicle_type[1].VehicleType_Length_Mean;
                AddVehicle2.Type.VehicleType_Length_Devi =
g_vehicle_type[1].VehicleType_Length_Devi;
                            }
                            //Passenger car
                            else
                            {
        AddVehicle2.Type.VehicleType_Length_Mean=g_vehicle_type[0].VehicleType_Length_Mean;
        AddVehicle2.Type.VehicleType_Length_Devi = g_vehicle_type[0].VehicleType_Length_Devi;
                            }
                            g_lane[i-10].AddedVehicleCount_Out ++;
                            //initial the added vehicle
                            AddVehicle2.ID = -999;

                            AddVehicle2.Position = g_environment.Motorway_End.x;
                            AddVehicle2.Speed = ran0.getNorDis(g_lane[i-
10].Lane_Speed_Mean_Out,g_lane[i-10].Lane_Speed_Devi_Out);
                            AddVehicle2.Length =
```

```
ran0.getNorDis(AddVehicle2.Type.VehicleType_Length_Mean,AddVehicle2.Type.VehicleType_Length_Devi);
                                        AddVehicle2.Acceleration = 0;
                                        AddVehicle2.pregip.Speed = AddVehicle2.Speed;
                                        AddVehicle2.pregip.Position = AddVehicle2.Position;
                                        AddVehicle2.pregip.Acceleration = AddVehicle2.Acceleration;
                                        AddVehicle2.preveh.Speed = AddVehicle2.Speed;
                                        AddVehicle2.preveh.Position = AddVehicle2.Position;
                                        AddVehicle2.preveh.Acceleration = AddVehicle2.Acceleration;
                                        AddVehicle2.IsPFVehicle = 0;

                                        vehicle_out[i].VehicleList.RemoveAll();
                                        vehicle_out[i].Add(AddVehicle2);

                                        //get the add vehicle time
                                        g_lane[i-10].AddVehicleTime_Out = GetAddVehicleTime(g_lane[i-
10].AddVehicleTime_Out,g_lane[i-10].Lane_Flow_Out);
                                }

                                PosIndex = vehicle_out[i].VehicleList.GetHeadPosition();
                                if(cur_time == tt2)
                                {
                                        Vehicle2 = vehicle_out[i].VehicleList.GetNext(PosIndex);
                                }
                                else
                                {
                                        while(PosIndex!=NULL)
                                        {
                                                Vehicle2 = vehicle_out[i].VehicleList.GetNext(PosIndex);
                                                Vehicle2.Position = Vehicle2.Position + computet *
Vehicle2.Speed;

        if(Vehicle2.Position>g_environment.Motorway_End.x+g_environment.Motorway_Buffer_Len)
                                                {
                                                        vehicle_out[i].VehicleList.RemoveAll();
                                                        Vehicle2.ID = -998;
                                                        break;
                                                }
                                        }
                                }
                        }
                        else
                        {
                                if(g_fixed == 0)
        {
                                        g_lane[i-10].AddVehicleTime_Out = -999;
                                }
                                else
                                {
                                        g_lane[i-10].AddVehicleTime_Out = g_lane[i-
10].AddVehicleTime_Out + 180;
                                }
                                Vehicle2.ID = -998;
        }


PosIndex=vehicle[i].VehicleList.GetHeadPosition();

//Updates all vehicles' position
while(PosIndex!=NULL)
{
        tmpPrePosIndex = PosIndex;
        Vehicle1 = vehicle[i].VehicleList.GetNext(PosIndex);
        PreVehicle1 = vehicle[i].CopyVehicle(Vehicle1);

        if(Vehicle1.CarCloseFollowing == 0)
        {
                Vehicle1.Position = Vehicle1.preveh.Position + computet *
```

```
(Vehicle1.preveh.Speed + Vehicle1.preveh.Speed + Vehicle1.dSpeed) / 2;
                                }
                                else if(Vehicle1.CarCloseFollowing == 1) //A1
                                {
                                        Vehicle1.Position = Vehicle1.preveh.Position + computet *
Vehicle1.preveh.Speed + .5*g_a1_closefollowing*computet*computet;
                                }
                                else
        //D1
                                {
                                        Vehicle1.Position = Vehicle1.preveh.Position + computet *
Vehicle1.preveh.Speed + .5*g_d1_closefollowing*computet*computet;
                                        if(Vehicle1.Position <= Vehicle1.preveh.Position)
                                                Vehicle1.Position = Vehicle1.preveh.Position + 1;
                                }

                                if(Vehicle2.ID == -999)
                                {
                                        Vehicle1.Gap = Vehicle2.Position - Vehicle2.Length -
Vehicle1.Position;
                                        Vehicle1.Headway = Vehicle2.Position - Vehicle1.Position;
                                }
                                else
                                {

        Vehicle1.Gap=g_environment.Motorway_End.x+g_environment.Motorway_Buffer_Len -
Vehicle1.Position;
        Vehicle1.Headway=g_environment.Motorway_End.x+g_environment.Motorway_Buffer_Len-
Vehicle1.Position;
                                }

                                vehicle[i].VehicleList.SetAt(tmpPrePosIndex,Vehicle1);
        FrontVehicle = vehicle[i].CopyVehicle(Vehicle1);

        k++;
}

k=0;
PosIndex=vehicle[i].VehicleList.GetHeadPosition();
PrePosIndex=vehicle[i].PreVehicleList.GetHeadPosition();
// Update all vehicles' speed
while(PosIndex!=NULL)
{
        tmpPrePosIndex = PosIndex;
        Vehicle1 = vehicle[i].VehicleList.GetNext(PosIndex);
        PreVehicle1 = vehicle[i].CopyVehicle(Vehicle1);

        Vehicle1.CarCurT = Vehicle1.CarCurT + computet * 10;

                if(Vehicle1.CarCurT < Vehicle1.CarT)
                {
                        if(Vehicle1.CarCloseFollowing == 0)
                        {
                                Vehicle1.Speed = Vehicle1.preveh.Speed +
Vehicle1.dSpeed;
                                Vehicle1.Acceleration =
Vehicle1.pregip.Acceleration;
                        }
                        else if(Vehicle1.CarCloseFollowing == 1) //A1
                        {
                                Vehicle1.Speed = Vehicle1.preveh.Speed +
g_a1_closefollowing * computet;
                                Vehicle1.Acceleration = g_a1_closefollowing;
                        }
                        else
```

```
                    //D1
                                        {
                                                Vehicle1.Speed    =    Vehicle1.preveh.Speed    +
g_d1_closefollowing * computet;
                                                Vehicle1.Acceleration = g_d1_closefollowing;
                                        }
                                }
                                //Start for next reaction time
                                else
                                {
                                        tempt = Vehicle1.CarT/10.0;

                                        if(Vehicle1.Congested)
                                        an      =       ran0.getNorDis(g_accmean_alert,
g_accdevi_alert);
                                        else
                                        an      =       ran0.getNorDis(g_accmean_surprised,
g_accdevi_surprised);

                                        float ttt_ppp;

                                        if(Vehicle1.IsPFVehicle == 1)
                                        {
                                                bn = g_merge_bn;
                                                if(g_merge_b1 == 1)
                                                {
                                                        b_severebra = g_merge_b2;
                                                }
                                                else
                                                {
                                                        b_severebra   =   __min(-3.0,   ((-2.0)*an-
3.0)/2);
                                                }
                                                Vehicle1.IsPFVehicle = 0;
                                                ttt_ppp = g_reactime_merge;
                                                an      =       ran0.getNorDis(g_accmean_alert,
g_accdevi_alert);
                                        }
                                        else
                                        {
                                                bn = (-2.0)*an;
                                                b_severebra = __min(-3.0, (bn-3.0)/2);
                                                ttt_ppp = tempt;
                                        }
                                        sn = Vehicle1.pregip.FrontLength;

vn=ran0.getNorDis(Vehicle1.Type.VehicleStatic_SpeedDesiredMean,Vehicle1.Type.VehicleStatic_SpeedDesir
edDevi);
                                        if(Vehicle1.yielding == 2)
                                        {
                                                before_yield_speed = Vehicle1.Speed;
                                                before_yield_acc = Vehicle1.Acceleration;
                                        }

                                        Vehicle1.Acceleration = an;

                                        float tmp_speed1,tmp_speed2;

                                        if(Vehicle1.CarCloseFollowing == 0)
                                        {
                                                if(k != 0)
                                                {
                                                        if(Vehicle1.yielding == 2)
                                                        {

fprintf(fp_yieldingall,"%6s\t%6.2f\t%5d\t%8.2f\t%8.2f\t%8.2f\t%17.2f\t%15.2f\t%8.2f\t%8.2f\t%8.2f\
```

```
t%12.2f\t%8.2f\t%16.2f\t%15.2f\t%21.2f\t%20.2f\t%9.2f\n","0",cur_time*computet,Vehicle1.ID,Vehicle1.Positi
on,Vehicle1.Speed,Vehicle1.Acceleration,before_yield_speed,before_yield_acc,an,bn,vn,b_severebra,ttt_ppp,V
ehicle1.pregip.Speed,Vehicle1.pregip.Position,Vehicle1.pregip.FrontSpeed,Vehicle1.pregip.FrontPosition,Vehi
cle1.pregip.FrontLength);
                                                                    Vehicle1.yielding = 1;
                                                                }
                                                            }
                                                            else
                                                            {
                                                                if(Vehicle1.pregip.FrontPosition >=
g_environment.Motorway_End.x && Vehicle1.pregip.FrontPosition <= g_environment.Motorway_End.x +
g_environment.Motorway_Buffer_Len)
                                                                {
        //                                                          Vehicle1.Speed =
__min(Vehicle1.pregip.Speed + 2.5 * an * ttt_ppp * (1-Vehicle1.pregip.Speed/vn) *
sqrt(0.025+Vehicle1.pregip.Speed/vn),
        //                                                                  __max(bn *
ttt_ppp + sqrt( bn * bn * ttt_ppp * ttt_ppp - bn * (2 * (Vehicle1.pregip.FrontPosition - sn -
Vehicle1.pregip.Position) - Vehicle1.pregip.Speed * ttt_ppp -
        //
            Vehicle1.pregip.FrontSpeed * Vehicle1.pregip.FrontSpeed /
b_severebra)),Vehicle1.pregip.Speed+b_severebra*ttt_ppp));
                                                                    if(Vehicle1.yielding == 2)
                                                                    {

            fprintf(fp_yieldingall,"%6s\t%6.2f\t%5d\t%8.2f\t%8.2f\t%8.2f\t%17.2f\t%15.2f\t%8.2f\t%8.2f\t%8.2f\
t%12.2f\t%8.2f\t%16.2f\t%15.2f\t%21.2f\t%20.2f\t%9.2f\n","NH",cur_time *
computet,Vehicle1.ID,Vehicle1.Position,Vehicle1.Speed,Vehicle1.Acceleration,before_yield_speed,before_yiel
d_acc,an,bn,vn,b_severebra,ttt_ppp,Vehicle1.pregip.Speed,Vehicle1.pregip.Position,Vehicle1.pregip.FrontSpee
d,Vehicle1.pregip.FrontPosition,Vehicle1.pregip.FrontLength);
                                                                        Vehicle1.yielding = 1;
                                                                    }
                                                                }
                                                                else
                                                                {
                                                                    if(Vehicle1.yielding == 2)
                                                                    {

            fprintf(fp_yieldingall,"%6s\t%6.2f\t%5d\t%8.2f\t%8.2f\t%8.2f\t%17.2f\t%15.2f\t%8.2f\t%8.2f\t%8.2f\
t%12.2f\t%8.2f\t%16.2f\t%15.2f\t%21.2f\t%20.2f\t%9.2f\n","NN",cur_time*computet,Vehicle1.ID,Vehicle1.Po
sition,Vehicle1.Speed,Vehicle1.Acceleration,before_yield_speed,before_yield_acc,an,bn,vn,b_severebra,ttt_ppp
,Vehicle1.pregip.Speed,Vehicle1.pregip.Position,Vehicle1.pregip.FrontSpeed,Vehicle1.pregip.FrontPosition,Ve
hicle1.pregip.FrontLength);
                                                                        Vehicle1.yielding = 1;
                                                                    }
                                                                }
                                                            }

                                                            Vehicle1.Speed = Vehicle1.NextSpeed;
                                                        }
                                                        else if(Vehicle1.CarCloseFollowing == 1) //A1
                                                        {
                                                            tmp_speed1 = Vehicle1.pregip.Speed + 2.5 * an *
ttt_ppp * (1-Vehicle1.pregip.Speed/vn) * sqrt(0.025+Vehicle1.pregip.Speed/vn);
                                                            Vehicle1.Speed = Vehicle1.pregip.Speed +
g_a1_closefollowing*tempt;
                                                            Vehicle1.Speed =
__min(tmp_speed1,Vehicle1.Speed);
                                                            Vehicle1.Acceleration = g_a1_closefollowing;
                                                            if(Vehicle1.yielding == 2)
                                                            {

            fprintf(fp_yieldingall,"%6s\t%6.2f\t%5d\t%8.2f\t%8.2f\t%8.2f\t%17.2f\t%15.2f\t%8.2f\t%8.2f\t%8.2f\
t%12.2f\t%8.2f\t%16.2f\t%15.2f\t%21.2f\t%20.2f\t%9.2f\n","A",cur_time *
computet,Vehicle1.ID,Vehicle1.Position,Vehicle1.Speed,Vehicle1.Acceleration,before_yield_speed,before_yiel
d_acc,an,bn,vn,b_severebra,ttt_ppp,Vehicle1.pregip.Speed,Vehicle1.pregip.Position,Vehicle1.pregip.FrontSpee
d,Vehicle1.pregip.FrontPosition,Vehicle1.pregip.FrontLength);
```

```
                                                    Vehicle1.yielding = 1;
                                                }
                                            }
                                            else
                                //D1
                                            {
                                                tmp_speed1 = Vehicle1.pregip.Speed + 2.5 * an *
ttt_ppp * (1-Vehicle1.pregip.Speed/vn) * sqrt(0.025+Vehicle1.pregip.Speed/vn);
                                                Vehicle1.Speed = Vehicle1.pregip.Speed +
g_d1_closefollowing*tempt;
                                                Vehicle1.Speed =
__min(tmp_speed1,Vehicle1.Speed);

                                                Vehicle1.Acceleration = g_d1_closefollowing;
                                                if(Vehicle1.yielding == 2)
                                                {

        fprintf(fp_yieldingall,"%6s\t%6.2f\t%5d\t%8.2f\t%8.2f\t%8.2f\t%17.2f\t%15.2f\t%8.2f\t%8.2f\t%8.2f\
t%12.2f\t%8.2f\t%16.2f\t%15.2f\t%21.2f\t%20.2f\t%9.2f\n","D",cur_time *
computet,Vehicle1.ID,Vehicle1.Position,Vehicle1.Speed,Vehicle1.Acceleration,before_yield_speed,before_yiel
d_acc,an,bn,vn,b_severebra,ttt_ppp,Vehicle1.pregip.Speed,Vehicle1.pregip.Position,Vehicle1.pregip.FrontSpee
d,Vehicle1.pregip.FrontPosition,Vehicle1.pregip.FrontLength);
                                                    Vehicle1.yielding = 1;
                                                }
                                            }

                                            Vehicle1.CarCurT = 0;
                                        }

                                        //Start to calculate vehicle's reactiontime, close-following or congested state, etc.
                                        if(Vehicle1.CarCurT == 0)
                                        {
                                            InMergeArea = VehicleInMerge(PreVehicle1.Position);

                                            if(k == 0)
                                            {
                                                if(Vehicle2.ID == -999)
                                                {
                                                    FrontVehicle.Speed = Vehicle2.Speed;
                                                    FrontVehicle.Position = Vehicle2.Position;
                                                    Front2Vehicle.Speed = Vehicle2.Speed;
                                                    Front2Vehicle.Position = Vehicle2.Position;
                                                }
                                                else
                                                {

        FrontVehicle.Speed=__min(Vehicle1.Type.VehicleStatic_SpeedLimit,
g_environment.Motorway_SpeedLimit);

        FrontVehicle.Position=g_environment.Motorway_End.x+g_environment.Motorway_Buffer_Len;

        Front2Vehicle.Speed=__min(Vehicle1.Type.VehicleStatic_SpeedLimit,
g_environment.Motorway_SpeedLimit);
Front2Vehicle.Position=g_environment.Motorway_End.x+g_environment.Motorway_Buffer_Len;
                                                }
                                            }

                                            if(Vehicle1.Speed < g_critical_speed) //Congested Traffic
                                            {
                                                Vehicle1.CarT = g_reactime_alert*10;
                                                Vehicle1.Congested = 1;
                                                Vehicle1.CarCloseFollowing = 0;
                                                Vehicle1.ReactionTime = g_reactime_alert;
                                            }
                                            else //Non Congested Traffic
                                            {
                                                if(k == 0 && Vehicle2.ID == -999)
```

```
{
        if(Vehicle1.Speed < g_critical_speed) //congested
        {
                Vehicle1.CarT = g_reactime_alert*10;
                Vehicle1.Congested = 1;
                Vehicle1.ReactionTime = g_reactime_alert;
        }
        else
        {
                Vehicle1.CarT = g_reactime_surprised*10;
                Vehicle1.Congested = 0;
                Vehicle1.ReactionTime =
Traffic
g_reactime_surprised;
        }
        Vehicle1.CarCloseFollowing = 0;
}
else
{
        if(g_gap_closefollowing == 1) //headway
        {
                headwaygap1=(FrontVehicle.Position-
Vehicle1.Position)/Vehicle1.Speed;
                headwaygap2=(Front2Vehicle.Position-
FrontVehicle.Position)/FrontVehicle.Speed;
        }
        else    //Gap
        {
                headwaygap1=(FrontVehicle.Position-
Vehicle1.Position- FrontVehicle.Length)/Vehicle1.Speed;
                headwaygap2 = (Front2Vehicle.Position -
FrontVehicle.Position - Front2Vehicle.Length)/FrontVehicle.Speed;
        }
        tempacc1=(FrontVehicle.Speed-Vehicle1.pregip.FrontSpeed)/PreVehicle1.CarT;
        tempacc2=(Front2Vehicle.Speed-Vehicle1.pregip.Front2Speed)/PreVehicle1.CarT;

        if(headwaygap1<g_tc_closefollowing &&
tempacc1>g_ac_closefollowing && (headwaygap2>g_tc_closefollowing || tempacc2>g_ac_closefollowing))
        //Meeting Close-following Requirement
        {
                long tmprand;
                float birand;
                tmprand = rand();
                birand =
ran0.bnldev(g_percentage_closefollowing/100.0,1,&tmprand);
                if(birand)
                {
                        tempDV = FrontVehicle.Speed-
Vehicle1.Speed;
                        tempDX =
FrontVehicle.Position-Vehicle1.Position;
                        tempAX =
g_ax_closefollowing+Vehicle1.Length;
                        tempBX = g_bx_closefollowing;
                        tempEX = g_ex_closefollowing;
                        tempABX =
tempAX+tempBX*sqrt(Vehicle1.Speed);
                        tempSDX =
tempAX+tempBX*(sqrt(Vehicle1.Speed*tempEX));

        if(tempDV>g_opdv_closefollowing || tempDV<g_cldv_closefollowing || tempDX>tempSDX ||
tempDX<tempABX)
                        {
                                Vehicle1.CarT =
g_reactime_alert * 10;
```

```
                                                                    Vehicle1.Congested =
1;
        Vehicle1.CarCloseFollowing = 0;
        Vehicle1.ReactionTime = g_reactime_alert;
                                                                }
                                                                else
                                                                {
                                                                    //D1
                                                                    if(tempDX                <
(tempSDX + tempABX) / 2)
                                                                    {
        Vehicle1.CarCloseFollowing = 2;
                                                                    }
                                                                    //A1
                                                                    else
                                                                    {
        Vehicle1.CarCloseFollowing = 1;
                                                                    }
                                                                    Vehicle1.CarT        =
g_reactime_closefollowing * 10;
                                                                    Vehicle1.Congested =
1;
        Vehicle1.ReactionTime = g_reactime_closefollowing;
                                                                }
                                                            }
                                                            else
                                                            {
                                                                Vehicle1.CarT                =
g_reactime_surprised * 10;
                                                                Vehicle1.Congested = 0;
                                                                Vehicle1.CarCloseFollowing = 0;
                                                                Vehicle1.ReactionTime        =
g_reactime_surprised;
                                                            }
                                                        }
                                                        else
                                                        {
                                                            Vehicle1.CarT  =  g_reactime_surprised  *
10;
                                                            Vehicle1.Congested = 1;
                                                            Vehicle1.CarCloseFollowing = 0;
                                                            Vehicle1.ReactionTime                =
g_reactime_surprised;
                                                        }
                                                    }
                                                }
                                                if(InMergeArea && Vehicle1.yielding)
                                                {
                                                    bool hasyieldvehicle;
                                                    VehicleStru YieldedVehicle;
                                                    hasyieldvehicle                                    =
GetLeaderAccLane(Vehicle1.Position,FrontVehicle.Position,YieldedVehicle);
                                                    if(hasyieldvehicle && !(Vehicle1.IsLaneChanging))
                                                    {
                                                        float tmp_timegap;
                                                        tmp_timegap    =    (YieldedVehicle.Position    -
Vehicle1.Position)/Vehicle1.Speed;
                                                        if(tmp_timegap    <=    g_yield_maxgap    &&
tmp_timegap >= g_yield_mingap)
                                                        {
```

```
YieldedVehicle.Position;
YieldedVehicle.Speed;
FrontVehicle.Position;
FrontVehicle.Speed;
YieldedVehicle.Length;
                Vehicle1.pregip.FrontPosition = fprintf(fp_yielding,"%16.2f\t%16d\t%16.4f\n",cur_time * computet,Vehicle1.ID,Vehicle1.Position);
                Vehicle1.pregip.FrontSpeed =
                Vehicle1.pregip.Front2Position =
                Vehicle1.pregip.Front2Speed =
                Vehicle1.pregip.FrontLength =
                yieldingcount ++;
                Vehicle1.yielding = 2;  /If there is an attempt merge and motorway driver is willing to yield, then the yielding value is 2; else value is 1 simply means the driver is will to provide yielding.
        }
        else
        {
                Vehicle1.pregip.FrontSpeed =
FrontVehicle.Speed;
                Vehicle1.pregip.FrontPosition =
FrontVehicle.Position;
                Vehicle1.pregip.Front2Speed =
Front2Vehicle.Speed;
                Vehicle1.pregip.Front2Position =
Front2Vehicle.Position;
                Vehicle1.pregip.FrontLength =
FrontVehicle.Length;
        }
}
else
{
        Vehicle1.pregip.FrontSpeed = FrontVehicle.Speed;
        Vehicle1.pregip.FrontPosition =
FrontVehicle.Position;
        Vehicle1.pregip.Front2Speed =
Front2Vehicle.Speed;
        Vehicle1.pregip.Front2Position =
Front2Vehicle.Position;
        Vehicle1.pregip.FrontLength = FrontVehicle.Length;
}
}
else
{
        Vehicle1.pregip.FrontSpeed = FrontVehicle.Speed;
        Vehicle1.pregip.FrontPosition = FrontVehicle.Position;
        Vehicle1.pregip.Front2Speed = Front2Vehicle.Speed;
        Vehicle1.pregip.Front2Position = Front2Vehicle.Position;
        Vehicle1.pregip.FrontLength = FrontVehicle.Length;
}
Vehicle1.pregip.Speed = Vehicle1.Speed;
Vehicle1.pregip.Position = Vehicle1.Position;
Vehicle1.pregip.Acceleration = Vehicle1.Acceleration;

if(Vehicle1.IsPFVehicle == 1)
{
        Vehicle1.CarT = g_reactime_merge * 10;
        Vehicle1.ReactionTime = g_reactime_merge;
}

if(Vehicle1.CarCloseFollowing == 0)
{
        tempt = Vehicle1.CarT/10.0;

        if(Vehicle1.Congested)
                an = ran0.getNorDis(g_accmean_alert,
```

```
g_accdevi_alert);

g_accdevi_surprised);

                                                                                else
                                                                                        an      =       ran0.getNorDis(g_accmean_surprised,

                                                                                float ttt_ppp1;

                                                                                if(Vehicle1.IsPFVehicle == 1)
                                                                                {
                                                                                        bn = g_merge_bn;
                                                                                        if(g_merge_b1 == 1)
                                                                                        {
                                                                                                b_severebra = g_merge_b2;
                                                                                        }
                                                                                        else
                                                                                        {
                                                                                                b_severebra   =   __min(-3.0,   ((-2.0)*an-
3.0)/2);

                                                                                        }
                                                                                        ttt_ppp1 = g_reactime_merge;
                                                                                        an          =           ran0.getNorDis(g_accmean_alert,
g_accdevi_alert);
                                                                                }
                                                                                else
                                                                                {
                                                                                        bn = (-2.0)*an;
                                                                                        b_severebra = __min(-3.0, (bn-3.0)/2);
                                                                                        ttt_ppp1 = tempt;
                                                                                }
                                                                                sn = Vehicle1.pregip.FrontLength;
                                                                                vn                                                              =
ran0.getNorDis(Vehicle1.Type.VehicleStatic_SpeedDesiredMean,Vehicle1.Type.VehicleStatic_SpeedDesiredD
evi);
                                                                                float tmp_speed1,tmp_speed2;

                                                                                if(k != 0)
                                                                                {
                                                                                        Vehicle1.NextSpeed                                      =
__min(Vehicle1.pregip.Speed   +   2.5   *   an   *   ttt_ppp1   *   (1-Vehicle1.pregip.Speed/vn)   *
sqrt(0.025+Vehicle1.pregip.Speed/vn),
                                                                                                                                                __max(bn * ttt_ppp1 + sqrt( bn *
bn * ttt_ppp1 * ttt_ppp1 - bn * (2 * (Vehicle1.pregip.FrontPosition - sn - Vehicle1.pregip.Position) -
Vehicle1.pregip.Speed * ttt_ppp1 -
                                                                                                                                                Vehicle1.pregip.FrontSpeed    *
Vehicle1.pregip.FrontSpeed / b_severebra)),Vehicle1.pregip.Speed+b_severebra*ttt_ppp1));
                                                                                }
                                                                                else
                                                                                {
                                                                                        if(Vehicle1.pregip.FrontPosition                        >=
g_environment.Motorway_End.x && Vehicle1.pregip.FrontPosition <= g_environment.Motorway_End.x +
g_environment.Motorway_Buffer_Len)
                                                                                        {
                                                                                                Vehicle1.NextSpeed                              =
__min(Vehicle1.pregip.Speed   +   2.5   *   an   *   ttt_ppp1   *   (1-Vehicle1.pregip.Speed/vn)   *
sqrt(0.025+Vehicle1.pregip.Speed/vn),
                                                                                                                                                __max(bn * ttt_ppp1 +
sqrt( bn * bn * ttt_ppp1 * ttt_ppp1 - bn * (2 * (Vehicle1.pregip.FrontPosition - sn - Vehicle1.pregip.Position) -
Vehicle1.pregip.Speed * ttt_ppp1 -

        Vehicle1.pregip.FrontSpeed              *               Vehicle1.pregip.FrontSpeed              /
b_severebra)),Vehicle1.pregip.Speed+b_severebra*ttt_ppp1));
                                                                                        }
                                                                                        else
                                                                                        {
                                                                                                tmp_speed1 = Vehicle1.pregip.Speed +
2.5 * an * ttt_ppp1 * (1-Vehicle1.pregip.Speed/vn) * sqrt(0.025+Vehicle1.pregip.Speed/vn);
```

```
Vehicle1.pregip.Speed + an * tempt;

__min(tmp_speed1,Vehicle1.NextSpeed);
                                                    Vehicle1.NextSpeed                    =

                                                    Vehicle1.NextSpeed                    =
                                                }
                                            }

                                            float dV;
                                            dV = Vehicle1.NextSpeed - Vehicle1.Speed;
                                            Vehicle1.dSpeed = dV / (Vehicle1.CarT/2);
                                    }

                            }
                                    Front2Vehicle = vehicle[i].CopyVehicle(FrontVehicle);
                                    FrontVehicle = vehicle[i].CopyVehicle(Vehicle1);

                                    vehicle[i].VehicleList.SetAt(tmpPrePosIndex,Vehicle1);

                                    k++;
                            }


                            PosIndex=vehicle[i].VehicleList.GetHeadPosition();
                            while(PosIndex!=NULL)
                            {
                                    tmpPrePosIndex = PosIndex;
                                    Vehicle1 = vehicle[i].VehicleList.GetNext(PosIndex);

        //                          Vehicle1.preveh.Acceleration = (Vehicle1.Speed - Vehicle1.preveh.Speed) /
computetet;

                                    Vehicle1.preveh.Speed = Vehicle1.Speed;
                                    Vehicle1.preveh.Position = Vehicle1.Position;
                                    Vehicle1.preveh.Length = Vehicle1.Length;
                                    Vehicle1.preveh.Acceleration = Vehicle1.Acceleration;
                                    Vehicle1.preveh.FrontSpeed = FrontVehicle.Speed;
                                    Vehicle1.preveh.FrontPosition = FrontVehicle.Position;
                                    Vehicle1.preveh.Front2Speed = Front2Vehicle.Speed;
                                    Vehicle1.preveh.Front2Position = Front2Vehicle.Position;

                                    vehicle[i].VehicleList.SetAt(tmpPrePosIndex,Vehicle1);
                            }

                            PosIndex=vehicle[i].VehicleList.GetHeadPosition();
                            while(PosIndex!=NULL)
                            {
                                    tmpPrePosIndex = PosIndex;
                                    Vehicle1 = vehicle[i].VehicleList.GetNext(PosIndex);

                                    if(Vehicle1.Position > g_environment.Motorway_End.x)
                                    {
                                            vehicle[i].VehicleList.RemoveAt(tmpPrePosIndex);
                                    }
                                    else
                                    {
                                            break;
                                    }
                            }

                            PosIndex=vehicle[i].VehicleList.GetHeadPosition();
                            while(PosIndex!=NULL)
                            {
                                    tmpPrePosIndex = PosIndex;
                                    Vehicle1 = vehicle[i].VehicleList.GetNext(PosIndex);

                                    if(Vehicle1.IsLaneChanging &&
```

```
                        (Vehicle1.Position - Vehicle1.LaneChangingPos) <= 5 &&
                        (Vehicle1.Position - Vehicle1.LaneChangingPos) >= -5 &&
                        (Vehicle1.Type.VehicleType_Name[0] == 'C'))
                {
                        vehicle[i].VehicleList.RemoveAt(tmpPrePosIndex);
                        fprintf(fp_lanechanging,"%16.2f\t%16d\t%16.4f\n",cur_time *
computet,Vehicle1.ID,Vehicle1.Position);
                        lanechangingcount ++;
                }
                else if(Vehicle1.IsLaneChanging_Down &&
                        (Vehicle1.Position - Vehicle1.LaneChangingPos_Down) <= 5 &&
                        (Vehicle1.Position - Vehicle1.LaneChangingPos_Down) >= -5 &&
                        (Vehicle1.Type.VehicleType_Name[0] == 'C'))
                {
                        vehicle[i].VehicleList.RemoveAt(tmpPrePosIndex);
                        fprintf(fp_lanechanging_down,"%16.2f\t%16d\t%16.4f\n",cur_time *
computet,Vehicle1.ID,Vehicle1.Position);
                        lanechangingcount_down ++;
                }

        }

        if(cur_time-1 >= g_warmup_period/computet)
        {
                VehicleStru tmppreveh;
                POSITION tmpprepos;

                fprintf(fp_individual[i],"%4.1f\t%6d\t",(cur_time-
1)*computet,vehicle[i].PreVehicleList.GetCount());

                tmpprepos=vehicle[i].PreVehicleList.GetHeadPosition();
                while(tmpprepos!=NULL)
                {
                        tmppreveh = vehicle[i].PreVehicleList.GetNext(tmpprepos);

        fprintf(fp_individual[i],"%5d\t%7.2f\t%8.2f\t%6.3f\t",tmppreveh.ID,tmppreveh.Position,tmppreveh.S
peed,tmppreveh.Acceleration);
                }
        //      fclose(fp_individual[i]);
                fprintf(fp_individual[i],"\n");


        /*      tmpprepos=vehicle[i].PreVehicleList.GetHeadPosition();
                while(tmpprepos!=NULL)
                {
                        tmppreveh = vehicle[i].PreVehicleList.GetNext(tmpprepos);
                        dbcancel(dbproc);
                        if(i==10)
                                dbcmd(dbproc,                                   "INSERT
Motorway_Lane_1(time,vehicle_id,vehicle_speed,vehicle_acceleration,vehicle_position)");
                        else if(i==11)
                                dbcmd(dbproc,                                   "INSERT
Motorway_Lane_2(time,vehicle_id,vehicle_speed,vehicle_acceleration,vehicle_position)");
                        dbfcmd(dbproc,"         VALUES(%d,%d,%f,%f,%f)",(int)((cur_time-
1)*computet*10+.5),tmppreveh.ID,tmppreveh.Speed,tmppreveh.Acceleration,tmppreveh.Position);
                        if(dbsqlexec(dbproc)==FALSE)
                        {
                                AfxMessageBox("error    first    insert:    INSERT
Motorway_Lane_1");
                                return;
                        }
                }
        */      }

        int tt,tt1;
        tt = (int)(g_lane[i-10].AddVehicleTime/computet+.5);
```

```
if(cur_time == tt)
{
        tt1 = vehicle[i].VehicleList.GetCount();
        if(tt1 == 0)
        {
                if(Vehicle2.ID == -999)
                {
                        LastVehicle.Position = Vehicle2.Position;
                        LastVehicle.Length = Vehicle2.Length;
                }
                else
                {
                        LastVehicle.Position = g_environment.Motorway_End.x +
g_environment.Motorway_Buffer_Len;
                        LastVehicle.Length =
g_vehicle_type[0].VehicleType_Length_Mean;
                }
        }
        else
        {
                LastVehicle = vehicle[i].VehicleList.GetTail();
        }

        if(LastVehicle.Position - g_environment.Motorway_Start.x - LastVehicle.Length
>= g_lane[i-10].Lane_Speed_Mean)
        {
                PosLastIndex = vehicle[i].VehicleList.GetTailPosition();
                if(g_lane[i-10].AddedVehicleCount+1 == g_lane[i-
10].CurStepAddHGVCount)
                {
                        AddVehicle.Type.VehicleType_ID =
g_vehicle_type[1].VehicleType_ID;

                strcpy(AddVehicle.Type.VehicleType_Name,g_vehicle_type[1].VehicleType_Name);
                        AddVehicle.Type.VehicleStatic_SpeedLimit =
g_vehicle_type[1].VehicleStatic_SpeedLimit;
                        AddVehicle.Type.VehicleStatic_SpeedDesiredMean =
g_vehicle_type[1].VehicleStatic_SpeedDesiredMean;
                        AddVehicle.Type.VehicleStatic_SpeedDesiredDevi =
g_vehicle_type[1].VehicleStatic_SpeedDesiredDevi;
                        AddVehicle.Type.VehicleType_Color =
g_vehicle_type[1].VehicleType_Color;
                        AddVehicle.Type.VehicleType_Length_Mean =
g_vehicle_type[1].VehicleType_Length_Mean;
                        AddVehicle.Type.VehicleType_Length_Devi =
g_vehicle_type[1].VehicleType_Length_Devi;
                        cur_vehicle_is_HGV = true;
                        g_lane[i-10].AddedHGVCount ++;
                        g_lane[i-10].CurStepAddHGVCount =
GetAddHGVCount(g_lane[i-10].CurAddHGVBase, g_lane[i-10].AddHGVStep);
                }
                //Passenger car
                else
                {
                        AddVehicle.Type.VehicleType_ID =
g_vehicle_type[0].VehicleType_ID;

                strcpy(AddVehicle.Type.VehicleType_Name,g_vehicle_type[0].VehicleType_Name);
                        AddVehicle.Type.VehicleStatic_SpeedLimit =
g_vehicle_type[0].VehicleStatic_SpeedLimit;
                        AddVehicle.Type.VehicleStatic_SpeedDesiredMean =
g_vehicle_type[0].VehicleStatic_SpeedDesiredMean;
                        AddVehicle.Type.VehicleStatic_SpeedDesiredDevi =
g_vehicle_type[0].VehicleStatic_SpeedDesiredDevi;
                        AddVehicle.Type.VehicleType_Color =
g_vehicle_type[0].VehicleType_Color;
                        AddVehicle.Type.VehicleType_Length_Mean =
```

```
g_vehicle_type[0].VehicleType_Length_Mean;
                                        AddVehicle.Type.VehicleType_Length_Devi                    =
g_vehicle_type[0].VehicleType_Length_Devi;
                                        cur_vehicle_is_HGV = false;
                                }
                                g_lane[i-10].AddedVehicleCount ++;
                                Vehicle_ID ++;

                                //initial the added vehicle
                                AddVehicle.ID = Vehicle_ID;

                                LastVehicle.BehID = AddVehicle.ID;
                                if(vehicle[i].VehicleList.GetCount()>0)
                                {
                                        vehicle[i].VehicleList.SetAt(PosLastIndex,LastVehicle);
                                }

                                AddVehicle.Position = g_environment.Motorway_Start.x;
                                AddVehicle.Speed          =          ran0.getNorDis(g_lane[i-
10].Lane_Speed_Mean,g_lane[i-10].Lane_Speed_Devi);
        //                      if(AddVehicle.Speed     >     g_environment.Onramp_SpeedLimit     ||
AddVehicle.Speed > AddVehicle.Type.VehicleStatic_SpeedLimit)
        //                              AddVehicle.Speed                                                =
__min(g_environment.Onramp_SpeedLimit,AddVehicle.Type.VehicleStatic_SpeedLimit);
                                AddVehicle.Length                                                       =
ran0.getNorDis(AddVehicle.Type.VehicleType_Length_Mean,AddVehicle.Type.VehicleType_Length_Devi);
                                AddVehicle.Gap          =          (float)(LastVehicle.Position          -
g_environment.Motorway_Start.x - AddVehicle.Length);
                                AddVehicle.CarCurT = 0;
                                AddVehicle.CarCloseFollowing = 0;
                                AddVehicle.PreID = LastVehicle.ID;
                                AddVehicle.BehID = 0;
                                AddVehicle.Headway          =          LastVehicle.Position          -
g_environment.Motorway_Start.x;
                                AddVehicle.Acceleration = 0;
                                AddVehicle.Courtesy = 0;
                                AddVehicle.DriverType = 0;
                                AddVehicle.IsFreeSpeed = 0;
                                AddVehicle.LaneID = i+1;
                                AddVehicle.pregip.Speed = AddVehicle.Speed;
                                AddVehicle.pregip.Position = AddVehicle.Position;
                                AddVehicle.pregip.Acceleration = AddVehicle.Acceleration;
                                AddVehicle.pregip.FrontSpeed = LastVehicle.Speed;
                                AddVehicle.pregip.FrontPosition = LastVehicle.Position;
                                AddVehicle.pregip.Front2Speed = Front2Vehicle.Speed;
                                AddVehicle.pregip.Front2Position = Front2Vehicle.Position;
                                AddVehicle.pregip.FrontLength = LastVehicle.Length;
                                AddVehicle.preveh.Speed = AddVehicle.Speed;
                                AddVehicle.preveh.Position = AddVehicle.Position;
                                AddVehicle.preveh.Acceleration = AddVehicle.Acceleration;
                                AddVehicle.preveh.FrontSpeed = LastVehicle.Speed;
                                AddVehicle.preveh.FrontPosition = LastVehicle.Position;
                                AddVehicle.preveh.Front2Speed = Front2Vehicle.Speed;
                                AddVehicle.preveh.Front2Position = Front2Vehicle.Position;
                                AddVehicle.IsPFVehicle = 0;

                                long tmprand;
                                tmprand = rand();
                                AddVehicle.IsLaneChanging                                               =
ran0.bnldev(g_percentage_lanechanging/100.0,1,&tmprand);
                                if(AddVehicle.IsLaneChanging == 1)
                                {
                                        AddVehicle.LaneChangingPos                                      =
g_environment.Motorway_Start.x      +      ran0.getRand01()      *      (g_environment.AccLane_End.x      -
g_environment.Motorway_Start.x);
                                        AddVehicle.IsLaneChanging_Down = 0;
```

```
                                        }
                                        else
                                        {
                                                AddVehicle.IsLaneChanging_Down                  =
ran0.bnldev(g_percentage_lanechanging_down/100.0,1,&tmprand);
                                                if(AddVehicle.IsLaneChanging_Down == 1)
                                                {
                                                        AddVehicle.LaneChangingPos_Down         =
g_environment.AccLane_End.x + ran0.getRand01() * (g_end_pos_down - g_environment.AccLane_End.x);
                                                }
                                        }

                                        AddVehicle.C                                            =
GetC(g_cc,AddVehicle.DriverType,AddVehicle.Alpha,AddVehicle.C1,AddVehicle.C2,AddVehicle.C3,AddVeh
icle.Alpha_Lead,AddVehicle.Alpha_Lag);
                                        if(AddVehicle.Speed > g_critical_speed)
                {
                                                AddVehicle.ReactionTime = g_reactime_surprised;
                                                AddVehicle.Congested = 0;
                                        }
                                        else
                                        {
                                                AddVehicle.ReactionTime = g_reactime_alert;
                                                AddVehicle.Congested = 1;
                                        }
                                        AddVehicle.CarT = AddVehicle.ReactionTime * 10;

                                        tmprand = rand();
                                        AddVehicle.yielding                                     =
ran0.bnldev(g_percentage_merge/100.0,1,&tmprand);

                                                tempt = AddVehicle.CarT/10.0;

                                        if(AddVehicle.Congested)
                                                an      =       ran0.getNorDis(g_accmean_alert,
g_accdevi_alert);
                                        else
                                                an      =       ran0.getNorDis(g_accmean_surprised,
g_accdevi_surprised);

                                        float ttt_ppp2;

                                        bn = (-2.0)*an;
                                        b_severebra = __min(-3.0, (bn-3.0)/2);
                                        ttt_ppp2 = tempt;
                                        sn = AddVehicle.pregip.FrontLength;

        vn=ran0.getNorDis(AddVehicle.Type.VehicleStatic_SpeedDesiredMean,AddVehicle.Type.VehicleSta
t
    ic_SpeedDesiredDevi);
                                        float tmp_speed1,tmp_speed2;
                                        AddVehicle.NextSpeed = __min(AddVehicle.pregip.Speed +
2.5 * an * ttt_ppp2 * (1-AddVehicle.pregip.Speed/vn) * sqrt(0.025+AddVehicle.pregip.Speed/vn),
                                                                __max(bn * ttt_ppp2 + sqrt( bn * bn *
ttt_ppp2 * ttt_ppp2 - bn * (2 * (AddVehicle.pregip.FrontPosition - sn - AddVehicle.pregip.Position) -
AddVehicle.pregip.Speed * ttt_ppp2 -
                                                                        AddVehicle.pregip.FrontSpeed    *
AddVehicle.pregip.FrontSpeed / b_severebra)),AddVehicle.pregip.Speed+b_severebra*ttt_ppp2));

                                                float dV;
                                                dV = AddVehicle.NextSpeed - AddVehicle.Speed;
                                                AddVehicle.dSpeed = dV / (AddVehicle.CarT/2);

                                //Add the first vehicle into vehicle list
                                vehicle[i].Add(AddVehicle);
                                if(AddVehicle.Type.VehicleType_Name[0] == 'C')
                                {
```

```
        fprintf(fp_vehiclestatic,"%5d\t%7.2f\t%8s\t%7d\t%3s\n",AddVehicle.ID,AddVehicle.Length,"0",cur_t
ime,"M");
        //                                                      fprintf(fp_newadd,"%5.1f\t%6d\t%12d\t%9s\t%6s\n",
        //                                                              cur_time                                        *
computet,AddVehicle.ID,AddVehicle.DriverType,"P","Motor");
                                                                fprintf(fp_newaddmotor,"%5.1f\t%6d\t%12d\t%9s\t%6s\n",
                                                                cur_time                                        *
computet,AddVehicle.ID,AddVehicle.DriverType,"P","Motor");
                                                }
                                                else if(AddVehicle.Type.VehicleType_Name[0] == 'H')
                                                {

        fprintf(fp_vehiclestatic,"%5d\t%7.2f\t%8s\t%7d\t%3s\n",AddVehicle.ID,AddVehicle.Length,"1",cur_t
ime,"M");
        //                                                      fprintf(fp_newadd,"%5.1f\t%6d\t%12d\t%9s\t%6s\n",
        //                                                              cur_time                                        *
computet,AddVehicle.ID,AddVehicle.DriverType,"HGV","Motor");
                                                                fprintf(fp_newaddmotor,"%5.1f\t%6d\t%12d\t%9s\t%6s\n",
                                                                cur_time                                        *
computet,AddVehicle.ID,AddVehicle.DriverType,"HGV","Motor");
                                                }

                                                //get the add vehicle time
                                                g_lane[i-10].AddVehicleTime     =       GetAddVehicleTime(g_lane[i-
10].AddVehicleTime,g_lane[i-10].Lane_Flow);
                                        }
                                        else
                                        {
                                                g_lane[i-10].AddVehicleTime ++;
                                        }
                        }

                        vehicle[i].CopyVehicleList();
                        vehicle[i].Save2TenVehicle();

                        vehicle_out[i].VehicleList.RemoveAll();
                        if(Vehicle2.ID == -999)
                        {
                                vehicle_out[i].Add(Vehicle2);

        fprintf(fp_fakeveh,"%5.1f\t%12.2f\t%12.2f\n",cur_time*computet,Vehicle2.Position,Vehicle2.Speed);
                        }
                        else
                        {
                                fprintf(fp_fakeveh,"%5.1f\t%12.2f\t%12.2f\n",cur_time*computet,-999.0,-999.0);
                        }
                        vehicle_out[i].CopyVehicleList();
        }
}
```

A.3 Accelerationlanemodel() Source Codes

```
void CMergeModel::AccelerationLaneModel ()
{
    VehicleStru Vehicle1,LastVehicle,AddVehicle;
    int i;
    int CurStepAddHGVCount;
    int CurAddHGVBase;
    bool cur_vehicle_is_HGV=false;

    float ACMinus, ACPlus;
    float AC_Max_Minus, AC_Max_Plus;
    float tmp_speed1, tmp_speed2;
    int list_vehicle_count;
    float BC;
    int tmp_count;
    int k;
    float tempt;

    float PLHeadway,PFHeadway;
    float AccHeadway;

    BOOL CanMerge=FALSE;

    //the front vehicle in this time period
    VehicleStru Front2Vehicle;
    VehicleStru FrontVehicle;
VehicleStru PLVehicle,PFVehicle;          //motorway  putative leader and putative follower
    VehicleStru PreVehicle1;
    int CooperationValue;
    float an,bn,sn,vn,b_severebra;
    float headwaygap1,headwaygap2,tempacc1,tempacc2;
    float tempDV,tempDX,tempABX,tempSDX,tempAX,tempBX,tempEX;

POSITION InsertPosIndex;
VehicleStru tmpVehicle;

POSITION PosIndex, tmpPrePosIndex;
float cur_lead, cur_lag, s_ave_lead, s_ave_lag, s_lead, s_lag;
BOOL IsDel=FALSE;

for(i=0;i<g_lane_ramp_count;i++)
{
        tmp_count=0;
        list_vehicle_count=0;
        CooperationValue=0;

        PosIndex=vehicle[i].VehicleList.GetHeadPosition();
        k=0;

        //Calculate all ramp vehicles' data
        while(PosIndex!=NULL)
        {
            tmpPrePosIndex=PosIndex;
            Vehicle1=vehicle[i].VehicleList.GetNext(PosIndex);
            PreVehicle1=vehicle[i].CopyVehicle(Vehicle1);

            if(Vehicle1.pregip.Position>=g_environment.AccLane_Start.x)
            {
                Vehicle1.CarCurT=Vehicle1.CarCurT+computet*10;
                if(Vehicle1.CarCurT<Vehicle1.CarT)
                {

        Vehicle1.Speed=Vehicle1.preveh.Speed+PreVehicle1.Acceleration*computet;
        Vehicle1.Position=Vehicle1.preveh.Position+Vehicle1.preveh.Speed*computet+PreVehicle1.Accelera
tion*computet*computet/2;
                                Vehicle1.Acceleration=Vehicle1.pregip.Acceleration;
```

```
FrontVehicle.Length<=0)

FrontVehicle.Length-1;



FrontVehicle.Length-Vehicle1.Position;

Vehicle1.Position;
                                                        }
                                                        if(k>0)
                                                        {
                                                                if(FrontVehicle.Position-Vehicle1.Position-

                                                                {
                                                                        Vehicle1.Position=FrontVehicle.Position-

                                                                }
                                                        }
                                                        if(Vehicle1.PreID != 0)
                                                        {
                                                                Vehicle1.Gap=FrontVehicle.Position-

                                                                Vehicle1.Headway=FrontVehicle.Position-
                                                        }
                                                        else
                                                        {

        Vehicle1.Gap=g_environment.AccLane_End.x+g_environment.AccLane_Buffer_Len-
Vehicle1.Position;

        Vehicle1.Headway=g_environment.AccLane_End.x+g_environment.AccLane_Buffer_Len-
Vehicle1.Position;
                                                        }
                                                }
                                                // reach the reaction time interval
                                                else
                                                {

        Vehicle1.Speed=Vehicle1.pregip.Speed+Vehicle1.preveh.Acceleration*g_reactime_acc;
        Vehicle1.Position=Vehicle1.pregip.Position+Vehicle1.pregip.Speed*g_reactime_acc+Vehicle1.preve
h.Acceleration*g_reactime_acc*g_reactime_acc/2;
                                                        if(k>0)
                                                        {
                                                                if(FrontVehicle.Position-Vehicle1.Position-
FrontVehicle.Length<=0)
                                                                {
        Vehicle1.Position=FrontVehicle.Position-FrontVehicle.Length-1;
                                                                }
                                                        }

                                                        //Get current time PF and PL vehicle ID and position
                                                        GetPLPFVehicle(Vehicle1.Position,PLVehicle,PFVehicle);


                                                        //Get current time acceleration rate
                                                        ACPlus=Vehicle1.C*(PLVehicle.Speed-
Vehicle1.Speed)/g_reactime_acc;
                                                        ACMinus                 =-Vehicle1.C*(PLVehicle.Speed-
Vehicle1.Speed)*(PLVehicle.Speed-Vehicle1.Speed)/(2*(PLVehicle.Position-Vehicle1.Position-
PLVehicle.Length));
                                                                if(Vehicle1.Type.VehicleType_Name[0] == 'C')

        Speed2AccDec(Vehicle1.Speed*3.6,AC_Max_Plus,AC_Max_Minus, 0);
                                                        else        //HGV

        Speed2AccDec(Vehicle1.Speed*3.6,AC_Max_Plus,AC_Max_Minus, 1);

        PLHeadway=(PLVehicle.Position-Vehicle1.Position-PLVehicle.Length)/Vehicle1.Speed;
        PFHeadway=(Vehicle1.Position-PFVehicle.Position-Vehicle1.Length)/PFVehicle.Speed;
                                                        if(PLHeadway<4 && PFHeadway<4)
                                                        {
    if(PLVehicle.Position-Vehicle1.Position-PLVehicle.Length<g_deltas        &&        Vehicle1.Speed-
PLVehicle.Speed>=-g_deltav && Vehicle1.Speed-PLVehicle.Speed<=g_deltav)
```

```
                                                    {
        ACMinus=-Vehicle1.C2*2*(g_deltas+PLVehicle.Length-(PLVehicle.Position-
Vehicle1.Position))/g_reactime_acc/g_reactime_acc;
        Vehicle1.Acceleration=__max(ACMinus,AC_Max_Minus);
        ACPlus=0;

                                                    }
                                                    else        if(Vehicle1.Position-PFVehicle.Position-
Vehicle1.Length<g_deltas &&
                                                                Vehicle1.Speed-PFVehicle.Speed>=-
g_deltav &&
                                                                Vehicle1.Speed-
PFVehicle.Speed<=g_deltav)
                                                    {
        ACPlus=Vehicle1.C2*2*(g_deltas+Vehicle1.Length-(Vehicle1.Position-
PFVehicle.Position))/g_reactime_acc/g_reactime_acc;

        Vehicle1.Acceleration=__min(ACPlus,AC_Max_Plus);
                                                                ACMinus=0;
                                                    }
                                                    else
                                                    {
                                                                if(Vehicle1.Speed<PLVehicle.Speed)
                                                                {

        ACPlus=Vehicle1.C1*(PLVehicle.Speed-Vehicle1.Speed)/g_reactime_acc;

        Vehicle1.Acceleration=__min(ACPlus,AC_Max_Plus);
                                                                            ACMinus=0;
                                                                }
                                                                else
                                                                {
                                                                            ACMinus=-
Vehicle1.C1*(PLVehicle.Speed-Vehicle1.Speed)*(PLVehicle.Speed-Vehicle1._
        Speed)/(2*(PLVehicle.Position-Vehicle1.Position-PLVehicle.Length));
                                                                                        if(ACMinus>0)
                                                                                                    ACMinus=-ACMinus;
         Vehicle1.Acceleration=__max(ACMinus,AC_Max_Minus);
                                                                                        ACPlus=0;
                                                                }
                                                    }
                                        }
                                        else if(PLHeadway<4 && PFHeadway>=4)
                                        {
        if(PLVehicle.Position-Vehicle1.Position-PLVehicle.Length<g_deltas &&
                                                Vehicle1.Speed-PLVehicle.Speed>=-g_deltav &&
                                                Vehicle1.Speed-PLVehicle.Speed<=g_deltav)
                                                    {
                                                                ACMinus=-
Vehicle1.C2*2*(g_deltas+PLVehicle.Length-(PLVehicle.Position-Vehicle1._
        Position))/g_reactime_acc/g_reactime_acc;
         Vehicle1.Acceleration=__max(ACMinus,AC_Max_Minus);
        ACPlus=0;
                                                    }
                                                    else
                                                    {
                                                                if(Vehicle1.Speed<PLVehicle.Speed)
                                                                {

        ACPlus=Vehicle1.C1*(PLVehicle.Speed-Vehicle1.Speed)/g_reactime_acc;

        Vehicle1.Acceleration=__min(ACPlus,AC_Max_Plus);
                                                                            ACMinus=0;
                                                                }
                                                                else
                                                                {
                                                                            ACMinus                    =-
```

```
Vehicle1.C1*(PLVehicle.Speed-Vehicle1.Speed)*(PLVehicle.Speed-Vehicle1.Speed)/(2*(PLVehicle.Position-
Vehicle1.Position-PLVehicle.Length));
                                                                    if(ACMinus>0)
                                                                        ACMinus=-ACMinus;

        Vehicle1.Acceleration=__max(ACMinus,AC_Max_Minus);
                                                                    ACPlus=0;
                                                                }
                                                            }
                                                        }
                                                        else if(PLHeadway>=4 && PFHeadway<4)
                                                        {
                                                            if(Vehicle1.Position-PFVehicle.Position-
Vehicle1.Length<g_deltas &&
                                                                Vehicle1.Speed-PFVehicle.Speed>=-
g_deltav &&
                                                                Vehicle1.Speed-
PFVehicle.Speed<=g_deltav)
                                                            {

        ACPlus=Vehicle1.C2*2*(g_deltas+Vehicle1.Length-(Vehicle1.Position-
PFVehicle.Position))/g_reactime_acc/g_reactime_acc;

        Vehicle1.Acceleration=__min(ACPlus,AC_Max_Plus);
                                                                ACMinus=0;
                                                            }
                                                            else
                                                            {
                                                                if(Vehicle1.Speed<PFVehicle.Speed)
                                                                {

        ACPlus=Vehicle1.C1*(PFVehicle.Speed-Vehicle1.Speed)/g_reactime_acc;

        Vehicle1.Acceleration=__min(ACPlus,AC_Max_Plus);
                                                                    ACMinus=0;
                                                                }
                                                                else
                                                                {
                                                                    Vehicle1.Acceleration=0;
                                                                    ACPlus=0;
                                                                    ACMinus=0;
                                                                }
                                                            }
                                                        }
                                                        else
                                                        {
                                                            CanMerge=TRUE;
                                                        }


tmp_speed1=Vehicle1.Speed+Vehicle1.Acceleration*g_reactime_acc;

                                                        if(Vehicle1.Position>g_environment.AccLane_End.x)
                                                        {
                                                            BC=g_max_b;
                                                        }
                                                        else
                                                        {
                                                            BC=__min(-g_max_b,
Vehicle1.C3*Vehicle1.Speed*Vehicle1.Speed/2/(g_environment.AccLane_End.x-Vehicle1.Position));
                                                            BC=__max(-g_min_b,BC);
                                                            BC=-BC;
                                                        }

                                                        if(Vehicle1.PreID != 0)
                                                        {
                                                            ////////////////////////Step2: Interacting with the vehicle
```

```
in front on the acceleration lane itself
                                                        //Get Parameter C
                                                        AccHeadway=(Vehicle1.preveh.Position-
Vehicle1.Position-Vehicle1.preveh.Length)/Vehicle1.Speed;
                                                        if(AccHeadway>=4)
                                                        {
                                                                tmp_speed2=999.0f;
                                                        }
                                                        else
                                                        {

        tmp_speed2=BC*g_reactime_acc+sqrt(BC*BC*g_reactime_acc*g_reactime_acc-
BC*(2*(FrontVehicle.Position-FrontVehicle.Length-Vehicle1.Position)-Vehicle1.Speed*g_reactime_acc-
FrontVehicle.Speed*FrontVehicle.Speed/g_1_b));
                                                        }
                                                }
                                                else
                                                {
                                                        AccHeadway=(g_environment.AccLane_End.x-
Vehicle1.Position)/Vehicle1.Speed;
                                                        tmp_speed2=999.0f;
                                                }

                                                //Get speed for next reaction time
                                                Vehicle1.NextSpeed=__min(tmp_speed1,tmp_speed2);
                                                //Get current Gap and Headway
                                                if(Vehicle1.PreID != 0)
                                                {
                                                        Vehicle1.Gap=FrontVehicle.Position-
FrontVehicle.Length-Vehicle1.Position;
                                                        Vehicle1.Headway=FrontVehicle.Position-
Vehicle1.Position;
                                                }
                                                else
                                                {

        Vehicle1.Gap=g_environment.AccLane_End.x+g_environment.AccLane_Buffer_Len-
Vehicle1.Position;

        Vehicle1.Headway=g_environment.AccLane_End.x+g_environment.AccLane_Buffer_Len-
Vehicle1.Position;
                                                }

                                                ////////////////////////////step 3
                                                //Lead and lag threshold
                                                //Get parameters of b'pl,bpf,bc' from file
                                                if(g_trymodel)
                                                {
    /*                                                  float tmptt;
                                                        tmptt=(-
2.0*ran0.getNorDis(g_pl_b_mean,g_pl_b_devi)-3.0)/2;
                                                        g_pl_b=__min(-3.0,tmptt);
                                                        tmptt=(-
2.0*ran0.getNorDis(g_pl_b_mean,g_pl_b_devi)-3.0)/2;
                                                        g_c_b=__min(-3.0,tmptt);
                                                        tmptt=(-
2.0*ran0.getNorDis(g_pl_b_mean,g_pl_b_devi)-3.0)/2;
                                                        g_pf_b=__min(-3.0,tmptt);
                                                        tmptt=(-
2.0*ran0.getNorDis(g_pl_b_mean,g_pl_b_devi)-3.0)/2;
                                                        g_l_b=__min(-3.0,tmptt);
                                                        if(g_l_b>-3 || g_pl_b>-3 || g_pl_b>-3 || g_c_b>-3)
                                                        {
                                                                int jil;
                                                                jil=1;
                                                        }
                                                        g_pl_b=__min(-3.0,(-
    */
```

```
2.0*ran0.getNorDis(g_pl_b_mean,g_pl_b_devi)-3.0)/2);
                                                        g_c_b=__min(-3.0,(-
2.0*ran0.getNorDis(g_c_b_mean,g_c_b_devi)-3.0)/2);
                                                        g_pf_b=__min(-3.0,(-
2.0*ran0.getNorDis(g_pf_b_mean,g_pf_b_devi)-3.0)/2);
                                                        g_l_b=__min(-3.0,(-
2.0*ran0.getNorDis(g_l_b_mean,g_l_b_devi)-3.0)/2);
                                                }

                                                // If no PL, there is no limitation of safe lead gap
                                                if(PLVehicle.ID == 0)
                                                {
                                                        s_ave_lead=0;
                                                }
                                                else
                                                {

        s_ave_lead=(PLVehicle.Speed*PLVehicle.Speed/g_pl_b-
Vehicle1.NextSpeed*Vehicle1.NextSpeed/BC+2*g_reactime_acc*Vehicle1.NextSpeed+Vehicle1.Speed*g_reac
time_acc)/2;
                                                }

    //                                          s_ave_lag=(Vehicle1.Speed*Vehicle1.Speed/g_c_b-
PFVehicle.Speed*PFVehicle.Speed/g_pf_b+2*g_reactime_acc*PFVehicle.Speed+PFVehicle.Speed*g_reactime
_acc)/2;
                                                CooperationValue=Vehicle1.yielding;
                                                if(PFVehicle.ID == -999)
                                                {
                                                        s_ave_lag=0;
                                                }
                                                else
                                                {
                                                        if (CooperationValue == 1)
                                                        {

        s_ave_lag=(Vehicle1.Speed*Vehicle1.Speed/g_c_b-
(PFVehicle.Speed+g_pf_b*g_reactime_acc)*(PFVehicle.Speed+g_pf_b*g_reactime_acc)/g_pf_b+2*g_reactime
_acc*(PFVehicle.Speed+g_pf_b*g_reactime_acc)+PFVehicle.Speed*g_reactime_acc)/2; //change made
                                                        }
                                                        if (CooperationValue == 0)
                                                        {

        s_ave_lag=(Vehicle1.Speed*Vehicle1.Speed/g_c_b-
PFVehicle.Speed*PFVehicle.Speed/g_pf_b+2*g_reactime_acc*PFVehicle.Speed+PFVehicle.Speed*g_reactime
_acc)/2; //same as before
                                                        }
                                                }

                                                //Calculate Lead and Lag thresholds
                                                s_lead=g_aa* ran0.getNorDis(s_ave_lead,g_std_lead);
                                                s_lag=g_aa* ran0.getNorDis(s_ave_lag,g_std_lead);

                                                if(s_lead<4.5)
                                                {
                                                        s_lead=4.5;
                                                }
                                                if(s_lag<4.5)
                                                {
                                                        s_lag=4.5;
                                                }

                                                //Get the values of current Lead and Lag gap on the motorway
                                                if(PLVehicle.ID == 0)
                                                {
                                                        cur_lead=PLVehicle.Position-Vehicle1.Position;
                                                }
                                                else
```

```
                                                {
                                                        cur_lead=PLVehicle.Position-PLVehicle.Length-
Vehicle1.Position;
                                                }
                                                cur_lag=Vehicle1.Position-Vehicle1.Length-
PFVehicle.Position;

                                                Vehicle1.CarCurT=0;

                                                //If ramp vehicle runs to the end, remove and save vehicle
propertis
        if(Vehicle1.Position>=g_environment.AccLane_End.x+g_environment.AccLane_Buffer_Len)
                                                {
                                                        Vehicle1.MergeTime=cur_time*computet;
                                                        StopedVehicleCount ++;
                                                        vehicle[i].DeleteVehicle(tmpPrePosIndex);;

            fprintf(fp_mergedata_stop,"%5.1f\t%6d\t%18.2f\t%12.2f\t%15.2f\t%12d\n",

            Vehicle1.MergeTime,Vehicle1.ID,Vehicle1.Speed,Vehicle1.Position,Vehicle1.Length,Vehicle1.Drive
rType);
        //                                              IsDel=TRUE;
                                                        continue;
                                                }
                                                //Decide whether to merge
                                                else if(cur_lead>s_lead && cur_lag>s_lag || CanMerge ||
vehicle[10].VehicleList.GetCount()<=0)
                                                {
                                                        POSITION MotorwayPFPos,PreMotorwayPFPos;

            MotorwayPFPos=vehicle[10].VehicleList.GetHeadPosition();
                                                        VehicleStru tmpPFVehicle;

                                                while(PosIndex!=NULL)
                                                {
                                                        PreMotorwayPFPos=MotorwayPFPos;

            tmpPFVehicle=vehicle[10].VehicleList.GetNext(MotorwayPFPos);
                                                        if(tmpPFVehicle.ID == PFVehicle.ID)
                                                        {
                                                                tmpPFVehicle.IsPFVehicle=1;

            vehicle[10].VehicleList.SetAt(PreMotorwayPFPos,tmpPFVehicle);
                                                                break;
                                                        }
                                                }

                                                //Successful merge, change the ramp vehicle as a motorway
vheicle

                                                        Vehicle1.LaneID=MaxRampLaneNo+1;
                                                        Vehicle1.MergeTime=cur_time*computet;
                                                        long tmprand;
        /*                                              tmprand=rand();

            Vehicle1.IsLaneChanging_Down=ran0.bnldev(g_percentage_lanechanging_down/100.0,1,&tmprand);
                                                        if(Vehicle1.IsLaneChanging_Down == 1)
                                                        {

            Vehicle1.LaneChangingPos_Down=g_environment.AccLane_End.x+ran0.getRand01()*(g_end_pos_d
own-g_environment.AccLane_End.x);
                                                        }

                                                        int tmppos;
                                                        //If takes the original gap
                                                        if(PLVehicle.ID == Vehicle1.LeadVehID)
                                                        {
                                                                OriginalGapVehCount ++;
```

```
        fprintf(fp_originalgap,"%16d\t%16d\t%16d\t%16.2f\n",Vehicle1.ID,PLVehicle.ID,PFVehicle.ID,Vehi
cle1.MergeTime);
                                                        }
                                                        else
                                                        {

        tmppos=FindVehPos(Vehicle1.LeadVehID);
                                                                //If takes following gap
                                                                if(tmppos<0)
                                                                {
                                                                }
                                                                else if(tmppos>PLVehicle.Position)
                                                                {
                                                                        FGapVehCount ++;

        fprintf(fp_followgap,"%16d\t%16d\t%16d\t%16.2f\n",Vehicle1.ID,PLVehicle.ID,PFVehicle.ID,Vehic
le1.MergeTime);
                                                                }
                                                                else // takes the previous gap
                                                                {
                                                                        PGapVehCount ++;

        fprintf(fp_pregap,"%16d\t%16d\t%16d\t%16.2f\n",Vehicle1.ID,PLVehicle.ID,PFVehicle.ID,Vehicle1
.MergeTime);
                                                                }
                                                        }
                                                        //save data

        fprintf(fp_mergedata,"%5.1f\t%6d\t%8.2f\t%8.2f\t%7.2f\t%7.2f\t%18.2f\t%18.2f\t%12.2f\t%15.2f\t%
14.2f\n",

        Vehicle1.MergeTime,Vehicle1.ID,cur_lead/Vehicle1.Speed,cur_lead,cur_lag/PFVehicle.Speed,cur_la
g,PLVehicle.Speed-Vehicle1.Speed,Vehicle1.Speed-PFVehicle.Speed,Vehicle1.Position,s_lead,s_lag);


        fprintf(fp_mergedata_jj,"%5.1f\t%6d\t%8.2f\t%8.2f\t%7.2f\t%7.2f\t%18.2f\t%19.2f\t%12.2f\t%15.2f\t
%14.2f\t%4.3f\t%12.2f\t%4.2f\t%3.2f\t%4.2f\t%4.2f\t%10.3f\t%10.3f\t%15.3f\t%15.3f\t%16.3f\t%16.3f\t%25.3
f\t%10.2f\t%10.2f\t%20.3f\t%12.2f\t%15.2f\t%12d\t%6.3f\n",

        Vehicle1.MergeTime,Vehicle1.ID,cur_lead,cur_lag,PLVehicle.Speed,PFVehicle.Speed,Vehicle1.Spee
d,Vehicle1.NextSpeed,Vehicle1.Position,s_lead,s_lag,Vehicle1.C,g_reactime_acc,g_pl_b,g_c_b,g_pf_b,g_l_b,

        ACPlus,ACMinus,AC_Max_Plus,AC_Max_Minus,PLHeadway,PFHeadway,AccHeadway,PLVehicle.
Position,PFVehicle.Position,Vehicle1.Acceleration,PLVehicle.Length,Vehicle1.Length,Vehicle1.DriverType,B
C);

                                                                //merge vehicle is added to the nearside motorway
                                                                Vehicle1.pregip.FrontSpeed=FrontVehicle.Speed;

        Vehicle1.pregip.FrontPosition=FrontVehicle.Position;
                                                                Vehicle1.pregip.Front2Speed=Front2Vehicle.Speed;

        Vehicle1.pregip.Front2Position=Front2Vehicle.Position;
                                                                Vehicle1.pregip.Speed=Vehicle1.Speed;
                                                                Vehicle1.pregip.Position=Vehicle1.Position;
                                                                Vehicle1.pregip.Acceleration=Vehicle1.Acceleration;
                                                                Vehicle1.pregip.FrontLength=FrontVehicle.Length;

                                                                Vehicle1.preveh.FrontSpeed=FrontVehicle.Speed;

        Vehicle1.preveh.FrontPosition=FrontVehicle.Position;
                                                                Vehicle1.preveh.Front2Speed=Front2Vehicle.Speed;

        Vehicle1.preveh.Front2Position=Front2Vehicle.Position;
                                                                Vehicle1.preveh.Speed=Vehicle1.Speed;
                                                                Vehicle1.preveh.Position=Vehicle1.Position;
```

```
                                                    Vehicle1.preveh.Length=Vehicle1.Length;
        Vehicle1.preveh.Acceleration=Vehicle1.Acceleration;
                                                if(Vehicle1.CarCloseFollowing == 0)
                                                {
                                                        tempt=Vehicle1.CarT/10.0;
                                                        if(Vehicle1.Congested)
                                                                an=ran0.getNorDis(g_accmean_alert,
g_accdevi_alert);
                                                        else
                                                                an=ran0.getNorDis(g_accmean_surprised,
g_accdevi_surprised);
                                                        float ttt_ppp1;
                                                        bn=(-2.0)*an;
                                                        b_severebra=__min(-3.0, (bn-3.0)/2);
                                                        ttt_ppp1=tempt;
                                                        sn=Vehicle1.pregip.FrontLength;
        vn=ran0.getNorDis(Vehicle1.Type.VehicleStatic_SpeedDesiredMean,Vehicle1.Type.VehicleStatic_S
peedDesiredDevi);
                                                        float tmp_speed1,tmp_speed2;
                                                        if(PLVehicle.ID == 0)
                {
        Vehicle1.NextSpeed=__min(Vehicle1.pregip.Speed+2.5*an*ttt_ppp1*(1-
Vehicle1.pregip.Speed/vn)*sqrt(0.025+Vehicle1.pregip.Speed/vn),
        __max(bn*ttt_ppp1+sqrt(         bn*bn*ttt_ppp1*ttt_ppp1-bn*(2*(Vehicle1.pregip.FrontPosition-sn-
Vehicle1.pregip.Position)-Vehicle1.pregip.Speed*ttt_ppp1-
        Vehicle1.pregip.FrontSpeed*Vehicle1.pregip.FrontSpeed/b_severebra)),Vehicle1.pregip.Speed+b_sev
erebra*ttt_ppp1));
                                                }
                                                else
                                                {
        if(Vehicle1.pregip.FrontPosition>=g_environment.Motorway_End.x                          &&
Vehicle1.pregip.FrontPosition<=g_environment.Motorway_End.x+g_environment.Motorway_Buffer_Len)
                                                        {
        Vehicle1.NextSpeed=__min(Vehicle1.pregip.Speed+2.5*an*ttt_ppp1*(1-
Vehicle1.pregip.Speed/vn)*sqrt(0.025+Vehicle1.pregip.Speed/vn),
        __max(bn*ttt_ppp1+sqrt(         bn*bn*ttt_ppp1*ttt_ppp1-bn*(2*(Vehicle1.pregip.FrontPosition-sn-
Vehicle1.pregip.Position)-Vehicle1.pregip.Speed*ttt_ppp1-
        Vehicle1.pregip.FrontSpeed*Vehicle1.pregip.FrontSpeed/b_severebra)),Vehicle1.pregip.Speed+b_sev
erebra*ttt_ppp1));
                                                        }
                                                        else
                                                        {
        tmp_speed1=Vehicle1.pregip.Speed+2.5*an*ttt_ppp1*(1-
Vehicle1.pregip.Speed/vn)*sqrt(0.025+Vehicle1.pregip.Speed/vn);
        Vehicle1.NextSpeed=Vehicle1.pregip.Speed+an*tempt;
        Vehicle1.NextSpeed=__min(tmp_speed1,Vehicle1.NextSpeed);
                                                        }
                                                }
```

```
                                        float dV;
                                        dV=Vehicle1.NextSpeed-Vehicle1.Speed;
                                        Vehicle1.dSpeed=dV/(Vehicle1.CarT/2);
                                }

vehicle[i].VehicleList.SetAt(tmpPrePosIndex,Vehicle1);

Front2Vehicle=vehicle[i].CopyVehicle(FrontVehicle);
                                                FrontVehicle=vehicle[i].CopyVehicle(Vehicle1);

                                                int tmp_InsertIndex=0;

InsertPosIndex=vehicle[10].VehicleList.GetHeadPosition();
                                                if(vehicle[10].VehicleList.GetCount()<=0)
                                                {

vehicle[10].VehicleList.AddTail(Vehicle1);
                                                }
                                                else if(PLVehicle.ID == 0)
                                        {
                                                        vehicle[10].Insert(Vehicle1,
InsertPosIndex);
                                                }
                                                else
                                                {
                                                        while(InsertPosIndex!=NULL)
                                                        {

tmpVehicle=vehicle[10].VehicleList.GetNext(InsertPosIndex);
                                                                if(tmpVehicle.ID        ==
PLVehicle.ID)
                                                                {
                                                                        int jh;

jh=vehicle[10].VehicleList.GetCount();

vehicle[10].Insert(Vehicle1, InsertPosIndex);
                                                                        break;
                                                                }
                                                                tmp_InsertIndex ++;
                                                }
                                }

                                vehicle[10].CopyVehicleList();

                                POSITION tp;
                                VehicleStru tv;
                                tp=vehicle[10].VehicleList.GetHeadPosition();
                                while(tp != NULL)
                                {
                                        tv=vehicle[10].VehicleList.GetNext(tp);
                                }

                                // remove from acceleration laen
                                vehicle[i].DeleteVehicle(tmpPrePosIndex);

                                CanMerge=FALSE;
                                continue;
                        }
                        else
                        {
                                //update

vehicle[i].VehicleList.SetAt(tmpPrePosIndex,Vehicle1);
if(Vehicle1.Position>=g_environment.AccLane_End.x+g_environment.AccLane_Buffer_Len)
```

```
                                                {
                                                        StopedVehicleCount ++;
        vehicle[i].DeleteVehicle(tmpPrePosIndex);;
                                                }
*/                                              }

                                                Vehicle1.pregip.Position=Vehicle1.Position;
                                                Vehicle1.pregip.Speed=Vehicle1.Speed;
                                                Vehicle1.pregip.Acceleration=Vehicle1.Acceleration;
                                                Vehicle1.pregip.FrontPosition=FrontVehicle.Position;
                                                Vehicle1.pregip.FrontSpeed=FrontVehicle.Speed;
                                                Vehicle1.pregip.Front2Position=Front2Vehicle.Position;
                                                Vehicle1.pregip.Front2Speed=Front2Vehicle.Speed;
                                                Vehicle1.pregip.FrontLength=FrontVehicle.Length;

/*                                              //assign the value of front vehicle
                                                Front2Vehicle=vehicle[i].CopyVehicle(FrontVehicle);
                                                FrontVehicle=vehicle[i].CopyVehicle(Vehicle1);
*/
                                                tmp_count ++;
                                        }
                                }
                                //if still on the ramp road, using car-following model to calculate
                                else
                                {
                                        if(Vehicle1.CarCloseFollowing == 0)
                                        {
    Vehicle1.Position=Vehicle1.preveh.Position+computet*Vehicle1.preveh.Speed;
                                        }
                                        else if(Vehicle1.CarCloseFollowing == 1) //A1
                                        {
        Vehicle1.Position=Vehicle1.preveh.Position+computet*Vehicle1.preveh.Speed+.5*g_a1_closefollowi
ng*computet*computet;
                                        }
                                        else
                        //D1
                                        {
        Vehicle1.Position=Vehicle1.preveh.Position+computet*Vehicle1.preveh.Speed+.5*g_d1_closefollowi
ng*computet*computet;
                                                if(Vehicle1.Position<=Vehicle1.preveh.Position)
                                                        Vehicle1.Position=Vehicle1.preveh.Position+1;
                                        }

                                        if(k>0)
                                        {
                                                if(FrontVehicle.Position-Vehicle1.Position-
FrontVehicle.Length<= 0)
                                                {
                                                        Vehicle1.Position=FrontVehicle.Position-
FrontVehicle.Length-1;
                                                }

                                                Vehicle1.Gap=FrontVehicle.Position-Vehicle1.Position-
Vehicle1.Length;
                                                Vehicle1.Headway=FrontVehicle.Position-Vehicle1.Position;
                                        }
                                        else
                                        {

        Vehicle1.Gap=g_environment.AccLane_End.x+g_environment.AccLane_Buffer_Len-
Vehicle1.Position;

        Vehicle1.Headway=g_environment.AccLane_End.x+g_environment.AccLane_Buffer_Len-
```

```
Vehicle1.Position;
                                                }


                                                Vehicle1.CarCurT=Vehicle1.CarCurT+computet*10;
                                                if(Vehicle1.CarCurT<Vehicle1.CarT)
                                                {
                                                        if(Vehicle1.CarCloseFollowing == 0)
                                                        {
                                                                Vehicle1.Speed=Vehicle1.preveh.Speed;
                                                                Vehicle1.Acceleration=Vehicle1.pregip.Acceleration;
                                                        }
                                                        else if(Vehicle1.CarCloseFollowing == 1) //A1
                                                        {

Vehicle1.Speed=Vehicle1.preveh.Speed+g_a1_closefollowing*computet;
                                                                Vehicle1.Acceleration=g_a1_closefollowing;
                                                        }
                                                        else
                                //D1
                                                        {

Vehicle1.Speed=Vehicle1.preveh.Speed+g_d1_closefollowing*computet;
                                                                Vehicle1.Acceleration=g_d1_closefollowing;
                                                        }
                                                }
                                                // Start for next reaction time
                                                else
                                                {
                                                        if(Vehicle1.Position>=g_environment.AccLane_Start.x)
                                                        {

GetPLPFVehicle(Vehicle1.Position,PLVehicle,PFVehicle);
                                                                Vehicle1.LeadVehID=PLVehicle.ID;
                                                }
                                                tempt=Vehicle1.CarT/10.0;

                                                if(Vehicle1.CarCloseFollowing == 0)
                                                {
                                                        if(Vehicle1.Congested)
                                                                an=ran0.getNorDis(g_accmean_alert,
g_accdevi_alert);
                                                        else
                                                                an=ran0.getNorDis(g_accmean_surprised,
g_accdevi_surprised);
                                                        Vehicle1.Acceleration=an;

                                                        if(k != 0)
                                                        {
                                                                bn=(-2.0)*an;
                                                                sn=Vehicle1.pregip.FrontLength;


        //
                vn=ran0.getNorDis(g_vn_mean,g_vn_devi);

                vn=ran0.getNorDis(Vehicle1.Type.VehicleStatic_SpeedDesiredMean,Vehicle1.Type.VehicleStatic_S
peedDesiredDevi);
                                                                b_severebra=__min(-3.0, (bn-3.0)/2);


                Vehicle1.Speed=__min(Vehicle1.pregip.Speed+2.5*an*tempt*(1-
Vehicle1.pregip.Speed/vn)*sqrt(0.025+Vehicle1.pregip.Speed/vn),

                bn*tempt+sqrt(                          bn*bn*tempt*tempt-bn*(2*(Vehicle1.pregip.FrontPosition-sn-
Vehicle1.pregip.Position)-Vehicle1.pregip.Speed*tempt-

                Vehicle1.pregip.FrontSpeed*Vehicle1.pregip.FrontSpeed/b_severebra)));
                                                                }
```

```
                                                    else
                                                    {
Vehicle1.Speed=Vehicle1.pregip.Speed+an*tempt;
                                                    }
                                            }
                                            else if(Vehicle1.CarCloseFollowing == 1) //A1
                                            {
Vehicle1.Speed=Vehicle1.pregip.Speed+g_a1_closefollowing*tempt;
                                                    Vehicle1.Acceleration=g_a1_closefollowing;
                                            }
                                            else
                        //D1
                                            {
Vehicle1.Speed=Vehicle1.pregip.Speed+g_d1_closefollowing*tempt;
                                                    Vehicle1.Acceleration=g_d1_closefollowing;
                                            }

                                            Vehicle1.CarCurT=0;
                                    }

                                    //
                                    if(Vehicle1.CarCurT == 0)
                                    {
                                            if(k == 0)
                                            {
                                                    FrontVehicle.Speed=Vehicle1.Speed;

FrontVehicle.Position=g_environment.AccLane_Buffer_Len+g_environment.AccLane_End.x;
                                                    Front2Vehicle.Speed=Vehicle1.Speed;
        Front2Vehicle.Position=g_environment.AccLane_Buffer_Len+g_environment.AccLane_End.x;
                                            }
                                            if(Vehicle1.Position>=g_environment.AccLane_Start.x)
                                            {
                                                    Vehicle1.CarT=g_reactime_acc*10;
                                                    Vehicle1.Congested=0;
                                                    Vehicle1.CarCloseFollowing=0;
                                                    Vehicle1.ReactionTime=g_reactime_acc;
                                            }
                                            else
                                            {
                                                    if(Vehicle1.Speed<g_critical_speed)      //Congested
Traffic
                                                    {
                                                            Vehicle1.CarT=g_reactime_alert*10;
                                                            Vehicle1.Congested=1;
                                                            Vehicle1.CarCloseFollowing=0;
                                                            Vehicle1.ReactionTime=g_reactime_alert;
                                                    }
                                                    else //Non Congested Traffic
                                                    {
                                                            if(k == 0)
                                                            {

if(Vehicle1.Speed<g_critical_speed) //congested Traffic
                                                                    {

Vehicle1.CarT=g_reactime_alert*10;
                                                                            Vehicle1.Congested=1;

Vehicle1.ReactionTime=g_reactime_alert;
                                                                    }
                                                                    else
                                                                    {
```

```
                Vehicle1.CarT=g_reactime_surprised*10;

                Vehicle1.ReactionTime=g_reactime_surprised;

//headway

                headwaygap1=(FrontVehicle.Position-Vehicle1.Position)/Vehicle1.Speed;

                headwaygap2=(Front2Vehicle.Position-FrontVehicle.Position)/FrontVehicle.Speed;

                headwaygap1=(FrontVehicle.Position-Vehicle1.Position-FrontVehicle.Length)/Vehicle1.Speed;

                headwaygap2=(Front2Vehicle.Position-FrontVehicle.Position-
Front2Vehicle.Length)/FrontVehicle.Speed;

                tempacc2=(Front2Vehicle.Speed-Vehicle1.pregip.Front2Speed)/PreVehicle1.CarT;

                if(headwaygap1<g_tc_closefollowing&&  tempacc1>g_ac_closefollowing  &&
(headwaygap2>g_tc_closefollowing || tempacc2>g_ac_closefollowing))  //Meeting  Close-following
Requirement

                birand=ran0.bnldev(g_percentage_closefollowing/100.0,1,&tmprand);

                tempDV=FrontVehicle.Speed-Vehicle1.Speed;

                tempDX=FrontVehicle.Position-Vehicle1.Position;

                tempAX=g_ax_closefollowing+Vehicle1.Length;

                tempBX=g_bx_closefollowing;

                tempEX=g_ex_closefollowing;

                tempABX=tempAX+tempBX*sqrt(Vehicle1.Speed);

                tempSDX=tempAX+tempBX*(sqrt(Vehicle1.Speed*tempEX));

                if(tempDV>g_opdv_closefollowing  ||  tempDV<g_cldv_closefollowing  ||  tempDX>tempSDX  ||
tempDX<tempABX)

                Vehicle1.CarT=g_reactime_alert*10;

                Vehicle1.Congested=1;

                Vehicle1.CarCloseFollowing=0;
                                                                Vehicle1.Congested=0;
                                                        }
                                                        Vehicle1.CarCloseFollowing=0;
                                                }
                                                else
                                                {
                                                        if(g_gap_closefollowing  ==  1)
                                                        {
                                                        }
                                                        else  //Gap
                                                        {
                                                        }
                                                        tempacc1=(FrontVehicle.Speed-
Vehicle1.pregip.FrontSpeed)/PreVehicle1.CarT;
                                                {
                                                        long tmprand;
                                                        float birand;
                                                        tmprand=rand();
                                                        if(birand)
                                                        {
                                                                {
```

```
Vehicle1.ReactionTime=g_reactime_alert;
                                                                    }
                                                                    else
                                                                    {

//D1

if(tempDX<(tempSDX+tempABX)/2)
                                                                        {

Vehicle1.CarCloseFollowing=2;
                                                                        }

//A1

else
                                                                        {

Vehicle1.CarCloseFollowing=1;
                                                                        }

Vehicle1.CarT=g_reactime_closefollowing*10;

Vehicle1.Congested=1;

Vehicle1.ReactionTime=g_reactime_closefollowing;
                                                                    }
                                                                }
                                                                else
                                                                {

Vehicle1.CarT=g_reactime_surprised*10;

Vehicle1.Congested=0;

Vehicle1.CarCloseFollowing=0;

Vehicle1.ReactionTime=g_reactime_surprised;
                                                                    }
                                                                }
                                                                else
                                                                {

Vehicle1.CarT=g_reactime_surprised*10;
                                                                        Vehicle1.Congested=1;

Vehicle1.CarCloseFollowing=0;

Vehicle1.ReactionTime=g_reactime_surprised;
                                                                    }
                                                                }
                                                            }
                                                        }

                                                        Vehicle1.pregip.FrontSpeed=FrontVehicle.Speed;
                                                        Vehicle1.pregip.FrontPosition=FrontVehicle.Position;
                                                        Vehicle1.pregip.Front2Speed=Front2Vehicle.Speed;
                                                        Vehicle1.pregip.Front2Position=Front2Vehicle.Position;
                                                        Vehicle1.pregip.Speed=Vehicle1.Speed;
                                                        Vehicle1.pregip.Position=Vehicle1.Position;
                                                        Vehicle1.pregip.Acceleration=Vehicle1.Acceleration;
                                                        Vehicle1.pregip.FrontLength=FrontVehicle.Length;
                                                    }
                                                }
```

```
                vehicle[i].VehicleList.SetAt(tmpPrePosIndex,Vehicle1);

                Front2Vehicle=vehicle[i].CopyVehicle(FrontVehicle);
                FrontVehicle=vehicle[i].CopyVehicle(Vehicle1);

                k++;
};

PosIndex=vehicle[i].VehicleList.GetHeadPosition();
//Update all vehcles' accelerations at pervious time
while(PosIndex!=NULL)
{
                tmpPrePosIndex=PosIndex;
                Vehicle1=vehicle[i].VehicleList.GetNext(PosIndex);
//                Vehicle1.preveh.Acceleration=(Vehicle1.Speed-Vehicle1.preveh.Speed)/computet;

                Vehicle1.preveh.Speed=Vehicle1.Speed;
                Vehicle1.preveh.Position=Vehicle1.Position;
                Vehicle1.preveh.Length=Vehicle1.Length;
                Vehicle1.preveh.Acceleration=Vehicle1.pregip.Acceleration;
                Vehicle1.preveh.FrontSpeed=FrontVehicle.Speed;
                Vehicle1.preveh.FrontPosition=FrontVehicle.Position;
                Vehicle1.preveh.Front2Speed=Front2Vehicle.Speed;
                Vehicle1.preveh.Front2Position=Front2Vehicle.Position;

                vehicle[i].VehicleList.SetAt(tmpPrePosIndex,Vehicle1);
}

if(cur_time-1>=g_warmup_period/computet)
{
                VehicleStru tmppreveh;
                POSITION tmpprepos;
                fprintf(fp_individual[i],"%4.1f\t%6d\t",(cur_time-
1)*computet,vehicle[i].PreVehicleList.GetCount());

                tmpprepos=vehicle[i].PreVehicleList.GetHeadPosition();
                while(tmpprepos!=NULL)
                {
                        tmppreveh=vehicle[i].PreVehicleList.GetNext(tmpprepos);


        fprintf(fp_individual[i],"%5d\t%7.2f\t%8.2f\t%6.3f\t",tmppreveh.ID,tmppreveh.Position,tmppreveh.S
peed,tmppreveh.Acceleration);
                }
                fprintf(fp_individual[i],"\n");


        /*              tmpprepos=vehicle[i].PreVehicleList.GetHeadPosition();
                while(tmpprepos!=NULL)
                {
                        tmppreveh=vehicle[i].PreVehicleList.GetNext(tmpprepos);
                dbcancel(dbproc);
                if(i==0)
                        dbcmd(dbproc,                                                   "INSERT
Ramp_Lane_1(time,vehicle_id,vehicle_speed,vehicle_acceleration,vehicle_position)");
                else if(i==1)
                        dbcmd(dbproc,                                                   "INSERT
Ramp_Lane_2(time,vehicle_id,vehicle_speed,vehicle_acceleration,vehicle_position)");
                dbfcmd(dbproc,"         VALUES(%d,%d,%f,%f,%f)",(int)((cur_time-
1)*computet*10+.5),tmppreveh.ID,tmppreveh.Speed,tmppreveh.Acceleration,tmppreveh.Position);
                if(dbsqlexec(dbproc)==FALSE)
                {
                        AfxMessageBox("error first insert: INSERT Ramp_Lane");
                        return;
                }
```

```
                    }
*/                  }

                    //To add vehicle
                    int tt;
                    tt=(int)(g_lane_ramp[i].AddVehicleTime/computet+.5);
                    if(cur_time == tt && !IsZeroFlowTime)
                    {
                            if(vehicle[i].VehicleList.GetCount()>0)
                                    LastVehicle=vehicle[i].VehicleList.GetTail();
                            else
                            {

        LastVehicle.Position=g_environment.AccLane_End.x+g_environment.AccLane_Buffer_Len;
                    LastVehicle.Speed=ran0.getNorDis(g_lane_ramp[i].Lane_Speed_Mean,g_lane_ramp[i].Lane_Speed_
Devi);
                                    if(LastVehicle.Speed>g_environment.Onramp_SpeedLimit)
                                            LastVehicle.Speed=g_environment.Onramp_SpeedLimit;
                            }
                            //HGV
                            if(g_lane_ramp[i].AddedVehicleCount+1 ==
g_lane_ramp[i].CurStepAddHGVCount)
                            {
                                    AddVehicle.Type.VehicleType_ID=g_vehicle_type[1].VehicleType_ID;

                    strcpy(AddVehicle.Type.VehicleType_Name,g_vehicle_type[1].VehicleType_Name);

                    AddVehicle.Type.VehicleStatic_SpeedLimit=g_vehicle_type[1].VehicleStatic_SpeedLimit;

                    AddVehicle.Type.VehicleStatic_SpeedDesiredMean=g_vehicle_type[1].VehicleStatic_SpeedDesired
Mean;

                    AddVehicle.Type.VehicleStatic_SpeedDesiredDevi=g_vehicle_type[1].VehicleStatic_SpeedDesiredD
evi;

                        AddVehicle.Type.VehicleType_Color=g_vehicle_type[1].VehicleType_Color;
                    AddVehicle.Type.VehicleType_Length_Mean=g_vehicle_type[1].VehicleType_Length_Mean;

                        AddVehicle.Type.VehicleType_Length_Devi=g_vehicle_type[1].VehicleType_Length_Devi;
                                            cur_vehicle_is_HGV=true;
                                            g_lane_ramp[i].AddedHGVCount ++;

                    g_lane_ramp[i].CurStepAddHGVCount=GetAddHGVCount(g_lane_ramp[i].CurAddHGVBase,
g_lane_ramp[i].AddHGVStep);
                            }
                            //Passenger car
                            else
                            {
                                    AddVehicle.Type.VehicleType_ID=g_vehicle_type[0].VehicleType_ID;

                    strcpy(AddVehicle.Type.VehicleType_Name,g_vehicle_type[0].VehicleType_Name);
                        AddVehicle.Type.VehicleStatic_SpeedLimit=g_vehicle_type[0].VehicleStatic_SpeedLimit;

                    AddVehicle.Type.VehicleStatic_SpeedDesiredMean=g_vehicle_type[0].VehicleStatic_SpeedDesired
Mean;

                    AddVehicle.Type.VehicleStatic_SpeedDesiredDevi=g_vehicle_type[0].VehicleStatic_SpeedDesiredD
evi;

                        AddVehicle.Type.VehicleType_Color=g_vehicle_type[0].VehicleType_Color;
                    AddVehicle.Type.VehicleType_Length_Mean=g_vehicle_type[0].VehicleType_Length_Mean;

                        AddVehicle.Type.VehicleType_Length_Devi=g_vehicle_type[0].VehicleType_Length_Devi;
                                            cur_vehicle_is_HGV=false;
                            }
```

```
                                g_lane_ramp[i].AddedVehicleCount ++;
                                Vehicle_ID ++;

                                //initial the added vehicle
                                AddVehicle.ID=Vehicle_ID;
                                AddVehicle.Position=g_environment.AccLane_Start.x-
g_environment.Onramp_Slip_Length;

        AddVehicle.Speed=ran0.getNorDis(g_lane_ramp[i].Lane_Speed_Mean,g_lane_ramp[i].Lane_Speed_
Devi);
    //                          if(AddVehicle.Speed>g_environment.Onramp_SpeedLimit                ||
AddVehicle.Speed>AddVehicle.Type.VehicleStatic_SpeedLimit)
    //
        AddVehicle.Speed=__min(g_environment.Onramp_SpeedLimit,AddVehicle.Type.VehicleStatic_Spee
dLimit);

        AddVehicle.Length=ran0.getNorDis(AddVehicle.Type.VehicleType_Length_Mean,AddVehicle.Type.
VehicleType_Length_Devi);

        AddVehicle.Gap=(float)(g_environment.AccLane_Length+g_environment.Onramp_Slip_Length+g_e
nvironment.AccLane_Buffer_Len-AddVehicle.Length);
                                AddVehicle.CarCurT=0;
                                AddVehicle.CarCloseFollowing=0;
                                AddVehicle.BehID=0;

        AddVehicle.Headway=(float)(g_environment.AccLane_Length+g_environment.Onramp_Slip_Length
+g_environment.AccLane_Buffer_Len);
                                AddVehicle.Acceleration=0;
                                AddVehicle.Courtesy=0;
                                AddVehicle.DriverType=0;
                                AddVehicle.IsFreeSpeed=0;
                                AddVehicle.LaneID=i+1;
                                AddVehicle.IsPFVehicle=0;
                                if(vehicle[i].VehicleList.GetCount()>0)
                                {
                                        AddVehicle.PreID=LastVehicle.ID;
                                        AddVehicle.pregip.FrontSpeed=LastVehicle.Speed;
                                        AddVehicle.pregip.FrontPosition=LastVehicle.Position;
                                        AddVehicle.pregip.Front2Speed=Front2Vehicle.Speed;
                                        AddVehicle.pregip.Front2Position=Front2Vehicle.Position;
                                        AddVehicle.pregip.FrontLength=LastVehicle.Length;
                                        AddVehicle.preveh.FrontSpeed=LastVehicle.Speed;
                                        AddVehicle.preveh.FrontPosition=LastVehicle.Position;
                                        AddVehicle.preveh.Front2Speed=Front2Vehicle.Speed;
                                        AddVehicle.preveh.Front2Position=Front2Vehicle.Position;
                                }
                                else
                                {
                                        AddVehicle.PreID=0;
                                        AddVehicle.pregip.FrontSpeed=LastVehicle.Speed;
                                        AddVehicle.pregip.FrontPosition=LastVehicle.Position;
                                        AddVehicle.pregip.Front2Speed=LastVehicle.Speed;
                                        AddVehicle.pregip.Front2Position=LastVehicle.Position;
                                        AddVehicle.pregip.FrontLength=LastVehicle.Length;
                                        AddVehicle.preveh.FrontSpeed=LastVehicle.Speed;
                                        AddVehicle.preveh.FrontPosition=LastVehicle.Position;
                                        AddVehicle.preveh.Front2Speed=LastVehicle.Speed;
                                        AddVehicle.preveh.Front2Position=LastVehicle.Position;
                                }

                                AddVehicle.yielding=0;
                                AddVehicle.NextPosition=LastVehicle.Position;
                                AddVehicle.NextSpeed=LastVehicle.Speed;

                                AddVehicle.pregip.Speed=AddVehicle.Speed;
                                AddVehicle.pregip.Position=AddVehicle.Position;
```

```
AddVehicle.pregip.Acceleration=AddVehicle.Acceleration;
AddVehicle.pregip.FrontLength=AddVehicle.Length;
AddVehicle.preveh.Speed=AddVehicle.Speed;
AddVehicle.preveh.Position=AddVehicle.Position;
AddVehicle.preveh.Acceleration=AddVehicle.Acceleration;
AddVehicle.MergeTime=0;


AddVehicle.C=GetC(g_cc,AddVehicle.DriverType,AddVehicle.Alpha,AddVehicle.C1,AddVehicle.C
2,AddVehicle.C3,AddVehicle.Alpha_Lead,AddVehicle.Alpha_Lag);
if(AddVehicle.Speed>g_critical_speed)
{
AddVehicle.ReactionTime=g_reactime_surprised;
AddVehicle.Congested=0;
}
else
{
AddVehicle.ReactionTime=g_reactime_alert;
AddVehicle.Congested=1;
}
AddVehicle.CarT=AddVehicle.ReactionTime*10;

POSITION pp;
pp=vehicle[i].VehicleList.GetHeadPosition();
//Add new vehicle into vehicle list
vehicle[i].Add(AddVehicle);
if(AddVehicle.Type.VehicleType_Name[0] == 'C')
{

fprintf(fp_vehiclestatic,"%5d\t%7.2f\t%8s\t%7d\t%6s\n",AddVehicle.ID,AddVehicle.Length,"0",cur_t
ime,"A");
fprintf(fp_newaddramp,"%5.1f\t%6d\t%12d\t%9s\t%3s\n",
cur_time*computet,AddVehicle.ID,AddVehicle.DriverType,"P","Acc");
}
else if(AddVehicle.Type.VehicleType_Name[0] == 'H')
{

fprintf(fp_vehiclestatic,"%5d\t%7.2f\t%8s\t%7d\t%6s\n",AddVehicle.ID,AddVehicle.Length,"1",cur_t
ime,"A");
cur_time*computet,AddVehicle.ID,AddVehicle.DriverType,"HGV","Acc");
fprintf(fp_newaddramp,"%5.1f\t%6d\t%12d\t%9s\t%3s\n",

cur_time*computet,AddVehicle.ID,AddVehicle.DriverType,"HGV","Acc");
}

//get the add vehicle time

g_lane_ramp[i].AddVehicleTime=GetAddVehicleTime(g_lane_ramp[i].AddVehicleTime,g_lane_ram
p[i].Lane_Flow);
}

if(vehicle[i].VehicleList.GetCount()>0)
vehicle[i].CopyVehicleList();
else
{
vehicle[i].PreVehicleList.RemoveAll();
}
}
}
```

Appendix B Motorway Traffic Hysteresis

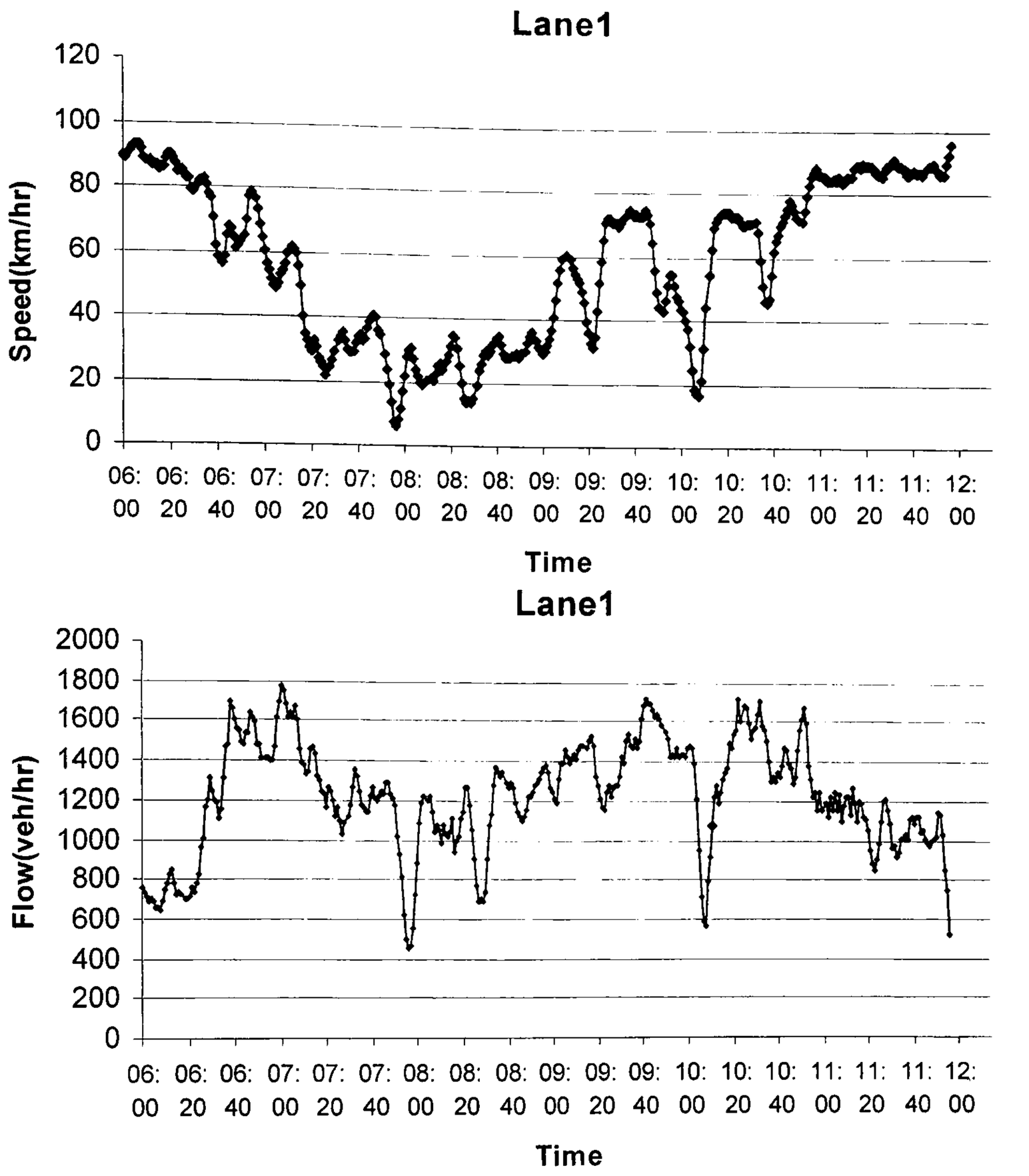


Figure B-1 Traffic hysteresis observed from Lane1, Detector 4806A, collected on 01 February 2001. It shows the speed-time, flow-time diagrams and two hysteresis observed at different morning intervals.

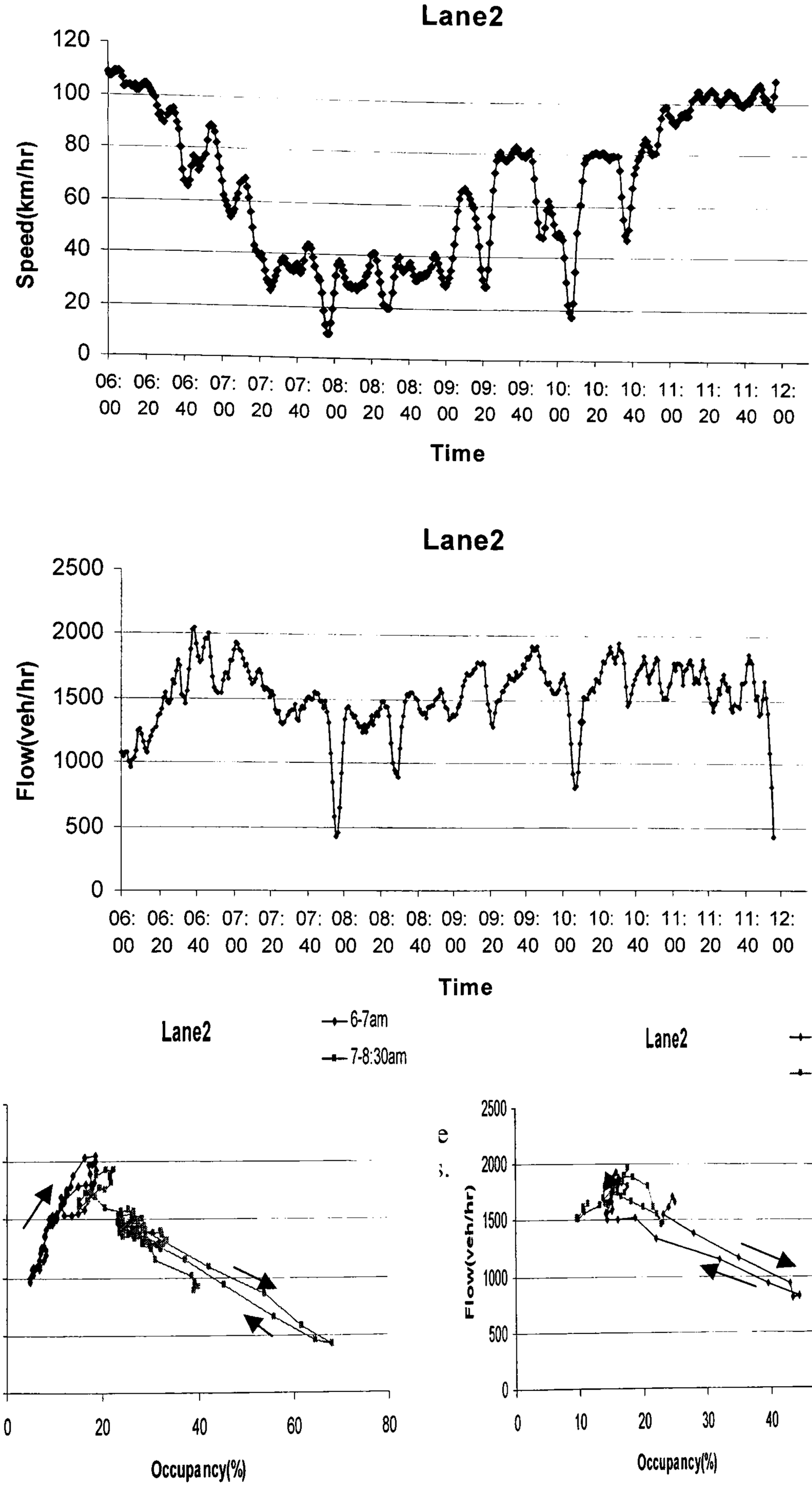


Figure B-2 Traffic hysteresis observed from Lane2, Detector 4806A, collected on 01 February 2001. It shows the speed-time, flow-time diagrams and two hysteresis observed at different morning intervals.

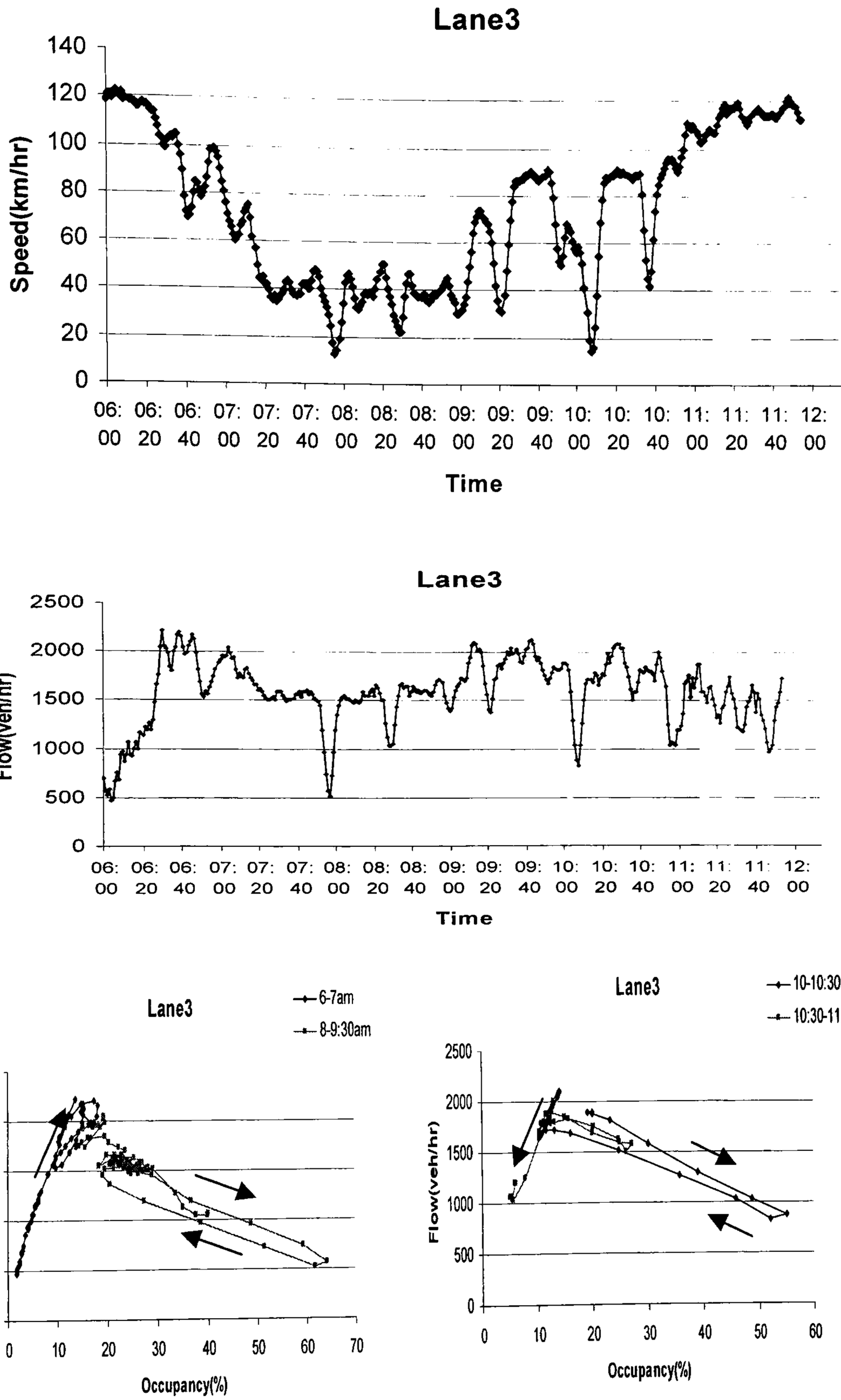


Figure B-3 Traffic hysteresis observed from Lane3, Detector 4806A, collected on 01 February 2001. It shows the speed-time, flow-time diagrams and two hysteresis observed at different morning intervals.

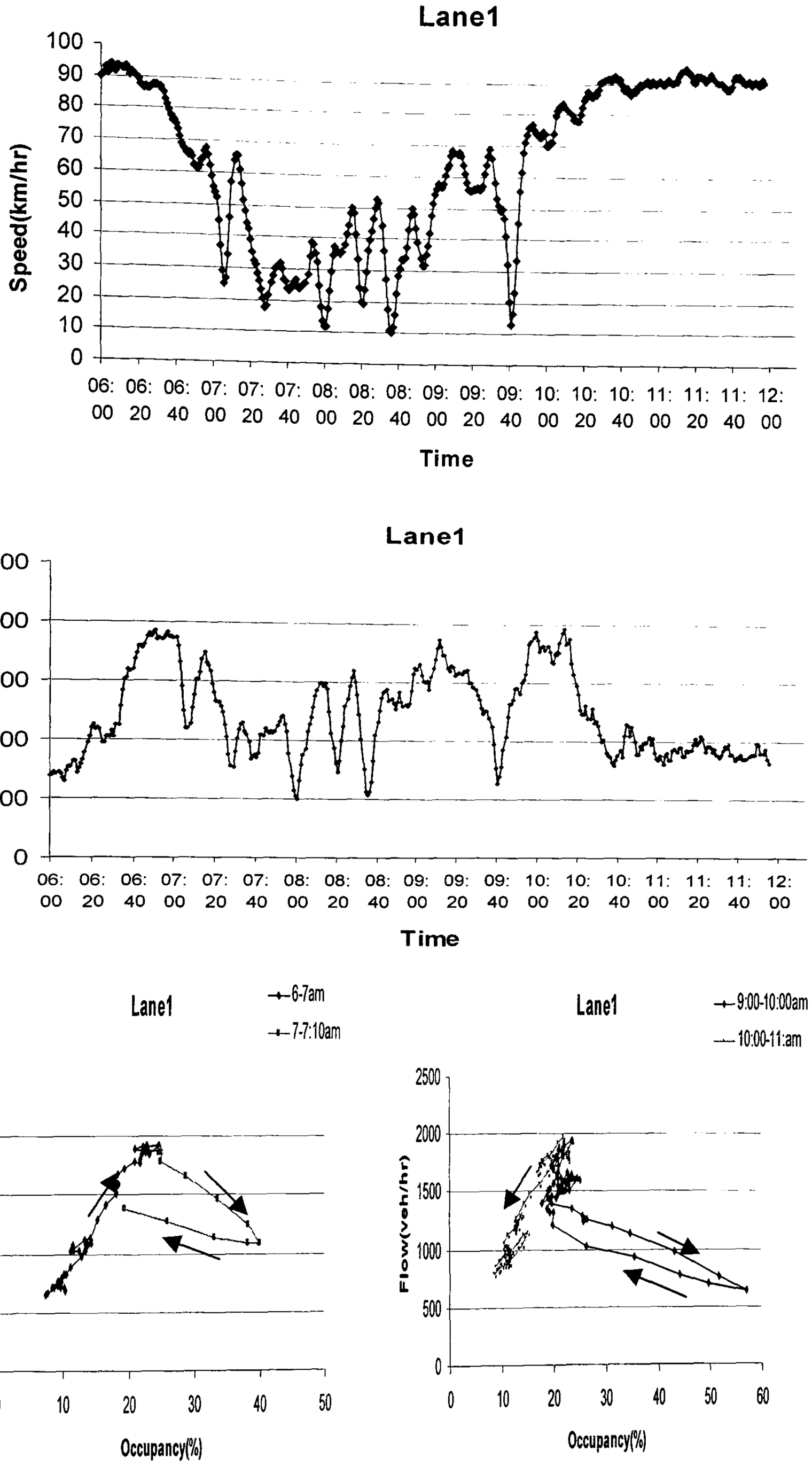


Figure B-4 Traffic hysteresis observed from Lane1, Detector 4806A, collected on 06 February 2001. It shows the speed-time, flow-time diagrams and two hysteresis observed at different morning intervals.

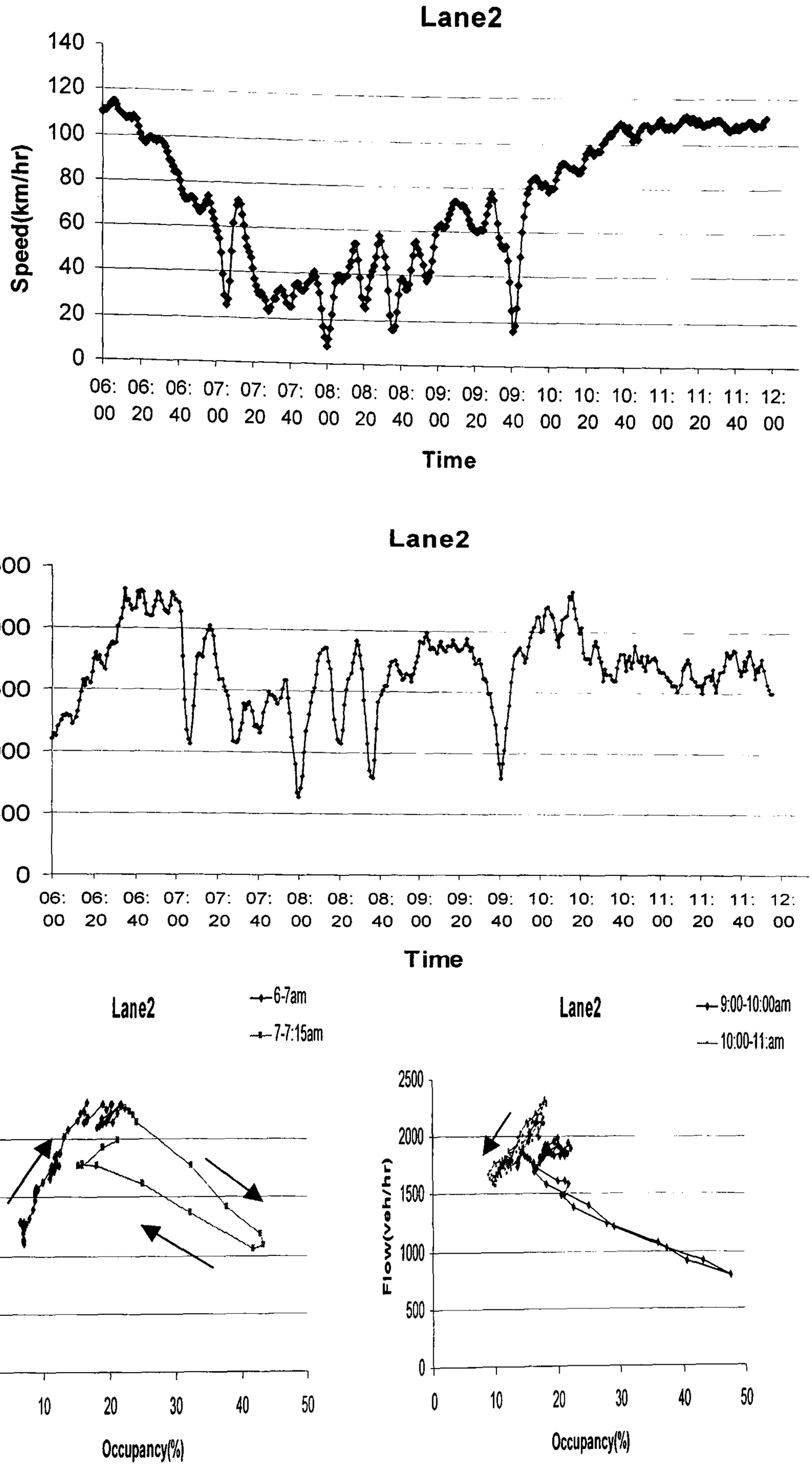


Figure B-5 Traffic hysteresis observed from Lane2, Detector 4806A, collected on 06 February 2001. It shows the speed-time, flow-time diagrams and two hysteresis observed at different morning intervals.

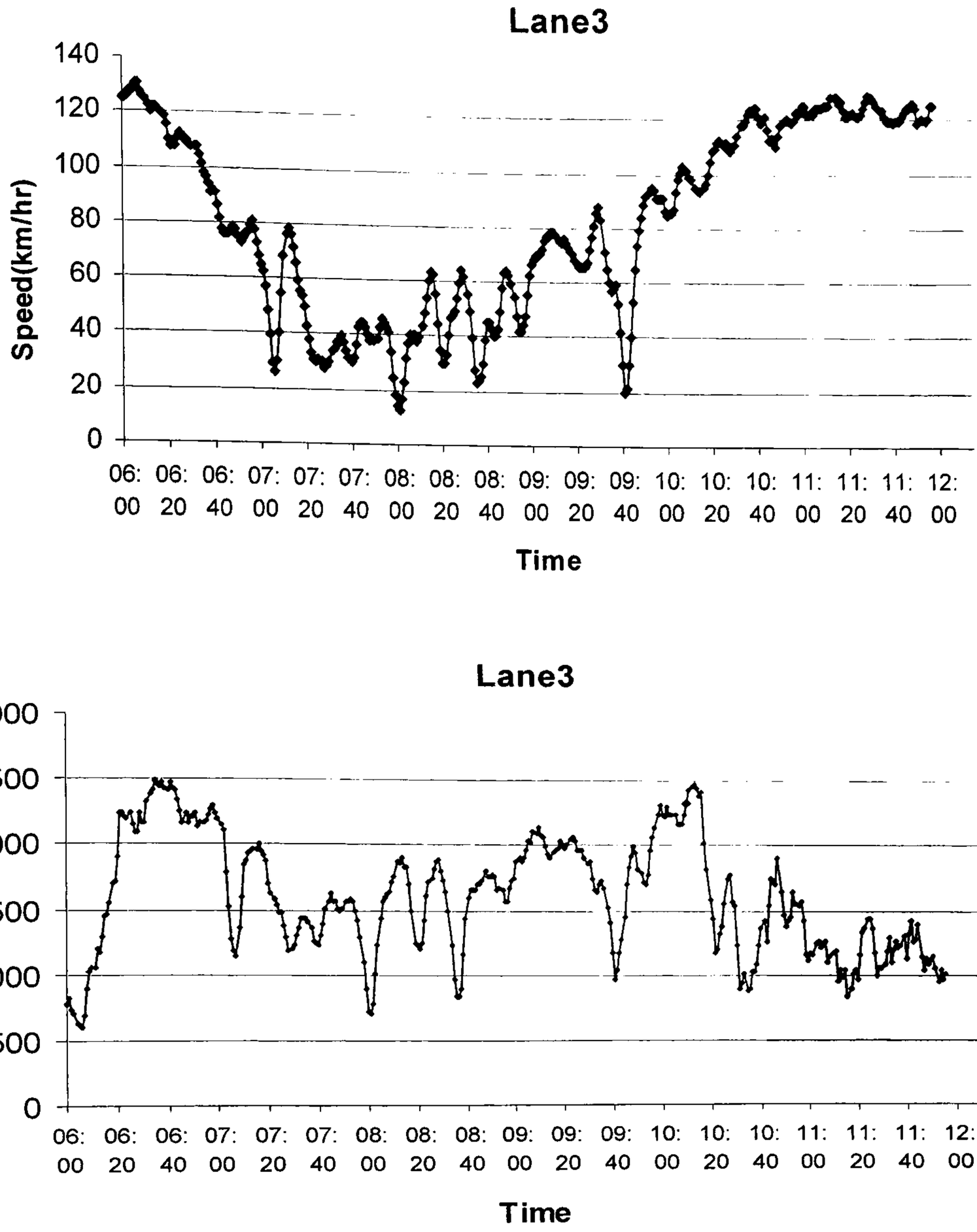
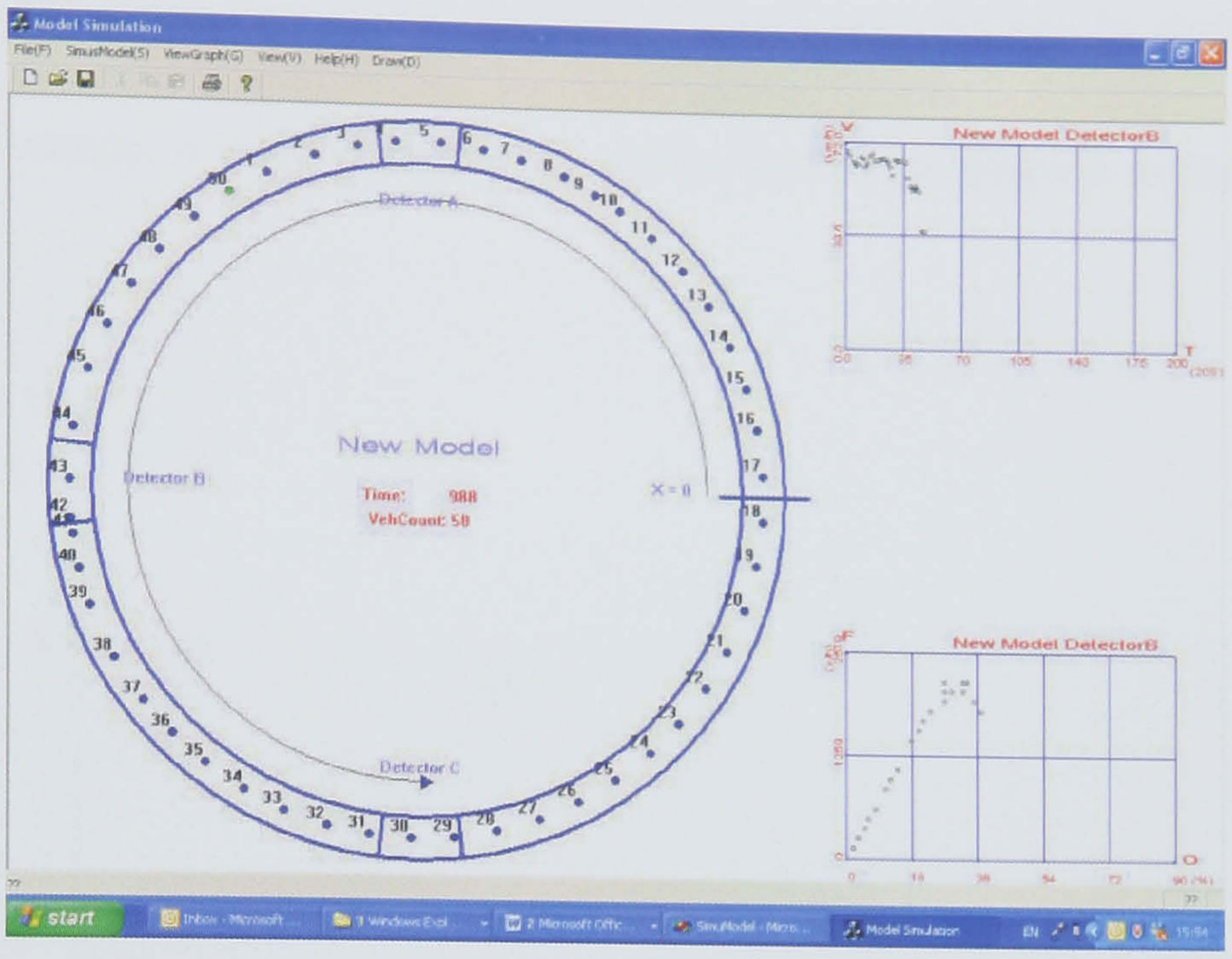
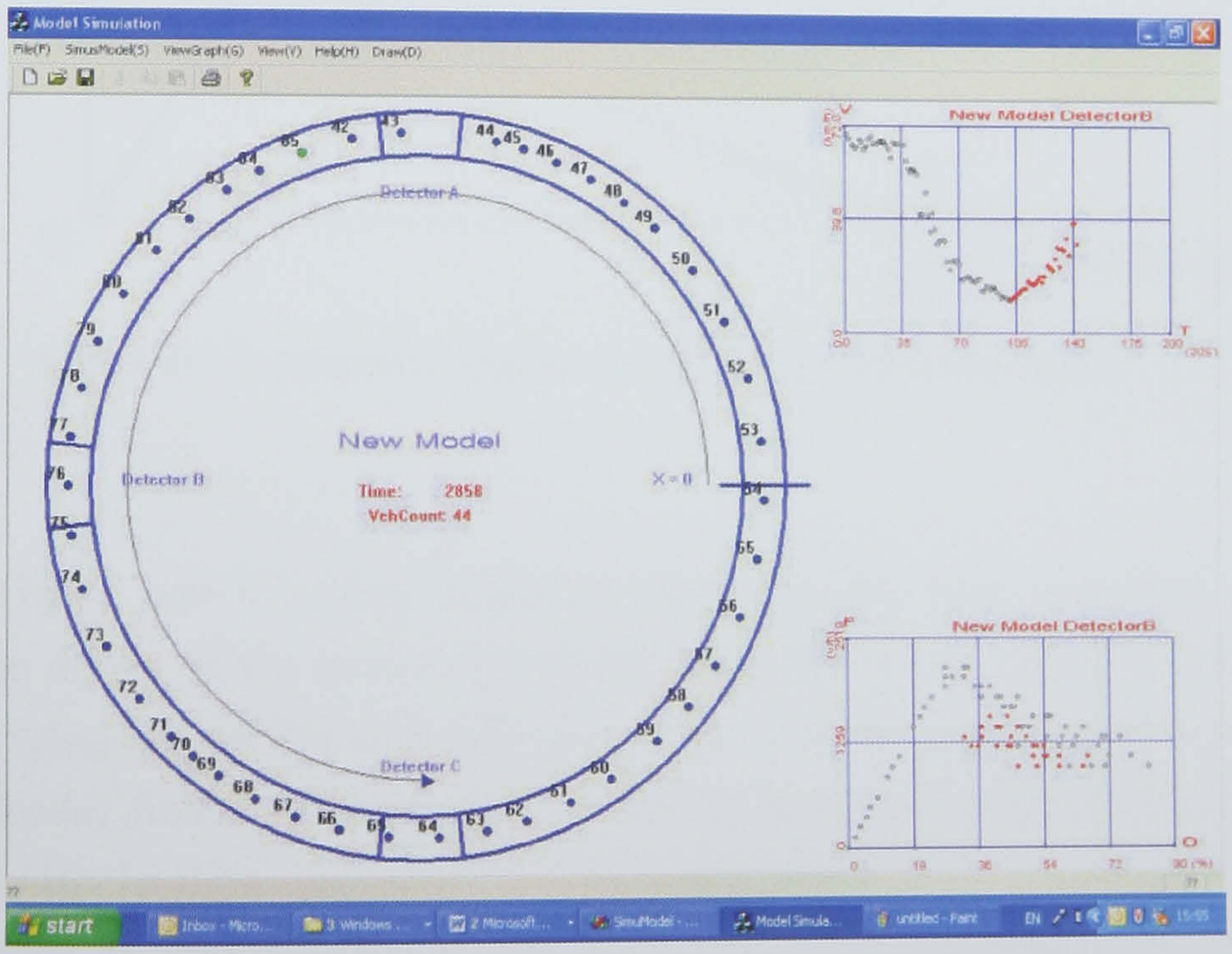


Figure B-6 Traffic hysteresis observed from Lane3, Detector 4806A, collected on 06 February 2001. It shows the speed-time, flow-time diagrams and two hysteresis observed at different morning intervals.

Appendix C Screenshots of Car-following Model Tests on the Ring Road



(a)



(b)

Figure C-1 Screenshots of the new proposed car-following model tested on a ring road. Each of them is composed of three diagrams- the animation of car movements (on the left side); the real-time speed-time plot (top right) and the flow-occupancy plot (bottom right). (a) illustrates the traffic build-up to congestions process; (b) illustrates the traffic recovery process with an apparent traffic hysteresis loop on the flow-occupancy plot.

Appendix D The mathematical derivations of Gipps' model

When the traffic flow is low, the Gipps' free-flow model can be simplified as assuming that vehicles drive at their free-flow speeds. The flow-density relationship is therefore simply a linear function (eq. (D-1), and is represented as line OA in Figure D-1.

$$q = v_f \rho \tag{D-1}$$

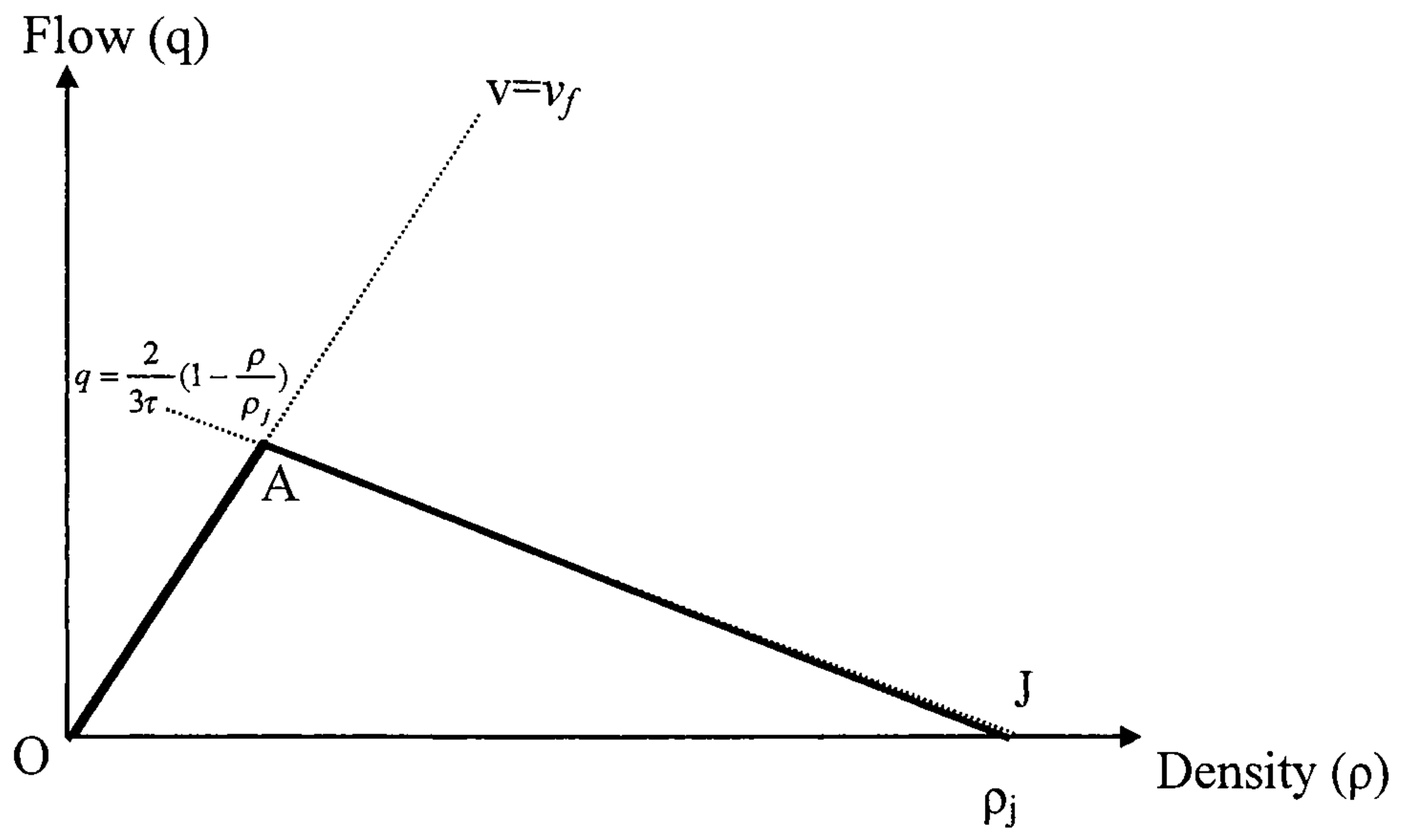


Figure D-1 Fundamental diagram of the Gipps' car-following model.

The Gipps' car-following model is based on the idea of safety distance keeping: the driver of the following vehicle can select a safe speed to ensure that he/she can bring his vehicle to a safe stop, if the vehicle ahead comes to a sudden stop. The model further assumes that all drivers have the same reaction time and acceleration/deceleration and they maintain the same behaviour throughout the whole process of traffic build-up and recovery. Gipps introduced an additional reaction delay as $\tau/2$ and demonstrated that this will enable the driver to avoid collision indefinitely. That is, the following inequality would always be satisfied:

$$x_{n-1}(t) + \frac{v^2{}_{n-1}(t)}{2b_{n-1}(t)} - L_n \geq x_n(t) + \frac{v_n{}^2(t+\tau)}{2b_n(t)} + \frac{1}{2}(v_n(t) + v_n(t+\tau))(\frac{3}{2}\tau) \tag{D-2}$$

The assertions for the steady-state macroscopic analysis of traffic are that all vehicles have identical behaviour and they drive at the same speed at all times. Therefore the speeds in eq. (D-2) can be substituted by:

$$v = v_i(t) = v_i(t+\tau), \qquad and\ L = L_i \qquad i = n, n-1$$

The vehicles all apply the same deceleration, i.e.

$$b = b_n(t) = b_{n-1}(t)$$

Therefore the inequality (D-2) can be written as:

$$x_{n-1}(t) + \frac{v^2}{2b} - L = x_n(t) + \frac{v^2}{2b} + v(\frac{3}{2}\tau) \qquad \text{(D-3a)}$$

or

$$x_{n-1}(t) - L = x_n(t) + \frac{3}{2}\tau v \qquad \text{(D-3b)}$$

The density of this uniform traffic is $\rho = [x_{n-1}(t) - x_n(t)]^{-1}$. At jam density, the minimum separation between vehicles is L, i.e. ρj=1/L. Hence eq. (D-4) gives the speed and density relationship as:

$$v = \frac{2}{3\tau}(\frac{1}{\rho} - \frac{1}{\rho_j}) \qquad \text{(D-4)}$$

The resulting flow-density relationship is (line A-J in Figure D-1):

$$q = \rho v = \frac{2}{3\tau}(1 - \frac{\rho}{\rho_j}) \qquad \text{(D-5)}$$

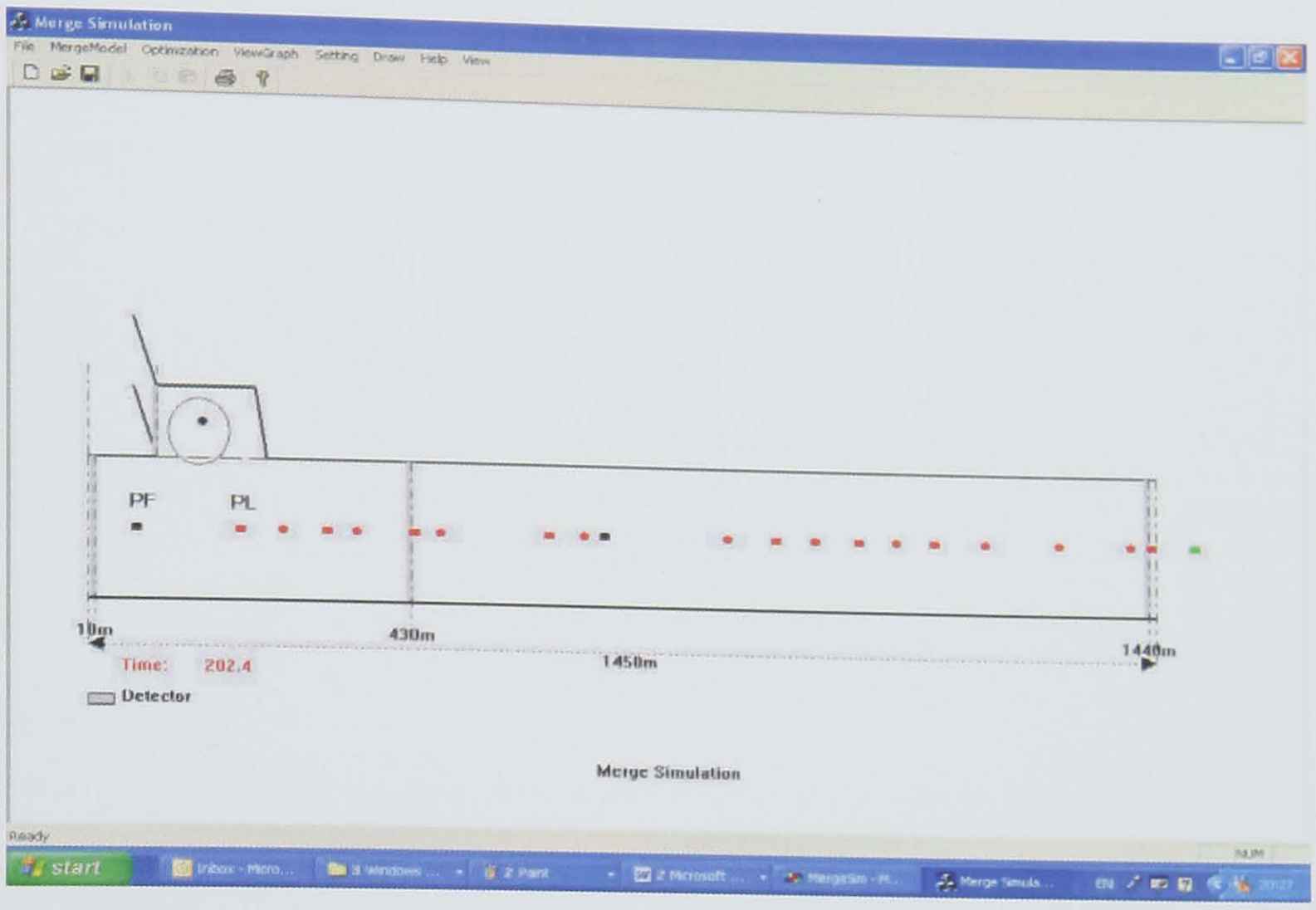
The speed-flow-density relationships for Gipps' model as represented in eq. (D-4) and (D-5) suggest that the speed and flow decrease with increase of the density. The intersection point A in Figure D-1 can be derived from eq. (D-1) and eq. (D-5) as:

$$\frac{2}{3\tau}(1-\frac{\rho_A}{\rho_j}) = \rho_A v_f$$

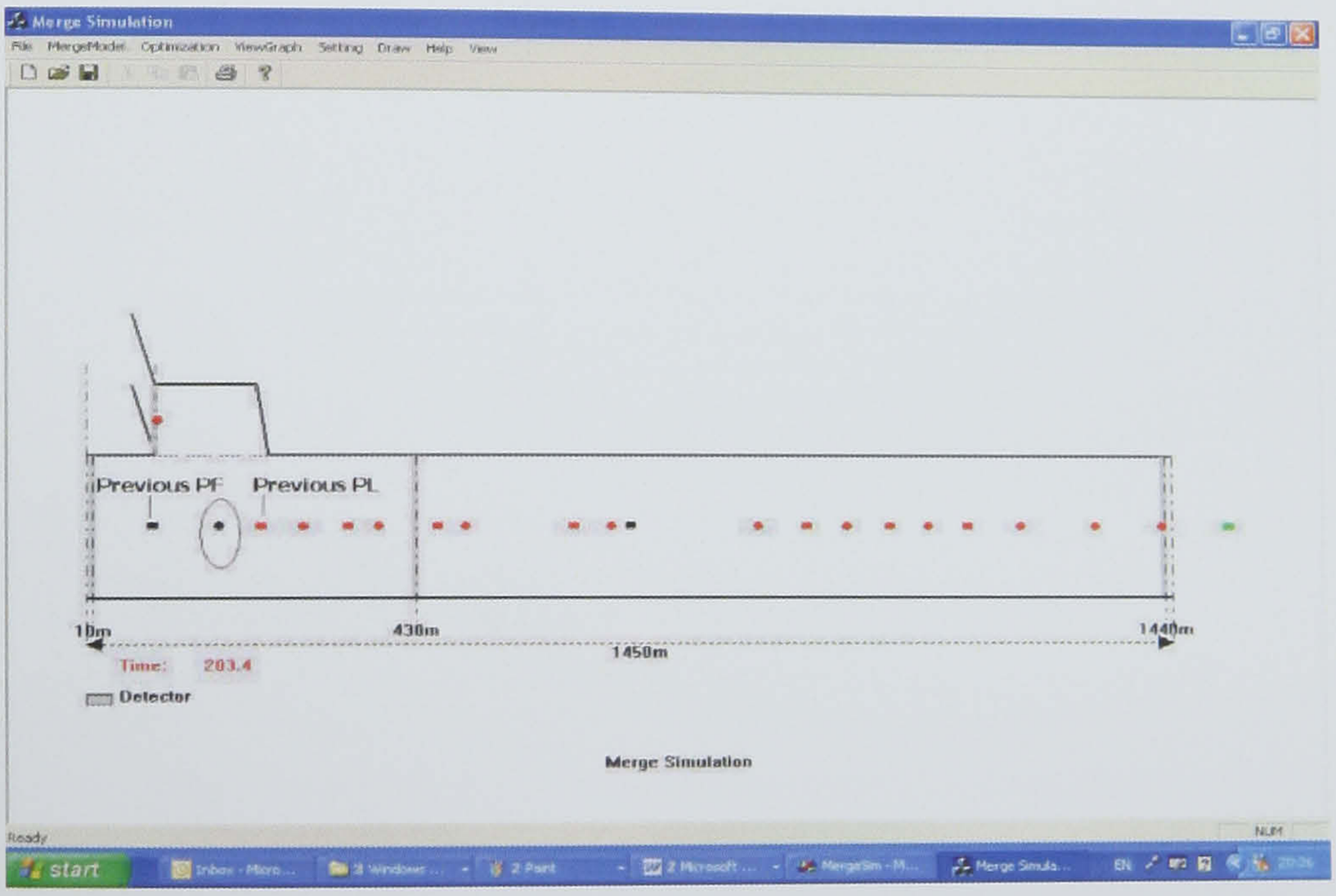
Hence:

$$\rho_A = \frac{2\rho_j}{3\tau\ v_f \rho_j + 2} \qquad \text{and} \qquad q_A = \frac{2\rho_j v_f}{3\tau\ v_f \rho_j + 2}$$

Appendix E- Screenshots of Integrated Simulation Model on a Motorway Merging Section



(a)



(b)

Figure E-1 Screenshots of one merging process of the integrated simulation model on a motorway merging section. (a) shows the merging vehicle (in the circled area) running on the acceleration lane and looking for gap; (b) illustrates the merging vehicle has successfully taken the original gap. HGVs are shown in black colour and Cars are shown in red colour; merging vehicles are in circular shape; motorway vehicles are shown as squares. Three detectors are located on the motorway road: one detector is located on the ramp road.