Micro-Simulation Modeling Approach to Applications of On-Line Simulation and Data Fusion

UCI-ITS-TS-WP-04-9

Lianyu Chu ¹
Will Recker ²

¹ California PATH, ATM Center, Institute of Transportation Studies
University of California, Irvine
klchu@translab.its.uci.edu

² Department of Civil and Environmental Engineering and
Institute of Transportation Studies, University of California, Irvine
wwrecker@uci.edu

January 2004

Institute of Transportation Studies
University of California, Irvine
Irvine, CA 92697-3600, U.S.A.
http://www.its.uci.edu
Micro-simulation Modeling Approach to Applications of On-line Simulation and Data Fusion

Lianyu Chu
Will Recker

California PATH Research Report
UCB-ITS-PRR-2004-1

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation; and the United States Department of Transportation, Federal Highway Administration.

The contents of this report reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California. This report does not constitute a standard, specification, or regulation.

Final Report for Task Order 4143

January 2004
ISSN 1055-1425

CALIFORNIA PARTNERS FOR ADVANCED TRANSIT AND HIGHWAYS
Micro-simulation Modeling Approach to Applications of On-line Simulation and Data Fusion

Lianyu Chu and Will Recker

California PATH ATMS Center
Institute of Transportation Studies
University of California at Irvine
Irvine, CA
ACKNOWLEDGEMENTS

Scott Atiken and Ewan Speirs, from Quadstone in Scotland, provided invaluable technical supports in the process of applying the PARAMICS model. Their continuous collaboration to the project greatly facilitated the work.
EXECUTIVE SUMMARY

This report summarizes research work conducted under TO4143 at the California PATH ATMS Center at the University of California, Irvine. This project has two tasks:

- Functionality enhancements of the PARAMICS simulation model through API programming for the on-line simulation application;
- On-line data fusion algorithm for a better section travel time estimation based on point detector data and probe vehicle data.

In order to conduct these two tasks, we complete the following two related studies, which are the basis of the two tasks of this project:

- Development of the capability-enhanced PARAMICS simulation environment through API programming;
- Calibration and validation of microscopic simulation models.
Chapter 1. Introduction

This study focuses on the use of microscopic simulation, i.e. PARAMICS, as a tool for on-line simulation and data fusion studies under a traffic system with various Intelligent Transportation Systems (ITS) components. These ITS components include actuated / adaptive signal control, ramp metering, traffic surveillance cameras, Changeable Message Sign (CMS), loop detector data collection, and the ITS communication system. A Traffic Management Center can obtain traffic data from multiple sources and control / manage traffic. This kind of traffic system has been implemented in the real world. In order to emulate this ITS-ready traffic system, the capabilities of commercial PARAMICS model needs to be enhanced through API programming. A series of API plugins are developed for this purpose.

A simulation model of the target network, which can be established under this ITS-capable traffic simulation model, needs to be calibrated first. Otherwise, it cannot be used for any application. Due to the lack of systematic rules and guidelines of calibration and validation of traffic models, a generic approach that is suitable for any microscopic simulation models is proposed. This multi-stage procedure is demonstrated in a calibration study with a corridor network of the southern California in PARAMICS.

Based on a calibrated network with various ITS components, we can implement and evaluate a variety of Advanced Transportation and Information Systems (ATMIS) applications. This project only focuses on studies or the on-line simulation and data fusion for a better section travel time estimation based on point detector data and probe vehicle data.

Potentially, the on-line simulation can be used to provide real-time traffic information and select the best control and management strategies according to the testing simulation runs of candidate control strategies using predicted traffic demands. The task of this project is to enhance the capabilities of PARAMICS to enable its application for on-line simulation. We establish the connection between real-world loop detector data and simulation. A simple OD estimation method is developed for the estimation of dynamic OD demand matrices for a freeway network based on real-world loop detector data. A Kalman filtering based traffic flow prediction algorithm is also developed. The predicted traffic flows serve for the prediction of future demand matrices. In terms of the demand loading or vehicle releasing from a zone, PARAMICS releases vehicles based on random numbers. Our demand loading model makes it possible for PARAMICS to release vehicles based on a certain distribution, i.e. the headway between two adjacent vehicles subject to a composite model, which is a combination of shifted negative exponential distribution (under very low traffic flow condition) and normal distribution with the mean of a constant headway (under congested or saturated traffic condition).
The data fusion study of this project is motivated by the recent advances in surveillance technologies, such as Global Positioning Systems (GPS), Automatic Vehicle Identification (AVI), and vehicle re-identification. These technologies promise high accuracy traffic information; however, their performance in field is insufficient yet mostly due to the lack of probe vehicles. Meantime, the most prevailing traffic surveillance system in the world is the conventional inductive loop detector system although this system often fails to provide accurate measures. We discuss how to improve travel time estimates from loop detectors by incorporating data from advanced surveillance technology, and propose a data fusion method based on Adaptive Kalman Filter (AKF) technique that dynamically calculates data noise variances by adapting to the real-time data. The proposed algorithm is tested and evaluated under both recurrent and non-recurrent congestion using the enhanced PARAMICS simulation model. In this study, as a most plausible case, we use two different sources of data: one from loop detectors and the other from a small sample of probe vehicles. The evaluation results show that the proposed fusion algorithm significantly improves section travel time estimates compared to the cases when single data source is used.

This report is organized as follows. Chapter 2 describes our development of the capability-enhanced PARAMICS simulation environment through API programming. Chapter 3 proposes a general approach to calibrate and validate microscopic simulation models and provides an example calibration study under PARAMICS environment. Chapter 4 explains how we enhance the capabilities of PARAMICS in order to make it fit to the needs of on-line simulation. Chapter 5 presents our efforts on the development of data fusion algorithm that can provide a better estimation of section travel time.
Chapter 2. Development of the Capability-Enhanced PARAMICS Simulation Environment

2.1 INTRODUCTION

Microscopic traffic simulation is a software tool to model the real-world traffic system, including the road, drivers, and vehicles, in fine details. In the micro-simulation process, the state of an individual vehicle is continuously or discretely calculated and predicted based on vehicle-vehicle interactions. The car-following, lane-changing and gap-acceptance models are the basic elements of a microscopic traffic simulator. Notable instances of micro-simulators include PARAMICS, CORSIM, VISSIM, AIMSUN2, TRANSIM, and MITSIM [1].

With the advancements of computer technology, micro-simulation has become an increasingly popular and effective tool for many applications, which are not amenable to study by other means. For the academic side, one application is to test new traffic models. For the practical side, most applications are the evaluation (and development) of Intelligent Transportation Systems (ITS) because they are difficult to be evaluated in the field operational tests and they need cost-benefit analyses before their implementation [2].

In general, ITS involves the introduction of a variety of advanced technologies, such as information technologies, to the transportation system in order to make the existing road network be efficiently used. Typical ITS applications include adaptive signal control, ramp metering, and dynamic route guidance system, etc. One of their common features is that they need the real-time traffic information, generally collected by detectors or probe vehicles. For a complex ITS strategy, such as Changeable Message Sign (CMS) routing, not only drivers' behaviors and routes but also traffic control facilities need to be controlled. Though current micro-simulators have enabled a lot of ITS features, it is still impossible to implement a complicated ITS strategy by the micro-simulator itself.

The direct way to enhance the capabilities of a micro-simulator is to work on the source codes. Another way is to code complementary or enhanced components through Application Programming Interface (API) programming and then interface them to the simulator. Since the source code is proprietary, API programming is a practical way for users. API programming needs to use provided API library of a micro-simulator, which includes a set of interface functions, through which users can access the core models of the micro-simulator. Most current popular commercial micro-simulators, including PARAMICS, VISSIM, and AIMSUN 2, provide a series of their API functions for users.

This chapter will describe our practices on developing a capability-enhanced PARAMICS simulation environment through API programming. It is organized as follows. First, we will give a brief overview of the micro-simulator, PARAMICS, used
for the capability enhancements. We explain how API works in PARAMICS, in what aspects we enhance the capabilities of PARAMICS, and the framework of the capability-enhanced PARAMICS simulation environment. Then, we describe the basic enhancement modules in more details. The advanced modules, developed on top of basic modules, are further introduced. Finally, concluding remarks are presented.

2.2 FRAMEWORK OF CAPABILITY ENHANCEMENTS

2.2.1 Overview of PARAMICS

PARAMICS (PARAllel MICrosopic Simulation) is a suite of microscopic simulation tools used to model the movement and behavior of individual vehicles on urban and highway road networks [3]. It offers very plausible detailed modeling for many components of the traffic system. Not only the characteristics of drivers, vehicles and the interactions between vehicles but also the network geometry can influence simulation results. PARAMICS is fit to ITS studies due to its high performance, scalability and the ability of modeling the emerging ITS infrastructures, such as loop detectors and VMS. In addition, PARAMICS provides users with API through which users can customize and extend many features of the underlying simulation model without having to deal with the underlying proprietary code. Though PARAMICS can model some simple ITS strategies, API programming is eventually required to implement more complicated ITS strategies.

2.2.2 How API works in PARAMICS

PARAMICS provides users with an API library that include a set of interface functions, which can be used to access its core models. Basically, PARAMICS provides two groups of interface functions, callback functions and control functions. The callback functions are used for providing information about the attributes of vehicles and their environment. There are two types of control functions, override and overload functions. Override functions are used to replace an internal function of standard simulation loop, such as car-following models. Overload functions are used to add additional functions to the PARAMICS simulation loop.

As shown in Figure 2.1, the simulation process is like this: after the start of simulation, some basic elements of the simulation, such as the speed and position of vehicles, traffic signals, etc., are updated at every time step. If an API module is involved in the simulation, it may work at every time step, or be triggered at a specific simulation time or by a specific event. In general, an API module gets necessary information from the simulation world through callback function and then affects the simulation through control functions.

2.2.3 Aspects of PARAMICS need to be complemented and enhanced
Each micro-simulator has its own features to simulate the real-world traffic. Specifically for PARAMICS, many aspects of PARAMICS need to be complemented and enhanced through API programming in order to better model ITS. Our current efforts are limited to the following basic aspects of the software.

![Diagram](image.png)

Figure 2.1 The PARAMICS simulation process with API modules

2.2.3.1 Routing

PARAMICS is a link-based simulator. Vehicles being simulated do not carry their whole routes but decide their route based on the routing table stored at each node along its route. These routing tables are pre-calculated based on the currently used traffic assignment method. A path-based routing mechanism is required for the applications of traveler information related ITS strategies, such as dynamic route guidance or CMS routing.

2.2.3.2 Real-time traffic information collection

The common feature of ITS is that ITS needs the real-time traffic information, generally collected by detectors or probe vehicles, for decision-making. PARAMICS can model loop detectors, the most frequently used sensors in the real world. However, aggregated loop data, which are generally provided by freeway systems and can be used for real-time
traffic control, cannot be obtained directly from PARAMICS simulation. Also, the concept of probe vehicles needs PARAMICS to track a certain percentage of probe vehicles and extract travel time or travel speed information from them. This cannot be done without the involvement of API programming.

2.2.3.3 Signal control

PARAMICS can basically model the fixed-time signal control. Besides, PARAMICS also provide a plan/phase language (i.e., a kind of script language) to simulate some simple actuated signal control logics. However, in the field, the widely used actuated signal controller uses the complex NEMA logic or type-170 logic. Our experiences found this script language is difficult to be used to model these complex control logics and to replicate these logics to multiple signalized intersections.

2.2.3.4 Ramp metering control

PARAMICS can model fixed-time ramp metering with multiple timing plans. However, a ramp-metering controller, developed in PARAMICS API, is required for the support of development of adaptive ramp metering algorithms, which have more complicated control logics. The ramp-metering controller should provide interface functions that can be used for querying the old metering rate and setting a new metering rate based on the adaptive ramp metering algorithms. When the adaptive ramp-metering algorithm is not activated, the fixed-time metering will be the default control.

2.2.3.5 Database connection

PARAMICS does not have the capability to connect with a database. The advantage of the use of database is that database can become the medium where API modules can exchange data with outside programs or applications.

2.2.3.6 Performance measure

PARAMICS has strong abilities on the collection of statistics data. Except for one general performance data, PARAMICS can output link-based, trip-based, intersection-based, and detector-based data. However, the current difficulties are:

1. With the increase of the size of the network, the number of links, trips, intersections, and detectors increases drastically in PARAMICS.
2. Large amount of data are required to be processed after simulation runs in order to obtain the expected performance measures.
3. Some performance measures, such as on-ramp waiting time, cannot be extracted from output measurement data.
4. PARAMICS has a restriction on the number of output files to be opened during simulation under WINDOWS version.

The use of API to collect performance measures can effectively decrease the amount of data post-processing works and obtain more performance measures directly.
2.2.4 Framework

The above capability enhancements are focused on the basic aspects of the microsimulator, PARAMICS. Each of these basic modules refers to an important aspect of simulation. These functionality enhancements can be classified into the following four categories:

1. Basic control modules, including signal, ramp metering and routing
2. Traffic information collection
3. Database connection
4. Performance measures

Figure 2.2 shows the framework of the capability-enhanced PARAMICS simulation environment. Any an API module in the enhanced PARAMICS environment exchanges dynamic data with the core PARAMICS model and other advanced API modules through the Dynamic Linking Library (DLL) mechanism.

Besides these basic modules, the capability-enhanced PARAMICS environment also includes advanced modules. Examples of advanced modules include ramp metering algorithms, adaptive signal control, and integrated control, which include several ITS components. These advanced modules are developed on top of basic enhancement modules. This hierarchical API development approach, demonstrated in Figure 2.3, can thus re-use the codes developed in the basic modules.

![Diagram of Framework of the capability-enhanced PARAMICS simulation](image)

Figure 2.2 Framework of the capability-enhanced PARAMICS simulation
2.3 BASIC MODULES

2.3.1 Basic control modules

2.3.1.1 Full-actuated Signal Control

This plug-in module implements the eight-phase, dual-ring, concurrent controller logic. The data input to this API is the signal timing plan, the geometry and detector information of each intersection. Interface functions have been provided by this API for external modules to acquire and change the default timing plan. This API provided a couple of interface functions for external API modules to acquire the current signal timing plan and set a new timing plan to a specific signal. An advanced signal control algorithm API can be further developed based on them. The prototypes of these interface functions are shown below [4].

```c
Signal* signal_get_parameters(char *nodeName);
```

Function: Querying the current signal timing plan of a specific actuated signal

Return Value: The current timing plan of an actuated signal.
Parameters: **nodeName** is the name of the signal node.

**Signal** is the structure of actuated signal data, whose definition is:

```c
typedef Signal {
    // intersection name and location
    char *nodeName;
    char *controllerLocation;

    // signal parameters
    int movements[8];
    float maximumGreen[8];
    float minimumGreen[8];
    float extension[8];
    float storedRed[8];
    float phaseGreenTime[8];
    float movementGreenTime[8];

    // current phase information
    int currentPhase;
    int expiredTime;
    float redTimeLeft;
    Bool cycleEndFlag;
} Signal;
```

**Void signal_set_parameters(Signal *sig);**

Function: Setting a new timing plan to a specific signal.

Return Value: None

Parameters: **sig** stores the new timing plan.

2.3.1.2 Ramp Metering Control

This plug-in module is designed to model pre-timed ramp metering control on either one-car-per-green basis or n-cars-per-green basis (with n > 1). It also supports multiple timing plans, HOV bypass, and the use of ramp detectors for metering control. The data input of this API is a time-of-day ramp control plan and the detector information of each meter. In addition, this API provided a couple of interface functions for external API modules to acquire the current metering rate and set a new metering rate to a specific ramp meter. An advanced ramp-metering algorithm API can be further developed based on these interface functions. The prototypes of them are shown below.

**RAMP *ramp_set_parameters (char *rampnode)**

Function: Querying the current metering plan of a specific ramp meter.

Return Value: The current metering control plan of an on-ramp signal.
Parameters: **rampnode** is the name of an on-ramp signal node.
**RAMP** is the structure of ramp control data, whose definition is:

```c
typedef Ramp
{
   // on-ramp signal node name and its location
   char *rampNode;
   char *controllerLocation;
   // ramp control types and parameters
   int ControlType;
   float meteringCycle;
};
```

Where **CTLTYPE** is the status (or type) of the ramp metering control, which can be 0 (if RAMP_CLOSURE), 1 (if RAMP_ON with single-entry metering), 2 (if RAMP_ON with platoon metering) and 9 (if RAMP_OFF).

**void ramp_set_parameters(RAMP *ramp,Bool status)**

**Function:** Setting a new metering rate to a specific ramp meter.

**Return Value:** None

**Parameters:**
- **ramp** stores the new metering control data of a specific on-ramp;
- **status** is a Boolean value. **status** = TRUE means to set a new metering rate based on an external algorithm; **status** = FALSE means to restore the default time-of-day timing plans.

2.3.1.3 Path-based routing

The path-based routing plug-in module establishes the mechanism for vehicles to follow a given path, which is an essential requirement for the simulation of driver responses to the information supply and the resulting route choice. Only those vehicles that need to follow specific paths will be guided by this API and other vehicles will select a route based on the internal routing method of PARAMICS. One interface function of this plugin, which is used to set a path for a vehicle to follow, is shown as follows:

**void uci_vehicle_route_set(void *Vp, VROUTE route)**

**Parameters:**
- **Vp:** the pointer of a vehicle, equivalent to vehicle ID
- **route** is the initial address of the whole path

**VROUTE** is a link list storing whole path the vehicle should follow, which is defined as:

```c
typedef VROUTE
{
   // link name along the route
   char *linkName;
   VROUTE *next;
};
```
2.3.2 Traffic Information Collection

2.3.2.1 Loop detector data

In the real world, loop detectors are placed on freeways and arterials for collecting aggregated data at a certain time interval (typically, 30 seconds) for the purposes of traffic analysis and traffic control. These aggregated loop data are stored either in the database or shared memory. Other traffic operation components can get access to these data through data communication networks and use them for generating real-time control strategies.

The loop data aggregator API works as the traffic data collection and provision server in the enhanced PARAMICS environment. It emulates the real-world data collection from inductive loop detectors and broadcasts the latest aggregated loop data to the dynamic memory during simulation. Other API modules can obtain these data in real time through the interface function provided by this API. In addition, this API can report the aggregated loop data to text files or the MYSQL database as performance measures for data analysis and performance comparison.

One interface function of this API can be used for querying the aggregated loop data at a detector station at a certain time interval. The aggregated loop data includes grouped volume, average occupancy and average speed, as well as lane-based volume, average occupancy and average speed.

** LOOPAGG loop_agg (char *detectorName) **

Return Value: The aggregated detector data of a loop detector

Parameters: detectorName: loop detector name

LOOPAGG is a structure that has the following definition:

```c

    type LOOPAGG
    {
        int detectorIndex;
        float AggregationTime;
        int lane;
        int g_vol;
        float g_occ;
        float g_spd;
        int *vol;
        float *occ;
        float *spd;
    };
```

where detectorIndex is the network-wide index for the detector;

13
Aggregation time is the time of the latest aggregation, determined by the loop data collection interval; 
\( g_{vol} \) is the total traffic counts passing all lanes of a detector station; 
\( g_{occ} \) is the average occupancy of all lanes at a detector station; 
\( g_{spd} \) is the average speed of all vehicles passing a detector station; 
\( *\text{vol}, *\text{occ}, \text{and } *\text{spd} \) are pointers for recording values of volume, occupancy and average speed at each lane of a detector station.

2.3.2.2 Point-to-point travel time data collection

Similarly, we develop a probe vehicle API to simulate point-to-point travel time data collection through GPS equipped vehicles. The sample rate, which is equivalent to the percentage of GPS equipped vehicles, can be specified through the control interface. Point-to-point travel time data can be output to the dynamic memory and text files or the MySQL database.

2.3.3 Database connection

MySQL database is currently regarded as the most popular and highly efficient Open Source SQL (Sequential Query Language) database in the world. We developed the interface functions, including a set of simple C routines programmed in MySQL API, in order to connect PARAMICS with MySQL database.

MySQL database can thus be used for storing intermediate simulation data or final simulation results if large amounts of data are generated. In addition, API modules of PARAMICS can also get access to the database during the simulation to obtain outside data.

2.3.4 Performance measures

An important aspect of the evaluation studies is to use some overall performance measures to evaluate how the implementation of an ITS strategy benefit the whole traffic system, including the freeway and arterial part of the network. To be more efficient to evaluation studies, we developed a MOE API for computing, gathering and reporting a number of user-preferred overall performance measures, each of which corresponds to a specific aspect of interest.

2.3.4.1 Overall performance

MOE #1 system efficiency measure: average system travel time (ASTT) of the whole simulation period. ASTT is calculated as the weighted mean of the average travel times of all OD pairs

\[
ASTT = \frac{\sum_{i,j} (T_{i,j} \cdot N_{i,j})}{\sum_{i,j} N_{i,j}}
\]  

(2.1)
where $N_{ij}$ is the total number of vehicles that actually traveled from origin $i$ to destination $j$; $T_{ij}$ is the average OD travel time from origin $i$ to destination $j$.

MOE #2 system reliability measure: average standard deviation of OD travel times (Std TT) of the entire simulation period. Std TT is calculated as the weighted standard deviation of the average travel times of all OD pairs for the whole study period:

$$\text{Std TT} = \frac{\sum_{i,j} (\text{Std}(T_{ij}) \cdot N_{ij})}{\sum_{i,j} N_{ij}}$$  \hspace{1cm} (2.2)

where Std$(T_{ij})$ is the standard deviation of the average OD travel time from origin $i$ to destination $j$.

2.3.4.2 Freeway Performance

MOE #3 freeway efficiency measure

(1) average mainline travel speed of the entire simulation period (AMTS)
(2) average mainline travel speed during the congestion period (peak_AMTS). The congestion period is defined as the congestion period of the baseline scenario.

MOE #4 on-ramp efficiency measure

(1) total on-ramp delay (TOD)
(2) time percentage of the on-ramp queue spillback to the local streets (POQS)

2.3.4.3 Arterial Performance

MOE #5 arterial efficiency measure

(1) average travel time from the upstream end to the downstream end of an arterial (ATT)
(2) the standard deviation of ATT (std_ATT)

2.4. ADVANCED MODULES

An advanced module, such as a signal coordination strategy, adaptive ramp metering algorithm, or an integrated control strategy, generally involves the control of one or several control components, including signals, ramps, VMS, and routing vehicles, based on traffic information. As introduced in Section 2.4, the capability-enhanced PARAMICS environment uses a hierarchical approach to develop advanced modules. In other words, advanced modules are developed based on several basic modules, which can be accessed by interface functions provided by basic modules.

As an example, Figure 2.4 describes how an advanced ramp-metering algorithm is developed. The advanced API is built on top of two basic plug-in modules, i.e., ramp metering controller and loop data aggregator. The entrance ramp signals in the simulation network are controlled by the ramp metering API, through which metering rates can be
queried and set by other API modules. The loop data aggregator API emulates the real-world loop data collection, typically with a thirty-second interval, and broadcast the latest loop data to the dynamic memory. The advanced API can access the dynamic memory and obtains the required loop data through interface functions provided by the loop data aggregation API. Then the metering rate for the next control interval is calculated based on the advanced ramp-metering algorithm. The new metering rate is sent back to the ramp controller API for implementation [5].

![Diagram](image)

**Figure 2.4** The hierarchical approach to develop advanced ramp metering module

### 2.5 CONCLUDING REMARKS

This chapter presents our practices on the development of a capability-enhanced PARAMICS simulation environment through integrating some plug-in modules implemented in PARAMICS API. These plug-in modules complement and enhance the functionalities of the commercial PARAMICS model. As a result, the capability-enhanced PARAMICS simulation can better model and evaluate ITS.

Our experiences show that API can be used to access the core models of a micro-simulator and potentially, researchers can use commercial micro-simulators as a shell for testing their own models and algorithms. Since other commercial micro-simulators, such as VISSIM and AIMSUN 2, also provide users with their own API functions, users can replicate our methods to enhance their capabilities.

Our current capability enhancements of PARAMICS only cover some aspects of the micro-simulator, more efforts are expected in order to make the enhanced PARAMICS better fit to ITS-related studies.
2.6 REFERENCES


Chapter 3. Calibration and Validation of Microscopic Simulation Model

3.1 INTRODUCTION

Simulation modeling is an increasingly popular and effective tool for analyzing transportation problems that are not amenable to study by other means. In the transportation simulation field, there is general agreement that microscopic simulation, i.e., a computational resolution down to the level of individual travelers, is now a viable alternative and may be the only answer to questions arising from a wide variety of problems. Recent advancements in computer technology have led to the development of high fidelity microscopic simulation models. Examples of widely used microscopic traffic simulation models are AIMSUN2, MITSIM, PARAMICS, and VISSIM.

A microscopic traffic simulation model generally includes physical components, such as the roadway network, traffic control systems, and driver-vehicle units, etc., and associated behavioral models, such as driving behavior models and route choice models. These components and models have complex data requirements and numerous model parameters. Although most simulators provide data input guidelines and default model parameters, these models nevertheless need to be calibrated for the specific study network and the intended applications [1].

In the traditional process of model calibration, model parameters are adjusted until reasonable (qualitative and quantitative) correspondence between the model and field-observed data is achieved. Such adjustments with multiple parameters are a time-consuming and tedious process. The trial-and-error method based on engineering judgment or experience is usually employed for model calibration. More systematic approaches include the gradient approach and Genetic Algorithms (GA) [2, 3]. These approaches regard the model calibration procedure as an optimization problem in which a combination of parameter values that best satisfies an objective function is searched.

Most calibration efforts reported in the literature have focused on either the calibration of driving behavior models [3, 4, 5, 6], or the calibration of a simple linear freeway network [2, 7, 8, 9]. However, these studies represent only an incomplete process of model calibration and validation. In order to analyze network-wide transportation problems, it is necessary to include not only driving behavior model calibration but also dynamic OD estimation and route choice in the calibration process.

The purpose of this paper is to provide a systematic, network-level calibration and validation framework for microscopic traffic simulation models. Along with the framework, we provide a case study to demonstrate how we calibrate a corridor network consisting of both a freeway and its adjacent parallel streets using the PARAMICS microscopic simulation model.
This chapter is organized as follows: an overall calibration and validation procedure is proposed in Section 2. The third section briefly describes the study network and data used in the calibration study. Section 4 explains the details of how the study network is calibrated using the proposed procedure. The model validation results are given in Section 5. Discussion and conclusions are presented in the last section.

3.2 OVERALL CALIBRATION/VALIDATION PROCEDURE

Model calibration involves checking the model results against observed data and adjusting parameters until the model results fall within an acceptable range of error. Various components in microscopic simulation models are calibrated according to the following procedure, as shown in Figure 3.1.

3.2.1 Basic Data Input / Network Coding

The first step is to input basic data, including network geometry, travel demand, drivers, vehicle characteristics, and traffic control systems. The accuracy and/or availability of these data are very important because these inputs directly affect the performance of the micro-simulation models, and the accuracy of such inputs needs to be taken into consideration during the model calibration and validation procedures. For most cases, demand data are not available. The proposed procedure includes the OD estimation as a part of calibration in step four.

3.2.2 Calibration of Driving Behavior Models

The second step involves the calibration of driving behavior models using either disaggregated data or aggregated data. The driving behavior models include car-following (or acceleration) and lane-changing models, which govern vehicular traffic movement and need to be calibrated for the specific region. Global parameters of these models are calibrated within the sub-network level.

3.2.3 Calibration of the Route Choice Behavior Model

The third step is to calibrate the route choice behavior model, which must be conducted on a network-wide level. The route choice behavior model can be calibrated using either aggregated data or by aggregating individual data obtained from driver surveys.

3.2.4 OD Estimation

The next step is OD estimation and/or model adjustment. This step involves several tasks if applicable.

3.2.4.1 Acquisition of OD demand pattern
A simulation model needs to have an OD demand pattern as a starting point of the calibration. A good source of this is from the planning models, such as Transplan and TransCAD based on the social-economic data of the target network. Of course, this OD demand pattern can also be obtained from the traditional four-step model calculation using those social-economic data.

![Flowchart of calibration procedure](image)

Figure 3.1 Flowchart of calibration procedure
3.2.4.2 Modification of route choices

This task involves the modification of the route/link specific cost in order to reflect reasonable route choices. In the OD estimation process, all model parameters and route choice behaviors need to be fixed first since this process is based on the traffic assignment matrix that is affected by any change in simulation input or parameters.

3.2.4.3 Fine-tuning the total OD matrix

The OD demand matrix from the planning model is not accurate enough and thus this matrix can become a starting point of the calculation of a better OD matrix. This OD matrix can be called the total OD demand matrix that needs to cover the whole simulation period. The estimation of the total OD matrix is an iterative process to match simulation results with the aggregated traffic volume observations at some specific measurement locations. The objective of this process is:

\[
\min \sum_i (VOL_{act}(i) - VOL_{est}(i))^2
\]  

(3.1)

where \( N \) is the total number of measurement locations.

This OD estimation process can be conducted outside of the microscopic simulation model. There are some software tools, such as TransCAD, QueenOD that can help the estimation of a static OD. However, this may cause some OD estimation errors if different traffic assignment methods are used in the OD estimation process and the microscopic simulation. The microscopic simulation models have started to develop their own OD estimation tool in order to avoid this problem. For example, INTEGRATION provides QueenOD as its OD estimation tool and PARAMICS provides an OD estimator tool in its new version.

3.2.4.4 Dynamic OD estimation

In order to obtain a more accurate simulation model, the dynamic (or time-dependent) OD estimation is required. Theoretically, it is a dynamic OD demand estimation problem, which may not need a total OD as basis. So far, there is not an effective method that can solve this problem [15,16], except that the FREQ model can be used for the estimation of time-dependent OD of freeway network for micro-simulation [7].

This estimation is generally based on the total OD demand matrix, estimated in a previous step. This dynamic OD estimation process can be regarded as a process that assign the total OD to detailed time slices. Though some existing OD estimation tools, i.e. QueenOD and Estimator of PARAMICS, have a certain potential to handle the dynamic OD estimation, their capabilities are not recognized.

3.2.5 Model Fine-tuning
The last step of calibration is model fine-tuning using aggregated traffic data in order to reflect network-level congestion effects. The driving behavior models need to be further validated locally (intersection-by-intersection or junction-by-junction) and adjusted to reflect the local characteristics. The local characteristics can be basically examined through the comparison of volume-occupancy curves drawn based on aggregated point detector data from both simulation and the real world.

This step of calibration is performed as a two-objective optimization process:

\[
\begin{align*}
\min & \sum_{r \in R} \sum_{t \in T} \left( \text{VOL}_{t, (a, t)} - \text{VOL}_{t, (a, t)}^d \right)^2 \\
\min & \sum_{r \in R} \sum_{t \in T} \left( \text{TT}_{t, (a, t)} - \text{TT}_{t, (a, t)}^d \right)^2
\end{align*}
\]  

(3.2)  

(3.3)  

The two objectives can be called volume match and travel time match. Some previous calibration efforts actually started from this point [2, 6]. This is the reason that we think that the current studies represent only an incomplete process of model calibration and validation process. Under the following situations, there is no problem to start calibrating a network from this step:

1. The network has been coded and roughly calibrated.
2. The driving behavior models have been calibrated and validated based on previous studies on the same network.
3. There is no data that can support the calibration of route choice model, or one of the route choice models in the microscopic simulator can be accepted.
4. The OD demand matrices have been given.

3.2.6 Model Validation

Model validation is typically an iterative process linked to each model calibration. The model validation is generally conducted with a different data set of larger area within the modeling network in order to check if the calibrated model parameters are suitable. Model validation is regarded as a final stage to investigate if each component adequately reproduces observed travel characteristics and the overall performance of the model is reasonable.

The following sections of this chapter will focus on a case study that follows the above procedure of model calibration and validation to calibrate a network in a PARAMICS (version 3.0 Build 7), a widely used microscopic simulation model developed by Quadstone in Scotland.

3.3 STUDY SITE AND CALIBRATION DATA ACQUISITION

The data are an important issue in the calibration of a microscopic simulation model. The more data and the better data we have, the better the calibration result can be achieved. In
this section, we talked about data issues based on our practices in the calibration of the case study network in PARAMICS.

3.3.1 Study Site

The study network is a highly congested corridor network in the city of Irvine, Orange County, California, as shown in Figure 3.2. The network includes a 6-mile section of freeway I-405, a 3-mile section of freeway I-5, a 3-mile section of freeway SR-133 and the adjacent surface streets. This network is calibrated to investigate the effectiveness of various Intelligent Transportation Systems (ITS) strategies on relieving traffic congestion happened along the northbound I-405 in the morning peak periods.

![Figure 3.2 Overview of the study network](image)

3.3.2 Preparation of Calibration Data

Microscopic simulation models have very complex data requirements and numerous model parameters. The more data that can be collected and used for calibration, the better the calibration can be.

The following data, including basic input data of the simulation and observed data for calibration/validation purpose are prepared.

1) Vehicle characteristics and performance data
   These data include vehicle length, maximum speed, maximum acceleration and deceleration rates, etc., which are obtained from California Department of Transportation (Caltrans).

2) Vehicle mix by type

23
Vehicle composition by type is determined by the statistical analysis of traffic flow data observed from surveillance videos at two freeway locations in the network.

3) Arterial volume data
We obtained 15-minute interval traffic counts at all important cordon points of the network. Some of these data were from the City of Irvine, collected in January 2002 and June 2001. Others were derived based on the surveillance video data, video-taped between March 27, 2002 and April 19, 2002. The data from the City of Irvine also includes traffic count data at important links inside the network.

4) Travel time data
We obtained floating car data of northbound and southbound freeway I-405 collected at Oct. 17 and Oct. 18 of 2001 from Caltrans.

5) Freeway loop detector data
We directly access mainline, on-ramp and off-ramp loop detector data from an on-line database at the University of California, Irvine.

3.3.3 Calibration Data Reduction

Because field observations vary from day to day due to the stochastic nature of traffic, our calibration objective is to re-construct the typical real-world traffic variation in simulation. We selected one day’s data for the calibration process.

The method we employed in this selection is to compare the peak-hour (i.e. 7-8 AM) volume of a candidate day at any selected loop station with the average peak-hour volume of all candidate days using the GEH statistic, used by British engineers [10]:

\[
GEH = \frac{(\text{vol}(i) - \text{ave}_\text{vol})^2}{(\text{vol}(i) + \text{ave}_\text{vol})/2}
\]

(3.4)

In our study, if the GEH values for more than 85% of the selected loop stations are less than 5, the demand pattern of the candidate day is typical. There are 35 selected loop stations, placed at the upstream end of each freeway, all on-ramps and all off-ramps. This study showed that the demand pattern at Oct. 17, 2001 is typical based on all weekdays’ data in October. Therefore, the travel time and loop data of Oct. 17 were chose for the calibration study.

3.4. CALIBRATION

3.4.1 Basic Data Input / Network Coding

We built the study network in PARAMICS based on aerial photos and road geometry and infrastructure maps obtained from Caltrans and the city of Irvine. The simulated network has the same configuration of loop detector stations as those in the real world and the
same traffic control operations, including actuated signal control and time-based ramp metering, modeled by full-actuated signal and ramp metering plugins, respectively [11]. The zone structure of the network is based on data from the OCTAM model.

The basic input data also include vehicle mix by type, vehicle characteristics and performance data, and some driving restrictions. Since PARAMICS regards each vehicle in the simulation as a Driver Vehicle Unit (DVU), driver data, represented by aggressiveness and awareness factors, are also inputs of simulation. For this study, we used the default distributions of these two factors.

3.4.2 Calibration of Driving Behavior Models

The driving behavior models of PARAMICS have been extensively studied [4, 5, 6]. Lee et al. [5] conducted a calibration study based on a sub-network of this study network, which shows calibrated values of the two major parameters of driving behavior models, i.e. the mean target headway and driver reaction time, are 0.625 and 0.415 respectively. These two values may be changed in the parameter fine-tuning step in order to match the observed congestion patterns.

3.4.3 Calibration of Route Choice Model

Due to the existence of freeways and parallel streets in the study network, the routing algorithm adopted in the simulation is important. We calibrated the network under stochastic assignment. Stochastic assignment in PARAMICS assumes that different drivers perceive different costs from a decision node to the destination. The perceived cost is calculated based on the given perturbation factor with a random number assigned to the vehicle, and the shortest perceived route is chosen at the decision node.

Since we have no data to calibrate perturbation factors, we take a value of 5% for all drivers based on the assumption that most drivers in the morning peak are familiar drivers who have a good knowledge of the road network and traffic condition. In addition, the “familiarity” attribute affects the route choice behaviors in PARAMICS. Since we calibrated the network using the morning peak demand when most travelers are commuters, we assumed that 95% of drivers are familiar drivers, who can choose from among both arterials and secondary streets.

3.4.4 OD Estimation

3.4.4.1 Reference OD matrix

A static origin and destination (OD) demand matrix is sub-extracted from the Orange County Transportation Authority Model (OCTAM) 2000 model. It is used as the reference OD matrix of this study.

The reference OD matrix is evenly loaded for the whole simulation period (subject to a flat demand “profile”). The vehicles are observed as vehicles move through the network.
Abnormal behaviors always correspond to network coding errors, which need to be identified and solved.

3.4.4.2 Modification of route choices

Based on the calibrated route choice model (i.e. stochastic assignment), we need to further fine-tune some local parameters, such as link costs, in order to have a reasonable resulting routing, which can be based on survey data. Since travel delays caused by intersection signals and freeway ramp control are not considered for traffic assignment in PARAMICS, we added tolls to on-ramp links to reflect on-ramp control, and decreased speed limit values of arterial links in order to reflect signal control.

3.4.4.3 Fine-tuning the total OD matrix

The reference OD matrix obtained from the OCTAM model needs to be adjusted for two reasons:

(1) It is for the morning peak hours from 6 to 9. However, the congestion in the study network cannot be totally cleared at 9 o’clock. Consequently, the OD demand matrix needs to be further expanded to 4 hours, i.e. from 6 to 10;

(2) It is not accurate because the data sets of OCTAM model are generally limited to the nearest decennial census year and the sub-extracted OD matrix is based on the four-step model of TRANPLAN. As a result, we need to adjust the whole OD matrix based on the reference OD, the assumed traffic assignment method and its resulting route choice. This is a static OD estimation problem, which has many solution methods. The least square method is most frequently used [12,13].

Since we have 15-minute interval traffic counts at all cordon points of the network, the total traffic attractions and generations of each zone are known. We assumed the same trip distribution as that of the reference OD matrix is applied to all zones in the adjusted OD matrix. The PURNESS technique is then used for balancing the adjusted OD table. If the total attractions are not equal to the total generations, the total generations are used as the total. For the following steps of OD modification, the total origin and destination demands will be basic constraints.

We evenly load the adjusted OD matrix for the whole simulation period (subject to a flat demand “profile”). Based on the simulation results, we can compare the observed and simulated total traffic counts at selected measurement locations with the objective function shown in Equation 3.1. In this study network, there are 52 selected measurement locations, including 13 mainline loop stations, 29 on-ramp loop stations, and 10 arterial links. The measure of the overall quality of the OD estimation is the GEH statistic:

\[
GEH = \frac{(VOL_{\text{obs}}(k) - VOL_{\text{sim}}(k))^2}{(VOL_{\text{obs}}(k) + VOL_{\text{sim}}(k))/2}
\]  

(3.5)

If the GEH values for more than 85% of the measurement locations are less than 5, the adjusted OD is acceptable.
An iteration process is required in order to obtain a satisfactory total OD matrix. Based on a simulation run and its resulting GEH values at measurement locations, we can further adjust the complete OD matrix according to assumption of the proportional assignment [14], which assumes that the link volumes are proportional to the OD flows. The modifications to vehicle routes may also need to be involved to generate an acceptable whole OD matrix.

Based on multiple iterations, a good and acceptable total OD can be obtained. Table 3.1 (left side) shows the calibration results at this step. It shows that, except for a single on-ramp location and one arterial link location, all other measurement locations have a GEH value lower than 5, which satisfies the calibration acceptance criteria of this step.

3.4.4.4 Reconstruction of time-dependent OD demands

The objective of this step is to assign the total OD to a series of consecutive time slices. Our method tries to simplify the complex time-dependent OD estimation problem through reconstructing the dynamic OD demands based on a set of demand profiles. PARAMICS has an enhanced feature of demand loading, i.e. the ability to specify different demand profiles for each OD pair. Through the use of "matrix" and "profile" files, a profile can be specified for a specific OD pair. Since the 15-minute interval traffic counts at all cordon points of the network are known, the profile of vehicle generation from any origin zone and that of vehicle attraction to any destination zones are thus known.

We further assume a number of initial demand profiles for all OD pairs based on the following criteria:

1. The demand profile from an arterial origin zone to any arterial destination zone has the same profile, which is the same as the vehicle generation profile from this origin zone;
2. The demand profiles from a freeway origin zone to a freeway destination zone, from a freeway origin zone to an arterial destination zone, from an arterial origin zone to a freeway destination zone will be based on the traffic count profile at corresponding loop stations placed at freeway mainline, on-ramp, and off-ramp, respectively.

Based on these criteria, we will further find the demand profile for each OD pair.

As we know, the OD estimation process is related to the other calibration process. Without the calibration of other components of the simulation model, the dynamic OD demands are almost impossible to be obtained because the traffic during peak hours involves the capacity of the network, which is what the calibration needs to find out.

As a result, this step has two calibration objective functions. The first one is an easy one, which is to minimize the deviation between the observed and corresponding simulated traffic counts at selected measurement locations for the peak hour of the simulation period:
\[
\min \sum_{i=0}^{N} (VOL_{obs}(n, \text{peak}_i, T) - VOL_{sim}(n, \text{peak}_i, T))^2
\]  

(3.6)

where \( N \) is the total number of measurement locations; \( VOL_{obs}(i, \text{peak}_i, T) \) and \( VOL_{sim}(i, \text{peak}_i, T) \) are total observed and simulated traffic counts for the peak hour at measurement location \( i \), respectively. The selected measurement points are the same as those in last step. The peak hour is defined as from 7 to 8 AM. The following criteria are required to be satisfied for this objective:

- The modeled peak hour volumes at measurement locations must be within 15 percent of the observed volumes for flows greater than 700 vphpl, or within 100 vph for flows less than 700 vph. These targets must be satisfied for 85 percent of the cases;
- Total screenline flows (normally >5 links) to be within five percent for nearly all screenlines;
- The GEH statistic to be less than five for individual flows for 85 percent of the cases, and less than four for screenline totals for nearly all screenlines;

The second objective function is to minimize the deviation between the observed and corresponding simulated traffic counts at selected measurement locations at five-minute interval. It can be formulate as:

\[
\min \sum_{i=0}^{N} \sum_{t=1}^{T} (VOL_{obs}(n, t) - VOL_{sim}(n, t))^2
\]  

(3.7)

where \( N \) and \( T \) are the number of measurement locations and time periods, respectively; \( VOL_{obs}(n, t) \) and \( VOL_{sim}(n, t) \) are observed and simulated traffic counts of time period \( t \) at measurement location \( i \), respectively. The length of each period is 5 minutes in this study.

This step of calibration is an iterative process. We mainly modify the demand profiles from a freeway origin zone to a freeway destination zone, from a freeway origin zone to an arterial destination zone, and from an arterial origin zone to a freeway destination zone in order to match the traffic counts at selected measurement locations. The trial-and-error method is used for the modification of demand profiles based on the following criteria:

1. The profile from a freeway origin zone to a freeway destination zone can be estimated based on the 15-minute loop data at a corresponding off-ramp location.
2. The profile from an arterial origin zone to a freeway destination zone can be estimated based on the 15-minute loop data at a corresponding on-ramp location.
3. The profiles from freeway origin zones to freeway destination zones are decided last.

Traffic count calibration here is an initial match of volume data; volume calibration will be fine-tuned in next step. The calibration results of this step are shown in Table 3.1 (right side), which shows the comparison of traffic counts of peak hour at those selected measurement locations.
<table>
<thead>
<tr>
<th>Mainline Detectors</th>
<th>Observed</th>
<th>Simulated</th>
<th>% diff</th>
<th>GEH</th>
<th>Observed</th>
<th>Simulated</th>
<th>% diff</th>
<th>GEH</th>
</tr>
</thead>
<tbody>
<tr>
<td>40505.63ml</td>
<td>6803</td>
<td>6908</td>
<td>-1.54</td>
<td>0.05</td>
<td>24505</td>
<td>24428</td>
<td>0.33</td>
<td>0.49</td>
</tr>
<tr>
<td>40505.31nl</td>
<td>9127</td>
<td>9096</td>
<td>-1.27</td>
<td>1.37</td>
<td>33274</td>
<td>32045</td>
<td>-3.78</td>
<td>2.46</td>
</tr>
<tr>
<td>40505.88ml</td>
<td>8322</td>
<td>8248</td>
<td>-0.92</td>
<td>0.81</td>
<td>32889</td>
<td>28980</td>
<td>-12.17</td>
<td>11.96</td>
</tr>
<tr>
<td>40505.74ml</td>
<td>9549</td>
<td>9377</td>
<td>-1.73</td>
<td>1.73</td>
<td>34277</td>
<td>33475</td>
<td>-2.39</td>
<td>2.46</td>
</tr>
<tr>
<td>40506.21nl</td>
<td>7690</td>
<td>8135</td>
<td>2.88</td>
<td>1.98</td>
<td>28265</td>
<td>27904</td>
<td>-1.29</td>
<td>2.09</td>
</tr>
<tr>
<td>40505.31ln</td>
<td>8206</td>
<td>8072</td>
<td>2.22</td>
<td>0.96</td>
<td>28501</td>
<td>27765</td>
<td>-2.36</td>
<td>4.21</td>
</tr>
<tr>
<td>40505.77ml</td>
<td>5583</td>
<td>5514</td>
<td>-1.30</td>
<td>0.93</td>
<td>20057</td>
<td>19638</td>
<td>-2.10</td>
<td>2.97</td>
</tr>
<tr>
<td>5x22.326</td>
<td>7533</td>
<td>7686</td>
<td>-2.05</td>
<td>1.75</td>
<td>26380</td>
<td>26014</td>
<td>-1.36</td>
<td>1.32</td>
</tr>
<tr>
<td>5x22.14nl</td>
<td>6499</td>
<td>6974</td>
<td>7.59</td>
<td>5.79</td>
<td>24484</td>
<td>24025</td>
<td>-2.28</td>
<td>2.82</td>
</tr>
<tr>
<td>13x39.37ml</td>
<td>510</td>
<td>471</td>
<td>-7.09</td>
<td>1.78</td>
<td>1496</td>
<td>1498</td>
<td>0.12</td>
<td>0.05</td>
</tr>
<tr>
<td>13x39.10nl</td>
<td>804</td>
<td>817</td>
<td>1.64</td>
<td>0.48</td>
<td>2534</td>
<td>2507</td>
<td>-1.11</td>
<td>1.44</td>
</tr>
<tr>
<td>13x39.10ml</td>
<td>2792</td>
<td>2674</td>
<td>-4.39</td>
<td>1.50</td>
<td>8587</td>
<td>8557</td>
<td>-0.35</td>
<td>0.00</td>
</tr>
<tr>
<td>13x39.37nl</td>
<td>1760</td>
<td>1652</td>
<td>-6.21</td>
<td>2.61</td>
<td>5233</td>
<td>5429</td>
<td>4.08</td>
<td>2.88</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ramp Detectors</th>
<th>Observed</th>
<th>Simulated</th>
<th>% diff</th>
<th>GEH</th>
</tr>
</thead>
<tbody>
<tr>
<td>40505.63rl</td>
<td>160</td>
<td>162</td>
<td>1.25</td>
<td>0.46</td>
</tr>
<tr>
<td>40505.70arb</td>
<td>512</td>
<td>507</td>
<td>-0.97</td>
<td>0.22</td>
</tr>
<tr>
<td>40505.11arb</td>
<td>110</td>
<td>149</td>
<td>35.43</td>
<td>44.7</td>
</tr>
<tr>
<td>40505.17rl</td>
<td>56</td>
<td>54</td>
<td>-3.57</td>
<td>0.27</td>
</tr>
<tr>
<td>40505.19rl</td>
<td>3227</td>
<td>2168</td>
<td>-31.55</td>
<td>1.33</td>
</tr>
<tr>
<td>40505.96rl</td>
<td>165</td>
<td>196</td>
<td>18.81</td>
<td>2.51</td>
</tr>
<tr>
<td>40506.25arb</td>
<td>436</td>
<td>442</td>
<td>1.36</td>
<td>0.29</td>
</tr>
<tr>
<td>40506.86rl</td>
<td>709</td>
<td>731</td>
<td>3.16</td>
<td>3.82</td>
</tr>
<tr>
<td>40506.86rb</td>
<td>307</td>
<td>320</td>
<td>4.67</td>
<td>0.73</td>
</tr>
<tr>
<td>40506.03arb</td>
<td>816</td>
<td>809</td>
<td>-0.85</td>
<td>0.25</td>
</tr>
<tr>
<td>40506.55rl</td>
<td>461</td>
<td>426</td>
<td>-7.60</td>
<td>1.42</td>
</tr>
<tr>
<td>40505.55arb</td>
<td>682</td>
<td>670</td>
<td>-1.84</td>
<td>0.46</td>
</tr>
<tr>
<td>40506.74rab</td>
<td>1026</td>
<td>1075</td>
<td>5.48</td>
<td>1.51</td>
</tr>
<tr>
<td>40506.69r</td>
<td>853</td>
<td>959</td>
<td>12.08</td>
<td>3.22</td>
</tr>
<tr>
<td>40505.68rb</td>
<td>318</td>
<td>276</td>
<td>-13.26</td>
<td>2.32</td>
</tr>
<tr>
<td>40506.50rb</td>
<td>241</td>
<td>281</td>
<td>16.56</td>
<td>2.48</td>
</tr>
<tr>
<td>40506.03rb</td>
<td>409</td>
<td>392</td>
<td>-3.55</td>
<td>0.85</td>
</tr>
<tr>
<td>40506.03rb</td>
<td>183</td>
<td>212</td>
<td>15.67</td>
<td>2.06</td>
</tr>
<tr>
<td>40506.94ar</td>
<td>624</td>
<td>587</td>
<td>-5.66</td>
<td>9.34</td>
</tr>
<tr>
<td>40506.88r</td>
<td>864</td>
<td>817</td>
<td>-5.75</td>
<td>1.82</td>
</tr>
<tr>
<td>40506.88rb</td>
<td>152</td>
<td>159</td>
<td>4.70</td>
<td>0.56</td>
</tr>
<tr>
<td>40506.94rb</td>
<td>582</td>
<td>571</td>
<td>-1.87</td>
<td>0.79</td>
</tr>
<tr>
<td>40506.57r</td>
<td>70</td>
<td>117</td>
<td>51.43</td>
<td>6.86</td>
</tr>
<tr>
<td>40506.56r</td>
<td>1546</td>
<td>1496</td>
<td>-3.14</td>
<td>1.28</td>
</tr>
<tr>
<td>40506.96rb</td>
<td>20</td>
<td>33</td>
<td>65.60</td>
<td>2.33</td>
</tr>
<tr>
<td>40506.77rb</td>
<td>9</td>
<td>11</td>
<td>22.22</td>
<td>0.63</td>
</tr>
<tr>
<td>5x22.1fr</td>
<td>742</td>
<td>802</td>
<td>7.80</td>
<td>2.16</td>
</tr>
<tr>
<td>5x22.1fr</td>
<td>84</td>
<td>94</td>
<td>11.63</td>
<td>1.96</td>
</tr>
<tr>
<td>5x22.2fr</td>
<td>199</td>
<td>232</td>
<td>16.60</td>
<td>2.25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Arterial Detectors</th>
<th>Observed</th>
<th>Simulated</th>
<th>% diff</th>
<th>GEH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jeffry 405-Alton</td>
<td>2119</td>
<td>1943</td>
<td>-7.74</td>
<td>3.45</td>
</tr>
<tr>
<td>Alton E of Jeffry</td>
<td>882</td>
<td>1057</td>
<td>17.78</td>
<td>5.62</td>
</tr>
<tr>
<td>Alton E of Jolliy</td>
<td>768</td>
<td>606</td>
<td>-25.35</td>
<td>5.90</td>
</tr>
<tr>
<td>Alton E of Semb</td>
<td>439</td>
<td>446</td>
<td>1.62</td>
<td>0.25</td>
</tr>
<tr>
<td>Alton E of Sand</td>
<td>624</td>
<td>443</td>
<td>-29.87</td>
<td>7.84</td>
</tr>
<tr>
<td>Alton E of Laga</td>
<td>804</td>
<td>619</td>
<td>-23.92</td>
<td>6.94</td>
</tr>
<tr>
<td>Barrville SR13-1CD</td>
<td>491</td>
<td>606</td>
<td>23.99</td>
<td>4.91</td>
</tr>
</tbody>
</table>

3.4.5 Model Fine-tuning

This step fine-tunes various parameters in order to re-construct traffic variations and match the congestion pattern of the study network. These parameters include:

1. Link specific parameters, including the signposting setting or the target headway of those links at critical bottleneck locations where a very minor change in capacity can have a major effect on congestion.

2. Global parameters for the car-following and lane-changing models, i.e., the mean target headway and driver reaction time. They are two key user-specified parameters in the car-following and lane-changing models that can drastically influence overall driver behaviors of the simulation.

3. Demand profiles from freeway origin zones to freeway destination zones may need to be further modified in order to adapt traffic congestion along freeways.

The objective function of this step is to minimize the deviation between the observed and the corresponding simulated 5min volume and point-to-point travel time measurements. It is a two-objective optimization process, whose formulations are shown in Equation 3.2 and 3. Based on travel time matching, the 5-min traffic counts at measurement points are further matched.

In our study, the point-to-point travel time matching was performed only for the northbound and southbound freeway I-405 between the interchanges at Irvine Center Drive and Culver Drive due to the lack of data for other trips.

Because of the high traffic demands during the peak hour, and recurrent congestion along the northbound I-405, some network coding problems may slow up and need to be corrected. Congestion and queuing phenomena on the northbound I-405 may take extra effort to modify demand profiles of some specific OD pairs. This step may involve multiple simulation runs in order to determine a good combination of these aforementioned parameters. The trial-and-error method is used for parameter modification.

The final calibrated mean target headway and driver reaction time are 0.78 and 0.66, respectively. Figure 3.3 shows the calibrated demand profiles for several major OD pairs.

3.5. OVERALL MODEL VALIDATION

The final model validation requires multiple simulation runs because of the stochastic nature of microscopic simulations. The method to determine the number of runs is described in Section 3.5.1.

3.5.1 Determination of Number of Simulation Runs
Microscopic simulation is a generally stochastic process, which rely upon random numbers to release vehicles, select vehicle type, select their destination and their route, and to determine their behavior as they move through the network. Therefore, the average results of several simulation runs using different seed number can reflect the traffic condition of a specific scenario.

Figure 3.3 Demand profiles for major OD pairs

In order to determine the number of simulation model runs, we need to know the variance of a number of performance measures from simulation results, which are unknown before simulations. The flow chart to determine the number of simulation runs is shown in Figure 3.4.

We execute nine simulation runs first and then calculate the number of runs needed according to the mean and standard deviation of a performance measure of these nine runs:

\[ N = \left( \frac{z_{1-\alpha/2} \cdot \delta}{\mu \cdot \varepsilon} \right)^2 \]  
(3.8)

where \( \mu \) and \( \delta \) are the mean and standard deviation of the performance measure based on the already conducted simulation runs; \( \varepsilon \) is the allowable error specified as a fraction of the mean \( \mu \); \( t_{1-\alpha/2} \) is the critical value of the t-distribution at the confidence interval of \( \alpha \). This calculation needs to be done for all performance measures of interest. The highest value from variances is the required number of runs. If the current number of runs is already larger than this value, the simulation of this scenario is ended. Otherwise, one additional run is performed and then the required number of runs needs to be recalculated.

3.5.2 Validation Results
For our case, the performance measures we use are the total vehicle hour traveled and the northbound point-to-point travel time (i.e. from ICD to Culver). The total number of simulation runs is 31 in order to achieve statistically meaningful performance measures. The measure of goodness of fit we used for evaluating the calibrated simulation model is the mean abstract percentage error (MAPE), which can be calculated by:

\[
MAPE = \frac{1}{T} \sum_{t=1}^{T} \left| \frac{M_{\text{true}}(t) - M_{\text{sim}}(t)}{M_{\text{true}}(t)} \right|
\]

(3.9)

Figure 3.4 Flow chart of the determination of number of simulation runs

We compared the simulation results with the loop data and the floating car data of Oct. 17, 2001. Figure 3.5 shows the comparison of observed and simulated 5-minute traffic counts at eight major freeway measurement locations. The MAPE error of traffic counts at these measurement locations range from 5.8% to 8.7%. The travel time calibration results are shown in Table 3.2. Figure 3.6 and 3.7 show the comparison of observed and simulated point-to-point travel time for the northbound and the southbound I-405, which have the MAPE errors of 8.5% and 3.1%, respectively.
In summary, simulated traffic counts and point-to-point travel time data correspond well to the observed measurements. The calibrated simulation model accurately captures the congestion patterns of the target network shown on Oct. 17, 2001.

Figure 3.5 Traffic counts calibration (5-minute volume) at major freeway measurement locations
<table>
<thead>
<tr>
<th>Mainline Trip Analysis</th>
<th>Start time</th>
<th>Arrival Time</th>
<th>Observed</th>
<th>Start time</th>
<th>Arrival time simulated</th>
<th>% diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>Southbound I-405 from Culver to ICD</td>
<td>6:00:22</td>
<td>6:04:38</td>
<td>256</td>
<td>6:00:00</td>
<td>6:15:00</td>
<td>264.5</td>
</tr>
<tr>
<td></td>
<td>6:20:14</td>
<td>6:29:36</td>
<td>262</td>
<td>6:20:00</td>
<td>6:30:00</td>
<td>257.8</td>
</tr>
<tr>
<td></td>
<td>6:47:01</td>
<td>6:51:17</td>
<td>256</td>
<td>6:49:00</td>
<td>6:50:00</td>
<td>250.7</td>
</tr>
<tr>
<td></td>
<td>7:06:34</td>
<td>7:10:56</td>
<td>262</td>
<td>7:05:00</td>
<td>7:10:00</td>
<td>259.6</td>
</tr>
<tr>
<td></td>
<td>7:24:45</td>
<td>7:28:54</td>
<td>249</td>
<td>7:25:00</td>
<td>7:30:00</td>
<td>263.1</td>
</tr>
<tr>
<td></td>
<td>7:46:23</td>
<td>7:50:48</td>
<td>265</td>
<td>7:45:00</td>
<td>7:50:00</td>
<td>278.9</td>
</tr>
<tr>
<td></td>
<td>8:05:14</td>
<td>8:09:41</td>
<td>267</td>
<td>8:05:00</td>
<td>8:10:00</td>
<td>326.8</td>
</tr>
<tr>
<td></td>
<td>8:24:23</td>
<td>8:28:44</td>
<td>261</td>
<td>8:25:00</td>
<td>8:30:00</td>
<td>262.6</td>
</tr>
<tr>
<td></td>
<td>8:43:42</td>
<td>8:47:47</td>
<td>245</td>
<td>8:45:00</td>
<td>8:50:00</td>
<td>259.5</td>
</tr>
<tr>
<td></td>
<td>9:04:27</td>
<td>9:08:34</td>
<td>247</td>
<td>9:05:00</td>
<td>9:10:00</td>
<td>248.4</td>
</tr>
<tr>
<td>MAPE</td>
<td>3.1%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Northbound I-405 from ICD to Culver</td>
<td>6:00:58</td>
<td>6:04:45</td>
<td>237</td>
<td>6:00:00</td>
<td>6:05:00</td>
<td>247.7</td>
</tr>
<tr>
<td></td>
<td>6:19:32</td>
<td>6:23:40</td>
<td>248</td>
<td>6:20:00</td>
<td>6:25:00</td>
<td>248.3</td>
</tr>
<tr>
<td></td>
<td>6:40:51</td>
<td>6:44:50</td>
<td>239</td>
<td>6:40:00</td>
<td>6:45:00</td>
<td>252.2</td>
</tr>
<tr>
<td></td>
<td>7:00:58</td>
<td>7:05:05</td>
<td>247</td>
<td>7:00:00</td>
<td>7:05:00</td>
<td>294.2</td>
</tr>
<tr>
<td></td>
<td>7:23:06</td>
<td>7:27:57</td>
<td>291</td>
<td>7:25:00</td>
<td>7:30:00</td>
<td>325.1</td>
</tr>
<tr>
<td></td>
<td>7:40:53</td>
<td>7:49:29</td>
<td>514</td>
<td>7:45:00</td>
<td>7:50:00</td>
<td>504.4</td>
</tr>
<tr>
<td></td>
<td>7:57:57</td>
<td>8:06:58</td>
<td>481</td>
<td>8:00:00</td>
<td>8:05:00</td>
<td>505.1</td>
</tr>
<tr>
<td></td>
<td>8:22:26</td>
<td>8:27:59</td>
<td>353</td>
<td>8:25:00</td>
<td>8:30:00</td>
<td>390.4</td>
</tr>
<tr>
<td></td>
<td>8:40:27</td>
<td>8:44:25</td>
<td>238</td>
<td>8:40:00</td>
<td>8:45:00</td>
<td>281.6</td>
</tr>
<tr>
<td></td>
<td>8:59:27</td>
<td>9:04:03</td>
<td>246</td>
<td>9:00:00</td>
<td>9:05:00</td>
<td>256.0</td>
</tr>
<tr>
<td>MAPE</td>
<td>8.5%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3.6 Observed and simulated travel time of northbound I-405
3.6. CONCLUDING REMARKS

This chapter proposes a calibration and validation procedure for microscopic simulation models. While previous studies focused mostly on driving behavior model calibration to study a section of freeway, this study provides a general scheme of model calibration and validation for network-level simulation, responding to the extended use of microscopic simulation models. Such extension requires more systematic approach in model calibration since various model components are included in the process. The proposed procedure is demonstrated via a case network that involves multiple stages, and the calibrated model shows reasonable performance in replicating the observed flow condition.

In this chapter, various components of models were addressed in the model calibration process; however, we used the default route choice model in PARAMICS because of the interaction between the route choice model and OD estimation problem. In the network level model calibration/validation process, the inter-relationship between route choice and OD estimation make the problems complicated unless one of them is externally determined. This will be one of topics to be further studied in micro-simulation calibration/validation process.

Another further topic worthy of study is the development of an automated and systematic tool for microscopic simulation model calibration/validation by incorporating optimization-based model calibration methods within the proposed multiple stage approach.

3.7 REFERENCES

35


Chapter 4 On-line simulation

4.1 FRAMEWORK OF ON-LINE SIMULATION IN PARAMICS

The goal of this sub-task of the project is to enhance the capabilities of the PARAMICS simulation model through API programming for the on-line simulation application. The framework of the on-line system to be implemented can be illustrated in Figure 4.1. The real-world loop detector data are inputs of the system.

Data of those loop detectors located at cordon points of the network are queried from the database. These raw data are processed in order to obtain lane-based speed estimation and aggregated volume, occupancy data at the detector station. The minimum aggregation cycle is equal to the time interval of loop detector data from the field, which is 30 seconds in California. A typical aggregation cycle is 5 minutes or 15 minutes.

![Diagram of framework](image)

Figure 4.1 Framework of the implemented on-line system in PARAMICS

If the system is working under prediction mode, the future traffic condition at cordon points of the network will be predicted using the aggregated loop data of last several time intervals. The Kalman filtering based traffic prediction is applied. A forecast lead time can be specified as an input of the prediction, which represent how long in the future the traffic condition is predicted. If the system is working under the responsive mode, this prediction process is ignored. The traffic flow data at cordon points of the network are then output to the dynamic OD estimation module, that performs the estimation of an OD matrix based on the data of the current time interval. The current dynamic OD estimation
module is a simple method that can only handle a freeway network. An advanced dynamic OD estimation method, which may needs a reference demand pattern as its input, can be inserted here for the capability enhancements of the on-line simulation system. Currently, there are some on-going projects focusing on this study.

After the acquisition of the OD matrix, the demand loading module employs releasing vehicles from its original zone to the network. A composite model is implemented in order to release vehicles with a certain headway distribution, i.e. shifted negative exponential distribution under very low traffic flow condition, normal distribution with the mean of a constant headway under congested traffic condition, and the combination of two distributions under other conditions.

As a result, PARAMICS is connected with the real-world loop data directly and can thus on-line simulate traffic. Some demand errors may cause more congested or less congested traffic condition in the simulation. A consistency checking module is used to keep the simulation as similar as that in the real world. The real-world loop detector data at measurement points of the network are involved in this process to detect the difference between the real world and the simulation. Due to the current theoretical limitation on consistency checking, the current project leaves this topic for the future.

The following sections of this chapter will talk about the details how we implement this on-line simulation system.

4.2 LOOP DETECTOR DATA PROCESSING

The objectives of the processing of raw loop detector data are to obtain: (1) the lane-based speed estimation; (2) the aggregation of volume and occupancy data of last several time intervals. These data will further be used to OD estimation and demand loading.

The loop detector data stored at the ATMS testbed database are obtained from the Front End Processor (FEP) of District 12 of Caltrans. These data are raw data and not processed by the ATMS system of Caltrans. A script language is used to process these data in order to store them to the Oracle database. These data can be queried for the research use inside our local network.

An example of the data we query from a typical mainline loop detector station is:

05/22/2001 06:00:00, 1201203, 5, 5, 1, 10, 0, 1, 9, 0, 2, 13, 0, 1, 7, 0, 0, 0, 0

which corresponds to the format:

DATE TIME, VDS_ID, # of LANES, # of LOOPS, LANE_1_VOL, LANE_1_OCC, LANE_1_STATUS, LANE_2_VOL, LANE_2_OCC, LANE_2_STATUS, ...., LANE_n_VOL, LANE_n_OCC, LANE_n_STATUS
“VDS_ID” represents vehicle detector station index. A detector station can have several loop detectors, each of which covers one lane. “# of LANES” represents the total number of lanes at the loop detector station. “# of LOOPS” represents the total number of installed loop detectors at the station. “LANE_1” represents the leftmost lane. The actual occupancy value of a lane is equal to the LANE_1_OCC / 900 because the scan rate of loop detectors is 30 Hz and the time interval of these data is 30 second. “LANE_1_STATUS” represents the status of the data, which shows “0” for good data and “1” for bad data.

The loop detector data from the field may suffer from a lot of problems. The reasons might be the communication, noises, and sensitivity of loops. The quality of the data cannot be detected. The data processing module has the following features:

1. Analyzing the existence of loop data
2. Filtering those data with poor quality
3. Aggregating volume and occupancy data at a certain aggregation cycle, typically, 5 minutes or 15 minutes
4. Estimating the average lane-based speed based on aggregated volume and occupancy data

The estimation of speed is based on the assumption of an effective vehicle length. For lane \( i \) of loop detector station \( k \), the average speed of time interval \( t \) is defined as

\[
v_{ij}(t) = \frac{q_{ij}(t)}{o_{ij}(t)} \times \frac{g_{ij}}{g_{ij}}
\]

(4.1)

where,
- \( q_{ij} \) = traffic count
- \( o_{ij} \) = occupancy
- \( g_{ij} \) = average effective vehicle length

The average effective vehicle length needs to be estimated before the operation of the algorithm. We take the default values of effective vehicle lengths from the ATMS system of District 7 and 12 in our algorithm. These default values are:

1. Leftmost lane: 18 feet
2. Middle lanes: 22 feet
3. Rightmost lane: 25 feet

4.3 TRAFFIC FLOW PREDICTION

If the system is working under prediction mode, data of those loop detectors located at cordon points of the network are outputted to the traffic flow prediction module for the prediction of the future traffic condition based on the aggregated loop data of last several time intervals. If the system is working under the responsive mode, this prediction process is bypassed.
A typical OD estimation & prediction method combines the estimation and prediction of OD together using Kalman filtering and Generalized Least Square (GLS) approach (Ashok et al, 1996). The proposed method here is to predict the future traffic flows at cordon points of network first and then estimate the OD matrix based on these predicted traffic flows.

The traffic flow prediction involves the prediction of traffic counts at loop detector stations of upstream of mainline freeway, on-ramps and off-ramps based on loop data of previous time periods of the same day.

4.3.1 Introduction

According to loop data, we can only know the traffic counts of last several time intervals. In order to reflect traffic counts of the current running time interval (or, next time interval), we have to use traffic flow prediction techniques. A number of traffic flow predictive models have been developed and proposed, including historic profile approaches (Shbalo, S., Bhat, et al 1992, Kreer, J. B., 1975, Hoffmann, G. and J. Janko, 1990, Jeffrey, D. J., 1987), non-parametric regression (Seungjae Lee, et al, 1998, Davis, G. A. and Nihan, N. L., 1991), time-series (Moorby, C. X. and Ratcliffe, B. G., 1988), neural networks (Baht Abdulhalil, et al, 1998, Jinsoo You and Tachangho, 1998), Kalman filtering (Young-Ihn Lee, Chan Young Choi, 1998), traffic simulation and Dynamic Traffic Assignment models.

4.3.2 Kalman Filtering based prediction method

The Kalman filter, proposed by Kalman in 1960, provides an efficient least square based solution to the estimate of the state vector $x$ of a discrete time controlled process. Kalman filtering has shown good performance in a lot of real-time applications. The Kalman filtering based prediction method is applied in this study.

The traffic flow at a detection station is the state variable for the Kalman filtering process. We can establish the state equation as follows:

$$ x_t = x_{t-1} + w_t + w_{t-1} $$

(4.2)

where $x_t$ is the traffic flow at time interval $t$, $w_t$ is the state noise, which has a normal distribution with zero mean and a variance of $Q_t$, $u_t$ is the predicted traffic flow increment based on the last several traffic flow data ($x_{t-n}, x_{t-n+1}$) using the linear regression approach, which has the objective to fit a linear relationship:

$$ x_t = a + b \cdot t $$

(4.3)

where $t$ varies from 1 to $n$, $a$ is the intercept of the of the line with the vertical axis and $b$ is the slope of the line. The least square solution for this regression problem is
\[ u_k = \frac{n \sum (x_{k-1} + v_k) + \sum x_{k-1} \cdot \sum x_{k-1}}{n \cdot (\sum x^2) - \sum x_{k-1}^2} \]  \tag{4.4}

The value of \( b \) is actually the predicted change of traffic flow at the detector station \( u_k \).

Let \( x_k \) denote the observation of traffic flow at time interval \( k \), which is obtained from the real-world loop detector. The measurement equation associated with the state variable \( x_k \) is given by:

\[ z_k = x_k + v_k \]  \tag{4.1}

where \( v_k \) denotes the measurement error, which has a normal distribution with zero mean and a variance of \( R_k \).

In many applications, the state noise covariance \( Q \) and measurement noise covariance \( R \) are often assumed to be constants, which are usually determined prior to the operation of the filter. \( R \) is generally determined based on off-line sample measurements. However, \( Q \) is more difficult to be determined because the process state is typically not observable. The methods described in Section 5.2.3 are used to the online estimation of \( R \) and \( Q \).

4.3.3 Solution to the Kalman filter based prediction

Let \( \hat{x}_k \) and \( P_k \) denote a priori state estimation and its estimation variance at time step \( k \), \( \bar{x}_k \) and \( \bar{P}_k \) denote the a posteriori state estimate and its estimation covariance at time step \( k \), and \( K_k \) denotes Kalman gain. The solution to the above Kalman filter problem is shown as follows:

Step 1: Initialization

\[ \bar{x}_0 = E[x_0], \quad \bar{P}_0 = E[(x_0 - \bar{x}_0)^2] \]  \tag{4.6}

Step 2: State propagation

\[ \bar{x}_k = \bar{x}_{k-1} + x_k \]  \tag{4.7}
\[ \bar{P}_k = \bar{P}_{k-1} + \Omega_{k-1} \]  \tag{4.8}

Step 3: Kalman gain calculation:

\[ K_k = \bar{P}_k (\bar{P}_k + R_k)^{-1} \]  \tag{4.9}

Step 4: State estimation

\[ \hat{x}_k = \bar{x}_k + K_k [z_k - \bar{z}_k] \]  \tag{4.10}

42
\[ \hat{P}_t = \hat{P}_1 - K_P \hat{P}_t \]  \hspace{1cm} (4.11)

Step 5: State prediction
If there is only one-step ahead prediction, the predicted traffic flow is:
\[ \hat{y}_{k+1} = \hat{P}_t + u_{k+1} \]  \hspace{1cm} (4.12)

where \( u_{k+1} \) is calculated based on Equation 4.4. The prediction variance is:
\[ \hat{P} = \hat{P}_t + Q_t \]  \hspace{1cm} (4.13)

If the prediction is for several time intervals ahead, the predicted traffic flow will be used as the observation for a re-iteration prediction process (i.e. Step 5, Step 3, and Step 4).

4.4 OD ESTIMATION MODEL

Traffic counts at all cordon points of the study network are needed for OD estimation. These cordon points include the upstream end of freeway, downstream end of freeway, all on-ramps and all off-ramps.

4.4.1 Volume conservation

The freeway system has the following input flows:
(1) traffic flow at the upstream end of freeway
(2) traffic flow at each ramp

Since there are correspondent loops located at on-ramps and the upstream end of freeway, the sum of entrance traffic counts is:
\[ \text{In}_\text{-Flow}(t) = \text{Up}_\text{-Flow}(t) + \sum \text{Ramp}_\text{-Flow}(t, i) \]  \hspace{1cm} (4.13)

where \( i \) is the index of ramps, \( \text{In}_\text{-Flow}(t) \) is the traffic count of time interval \( t \) at the upstream end of freeway, \( \text{Ramp}_\text{-Flow}(t, i) \) is the traffic count of time interval \( t \) on ramp \( i \). If there is no congestion at the freeway origin locations, \( \text{In}_\text{-flow}(t) \) is the demand entering the freeway section.

The system has the following outflows:
(1) traffic flow at each off-ramp
(2) traffic flow at the downstream end of freeway

Since there are correspondent loops located at off-ramps and the downstream end of freeway, the sum of exit traffic counts is:
\[ \text{Out}_\text{-Flow}(t) = \text{Down}_\text{-Flow}(t) + \sum \text{Offramp}_\text{-Flow}(t, i) \]  \hspace{1cm} (4.14)
The principle of the technique of OD estimation is:

\[ \text{In}_i \text{Flow}(t) - \text{Out}_j \text{Flow}(t) \]  

(4.15)

Assuming the error caused by loop detector detection can be ignored, this formula is true if the traffic is stable. Under this condition, we can further estimate each element in the OD table through distributing off-ramp traffic to upstream origins/zones.

4.4.2 Model for distributing off-ramp traffic to upstream zones

This model is originally from FREQ model, which is based on the intuitive proportionality scheme.

\[ OD_{ij} = D_j \cdot \sum_{k \in A} \frac{Q_i}{Q_j} \]  

(4.16)

where A is a set that includes all upstream zones of zone j (i.e., all on-ramps and the upstream end of freeway); D_j is the total off-ramp traffic flow (or, exit traffic flow from freeway); Q_j is the total flow from zone i to freeway mainline (or, entrance traffic flow to freeway); OD_{ij} is the demand from origin i to j.

4.5 DEMAND LOADING / VEHICLE RELEASING

PARAMICS release vehicles to the links whose midpoint lies within origin zones. The location that vehicles may be released can be any point on the link. This simulates the parked vehicles moving away from the kerb, out of driveways and emerging from unmodelled side streets.

However, for a network with freeways and arterials, there are some internal zones and some cordon points, located at the boundary of freeways and arterials. The vehicle release pattern of PARAMICS can only simulate the vehicle release pattern from internal zones. How to release vehicles from a cordon point of a network is our interest. The study on time headway distribution was related to this topic.

4.5.1 Random model

When the traffic is not congested, or has a low density, Poisson count distribution can be used to describe the distribution of the number of time periods, which contain different flow level. The negative exponential distribution can be further derived from the Poisson count distribution for the description of the number of individual time headways in various time headway intervals. Its probability distribution function is:

\[ P(k \leq t) = 1 - e^{-\lambda t} \text{ or} \]

\[ P(k > t) = e^{-\lambda t} \]  

(4.17)
where $t$ is a time headway value, $\bar{t}$ is the average time headway or the reciprocal of the departure rate (unit: veh/sec). Since $0 \leq P(\bar{t} \geq t) \leq 1$, we can assume $P(\bar{t} \geq t)$ is a random number ranging from 0 to 1. Then the corresponding time headway of the random number is:

$$t = \ln(X)^* \bar{t} \tag{4.18}$$

where $X = P(\bar{t} \geq t)$, which is a random number uniformly distributed between 0 and 1. Therefore, the inter-departure time from an origin zone is:

$$t_{i+1} - t_i = -\ln(X)^* \bar{t} \tag{4.19}$$

where $t_{i+1}$ and $t_i$ are the departure time of two vehicles. The negative exponential distribution has the inherent feature that the smallest headways are most likely occur. However, it is impossible to have such a low time headway under the low-density traffic. The shifted negative exponential distribution is thus employed. Its probability distribution function is:

$$P(\bar{t} \geq t) = e^{-\frac{t-a}{\bar{t}}} \tag{4.20}$$

where $a$ is the minimum time headway under the low-density traffic. The inter-departure time is:

$$t_{i+1} - t_i = \alpha - f(\bar{t} = \alpha)^* (t - \alpha) \tag{4.21}$$

4.5.2 Constant headway model

If the traffic is congested, vehicles are traveling under car-following regime. Vehicles departing from an origin zone can be thought to have a constant headway. The inter-departure time between two consecutive vehicles is the reciprocal of the departure rate, i.e.,

$$t_{i+1} - t_i = \frac{1}{\bar{t}} \tag{4.22}$$

The normal distribution is used to describe the condition that drivers attempt to drive at a constant time headway but the actual headway varies around the expected headway because of drivers’ perception error.

4.5.3 Composite model

The random model is suitable for the very low traffic flow conditions and the constant headway model is suitable for the very high traffic flow conditions. It is hard to identify if the traffic condition is under random state or constant headway state. We can thus integrate above two models to a new model via the use of a distribution factor, which has
a value between 0 and 1. The distribution factor determines the model selected for the
departure of vehicles. For example, 0.3 of the distribution factor indicates that 30% of
vehicles departs according to constant headway model and other 70% according to the
random model.

\[ t_{rel} = t_i \cdot \frac{1}{\alpha - \frac{\lambda}{\alpha} (X - i)} \]
\[ \Phi > df \]
\[ \Phi < df \]

(4.23)

where \( F \) is a random number uniformly distributed between 0 and 1, \( df \) is the distribution
factor, which can be determined based on the following equation:

\[ df = 1.5 / \dot{i} \]

(4.24)

where 1.5 second is regarded as the mean time headway when all vehicles are under very
high traffic flow conditions. \( \dot{i} \) can be obtained if we know the total number of vehicles
that will be released from an origin zone.

4.5.4 Random number generator

PARAMICS uses the Marsaglia random number generator. However, we take the
modified version of the random number generator proposed by Park & Miller to generate
random number for vehicle releasing. John Burton of G & A Technical Software, Inc. is
the author of this modified version of random number generator, which is also the
random number generator of Mitsim. The advantage of this generator is that it can run on
any machine with a 32-bit integer, without overflow.

The source code for this random number generator is shown below, in which “seed” is a
global parameter.

def double rand ( void )
{
    // Constants for linear congruential random number generator.
    const long int M = 2147483647; // M = modulus (2^31)
    const long int A = 48271;
    const long int Q = M / A;
    const long int R = M % A;

    seed = A * ( seed % Q ) + R * ( seed / Q );
    seed = ( seed > 0 ) ? ( seed ) : ( seed + M );

    return (double) seed / (double) M ;
}

4.6 REFERENCES


Brian L. Smith and Michael J. Demetsky Short-term Traffic Flow Prediction: Neural Network Approach, Transportation Research Record 1453, PP98-104


Seungjae Lee, Daehyon Kim, Juyoung Kim, Bumchul Cho, Comparison of Models for Predict in Short-term Travel Speeds, Proceedings of 5th World Congress on ITS, Korea, 1998


Young-Ihn Lee, Chan Young Choi, Development of Link Travel Time Prediction Algorithm for Urban Expressway, Proceedings of 5th World Congress on ITS, Korea, 1998.
Chapter 5. Data Fusion for Better Travel Time Estimation

5.1 INTRODUCTION

In current traffic surveillance systems, point detection systems, such as inductance loops, microwave detectors, and ultrasonic detectors, have been used for traffic data collection. Among these detection technologies, most agencies rely on inductance loop detectors, mostly single loops. Some double loop detectors have been installed because double loops can provide accurate point speed estimation besides the traditional traffic count and occupancy outputs. Although most of the current traffic control and management systems, such as ramp metering and incident detection, depend on these point-based data for operation, these point detection systems have limitation in providing section travel times that are preferable for other Advanced Transportation Management and Information Systems (ATMIS) applications, such as the traffic information and route guidance systems. The methods estimating section travel time from single loop detector data involve the conversion of point volume and occupancy data to travel time based on traffic flow theory [1]. This estimation is inaccurate not only because of the conversion error but also because of the point detection system’s limitation in capturing area-wide traffic dynamics.

Recent advances in sensing technologies, such as Automatic Global Positioning Systems (GPS) [2], Vehicle Identification (AVI) [3], cellular phone positioning [4], and vehicle re-identification technology based on advanced loop detectors [5] or video imaging technologies [6], provide information on timestamps when vehicles pass predefined locations. In such circumstances, the travel time during a time interval can be calculated by averaging individual vehicles’ travel times from upstream to downstream. Currently, the GPS-based on-board navigation systems have been more and more installed to passenger cars, which will become potential probes if a suitable communication method is available. These new technologies are regarded as probe-based detection systems since they collect travel time information directly from probe vehicles that they trace or match. Even though these systems are expected to provide high fidelity traffic information when there are enough probe vehicles, these systems are insufficient yet to play a major role as main detection systems due to the lack of probe rates.

It seems inevitable to have multiple data sources for ATMIS applications. They will be either point detection data or probe vehicle data. In terms of the acquisition of section travel time from these data sources, the probe-based detection systems may suffer from the low frequency of probes or the situation that the probe vehicle travel time cannot reflect the travel times of all vehicles that traversed the section while the point detectors suffer from the inability in obtaining travel time data directly. In other words, each data source provides traffic information with different characteristics, but has its own limitation. However, these different sources of data could be complementary each other, and it will be possible to achieve better measures when they are considered.
simultaneously. The main interest of this paper is how to use both probe-based data source and point-based data source for better section travel time estimation.

Data fusion is a technique to combine data from several sources in order to provide comprehensive and accurate information. Several data fusion techniques have been developed and most of them are derived from physical models, feature-based inference models, and cognitive-based models [7]. Physical models estimate the classification and identity of an object by matching modeled or pre-stored object signatures to the observed data. An example of physical models is Kalman filtering, which is most often used to positional estimation and target tracking applications [8, 9]. Feature-based inference models classify or identify objects by mapping data, such as statistical knowledge about an object or recognition of its features. There are many feature-based algorithms, such as Bayesian, Dempster-Shafer, artificial neural network, entropy method, voting method, figure of merit, etc. Cognitive-based models emulate and automate the decision-making processes used by human analysts. Examples of this type of models are knowledge-based expert system and fuzzy logic method.

This chapter proposes a data fusion algorithm for section travel time estimation that fuses both point detector data and area-wide probe data using the Adaptive Kalman Filtering (AKF) technique. This paper is organized as follows. Section 2 describes the methodology of the proposed AKF based section travel time fusion algorithm, which has the capability to dynamically estimate the variances of state noise and measurement noise. In Section 3, the proposed algorithm is implemented in a microscopic simulation environment and the algorithm is evaluated under both recurrent and non-recurrent scenarios. Finally, the concluding remarks are given.

5.2. METHODOLOGY

5.2.1. Kalman Filter

The Kalman filter, proposed by Kalman in 1960, provides an efficient least square based solution to the estimate of the state vector x of a discrete time controlled process that can be described by the linear stochastic difference equation [10]:

\[ x_k = \Phi x_{k-1} + u_k + w_{k-1} \]  

(5.1)

with a measurement vector y that is

\[ y_k = H x_k + v_k \]  

(5.2)

\( w_k \) and \( v_k \) are the state and measurement noises, respectively. They are assumed to be uncorrelated white noises with normal distributions:

\[ p(x) \sim (0, Q) \]  

(5.3)
\[ p(x) = (0, R) \]  

(5.4)

In many applications, the state noise covariance \( Q \) and measurement noise covariance \( R \) are often assumed to be constants, which are usually determined prior to the operation of the filter. \( R \) is generally determined based on off-line sample measurements. However, \( Q \) is more difficult to be determined because the process state is typically not observable.

In Equation 5.1, \( \Phi \) is a transition matrix that connects the state at the previous time step \( k-1 \) with the current time step \( k \). \( u \) is an optional control input to the state \( x \). In Equation 5.2, matrix \( H \) relates the state to the measurement \( y \). In practice, \( \gamma \), \( u \) and \( H \) might be time-variant. If \( Q \) and \( R \) are exactly known, the solution to the Kalman filter problem is:

\[ \bar{x}_k = \Phi_{k-1,k} \bar{x}_{k-1} + u_k \]  

(5.5)

\[ P_k = \Phi_{k-1,k} P_{k-1} \Phi^T_{k-1,k} + Q_{k-1} \]  

(5.6)

\[ K_k = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1} \]  

(5.7)

\[ \hat{x}_k = \bar{x}_k + K_k (y_k - H_k \bar{x}_k) \]  

(5.8)

\[ P_k = P_k - K_k H_k P_k \]  

(5.9)

where \( \bar{x}_k \) and \( P_k \) are defined as a priori state estimation and its estimation covariance at time step \( k \). \( \hat{x}_k \) and \( P_k \) are defined as the a posteriori state estimate and its estimation covariance at time step \( k \). \( K_k \) is defined as Kalman gain.

Equations 5.5 and 5.6 propagate the current state and error covariance estimates to obtain a priori state estimation for the next time step. Equations 5.8 and 5.9 are used to incorporate the measurement into the a priori state estimation in order to obtain an improved a posteriori estimate.

### 5.2.2 Application of Kalman Filter to the Section Travel Time Estimation

Kalman filtering has been used to solve some traffic problems, such as the dynamic estimation of traffic density [11, 12], freeway OD demand matrices [13], the prediction of traffic volume and travel time [14, 15, 16]. This paper uses Kalman filter to associate two data sources, point detector data and probe vehicle data. While both traffic count and occupancy data are provided from the point detectors, the section travel time can be directly sampled from probe vehicles. The objective is to obtain a better estimation of section travel time. This section provides an overall description of the model.

#### 5.2.2.1 Travel Time from Section Density

The model we are proposing here is based on the conservation or continuity equation. This equation shows the same form as in fluid flow, and the solution of the conservation equation in traffic flow was first proposed by Lighthill and Whitham[17] and Richards [18]. The fluid conservation equation characterizes compressible flow, that is,
\[
\frac{\partial q}{\partial x} + \frac{\partial k}{\partial t} = 0 \quad \text{(or traffic generation rate)} \tag{5.10}
\]

where,

\( q \) = flow (vehicles/hour)

\( k \) = density (vehicles/mile)

\( x \) = location

\( t \) = time

If the speed of such traffic fluids is \( v \), we have the following basic identity:

\[
q = k \cdot v \tag{5.11}
\]

From this equation, travel time can be derived as a function of section density and traffic flow rate. For a section with detector stations at its boundaries, the traffic flow passing this section (i.e., \( q \)) can be estimated as a linear function of the traffic flow passing the upstream detector station \( q_u \) and downstream detector station \( q_d \). Assuming that the traffic inside the section is homogeneous, an intuitive estimate of the section travel time is:

\[
\frac{\Delta x}{v} = \frac{\Delta t}{\frac{\Delta x}{q}} = \frac{\Delta x}{\Delta t} = \frac{k}{a \cdot q_u + (1-a) \cdot q_d} \tag{5.12}
\]

where,

\( \frac{\Delta x}{v} \) = section-density-based travel time

\( \Delta x \) = length of the section

\( a \) = a smooth parameter

Equation 5.12 actually establishes the relationship among the section travel time, section density, and passing flow of section measured by two point detectors at boundaries of the section. As in Equation 5.12, the section travel time can be estimated as long as accurate measures of section density.

Figure 5.1 shows a typical section of urban freeway, which includes one on-ramp and one off-ramp. Let \( N(t) \) denote the number of vehicles within a section at time \( t \) and let \( N_u(t) \), \( N_d(t) \), represent the number of vehicles passing the upstream and downstream detector stations during time \( (t-1, t) \). \( N_u(t) \) and \( N_d(t) \) are the traffic counts entering and exiting the freeway section from on-ramps and off-ramps, respectively. If the count information is precise, then the sequence \( N(t) \) can be represented by:

\[
N(t) = N(t-1) + (N_u(t) + N_d(t) - [N_u(t-1) + N_d(t-1)]) = N(t-1) + \Delta N(t) \tag{5.13}
\]

Let \( k(t) \) denote the section density at time \( t \). \( k(t) \) can be estimated based on the relationship between \( N(t) \) and \( k(t) \):

\[
k(t) = k(t-1) + \frac{\Delta t}{L \cdot \Delta x} \cdot \Delta N(t) \tag{5.14}
\]

where,
L = number of lanes on mainline freeway
\( t \) = time interval of detector data aggregation

![Figure 5.1 A typical section of freeway](image)

The section density is a non-observable state variable. As a deterministic equation, the section density sequence may not be correctly or robustly estimated using Equation 5.14 if there are any input data errors. The worst situation is that this equation cannot converge and thus lose the ability to estimate the section density. A solution to this is to introduce a feedback control mechanism, such as Kalman filtering, to this estimation process.

5.2.2.2 Travel Time from Probe Vehicles

When probe vehicle data are available, the travel time during a time interval can be calculated by averaging individual vehicles’ travel times from upstream to downstream. That is, the average travel time \( \tau_{ud} \) is calculated based on the vehicles that arrived at the downstream station during a given time-step [19].

\[
\tau_{ud} = \frac{\sum (t_u^d - t_u^s)}{N}
\]  
(5.15)

where,

- \( N \) = number of sample vehicles arriving the downstream station
- \( t_u^d \) = time when vehicle \( n \) passes the downstream station \( d \)
- \( t_u^s \) = time when vehicle \( n \) passes the upstream station \( u \)

5.2.2.3 Design of Kalman Filter

We apply Kalman filtering to this travel time estimation process. We use both data sources: one from point detectors and the other from probe vehicles. Based on Equation 5.14 and Equation 5.12, the state and measurement equations of the Kalman filter are:

State equation:

\[ k(t) = k(t-1) + u(t) + w(t-1) \]  
(5.16)

Measurement equation:

\[ n(t) = H(t) * k(t) + v(t) \]  
(5.17)

where \( v(t) \) is the system noise, which is a zero-mean Gaussian white noise with the variance of \( Q \). \( v(t) \) is the measurement noise, i.e., the noise of section travel time.
estimates from probe vehicles. \(v(t)\) is a white noise with the variance of \(R\). In addition, according to Equation 5.14, we have:

\[
u(t) = \frac{\Delta \tau}{L \Delta \tau} \cdot \Delta N(t) \tag{5.18}\]

Based on Equation 5.12, we set \(a = 0.5\) in this study. Then we have:

\[
H(t) = \frac{\Delta \tau}{(q_x(t) + q_y(t))/2} \tag{5.19}
\]

Note in this proposed model, the transition matrix, \(F\), is a constant of 1. In addition, we use \(k\) instead of \(x\) and \(t\) instead of \(y\).

The solution of this Kalman filter is shown in Equation 5.5 to Equation 5.9. At each time step, the Kalman filter will output a posteriori state estimate of the section density, which can be further used to calculate the estimated section travel time using Equation 5.12.

5.2.3 Adaptive Kalman Filter: estimation of noise statistics

A well-known limitation of the application of Kalman filter is that it is hard to obtain the estimates of state noise covariance \(Q\) and measurement noise covariance \(R\), which might change with each time step or measurement in a real-world system. If \(R\) and \(Q\) cannot reasonably represent the features of the state and measurement noises, the performance of Kalman filtering will degrade. The inappropriate values of covariance matrices \(R\) and \(Q\) can cause the divergence of the optimal estimator of Kalman filter.

Accurate knowledge of noise covariance \(R\) and \(Q\) is critical for the successful application of Kalman filter to those processes, such as the section travel time estimation process, with time-variant noise statistics. Several researchers have studied on the so-called Adaptive Kalman Filter (AKF), which incorporates the on-line estimation of \(R\) and \(Q\) with the Kalman Filter process. The methods of AKF were classified into four categories, Bayesian, maximum likelihood, correlation, and covariance matching [20].

This paper uses a covariance matching method based on the research work done by Myers K.A. and Tapley B.D. [21]. The basic concept of this method is to match the estimated covariance with theoretical covariance.

5.2.3.1 Estimation of \(R\)

Based on Equation 5.17, the measurement noise \(v(t)\) cannot be determined because the true state vector \(k(t)\) is unknown. An intuitive approximation of the measurement noise at time step \(j\) is given by:

\[
\eta_j = v_j - H \tilde{k}_j \tag{5.20}
\]
where \( r_j \) is defined as the measurement noise sample, it is also known as the innovation or the measurement residual. The innovation sequence is a zero mean Gaussian white noise sequence. This is a necessary and sufficient condition for the optimality of a Kalman filter [22].

Assuming that the noise sample sequence \( r_j \) can be representative of the true measurement noise sequence \( v_j \), \( r_j \) and \( v_j \) should be independent and have the same distribution. Then the actual covariance of \( v_j \) can be approximated by the unbiased estimate of its sample covariance:

\[
\hat{\Sigma}_v = \frac{1}{N-1} \sum_{j=1}^{N} r_j v_j^T
\]  
(5.21)

where \( N \) is the number of measurement noise samples, which is selected empirically in order to provide a reasonable noise statistics based on the latest data.

Based on Equations 5.20 and 5.21, the expected value of \( \hat{\Sigma}_v \) is:

\[
E(\hat{\Sigma}_v) = \frac{1}{N} \sum_{j=1}^{N} H_j \hat{\Sigma}_{\hat{y}j} H_j^T + R
\]  
(5.22)

The unbiased estimation of \( R \) will be:

\[
\hat{R} = \frac{1}{N-1} \sum_{j=1}^{N} (r_j v_j^T - \frac{N-1}{N} H_j \hat{\Sigma}_{\hat{y}j} H_j^T)
\]  
(5.23)

5.2.3.2 Estimation of \( Q \)

Based on Equation 5.6, the state noise \( w(t) \) cannot be determined because the true state vectors \( k(t) \) and \( k(t-1) \) are unknown. The intuitive approximation of the state noise at time step \( j \) is given by:

\[
\eta_j = \hat{k}_j - \Phi_{j,j} \hat{k}_{j-1}
\]  
(5.24)

where \( \eta_j \) is defined as the state noise sample. It is a zero mean white noise. Assuming that the noise sample sequence \( \eta_j \) can be representative of the true state noise sequence \( \omega_j \), \( \omega_j \) and \( \eta_j \) should be independent and have the same distribution. Then the actual covariance of \( \omega_j \) can be approximated by the unbiased estimate of its sample covariance:

\[
\hat{\Sigma}_\omega = \frac{1}{N-1} \sum_{j=1}^{N} \eta_j \eta_j^T
\]  
(5.25)
where \( N \) is the number of measurement noise samples, which is selected empirically in order to provide a reasonable noise statistics based on the latest data. Following the same procedure in estimating \( R \), the unbiased estimation of \( Q \) will be:

\[
\hat{Q} = \frac{1}{N-1} \sum_{i=1}^{N} \left[ \Phi_{x,x,i} - \left( \frac{N-1}{N} \right) \Phi_{x,x,i-1} + \Phi_{x,u,i} - \Phi_{x,u,i-1} \right]
\]  

(5.26)

5.2.4 Summary of the AKF fusion algorithm

The proposed AKF based fusion algorithm can be summarized as follows. At each time step,

1. Calculating \( u(t) \) and \( H(t) \) based on the data of last time interval from point detector using Equations 5.18 and 5.19.
2. State propagation: calculating a priori estimate of \( k(t) \) and estimation covariance using Equations 5.5 and 5.6.
3. Estimating \( R \) using Equation 5.23 based on last \( N \) measurement noise samples.
4. Updating Kalman gain using Equation 5.7.
5. State estimation: calculating a posteriori estimate of \( k(t) \) and estimation covariance using Equations 5.8 and 5.9.
6. Estimating \( Q \) using Equation 5.26 based on last \( N \) state noise samples.
7. Calculating the section travel time based on Equation 5.12.

The proposed fusion algorithm can be executed without specifying initial estimates of \( R \) and \( Q \), although good initial values of them can make the algorithm perform better at the beginning of its operation.

5.3. EVALUATION

This study evaluates the proposed algorithms by comparing various travel time estimation methods. The evaluation is performed in a stretch of freeway using a microscopic traffic simulation model, PARAMICS (Version 4 Build 10) based on real world data. We consider detection errors as well as non-recurrent traffic congestion for more realistic evaluation.

5.3.1 Study Site

The study site is a six-mile stretch of northbound freeway I-405, between junctions of freeway I-5 and Culver Drive, in Orange County, California. The schematic representation of the study site is illustrated in Figure 5.2. The line across the freeway lanes represents mainline detector, whose location is shown on the bottom by its post-mile. As a major freeway linking Orange County to Los Angeles, this stretch of freeway experiences heavy traffic congestion during the morning peak hours, which is caused by heavy traffic flows from freeway SR-133 and Jeffery Dr. In this study, we only considered the section from Sand Canyon Dr to Jeffery Dr, which includes one on-ramp.
and off-ramp. This section features a lot of shockwaves during the morning peak hours because of the strong bottleneck point at Jeffery Dr.

![Study section](image)

Figure 5.2 Schematic figure of the study site

This network used in PARAMIC simulation has been calibrated based on previous simulation studies [23]. The time-dependent OD demands, estimated based on the real world traffic data of May 22, 2001, were used for this simulation study. The simulation period is from 6:30 to 9:00 AM.

5.3.2 Travel Time Estimation Methods

In the previous section, we described three ways of estimating section travel time: travel time estimation from section density, travel time estimation from probe vehicles, and travel time estimation fusing two data sources. In addition to these three methods, a benchmark travel time and a simple travel time estimation method from point detection data are considered for the comparison purpose.

5.3.2.1 Benchmark Travel Time

Within a discrete time and space domain, the representative travel time can be defined as a mean travel time within the closed area defined by the time \((t-1 \text{ and } t)\) and space \((x_a \text{ and } x_b)\), as shown in Figure 5.3.

The true space-mean speed for vehicles within the closed area is equal to the total travel distance divided by the total travel time of all vehicles in the closed area [24]. An unbiased estimate of the space-mean speed is:

\[
\bar{v} = \frac{\sum_{n=1}^{N} \left( \text{min}(x_n, x_b) - \text{max}(x_n, x_a) \right)}{\sum_{n=1}^{N} \left[ \text{min}(t_n, t_b) - \text{max}(t_n, t_a) \right]},
\]

(5.27)

where,

- \(N\) = number of vehicles traversing the section during the time interval
- \(x_n\) = position of vehicle \(n\) at time \(t\)
\[ x_u = \text{position of the upstream loop detector station u} \]
\[ x_d = \text{position of the downstream loop detector station d} \]
\[ t_d = \text{time when vehicle n passes the downstream station d} \]
\[ t_u = \text{time when vehicle n passes the upstream station u} \]

Figure 5.3 A temporal and spatial illustration of section travel time

The average section travel time can then be estimated.

\[ t_s = \frac{\Delta x}{v} \tag{5.28} \]

where,
\[ t_s = \text{average section travel time} \]
\[ \Delta x = \text{length of the section, i.e., } x_d - x_u \]

Such an average section travel time can be considered as a “true” mean travel time that represents the section. In this study, this section travel time is used as a benchmark travel time.

5.3.2.2 Section Travel Time from Point Detector Data

The general method to estimate section travel time from loop detector data is based on average speed estimates at the boundary detector stations. The average speed by lane can be obtained directly from double loops or estimated from single loops \([25, 26]\). Based on these lane-based speeds, the station speed is generally defined as the weighted average of lane speeds.

\[ v_s(t) = \frac{\sum \left( q_{x_d}(t) \times v_{x_d}(t) \right)}{\sum q_{x_d}(t)} \tag{5.29} \]

where,
\[ L = \text{number of lanes at loop detector station k} \]
\[ q_{x_d}(t) = \text{traffic count at lane i of upstream detector station k} \]
\( v_{k,i} = \text{traffic count at lane } i \text{ of downstream detector station } k \)

Then the section travel time can be estimated \[ v_p = \frac{1}{2} \left( \frac{v_u + v_d}{v_u} \right) \] \hspace{1cm} (5.30)

where,

- \( v_u \) = speed at upstream detector
- \( v_d \) = speed at downstream detector

In this study, we only consider the double loop case in order to avoid handling the speed estimation from single loop data.

5.3.2.3 Implementation of Travel Time Estimation Methods

We implement the proposed AKF fusion algorithm and other above-mentioned travel time estimation methods from double loop speed data, section density calculation, and probe vehicle data in a microscopic traffic simulation model, PARAMICS. These travel time estimates are compared with the benchmark travel time described above. These travel time estimation methods are summarized in Table 5.1.

<table>
<thead>
<tr>
<th>Method</th>
<th>Data source</th>
<th>Implementation Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Travel Time</td>
<td>Individual vehicle’s trajectory from simulation</td>
<td></td>
</tr>
<tr>
<td>Double Loop Speed-based Method</td>
<td>Speed data from double loop detectors</td>
<td>Equation 5.29, 5.30</td>
</tr>
<tr>
<td>Probe-based Method</td>
<td>Travel time from probe vehicles</td>
<td>Equation 5.15</td>
</tr>
<tr>
<td>Density-based Method</td>
<td>Traffic counts from loop detectors</td>
<td>Equation 5.14, 5.12</td>
</tr>
<tr>
<td>AKF-based Data Fusion Algorithm</td>
<td>(1) Traffic counts from loop detectors (2) Travel time from probe vehicles</td>
<td>Equation 5.15, 5.16, 5.17, 5.12</td>
</tr>
</tbody>
</table>

In the probe-based method, we do not consider individual vehicles’ travel time errors (introduced by sensing technologies) but their sampling errors. The performance of this probe-based method varies by sampling rates of probe vehicles. In this method, there may not have a probe within a time interval when the sampling rate is low. In such circumstances, the current time step’s travel time is assumed to be same as previous time step’s travel time.
In this study, we considered detection errors of point detectors. According to an evaluation report conducted by Texas Transportation Institute [28], single loop detectors can provide a typical 98% accuracy on loop count data and double loop detectors can provide a typical 96% accuracy on speed data. We understand this accuracy, \( d \), as the average abstract relative error of \( 1 - d \). Due to the lack of the knowledge about this error distribution, we assumed it as a uniform distribution varying from -4% to 4% with the abstract error mean of 2% and zero mean. It has the following properties of the probability density function:

\[
f(x) = \begin{cases} 
0 & \text{if } x < -(1 - \delta)^2, x > (1 - \delta)^2 \\
\frac{1}{(1 - \delta)^2} & \text{if } -(1 - \delta)^2 \leq x \leq (1 - \delta)^2 
\end{cases}
\]  

(5.31)

where, 
\( d \) = accuracy of loop count, which is 98% in this study

5.3.3 Performance Indices

Two error indices, Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE), are employed for the performance evaluation of section travel time methods. These two indices are defined as:

\[
RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (z_i - \hat{z}_i)^2}
\]

(5.32)

\[
MAPE = \frac{1}{N} \sum_{i=1}^{N} \left| \frac{z_i - \hat{z}_i}{z_i} \right|
\]

(5.33)

where, 
\( z_i \) = the true value of variable \( z \) at sampling point \( i \)
\( \hat{z}_i \) = the estimated value of variable \( z \) at sampling point \( i \)
\( N \) = total number of samples of variable \( z \)

5.3.4 Evaluation Scenarios

We considered the following four scenarios:

1. Recurrent congestion: without loop error
2. Incident: without loop error
3. Recurrent congestion: with 98% accuracy of loop count
4. Incident: with 98% accuracy of loop count

In the two incident scenarios, an incident was injected to the middle of the study section, as shown in Figure 5.3 (the location is close to the off-ramp at Jeffery Dr). The incident was assumed to block a lane for 10 minutes from 8:20 to 8:30 AM.
5.3.5 Evaluation Results

All simulations started from 6:30 AM and ended at 9:00 AM. The first 15 minutes of each simulation run is regarded as the warm-up period. Since the fusion algorithm needs some time to initialize and fine-tune parameters, its performance was compared with other algorithms between the time period from 7:00 to 9:00 AM.

The section travel time was estimated based on a 30-sec interval for all methods under all scenarios. The estimated travel times were compared with the benchmark section travel time in order to obtain the two performance measures, including MAPE and RMSE. Table 5.2 and 5.3 shows the performance of all evaluated algorithms under four scenarios in terms of MAPE and RMSE, respectively. Based these performance measures, we find that the fusion algorithm can generate a better section travel time estimate compared to all other evaluated estimation algorithms under all scenarios.

<table>
<thead>
<tr>
<th></th>
<th>Scenario 1</th>
<th>Scenario 2</th>
<th>Scenario 3</th>
<th>Scenario 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Double Loop Speed-based Method</td>
<td>24.3%</td>
<td>24.4%</td>
<td>22.0%</td>
<td>20.0%</td>
</tr>
<tr>
<td>Probe-based Method (5% probe rate)</td>
<td>15.3%</td>
<td>16.1%</td>
<td>15.3%</td>
<td>16.1%</td>
</tr>
<tr>
<td>Density-based Method</td>
<td>10.9%</td>
<td>7.6%</td>
<td>18.8%</td>
<td>16.3%</td>
</tr>
<tr>
<td>AKF Fusion Algorithm</td>
<td>7.2%</td>
<td>7.8%</td>
<td>7.9%</td>
<td>8.2%</td>
</tr>
</tbody>
</table>

Table 5.2 MAPE of various estimation algorithms under different scenarios

<table>
<thead>
<tr>
<th></th>
<th>Scenario 1</th>
<th>Scenario 2</th>
<th>Scenario 3</th>
<th>Scenario 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Double Loop Speed-based Method</td>
<td>34.1</td>
<td>43.7</td>
<td>32.2</td>
<td>33.1</td>
</tr>
<tr>
<td>Probe-based Method (5% probe rate)</td>
<td>25.0</td>
<td>34.1</td>
<td>25.0</td>
<td>34.1</td>
</tr>
<tr>
<td>Density-based Method</td>
<td>12.5</td>
<td>28.4</td>
<td>19.8</td>
<td>35.9</td>
</tr>
<tr>
<td>AKF Fusion Algorithm</td>
<td>11.5</td>
<td>20.8</td>
<td>10.7</td>
<td>18.6</td>
</tr>
</tbody>
</table>

Table 5.3 RMSE of various estimation algorithms under different scenarios

The density-based method shows better performance compared to probe-based and double loop detector based methods under ideal detection conditions, i.e. Scenario 1 and Scenario 2. Due to the involvement of detection errors in Scenario 3 and 4, the performance of the density-based method becomes worse, as illustrated in Table 5.2 and 5.3. But it still has a comparable performance with the probe-based method.

61
Based on the description of the proposed fusion algorithm, the density-based concept is used in formulating the state equation and the probe based section travel time estimate is used as the measurement. The fusion algorithm improves the estimation of the section density (and thus the estimation of the section travel time) through the incorporation of probe-based travel time data with point detector data. Compared to the probe-based and density-based algorithms, the fusion algorithm improves section travel time estimate significantly in terms of both MAPE and RMSE, as shown in Table 5.2 and 5.3.

Figure 5.4 Comparison of section travel time estimation using the probe-based method (5% sampling rate) and the AKF fusion algorithm under Scenario 3

Figure 5.5 Comparison of section travel time estimation using the probe-based method (5% sampling rate) and the AKF fusion algorithm under Scenario 4

Figures 5.4 and 5.5 further compare the section travel time estimation using the probe-based method and the fusion algorithm under Scenarios 3 and 4, respectively. Regardless
of the obvious incorrect section travel time estimates from the probe-based algorithm during the congestion period, the fusion algorithm can provide a better section travel time estimate for the whole study period. Therefore, the fusion algorithm effectively captures the variation of section travel time due to the integration of both loop count data from single loops and probe vehicle data.

The proposed AKF fusion algorithm works in a way like this: the state variable, i.e. section density, is estimated by the state equation, and then the state variable is further corrected based on the measurement data from probes and the Kalman gain, estimated based on noise variances and the current system estimation variance. The key point of the proposed fusion algorithm is its on-line estimation of state noise variance R and measurement noise variance Q. The estimated R and Q under Scenario 4 are shown in Figure 5.6. Since R is the variance of travel time estimation error from probes and has a high value, we used ln(R) instead of R in order to show it together with Q in a figure. This figure shows that the system model (i.e. state equation) becomes more and more accurate with the progression of the fusion process. During the incident period, the measurement noise variance Q keeps a small value, which represents that the system model is pretty accurate. However, the variance of the measurement noise becomes larger because the probe-based algorithm cannot provide a good section travel time estimate right after the occurrence of the incident, as illustrated in Figure 5.5.

Figure 5.6 The on-line estimation of noise variances of R and Q using the AKF fusion algorithm under scenario 4

The adaptive filtering mechanism of the proposed fusion algorithm provides an appropriate feedback to the estimation of the section density at every time step, which causes an optimized section density estimate. Figure 5.7 compares the section density estimation using the proposed fusion algorithm and the section density based algorithm under Scenario 4. The section density estimated by the fusion algorithm has a MAPE error of 5.3%. It is much better than the density estimate from the section density based algorithm that has a MAPE error as high as 16.9%. From this point of view, the proposed fusion algorithm has an accurate estimation of the section density and thus leads to the accurate section travel time estimation.
Figure 5.7 Comparison of the estimation of section density using the AKF fusion algorithm and the density-based method under Scenario 4

Figure 5.8 Section travel time estimation using the double loop speed-based method under Scenario 3

As a comparison, this study compared the fusion algorithm with the double loop based section travel time estimation algorithm. Based on Table 5.2 and 5.3, this double loop based algorithm performs the worst among all algorithms. As shown in Figure 5.8, the section travel time is underestimated by double loop data. This result is consistent with the current knowledge about the point detector, which provides a higher average point speed than the average [29], and thus causes a lower travel time estimate using Equation 5.30. Therefore, the estimation of section travel time using point detector data is not
robust and the estimation accuracy cannot be guaranteed because of various location specific issues, such as vehicle weaving, curve link, etc. This explains that point detectors cannot capture the actual traffic condition. It also implies that an accurate estimation of point speed using point detector data cannot be used to improve the provision of accurate area-wide traffic condition.

As shown in Figure 5.9, a larger sample size of probes improves the accuracy of the travel time estimate, although this accuracy improvement is not obvious after the sample rate is higher than 20%. However, this travel time, calculated using the arrival-based method, does not represent the benchmark travel time well. However, the proposed fusion algorithm significantly improve the estimate of section travel time through the integration of two data sources. It performs well even a small sample size of probes. For example, 1% probes can provide a 11.8% MAPE error which decreases to 8.4% with 3% probes. Figure 5.9 also shows that the fusion of loop data with a 10% sample size of probes can provide a highly stable accuracy of the section travel time estimate. A higher sample size cannot provide a more accurate travel time estimate. This implies that the proposed fusion algorithm has reached its maximal performance. It is because any model, including the measurement equation of the proposed AKF fusion algorithm, cannot perfectly capture the randomness of the traffic system.

![Figure 5.9 Comparison of section travel time estimation using the AKF fusion algorithm and the probe-based method with respect to the sampling rate of probes](image)

5.4 CONCLUDING REMARKS

This travel time data fusion problem is essentially a traffic congestion recognition and travel time estimation problem plus the traditional data fusion problem. According to the feature of available data sources, the section travel time can be estimated based on either limited sample of a probe-based data source or statistical data from a point detector data source. Each data source has different features on its data collection method and thus
shows different accuracy of section travel time estimation under different traffic conditions.

The proposed fusion algorithm is established based on traditional flow theory and Kalman filtering theory. It appropriately fuses the single loop detector data (i.e. traffic count data only) with probe vehicle data (i.e. sample section travel time data) in order to obtain a more accurate section travel time estimate. The state noise variance R and measurement noise variance Q of the Kalman filter are estimated dynamically by adapting to the real-time data. The proposed fusion algorithm was evaluated through the comparison with other algorithms, including the probe-based method, the density-based method and the double-loop speed based method, under both recurrent and incident scenarios. The fusion algorithm showed much better performance.

The detectors in the field are not perfect and each detector has its own error features. There is currently no enough knowledge to show what the detection errors are. This paper assumes loop detector errors are uniform distributed with certain accuracy. It may not be the case in the real world. The future work will be to test the proposed fusion algorithm using the data from real world. In addition, since the proposed algorithm can self start without parameter inputs and adapt to the real data for noise measurement, it is eligible for on-line applications.

The proposed data fusion algorithm is a generic algorithm. It can also handle different combinations of data sources, such as the fusion of single loop data with double loop speed data. It may be possible to use the single loop data only because we did not use the occupancy data of single loop detector outputs yet. This may need some investigations on the possibilities based on both simulated data and real world data. Also, the proposed fusion algorithm can only work with two data sources. Potentially, it can be applied to the multiple data sources.

5.5 REFERENCES