

GETRAM

Open Traffic Simulation Environment

AIMSUN

VERSION 4.2

User Manual

February 2004



Table of Contents

1.	AIMSUN: SYSTEM DESCRIPTION	1
1.1	INTRODUCTION	1
1.2	FUNCTIONAL CAPABILITIES	3
1.2.1	GETRAM v4.0 New Features	4
1.2.2	GETRAM v4.1 New Features	5
1.2.3	GETRAM v4.2 New Features	6
1.3	SYSTEM STRUCTURE.....	9
1.4	SIMULATION PROCESS	11
1.5	THE AIMSUN GRAPHICAL USER INTERFACE	13
1.5.1	The Command Menu	13
1.5.2	The Toolbar	17
1.5.3	The Status and Control Bar	17
1.6	INPUT DATA REQUIREMENTS	17
1.6.1	Network Layout.....	17
1.6.2	Traffic Demand Data.....	17
1.6.3	Traffic Control.....	17
1.6.4	Public Transport	17
2.	NETWORK MODELLING	19
2.1	NETWORK STRUCTURE	19
2.2	NODES	21
2.2.1	Junctions and Joins	21
2.2.2	Yellow Box Junction	22
2.2.3	Junction Dialog Windows	22
2.2.4	Turning Movements	23
2.3	SECTIONS	25
2.3.1	Polysections.....	25
2.3.2	Section Dialog Windows.....	26
2.3.3	Reserved Lanes.....	27
2.3.4	Solid Lines.....	28
2.4	CENTROIDS	29
2.4.1	Network Zones	29
2.4.2	Centroid Dialog Windows.....	30
2.5	NETWORK EQUIPMENT.....	32
2.5.1	Traffic Control Devices.....	32
2.5.2	Detectors.....	32
2.5.3	Variable Message Signs (VMS)	34
3.	TRAFFIC MODELLING.....	37
3.1	TRAFFIC DEMAND DATA.....	37
3.1.1	Vehicle Classification.....	37
3.1.2	Input Flows and Turning Proportions.....	37
3.1.3	O-D Matrices and Routes	39
3.2	TRAFFIC GENERATION	41
3.2.1	Exponential.....	41
3.2.2	Uniform	41
3.2.3	Normal.....	42
3.2.4	Constant.....	42
3.2.5	“ASAP”	42
3.2.6	External	42
3.2.7	Dealing with a fractional number of trips.....	42
3.3	VEHICLE ENTRANCE PROCESS.....	45
3.3.1	Function to determine whether the entrance is possible	45
3.3.2	Virtual Entrance Queues.....	47
3.4	VEHICLE MODELLING PARAMETERS	49

3.4.1	Vehicle Attributes.....	49
3.4.2	Local Parameters	51
3.4.3	Global modelling parameters.....	53
3.4.3.1	General Parameters	53
3.4.3.2	Car-Following Model	54
3.4.3.3	2-lanes Car-Following Parameters	54
3.4.3.4	Lane-changing Parameters	55
3.4.4	Other Global Parameters	55
3.5	MODELLING VEHICLE MOVEMENT	57
3.5.1	Car-Following Model	57
3.5.2	Estimation of leader's deceleration	58
3.5.2	Calculating the speed for a vehicle at a section	58
3.5.3	The 2-lanes car-following model.....	59
3.5.4	Modelling the influence of the Section Slope	60
3.5.5	Lane-Changing Model.....	60
3.5.6	Overtaking Manoeuvre.....	64
3.5.7	On-Ramp Model.....	65
3.5.8	Off-Ramp Model	66
3.5.9	Look Ahead Model.....	67
3.6	TRAFFIC INCIDENTS	71
4.	DYNAMIC TRAFFIC ASSIGNMENT	73
4.1	PATH DEFINITION.....	74
4.1.1	Network Representation	74
4.1.2	Link Cost Functions	75
4.1.2.1	Link Capacity	76
4.1.2.2	Initial Cost Function	78
4.1.2.3	Dynamic Cost Function.....	79
4.1.2.4	User-defined link cost functions.....	82
4.1.3	Shortest Path Algorithm	82
4.1.3.1	Initial Shortest Path	84
4.2	PATH SELECTION	86
4.2.1	User-defined Assignment	87
4.2.2	Route Choice Models	87
4.2.2.1	Fixed routes mode vs. Variable route mode	87
4.2.2.2	Binomial Model.....	91
4.2.2.3	Proportional	91
4.2.2.4	Multinomial Logit	94
4.2.2.5	C Logit.....	96
4.2.2.6	User-defined Route Choice Model	99
4.2.3	Determine the finite set of path in the decision process	99
4.2.4	Initial Assignment	100
4.2.5	En-Route Assignment.....	102
4.2.6	Entrance/Exit Section decision.....	102
4.3	PATH ANALYSIS TOOL.....	104
4.3.1	Path Analysis.....	104
4.3.1.1	Shortest Path Information.....	104
4.3.1.2	User-defined Path Information	105
4.3.1.3	Shortest Path Display	106
4.3.1.4	Initial Path Assignment Information	108
4.3.1.5	En-Route Path Assignment Information.....	109
4.3.2	Simulation Output	110
4.3.2.1	Path Information Output.....	110
4.3.2.2	Link Costs Output.....	110
4.3.2.3	Vehicle Information.....	112
4.3.2.4	Vehicle Colouring	112

5.	MODELLING TRAFFIC CONTROL AND MANAGEMENT	115
5.1	TRAFFIC SIGNAL CONTROL.....	115
5.1.1	Signal Groups and Phases	115
5.1.2	Displaying Signal Control at Junctions	116
5.1.2.1	Signal Groups	116
5.1.2.2	Control Plan.....	117
5.1.2.3	Traffic Lights of a Section.....	118
5.2	YIELD AND STOP SIGNS.....	121
5.2.1	Gap-Acceptance Model	121
5.2.2	Give Way Priority Definition	122
5.3	RAMP METERING.....	124
5.3.1	Green Time Metering	125
5.3.2	Flow Metering	126
5.3.3	Delay Metering.....	127
5.4	TRAFFIC MANAGEMENT WITH VMS's	129
5.4.1	Actions for a Traffic Result Based Simulation.....	130
5.4.2	Examples of Actions.....	132
5.4.3	Actions for a Route-Based Simulation	134
6.	PUBLIC TRANSPORT MODELLING.....	139
6.1	PUBLIC TRANSPORT LINE	139
6.1.1	Public Transport Route.....	139
6.1.2	Public Transport Stops	140
6.1.3	Reserved Public Transport Lanes	143
6.2	TIMETABLES.....	144
6.3	PUBLIC TRANSPORT VEHICLE'S MODELING	145
6.3.1	Vehicle Generation Model.....	146
6.3.2	Vehicle Movement Model.....	146
6.3.3	Stop Model	146
7.	SIMULATION EXPERIMENTS	148
7.1	LOADING SCENARIOS	148
7.1.1	Loading the Network.....	149
7.1.2	Loading the Traffic Demand Data.....	151
7.1.2.1	Loading a Traffic Result.....	152
7.1.2.2	Loading an O/D Matrix	154
7.1.3	Loading a Traffic Control Plan.....	155
7.1.4	Loading a Public Transport Plan	156
7.2	RUNNING THE SIMULATION.....	157
7.2.1	Parameters of the Experiment.....	157
7.2.2	Simulation Run Time	157
7.2.3	Running Modes	158
7.2.4	Stopping the Simulation	159
7.2.5	Simulation Buttons	159
7.2.6	Running AIMSUN from a DOS Command line.....	160
7.2.7	Running AIMSUN without GUI: AConsole	160
7.3	SAVING THE SIMULATION STATE.....	162
7.3.1	Save Concrete State.....	163
7.3.2	Save Average State.....	163
7.4	RUNNING DIFFERENT REPLICATIONS OF THE SIMULATION	165
7.4.1	Calculation of the number of replications	166
7.4.2	Replication Information Output.....	167
8.	GRAPHICAL USER INTERFACE	169
8.1	ANIMATION OF THE SIMULATION	169
8.1.1	Vehicle Detail	169
8.1.2	Smoother Animation	170

8.2	VIEW COMMAND MENU	171
8.2.1	Vehicle Colouring	171
8.2.2	Displaying Number of Vehicles	172
8.2.3	Displaying Vehicles Attributes.....	173
8.2.4	Show Objects.....	174
8.2.5	Show Speed Limits.....	175
8.2.6	Show Virtual Queues.....	176
8.2.7	Show Occupied Detectors	176
8.2.8	Background	176
8.2.9	Preferences	177
8.2.9.1	Colours	177
8.2.9.2	Backgrounds.....	180
8.2.9.3	Shapes.....	181
8.3	THE 'WINDOW' COMMAND MENU.....	182
8.3.1	Show Scenario Info	182
8.3.2	Show Legend.....	182
8.3.3	Show Log Window.....	183
8.3.4	Hide / Show Status Bar.....	183
8.3.5	Hide / Show Toolbar	183
8.4	CLICKABLE DETECTION.....	184
8.4.1	Enabling Clickable Detection Feature.....	184
8.4.2	Activating Clickable Detection Capability.....	184
8.4.3	Detection Events.....	185
8.4.3.1	Double Clicks	185
8.4.3.2	Single Clicks.....	186
8.4.3.3	Combining single and double clicks.....	187
8.4.3.4	Detection Events Log File	187
8.4.3.5	Detection events visualisation	188
9.	SIMULATION OUTPUTS.....	189
9.1	STATISTICAL SIMULATION RESULTS	189
9.1.1	Statistical Traffic Measures	189
9.1.2	Calculation of Traffic Statistics	193
9.1.2.1	System Statistics.....	193
9.1.2.2	Turning and Section Statistics	195
9.1.2.3	Streams Statistics.....	200
9.1.2.4	OD Statistics.....	202
9.1.2.5	Public Transport Statistics.....	207
9.1.3	Customisation of Statistical Output.....	209
9.1.4	Statistics File Reports	214
9.1.5	Statistics Current Reports.....	215
9.1.6	Statistics Current Graphics.....	217
9.2	OUTPUT DETECTION	220
9.2.1	Detection File Reports.....	220
9.2.2	Detection Current Reports	221
9.2.3	Detection Current Graphics.....	222
9.2.4	Detection Data Gathering.....	223
9.3	STORING SIMULATION RESULTS USING ODBC	225
9.3.1	Output Location.....	225
9.3.2	Creating a Data Source	225
9.4	RECORDED SIMULATION	228
10.	ENVIRONMENTAL MODELS.....	229
10.1	FUEL CONSUMPTION MODEL.....	229
10.2	POLLUTION EMISSION MODEL	232
11.	CALIBRATION AND VALIDATION OF AIMSUN MODELS	235

11.1 METHODOLOGY FOR BUILDING SIMULATION MODELS	235
11.2 THE VALIDATION PROCESS: BUILDING VALID AND CREDIBLE SIMULATION MODELS	238
11.3 SPECIFICS FOR THE VALIDATION OF TRAFFIC SIMULATION MODELS	241
11.3.1 Develop a traffic simulation model with high face validity with GETRAM	241
11.3.2 Test the assumptions of the model empirically.	242
11.4 STATISTICAL METHODS FOR MODEL VALIDATION	244
11.5 A CASE STUDY: THE I-35W FREEWAY IN MINNEAPOLIS.....	247
11.5.1 Test Site Description	247
11.5.2 Simulation Results.....	248
11.5.3 Model Validation.....	248
11.5.3.1 Detector 429	248
11.5.3.2 Paired T-Test and Confidence Interval for detector 429	248
11.5.3.3 Two Sample T-Test and Confidence Interval for detector 429	249
11.5.3.4 Regression Analysis for detector 429.....	249
11.5.3.5 How close the observed and simulated series are?	250
11.5.3.6 Detector 426	251
11.6 TUNNING VEHICLE MODELLING PARAMETERS IN AIMSUN	253
11.6.1 Influence of Global Parameters	253
11.6.2 Influence of Section Parameters	253
11.6.3 Influence of Vehicle Parameters.....	254
11.6.4 On-ramp Calibration.....	254
11.7 AIMSUN VALIDATION TOOL	255
11.7.1 Comparing two sets of data	255
11.7.2 How to add Real Data in Output Data Base	257
APPENDIX 1: DESCRIPTION OF PARAMETER'S FILES	259
APPENDIX 2: OUTPUT DATABASE DEFINITION.....	265
APPENDIX 3: CALIBRATION OF AIMSUN CAR-FOLLOWING MODEL ..	277
1. BACKGROUND	277
2. MODELING IN AIMSUN	278
3. MODEL CALIBRATION AND TESTING	279
3.1 Direct test of the car-following model.....	279
3.2 Testing the ability to reproduce macroscopic behaviour.....	281
REFERENCES.....	283

1. AIMSUN: SYSTEM DESCRIPTION

1.1 INTRODUCTION

AIMSUN (Advanced Interactive Microscopic Simulator for Urban and Non-Urban Networks), [BAR94], is a microscopic traffic simulator that can deal with different traffic networks: urban networks, freeways, highways, ring roads, arterial and any combination thereof. It has been designed and implemented as a tool for traffic analysis to help traffic engineers in the design and assessment of traffic systems. It has proven to be very useful for testing new traffic control systems and management policies, either based on traditional technologies or as implementation of Intelligent Transport Systems.

AIMSUN can be used for simulating Advanced Transport Telematic Applications. It can simulate adaptive traffic control systems such as SCOOT, C-Regelaar and Balance; vehicle actuated, control systems that give priority to public transport, Advanced Traffic Management Systems (using VMS, traffic calming strategies, ramp metering policies, etc), Vehicle Guidance Systems, Public Transport Vehicle Scheduling and Control Systems or applications aimed at estimating the environmental impact of pollutant emissions, and energy consumption.

AIMSUN follows a microscopic simulation approach. This means that the behaviour of each vehicle in the network is continuously modelled throughout the simulation time period while it travels through the traffic network, according to several vehicle behaviour models (e.g., car following, lane changing). AIMSUN is a combined discrete/continuous simulator. This means that there are some elements of the system (vehicles, detectors) whose states change continuously over simulated time, which is split into short fixed time intervals called simulation cycles. There are other elements (traffic signals, entrance points) whose states change discretely at specific points in simulation time. The system provides highly detailed modelling of the traffic network, it distinguishes between different types of vehicles and drivers, it enables a wide range of network geometries to be dealt with, and it can also model incidents, conflicting manoeuvres, etc. Most traffic equipment present in a real traffic network is also modelled in AIMSUN: traffic lights, traffic detectors, Variable Message Signs, ramp metering devices, etc.

The input data required by AIMSUN is a simulation scenario, and a set of simulation parameters that define the experiment. The scenario is composed of four types of data: network description, traffic control plans, traffic demand data and public transport plans. The network description contains information about the geometry of the network, turning movements, layout of links (or sections) and junctions and location of detectors along the network. The traffic control plans are composed of the description of phases and its duration for signal controlled junctions, the priority definition for unsigned junctions and the ramp-metering information. The traffic demand data may be defined in two different ways: 1) the traffic volumes at the input sections, the turning proportions and the initial state of the network; or 2) an O-D matrix, which is the number of trips going from every origin centroid to any destination one. In the first case, vehicles are distributed stochastically around the network according to the turning proportions at intersections, while in the second case, vehicles are assigned to specific routes from the origin to the destination. Different route selection modes are implemented: fixed or variable, static or dynamic. Finally, a public transport plan comprises the definition of bus lines (routes and bus stops) and the timetables for each line, including stop times.

Both traffic control plans and traffic demand data can be fixed or variable over time. The simulation parameters are fixed values that describe the experiment (simulation time, warm-up period, statistics intervals, etc) and some variable parameters used to calibrate the models (reaction times, lane changing zones, etc).

The outputs provided by AIMSUN are a continuous animated graphical representation of the traffic network performance, both in 2D and 3D, statistical output data (flow, speed, journey times, delays, stops), and data gathered by the simulated detectors (counts, occupancy, speed). Both the statistical and detection data can be graphical (plots) or numerical, the latter can be stored in ASCII files or database (Access or ODBC).

The traffic network state can be stored at any given moment of the simulation so that it can be retrieved and used as an initial state in future simulation experiments. This state can also be obtained from other simulation or assignment models. The state of the network is defined by flows in the input sections, turning proportions for all sections, number of vehicles moving and stopped in each section for each turning, and the mean speed for vehicles travelling in each section.

AIMSUN is integrated with the GETRAM Simulation Environment (Generic Environment for Traffic Analysis and Modelling, [GRA93]). It consists of a traffic network graphical editor (TEDi), a network database, a module for storing results and an Application Programming Interface which allows interfacing to assignment models and to other simulation models. Currently, interfaces to the EMME/2 and QUESTOR transport planning models available.

There is an interface to TRANSYT/10, which is used to convert a GETRAM network into the TRANSYT/10 input data file format. TRANSYT/10 can be run directly from TEDi, the GETRAM editor, and the optimised control plans can be loaded into AIMSUN, which can then provide the TRANSYT/10 units and measures of performance as additional output. The translated GETRAM network can be also loaded into TranEd, a TRANSYT/10 graphical editor.

There is an interface to SCATS. Through Tedi the user can define signalised junctions to be controlled by SCATS adaptive control system. Then, AIMSUN runs in parallel with SCATS, providing detection measures to SCATS, which send back to AIMSUN the corresponding adaptive traffic control actions during simulation.

On the other hand, a special module called GETRAM Extension (API) is provided. This has a DLL Library of functions that enable the user to develop any external applications that may need access to AIMSUN internal data during simulation run time. This module can be used to develop interfaces to external control applications, such as, for example, SCOOT, C-Regelaar, Balance, etc. Using the set of DLL functions, any external application may access the data produced by simulated detectors (e.g. flow, occupancy, etc.). This data can then used to feed in a control plan, and take control of the traffic signals, VMS's and ramp metering signs being simulated (e.g. change the traffic signal state, the phase duration, display a message on a VMS, etc.). This module can also use detailed simulated vehicle data (e.g. fuel consumption and pollution emissions) to feed a user model, or to keep track of a guided vehicle through the network using an external vehicle guidance system.

1.2 FUNCTIONAL CAPABILITIES

AIMSUN's capabilities are summarised below:

- AIMSUN can deal with different types of traffic networks: urban networks, freeways, highways, ring roads, arterials and any combination thereof.
- Depending on the available traffic demand data, two different types of simulation are considered: 1) based on traffic flows and turning proportions, 2) based on O-D matrices and route choice models.
- Binomial, Logit and C-Logit Route Choice models are available as default. The user can also use his/her own Route Choice model.
- Shortest paths can be calculated according to default or user-defined Cost Functions.
- Different types of traffic control can be modelled: traffic signals (fixed, variable and adaptive), unsignalised junctions (give-way and stop signs) ramp metering and tolls.
- Vehicle manoeuvres are modelled in detail using car following, lane changing and gap acceptance models. Both car following and lane changing are based on Gipps models.
- 'Look ahead' model: vehicles can take lane change decisions based not only on the immediate next turning movement, but on a set of next turning movements.
- Different types of vehicles can be modelled: cars, buses, trucks, etc. They can be grouped into classes, and reserved lanes for given classes can also be taken into account.
- The user may choose from different headway models for vehicle generation: constant, uniform, normal, exponential, capacity and user-defined.
- Public Transport modelling: bus lines (routes and bus stops), timetables (departure frequency or fixed schedule), stop times.
- Due to the detailed modelling of each vehicle in the network, AIMSUN can simulate all kinds of traffic detectors that are defined by measurement capability: counts, occupancy, presence, speed.
- Detailed statistical output is provided: flows, speed, travel time, queue length, etc. Measures can be aggregated for the whole system or disaggregated by sections, turns, origins, destinations etc.
- The user may decide to store the simulation outputs either as ASCII files or as a database, the latter using an ODBC format.
- AIMSUN is integrated with the GETRAM simulation environment, which consists basically of a graphical network editor (TEDI) and a database in which to store networks.
- Network model building, despite the amount of detail required for microscopic modelling, is a straightforward task thanks to the graphical network editor, TEDI.
- Through a user-friendly interface the user can define scenarios and simulation experiments.
- A Scenario is the combination of a network description, a traffic demand data, a traffic control plan, a public transport plan and a set of initial conditions.
- A simulation experiment is a set of replications of the simulation to be run, and the random seed for each replication. Experiments can be run in Batch mode and the results of each replication are stored.
- It also provides a picture of the network as graphical output and an animated representation of the vehicles running in it.
- The user may define traffic incidents, before or during the simulation run. A list of incidents may be stored for use in future simulation runs.
- Variable Message Signs and the influence that the displayed messages may have on the driver's behaviour can also be simulated. VMS can be also used to model Speed Control.
- Interfacing with external applications, such as Adaptive Control Systems or Consumption Models, is possible using the GETRAM Extension Module (API). This interface is based on DLL programming.
- It can also interface with other traffic simulation or assignment models, such as EMME/2 or QUESTOR.
- AIMSUN is programmed in C and C++. It runs under Windows NT, Windows '95, Windows '98 and Linux. In Windows systems, an X-Server such as eXceed or X-Win32 is required.

1.2.1 GETRAM v4.0 New Features

TEDI

- Public Transport: ability to edit public transport routes, bus stops, timetables, stop times, etc.
- User-defined Route Choice models or Cost Functions to be used for the Shortest Path calculation can be written using an interactive function editor.
- User-defined Section Cost: a new section attribute representing some economic or monetary cost, such as tolls, etc. This attribute can be used in any user-defined function.
- There is an interface to TRANSYT/10 embedded into TEDI, whose main function is to convert a GETRAM network into the TRANSYT/10 input data file format. The translated GETRAM network can also be loaded onto the TranEd, graphical editor. Control plans calculated by TRANSYT/10 are imported directly into TEDI, and can be saved as GETRAM control plans.
- A network can be exported via the GETRAM-GIS interface into a Shapefile, a standard format that can be imported by most Geographic Information Systems, including ArcInfo, ArcView and MapInfo. Simulation results that have been saved as ODBC can be presented in a graphical format using the GIS interface.
- It accepts as a background, not only DXF format files but also BMP, GIF, TIFF and JPEG formats.
- Undo and Redo functions in the Edit menu, only related to movements of objects, rotations and modifications of (poly)sections and objects (texts and blocks).
- Ability to draw lines to build up decorative shapes.
- Scaleable texts, while zooming in and out.
- Navigator window, to help positioning in large network models.
- New Object Browser, which provides a class structure of the network objects.
- Various enhancements to dialog windows and graphical editing functions.

AIMSUN

- Public Transport Modelling. Buses depart at line-origins according to a timetable, they follow a particular bus line and stop for a certain time at the corresponding bus stops along the line.
- New Route Choice models are available: C-Logit (Modified Logit) model and User-Defined models.
- A cost function library is provided for shortest path calculation, and the user is also given the option of defining and using his/her own user-defined Cost Functions.
- For the calculation of cost functions, capacity is taken into account during the whole simulation process. Capacity is now considered at the level of turning, which is more accurate.
- No fixed relationship between the statistics report interval and shortest path calculation cycle.
- Look ahead model: vehicles can make lane change decisions based not only on the immediate next turning movement, but on a set of next turning movements.
- Preparing a simulation experiment is now faster. Just load a scenario and run. A scenario is the combination of a network description, traffic demand data, a traffic control plan, a public transport plan and a set of initial conditions.
- The simulation output data can be stored directly as an ACCESS data file (so defining an ODBC data source is not necessary).
- Option 'Flush' for gathering statistics: using this, the statistics will be saved onto the disk, but not stored in the memory. For big networks this option will increase the performance of the simulation.
- New graphical capabilities for presenting statistical outputs: time plots scale can be user-defined.
- Traffic detectors can be monitored during simulation: detector measures can be provided step by step, both numerically or graphically, as a time plot.
- TRANSYT/10 optimised control plans can be loaded into AIMSUN, which can then provide the TRANSYT/10 units and measures of performance as additional output information.
- Various enhancements to the user interface. New tab folder dialog windows are used to display different model information.

GETRAM Extensions

- GETRAM Extensions can now be programmed not only in C/C++, as a DLL library, but also using Python scripting language.
- Various enhancements have been made to functions related to vehicle information. The user can modify the attributes of a particular vehicle, not only dynamic attributes such as position and speed, but also static attributes such as desired speed, desired acceleration, etc.
- Functions related to Public Transport have been included: generate bus departure, modify the stop time, and access or modify any public transport vehicle attributes.
- Access to the scenario data, such as network path, traffic result or OD matrix name, control and public transport plans are also possible via the new GETRAM Extension Functions.

1.2.2 GETRAM v4.1 New Features**TEDI**

- Drawing circles.
- New editing capabilities related to the Path Analysis Tool.
- Accept decimals in the OD matrix
- Manipulation of OD Matrices: Multiply all the terms, or a row or column of a matrix by a factor.
- Ability to import/export all time slices and vehicle types of an OD matrix from/to one ASCII file.
- A first level of 3D editing in Tedi.
- Saving networks in binary format (not ASCII), to save time and disk space.
- Increase maximum number of turnings in a junction (from 50 to 99).

AIMSUN**Route Choice Model and Path Analysis Tool**

- Introduction of historical paths defined by the user. The user may edit historical paths in Tedi and use them in AIMSUN simulator. A percentage of vehicles following each historical path can be also defined.
- It is also possible to use historical costs (driver's memory) when calculating the shortest paths during simulation.
- Gather and provide statistical data per path and time slice. All this information may be saved in ODBC format.

Calibration and Result Analysis Tool

- Visualisation of real data from detectors to compare with simulated data.
- Tools for a first level of Validation: calculate correlation coefficient and Tail coefficients (R^2 , RMS, U , U_m , U_s , U_c .)
- Automatic calculation of averages and deviations of a set of replications, both for detection data and statistical outputs.

Traffic Modelling

- Virtual entrance queues at input section: distinguish by vehicle type.
- Introduction of a new parameter: Yellow Box Speed. It is a local parameter that will be used in the Yellow Box Model instead of the queue leaving speed, which can be useful for calibrating the capacity of a yellow box junction.
- Unify the zone distances of different sections of a polysection: e.g. use the distances of the last section of a polysection.
- Detection capabilities: detect headway between vehicles, consider density toggle button.
- New statistical measure: Total Vehicles Hours.

Traffic Management and Control

- Ability to model NEMA standards adaptive control plans

- Ability to import SYNCHRO control plans (with certain limitations: CSB format files, in case of adaptive control only 1 barrier and 1 ring are accepted).
- Logical name for a signal group in the control in correspondence with the C-Regelaar
- Detectors, VMS and Meterings have Identifier Number and Name (names can be duplicated).

Graphical User Interface

- 3D animation
- 2D smoother animation: redrawing frequency parameter can be smaller than simulation step.
- More realistic vehicle icons.
- Simulation buttons (play, stop, pause, FF, RW, etc.), ability to select the desired simulation speed.
- Legend in a separated window (definition of more colours for vehicle types and centroids).
- AConsole: Running Aimsun fully in Batch, without user interface.
- Animation: only vehicles on top of the background.
- Information about the scenario is included in the simulation results.
- Simulation run during warm-up period can be cancelled.
- The list of selected traffic states and traffic control plans is shown in the Scenario Dialog.
- New Show Objects and Preferences dialog.
- Additional information is given in the 'Number of Vehicles' dialog (Vehicles in, out, waiting, etc.)

GETRAM Extensions (API)

- Detection of Vehicle Identifiers (i.e. plate number identification)
- Detection of other vehicle attributes (i.e. length, acceleration capability, etc.)
- Possibility of getting additional information for the detectors (detector name, detector properties)
- Access to detectors and VMS's can be done by object identifier
- Access to general parameters of the network, such as name of vehicle types, network units, etc.
- Access to signal groups not only by identifier but also by logical name

EMME/2 Interface

- Import Public Transport from EMME/2 to GETRAM
- Export Detector information from GETRAM to EMME/2

TRANSYT/10 Interface

- Include the TRANSYT Network Performance Index as well as other network measures, such as the Total Time Spent and Total Uniform Delay.
- New procedure to calculate the Saturation Flow of a link, based on the Transport and Road Research Laboratory report RR67 'The Prediction of Saturation Flows for Road Junctions Controlled by Traffic Signals'
- PCU's units are considered. The user can define the PCU's equivalence for each vehicle type.

SCATS Interface

- The user can define signalised junctions to be controlled by SCATS adaptive control system.
- AIMSUN simulation runs in parallel with SCATS, providing detection measures to SCATS, which send back to AIMSUN the corresponding adaptive traffic control actions during simulation.

1.2.3 GETRAM v4.2 New Features

TEDI

- Import/export traffic states from/to ASCII files.
- Graphical representation of traffic control plans: standard control diagrams green-amber-red.
- New cost function (Past Cost) in Function Editor
- Default Functions hierarchy
- Ability to import all SYNCHRO control plans (multiple barrier and multiple ring).

- Edition of Routes and PT lines automatically using Shortest Path
- Automatic Generation of 3D polygons from a DXF background
- Standard default vehicle types library
- On-Line Help

AIMSUN

Dynamic Traffic Assignment: Route Choice Model and Path Analysis Tool

- Visualisation of simultaneously shortest Paths from: one origin to destination, one section to one destination and all origins to one destination
- New cost function (Past Cost) in Function Editor
- Route Choice Model considering different alternative paths.
- Use link costs calculated from polysection statistics
- Initial K-Shortest Paths At the beginning of the simulation, it calculates k initial shortest path (k defined by the user)
- Proportional Route Choice Model: $tt_k^{-\alpha} / \sum (tt_i^{-\alpha})$

Traffic Modelling

- Improvements to Car Following Model: Implement modification to Gipps model to avoid vehicles driving bumper-to-bumper.
- Lane-changing zone for PT-Vehicles: define new parameter 'distance to stop' for each bus stop.
- Different speed limits per lane in a section. This can be used also to influence the distribution of traffic among the different lanes of a section.
- Variable order in the entities updating sequence to improve variability.
- Maximum Give Way Time as a local section parameter (increase, decrease vehicle attribute)
- Possibility of defining Vehicles as 'Equipped' (for detection purposes)
- Possibility of deactivate the Yellow Box behaviour for particular turning movements of a Yellow Box Junction (Yellow Box Speed = 0).
- Improve give-way model: consider vehicle trajectories, not only lane turning movements, and 'view all approaching polysection' capability.
- Improvements in the vehicle generation process to better achieve the defined input flows

Traffic Management and Control

- NEMA Standards: Dual Ring
- SYNCHRO: Complete User Interface
- Controllers: to specify interfaces with adaptive systems (SCATS, UTOPIA).
- Bus Pre-emption
- Ability to define and model different levels of priority among the turning movements of a junction.
- New VMS Actions: Change speed limit (per vehicle type)

Validation and Result Analysis Tools

- Calculation of average for selected replications
- Validation tool per vehicle type

Simulation Outputs

- Statistical outputs per polysection.
- Statistical data to be gathered during simulation defined by the user
- Actuated Control output:
 - display of phase diagram, and
 - during the simulation state of actuated control
 - during the simulation state of bus-preemption
- Detection of equipped vehicles

- Instant Detection: detection can be gathered at small time intervals, that can be different (even smaller) than the Simulation Step; to be used only in relation with GETRAM Extensions and adaptive control Interfaces (SCATS and UTOPIA).

Graphical User Interface

- Graphical Statistical outputs per polysection.
- 2D Articulated Vehicles
- Vehicle colouring per OD pair
- AIMSUN (and AIMSUN 3D) try to locate the network referenced in the scenario file if it has been moved. AIMSUN tries first to find the network automatically. If it cannot find the network automatically then it will ask the user for the network location (using a file dialog).
- Vehicle Colouring option selected is kept during the simulation session.
- Various improvements to the GUI: better network drawing at large scales, better visualization of paths, improvements to some dialogs, etc.
- Through ODBC option, it is now possible to access to databases where username and password are required (i.e. Oracle, ...)
- On-Line Help

AIMSUN 3D

- Improve 3D Articulated Vehicles
- On-Line Help

GETRAM Extensions

- Change speed limit of a section per vehicle type
- Close lane of a section per vehicle type
- Change next turning (per Origin, Destination, vehicle type)
- Change next turning per vehicle type (Traffic Result)
- Change destination centroid (per Origin, Destination, vehicle type)
- Change turning proportion per vehicle type (absolute value)
- Generate vehicle inputs in OD mode following AIMSUN procedure
- Dynamic modifications in OD matrix values.

EMME/2 Interface

- Automate the Traffic Results Import from EMME/2
- Automate the PT-Lines Import from EMME/2
- Import Detectors Layout

UTOPIA Interface

- The user can define signalised junctions to be controlled by UTOPIA adaptive control system.
- AIMSUN simulation runs in parallel with UTOPIA, providing detection measures to UTOPIA, which send back to AIMSUN the corresponding adaptive traffic control actions during simulation.

1.3 SYSTEM STRUCTURE

Figure 1-1 shows the AIMSUN functional structure and its integration within the GETRAM Simulation Environment. The following modules make up AIMSUN:

- User Interface: enables communication between user and simulator (see Figure 1-2). This interface module is introduced in section 1.5.
- Pre-Simulator Module: reads a scenario description from the Network database to simulate and build up the running model (data structures). This module performs a first stage of the calibration process, which is the model consistency checking. Chapter 2 describes the mechanisms used to represent the network.
- Simulation Module: performs simulation of the selected scenario. Chapter 3 describes the traffic models used while chapter 5 describes how traffic control is modelled.
- Shortest Route Module: for the Dynamic Traffic Assignment, this module calculates the current shortest routes from every section to every destination. Chapter 4 describes the functioning of the Dynamic Traffic Assignment in AIMSUN.
- GETRAM Extension (API): External Applications Interface that provides simulated information to an external system and accepts various control and management actions. This module is described in the GETRAM Extension User Manual.

Figure 1-1: GETRAM-AIMSUN Conceptual diagram

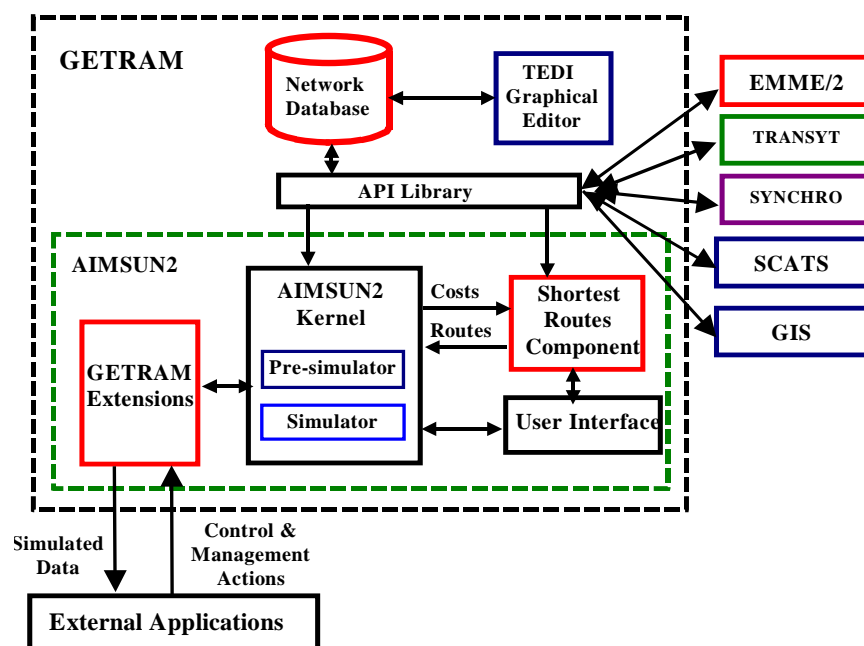
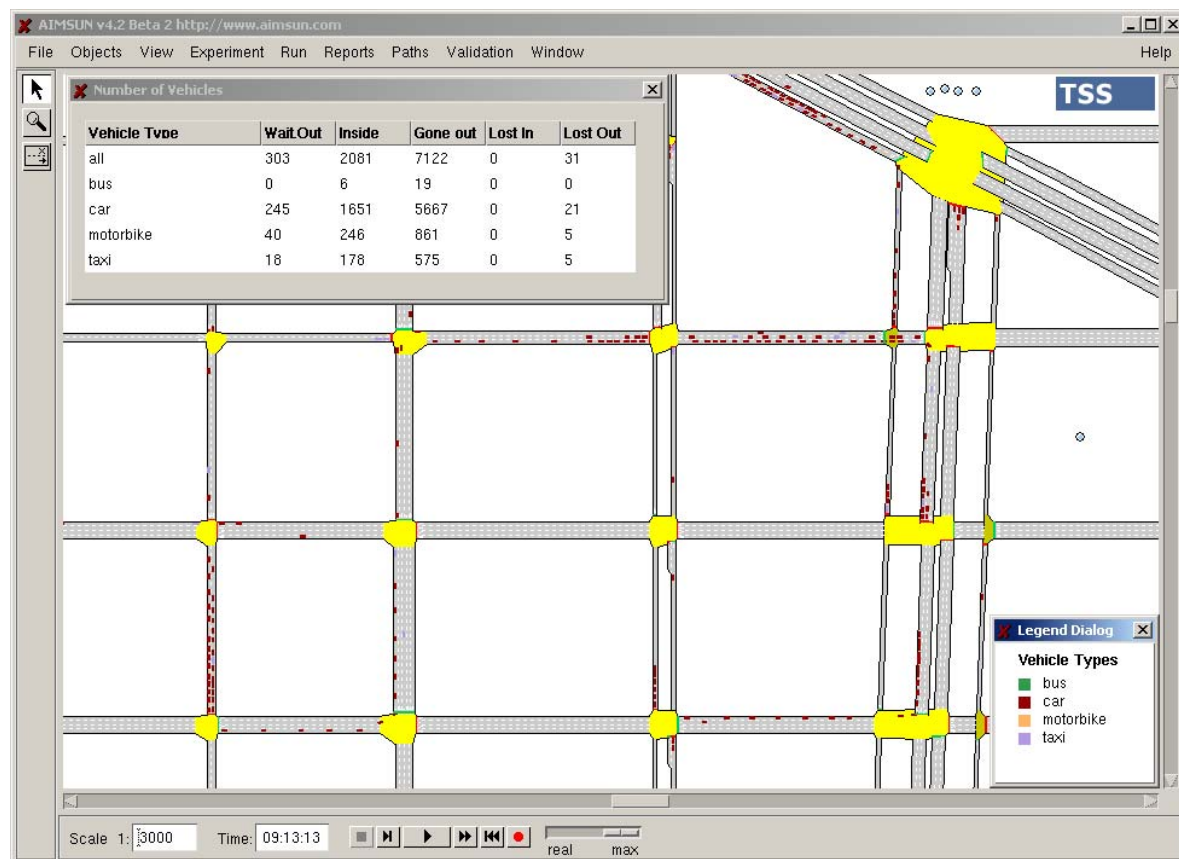
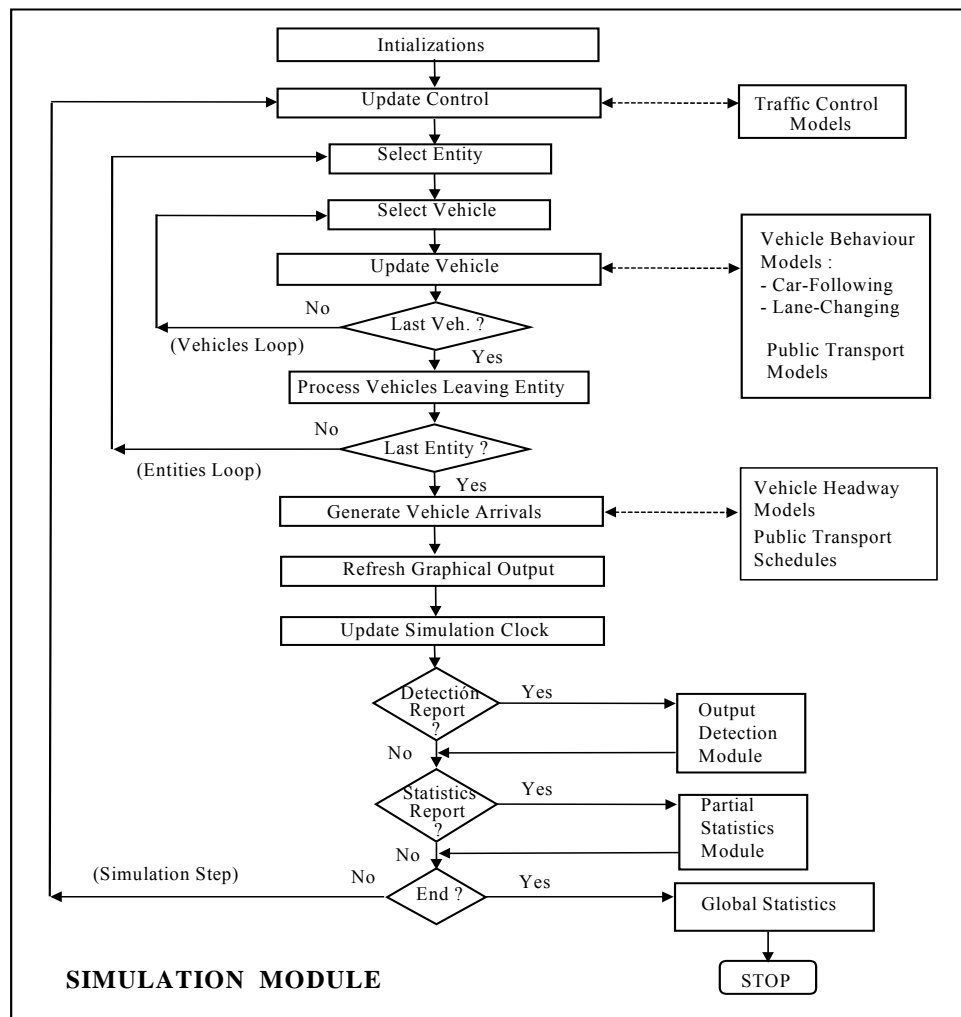


Figure 1-2: A Snapshot of the AIMSUN User Interface

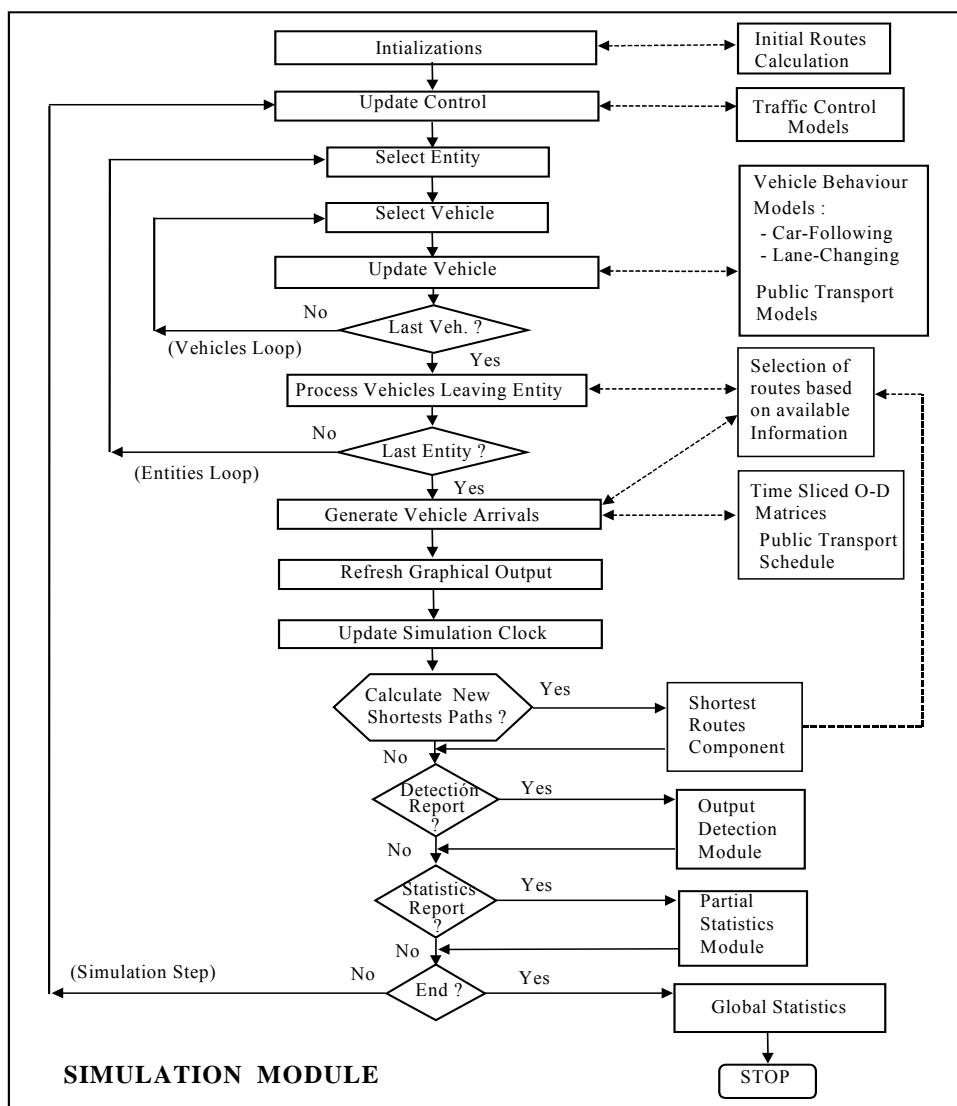
1.4 SIMULATION PROCESS

The logic of the simulation process in AIMSUN is illustrated in Figure 1-3. It can be considered as a hybrid simulation process, combining an event scheduling approach with activity scanning. At each time interval (simulation step), the simulation cycle updates the unconditional events scheduling list (i.e. events such as traffic light changes which do not depend on the termination of other activities). The “Update Control” box in the flow chart represents this step. After this updating process, a set of nested loops starts to update the states of the entities (road sections and junctions) and vehicles in the model. Once the last entity has been updated, the simulator performs the remaining operations such as inputting new vehicles, collecting new data, etc.

Figure 1-3: The Simulation Process in AIMSUN



Depending on the type of simulation, new vehicles are input into the network according to flow generation procedures (headway distributions for example) at input sections, or using time sliced O-D matrices and explicit route selection. In this case, the simulation process includes an initial computation of routes going from every section to every destination according to link cost criteria specified by the user. Figure 1-4 shows the simulation process for the Route Based model. In this case, a shortest route component periodically calculates the new shortest routes according to the new travel times provided by the simulator, and a route selection model assigns the vehicles to these routes during the current time interval. Vehicles keep their assigned route from origin to destination unless they have been identified as “guided” at generation time. Then they can dynamically change their trajectory en route as required for simulating vehicle guidance and vehicle information systems.

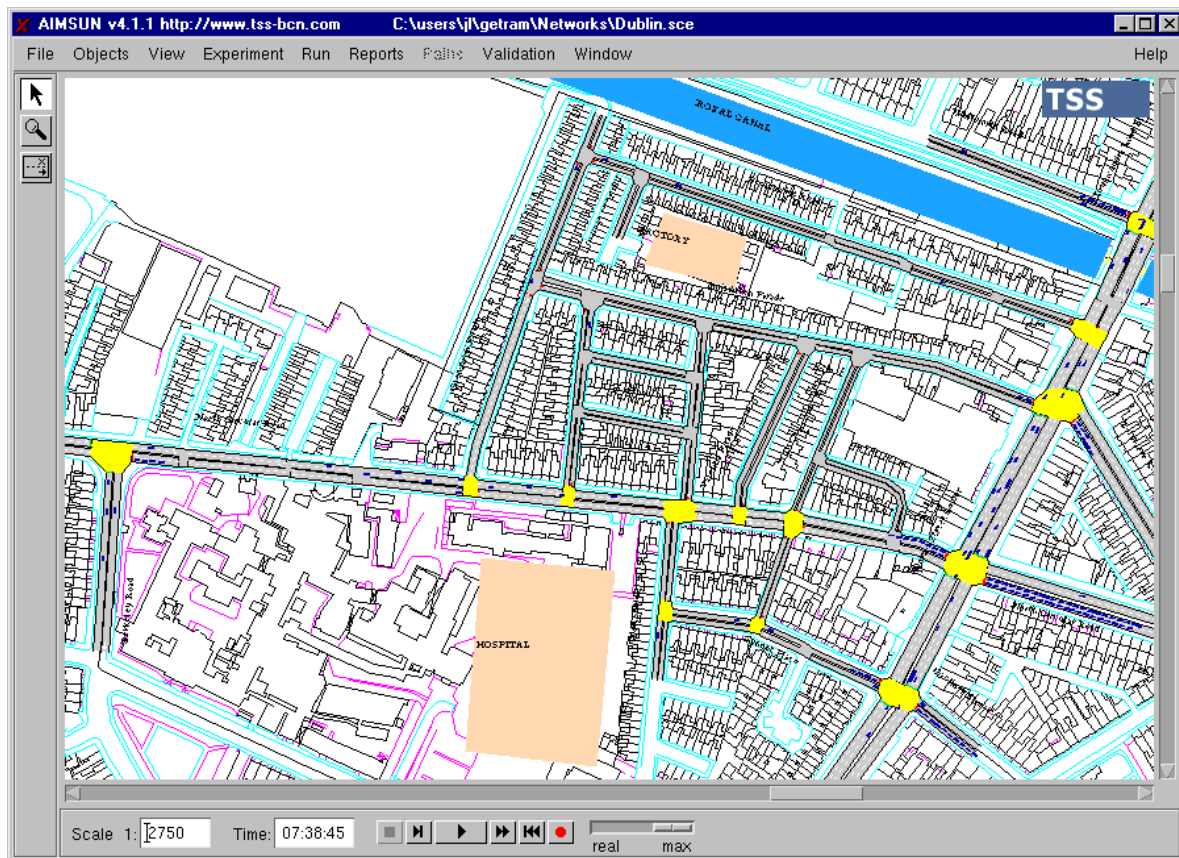
Figure 1-4: The AIMSUN Simulation Process (Route-Based)

1.5 THE AIMSUN GRAPHICAL USER INTERFACE

The AIMSUN Graphical User Interface provides the necessary tools for preparing simulation experiments, selecting scenarios to simulate, defining simulation parameters, interacting with the simulator while it is running, viewing the animated graphical output and analysing statistical results.

This user interface is a windows-based system. The main window has four parts or areas: the menu bar on the top, the toolbar on the left, the network display area in the centre and the status bar at the bottom. Figure 1-5 displays this main window, to which an example network has been loaded.

Figure 1-5: Another snapshot of the graphical user interface



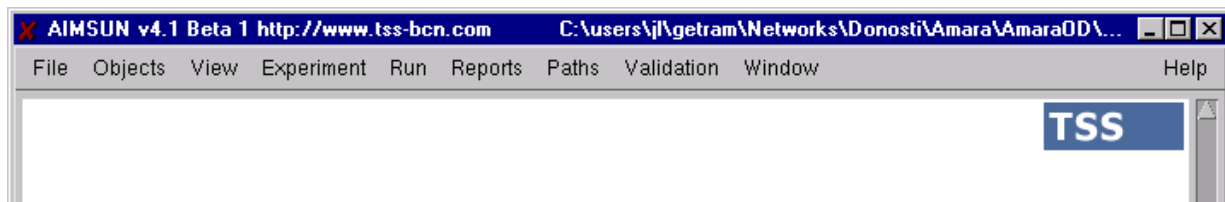
The network display area is the central section of the main window. This is where the image of the network appears. There are also scroll bars for moving the image vertically and horizontally (see Figure 1-5). Moving the mouse cursor while keeping the right mouse button pressed down will move the network in the corresponding direction.

The left mouse button is generally used for clicking on menu commands and selecting network objects, the latter by double clicking on the object. When using a three-button mouse, the middle button can be used to zoom in and out, by clicking or shift clicking respectively.

1.5.1 The Command Menu

The Menu Bar is located at the top of the window and features nine options: File, Objects, View, Experiment, Run, Reports, Paths, Validation and Window (Figure 1-6). Each of these comprises a pull-down menu containing a set of related commands. There is also a Help option, although on-line help is not provided, only information about the available License and the current AIMSUN version. The following is a brief description of all the AIMSUN Interface Commands.

Figure 1-6: Menu Bar



File

Includes all operations related to the GETRAM database and other files: scenarios, networks, control plans, traffic demand data, actions related to message signs and simulation state.

- **Load Scenario:** loads a simulation scenario, defined by a network, a traffic result or an OD matrix, a traffic control plan and a public transport plan.
- **Reload Scenario:** reloads the current simulation scenario.
- **Save Scenario:** saves the currently loaded network, traffic demand data, traffic control plan and public transport plan as a simulation scenario.
- **Unload Scenario:** clears the current simulation scenario.
- **Load Network:** loads a network description (sections, junctions and junctures, turnings, detectors, Variable Message Signs, ramp metering) and builds the structure of the simulation model.
- **Unload Network:** clears the simulation model of the current network.
- **Load Traffic Result:** loads the traffic demand data to be simulated, defined as input flows and turning proportions, and feeds the simulation model with this data.
- **Unload Traffic Result:** clears the current traffic demand data.
- **Load OD Matrix:** loads the network zones definition (centroids) and the demand data to simulate, defined as an O-D Matrix.
- **Unload OD Matrix:** clears the current OD matrix and centroids definition.
- **Load Control:** loads a traffic control plan, consisting of signal groups and phases, and completes the simulation model with it.
- **Unload Control:** clears the current traffic control plan.
- **Load Public Transport:** loads a public transport plan, consisting of bus lines and timetables.
- **Unload Public Transport:** clears the current public transport plan.
- **Load Extensions:** it is used for defining and loading a set of DLL libraries to be used during the simulation run. It is only available with Kit 1 (GETRAM Extensions).
- **Save Message Actions:** saves the definition of action associated with the messages that can be used in the Variable Message Signs (VMS's).
- **Save Current State:** saves the current state of the simulation as a new state result in the GETRAM Data Base, which can be used or continued in future simulation experiments.
- **Quit:** exits AIMSUN.

Objects

Set of options for finding and viewing information on any object within the model. Also, options for editing these objects. These options are grouped into two sets: Show and Edit.

(SHOW)

- **Sections:** to find a section and show its characteristics (lanes, turns allowed at each lane, input flows, turning proportions, detectors, metering).
- **Junctions:** to find a junction and show its characteristics (turning movements, signal groups and phases).
- **Detectors:** to find a detector and show its characteristics (position, length, lanes, measuring capabilities)
- **Meterings:** to find a ramp metering device and show its characteristics (type, position, control parameters)
- **VMS's:** to find a Variable Message Sign and show its characteristics (position, accepted messages and actions). The user can also activate messages that will trigger the execution of its associated actions (impact in the driver's behaviour).

- **Centroids:** to find a centroid and show its characteristics (type, connections, OD matrix data). This is also used to select a set of centroids in order to colour the vehicles going to or coming from those centroids.
- **Controllers:** to find a controller and show its characteristics.
- **Public Transport:** to view Public Transport Lines and Bus Stops.

(EDIT)

- **Actions:** to edit and show actions related to VMS messages.
- **Streams:** to edit and show streams (set of consecutive sections, similar to a route), which are used for the only purpose of collecting statistical data for analysis.

View

Set of options regarding the presentation of the network image.

- **Whole Network:** the scale is automatically converted to the minimum needed to allow the whole network image to fit into the window.
- **Minimum Cars Scale:** the scale is automatically converted to the minimum needed to allow the cars to be represented individually.
- **Vehicle Colouring:** colours all the vehicles in the network in order to identify either the vehicle type, the next turning movement, the origin or destination centroid (for some previously selected centroids), the Public Transport vehicles, lost vehicles, or to colour the vehicles in a section according to their destination.
- **Number of Vehicles:** to display, during the simulation run, the number of vehicles driving in the network and how many are waiting to enter or have already left the network, distinguishing by vehicle type.
- **Show Objects:** to select which objects must be displayed or not and whether or not to display the names and/or identifiers of the displayed objects.
- **Show/Hide Speed Limits:** to display the section speed limits icons on the network.
- **Show/Hide Virtual Queues:** to display the Virtual Queue icons on the network. These icons are used to access to the information about vehicles waiting to enter into the network, i.e. queuing at the input sections.
- **Vehicle Low/High detail:** to select the level of detail to use for the vehicle drawing.
- **Show/Hide Occupied Detectors:** to highlight in a different color whenever a vehicle is stepping onto the detector (i.e. it is occupied).
- **Background:** to register and load background DXF files.
- **Preferences:** to select the colours to be used for highlighting various objects when they are selected and the colours for drawing the different types of vehicles.

Experiment

Set of options for defining parameters for the simulation experiment.

- **Run Time:** to specify time parameters (start-up and end time of simulation, warm-up period and redrawing frequency)
- **Modelling:** to specify the global modelling parameters, related to the vehicle behaviour models used in AIMSUN.
- **Input:** contains an extended menu with one option related to the simulation inputs:
 - **Past Costs:** to specify where to find the section's Past Costs, calculated in a previous simulation run and stored into a database in ODBC format.
- **Output:** contains an extended menu with three option related to the simulation outputs:
 - **Output Location:** to specify where to store the simulation outputs (statistics, detection and recorded simulation) either as ASCII files or as a database, the latter either using an ODBC format or directly into MS Access.
 - **Statistics:** to specify parameters defining the statistical reporting to be produced by the simulation (level of detail and periodicity).
 - **Detection:** to specify whether or not to carry out detection, the type of detection and frequency.
 - **Paths:** to specify whether or not to gather statistics on Paths usage.

- **Replications:** to specify whether to make a single or a multiple run and the numbers used as initial seeds for the random generator. If it is a multiple run, you can also specify whether or not to calculate the average of the replications.
- **Route Choice:** to specify the Route Choice Model to be used (fixed or variable, static or dynamic, binomial, Logit, C-Logit or user-defined) and some related parameters.
- **Incidents:** to view the Incidents Log File and find incidents, showing their characteristics and status (active, done or pending) and deleting or saving them in a log file for future experiments.
- **VMS Messages:** to view the Messages Log file. The user may load/save a sequence of messages to be activated on the VMS's during the simulation. The user may also trace the activated messages during the simulation run.
- **Environmental Models:** to define whether or not to consider Fuel Consumption and/or Pollution Emission modelling.

Run

Different options for running and stopping the simulation.

- **Run:** to run the simulation with continuous animated graphical representation of the traffic. If the scale is set to the Minimum Cars Scale or lower, all the vehicles are shown as they move along the network, otherwise only a range of colours representing density is displayed.
- **Step:** to run a single simulation step, the graphical display is refreshed.
- **Batch:** to run the simulation without refreshing the graphical display (this is much faster than Run mode).
- **Stop:** to stop the simulation immediately, either Run or Batch.
- **Begin:** to clear the current simulation experiment.
- **Stop at:** to stop the simulation when a certain time has elapsed.

Reports

Options for viewing statistical reports and detection data obtained from current or previous simulation experiments. The statistics gathered by AIMSUN are: mean flows, travel times, delay times, mean speeds, density, stops and queue lengths. All these variables can be specified at different levels of aggregation: the whole system, sections, turning movements, streams (set of consecutive sections), origin or destination centroids or O/D pairs. The statistics can be global (the whole simulation time) or periodic (certain statistical interval predefined). There are three possible options:

- **File Reports:** to load statistical reports stored in files, which may have been saved from previous or current simulation runs. It can also be used to load detection reports.
- **Current Report:** to view statistics or detection data gathered during the current simulation experiment.
- **Current Graphics:** to display time series plots of statistical or detection data gathered during the current simulation experiment, or to show the sections of the network coloured according to a range of colours that represents different values for a set of traffic measurements, such as flow, speed, queue length, etc.

Paths:

Set of options related to route choice and path information.

- **Shortest Paths Info:** the user can view shortest paths that are being used by vehicles during simulation and get path information, such as cost, travel time or distance
- **Shortest Paths Display:** the user can view all shortest paths that are being used by vehicles during simulation, compare different paths or view path trees.
- **User-defined Paths:** the user can view all user-defined paths.
- **Initial Path Assignment:** the user can view all probabilities considered when a vehicle enters in the system.
- **Dynamic Path Assignment:** the user can view all probabilities considered when a vehicle makes a path reassignment during the trip.

Validation:

Commands of the Validation Tool for comparing different sets of data (simulated vs. real or simulated vs. simulated).

- **Database:** to select the database where the data is stored.
- **Average:** to calculate the average results for selected replications
- **Charts:** to plot as time series the selected sets of data. The user can previously select the sets of data and variable to compare, and the coefficients to calculate.

Window

Set of options for managing windows.

- **Show Scenario Info:** to open the Scenario dialog window containing the definition of the current Scenario (network, demand data, control plan, public transport plan, etc.).
- **Show/Hide Legend:** to select whether or not to display a legend in the window, which either provides the scale of colours for the density or for the vehicle colouring option.
- **Show Log Window:** to open the Log Window containing the list of Warning Messages that occurred during the session.
- **Hide/Show Status Bar:** to select whether or not to display the Scale and Time at the bottom of the main window.
- **Hide/Show Toolbar:** to select whether or not to display the tool bar on the left-hand side of the main window.

Help

Provides general information about the software.

- **Contents:** Getting Started on-line manual.
- **Licenses:** type and status of the AIMSUN license in use.
- **On version:** current AIMSUN version.

1.5.2 The Toolbar

The Toolbar (Figure 1-7), which can be displayed or not using 'Hide/Show Toolbar', is located on the left side of the main window and has three icons with the following functions:

Figure 1-7: The Toolbar



- **Select (arrow):** the mouse cursor becomes an arrow. The user can open any object of the network by double clicking on it.
- **Zooming (lens):** the mouse cursor becomes a magnifying lens. Clicking on the left mouse button it causes it to zooms in, while pressing the shift key at the same time produces zooms out.
- **Incidents:** the mouse cursor becomes an arrow; clicking on a lane of any section the user can define incidents or lane blockages (see section 3.6).

1.5.3 The Status and Control Bar

The Status bar containing the Scale, Time and Simulation Buttons is located at the bottom of the main window (see Figure 1-8).

Figure 1-8: Status Bar

The Scale of the network image is displayed in the format 1:N. This dialog window can be used to directly enter a value for variable N, although the scale can also be fixed using the zoom in and zoom out function.

The Time or Simulation Clock is displayed in hours, minutes and seconds and is automatically updated during the simulation run. The user cannot edit this dialog window directly.

The Simulation Buttons are used to select the different options for running and stopping the simulation, as in the Run Command Menu, as well as to record simulations for 3D animation.

1.6 INPUT DATA REQUIREMENTS

Microscopic simulation is characterised by the high level of detail at which the system is modelled. The quality of the model is highly dependent on the availability and accuracy of the input data. Therefore the user must be aware that in order to build a good AIMSUN model, the following data is required:

1.6.1 Network Layout

An AIMSUN traffic network model is composed of a set of sections (one-way links) connected to each other through nodes (intersections), which may contain different traffic features. To build the network model, the following input data is required:

- Map of the area, preferably a digitised map in .DXF format.
- Details of the number of lanes for every section, reserved lanes and side lanes (on and off ramps).
- Possible turning movements for every junction, including details about the lanes from which each turning is allowed and solid lines marked on the pavement.
- Speed limits for every section and turning speed for allowed turns at every intersection.
- Detectors: position and measuring capabilities.
- Variable Message Signs: position and (optionally) the possible messages.

1.6.2 Traffic Demand Data

Traffic demand data can be defined in two different ways:

- by the traffic flows at the sections
- by an O/D matrix

Depending on the type of model selected, the following input data must be provided:

1. Traffic Flows

- Vehicle types and their attributes
- Vehicle classes (for reserved lanes)
- Flows at the input sections (entrances to the network) for each vehicle type
- Turning proportions at all sections for each vehicle type

2. O/D Matrix

- Centroid definitions: traffic sources and sinks
- Vehicle Types and attributes
- Vehicle Classes (for reserved lanes)
- Number of trips going from every origin centroid to any destination one

1.6.3 Traffic Control

AIMSUN takes into account different types of traffic control: traffic signals, give-way signs and ramp metering. The first and second types are used for junction nodes, while the third type is for sections that end up in join nodes. The input data required to define the traffic control is as follows:

- Signalised junctions: location of signals, the signal groups into which turning movements are grouped, the sequence of phases and, for each one the signal groups that have right of way, the offset for the junction and duration of each phase.
- Unsignalised junctions: definition of priority rules and location of Yield and/or Stop Signs.
- Ramp metering: location, type of metering, control parameters (green time, flow or delay time).

1.6.4 Public Transport

The user may opt to have AIMSUN Public Transport into account. The input data required to define Public Transport is as follows:

- Public Transport Lines: a set of consecutive sections composing the route of a particular bus.

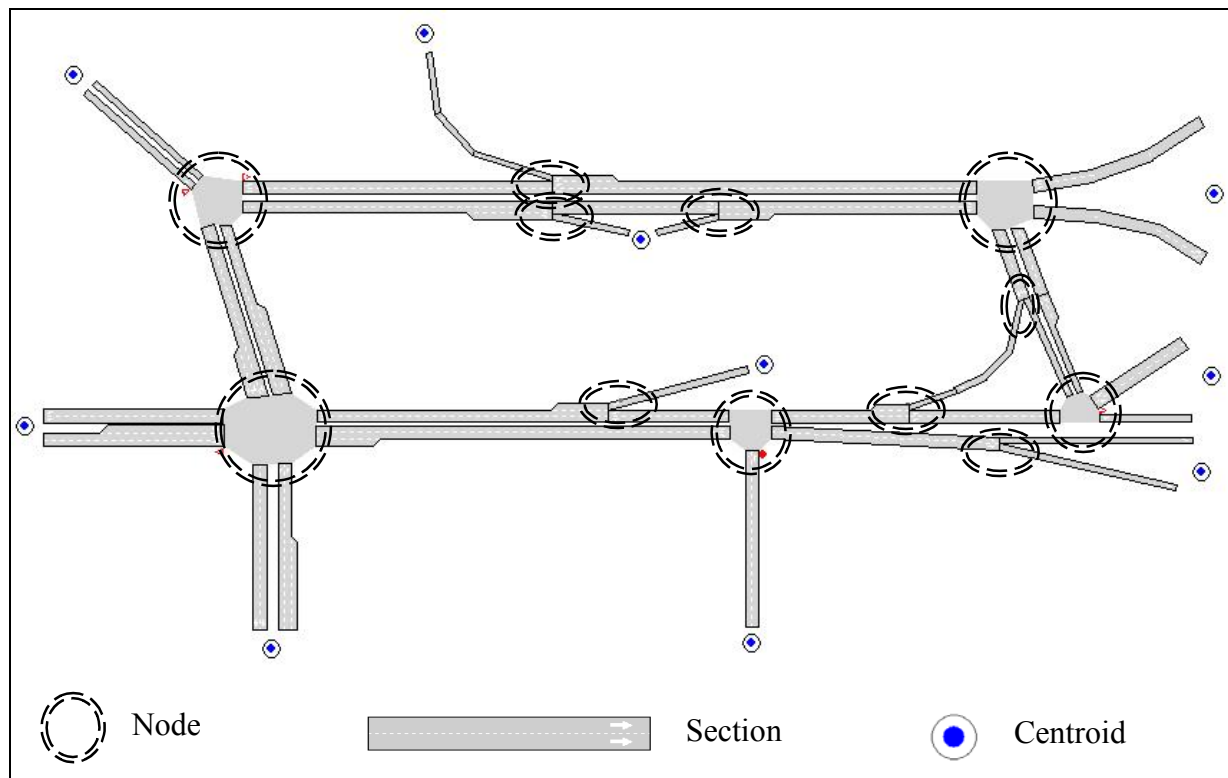
- Reserved lanes
- Bus Stops: location, length and type of bus stops in the network.
- Allocation of Bus Stops to Public Transport Lines.
- Timetable: departures schedule (fixed times or frequency), type of vehicle, and stop times (mean and deviation) for each bus stop.

2. NETWORK MODELLING

2.1 NETWORK STRUCTURE

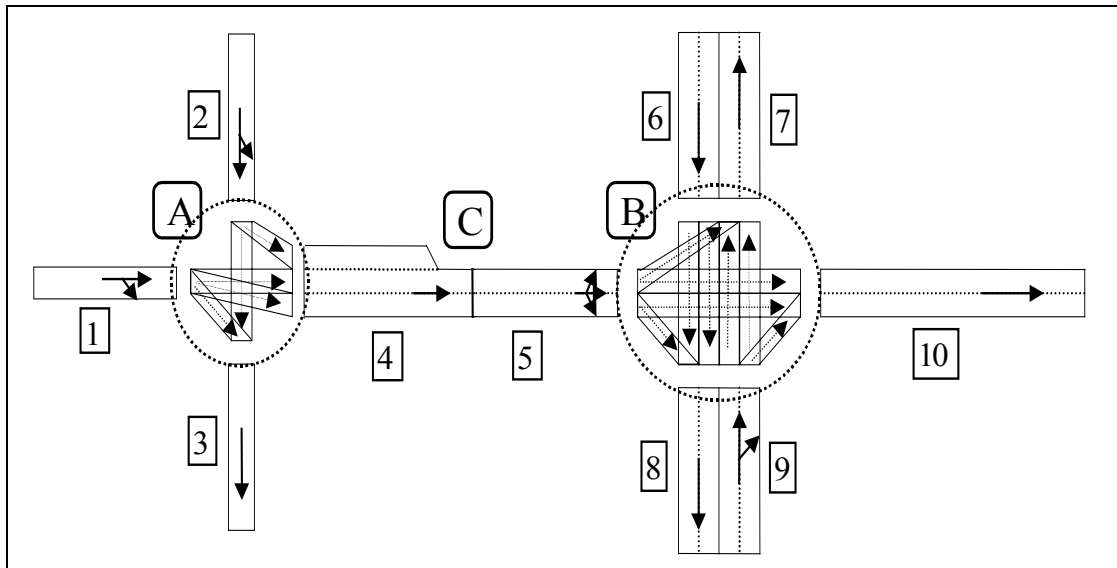
An AIMSUN traffic network model is composed of a set of sections or polysections (one-way links) connected to each other via nodes. Sections and nodes may be connected to centroids, which can be thought of as traffic sources or sinks. Figure 2-1 shows an example of a network model.

Figure 2-1: Example of a network model



Although, from the modeller's point of view, the network objects are nodes and sections, internally the basic modelling structure is the entity. The AIMSUN Pre-Simulator Module automatically makes this translation. Sections are composed of section-entities, which correspond to lanes, and nodes are made up of node-entities which are the areas connecting input and output lanes, where turns take place. Therefore, a network can be seen as a set of related entities (see Figure 2-2). Vehicles move along the network through entities according to driver behaviour models, which are a function of their state, defined by the current and adjacent entities.

In the network of Figure 2-2 there are ten sections (numbered from 1 to 10), and three nodes (A, B, and C). Every section is broken up into entities, each of which corresponds to one lane. For instance, section 4 has three entities, including the on-ramp lane. On the other hand, each node is broken up into entities, each of which corresponds to a turning movement (from lane to lane). For instance, node A has five entities, corresponding to all turns possible from sections 1 and 2 to sections 3 and 4. Node C is a direct connection between two sections and therefore it has no entities.

Figure 2-2: Breaking a network down

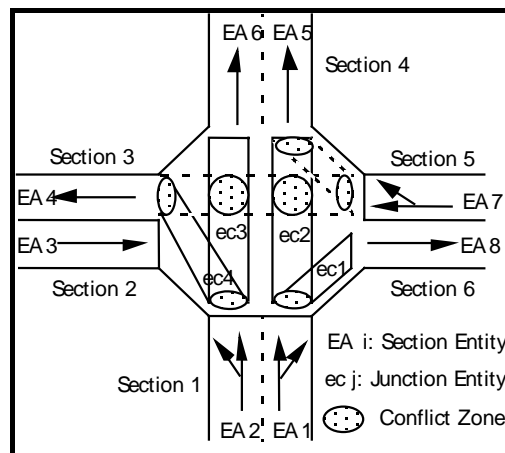
2.2 NODES

2.2.1 Junctions and Joins

A node is the intersection of two or more sections. Nodes are points in the network where traffic from one section is distributed among other sections. There are two types of nodes: junctions and joins.

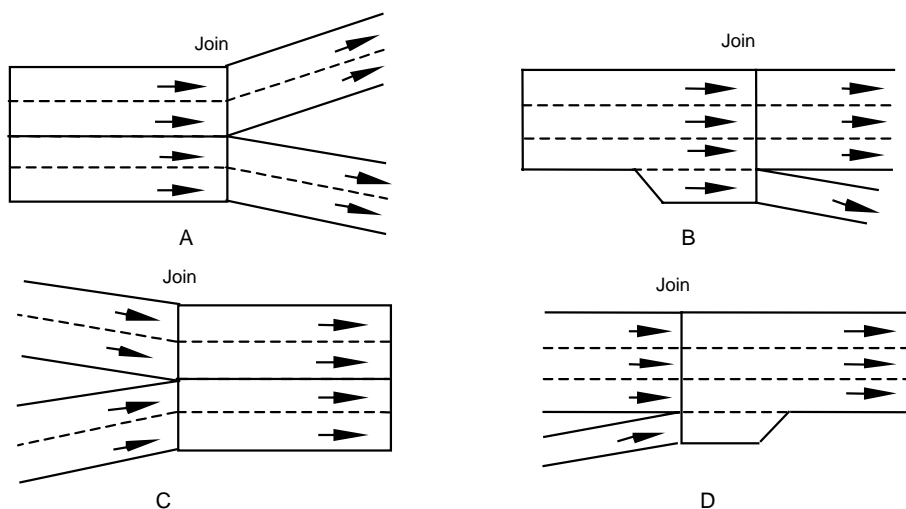
- Junctions:** The input and output sections at a node determine an area that vehicles use to go from one section to another. This is the junction. It is modelled by junction-entities representing all possible connections between input and output lanes. As shown in Figure 2-3, a junction-entity is the space that a vehicle has to drive through the junction. A special feature of this type of entity is the ability to model conflict zones in order to avoid collisions between vehicles making different turning movements. This is the most common type of node in urban networks. Figure 2-3 also shows the conflict zones needed to model the interactions between the turning movements of section 1 and 5.

Figure 2-3: Modelling a junction.



- Joins:** The input and output sections of a node are connected directly, in such a way that it is not necessary to use junction-entities. In this case, the number of entities entering a node is equal to the number of entities exiting it. Figure 2-4 shows some examples. This is the most frequent node on freeways and ring roads.

Figure 2-4: Examples of joins



2.2.2 Yellow Box Junction

A junction can be defined as a Yellow Box. The purpose of a Yellow Box model is to prevent vehicles from stopping inside the junction area. Therefore, vehicles will not attempt to move into a Yellow Box junction when there is not enough room in the destination section because the end of the queue reaches the junction. In fact, a vehicle will avoid entering a Yellow Box junction whenever the preceding vehicle is moving at a speed lower than 'Yellow Box Speed', which is a local section parameter (see section 3.4.2).

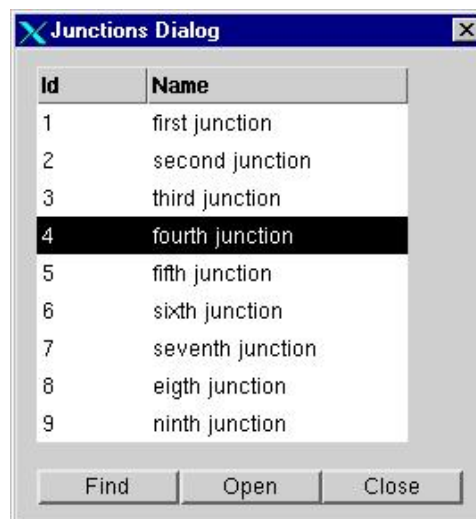
The purpose of the Yellow Box model is to avoid blockages of junctions, which may lead to a gridlock situation in urban scenarios. To denote a Yellow Box Junction, a toggle button is activated in the Junctions dialog window (see Figure 2-6) and the junction area is coloured yellow, instead of grey.

The user can deactivate the Yellow Box model of a particular section by setting the Yellow Box Speed parameter of that section to 0.

2.2.3 Junction Dialog Windows

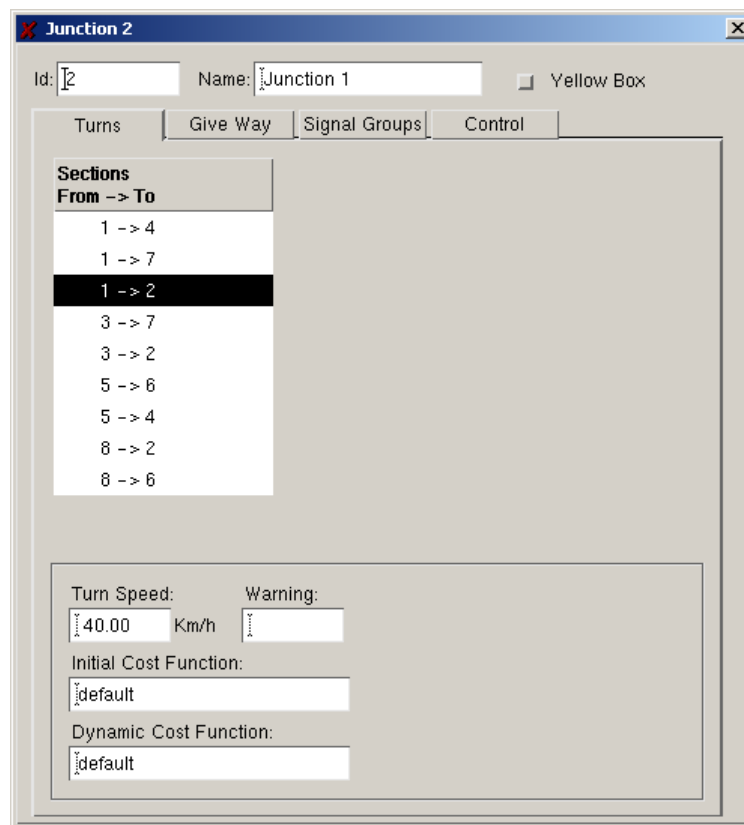
When you select 'Objects / Junctions' from the menu bar, the 'Junctions' search window (shown in Figure 2-5) is displayed. The list of junction identifiers and junction names is displayed. To go to a junction, select it by clicking on the list box and press the 'Find' button.

Figure 2-5: Junctions Search Window



To view the junction information you can press the 'Open' button in the 'Junctions' search window or simply double-click on the junction area in the network. The 'Junction' information window will then appear (see Figure 2-6). This window is only intended for displaying information and cannot be edited (editing has to be carried out in the TEDI editor).

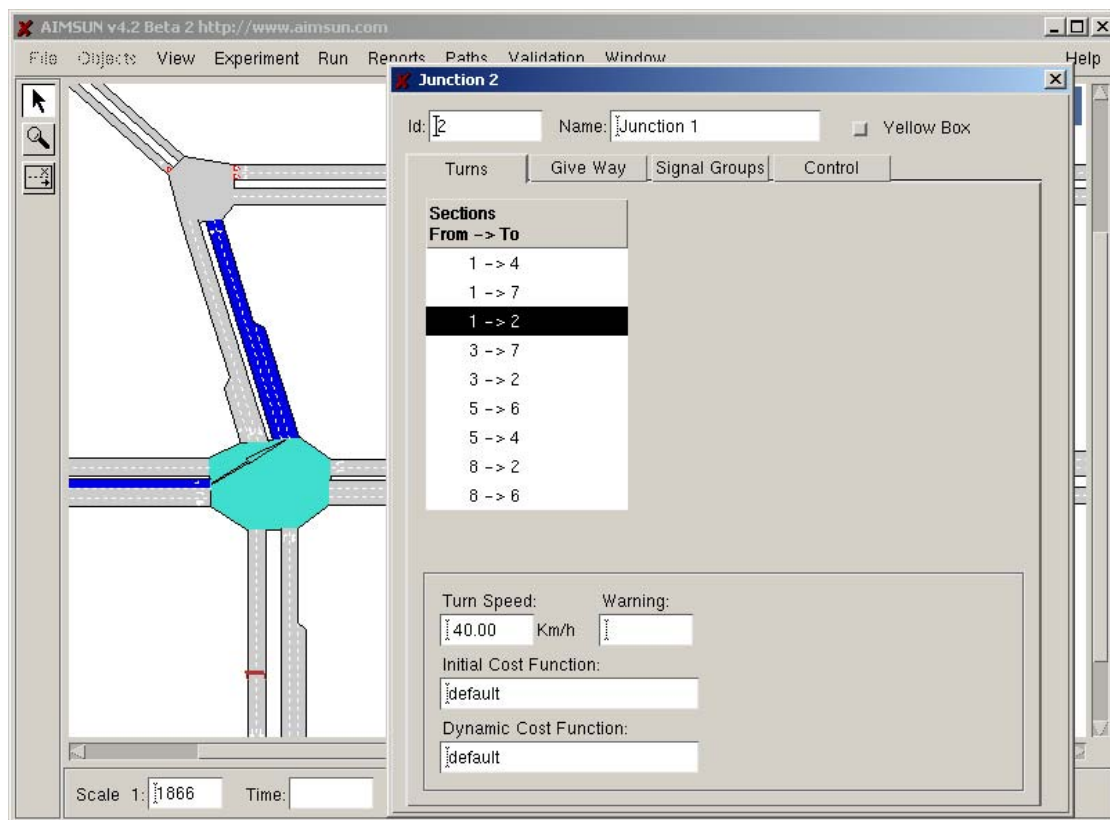
This window contains the junction identifier (integer), the name of the junction (string of characters) and a toggle button indicating whether the junction is to be modelled as a Yellow Box or not. There are also four tab folders: 'Turns', 'Give Way', 'Signal Groups' and 'Control', whose function is to provide more detailed information on these attributes of the junction. The contents of 'Signal Groups' and 'Control' tab folders are explained later in section 5.1.2. The contents of 'Give Way' is explained in section 5.2.2.

Figure 2-6: Junctions Dialog Window

2.2.4 Turning Movements

When you first open the Junctions dialog window, the contents of the 'Turns' tab folder are displayed (see Figure 2-7). A list box contains the list of turning movements for the junction. Each turning movement is given as the pair of identifiers for the origin and destination sections involved in the turning movement. The user may select any turn from the list box by clicking on it and an arrow showing the turning movement will be displayed and the lanes involved in the turn highlighted. The following additional data for the turning movement is then displayed:

- Recommended Turning Speed. Vehicles will try to follow this speed when making this turning movement, although any other specific desired speed of vehicles is also taken into account.
- Traffic Sign ('Warning'), if any, which applies to vehicles making that turn. It can be a Yield sign, a Stop sign or not present.
- Initial Cost Function, is used to calculate the initial cost of this turning movement at the beginning of the simulation. This is only done when running Route Based simulation (see section 4.2.2).
- Dynamic Cost Function, is the function used to calculate the cost of this turning movement at any time during the simulation run. This is only done when running Route Based simulation (see section 4.2.3).

Figure 2-7: Junction-Turns Dialog Window

2.3 SECTIONS

A section is a set of adjacent lanes having the same direction, which connects two nodes. Side lanes, which can be used to model acceleration or deceleration lanes at on-ramps and off-ramps, are also allowed for. Figure 2-8 displays some examples of sections. A section is modelled as a set of section-entities, each one corresponding to a lane (see Figure 2-9). A section is connected to others via a node. Each section contains a set of allowed turning movements, which is also associated with each individual lane.

Figure 2-8: Examples of sections.

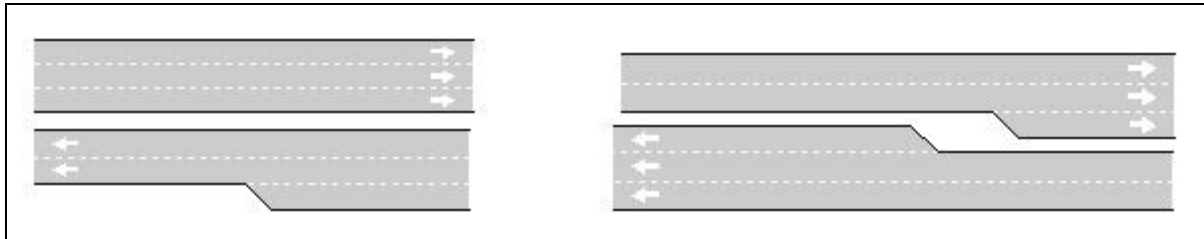
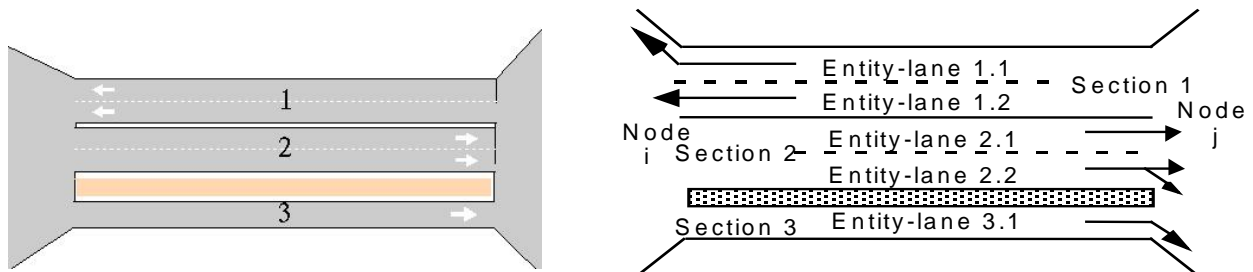


Figure 2-9: Modelling of a section.

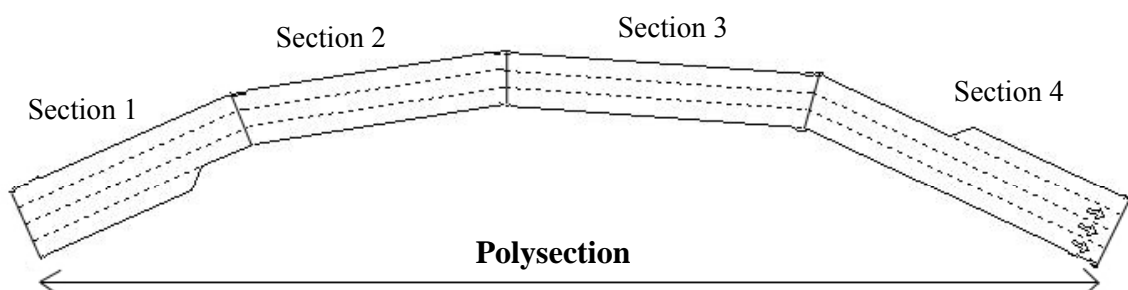


2.3.1 Polysections

Sections may be grouped into polysections (see Figure 2-10). A polysection is a set of consecutive sections connected through joins that satisfies the following requirements:

- All the joins are one-section-to-one-section joins, with equal numbers of origin lanes and destination lanes.
- There may be only one side lane on each side of the polysection, either in the first section of the group or in the last one.

Figure 2-10: Example of a polysection

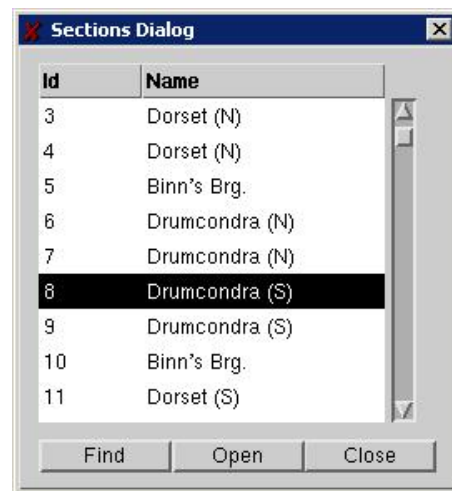


The main purpose of polysections is to enable the user to define longer sections more easily. They can be used to provide an approximation of curved sections, and to achieve more accurate behaviour for the Lane Changing Model (see section 3.5) since longer zones can be defined.

2.3.2 Section Dialog Windows

When you select 'Objects / Sections' from the menu bar, the 'Sections' search window is displayed (see Figure 2-11). The list of section identifiers and section names is shown. To go to a section, select it by clicking on the 'Sections' list box and press the 'Find' button.

Figure 2-11: Sections Search Window



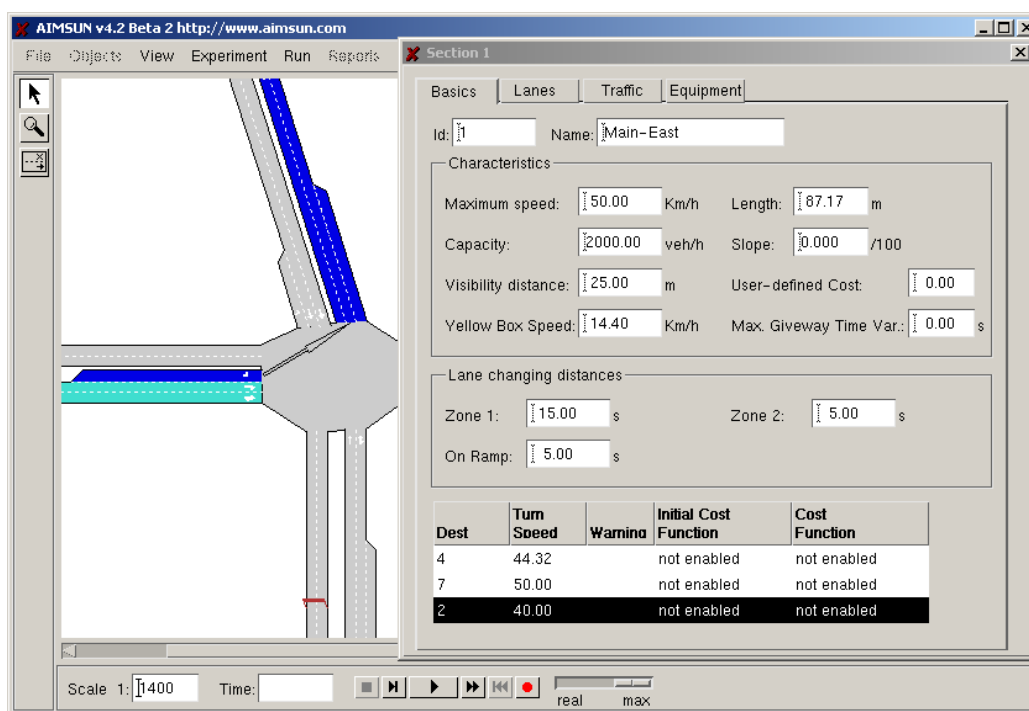
To view the section information, you select a section from the list by clicking on it and then pressing the 'Open' button in the 'Sections' search window. You can also simply double-click on the section area directly in the network. The 'Section' dialog window will appear (see Figure 2-12).

This window is only intended for displaying information and cannot be edited (editing is carried out in the TEDI editor). There are four tab folders: Basics, Lanes, Traffic and Equipment. When you first open the Sections dialog window, the contents of the 'Basics' tab folder are displayed (as shown in Figure 2-12) and the following information appears:

- The section identifier (integer) and the name of the section (string of characters).
- Section Characteristics
 - Maximum Speed: the speed limit for that section. Optionally, the user may define different Speed Limits for each lane in the section (see figure 2-13)
 - Capacity, in vehicle/hour. This is only used in the cost function for the calculation of shortest paths (see section 4.2).
 - Visibility Distance, in meters (see sections 3.4.2 and 5.2)
 - Yellow Box Speed: parameter used in the Yellow Box Model for vehicles approaching to the junction through this section (see sections 2.2.2 and 3.4.2)
 - Length of the section, in meters.
 - Slope: units represent percentage, which is the height corresponding to 100 meters length (see sections 3.4.2 and 3.5.4).
 - User-defined Section Cost: any cost expressed in terms of time representing an economic value associated with the section that can be used in any cost function for calculating route costs.
 - Maximum Give Way Time Variability (see section 3.4.2)
- Lane Changing Distances (see sections 3.4.2 and 3.5.5)
 - Zone 1: distance zone 1, in seconds.
 - Zone 2: distance zone 2, in seconds.
 - On Ramp: distance on ramp, in seconds.
- Turning Information: a list box containing the identifiers of the sections onto which a vehicle can turn from the current section, the turning speed, the sign (yield or stop), if any, Initial Cost Function and the

corresponding Cost Function names. Clicking on any turn in the list box will display an arrow showing the turning movement and the lanes involved in the turn will be highlighted.

Figure 2-12: Section-Basics Dialog Window



The contents of the Traffic tab folder are explained later in section 3.1.2. The contents of the Equipment tab folder (consisting of Detectors, Metering and VMS's) are explained in sections 2.5.2, 2.5.1 and 2.5.3, respectively.

2.3.3 Reserved Lanes

Reserved lanes can be defined in any section. By reserved lanes we mean those lanes where entry is only permitted to vehicles that belong to a certain class (a set of vehicle types). For each section, at most one set of reserved lanes can be defined. The lanes must be adjacent and at one side of the section. Two different types of reserved lanes are allowed: optional and compulsory.

An optional reserved lane means that allowed vehicles may use it or not, while disallowed vehicles cannot use it at all. If an allowed vehicle is in the lane, the vehicle does not have to leave it; if an allowed vehicle is not using the lane, the vehicle may use it to overtake or whenever it is needed. The vehicle does not, however, have to make a special effort to enter the lane. Disallowed vehicles would be compelled to leave a reserved lane and will not enter it unless required for some turning movement. This type of reserved lane can be used, for instance, to model trucks being forbidden to overtake in the left lane.

A compulsory reserved lane means that allowed vehicles must use it, while disallowed vehicles cannot use it at all. Allowed vehicles must try to reach the lane if possible, and when they get into it, they must remain inside. Disallowed vehicles behave in the same way as in the optional reserved lanes. This is the type of behaviour required to model bus lanes properly.

In both cases, non-allowed vehicles can only enter reserved lanes in exceptional circumstances (blocking of other lanes, turning movements only allowed in these lanes, etc).

Information about Reserved Lanes is shown in tab folder 'Lanes' of the Section dialog window (see Figure 2-13). If there are any reserved lanes in this section, the type of reserved lane, either compulsory or optional, and the name of the allowed class are displayed.

Figure 2-13: Section-Lanes Dialog Window

Lane	Solid Left Lines		Solid Right Lines		Speed
	From	To	From	To	
1	30.00	94.97			40.00
2					
3					

Reserved lanes

Type:

Class:

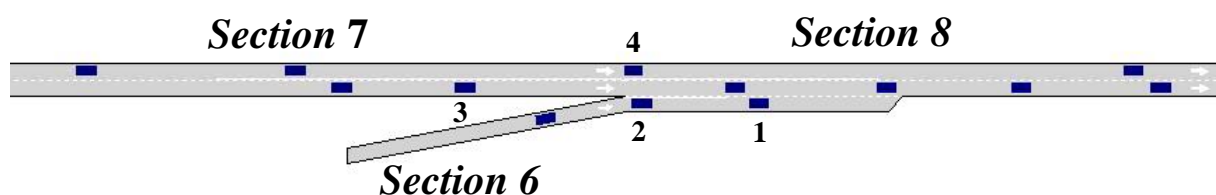
2.3.4 Solid Lines

Restrictions on changing lanes are modelled by defining solid lines between lanes. Lane changing is not allowed wherever there is a solid line between two lanes. Using this feature the user may avoid undesired lane changing at conflicting points, such as around the area of an on-ramp.

For each pair of adjacent lanes the user may define a solid line to the right, to the left or on both sides. Where no solid line is defined, a dashed line is assumed. A solid line on the right side of a lane means that no lane change is allowed from the right side of that lane to the lane on the right.

A solid line on the left side means that no lane change is allowed from the left side of the lane to the lane on its left. Finally, a solid line on both sides means that no lane change at all is allowed between the lanes. Solid lines can be defined for the whole lane length or only part of it. Figure 2-13 shows the Lanes Folder of the Section dialog window in which information about solid lines is displayed in the AIMSUN interface.

Solid lines are drawn on the section so that the user can identify them directly in the network. Figure 2-14 shows the graphical representation of section 8 corresponding to the data displayed in the dialog window in Figure 2-13. In this example, there is a short solid line between the on-ramp lane (lane number 1) and the rightmost main lane (lane number 2), so vehicles cannot merge from the on-ramp in the first 20 meters, (as is the case for vehicle 2). There is also a solid line between the left lane (lane number 3) and the right lane, intended to create more gaps for the approaching on-ramp. This is the case for vehicle 4, which is not allowed to change to the right lane, since this would disrupt merging for vehicles 1 and 2.

Figure 2-14: Example of Solid Lines

2.4 CENTROIDS

2.4.1 Network Zones

Section and node objects can be grouped into zones. Each zone corresponds to a centroid. A centroid can be considered as a traffic source, a traffic sink or both. Consequently, a centroid may have a set of nodes and/or sections connected 'TO' and 'FROM' it. A connection 'TO' means that vehicles are going TO the object from the centroid. This is a source or origin. A connection 'FROM' means that vehicles are going FROM the object to the centroid. This is a sink or destination.

Assignment of a vehicle to a section depends on how the zone is represented in the network. Two levels of representation are possible: macroscopic and microscopic.

In the macroscopic representation, a zone is seen as a special node (centroid node), which constitutes the start and/or the exit of a list of sections (arc). Incoming arcs model the sections that lead to an entry gate for the zone. Outgoing arcs mark the exit gates. Note that this level of representation does not allow a centroid to appear as an intermediate node in a route. This is carefully considered when the system is calculating the shortest routes. The outgoing arcs serve as entities where the vehicles finish their trip and are then absorbed (taken out of the network). When a vehicle is generated within a zone, it is assigned to the exit section specified by its route. An example of this level of representation is displayed in Figure 2-15.

On the other hand, at the microscopic level, the details for all zones are modelled. The concept of a zone exists only to model the demand. Hence, the departure point of a vehicle is a section belonging to the zone where it was generated. The demand is subdivided into a set of sections belonging to a zone. The subdivision is carried out on the basis of probability by using a chosen probability distribution (e.g. uniform distribution). This level of representation requires the calculation of shortest routes from every section to all destination sections. An example of this level of representation is illustrated in Figure 2-16.

Figure 2-15: Centroids connected to Section

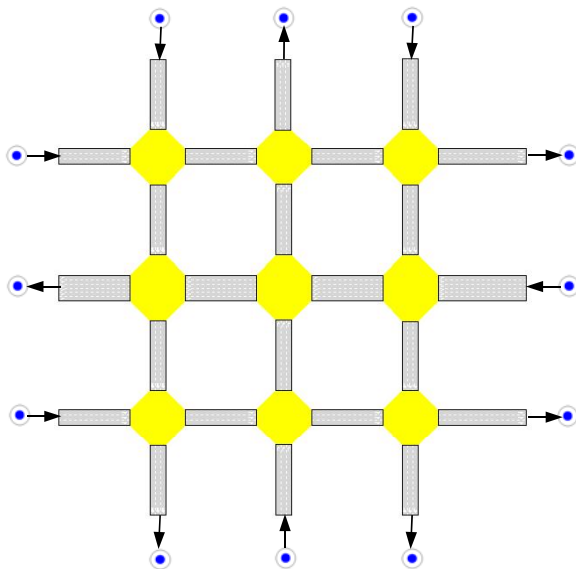
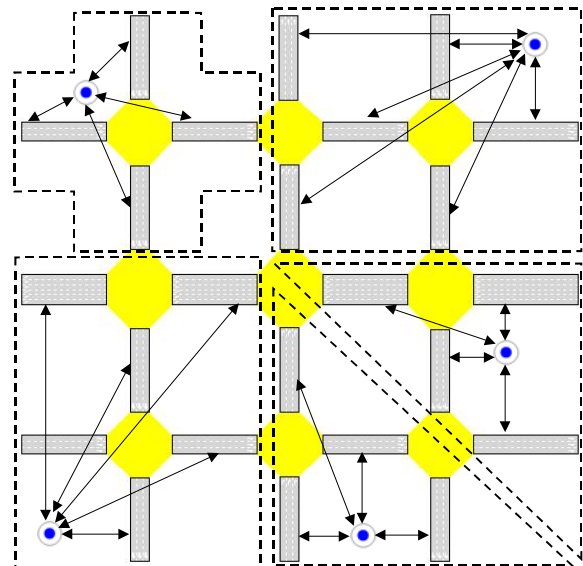


Figure 2-16: Centroids connected to zones

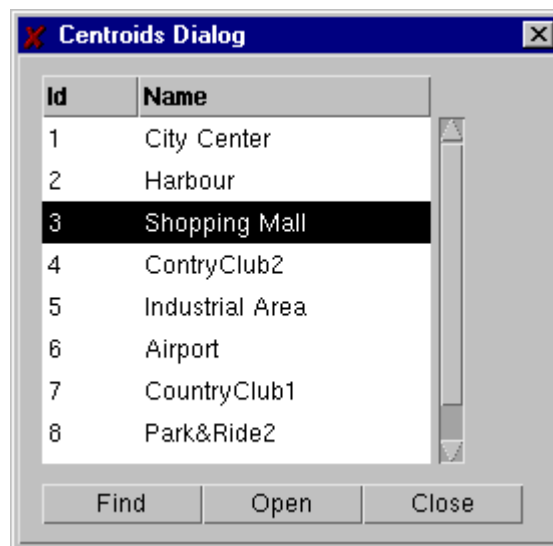


O-D matrices are related to centroids. An O-D matrix gives the number of trips from each origin centroid to each destination centroid, for each time slice and for each vehicle type. Chapter 4 of this document describes how vehicles are assigned to the different objects (i.e., sections and nodes).

2.4.2 Centroid Dialog Windows

When you select 'Objects / Centroids' from the menu bar, the 'Centroid Dialog' search window is displayed (see Figure 2-17). The list of centroid identifiers and centroid names is shown. To pan to a centroid, select it by clicking on the list box and press the 'Find' button.

Figure 2-17: Centroid Search Dialog Window



To view the centroid information you can either press the 'Open' button in the 'Centroids Dialog' search window after selecting one of them, or simply double-click on the centroid icon directly in the network. The 'Centroid Dialog' information window (shown in Figure 2-18) then appears. This window is only provided for the purpose of displaying information and cannot be edited (editing has to be carried out in the TEDI editor). The information presented in this window is the following:

- Centroid identifier (a number).
- Centroid name (a string of characters).
- Two toggle buttons indicating how the vehicles are going to be distributed among the objects (sections or nodes) connected to/from the centroid when they enter/exit the network. There are two toggles: Use Origin Percentages and Use Destination Percentages (see section 4.7.2)
- A list box containing the connections information. The following data is provided for each connection:
 - Type of connection: a connection 'to' means from the centroid to the object, this is an origin'. A connection 'from' means from the object to the centroid, this is a 'destination'.
 - Type of object: section or node.
 - Object identifier: identifier number of the section or node.
 - Percentage: percentage of vehicles generated/attracted by this centroid that will use this connection to enter/exit the network, only if the 'Use Origin/Destination Percentages' option has been selected.
 - Position in the object where the physical connection has been made.
- The demand data from the O/D matrix for this centroid, (only if it is an origin, see section 3.1.3.)

In previous versions of AIMSUN GUI, the centroid connections were only drawn when the centroid was selected and there was no difference between from and to connections. Now the user can choose if he wants to always see them or not, in the dialog Show Objects. Furthermore, From connections are drawn in black and To connections in blue, as in Tedi (see figure 2-19).

Figure 2-18: Centroid Dialog Window

Centroid 2

Id: Name:

☐ Use Origin Percent. ☐ Use Dest. Percent.

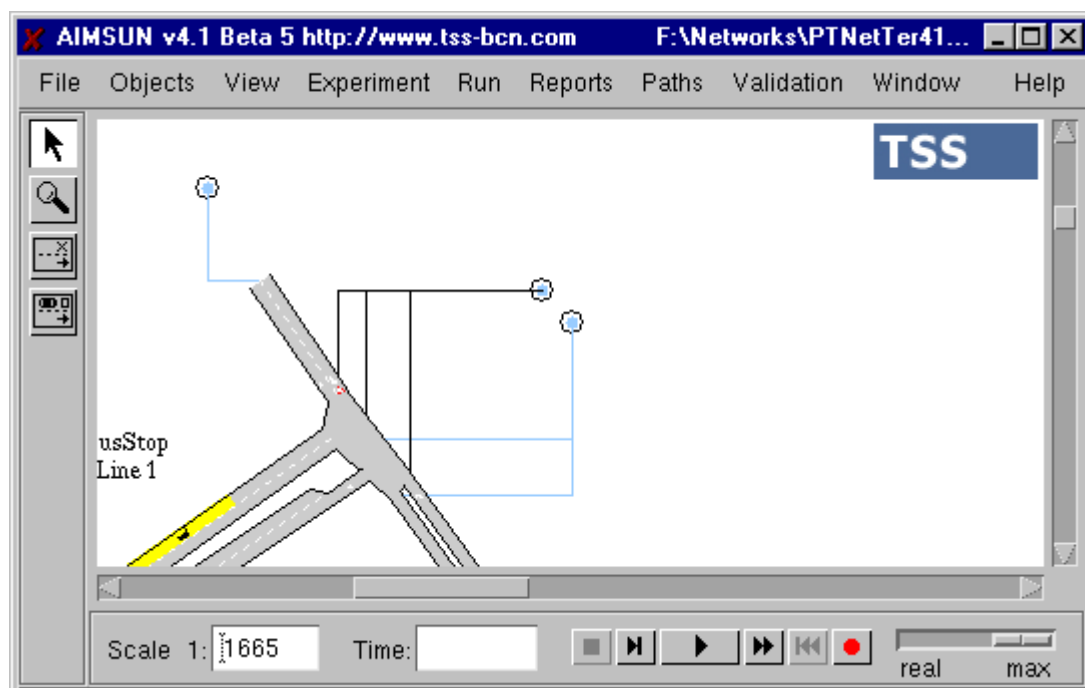
Connections

Type	Object	Id	%	Position
To	Section	908	100.000	1.837
From	Section	907	100.000	15.677

OD Matrix

Time Slice	Dest.	Total #Trips
09:00:00	11	0.2
09:15:00	23	1.1
09:30:00	24	1.0
09:45:00	25	1.3
	26	1.6

Vehicle Type	#Veh.
car	1.2
LGV	0.1
OGV1	0.0
OGV2	0.0

Figure 2-19: Centroid Connections

2.5 NETWORK EQUIPMENT

AIMSUN can model most of the devices that can be found in any real traffic network. Traffic signals, traffic signs, ramp metering, detectors and variable message signs (VMS) are all taken into account.

2.5.1 Traffic Control Devices

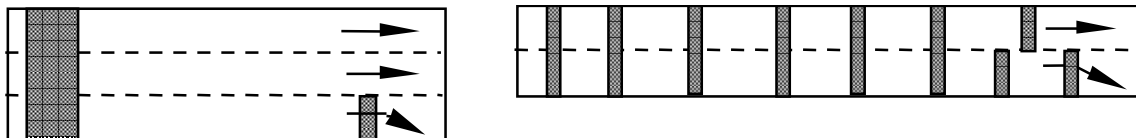
These are some of the standard basic objects that are required to implement traffic control in the real traffic network. These objects, which are explained in greater detail later in chapter 5, are the following:

- **Traffic Signals.** These are always related to junction node types and it is assumed that the stop line is always located at the end of the approaching sections. The control plan is defined at the level of the whole intersection. This is done by specifying a set of phases, the signal groups affected for each one and the set of permitted turning movements.
- **Traffic Signs.** In order to define rights of way for turns at unsignalised intersections, two types of signs are considered: give-way, or yield, signs and stop signs.
- **Ramp Meters.** This type of control is used to limit the input flow at ramps entering certain type of roads, such as freeways, urban motorways, etc. This is done with the purpose of maintaining certain levels of safety, as well as guaranteeing free flow in the main traffic stream. Ramp meters usually appear in sections approaching juncture-type nodes and they affect all the lanes in the section.

2.5.2 Detectors

Another object included with the network equipment that can be modelled by AIMSUN is the traffic detector. A traffic detector is any device that is able to take measurements of traffic variables. A detector can be located at any point inside a section, it can cover one or more lanes and be of any required length. A section can have more than one detector, but a detector must belong only to one section. Figure 2-20 provides some examples of detector layouts. AIMSUN can model different types of traffic detectors, which are defined by their measurement capability: vehicle counts, speed, presence and occupancy.

Figure 2-20: Examples of detector layouts



The two main types of detection implemented are the Common Detection Model and the ETCMS (External Traffic Control System) Detection Model. Common detection output data is produced by AIMSUN periodically, provided that detectors are defined in the network and the user has set the Aggregated Detection toggle button to ON in the Detection Parameter Window before running the simulation experiment and providing a detection interval has been defined. ETCMS detection output is produced the user sets the Cycle Detection toggle button to ON in the Detection Parameter Window before running the simulation experiment and providing a detection interval

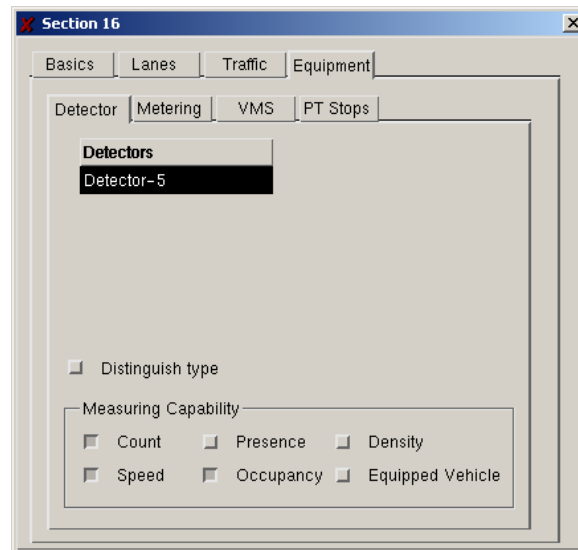
In the Common Detection Model, the data produced is stored in ASCII files, and depends on the measuring capabilities of the detectors. There is a data line for each detector, which contains the detector identifier and the list of data gathered. These may include Count (number of vehicles per interval), Occupancy (percentage of a time interval a vehicle is measured by the detector), Speed (mean speed for vehicles crossing the detector during the interval) and Density (number of vehicles per kilometre), and Headway (mean headway for vehicles crossing the detector during the interval). When there are detectors with 'Equipped Vehicle' capability, the AIMSUN generates a log of all Equipped vehicle detected during the simulation, providing the detection time, the detector identifier, the vehicle identifier, the vehicle type name and the public transport line identifier if it is a public transport vehicle.

In the ETCMS type Detection Model, the data are given at every cycle (could be different from the simulation step). The data gathered are Count (number of vehicles), Speed (mean speed for vehicles crossing

the detector during the step), Occupancy (percentage of time step the detector is pressed), Presence (whether a vehicle is over the detector or not), Density (number of vehicles per kilometre) and Headway (mean headway for vehicles crossing the detector during the interval).

Detectors are always associated with Sections because they can only be located inside a section. When you select the 'Equipment' tab folder in the 'Section' dialog window, four additional tab folders appear. The first is the 'Detector' tab folder, which contains the information about the detectors defined in that section (see Figure 2-21). A section can have more than one detector. Consequently, a list of detectors for that section is displayed. Selecting a Detector from the list box will activate the toggle buttons corresponding to the Measuring Capabilities for that detector: count, speed, presence, occupancy and density. The toggle button indicating whether the detector can distinguish between different vehicle types is also displayed.

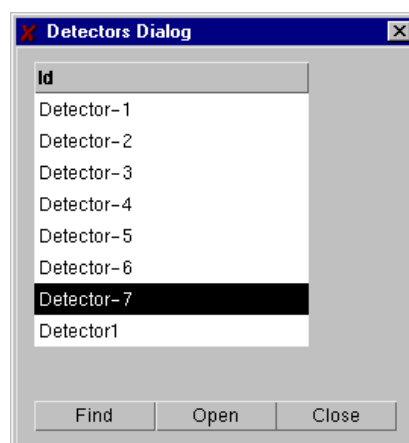
Figure 2-21: Section-Detectors Information window



You can also access Detector Information by using the 'Objects / Detectors' command. This displays a 'Detectors' search window (shown in Figure 2-22). The user can search for a detector by clicking on the name in the list box and pressing the Find button. When you press the 'Open' button, the 'Detector' information window (see Figure 2-23) is displayed. Double-clicking directly on the detector image in the network has the same effect.

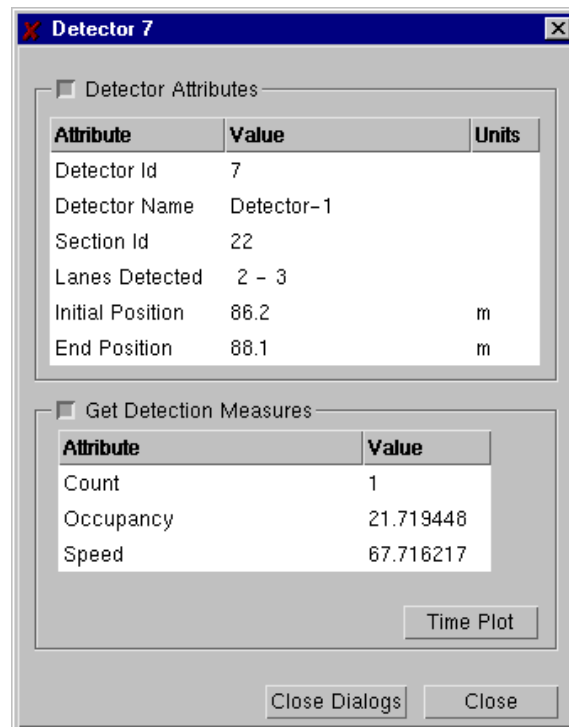
This window provides some additional attributes of the detector, such as the lanes covered by the detector (numbered from right to left) and the initial and final position of the detector in the section, measured from the beginning of the section. It can also be used to graphically monitor the data gathered and detection measures cycle by cycle (see section 9.2 for a detailed description).

Figure 2-22: Detector search window



If the network contains any traffic detectors, the user can select which detection model to apply by using the 'Experiment / Output / Detection' command from the menu bar (see section 9.2 for a description of the Detection Output Data).

Figure 2-23: Detector Information window

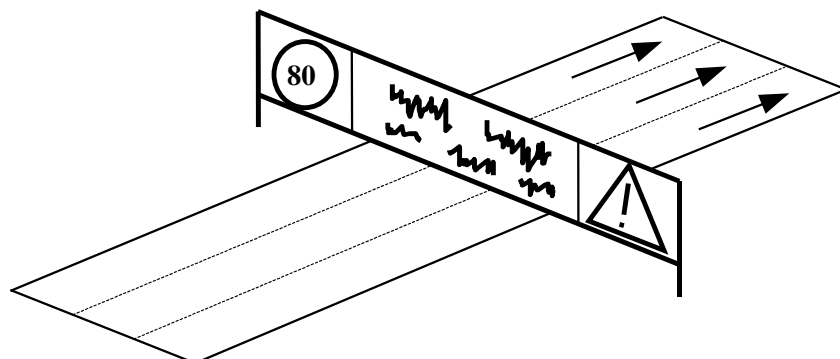


2.5.3 Variable Message Signs (VMS)

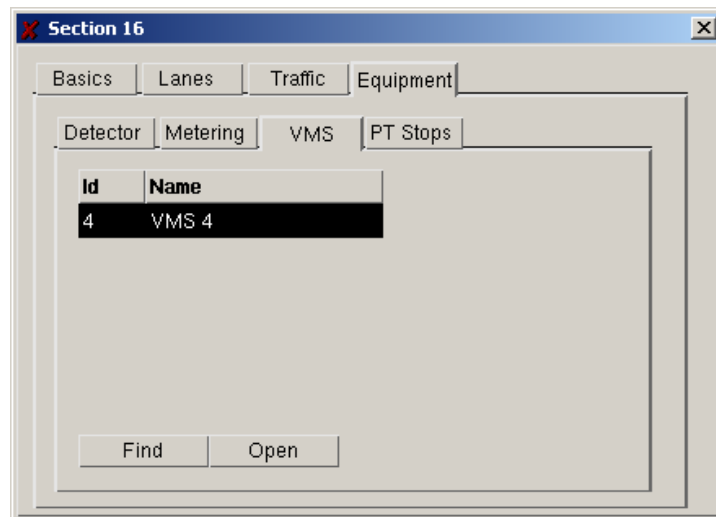
VMS's are display panels, located on the sections, which are used to provide information and recommendations for drivers and are even used to prohibit certain behaviour (see Figure 2-24). Each VMS has a set of possible messages and each message has a set of actions, which represent the influence the message has on the driver's behaviour.

The types of message that can be modelled include modifications to speed limits, recommendations for alternative routes and information on congestion or incidents. The modeller can define a set of actions, to represent the impact of these messages. This is described in section 5.4.

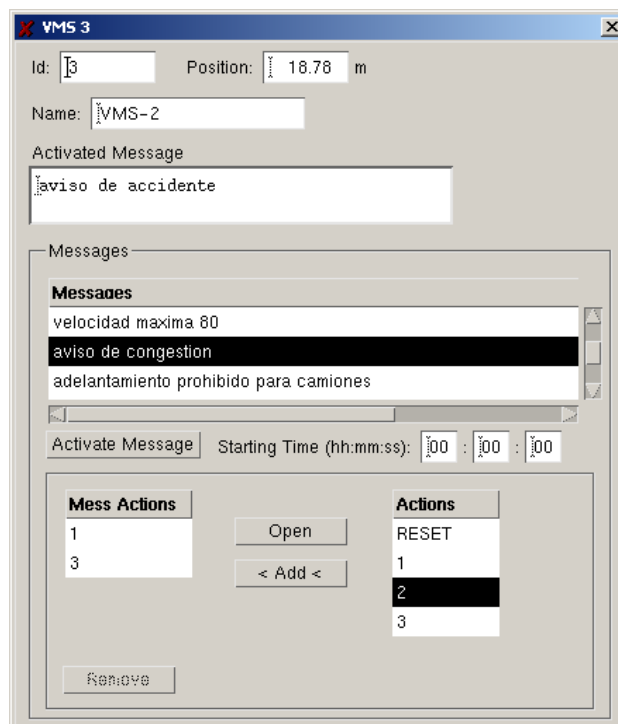
Figure 2-24: Variable Message Sign



The user may access VMS information via the 'Sections' dialog window. If you select the 'Equipment' tab folder, three additional tab folders are displayed. One of these is the 'VMS' tab folder, which contains the information about the VMS defined in that section (see Figure 2-25). A section can have more than one VMS. Consequently, a list of VMS for that section is displayed.

Figure 2-25: VMS Information window

You can also access VMS information by selecting the 'Objects / VMS's' command. The 'VMS's' search window (similar to the Detector search window displayed in Figure 2-21) appears, displaying the list of VMS names. You can go to a VMS by selecting it or by clicking on the list box and then pressing the 'Find' button. To view the VMS information, you press the 'Open' button in the 'VMS's' search window or simply double-click on the VMS directly in the network. In both cases the 'VMS' information window is displayed (see Figure 2-26).

Figure 2-26: VMS Information window

The information displayed in this window includes the following:

- VMS Identifier (numerical id.).
- VMS name (a string of characters).
- Position where it is located in the section. This is measured as the distance from the entrance point of the section to the VMS.

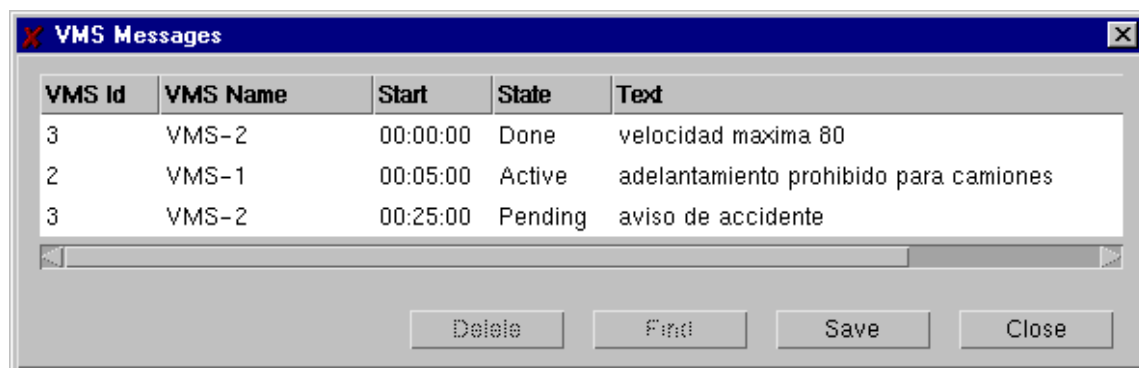
- Current Activated Message, if any. In AIMSUN, Messages are considered as being alphanumeric. A pictogram message, however, (e.g. a speed limit sign or a warning sign) can be assumed to be represented by a string of characters (e.g. 'Max. Speed = 60' or 'Warning').
- List of possible messages for this VMS. Only messages included in this list may be accepted for display by this VMS.
- Starting Time. This is the time at which the message must be activated.
- List of all Actions available for this network (explained in section 5.4).

When selecting a message, a list of Actions associated with that message appears in the list box named 'Mess. Actions'. The list box named 'Actions' contains all actions available for this network. The user can add actions to the list of actions associated with a message by selecting an action from the 'Actions' list box and clicking on the 'Add' button. This button turns into a 'Remove' button when an action from the 'Mess. Actions' list box is selected instead. Actions are explained in section 5.4.

The user can activate any message from the messages list box by clicking on it and pressing the 'Activate Message' button. Messages cannot be edited via this window, only activated. By default, the current simulation time is displayed in the 'Starting Time' dialog window, which means that the message will be activated at the current time. Activating a message at the current simulation time will cause the message to be displayed as Activated Message and the Actions associated with it will be implemented.

The user can also edit the value of 'Starting Time' to specify at what time during the simulation the message has to be activated. Therefore, the user may define a sequence of messages to be activated on the different VMS's during the simulation run. This can be done at any time during the simulation run, either via the AIMSUN graphical user interface or via the GETRAM Extensions. The user may also monitor the activated messages. This is done using the 'Experiment/VMS Messages' command, which displays the dialog window displayed in Figure 2-27.

Figure 2-27: VMS Messages Log Dialog Window



The VMS Messages dialog window provides a list of messages that have been, are, or will be activated in the current simulation experiment. The following information is provided:

- Identifier and Name of the VMS where the message is displayed.
- Text of the message.
- Starting Time when the message is to be activated.
- State of the message: Done (it has been displayed in the past), Active (it is currently displayed) and Pending (it is to be displayed in the future).

The Messages Log file can be stored using the 'Save' button in the VMS Messages dialog window. Saved message log files can be loaded and used in other simulation experiments. This is done by clicking on the 'Load Initial Messages' toggle button, which you can find in the Load Traffic Results and Load O/D Matrix dialog windows.

Messages can be deleted from the list using the 'Delete' button. Deleting an active message will not cause the message to be cleared from the VMS, but only to be removed from the list. There is also a 'Find' button to locate the corresponding VMS in the network.

3. TRAFFIC MODELLING

3.1 TRAFFIC DEMAND DATA

Depending on the available traffic demand data, two different types of simulation are considered in AIMSUN. One is based on input traffic flows and turning percentages, the other is based on O-D matrices and routes or paths.

3.1.1 Vehicle Classification

Regardless of the type of simulation, vehicles can be grouped at two different levels: vehicle classes and vehicle types.

Vehicle Classes are only used for the reserved lane definition. For instance, these can include Public, Private, Emergency and HOV (high occupancy vehicle) classes, in order to be able to define reserved lanes in the network model for Public and Emergency classes as well as lanes for the HOV class. The use of Classes is optional, it is only required when there are reserved lanes in the network model.

Vehicle Types refers to the different kinds of vehicles for which the traffic demand data is defined. Therefore, input flows and turning proportions as well as number of trips in the OD matrices are distinguished for each vehicle type. For instance, vehicle types may be car, taxi, private-bus, public-bus, HGV (heavy goods vehicle), truck, ambulance, police-car, HOV-car. The use of Types in a model is always required, otherwise it is not possible to define traffic demand data. Physical characteristics or vehicle attributes such as width, length, speed, acceleration, deceleration, etc, may be defined for every Type.

A Class can be composed of one or more Types, and a Type may belong to none, one or several Classes. For example:

<u>Class</u>	consists of	<u>Types</u>
Public		public-bus, taxi, ambulance, police-car
Private		car, private-bus, HGV, truck, HOV-car
Emergency		ambulance, police-car
HOV		HOV-car

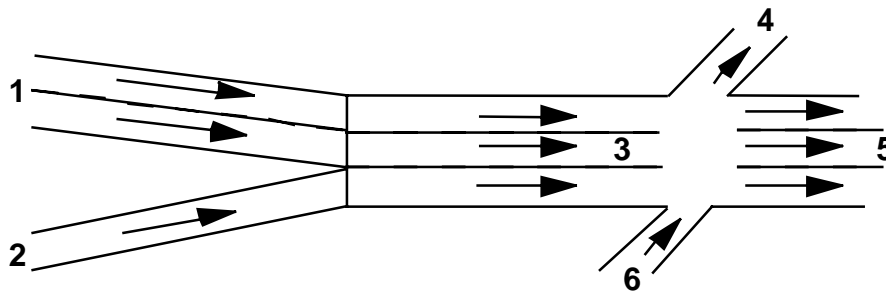
A reserved lane may be defined for a certain class. For instance, a reserved lane may be defined for Public transport. This lane would only be available for public-buses, taxis, ambulances and police-cars.

3.1.2 Input Flows and Turning Proportions

The traffic demand data is composed of the input flows at the input sections of the network and the turning proportions at every node of the network. Both can be defined by vehicle type. These data can be obtained as a result of a previous assignment model, from data collected by detectors or defined by the user as an experimental hypothesis.

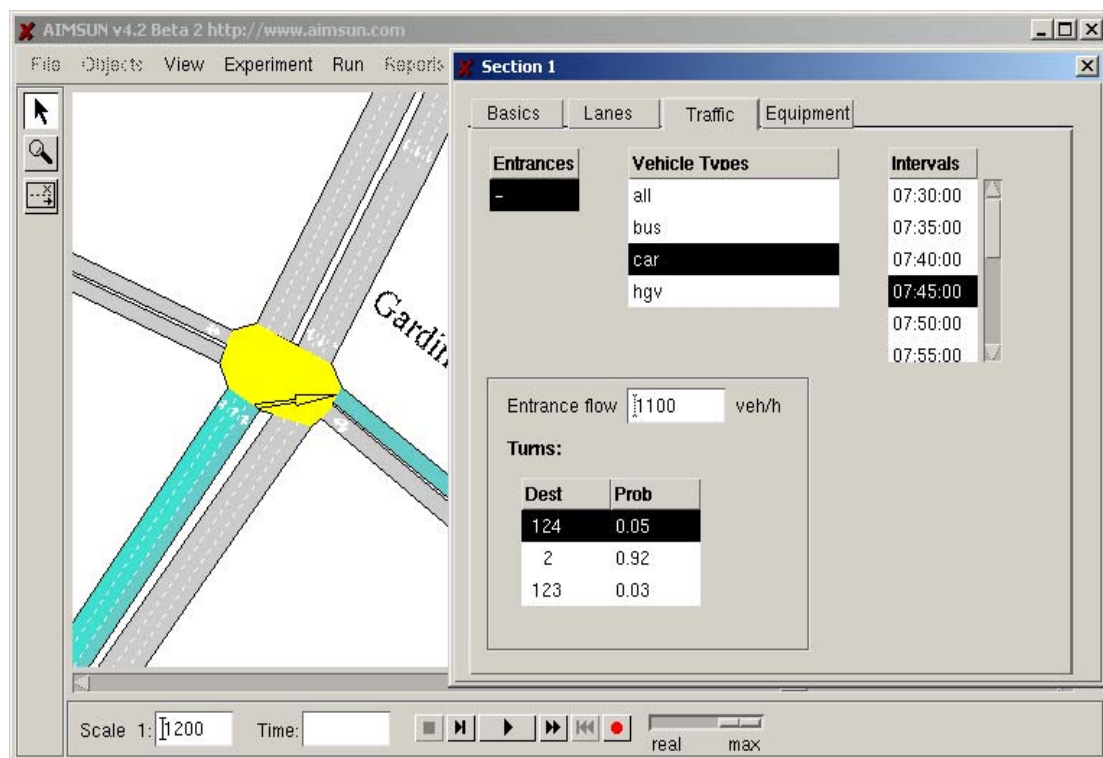
Vehicles are generated and input into the network through the input sections, following a random generation model (see section 3.2) based on the mean input flows for those sections. They are then distributed randomly throughout the network in accordance with the turning proportions defined at each section of the network. This means that vehicles do not “know” their complete path along the network, but only about their next turning movement.

AIMSUN can distinguish between the different entrances to a section, so different turning proportions can be defined. Each turning proportion will affect only the vehicles entering the section from each different entrance. Figure 3-1 gives an example of this. Section 3 has two possible entrances: sections 1 and 2. Therefore, the turning proportions for section 3 to sections 4 and 5 may be different for vehicles coming from section 1 to those coming from section 2. The same happens with section 5, whose entrances are sections 3 and 6.

Figure 3-1: Entrances to a Section

Intervals between arrivals are determined from the mean flows of the input sections, applying any of the random distributions explained in section 3.2. Vehicle type is assigned as a function of the proportions defined for each section. The input lane depends on the state of all lanes composing the section, the existence of reserved lanes and the vehicle type. Rightmost lanes (or leftmost lanes, depending on the type of driving) are more likely to be used than central lanes.

When a Traffic Result has been loaded, you can consult the traffic demand data for a section by selecting the 'Traffic' tab folder (see Figure3-2). If an OD matrix has been loaded instead of a Traffic Result, the 'Traffic' tab folder will contain no data.

Figure 3-2: Traffic Information Window

This traffic information window contains the input traffic flows and/or the turning proportions. You can view it by selecting an entrance section by clicking on an identifier in the 'Entrances' list box. If the section is an input section, then there will be no entrance sections and therefore only a dash will appear in the 'Entrances' list box. Then select a vehicle type from the 'Vehicle Types' list box in a similar manner, and finally a time interval from the 'Intervals' list box. The entrance flow for this section and the turning proportions for each possible destination section will be shown. If the section is not an input section, no entrance flow will be displayed. Clicking on any turn in the list box displays an arrow showing the turning movement as well as highlighting the lanes involved in the turn.

3.1.3 O-D Matrices and Routes

The traffic conditions to be simulated are defined by an O-D matrix, which gives the number of trips from every origin centroid to every destination centroid, for each time slice and for each vehicle type. Vehicles are generated at each origin centroid and input into the network via the objects (sections or nodes) connected as 'TO' this source centroid. Then, vehicles are distributed along the network following shortest paths between origin and destination centroids. Finally, vehicles exit the network via the objects connected as 'FROM' the destination or sink centroid.

When a vehicle is generated, the assignment of the vehicle to the objects connected to the centroid (i.e. sections and nodes) can be made based on probability, or made to depend on the path to destination. Using the probability-based approach, the user specifies a proportion of vehicles taking each of the possible objects connected to the centroid. On the other hand, using the destination-dependant approach, the system decides to which object each vehicle must be assigned, taking into account the best path to the actual destination of the vehicle.

O/D Matrix information can be accessed via the Centroid dialog window (see Figure 3-3), which can be opened either with the 'Objects / Centroids' command from the menu bar or just by double clicking on a centroid.

This dialog window displays the demand data from the O/D matrix for this centroid, only if it is an origin. This information is presented in various list boxes:

- Time Slice: time (hours, minutes and seconds) at the beginning of the time slice.
- Destination Centroids reached from this origin and the total number of trips going to each destination during the time slice.
- Distribution of the total demand for each pair O-D among the different vehicle types.

To obtain the information, first select one time slice by clicking in the 'Time Slice' list box. The Destinations and Total Trips for this time slice will appear in the 'Dest.' list box. Then, to view the Vehicle Types distribution, click on a destination in the 'Dest.' List box, and this information will be displayed in the 'Vehicle Type' list box.

Chapter 4 explains in more detail how this type of O/D matrix and paths-between-centroids simulation works.

Figure 3-3: Centroid Dialog Window

Centroid 5

Id: 5 Name: Sections15_16_20_21

☐ Use Origin Percent. ☐ Use Dest. Percent.

Connections

Type	Object	Id	%	Position
To	Section	15	50.000	0.000
To	Section	16	50.000	0.000
From	Section	20	50.000	0.000
From	Section	21	50.000	0.000

OD Matrix

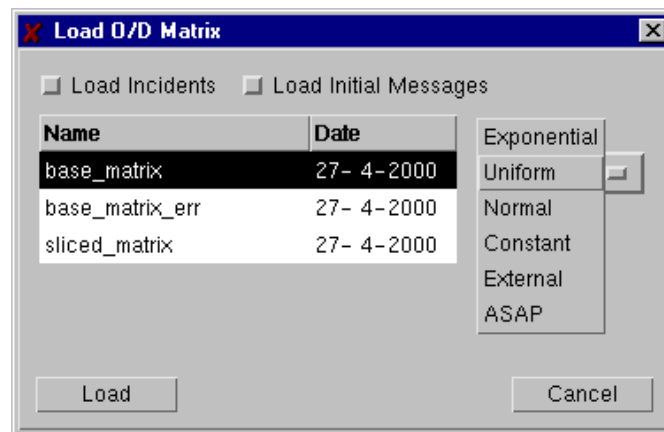
Time Slice	Dest.	Total #Trips
00:00:00	1	25.0
00:15:00	6	30.0
00:30:00	7	25.0
00:45:00	9	25.0

Vehicle Type	#Veh.
bus	0.0
car	30.0

3.2 TRAFFIC GENERATION

The time interval between two consecutive vehicle arrivals –headway– is sampled from a random distribution –headway model. When loading either a traffic result or an OD matrix into AIMSUN, the user may select among different headway models: exponential, uniform, normal, constant, “ASAP” and external. This is done via the option menu ‘Arrivals’ that appears both in the ‘Load Traffic Result’ and ‘Load O/D Matrix’ dialog windows (see Figure 3-4). ‘Exponential’ is the default distribution when simulating with traffic results, while ‘uniform’ is the default distribution when using the OD matrix.

Figure 3-4: Headway model selection



3.2.1 Exponential

Time intervals between two consecutive vehicle arrivals (headway) at input sections are sampled from an exponential distribution [COW75]. The mean input flow (in vehicles/second) is λ , and the mean time headway is calculated as $1/\lambda$ seconds.

The algorithm for calculating the time headway (t) is the following:

```

u = random (0,1)
if ( $\lambda > 0.0$ )
    t = ((-1/ $\lambda$ )*ln(u))
else
    t = max_float
endif

```

3.2.2 Uniform

Time intervals between two consecutive vehicle arrivals (headway) at input sections are sampled from a uniform distribution. The mean headway (T) is taken as $1/\lambda$ seconds, being λ the mean input flow (in vehicles/second), and the range used for the uniform is $[T-T/2, T+T/2]$.

The algorithm for calculating the time headway (t) is the following:

```

if ( $\lambda > 0.0$ )
    T = 1/ $\lambda$ 
    u = random (0,1)
    minU = -T/2
    maxU = T/2
    t = T+[minU+(maxU-minU)*u]
else
    t = max_float
endif

```

3.2.3 Normal

Time intervals between two consecutive vehicle arrivals (headway) at input sections are sampled from a truncated normal distribution. The mean headway (T) is taken as $1/\lambda$ seconds, where λ is the mean input flow (in vehicles/second), and the variance (σ) is taken as 10% of the mean. The range of the truncated normal is $[T-2*\sigma, T+2*\sigma]$.

The algorithm for calculating the time headway (t) is the following:

```

if ( $\lambda > 0.0$ )
     $T = 1/\lambda$ 
     $n = t\_normal(1, 0.1)$           (truncated normal)
     $t = \text{maximum}[\varepsilon, n*T]$       ( $\varepsilon \rightarrow 0, \varepsilon > 0$ )
else
     $t = \text{max\_float}$ 
endif

```

3.2.4 Constant

Time intervals between two consecutive vehicle arrivals (headway) at input sections are always constant. The headway (t) is taken as $1/\lambda$ seconds, being λ the mean input flow (in vehicles/second).

The algorithm for calculating the time headway (t) is the following:

```

if ( $\lambda > 0.0$ )
     $t = 1/\lambda$ 
else
     $t = \text{max\_float}$ 
endif

```

3.2.5 “ASAP”

In this generation model, vehicles are entered in the network ‘as soon as possible’, i.e. as soon as there is some space available in the input section. This model is intended to make the most use of the network entrance capacity. It could be used, for example, for simulating evacuation situations. In this case no headway is generated. At the beginning of each time slice, the total flow to be input during the slice is ‘piled up’ at the entrance section and vehicles are entered in the section one after the other as soon as the previous one has left enough space.

3.2.6 External

This option means that the user will introduce the vehicles into the network via the GETRAM Extensions module. No vehicles are generated or input into the network via any section by the simulator itself. Therefore an external DLL, user-defined program, is required to feed the network with vehicles.

3.2.7 Dealing with a fractional number of trips

When the demand for a particular OD pair during a certain time slice is less than 1, it is possible that no vehicle is scheduled for that particular slice. This apparently slight error may become very significant when the simulation period is composed by a lot of time slices, or when there are a lot of OD pairs containing these small amounts. This can cause a situation in which a representative number of trips are not being generated.

This problem can be attributed to the nature of the vehicle generation process in AIMSUN. To calculate a vehicle arrival time, the time intervals between two consecutive vehicle arrivals (headway) are sampled from any of the available random distributions. When the flow is small, this time headway can be bigger than the time slice itself, which leads to no arrival during that slice, but during the next slice instead. However, when the simulation reaches the new slice, new arrivals according to the new input demand are scheduled and the pending arrivals are removed.

For example, let us assume that $\lambda=0.2$ vehicles/minute is the mean input demand for a particular OD pair. The mean time headway is calculated as $1/\lambda = 5$ minutes. If we have a time slice of 1 minute, this means that during this time slice no vehicles would arrive, as the scheduled time for the next arrival goes beyond the end of the time slice.

To solve this problem, the generation process is recalculated for every time slice and is carried out for every vehicle type. In each slice, AIMSUN calculates the generation times for a specific vehicle type according to the following algorithm:

GenerationTimePending: represents the first arrival time that is greater than the end of time slice

NextTimeGeneration: represents the next time of arrival in the slice

Headway: represents the time interval of the next arrival according to the chosen distribution

NextTimeGeneration = Initial Time of Slice i-th

while (NextTimeGeneration <= End Time of Slice i-th) **do**

if (it is the first slice) **then**

 NextTimeGeneration = PreviousTimeGeneration + Headway /2

else

 NextTimeGeneration = PreviousTimeGeneration + Headway

if (it is the first generation in the slice) **then**

 NextTimeGeneration = Minimum(NextTimeGeneration,
 GenerationTimePending)

endif

endif

 PreviousTimeGeneration = NextTimeGeneration

enddo

GenerationTimePending = PreviousTimeGeneration

Example

Consider a situation with four time slices, each one has 10 minutes of duration and a constant distribution has been chosen. Table 3-1 shows the input demands for a particular OD pair.

Table 3-1: Example

Number of Slice	Initial Time (minutes)	Number Trips
1	0	0.6
2	10	0.5
3	20	0.4
4	30	0.5

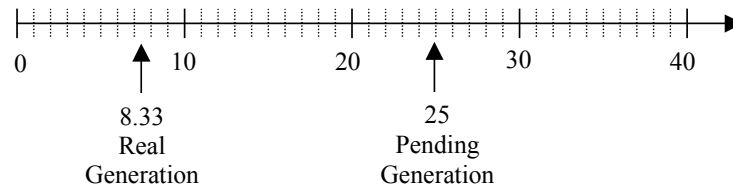
In the first slice,

Headway = $10/0.6 = 16.66$ minutes

It generates:

Generation Time 1 = $0 + 16.66/2 = 8.33$ minutes

GenerationTimePending = $8.33 + 16.66 = 25$ minutes

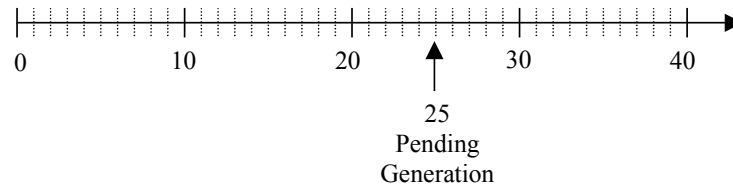


In the second slice,

$$\text{Headway} = 10/0.5 = 20 \text{ minutes}$$

It generates:

$$\text{GenerationTimePending} = \text{Minimum} (10 + 20, 25) = 25 \text{ minutes}$$



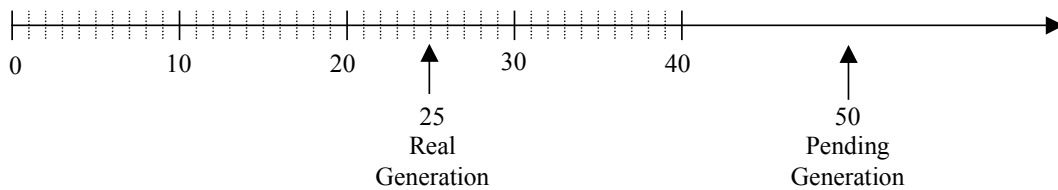
In the third slice,

$$\text{Headway} = 10/0.4 = 25 \text{ minutes}$$

It generates:

$$\text{Generation Time 1} = \text{Minimum} (20 + 25, 25) = 25 \text{ minutes}$$

$$\text{GenerationTimePending} = 25 + 25 = 50 \text{ minutes}$$

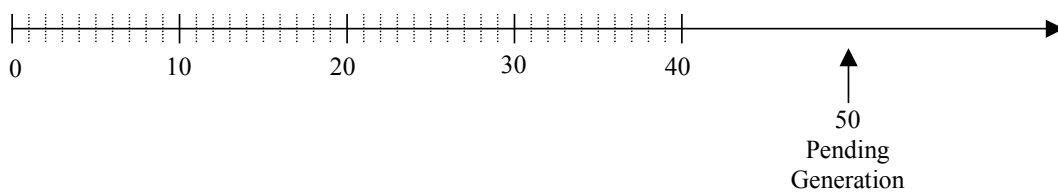


In the fourth slice,

$$\text{Headway} = 10/0.5 = 20 \text{ minutes}$$

It generates:

$$\text{GenerationTimePending} = \text{Minimum} (30 + 20, 50) = 50 \text{ minutes}$$



Considering all slices, the real generation of vehicles is carried out at 8.33 and 25. Therefore, 1 vehicle arrives during the first slice, and a second vehicle arrives during the third slice.

According to the input demand defined in Table 3-1, $0.6 + 0.5 + 0.4 + 0.5 = 2$ vehicles must be generated, which is in fact what we have achieved with the procedure described above.

3.3 VEHICLE ENTRANCE PROCESS

The above Headway Distributions determine the theoretical arrival time for each vehicle. It is then necessary to check whether the arrival is physically feasible or not. The Vehicle Entrance Process is defined by the following procedure:

```

If (IsThereSpace) then
    Enter a new vehicle
Else
    Add Vehicle in the Virtual queue
Endif

```

3.3.1 Function to determine whether the entrance is possible

The calculation for determining whether there is enough space to enter is made by considering the parameters of the entrance section, the parameters of the first vehicle (called leader) positioned in the entrance section, and the parameters of the vehicle type that try to enter, and the parameters of the entrance section.

The parameters of Entrance Section:

- the mean of Maximum Speed (MaxSectionSpeed).

The parameters of the leader are:

- the current position at entrance time (Pos).
- the current speed at entrance time (Speed).
- the maximum deceleration (MaxDecel).
- the minimum distance between vehicles (MinDist).
- the Length (L).
- the braking distance as the distance necessary to stop the vehicle applying the maximum deceleration. It is defined as:

$$BrakingD = \frac{Speed^2}{-2 * MaxDecel}$$

The parameters of the Vehicle Type:

- the mean of Maximum Speed (MaxSpeedAvg).
- the mean of Speed Acceptance (θ).
- the mean of Maximum Deceleration (MaxDecelAvg).
- The maximum desired speed on the entrance section :

$$MaxDesiredSpeed = Min(\theta * MaxSectionSpeed, MaxSpeedAvg)$$

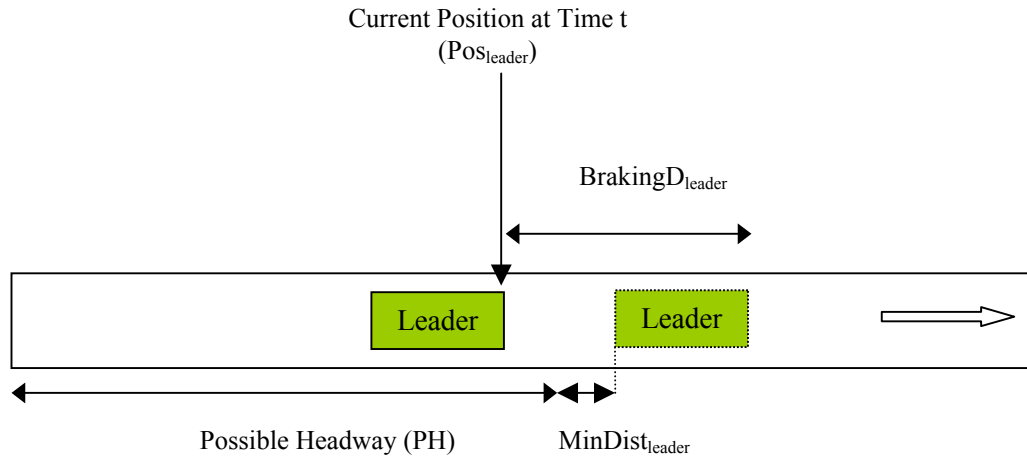
- the average braking distance of a vehicle type as the necessary distance to stop the vehicle applying the maximum deceleration. It is defined as:

$$AvgBrakingD = \frac{MaxDesiredSpeed^2}{-2 * MaxDecelAvg}$$

- the Minimum Distance to enter is defined taking the average braking distance (MinDistEnter)

$$MinDistEnter = AvgBrakingD$$

Considering all these parameters, the possible space to enter (PH: Possible Headway) is defined as:



$$PH = Pos_{leader} + BrakingD_{leader} - L_{leader} - MinDist_{leader}$$

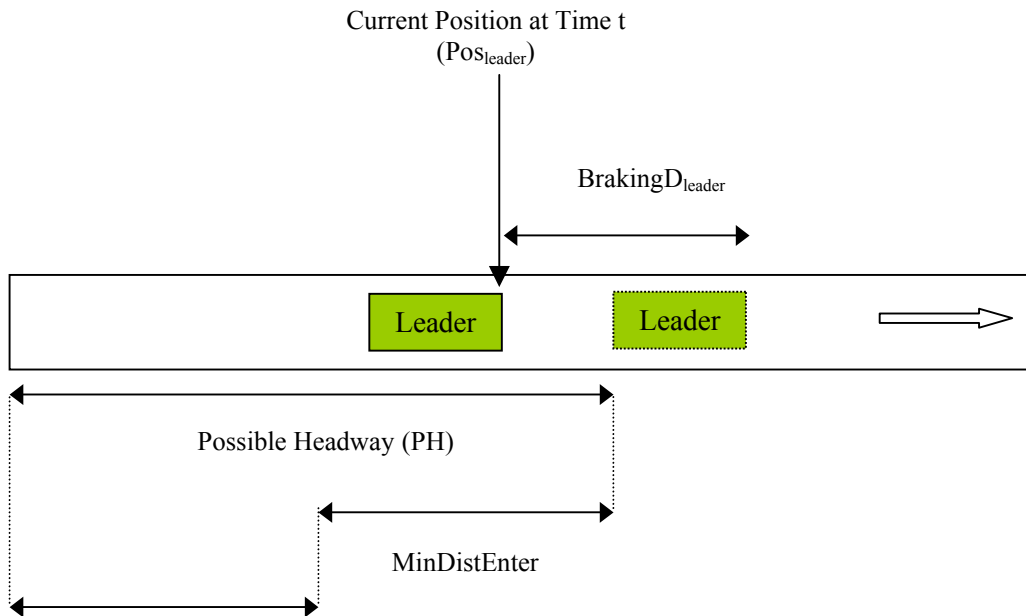
Taking all these values the function to evaluate the availability to enter is defined as:

```

if (PH >= MinDistEnter) then
    The vehicle has enough space
Else
    The vehicle does not have enough space
Endif

```

To enter the vehicle it is necessary to assign the entrance time into the system and its position.



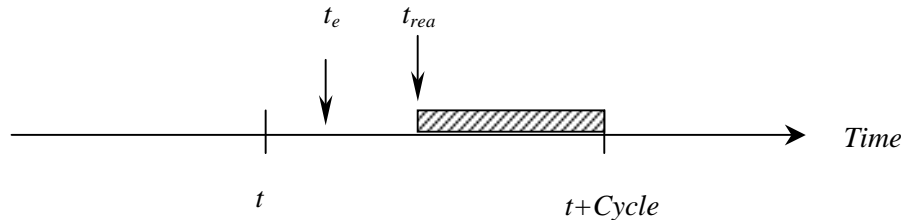
The necessary travel distance d (defined as $PH - MinDistEnter$) is:

$$t_d = \frac{PH - MinDistEnter}{MaxDesiredSpeed}$$

Therefore, the entrance time t_{real} is defined by:

$$t_{real} = MAXIMUM(t_e, (t + Cycle) - t_d)$$

where: t_e is the theoretical entrance time
 t is the simulation time
 Cycle is the simulation step



When the entrance time is known, then the entrance position is defined by:

$$entrancePosition = MaxDesiredSpeed * ((t + Cycle) - t_{real})$$

3.3.2 Virtual Entrance Queues

When a vehicle entrance is scheduled and there is not enough space to enter, then this vehicle is stocked in a virtual queue for each section. The definition of these entrance virtual queues depends of the traffic generation and the headway models.

- Traffic demand defined as input traffic flows and turning percentages
 - Traffic generation model: Exponential, Uniform, Normal and Constant.
 The virtual entrance queue is defined as a list of vehicle types. The entrance process consists in taking the first element of the list and generating the vehicle. If the size of the list is greater than X, then AIMSUN gives a warning message.
 An example of virtual queue for one entrance section could be:

car	car	bus	taxi	car	truck	car	...
-----	-----	-----	------	-----	-------	-----	-----

- Traffic generation model: ASAP.
 The virtual entrance queue is defined as a list of elements, where each element represents the name of a vehicle type and the number of its vehicles pending to enter. The entrance process consists of taking a vehicle type randomly.
 An example of a virtual queue for one entrance section could be:

car	10
bus	2
taxi	4
...	...
truck	2

- Traffic demand defined as an OD Matrix
 - Traffic generation model: Exponential, Uniform, Normal and Constant.
 The virtual entrance queue is defined as a list of elements, where each element represents the vehicle type identifier, its origin and its destination. The entrance process consists of taking the first element of the list and generating the vehicle. If the size of the list is greater than X, then AIMSUN gives a warning message.
 An example of a virtual queue for one entrance section could be:

car	car	bus	taxi	car	truck	car	...
from 1	from 1	from 4	from 1	from 1	from 1	from 1	...
to 2	to 3	to 2	to 2	to 2	to 2	to 3	...

- traffic generation model: ASAP.
The virtual entrance queues are not defined. Instead of using them for each entrance section, AIMSUN uses the same mechanism implemented in previous versions.

3.4 VEHICLE MODELLING PARAMETERS

In a microscopic simulation approach, vehicle manoeuvres are modelled in detail using car following, lane changing and gap acceptance models. These vehicle behaviour models are a function of several parameters that allow modelling of different types of vehicles: cars, buses, trucks, etc., and a variation of individual vehicles in each type.

The user can set these parameters, depending on the characteristics of the traffic to be reproduced. For example, the behaviour of drivers in an urban network may be different from their behaviour on a motorway. The basic model is the same, but variables such as lane changing zones or reaction times could be different. These parameters can also vary from one country to another or from one environment (rural) to another (urban). Setting appropriate values for these parameters is part of the model calibration process.

These parameters can be grouped into three categories according to the level at which they are defined: vehicle attributes, local section parameters and global network parameters.

3.4.1 Vehicle Attributes

These parameters are defined at the level of vehicle type (e.g. car, bus, truck, etc.). It is possible to define not only the mean values for the attributes of each vehicle type, but also the deviation, minimum and maximum values. The particular characteristics for each vehicle are sampled from a truncated Normal distribution. The attributes that characterise a vehicle type are the following:

Name

Character string.

Identifies the Type of vehicle.

Length

Mean, deviation, maximum and minimum values.

The length, in meters, for this type of vehicle. This parameter is used both for graphical and modelling purposes. It has a direct influence on traffic modelling, as vehicle length is taken into account in all vehicle behaviour models.

Width

Mean, deviation, maximum and minimum values.

This is the width, in meters, for this type of vehicle. This parameter is only used for graphical purposes and does not have a direct influence on the traffic modelling.

Maximum desired speed

Mean, deviation, maximum and minimum values.

This is the maximum speed, in km/h, that this type of vehicle can travel at any point in the network. See the next section for details on how maximum desired speed for a vehicle is calculated for different parts of the network.

For example, one could define a 'sports car' vehicle type with a mean desired speed of 110 km/h, a deviation of 10 km/h, a maximum value of 140 km/h and a minimum value of 90 km/h. Sports cars would be sampled from a Normal (110,10) distribution.

Maximum acceleration

Mean, deviation, maximum and minimum values.

This is the maximum acceleration, in m/s^2 , that the vehicle can achieve under any circumstances. This acceleration is as used in the Gipps car-following model.

Normal deceleration

Mean, deviation, maximum and minimum values.

This is the maximum deceleration, in m/s^2 , that the vehicle can use under normal conditions. This deceleration is as used in the Gipps car-following model.

Maximum deceleration

Mean, deviation, maximum and minimum values.

This is the most severe braking, in m/s^2 , a vehicle can apply under special circumstances, such as emergency braking.

Speed acceptance

Mean, deviation, maximum and minimum values.

This parameter ($\theta \geq 0$) can be interpreted as the ‘level of goodness’ of the drivers or the degree of acceptance of speed limits. $\theta \geq 1$ means that the vehicle will take as maximum speed for a section a value greater than the speed limit, while $\theta \leq 1$ means that the vehicle will use a lower speed limit. See the next section for details on how maximum desired speed for a vehicle is calculated for different parts of the network.

Minimum distance between vehicles

Mean, deviation, maximum and minimum values.

This is the distance, in meters, that a vehicle keeps between itself and the preceding vehicle when stopped.

Maximum give-way time

Mean, deviation, maximum and minimum values.

When a vehicle is in a give-way situation, for example at a Yield or Stop sign in a junction or an on-ramp in a freeway, it applies either the normal gap-acceptance model or a lane-changing model in order to cross or merge with traffic, respectively. When a vehicle has been at a standstill for more than this Give-way Time (in seconds), it will become more aggressive and will reduce the acceptance margins. This period is also used in the Lane-Changing model as the time that a vehicle accepts being at a standstill while waiting for a gap to be created in the desired turning lane before giving up and continuing ahead.

Guided vehicles

This is the percentage of vehicles that are guided throughout the network. This is only taken into account when simulating with the Route-Based model, i.e. traffic demand defined as an OD matrix and vehicles following paths to destination.

Equipped vehicles

This is the percentage of vehicles that are equipped. Equipped vehicles can be detected by detectors with measuring capability ‘Equipped’, which provide the detection time, the detector identifier, the vehicle identifier, the vehicle type name and the public transport line identifier if it is a public transport vehicle.

Guidance acceptance

Mean, deviation, maximum and minimum values.

This parameter ($0 \leq \lambda \leq 1$) gives the level of compliance of this vehicle type with the guidance indications, such as information given through Variable Messages Signs or particular Vehicle Guidance Systems. This parameter represents the probability that a vehicle will follow a recommendation. The modelling of drivers’ reactions to guidance is explained in greater detail later on.

Cruising tolerance

This parameter is used both in the Fuel Consumption and in the Pollution Emission models (see Chapter 10). Vehicles whose acceleration or deceleration (in m/s^2) is less than this coefficient are considered to be cruising vehicles (vehicles driving at a relatively constant speed) as opposed to vehicles accelerating or decelerating.

Fuel consumption parameters

For each vehicle type, the following six parameters, which specify the vehicle’s fuel consumption rates, must be specified:

- Fuel consumption rate for idling vehicles in ml/s
- Fuel consumption rate for accelerating vehicles, in ml/s
- Fuel consumption rate, in litres per 100 km, for vehicles travelling at a constant speed of 90 km/h
- Fuel consumption rate, in litres per 100 km, for vehicles travelling at a constant speed of 120 km/h

- Speed at which the fuel consumption rate, in ml/s, is at a minimum for a vehicle cruising at constant speed
- Fuel consumption rate for decelerating vehicles in ml/s.

The Fuel Consumption Model is explained in greater detail in section 10.1.

Pollution emission parameters

For each vehicle type, a set of pollutants can be defined (i.e. CO, NO_x, HC). For each pollutant, the following parameters are required:

- Emission rate for accelerating vehicles in g/s
- Emission rate for decelerating vehicles in g/s
- Emission rate for idling vehicles in g/s
- A look-up table for vehicles cruising at a constant speed consisting of a set of pairs (speed break point (km/h), Emission rate (g/s), for a maximum of 15 break points.

The Pollution Emission Model is explained in detail in section 10.2

3.4.2 Local Parameters

Certain parameters may affect vehicle behaviour, although they are not defined at the level of vehicle type, but are related to sections. This means that these parameters are applied locally to the vehicles while they are driving along the section, but they change as the vehicle enters in a new section. These parameters are the following:

Section Speed Limit

Maximum allowed speed (in km/h) for the vehicles travelling through a section. Depending on the characteristics of the drivers, they may or may not follow speed limit recommendations (see “Calculating the speed for a vehicle at a section”).

Lane Speed Limit

Maximum allowed speed (in km/h) for the vehicles travelling through a particular lane. By default, the Section Speed Limit is applied to all the lanes of the section. The user can optionally define different Speed limits for each lane of the sections.

Turning Speed

Maximum speed (in km/h) at which a vehicle will travel when making the turn. Depending on the characteristics of the drivers, they will use a higher or lower speed (see “Calculating the speed for a vehicle at a section”). A vehicle driving through a section will start to decelerate while approaching the turning in order to reach its turning speed at the end of the section. The turning speed is maintained during the turn and, when entering the next section, the vehicle will start to accelerate again according to its desired speed for this section.

Visibility Distance at Junctions

When a vehicle is approaching a junction through a section where there is a Yield or Give-way sign at the end, the gap-acceptance model is applied. It will be applied whenever the distance from the vehicle to the end of the section is less than this visibility distance (in meters).

Yellow Box Speed

A vehicle approaching a Yellow Box Junction will avoid entering the junction area whenever the preceding vehicle is moving at a speed below this parameter (in km/h). The user can deactivate the Yellow Box model of a particular section by setting the Yellow Box Speed of that section to 0.

Distance Zone 1

Distance from the end of Zone 1 to the next turning point (i.e. the end of the current section). This distance is defined as the time (in seconds) needed to travel a certain distance. This distance in time is converted into length using a function of the section speed limit and the desired speed of each vehicle at the section (see

section 3.5.5). The reason for using time is that it makes the variable distance dependent on the vehicle characteristics. This parameter is used in the Lane-Changing model. In fact, Distance Zone 1 minus Distance Zone 2 is equal to the length of zone 3 (see Figure 3-5).

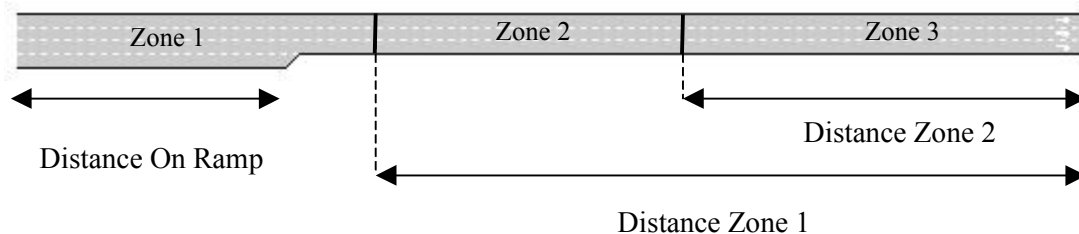
Distance Zone 2

Distance (in seconds) from the end of Zone 2 to the next turning point, which is equal to the end of the current section. This distance in time is converted into length using a function of the section speed limit and the desired speed of each vehicle at the section (see section 3.5.5). This parameter is used in the Lane-Changing model. In fact, Distance Zone 2 is equal to the length of Zone 3 (see Figure 3-5).

Time Distance On-Ramp

This parameter is the distance, in seconds, from those side lanes that are considered to be on-ramp lanes (see Figure 3-5). This is used for the special lane-changing model applied at the on-ramps, as described in section 3.5.7. This means that different criterion for determining whether to consider side lanes as on-ramps or not may be applied to different parts of the network. The user may have local control of the on-ramp lane-changing model, making the calibration process easier. This distance in time is converted into length using a function of the section speed limit and the desired speed of each vehicle at the section (see Figure 3-5).

Figure 3-5: Distance Parameters in a Section



Section Slope

The influence of the section slope on vehicle movement is modelled by means of an increase or reduction of the acceleration and braking capabilities (see section 3.5.4). Slope units are represented as percentages, reflecting the height corresponding to 100 meters length.

Figure 3-6 shows the Basic folder of the Section dialog window, in which these local section parameters are displayed. These data cannot be modified here, this is carried out in the Tedi editor, using the Section dialog window. Therefore they are saved together with the network description. Running experiments with different values for these parameters requires unloading of the network, modification via the Tedi, and subsequent reloading of the modified network into AIMSUN.

Maximum Give Way Time Variability

Local variability of the Maximum Give Way Time attribute of the vehicle. It is an absolute value that increases or decreases the vehicle's value. This parameter is useful to locally calibrate give way situations.

Figure 3-6: Section Dialog Window. Basic Folder

Dest	Turn Speed	Warning	Initial Cost Function	Cost Function
4	49.26		not enabled	not enabled
7	50.00		not enabled	not enabled
2	40.29	Yield	not enabled	not enabled

3.4.3 Global modelling parameters

This is a set of parameters related to vehicle behaviour models that is valid throughout the whole network and which is defined neither at the level of vehicle type nor at the section level. They are used for all vehicles driving anywhere in the network during the entire simulation experiment.

3.4.3.1 General Parameters

Driver's reaction time (also simulation step)

This is the time it takes a driver to react to speed changes in the preceding vehicle. It is used in the car-following model and for implementation reasons it is also taken as the Simulation Time Step or cycle.

Reaction time at stop

This is the time it takes for a stopped vehicle to react to the acceleration of the vehicle in front, or to a traffic light changing to green. Reaction time at stop is used as reaction time only for vehicles that start from a stopped condition, while the normal reaction time is used for vehicles that are moving. Reaction time at stop has a strong influence in the queue discharge behaviour, and it therefore gives the user much more control of queue modelling.

Queuing Up Speed

Vehicles whose speed decreases below this threshold value (in m/s) are considered to be stopped and consequently, to join a queue. This parameter affects statistical data gathering for stops and queues.

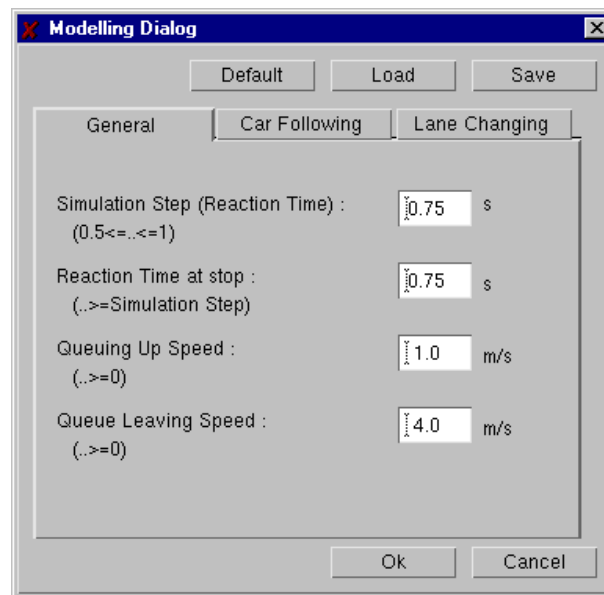
Queue Leaving Speed

Vehicles that are stopped in a queue whose speed increases above this threshold value (in m/s) are considered to leave the queue and no longer to be at a standstill. This parameter affects statistical data gathering for stops and queues.

Queuing Up and Queue Leaving Speed parameters also have an influence on the lane-changing model. A vehicle driving in zone 3 of a section is not willing to wait for a gap for longer than the Maximum Give Way Time at a standstill, and the condition of standstill is determined by these parameters.

The Global Parameters of Reaction Time, Reaction Time at stop, Queuing Up Speed and Queue Leaving Speed can be defined via the General tab folder in the Modelling dialog window (see Figure 3-7). You can view this window by selecting the '*Experiment / Modelling*' command.

Figure 3-7: General Modelling Parameters Dialog Window



3.4.3.2 Car-Following Model

This option is only available in version v4.1.3 or later.

The user may select between using the Car-Following Model as it has been defined in version previous to v4.1.3 (option 4.1) or use the new model which includes a modification in the way the leader deceleration is estimated (option 4.2). See section 3.5.1 for details.

The Global Parameter Car-Following Model can be defined via the Car Following tab folder of the Modelling dialog window (see Figure 3-8). You can view this window by selecting the '*Experiment / Modelling*' command.

3.4.3.3 2-lanes Car-Following Parameters

Set of global parameters used in the 2-lanes car following model.

Number of vehicles

Maximum number of vehicles to be considered in the 2-lane variation of the Car-Following Model, which is used for modelling the influence of adjacent lanes in the Car-Following Model, as described in section 3.5.3.

Maximum Distance

Maximum distance ahead (in meters) to be considered in the 2-lane Car-Following Model, as described in section 3.5.3.

Maximum Speed Difference

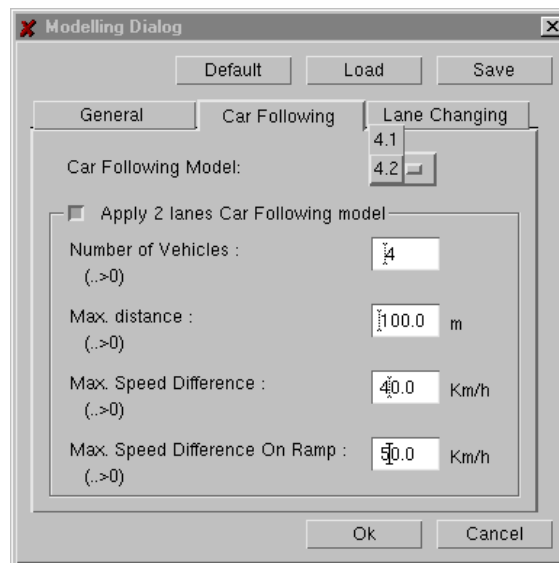
Maximum speed difference (in km/h) between one lane and the adjacent lane in the 2-lane Car-Following Model, as described in section 3.5.3.

Maximum Speed Difference On-Ramp

Maximum speed difference (in km/h) between the main lane and an on-ramp lane in the 2-lane Car-Following Model, as described in section 3.5.3.

The Global Parameters Number of vehicles, Maximum distance, Maximum speed difference and Maximum Speed Difference On-Ramp can be defined via the Car Following tab folder of the Modelling dialog window (see Figure 3-8). You can view this window by selecting the *'Experiment / Modelling' command*.

Figure 3-8: Car Following Modelling Parameters Dialog Window



3.4.3.4 Lane-changing Parameters

Set of global parameters used in the lane-changing model.

Percent Overtake

This represents the percentage of the speed from which a vehicle decides to overtake. The value has to be greater than zero and less than or equal to one. It is used for modelling the overtaking decision as explained in section 3.5.6.

Percent Recover

This is the percentage of the speed from which a vehicle decides to return to the slower lane after overtaking. The value has to be greater than zero and less than or equal to one. It is used for modelling the overtaking decision as explained in section 3.5.6.

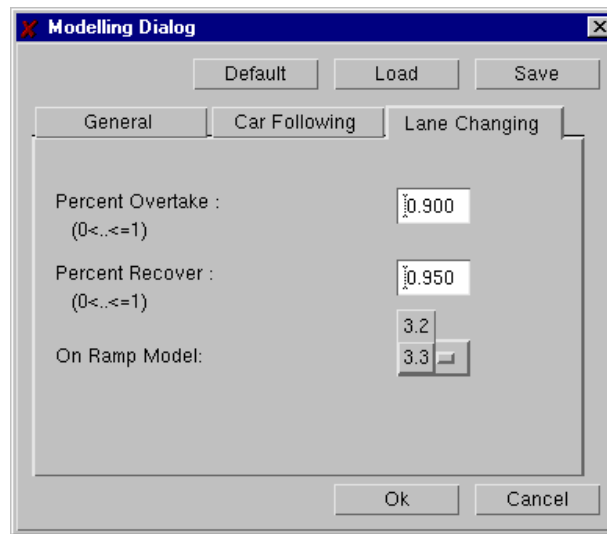
On-Ramp Model

The user may select whether to use the v3.2 On-Ramp model or the v3.3, whose differences are explained in section 3.5.7. The purpose is to keep consistency with previous model that the user may have calibrated with previous versions.

The Global Parameters of Percent Overtake, Percent Recover and On-Ramp Model can be defined via the Lane Changing tab folder of the Modelling dialog window (see Figure 3-9). You can view this window by selecting the *'Experiment / Modelling' command*.

3.4.4 Other Global Parameters

Apart from the parameters described above, two further Global Parameters can be defined in TEDI for the whole network: 'Road side of vehicle movement' and 'Distinguish destination lanes in turnings'. The value of these parameters can affect AIMSUN modelling as follows.

Figure 3-9: Lane-changing Modelling Parameters Dialog Window

Road side of vehicle movement

When editing a network with TEDI, the user can define whether the vehicles drive on the left or on the right side of the road, as a global parameter for the whole network. This parameter is taken into account in the AIMSUN lane-changing model in the following way:

- When a vehicle attempts to overtake another vehicle it will try to do so using the adjacent left lane when driving on the right side, or the adjacent right lane when driving on the left.
- When a vehicle is driving fast enough and wants to move back to the slower lane, it will try to go to the rightmost lane when driving on the right and to the leftmost lane when driving on the left.
- When a vehicle enters the network via a section, it will prefer to use the rightmost lane when driving on the right and the leftmost lane when driving on the left.

As a result of this modelling, in a left-hand driving network, vehicles tend to drive using the leftmost lane of the sections and in networks driving on the right, vehicles tend to prefer the rightmost lanes. The effect of this parameter on simulation results varies, depending on the geometry and type of the network (urban or interurban), but it always affects the behaviour of the vehicles.

Distinguish destination lanes in turnings

This parameter states whether or not the user is able to specify the possible destination lanes when defining a turning movement.

When this parameter is set to NO, it is assumed that all lanes of the destination section in a turning are possible. This means that when defining a turning, the user may choose the origin section, the lanes from this section where the turning is allowed and the destination section, but not the lanes of the destination section. With TEDI, the user clicks on the origin lanes and in the destination section.

If this parameter is set to YES, when defining a turning the user can choose, not only the lanes from the origin section where the turning will be allowed, but also the lanes in the destination section. With TEDI, the user clicks on the origin lanes and on the destination lanes.

AIMSUN will then take into account the lanes composing a turning. When a vehicle makes a turning it will be able to use only the specified lanes, both for the origin and destination sections.

3.5 MODELLING VEHICLE MOVEMENT

During their journey along the network, vehicles are updated according to vehicle behaviour models: “Car-Following” and “Lane-Changing”. Drivers tend to travel at their desired speed in each section but the environment (i.e. preceding vehicle, adjacent vehicles, traffic signals, signs, blockages, etc) conditions their behaviour.

Simulation time is split into small time intervals called simulation cycles or simulation steps (Δt). For reasons of efficiency, these cycles are set equal to the driver’s reaction time (parameter of the car-following model). The user can set this value within the range $(0.5 \leq \Delta t \leq 1.25)$, although a value between 0.65 and 0.80 seconds is advisable. In any case, the reaction time (step) has to be considered as a parameter to calibrate. The objective is to achieve simulated flows similar to the real ones, therefore it is a modeller’s task to tune up the simulation step in order to get the best results. To start with, a value of $\Delta t = 0.75$ seconds is recommended, given that good results have been obtained with it.

A simulation step may affect not only the computing performance but also some simulation outputs, such as the section capacities, for example. The smaller the simulation step is, the higher capacity values are obtained. The reason for this is because the drivers are “more skilful”, as they have shorter reaction times. They can drive closer to the preceding vehicles, they can find gaps more easily, they can accelerate earlier, they have more opportunities to enter the network, etc.

Each simulation cycle, the position and speed of every vehicle in the system is updated according to the following algorithm:

```

if (it is necessary to change lanes) then
    Apply Lane-Changing Model
endif

if (the vehicle has not changed lanes) then
    Apply Car-Following Model
endif

```

Once all vehicles have been updated for the current cycle, vehicles scheduled to arrive during this cycle are introduced into the system and the next vehicle arrival times are generated.

3.5.1 Car-Following Model

The car-following model implemented in AIMSUN is based on the Gipps model ([GIP81] and [GIP86b]). It can actually be considered as an *ad hoc* development of this empirical model, in which the model parameters are not global but determined by the influence of local parameters depending on the “type of driver” (limit speed acceptance of the vehicle), the geometry of the section (speed limit on the section, speed limits on turnings, etc.), the influence of vehicles on adjacent lanes, etc.

It basically consists of two components, acceleration and deceleration. The first represents the intention of a vehicle to achieve a certain desired speed, while the second reproduces the limitations imposed by the preceding vehicle when trying to drive at the desired speed.

This model states that the maximum speed to which a vehicle (n) can accelerate during a time period ($t, t+T$) is given by:

$$V_a(n, t + T) = V(n, t) + 2.5a(n)T \left(1 - \frac{V(n, t)}{V^*(n)} \right) \sqrt{0.025 + \frac{V(n, t)}{V^*(n)}}$$

where:

- $V(n, t)$ is the speed of vehicle n at time t ;
- $V^*(n)$ is the desired speed of the vehicle (n) for current section;
- $a(n)$ is the maximum acceleration for vehicle n ;
- T is the reaction time = updating interval = simulation step.

On the other hand, the maximum speed that the same vehicle (n) can reach during the same time interval (t, t+T), according to its own characteristics and the limitations imposed by the presence of the leader vehicle is:

$$V_b(n, t + T) = d(n)T + \sqrt{d(n)^2 T^2 - d(n) \left[2\{x(n-1, t) - s(n-1) - x(n, t)\} - V(n, t)T - \frac{V(n-1, t)^2}{d'(n-1)} \right]}$$

where:

- d(n) (< 0) is the maximum deceleration desired by vehicle n;
- x(n, t) is position of vehicle n at time t;
- x(n-1, t) is position of preceding vehicle (n-1) at time t;
- s(n-1) is the effective length of vehicle (n-1);
- d'(n-1) is an estimation of vehicle (n-1) desired deceleration.

In any case, the definitive speed for vehicle n during time interval (t, t+T) is the minimum of those previously defined speeds:

$$V(n, t + T) = \min \{ V_a(n, t + T), V_b(n, t + T) \}$$

Then, the position of vehicle n inside the current lane is updated taking this speed into the movement equation:

$$x(n, t + T) = x(n, t) + V(n, t + T)T$$

3.5.2 Estimation of leader's deceleration

In versions previous to v4.1.3, the estimation of leader's deceleration was taken as the proper desired deceleration of the leader vehicle. In case that the ratio between the follower's and leader's deceleration capabilities is relatively high, the above deceleration component of the car-following model presents instabilities which may cause some vehicles to drive too close to the leader.

$$\text{Model v4.1: } d'(n-1) = d(n-1)$$

Since version v4.1.3 it is the user choice to select a new mode of estimating the leader's deceleration, which avoids up to certain extent this malfunction (see section 3.4.3.2). In this case, the estimation of leader's deceleration is taken as the average between the leader's deceleration and the follower's deceleration.

$$\text{Model v4.2: } d'(n-1) = \frac{1}{2}\{d(n) + d(n-1)\}$$

Model v4.2 reduces the difference between both parameters thus decreasing the probability of such instabilities. Although on average both models provide similar vehicle headways, model v4.2 avoids certain vehicles driving too close to the leader at high speeds. In case that there are no big differences among the maximum and minimum deceleration capabilities of different vehicles in the network, model v4.1 can be used. In case that the variation between decelerations is relatively high, model v4.2 is recommended.

3.5.2 Calculating the speed for a vehicle at a section

The car-following model is such that a leading vehicle, i.e. a vehicle driving freely, without any vehicle affecting its behaviour, would try to drive at its maximum desired speed. Three parameters are used to calculate the maximum desired speed of a vehicle while driving on a particular section or turning, two are related to the vehicle and one to the section or turning:

1. Maximum desired speed of the vehicle *i*: $v_{\max}(i)$
2. Speed acceptance of the vehicle *i*: $\theta(i)$

3. Speed limit of the section or turning s : $S_{limit}(s)$

The speed limit for a vehicle i on a section or turning s , $s_{limit}(i, s)$, is calculated as:

$$s_{limit}(i, s) = S_{limit}(s) \cdot \theta(i)$$

Then, the maximum desired speed of vehicle i on a section or turning s , $v_{max}(i, s)$ is calculated as:

$$v_{max}(i, s) = MIN[s_{limit}(i, s), v_{max}(i)]$$

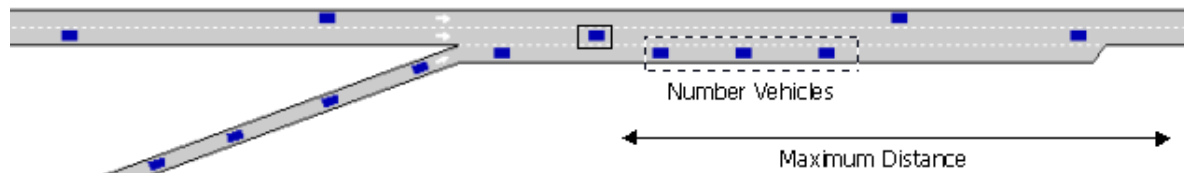
This maximum desired speed $v_{max}(i, s)$ is the same as that referred to above, in the Gipps car-following model, as $V^*(n)$.

3.5.3 The 2-lanes car-following model

The purpose is to model the influence of adjacent lanes in the car-following model. When a vehicle is driving along a section, we consider the influence that a certain number of vehicles (*Number Vehicles*) driving slower in the adjacent right-side lane—or left-side lane, when driving on the left—, may have on the vehicle. Basically, the model determines a new maximum desired speed of a vehicle in the section, which will then be used in the car-following model.

The model first calculates the mean speed for *Number Vehicles* driving downstream of the vehicle in the adjacent slower lane (*MeanSpeedVehiclesDown*). Only vehicles within a certain distance (*Maximum Distance*) from the current vehicle are taken into account (see Figure 3-10). If less than *Number Vehicles* are involved, the desired speed of the current vehicle at the section is used to complement up to the *Number Vehicles* values in order to get a more meaningful average.

Figure 3-10: 2-Lanes car following



We distinguish between two cases: 1) the adjacent lane is an on-ramp, and 2) the adjacent lane is any other type of lane. Apart from *Number Vehicles* and *Maximum Distance* parameters, the user can define two additional parameters, *Maximum Speed Difference* and *Maximum Speed Difference On-Ramp*. Then, the final desired speed of a vehicle at a section is calculated as follows:

```

if (the adjacent slower lane is an On-ramp) {
    MaximumSpeed = MeanSpeedVehiclesDown + MaximumSpeedDifferenceOnRamp
}
else {
    MaximumSpeed = MeanSpeedVehiclesDown + MaximumSpeedDifference
}
DesiredSpeed = Minimum (  $v_{max}(i, s)$ ,  $\theta(i) * MaximumSpeed$  )

```

This procedure ensures that the differences of speeds between two adjacent lanes will approximately be always lower than *Maximum Speed Difference* or *Maximum Speed Difference On-Ramp*, respectively.

Example

The following example is intended to clarify the influence of the *Number of Vehicles* parameter. Let us assume that we set 4 cars and 100 Mts. respectively for *Number of Vehicles* and *Maximum Distance*. It means that a vehicle looks at most to the first 4 cars in front of him in the adjacent lane that are located in the next 100 Mts. from its position. If there is only 1 car in this 100 Mts., only the speed of this car is

considered, and to get the 4 vehicle's average speed, 3 additional 'dummy' cars with free flow speed are taken.

For instance, assume that in the next 100 Mts. there is an on ramp with 1 vehicle stopped and speed limit of the section is 60 km/h. The average speed considered in the 2 lanes car following would be:

$$(0 + 60 + 60 + 60)/4 = 45 \text{ km/h.}$$

If the maximum speed difference at on-ramps is 50 km/h, it would mean that vehicles in the main lane would accept driving to $50 + 45 = 95 \text{ km/h}$.

On the other hand, if there are 4 cars or more stopped in the ramp, the average speed would be 0 km/h and it would mean that vehicles in the main lane would accept driving to $50 + 0 = 50 \text{ km/h}$. If the 4 cars are stopped farther than 100 Mts., they would not affect the vehicle yet.

There is a lack in the 2-lanes car following model: it does not go beyond the limits of the current section. It means that a vehicle approaching the on-ramp will only adapt its speed to the stopped vehicles once it is driving side by side to the on-ramp lane. This might be the reason why some vehicles may drive fast near a congested on-ramp lane. However, this behaviour may happen when crossing from one section to the next one, but once they enter into the next section vehicles realise that there is a congested on-ramp and should decelerate.

3.5.4 Modelling the influence of the Section Slope

The influence of the section slope on vehicle movement is modelled by means of an increase or reduction of acceleration and braking capability. The following algorithm is applied to calculate the maximum acceleration for a vehicle that will be used in the car-following model. It is a function of the slope and the maximum desired acceleration for the vehicle:

```

if (slope != 0){
    accel = Maximum(vehicle_acc - slope*9.81/100.0, vehicle_acc*0.1)
}else{
    accel = vehicle_acc
}

```

Slope units represent a percentage, which is the height corresponding to 100 meters length. In order to avoid zero or negative acceleration values, a minimum value of 10% of the maximum desired acceleration for the vehicle is used.

3.5.5 Lane-Changing Model

The lane-changing model can also be considered as a development of the Gipps lane-changing model ([GIP86a] and [GIP86b]). Lane change is modelled as a decision process, analysing the necessity of the lane change (such as for turning manoeuvres determined by the route), the desirability of the lane change (to reach the desired speed when the leader vehicle is slower, for example), and the feasibility conditions for the lane change that are also local, depending on the location of the vehicle in the road network.

The lane-changing model is a decision model that approximates the driver's behaviour in the following manner:

- Each time a vehicle has to be updated we ask the following question: *Is it necessary to change lanes?* The answer to this question depends on several factors: the turning feasibility in the current lane, the distance to the next turning and the traffic conditions in the current lane. The traffic conditions are measured in terms of speed and queue lengths. When a driver is going slower than he wishes, he tries to overtake the preceding vehicle. On the other hand, when he is travelling fast enough, he tends to go back into the slower lane.
- If we answer the previous question in the affirmative, to successfully change lanes, we must first answer two further questions:

• *Is it desirable to change lanes?*

Check whether there will be any improvement in the traffic conditions for the driver as a result of lane changing. This improvement is measured in terms of speed and distance. If the speed in the future lane is fast enough compared to the current lane, or if the queue is short enough, then it is desirable to change lanes.

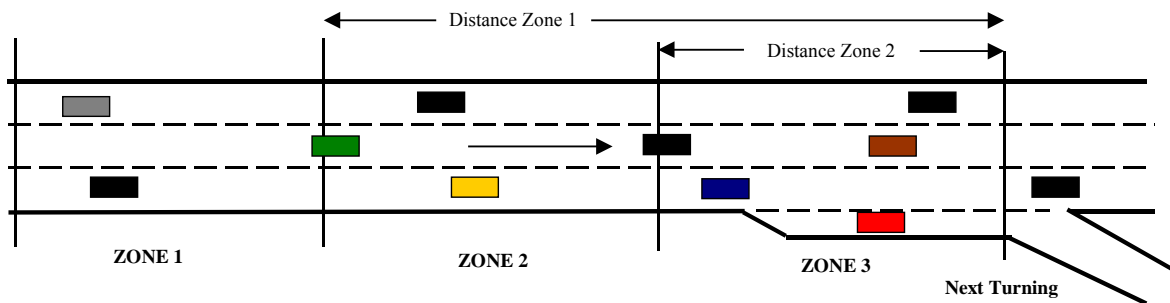
• *Is it possible to change lanes?*

Verify that there is enough of a gap to make the lane change with complete safety. For this purpose, we calculate both the braking imposed by the future downstream vehicle to the changing vehicle and the braking imposed by the changing vehicle to the future upstream vehicle. If both braking ratios are acceptable then the lane changing is possible.

In order to achieve a more accurate representation of the driver's behaviour in the lane-changing decision process, three different zones inside a section are considered, each one corresponding to a different lane changing motivation. These zones are characterised by the distance up to the end of the section, i.e., the next point of turning (see Figure 3-11).

- **Zone 1:** This is the farthest distance from the next turning point. The lane-changing decisions are mainly governed by the traffic conditions of the lanes involved. The feasibility of the next desired turning movement is not yet taken into account. To measure the improvement that the driver will get from changing lanes, we consider several parameters: desired speed of driver, speed and distance of current preceding vehicle, speed and distance of future preceding vehicle.
- **Zone 2:** This is the intermediate zone. It is mainly the desired turning lane that affects the lane-changing decision. Vehicles not driving in valid lanes (i.e. lanes where the desired turning movement can be made) tend to get closer to the correct side of the road from which the turn is allowed. Vehicles look for a gap may try to adapt to it, but do not affect the behaviour of vehicles in the adjacent lanes.
- **Zone 3:** This is the shortest distance to the next turning point. Vehicles are forced to reach their desired turning lanes, reducing speed if necessary, and even coming to a complete stop in order to make the change possible. Also, vehicles in the adjacent lane can modify their behaviour in order to provide a gap big enough for the vehicle to succeed in changing lanes.

Figure 3-11: Lane Changing Zones



Lane changing zones are defined by two parameters, Distance to Zone 1 and Distance to Zone 2. These parameters are defined in time (seconds) and they are converted into distance whenever it is required for each vehicle i at each section s using the following function:

$$D_m = D_t \cdot S_{limit}(s) \cdot \left[\frac{S_{limit}(s)}{v_{max}(i, s)} \right]$$

where:

D_m : Distance in meters

D_t : Distance in seconds

$S_{limit}(s)$: Speed limit of the section s

$v_{max}(i, s)$: Maximum desired speed of vehicle i on a section or turning s

This function ensures that faster vehicles (with respect to the speed limit) have shorter zones than slower vehicles. This procedure also ensures that not all vehicles consider exactly the same zone lengths, thus giving more variability to the vehicle behaviour, which provides a more realistic model.

The scope of lane changing zones does not extend beyond the section boundaries when it is connected to the previous sections by junctions or joins. However, in the case of sections grouped as polysections, the zones are considered for the whole set of sections.

The zones in a polysection are defined from the end of the polysection, i.e. from the last section of the polysection. However, the particular zone distance parameter that a vehicle uses at a particular location in the polysection is the one defined in the current section. The most desirable practice would be to define the same zone distances at all sections of the polysection (this would be the most consistent). However, the user may decide to define different values for different sections of the same polysection. This would lead to a situation in which the length of zones varies as the vehicle is moving from section to section in the polysection. Keep in mind, however, that all zone distance parameters of the sections of a polysection are referred to the end of the polysection. Let us use two examples:

Example 1

A polysection composed of 3 sections, ordered 1, 2 and 3. (Figure 3-12)

Section 1: Length 100mts. Distance Zone 2 = 10 sec. Speed 72 km/h (20 m/s)

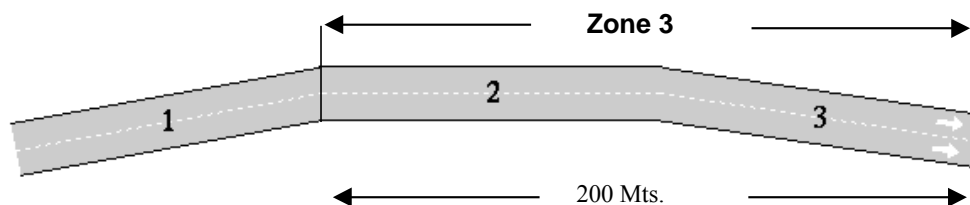
Section 2: Length 100mts. Distance Zone 2 = 10 sec. Speed 72 km/h. (20 m/s)

Section 3: Length 100mts. Distance Zone 2 = 10 sec. Speed 72 km/h. (20 m/s)

Length of polysection = 300 Mts.

Length of zone 3 = 10 sec * 20 m/s = 200 Mts. (from the end of polysection = end of section 3). All vehicles driving in any section will consider 200 Mts. the length of zone 3, which means as soon as they enter section 2. Therefore section 1 is zone 2 and sections 2 and 3 are zone 3.

Figure 3-12: Example 1: Zones in a polysection



Example 2

The same polysection composed of 3 sections ordered 1, 2 and 3. (Figure 3-13)

Section 1: Length 100mts. Distance Zone 2 = 15 sec. Speed 72 km/h (20 m/s)

Section 2: Length 100mts. Distance Zone 2 = 5 sec. Speed 72 km/h. (20 m/s)

Section 3: Length 100mts. Distance Zone 2 = 10 sec. Speed 72 km/h. (20 m/s)

Length of polysection = 300 Mts.

Length of zone 3 depends on whether a vehicle is driving in section 1, 2 or 3.

Vehicle driving in section 1

Length of zone 3 = 15 sec * 20 m/s = 300 Mts.

(from the end of polysection = start of section 1). The vehicle is now in zone 3.

Vehicle driving in section 2

Length of zone 3 = 5 sec * 20 m/s = 100 Mts.

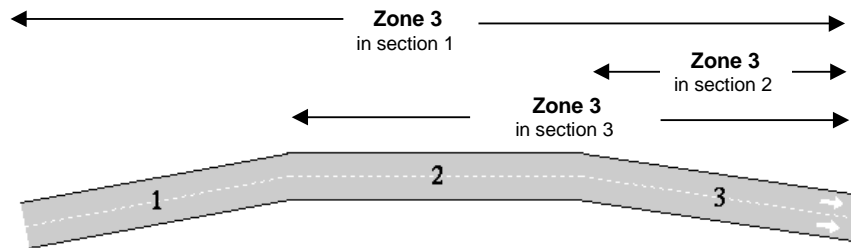
(from the end of polysection = end of section 2). The vehicle is now back to zone 2 !!

Vehicle driving in section 3

Length of zone 3 = 10 sec * 20 m/s = 200 Mts.

(from the end of polysection = end of section 1). The vehicle is again in zone 3.

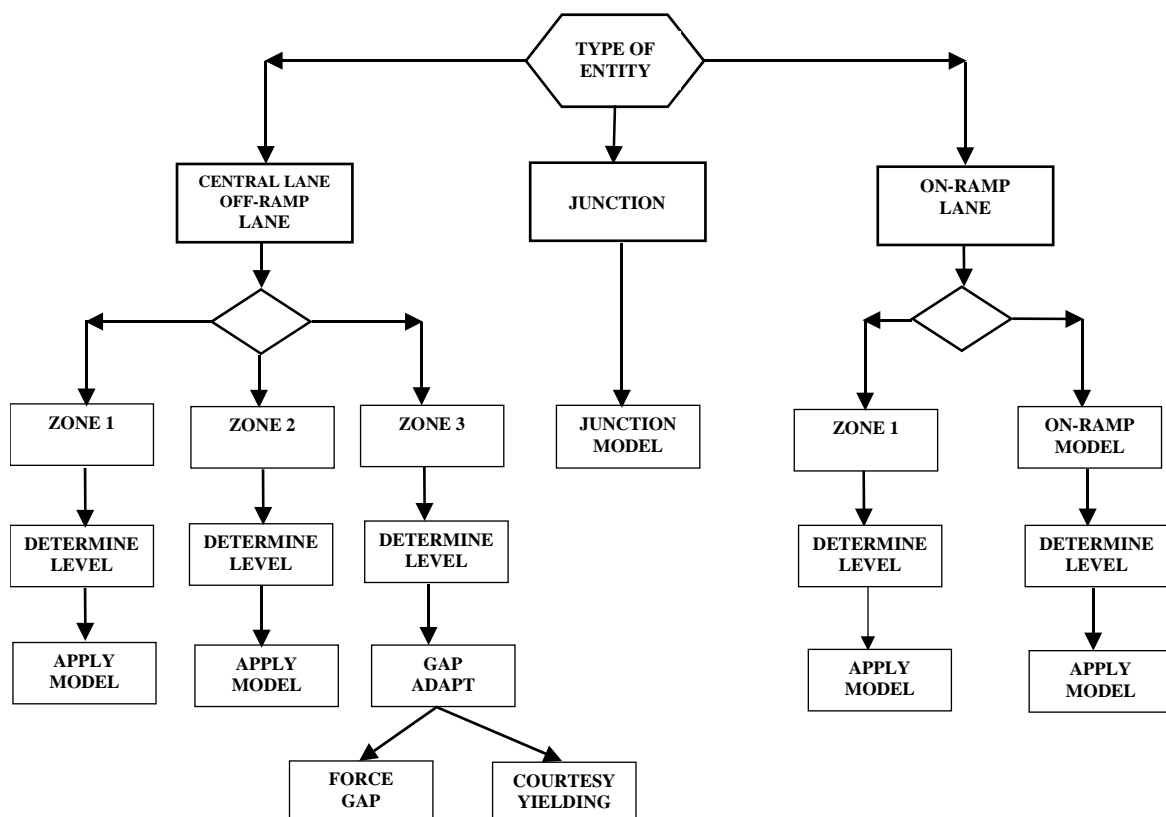
Figure 3-13: Example 2: Zones in a polysection



To make the examples easier, we have assumed in both examples that the vehicle desired speed of all vehicles is equal to the section speed limit (72 km/h). Notice that example 2 illustrates an inconsistent zone definition and should be avoided.

An overview of the lane-changing model is displayed in Figure 3-14. The system identifies the type of entity (central lane, off-ramp lane, junction, on-ramp etc.) in which the manoeuvre is to be carried out, and then determines how zone modelling should be applied. The current traffic conditions are analysed, and the level at which the lane change can be performed is determined, and then the corresponding model is applied.

Figure 3-14: Lane-changing model logic



Gap Acceptance Model

To answer the question “*Is it possible to change lanes?*” the following algorithm is applied to check whether a gap is acceptable or not.

```

Get downstream and upstream vehicles in target lane
Calculate gap between downstream and upstream vehicles: TargetGap
if ((TargetGap > VehicleLengh) & (it is aligned)) then
    Calculate the distance between vehicle and downstream vehicle in target lane: DistanceDown
    Calculate the speed imposed by downstream vehicle to vehicle, according to Gipps
        Car-following Model: ImposedDownSpeed
    if (ImposedDownSpeed is acceptable for vehicle, according to the deceleration rate) then
        Calculate the distance between upstream vehicle in target lane and vehicle: DistanceUp
        Calculate the speed imposed by vehicle to upstream vehicle, according to Gipps
            Car-following Model: ImposedUpSpeed
        if (ImposedUpSpeed is acceptable for upstream vehicle, according to the deceleration rate) then
            Lane Change is Feasible
            CarryOutLaneChange
        else
            The gap is not acceptable because of the upstream vehicle
        endif
    else
        The gap is not acceptable because of the downstream vehicle
    endif
else
    There is no gap aligned with the vehicle
endif

```

3.5.6 Overtaking Manoeuvre

An overtaking manoeuvre takes place mainly in Zone 1, although it can also take place in Zones 2 when the vehicle is in the appropriate turning lane. In order to promote or discourage overtaking, there are two parameters that the user can define: *Percent Overtake* and *Percent Recover*.

Percent Overtake is the percentage of the desired speed of a vehicle below which the vehicle may decide to overtake. This means that whenever the leading vehicle is driving slower than *Percent Overtake* % of the follower's desired speed, the following vehicle will try to overtake. The default value is 0.90.

Percent Recover is the percentage of the desired speed of a vehicle above which a vehicle may decide to get back into the slower lane. This means that whenever the lead vehicle is driving faster than *Percent Recover* % of the follower's desired speed, the following vehicle will try to get back into the rightmost (or leftmost) lane. The default value is 0.95.

We recommend that you use Percentage Recover values that are greater than Percentage Overtake, otherwise some overtaking manoeuvres may be aborted. Even more importantly, combinations of parameters in which Percentage Recover are considerably smaller than Percentage Overtake may lead to strange behaviour patterns. For instance, let us assume a value of 0.90 for Percentage Overtake and a value of 0.50 for Percentage Recover:

1. Consider a vehicle driving at a speed of 90 km/h with a desired speed of 100 km/h and a lead vehicle driving at a speed of 88 km/h.
2. As the lead vehicle is driving slower than 90% of the following vehicle's desired speed, the vehicle decides to overtake.

3. The vehicle is overtaking at 90 km/h. As it is now driving faster than the 50% of its desired speed, it decides to return to the slower lane.

Percentages that are too small, such as 0.01, 0.1, 0.2, are not appropriate. For example, Percentage Overtake equal to 0.01 would mean only overtaking those vehicles driving slower than 1% of desired speed (i.e., those that are almost stopped). Percentage Recover equal to 0.01 would mean returning to the slow lane when driving faster than 1% of desired speed (which means returning to the slow lane when almost stopped).

These parameters mainly influence the lane-changing model in zone 1, they have some influence in zone 2, but no influence in zone 3. Therefore, sections shorter than Distance Zone 2 may not be affected by these parameters at all. The reason that these parameters have no effect in zone 3 is because the main motivation to change lane in zone 3 is to get to the proper turning lane and not to achieve a desired speed as in zones 1 and 2.

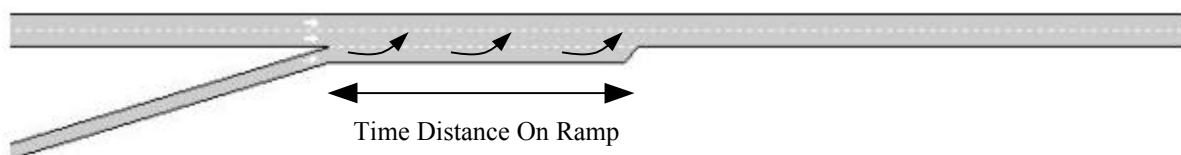
3.5.7 On-Ramp Model

The On-Ramp model is a particular instance of a simple Merging Model. A special Lane Changing Model is applied to vehicles while driving in an On-Ramp lane in order to model the manoeuvre of merging into the main stream. An additional zone parameter, *Time Distance On Ramp*, may be defined for each section. The purpose of this parameter is to distinguish between side lanes used as on-ramps and those long side lanes that are used as slow lanes or overtaking lanes that drop down. *Time Distance On Ramp* is the distance, in seconds, to the end of the lane from which a side lane is considered to be an on-ramp lane.

Vehicles driving on a side lane that are farther than *Time Distance On Ramp* from the end of the lane behave as if they were in Zone 1 of a normal lane. When they are closer than *Time Distance On Ramp* to the end of the lane, they behave as though they had to merge from an on-ramp.

Figure 3-15 displays an example of a side lane, all of which is considered as being an on-ramp. Therefore, vehicles try to merge into the main stream as soon as they enter the on-ramp lane. Also, vehicles in the main stream will not use the on-ramp as a slow lane.

Figure 3-15: On-ramp lane



On the other hand, Figure 3-16 displays a very long lateral lane, only the last part of which is considered as an on-ramp and used for merging. The first part of the lane can be used as a normal lane and therefore vehicles may merge into the main stream if they are overtaking, and vehicles from the main stream may enter the side lane to use it as a slow lane. When vehicles get closer than *Time Distance On Ramp* to the end of the side lane, they start trying to merge into the main stream.

Figure 3-16: Long On-ramp lane



The third example, Figure 3-17, displays a side lane that is used as a slow lane. Vehicles enter the lane considering it as the slower lane and they will use it until they reach the final *Time Distance On Ramp* area, when they will try to merge into the main stream again.

Figure 3-17: Slow lane



The final example in Figure 3-18 displays a side lane used as an overtaking lane. A merging behaviour, similar to that for an on-ramp, is applied at the end of the lane to merge back into the main stream.

Figure 3-18: Overtaking lane

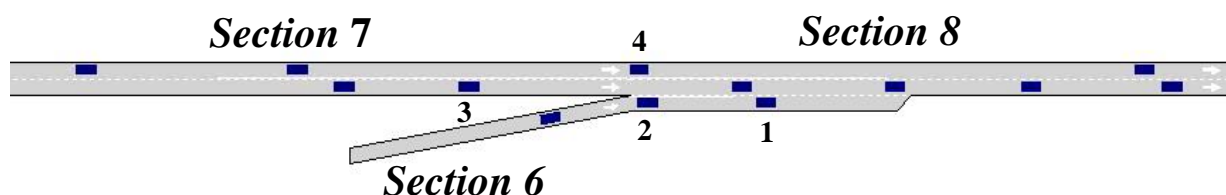


Lane changing in an on-ramp model takes into account aspects such as whether or not the vehicle is the first at the on-ramp, whether the vehicle is braking or even has reached a stop and how long the vehicle has been waiting for a gap. Another vehicle parameter, *Maximum Give-way Time*, is used to determine how long a vehicle is willing to wait before getting impatient. When a vehicle has been stopped at the end of the on-ramp more than the *Maximum Give-way Time*, the Lane Changing as in Zone 3 is applied. This influences the behaviour of adjacent vehicles in order to create suitable gaps, even making them to stop if necessary, although the deceleration and acceleration rates used to update the vehicle are not changed.

On the other hand, a special behaviour model for vehicles driving in the main lane when approaching an on-ramp has been implemented. The logic of the lane change model is not modified. What we have done is to include a new 'reason' for making lane changing 'necessary'. A vehicle driving in zone 3 of a section that is approaching a section with an on-ramp will check whether there are vehicles trying to merge from the ramp into the main stream. If so, it will decide that it is necessary to move to the left. This is also applied to vehicles driving on the same section as the on-ramp but before the end of the ramp. The lane change applied here is as in zone 1, so it is not very urgent. The result is that vehicles try to change to the outer lane in order to facilitate the entrance of other vehicles into the on-ramp. For example, in Figure 3-19, vehicle 3 will try to change to the left lane in order to let vehicles 1 and 2 merge without problems. However, keep in mind that vehicles tend to go to the leftmost lane only in case that there is enough safe space in that lane. Whenever the left lane is also congested, vehicles may not manage to move to the left, as they apply lane change model as zone 1, which is low intensity. Notice that this behaviour model is applied from zone 3 of the previous section and is maintained until the vehicle has passed beyond the point where the off-ramp ends.

The lane changing in AIMSUN v3.3 has been improved as compared to v3.2, so that the on-ramp model has been improved so that more vehicles can merge into the main stream without reaching the end of the on-ramp. In any case and in order to remain consistent with previously calibrated networks, the user may choose between using the previous v3.2 on-ramp model or the new v3.3 or later.

Figure 3-19: On-Ramp Model



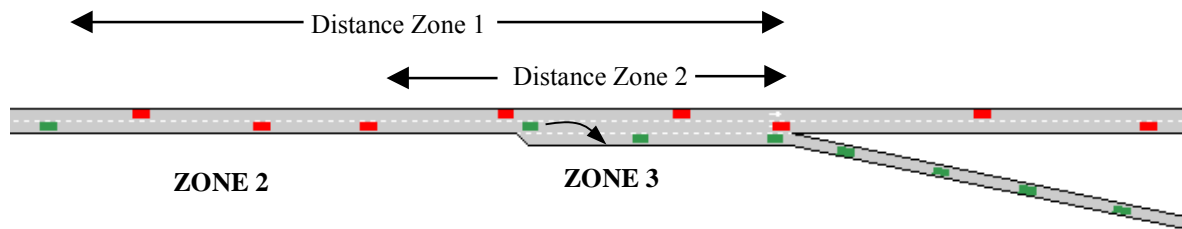
3.5.8 Off-Ramp Model

The Off-Ramp model is a particular instance of a Diverging Model. The standard Lane Changing Model is applied to model this simple diverging manoeuvre made by a vehicle driving on a freeway and attempting to exit the main stream via an Off-Ramp lane (see Figure 3-20).

A vehicle that is driving on a freeway and which wishes to exit via the next off-ramp, starts trying to move to the rightmost (or leftmost) lane of the main stream when it is in zone 2. As soon as the vehicle is aligned with the off-ramp lane, it will try to enter the side lane. The behaviour of the vehicle when looking for acceptable gaps when trying to enter the side lane will depend on the Lane Changing Zone, as it follows the

standard Lane Changing Model. If there is no available gap in the off-ramp side lane because of a heavily congested exit, a vehicle may even come to a full stop and wait for a gap in order to not miss its desired exit. The time a vehicle is willing to wait in this situation is limited by vehicle parameter *Maximum Give Way Time*.

Figure 3-20: Off-Ramp Model



A vehicle that is approaching an off-ramp lane without any disturbance from other vehicles will tend to reach the end of the off-ramp side lane at its desired turning speed which, as explained in section 3.5.2, is a function of its desired speed and the maximum turning speed.

3.5.9 Look Ahead Model

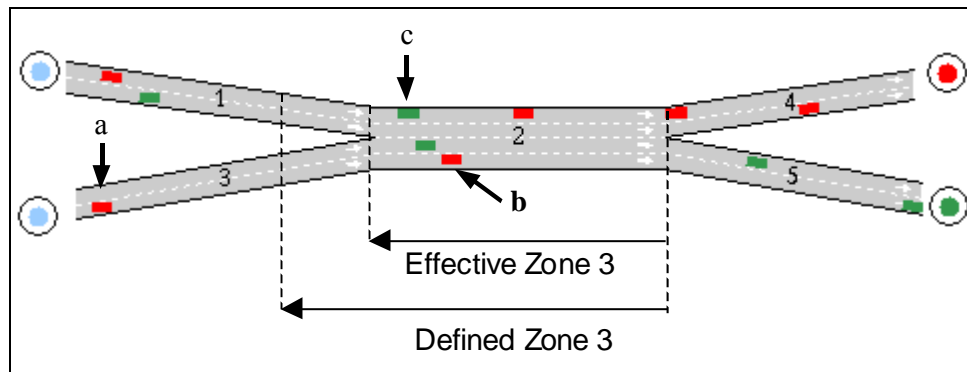
Until now we have made the assumption that a vehicle driving along a section only has knowledge of its next turning movement, that is the turning it will take when arriving at the end of the current section or polysection. This means that the lane changing decisions of each particular vehicle are made according to the next turning movement in the next junction or join. In urban networks where there are short sections or in a freeway situation where weaving sections may be relatively short, it is possible that some vehicles will not reach the appropriate turning lane and consequently miss the next turn. This situation could occur when traffic conditions are very congested and if we only take into account the next turning movement in the lane changing decisions.

Tuning some modelling parameters such as lane changing zone distances, simulation step, acceleration rates etc. could improve the behaviour in order to minimise the number of lost vehicles. Also, using polysections instead of sections, when feasible, to model streets or weaving areas might help to improve the situation, but this is not enough.

The following examples assume that there is no Look Ahead Model and they are compared later with a situation involving a Look Ahead Model. The objective is to illustrate the utility of this model and help to understand it.

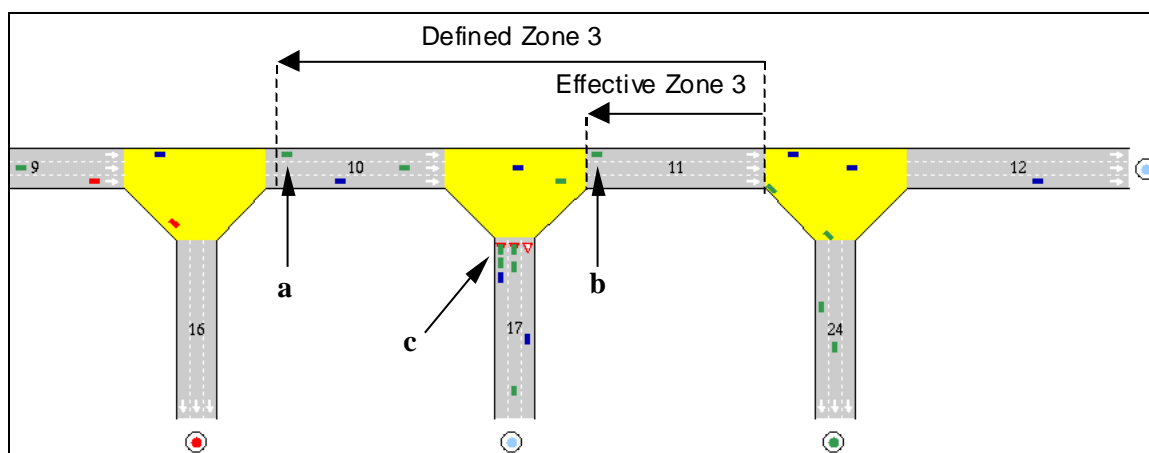
Consider the network in Figure 3-21 displaying a weaving situation. Vehicle **a** in section 3 is only considering the next turning movement, which is from section 3 to section 2. Although it wishes to go to the left (red) destination, it keeps driving on the right, as the next turn is feasible in this lane. When this vehicle reaches section 2 it will realise that the next desired turn is from section 2 to section 4, in order to reach red destination, so it will start looking for a gap, trying to get into the leftmost lanes. This is the case for vehicle **b**, which has to move two lanes to the left in order to get to a feasible lane. The same happens to vehicle **c**, which wants to reach the right (green) destination.

Notice that, in Figure 3-21, Zone 3 of section 2 extends back beyond the limits of section 2. One may think that this should mean that vehicles driving in sections 1 and 3 but also inside Zone 3 of section 2 should behave according to the turning movements of section 2. However, this is not the case in a situation without Look Ahead, as all vehicles only have knowledge of the immediate turn. Only when they enter section 2, do they realise that they are driving in Zone 3, and at that point reaching the appropriate turning lane becomes urgent.

Figure 3-21: Weaving without Look Ahead Model

Another example is provided in Figure 3-22. This case is an urban situation in which there are short sections between consecutive junctions. It is assumed that vehicles can only turn right using the rightmost lane. Vehicle **a** in section 10 should turn right in the second junction ahead, which is less than 200 Mts. However, it is driving in the leftmost lane. Even vehicle **b** that has to turn right in the next junction is driving on the left because it has just entered section 11 and has not had time to change lanes yet. Also, vehicles **c** that are yielding in the left lane of section 17 waiting to turn right, are supposed to turn right again in the next junction. However, they will enter section 11 through the leftmost lane and will be in a similar position to vehicle **b**. As in the previous example, Zone 3 of section 11 extends back through section 10, but it does not affect the lane changing decisions in the upstream section.

This method for modelling the turning decisions can cause vehicles to miss their desired turn because they will be unable to reach the appropriate turning lane on time. When simulating in Result Based mode, this could cause the simulation of turning proportions that cannot be matched exactly with the input data. When simulating in Route-Based mode, this could cause some vehicles to get lost and to remain unable to reach their destination.

Figure 3-22: Urban situation without Look Ahead Model

In order to avoid this undesirable behaviour as much as possible, we have included the Look Ahead model in AIMSUN v4.0, whose main purpose is to make the vehicles reach the turning lane earlier. The idea is to provide vehicles with the knowledge of additional next turning movements, rather than just one, enabling them to make decisions based not on the immediate next turning movement, but on a set of next turning movements. For the purposes of reducing computing time and memory requirements we have decided to reduce this set to two turning movements ahead. We do not consider this to be a limitation, as very few lane-changing decisions are made while considering more than the next two turns.

The Look Ahead model basically consists of four points:

1. At any time, each vehicle knows the next two turning movements, so the lane changing decisions are influenced by two consecutive turns.

2. Lane changing zones 2 and 3 of any section are extended back beyond the limits of the section, therefore affecting the upstream sections.
3. The next turning movement also influences the turning manoeuvres so the selection of destination lane is made based also on the next turn.
4. Greater variability is given to the Lane Changing Zones in order to distribute the lane changing manoeuvres along a longer distance.

These four points are now described in greater detail.

Influence of two turning movements in lane changing

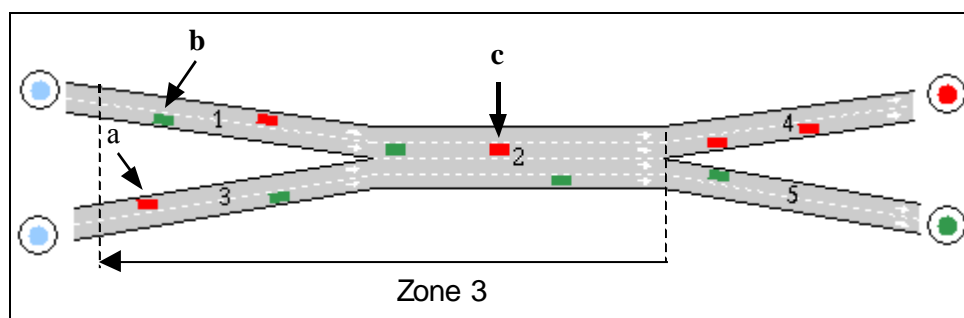
When a vehicle is generated and introduced into the network, two turning movements are calculated: the *next turn* and the *ahead turn*. This means that the vehicle has knowledge of the first three consecutive sections (or polysections) that it will have to follow: the current input section, the second defined by the *next turn*, and a third section that is defined by the second section and the *ahead turn*.

Then, each time a vehicle enters a new section (or polysection), the *ahead turn* becomes the *next turn* and a new *ahead turn* is calculated. The turning movements between sections of a polysection are not considered. Therefore when a turn leads to a polysection, the next or ahead turns refer to a turn of the last section of the polysection, never to internal turns of the polysection.

The behaviour of a vehicle driving in zones 2 or 3 of a section is mainly governed by the *next turn*, therefore it will first try to reach a lane where the *next turn* is feasible. Once it is driving in an appropriate lane with respect to the *next turn*, it will take into account the *ahead turn*. This involves checking whether it is already driving inside the extended zone 2 or 3 of the next section. In this case it will try to find the best lane in the current section that, still allowing for *next turn*, will send the vehicle either to a lane in the next section where the *ahead turn* is feasible or to the lane closest to a feasible one.

This is illustrated in the example displayed in Figure 3-23. Vehicle **a** in section 3 is already located in the left lane in order to be as close as possible to the left when entering section 2. In this case the *next turn* is from 3 to 2 while the *ahead turn* will be from 2 to 4. Therefore, although the rightmost lane would be appropriate for the *next turn*, it is not appropriate for the *ahead turn*. Similar behaviour can be seen in vehicle **b**, which has changed to the rightmost lane in section 1 in order to reach the *ahead turn* to the right.

Figure 3-23: Weaving with Look Ahead Model



Extending back Zones 2 and 3

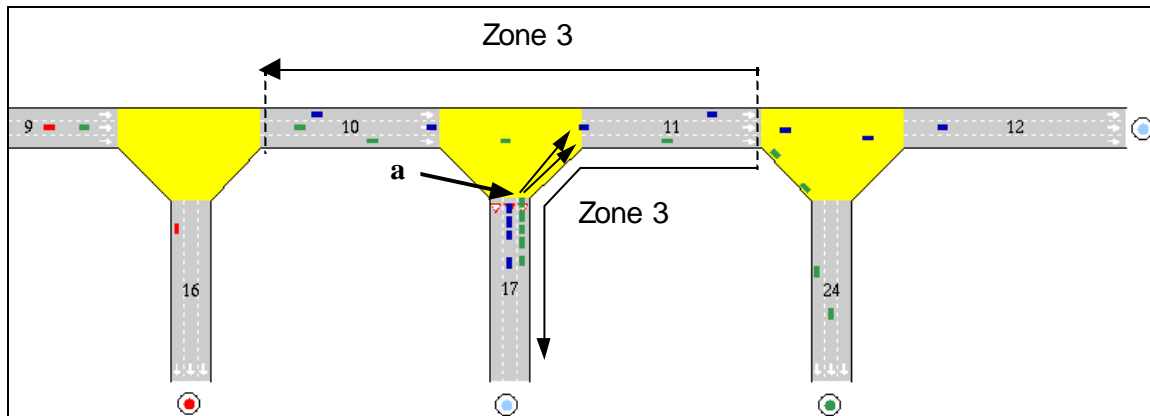
If Zone 2 or 3 of a section is longer than the length of that section, the zone is extended beyond the limits of the section, therefore entering into the upstream section. In order to make the Look Ahead model work properly it is now very important to define the lane changing zones appropriately. The Look Ahead model will not be applied whenever the lane changing zones 2 or 3 are shorter than a section. If a polysection is involved, the user must take care to properly define the zones of each section of the polysection. The zone extension is illustrated in Figure 3-24.

Turning manoeuvres

When a vehicle reaches the end of a lane in a section and enters into a junction, it may have the option of choosing among different connections, i.e. movements from origin lane (the current one) and destination lanes (the lane in the next section).

For instance, in Figure 3-24, vehicle **a** is going to turn right from the rightmost lane of section 17 to section 11. However, it can choose between the different destination lanes in section 11. As this vehicle's ahead turn is to the right again, it will enter section 11 via the rightmost lane instead of using the central one.

Figure 3-24: Turning manoeuvres with Look Ahead Model



Variability of Lane Changing Zones

When a vehicle crosses from zone 1 to zone 2 there is a change in the vehicle's behaviour, as the next turn now becomes relevant. Also, crossing from zone 2 to zone 3 produces a change in the behavioural rules of the vehicles, as reaching the turning lane now becomes urgent. In order to distribute these changes of behaviour along a longer distance, greater variability is given to the Lane Changing Zones. These zones are calculated especially for each vehicle according to the following equation:

$$\text{Distance Zone } n \text{ for vehicle } v \text{ in section } s \text{ (in meters)} = \\ \text{Distance Zone } n \text{ (in seconds)} * \text{Speed Limit of Section } s * \text{Vehicle } v \text{ Coefficient}$$

$$\text{Vehicle } v \text{ Coefficient} = \text{Speed Limit of Section } s / \text{Desired speed of Vehicle } v \text{ in section } s$$

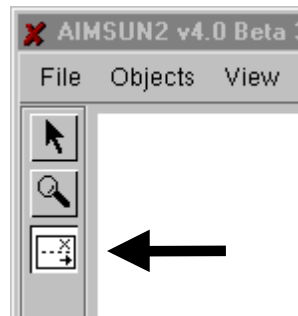
This algorithm ensures that for vehicles whose desired speed is slower than the speed limit, the lane changing zones will be longer than for vehicles whose desired speed is greater than the speed limit. This means, for instance, that a heavy truck will try to reach the appropriate turning lane earlier than a speed car.

3.6 TRAFFIC INCIDENTS

Traffic incidents can be simulated in AIMSUN. We consider a Traffic Incident to be any traffic event that causes lanes blockage over a certain time period. Examples of incidents are: a heavy goods vehicle loading or unloading, a taxi picking up or dropping off a passenger, a broken down vehicle, road works, etc.

The user can create traffic incidents using the Tool Bar. To define an incident, press on the Incidents button, which is the third icon button in the Icon Tool Bar, as indicated in Figure 3-25.

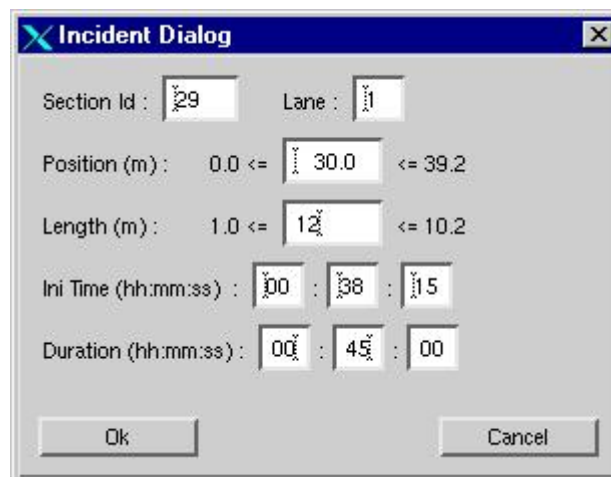
Figure 3-25: Incidents option in tool bar



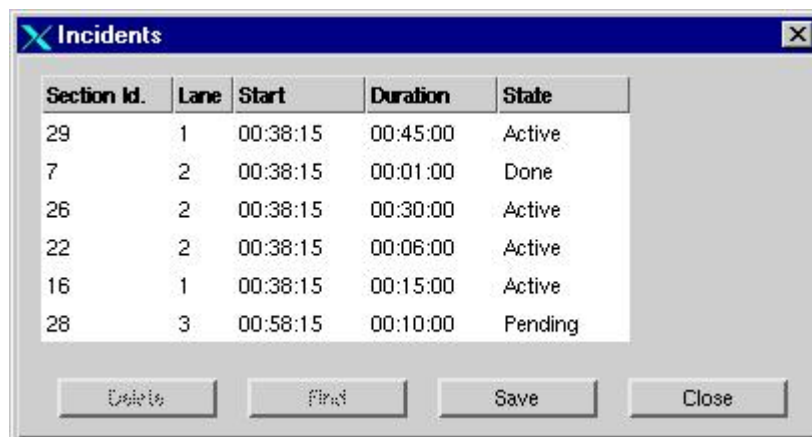
When this icon appears to be pressed down click on the section and lane in the network display to identify the exact position at which the incident is to occur and the 'Incident Definition' window (see Figure 3-26), will appear.

This window displays the section identifier, the lane number (numbered from right to left) and the 'Position' (distance from the beginning of the section) that the user has just selected. It is then necessary to input the 'Length' of the incident (distance from the beginning of the incident until the end). The time at which the incident will take place ('Ini. Time') is set by default to the current simulation time, but the user can change this time if desired. Finally, the 'Duration' of the incident must be defined.

Figure 3-26: Incident Definition window



At any time during a simulation, the user can view the list of defined incidents for the current simulation run. This is done by selecting 'Experiment / Incidents' from the menu bar. The 'Incidents' log window will appear as shown in Figure 3-27. For each incident defined, the section identifier, lane number, starting time, duration and state is displayed. The state can be 'Done', 'Active' or 'Pending'. 'Done' means that the incident has finished, 'Active' means that it is currently happening and 'Pending' means that it has yet to occur.

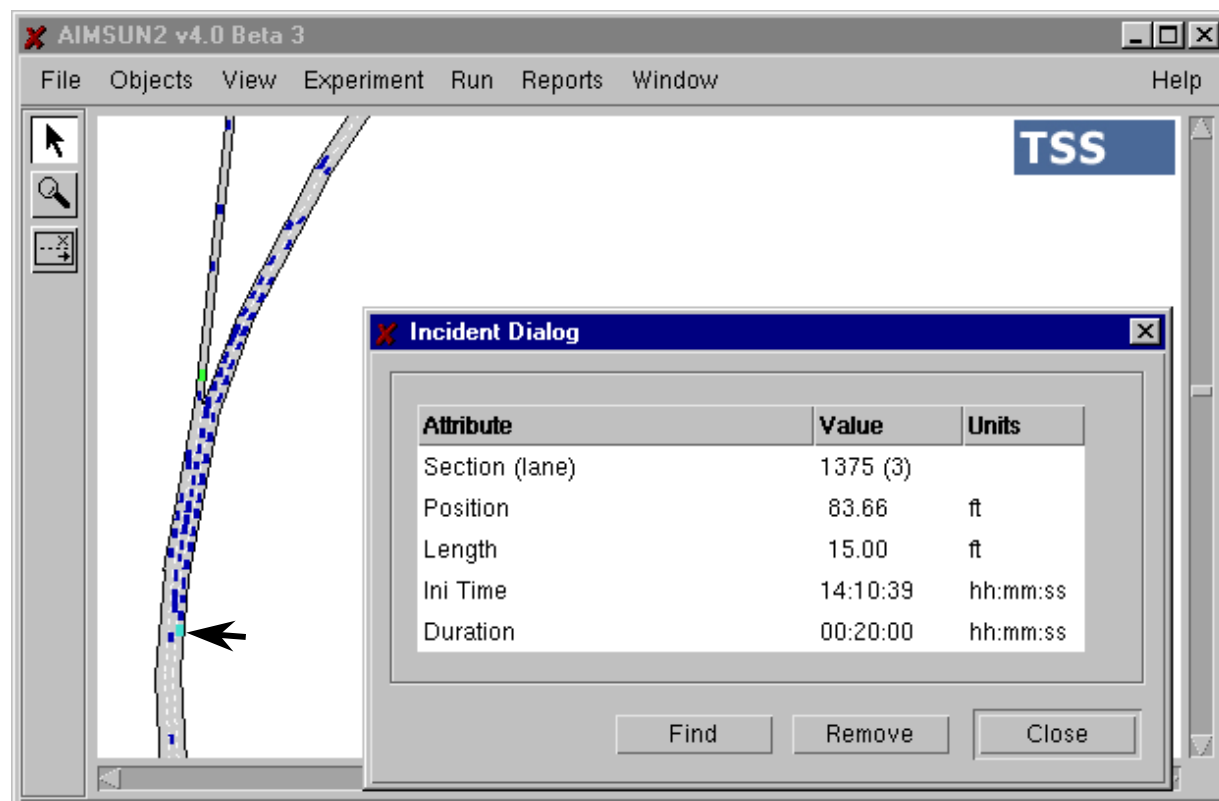
Figure 3-27: Incident log window


Section Id.	Lane	Start	Duration	State
29	1	00:38:15	00:45:00	Active
7	2	00:38:15	00:01:00	Done
26	2	00:38:15	00:30:00	Active
22	2	00:38:15	00:06:00	Active
16	1	00:38:15	00:15:00	Active
28	3	00:58:15	00:10:00	Pending

Buttons: Delete, Find, Save, Close

The user can delete an incident from the list at any time, independently of the state, by selecting the incident from the list box and pressing 'Delete'. An Active incident can be located using the 'Find' button. Finally, the whole set of incidents can be saved to a log file, which can be used for future simulation experiments. This is done using the 'Load Incidents' toggle button, which can be found in both the 'Load Traffic Result' and the 'Load O/D Matrix' windows (see section 7.1.2), when first loading the traffic demand data.

The user can also view incident data by double clicking on the incident icon directly in the network. This displays the Incident dialog window (see Figure 3-28). This dialog window provides the incident information: Section Identifier, lane number, length, time at which the incident started, and expected duration. An incident can be removed by selecting it and clicking on the 'Remove' button. This causes the incident to finish, the Incident State is changed from 'Active' to 'Done', but it is not removed from the Incidents Log.

Figure 3-28: Incident Dialogue Window

4. DYNAMIC TRAFFIC ASSIGNMENT

For Dynamic Traffic Assignment, the traffic demand is defined in terms of an O-D matrix, which gives the number of trips from every origin centroid to every destination centroid, for each time slice, for each vehicle type. When a vehicle is generated at an origin it is assigned to one of the available paths, connecting this origin to the vehicle's destination. The vehicle will travel along this path until reaching its destination unless it is allowed to dynamically change the route en-route (i.e. the guided vehicles) when a better route exists from its current position on the assigned path to its destination. The simulation process based on time dependent paths consists of the following steps:

Step 0: Calculate initial shortest path(s) for each O/D pair using the defined initial costs.

Step 1: Simulate for a predefined time interval (e.g. 5 minutes) assigning to the available path the fraction of the trips between each O/D pair for that time interval according to the selected route choice model and obtain new average link travel times as a result of the simulation.

Step 2: Recalculate shortest path, taking into account the experimented average link travel times.

Step 3: If there are guided vehicles, or variable message signs suggesting rerouting, provide the information calculated in 2 to the drivers that are dynamically allowed to reroute on trip.

Step 4: Go to step 1.

Which repeat until all the demand has been assigned instead of the convergence criteria used by the analytical model to control the length of the simulation.

To model the Dynamic Traffic Assignment procedure two main tasks have to be addressed:

Paths (routes can be used as synonymous) from origins to destinations have to be computed at each time interval.

The Path Selection (or Assignment) is the process applied to each vehicle, either when it enters the system or during its trip. The process consists on the selection of one path or route between all available paths in order to reach its destination.

4.1 PATH DEFINITION

The available paths from one origin to one destination, taken into account in the Path Selection process of a vehicle, can be either defined by the user (User-defined paths) or calculated, applying a shortest path algorithm, which uses the concept of “cost”, that will be explained in detail in section 4.1.2.

User-defined Paths: These paths correspond to the idea of well-known paths, or the most familiar paths for the drivers, from one origin to one destination according to the analyst knowledge of the modelled network, and are predefined by the analyst using the network editor, or taken as an output from other traffic simulators or transportation models, either macroscopic (i.e. a transport planning) or microscopic.

Calculated Shortest Paths: Resulting from applying the shortest path algorithm to a network representation in terms of links and nodes, where each link has associated a cost function (i.e. travel time) that will be used in the shortest path calculation.

4.1.1 Network Representation

The network representation used for the microscopic simulation, in terms of sections and intersection, is not adequate for the shortest path computation procedure, which requires a link-node representation. To account explicitly for turning movements in the translation from the AIMSUN representation to the link-node representation a link, connecting two nodes, models both a section and a turning movement. Therefore, each AIMSUN section is split into as many links as turning movements. The computation of shortest paths uses a label-setting method, where the labels are associated with an link, which means that different costs can be assigned to each turning of a section.

Figure 4.1 depicts an example of a AIMSUN network composed of sections, junctions and joins. The corresponding network representation used by the shortest routes component, composed of nodes and links, is shown in Figure 4-2. Notice that for each section, a node is created and there is an link for each turning movement. The cost assigned to each arc is a function of the travel time of the section plus the travel time of the turning movement.

Figure 4-1: Example of AIMSUN Network

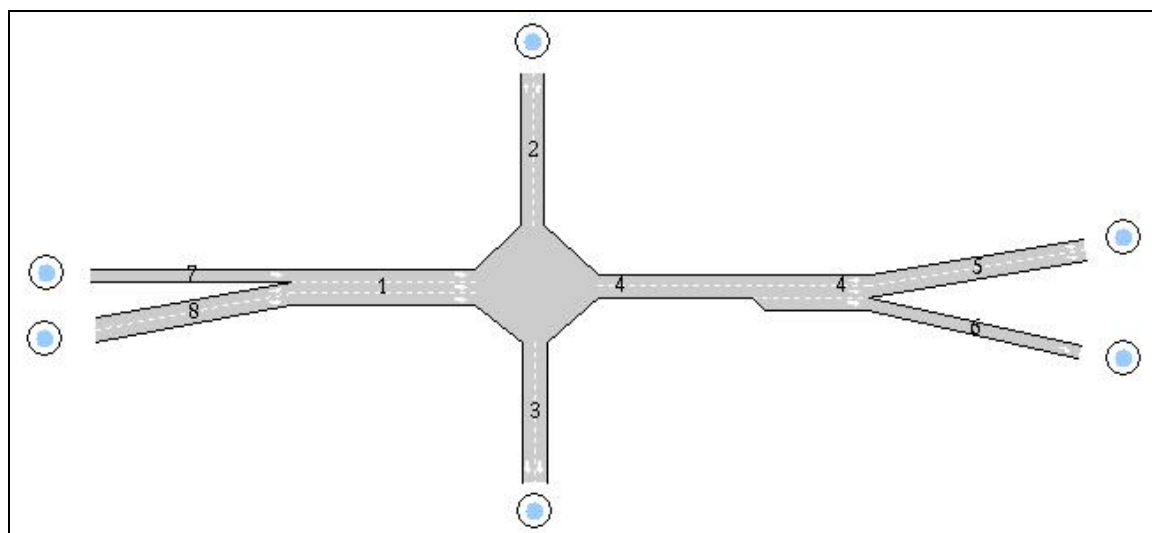
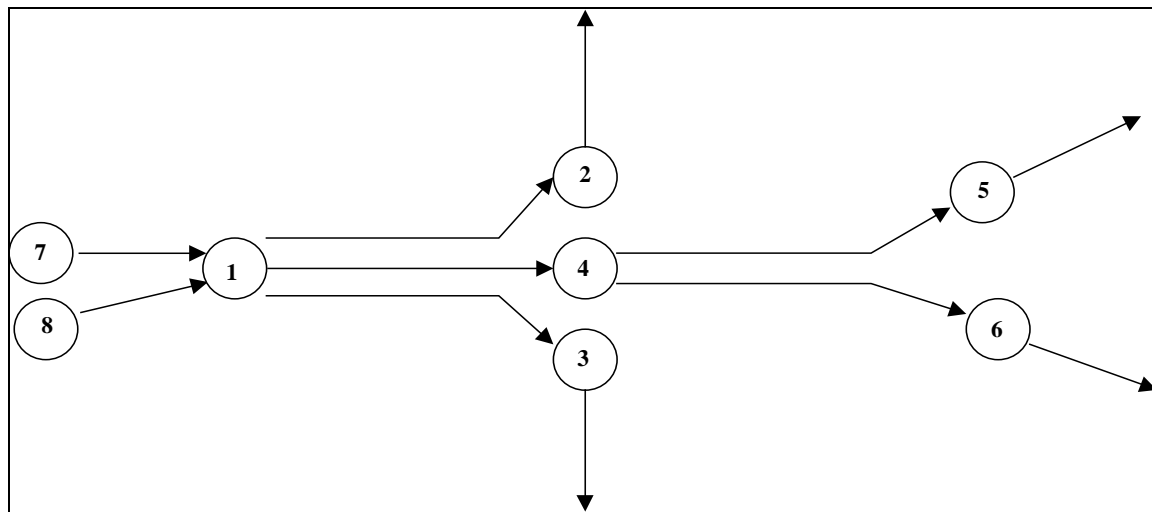
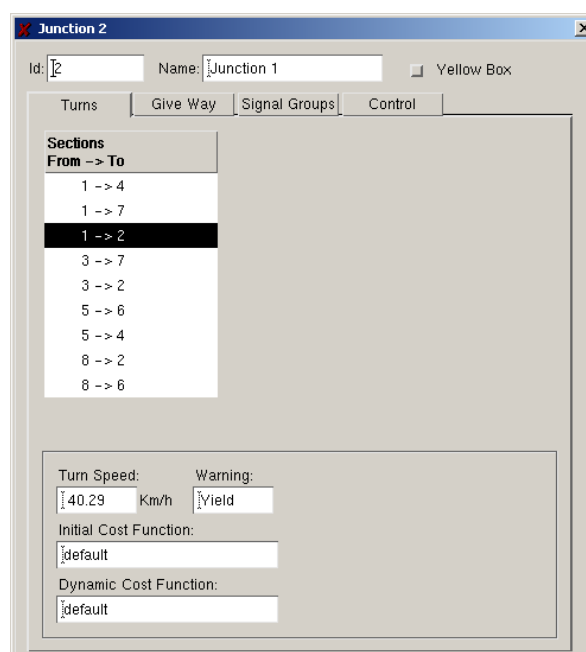


Figure 4-2: Representation of previous Network for Shortest Path Calculation

4.1.2 Link Cost Functions

In the link-node representation of the network used in the shortest routes calculation, each link is associated with a cost function. Two types of link cost functions are used for calculating the shortest path, depending on whether or not simulated data (i.e. simulated travel time) are available for use. These are the *Initial Cost Function* and the *Dynamic Cost Function*. In both cases, there is a default cost function, which represents link travel time in seconds (a section travel time plus the turning movement travel time, this last term introduces the penalty of the turning movement, if it exists).

For each particular arc, the user may choose between using Initial Cost Function or Cost Function as the default cost function, or using any other cost function defined by the user. This is done via Tedi, although the user can, through the AIMSUN interface, view the cost function names that are associated to each arc by clicking on the Turns list box of the Junction dialog window (see Figure 4-3).

Figure 4-3: Cost Functions for each turn

4.1.2.1 Link Capacity

A numerical attribute of a link is the theoretical capacity that can also be taken into account as argument in the cost function. This capacity is calculated using the theoretical user defined section capacity of the sections to which the link belongs. The calculation of the link capacity considers the number of lanes of the section that are used for the turning movement corresponding to each link and the number of turning movements shared for each lane.

The weight of a lane i is defined as:

$$WL_i = \begin{cases} 1 & , \text{ if } i^{\text{th}} \text{ lane is a central lane} \\ L_i / L_s & , \text{ if } i^{\text{th}} \text{ lane is an exit side lane} \end{cases}$$

where L_i is the length of lane i and L_s is the length of section s , which contains lane i

The weight of a lane denotes the lane contribution in the calculation of the link capacity. When the lane is a central, it has a complete contribution to the capacity link, that is 1 as weight and when this lane is a exit side lane then the contribution is proportional to its length compared to the length. of section in which belongs. In Figure 4-4 the weight of the side lane is 0.2 (10/50) and in Figure 4-5 the weight of the side lane is 0.8 (40/50)

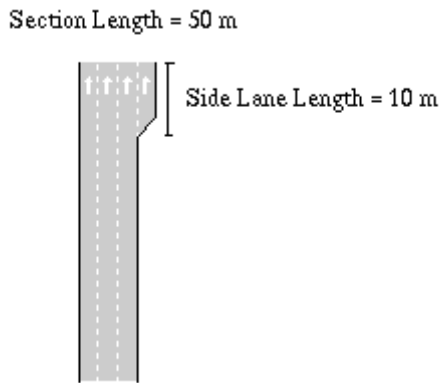


Figure 4-4: Example of short side lane

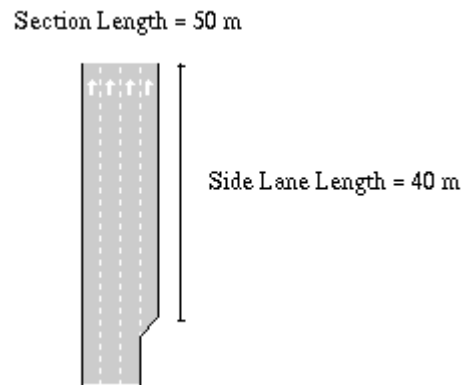


Figure 4-5: Example of long side lane

Then, the generic capacity per lane in section s is defined by:

$$GCL_s = C_s / \sum_{i \in \text{exit lane}} WL_i$$

where C_s is the capacity of section s .

The capacity of link j (composed by section s plus turning movement t) is defined by:

$$CL_j = \sum_{k \in \text{exit lanes of link } j} GCL_s * WL_k / NTS_k$$

where NTS_k is the number of turning movements that lane k shares.

Figure 4-6 shows an AIMSUN network composed by 3 sections (Section 1, Section 2 and Section 3) and an intersection defined by two turning movements: One from Section 1 to Section 2, shown in Figure 4-7 (using the two right most lanes in section 1) and the second, Figure 4-8 is from Section 1 to Section 3 (using the two left most lanes in section 1)

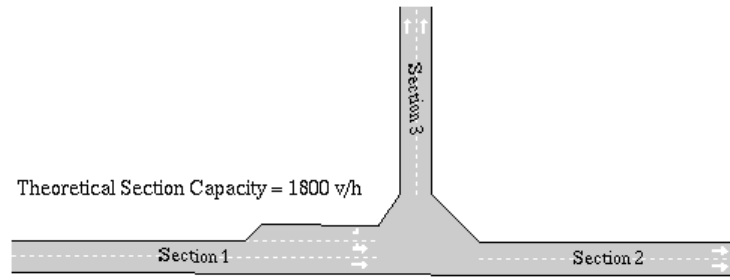


Figure 4-6: Example of AIMSUN network

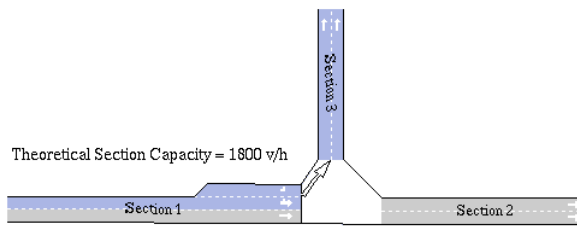


Figure 4-7: Left turning

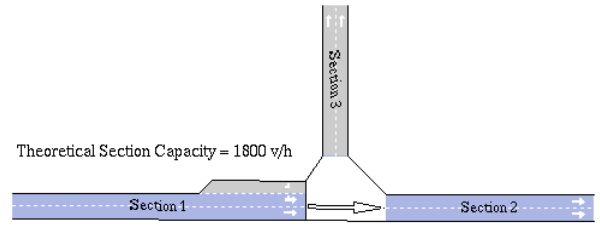


Figure 4-8: Right turning

The corresponding network representation used by the shortest routes algorithm, composed of nodes and links, is shown in Figure 4-9.

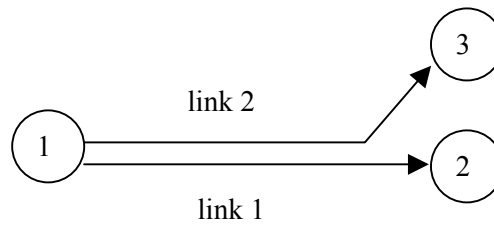


Figure 4-9: Network of Figure 4-6 represented in terms of links and nodes.

The generic capacity per lane in section 1, where side lane has a weight of 0,35:

$$GCL_1 = C_1 / \sum_{i \in \text{exit lane}} WL_i = 1800 / (1 + 1 + 0,35) = 765,96$$

The capacity of link 1 is:

$$CL_1 = \sum_{k \in \text{exit lanes of link 1}} GCL_1 * WL_k / NTS_k = (765,96 * 1/1) + (765,96 * 1/2) = 1148,94$$

and the capacity of link 2 is:

$$CL_2 = \sum_{k \in \text{exit lanes of link 2}} GCL_1 * WL_k / NTS_k = (765,96 * 1/2) + (765,96 * 0,35/1) = 651,06$$

4.1.2.2 Initial Cost Function

The *Initial Cost Function* is used at the beginning of the simulation when there is not yet simulated data gathered to calculate the travel times. In this case, the cost of each link is calculated as a function of the travel time in free flow conditions and the capacity of the link. The travel time in free flow conditions is the time it would take a vehicle to cross the link, that is the section plus the turn, assuming that the vehicle is travelling at the maximum allowed speed along the section and at the maximum turning speed along the turning movement. No penalty for traffic lights or traffic signals is considered directly.

There are two types of default initial cost function. The first does not consider the vehicle types $IniCost_j$, that is, all vehicle types have the same initial cost. The second considers the vehicle types $IniCost_{j, vt}$, which means that there is an initial cost function per vehicle type. Whether to choose the initial cost function distinguishing per vehicle type or not is determined by the presence of reserved lanes in the network. By default, AIMSUN takes the initial cost function not considering vehicle types. However, if there are reserved lanes, it takes the initial cost function considering vehicle types.

The initial cost of link j , $IniCost_j$, taking into account the penalty for the turning t that belongs to the link, and its capacity CL_j is calculated as follows:

$$IniCost_j = TravelTFF_j + TravelTFF_j * \varphi * (1 - CL_j / CL_{max})$$

Where:

$TravelTFF_j$ is the estimated travel time of the link j in free flow conditions

$$TravelTFF_j = Length_s / SpeedLimit_s + Length_t / SpeedLimit_t$$

where $Length_s$ and $SpeedLimit_s$ are the length and the speed limit, respectively, of section s that belongs to link j and $Length_t$ and $SpeedLimit_t$ are the length and the speed limit, respectively, of turning t that belongs to link j .

φ is a user-defined capacity weight parameter that allows the user to control the influence that the link capacity has in the cost in relation with the travel time, and

CL_{max} is the theoretically estimated maximum link capacity in the network.

The initial cost of link j per vehicle type vt , $IniCost_{j, vt}$, is calculated as follows:

$$IniCost_{j, vt} = TravelTFF_{j, vt} + TravelTFF_{j, vt} * \varphi * (1 - CL_j / CL_{max})$$

Where:

$TravelTFF_{j, vt}$ is the estimated travel time of vehicle type vt in the link j in free flow conditions

$$TravelTFF_{j, vt} = \frac{Length_s}{\text{Min}(SpeedLimit_s * \theta_{vt}, MaxSpeed_{vt})} + \frac{Length_t}{\text{Min}(SpeedLimit_t * \theta_{vt}, MaxSpeed_{vt})}$$

Where the new parameters, compared with the travel time in free-flow conditions without distinguishing per vehicle type, are the Average Maximum Speed of vehicle type vt $MaxSpeed_{vt}$ and the Speed Acceptance of vehicle type vt θ_{vt} . This parameter ($\theta \geq 0$) can be interpreted as the 'level of goodness' of the drivers or the degree of acceptance of speed limits. $\theta_{vt} \geq 1$ means that the vehicle will take as maximum speed for a section a value greater than the speed limit, while $\theta_{vt} \leq 1$ means that the vehicle will use a lower speed limit. φ is a user-defined capacity weight parameter that allows the user to control the influence that the link capacity has in the cost in relation with the travel time, and CL_{max} is the theoretically estimated maximum link capacity in the network.

4.1.2.3 Dynamic Cost Function

The *Dynamic Cost Function* is used when there is simulated travel time data available, and therefore it can only be used when the simulation has already started and statistical data has been gathered. The default current cost for each section is the mean travel time, in seconds, for all simulated vehicles that have crossed the link during the last time data gathering period.

As for the *Initial Cost Function*, there are also two types of default dynamic cost function. The first does not consider vehicle types $DynCost_j$, which means all vehicle types have the same cost. The second considers all vehicle types $DynCost_{j, vt}$, which means that there is a dynamic cost function per vehicle type. Whether to use the dynamic cost function distinguishing per vehicle type or not is determined by the presence of reserved lanes in the network.

The default Dynamic Cost for link j common to all vehicle types, $DynCost_j$, is the mean travel time, in seconds, for all simulated vehicles that have crossed the link during the last statistics gathering period ($TravelTime_j$). The travel time for link j includes the travel time of section s plus the travel time for turning movement t .

As there may be situations in which no vehicles have crossed a link, so no information about travel time is available, the following algorithm is applied to calculate $EstimatedTravelTime_j$:

```

if ( $Flow_j > 0$ ) then
     $EstimatedTravelTime_j = TravelTime_j$ 
else
    if (there is any vehicle stopped) then
         $EstimatedTravelTime_j = AvgTimeIn_s$ 
    else
         $EstimatedTravelTime_j = AverageSectionTravelTime_s$ 
    endif
endif
 $EstimatedTravelTime_j = Maximum(EstimatedTravelTime_j, TravelFF_j)$ 

```

According to this algorithm, when some vehicles have crossed link j during the last data gathering period ($Flow_j > 0$), the estimated travel time ($EstimatedTravelTime_j$) is taken as the simulated mean travel time. If no vehicle has crossed link j , we should distinguish between a totally congested link and an empty link. In the first case, the travel time is calculated as the average waiting time for vehicles waiting in front of the queue in section s ($AvgTimeIn_s$). In the second case, the travel time is taken as the section travel time, which means considering all the turning movements that have as origin section s ($AverageSectionTravelTime_s$). All calculated travel time have to be greater than or equal to the travel time of the link in free-flow conditions.

Finally, the Dynamic cost of link j , $DynCost_j$, taking into account the penalty for the turning t that belongs to the link, and its capacity CL_j is calculated as:

$$DynCost_j = EstimatedTravelTime_j + EstimatedTravelTime_j * \varphi * (1 - CL_j / CL_{max})$$

where:

$EstimatedTravelTime_j$ is the estimated travel time of the link j calculated following previous algorithm

φ is a user-defined capacity weight parameter that allows the user to control the influence that the link capacity has in the cost in relation with the travel time, and

CL_{max} is the theoretically estimated maximum link capacity in the network.

The default dynamic cost for link j and vehicle type vt , $DynCost_{j,vt}$ is the mean travel time, in seconds, for all simulated vehicles of type vt that have crossed the link during the last data gathering period ($TravelTime_{j,vt}$). The travel time for link j includes the travel time of section s plus the travel time for turning movement t .

As there may be situations in which no vehicles that belongs to vehicle type vt have crossed a link, so no information about travel time is available, the following algorithm is applied to calculate $EstimatedTravelTime_{j,vt}$:

```

if ( $Flow_{j,vt} > 0$ ) then
     $EstimatedTravelTime_{j,vt} = TravelTime_{j,vt}$ 
else
    if (there is any vehicle  $vt$  stopped) then
         $EstimatedTravelTime_{j,vt} = AvgTimeIn_{s, vt}$ 
    else
        if (link  $j$  has reserved lanes of vehicle class  $cl$ ) then
            if ( $vt$  belongs to  $cl$ ) then
                if ( $FlowClass_{j, cl} > 0$ ) then
                     $EstimatedTravelTime_{j,vt} = TravelTimeClass_{j, cl}$ 
                else
                    if (there is any vehicle belonging  $cl$  stopped) then
                         $EstimatedTravelTime_{j,vt} = AvgTimeInClass_{s, cl}$ 
                    else
                         $EstimatedTravelTime_{j,vt} = AverageSectionTravelTime_{s, cl}$ 
                    endif
                endif
            else
                if ( $FlowClass_{j, not\ cl} > 0$ ) then
                     $EstimatedTravelTime_{j,vt} = TravelTimeClass_{j, not\ cl}$ 
                else
                    if (there is any vehicle not belonging  $cl$  stopped) then
                         $EstimatedTravelTime_{j,vt} = AvgTimeInClass_{s, not\ cl}$ 
                    else
                         $EstimatedTravelTime_{j,vt} = AverageSectionTravelTime_{s, not\ cl}$ 
                    endif
                endif
            endif
        else
            if ( $Flow_j > 0$ ) then
                 $EstimatedTravelTime_j = TravelTime_j$ 
            else
                if (there is any vehicle stopped) then
                     $EstimatedTravelTime_j = AvgTimeIn_s$ 
                else
                     $EstimatedTravelTime_j = AverageSectionTravelTime_s$ 
                endif
            endif
        endif
    endif
endif
 $EstimatedTravelTime_{j, vt} = Maximum(EstimatedTravelTime_{j, vt}, TravelFF_{j, vt})$ 

```

According to this algorithm, when a vehicle of type vt has crossed link j during the last data gathering period ($Flow_{j,vt} > 0$), the current cost is taken as the simulated mean travel time. If no vehicle of type vt

has crossed link j , we distinguish between different cases of costs calculated according to the following steps (each step is carried out if no information is available for the preceding step):

$AvgTimeIn_{s, vt}$: Average waiting time for first vehicle of vehicle type vt in front of the queue in the section

If section s has reserved lanes for vehicle class cl :

If vehicle type vt uses the reserved lane:

$TravelTimeClass_{j, cl}$: Mean Travel Time of link j aggregating all vehicle types of vehicle class cl .

$AvgTimeInClass_{s, cl}$: Average waiting time for first vehicle of vehicle class cl in front of the queue in the section s .

$AverageSectionTravelTime_{s, cl}$: Average section travel time of all vehicles of vehicle class cl .

If vehicle type vt does not use the reserved lane:

$TravelTimeClass_{j, not\ cl}$: Mean Travel Time of link j aggregating all vehicle types not belonging to vehicle class cl .

$AvgTimeInClass_{s, not\ cl}$: Average waiting time for first vehicle not belonging to vehicle class cl in front of the queue in the section s .

$AverageSectionTravelTime_{s, not\ cl}$: Average section travel time of all vehicles not belonging to vehicle class cl .

If section s doesn't have reserved lanes:

$TravelTime_j$: Mean Travel Time of link j aggregating all vehicle types.

$AvgTimeIn_s$: Average waiting time for first vehicle in front of the queue in section s .

$AverageSectionTravelTime_s$: Average section travel time for all types.

All calculated travel time have to be greater than or equal to the travel time of the link in free-flow conditions.

Finally, the Dynamic cost of link j of vehicle type vt , $DynCost_{j, vt}$, taking into account the penalty for the turning t that belongs to the link, and its capacity CL_j is calculated as

$$DynCost_{j, vt} = EstimatedTravelTime_{j, vt} + EstimatedTravelTime_{j, vt} * \varphi * (1 - CL_j / CL_{max})$$

where:

$EstimatedTravelTime_{j, vt}$ is the estimated travel time of vehicle type vt of link j calculated following previous algorithm.

φ is a user-defined capacity weight parameter that allows the user to control the influence that the link capacity has in the cost in relation with the travel time, and

CL_{max} is the theoretically estimated maximum link capacity in the network.

4.1.2.4 User-defined link cost functions

The default cost functions, described above, are basically defined in terms of link travel time and don't consider explicitly other wide variety of link costs, for example toll pricing, historical travel times representing driver's experience from previous days, combinations of various link numerical attributes as for instance travel times, delay times, length and capacity, etc. Therefore, for each particular link, the user may choose between using Initial Cost Function or Dynamic Cost Function as the default cost function, or using any other cost function defined by the user using the function editor when he wants to take into account other arguments represented by other link numerical attributes.

The user-defined cost functions can be formulated in terms of the most common mathematical functions and operators (+, -, *, /, ln, log, exp, etc.). The function terms can be defined in terms of parameters, constants and variables and must correspond to numerical attributes of any object in the model (links, sections, turnings, vehicle types, etc), whose values could be either fixed (i.e. lengths, theoretical capacities, number of lanes, etc) or change during the simulation (i.e. link flows, average link speeds, average link travel times, etc).

Two types of User-Defined cost functions can be distinguished: Basic Cost Functions (named *Cost Function*) and Cost Function considering Vehicle Type (named *Cost Function with Vehicle Type*).

Basic Cost Functions do not distinguish between vehicle types and therefore cannot make use of variables that have any vehicle type reference. The parameter for this type of function is only the link, expressed as two parameters : S (section reference of the link) and T (turning reference of the link). Cost Functions considering Vehicle Type can distinguish between vehicle types and consequently can make use of variables that have vehicle type reference. The parameter for this type of function is the link, expressed as two parameters : S (section reference of the link) and T (turning reference of the link). and the vehicle type VT reference.

If the User-Defined Cost Function considers Vehicle Type data associated to any link of the network, the calculation of shortest paths is made per vehicle type, otherwise it is common for all vehicle types.

4.1.3 Shortest Path Algorithm

During the simulation, the computation of shortest paths is predefined every prefixed time interval Δt . The shortest path routine is a variation of Dijkstra's label-setting algorithm (Dijkstra 1959) and it provides as a result the shortest path tree for each destination centroid. Therefore this structure of shortest path tree provides the shortest path from the start of every section to one destination. The penalties associated with turning movements are taken into account. Therefore, the cost labels are attached to links instead of nodes, as is usual. The link candidate list is stored as a heap data structure. During each iteration of the algorithm, the link with the minimum value of cost is removed from the heap and all links connected backward are added to the heap in the correct position.

The shortest path routine is based on link cost functions, and before apply it, the costs of all links are evaluated/updated. At the beginning of the simulation, the Initial Cost Function is evaluated per each link, and during the next time intervals, the Dynamic Cost Function, instead of Initial Cost Function, are evaluated.

The shortest path routine generates a shortest path tree for each destination centroid d ($SPT_d, d \in D$), but there is an extra step that identifies new paths for all O-D pair $i \in I$, taking $SPT_d, d \in D$, and adds to the set of alternatives path K_i of O-D pair i . From one shortest path tree, there are as many paths SP_{con} as connectors **con** has the origin centroid.

The generic schema of the Dynamic Traffic Assignment is:

Step 0: Calculate initial shortest path(s) for each O/D pair using the defined initial costs.

Step 1: Simulate for a predefined time interval (e.g. 5 minutes) assigning to the available path the fraction of the trips between each O/D pair for that time interval according to the selected route choice model and obtain new average link travel times as a result of the simulation.

Step 2: Recalculate shortest path, taking into account the experienced average link travel times.

Step 3: If there are guided vehicles, or variable message signs suggesting rerouting, provide the information calculated in 2 to the drivers that are dynamically allowed to reroute on trip.

Step 4: Go to step 1.

And the algorithm, including the details of shortest path calculation, is the following:

Step 0: Calculate initial shortest path(s) for each O/D pair using the defined initial costs

Step 0.1: Initialization:

Evaluate Initial Cost Function per each link j :

for each $j \in 1 \dots L$: $Cost_j = InitialCost_j$

Step 0.2: Apply Shortest Path routine:

for each destination centroid d :

Calculate Shortest Path Tree SPT_d using $Cost_j$ $j \in 1 \dots L$

Step 0.3: Identify Shortest Path from Shortest Path Tree:

for each O-D pair i (from origin centroid o to destination d)

Add to Path(s) SP_{con} to K_i

Step 1: Simulate for a predefined time interval Δt assigning to the available path K_i the fraction of the trips between each O/D pair i for that time interval according to the selected route choice model.

Step 2: Recalculate shortest path, taking into account the experimented average link travel times.

Step 2.1: Update Link Cost Functions:

Evaluate Dynamic Cost Function per each link j :

for each $j \in 1 \dots L$: $Cost_j = DynamicCost_j$

Step 2.2: Apply Shortest Path routine:

for each destination centroid d :

Calculate Shortest Path Tree SPT_d using $Cost_j$ $j \in 1 \dots L$

Step 2.3: Identify Shortest Path from Shortest Path Tree:

for each O-D pair i (from origin centroid o to destination d)

Add to Path(s) SP_{con} to K_i

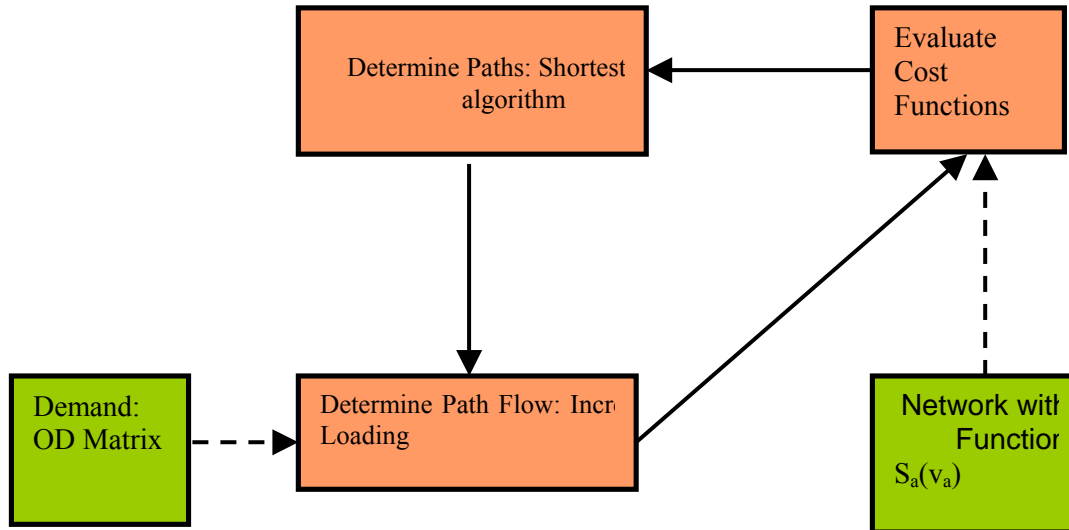
Step 3: If there are guided vehicles, or variable message signs suggesting rerouting, provide the information calculated in 2 to the drivers that are dynamically allowed to reroute on trip.

Step 4: Go to step 1.

4.1.3.1 Initial Shortest Path

At the beginning of the simulation, using the *Initial Cost* function, one shortest path tree is calculated per each destination centroid, so during the first interval all vehicles are assigned to the same alternative. In order to start considering more than one alternative, as a way to anticipate the assignment process, at the beginning of the simulation, k -shortest path tree are calculated.

Figure 4-10: Generic scheme of k-Shortest Path Algorithm



The algorithm calculates every iteration a new shortest path until the number of shortest path available reaches the parameter *InitialK-SP*. The Figure 4-10 shows the generic scheme of the algorithm where, iteratively:

- evaluates the cost function in each link (first iteration the cost function is the travel time in free-flow conditions)
- calculates a new shortest path
- determine the path flow (using an incremental loading procedure, described bellow) and update the flow in each link

The components of this algorithm are:

Shortest Path Algorithm: The computation of the shortest path corresponds a variation of Dijkstra's label setting algorithm.

Link Cost Function: The cost function of link a ($s_a(v_a)$) is function of the flow in link a v_a

$$s_a(v_a) = tt_0 \left(1 + \beta \left[\frac{v_a}{Ca} \right]^\alpha \right)$$

Incremental Loading Algorithm:

The path flow rates in the feasible region Ω satisfy the conservation flow and non-negativity constraint (where the traffic demand of O-D pair i is denoted by g_i). That is:

$$\Omega = h_k^i : \sum_{k \in K_i} h_k^i = g_i, i \in I; h_k^i \geq 0$$

For each O-D pair $i \in I$ and path h_k^i , evaluate the path flow assigned to path h_k^i $k \in K_i$ at iteration n $h_k^i(n)$:

$$h_k^i(n) = h_k^i(n-1) + \lambda(g_i - h_k^i(n-1))$$

$$h_l^i(n) = h_l^i(n-1) - \lambda(h_l^i(n-1)), \quad l = 1 \dots (k-1)$$

where $\lambda = 1/n+1$

The algorithm to calculate the k -shortest path can be stated as follows (n is the iteration index and k is the shortest path index):

Step 0: Initialization : $n=0$ and $k=1$

Compute the k -th shortest path based on the free-flow travel times.

For each O-D pair $i \in I$, assign $h_k^i(n) = g_i$

Step 1: Compute the flow v_a for each link a :

$$v_a = \sum_{i \in I} \sum_{k \in K_i} \delta_{a,k}^i h_k^i$$

where $\delta_a = \begin{cases} 1, & \text{arc } a \text{ belongs to path } k \text{ of O-D pair } i \\ 0, & \text{otherwise} \end{cases}$

Evaluate the cost function of each arc a ($s_a(v_a)$).

Step 2: $k = k+1, n = n+1$

Compute the k -th shortest path based cost function of each arc a ($s_a(v_a)$).

Incremental Loading: For each O-D pair $i \in I$, evaluate:

$$h_k^i(n) = h_k^i(n-1) + \lambda(g_i - h_k^i(n-1))$$

for $l = 1$ to $(k-1)$

$$h_l^i(n) = h_l^i(n-1) - \lambda(h_l^i(n-1)), \quad l = 1 \dots (k-1)$$

where $\lambda = 1/n+1$

Step 3: If k is equal to total number of shortest path *InitialK-SP* then STOP

Otherwise, return Step 1

4.2 PATH SELECTION

The path selection based on discrete route choice models, estimates the path flow rates. The path selection models the driver's decision of which path to take from one set of alternatives, connecting one origin to one destination. This decision process is made when a vehicle enters the system (Initial Assignment) and during its trip, when new alternatives are available (En-Route Assignment).

Given a finite set of alternative paths, the path selection calculates the probability of each available path and then the driver's decision is modelled by randomly selecting an alternative path according to the probabilities assigned to each alternative. This process corresponds to Step 1 in the generic algorithm of Dynamic Assignment and then the algorithm is:

Step 0: Calculate initial shortest path(s) for each O/D pair using the defined initial costs

Step 0.1: Initialization:

Evaluate Initial Cost Function per each link j :
for each $j \in 1 \dots L$: $Cost_j = InitialCost_j$

Step 0.2: Apply Shortest Path routine:

for each destination centroid d :

Calculate Shortest Path Tree SPT_d using $Cost_j$ $j \in 1 \dots L$

Step 0.3: Identify Shortest Path from Shortest Path Tree:

for each O-D pair i (from origin centroid o to destination d)

Add to Path(s) SP_{con} to K_i

Step 1: Simulate for a predefined time interval Δt assigning to the available path K_i the fraction of the trips between each O/D pair i for that time interval according to the selected route choice model.

Step 1.1: Assignment of path probabilities:

for each O-D pair i :

Calculate P_k using Route Choice Model, where $k \in K_i$

Step 1.2: Simulate for a predefined time interval Δt , generating the fraction of the vehicles between each O/D pair i for that time interval, selecting randomly the path according probabilities P_k , $k \in K_i$

Step 2: Recalculate shortest path, taking into account the experimented average link travel times.

Step 2.1: Update Link Cost Functions:

Evaluate Dynamic Cost Function per each link j :

for each $j \in 1 \dots L$: $Cost_j = DynamicCost_j$

Step 2.2: Apply Shortest Path routine:

for each destination centroid d :

Calculate Shortest Path Tree SPT_d using $Cost_j$ $j \in 1 \dots L$

Step 2.3: Identify Shortest Path from Shortest Path Tree:

for each O-D pair i (from origin centroid o to destination d)

Add to Path(s) SP_{con} to K_i

Step 3: If there are guided vehicles, or variable message signs suggesting rerouting, provide the information calculated in 2 to the drivers that are dynamically allowed to reroute on trip.

Step 4: Go to step 1.

The candidate paths can be of two different types (explained in section 4.1): User-defined Paths (UdP) and Calculated Shortest Paths, whose can be calculated using the Initial Cost Function *Initial Shortest Paths* (ISP) or calculated using the Dynamic Cost Function, *Dynamic shortest paths* (DSP).

A vehicle of vehicle type vt travelling from O-D pair i , can choose one path according to the user-defined assignment or as a result of a Route Choice model from the set of alternative paths K_i :

N User-defined Paths: $UdP_n^i, n=1..N$
 M Initial Shortest Paths: $ISP_m^i, m=1..M$
 P Timely Updated Shortest Paths: $DSP_p^i, p=1..P$

4.2.1 User-defined Assignment

The User-defined assignment: For each user-defined path, the user determines the probability of usage, distinguishing per vehicle type. The user defines the probability of use of user-defined paths and the probability of use of the initial shortest path:

$P(UdP_n^i, vt)$: Probability of use UdP_n^i by a vehicle type vt
 $P(ISP_m^i, vt)$: Probability of use ISP_m^i by a vehicle type vt

Satisfying the condition:

$$\sum_{n=1}^N P(UdP_n^i, vt) + \sum_{m=1}^M P(ISP_m^i, vt) \leq 1, i \in I$$

4.2.2 Route Choice Models

At any time during the simulation, there will be a finite set of alternative paths for each O-D pair. The emulation of the driver's decision of selection one of the available paths, that is to assign a trip to a path, can be done with a Route Choice model. The Route Choice models are usually inspired in the discrete choice theory that determines the probability for choosing an alternative from a finite set of alternatives as a function of its utility. From transportation point of view, the most common value associated to a trip is the travel time or travel cost, which represents a disutility. Therefore Route choice models should be formulated in terms of this negative utility. The most common concept of path cost assumes that is additive, so the cost of path i CP_i is computed as the sum of the costs of the links $Cost_j$ (explained above) composing the path:

$$CP_i = \sum_{link \ j \in Path_i} Cost_j$$

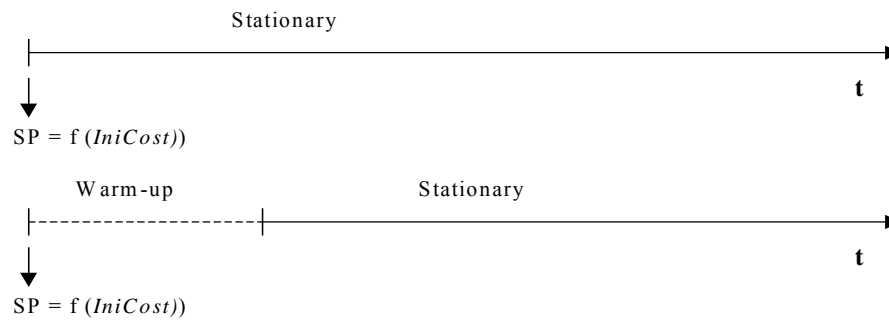
The default Route Choice models available are: Proportional, Multinomial Logit and C-Logit, but the user can also define his/her own user-defined route choice model using the function editor.

4.2.2.1 Fixed routes mode vs. Variable route mode

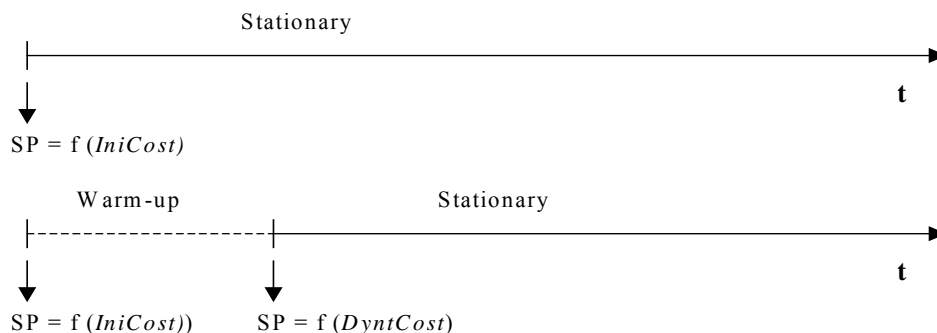
In the Fixed Routes Mode, shortest path trees are calculated from every section to every destination centroid at the beginning of the simulation. Then, during the simulation, vehicles are generated at origin centroids and assigned to the shortest route to their destination centroid. There is no need for a Route Choice Model as there are no alternative routes. No new routes are recalculated during simulation. Therefore, all vehicles always follow the shortest path and no decisions about changing to another path can be made during the trip.

Depending on the type of cost function used for the initial shortest path calculations, there are two alternative fixed route models. These are the *Fixed-Distance* and the *Fixed-Time* models.

In the *Fixed-Distance Model*, the paths are calculated at the beginning of a simulation, taking the Initial Cost as the cost of each arc, whether a warm-up period is defined or not. If there is a warm-up period, no new shortest paths are calculated when it ends, and therefore the same shortest path trees are used during the stationary simulation period the same shortest path trees are used. illustrates when the shortest paths (SP) are calculated in a time diagram of the simulation period.

Figure 4-11: Calculation of shortest paths in a fixed distance model

The *Fixed-Time Model* works similarly to the *Fixed-Distance Model*, except when a Warm-up period is defined. In this case, initial paths are calculated at the beginning of the Warm-up in the same way using the Initial Costs. When the Warm-up period is over, however, and the stationary simulation starts, new initial paths are calculated using the Cost Function (calculated using the statistical data gathered during the simulation warm-up) for arc costs. Figure 4-12 illustrates the shortest paths (SP) calculations in a time diagram of the simulation period.

Figure 4-12: Calculation of shortest paths in a fixed-time model

In the Fixed-Distance Model, the cost is very theoretical and does not take into account network congestion, only the length of the paths and the allowed speed. We call it *Fixed-Distance* to denote that the cost is mainly based on the distances, together with the speed limits and the capacity, but not on the traffic conditions at any given time. In the Fixed-Time Model the cost is influenced by the traffic conditions at certain times and therefore it represents the travel time more accurately. To denote this difference, we call it *Fixed-Time*.

In the Variable Routes Mode, the simulation process includes an initial calculation of shortest routes going from every section to every destination, a shortest route component which calculates periodically the new shortest routes according to the new travel times provided by the simulator, and a route selection model.

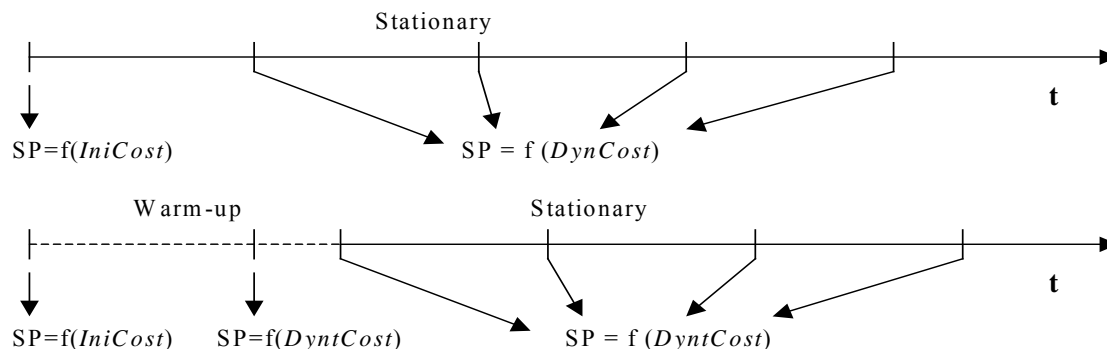
At the beginning of the simulation, shortest path trees are calculated from every section to each destination centroid, taking as arc costs the Initial Cost Function, as in the previous case. If a Warm-up period is defined, these paths are calculated at the beginning of the Warm-up. If not, they are calculated at the beginning of the stationary simulation period.

During simulation, new paths are recalculated in every time interval, taking as link costs the simulated travel times obtained for each arc during the last interval. This is the Cost Function explained above. Figure 4-13 illustrates when the shortest paths (SP) are calculated over the simulation period and what cost functions are used.

The user may define the time interval for recalculation of paths and the maximum number of path trees to be maintained during the simulation. When the maximum number of path trees (K) is reached, the

oldest paths will be removed as soon as no vehicle is following them. It is assumed that vehicles only choose between the most recent K path trees. Therefore, the oldest ones will become obsolete and unused.

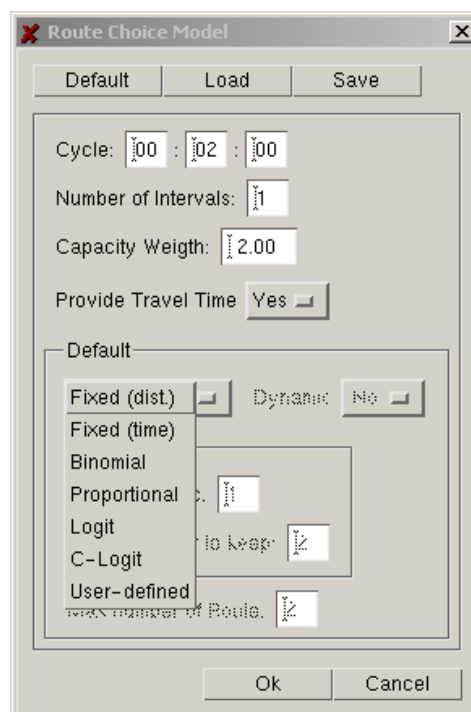
Figure 4-13: Calculation of shortest paths in a variable routes model



Currently three choice models are implemented. They are used either when assigning the initial path for a vehicle at the beginning of its trip, or when having to decide whether or not to change path en-route within dynamic modelling. These models are the Binomial, Proportional, the Multinomial Logit and the C-Logit models. On the other hand, users can also define their own Route Choice Models, via the Tedi Function Editor.

When an O/D matrix has been loaded in AIMSUN, it means that the simulation experiment is going to be Route-Based. Before running the model, the user can define which type of Route Choice Model to apply. If you select the 'Experiment / Route Choice', menu command, the 'Route Choice Model' window to appear (see Figure 4-14)

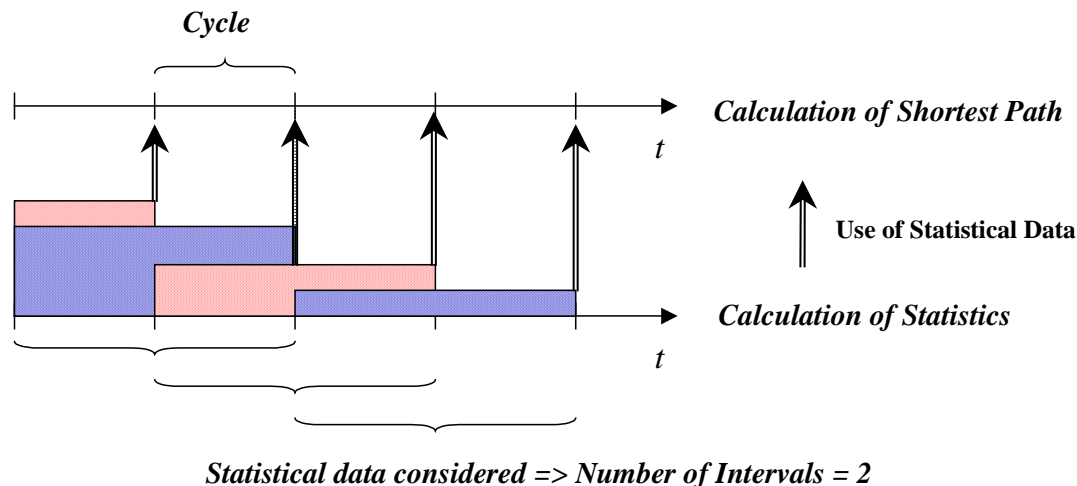
Figure 4-14: Route choice model window



In this window, the user first defines the interval between recalculations of shortest paths ('Cycle' field in hours: minutes: seconds) and the 'Number of Intervals' to be considered in each recalculation. This means that, whenever a variable model (Binomial, Proportional, Logit, C-Logit or User Defined) is selected, new paths will be calculated every 'Cycle' and the statistical information gathered during the last 'Number of Intervals' will be used for calculating the cost functions (see Figure 4-15). The user can

also define the Capacity Weight parameter to be used in calculation of the arc costs, and select among Fixed-Distance, Fixed-Time, Binomial, Logit, C-Logit or User-Defined route choice models using the option menu. Finally, it is possible to define whether to 'Provide Travel Times' apart from the costs in the 'Paths Dialog' (see section 4.3.1).

Figure 4-15: Use of Statistical data in the calculation of Shortest Paths



Dynamic: If any of the variable models (Binomial, Proportional, Logit, C-Logit or User-Defined) have been selected, the user can choose between variable route choice Static, or Dynamic (allow en-route Dynamic Assignment). In the option menu for this purpose, the user selects between Dynamic 'Yes' or 'No'

Then, depending on the selected model, some additional parameters that are related to the particular model have to be defined (see sections 4.2.2.2 to 4.2.2.6)

Initial K-SP defines the number of shortest path calculated at the beginning of the simulation. If this parameter is equal to 1, at the beginning of the simulation is calculated only one shortest path tree per destination centroid, considering the *Initial Cost Function* (see section 4.1.2.2). Otherwise it calculates the k-shortest path tree using the algorithm explained in section 4.1.3.1.

Max number to keep defines the maximum number of shortest path tree per destination centroid kept in memory during the simulation. One step of Dynamic Traffic Assignment algorithm is identify the set of alternatives path K_i for each O-D pair i and the *Max number to keep* parameter determines the number of the last shortest path tree to consider in order to generate all alternative paths of K_i . The aim of this parameter is save memory during the simulation and increase the performance. If this parameter multiplied by the *Cycle* is greater than the simulation time then all shortest path tree calculated are kept during entire simulation.

Max number to Route defines the maximum number of different paths used in the path selection process. (see section 4.2.3)

The Route Choice Model dialog window also contains the following buttons:

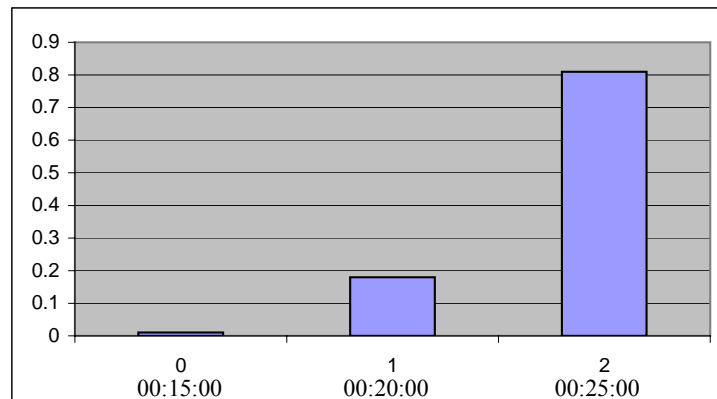
- Default: Loads the default Route Choice parameters that are common for all networks.
- Load: Loads the last saved Route Choice parameters for the current network.
- OK: Accepts the parameters displayed in the dialog window as input for the current experiment.
- Save: Saves the parameters displayed in the dialog window for the current network.
- Cancel: Closes the dialog window without accepting modifications.

4.2.2.2 Binomial Model

A Binomial ($k-1, p$) distribution is used to find the probability of selecting each path. Parameter k is the number of available paths and p is the “success” probability. This model does not consider the travel costs in the decision process, but only the time at which the path was calculated. Selecting a small p will mean that oldest paths will be more likely to be used, while selecting high values of p , will cause the more recent paths to be taken more frequently.

For example, if we want to keep three alternative paths and let the newest paths be used more often, we might define $k=3$ and $p=0.9$. Then the possible values for $X = \text{Binomial}(2, 0.9)$ are $X = 0$, $X = 1$ and $X = 2$, which are respectively associated to the last three paths calculated. Let us assume that we have defined a Cycle of recalculation of shortest paths for 5 minutes and that we are at time 25:30 in a simulation run. In this case, the last three paths calculated have been calculated at times 15:00, 20:00 and 25:00, thus correspondingly $X = 0$ to 15:00, $X = 1$ to 20:00 and $X = 2$ to 25:00. Then, the probability of selecting the oldest path is $P(X=0) = 0.01$, the probability of selecting the second path is $P(X=1) = 0.18$ and the probability of selecting the newest path is $P(X=2) = 0.81$ (see Figure 4-16).

Figure 4-16: Binomial Model ($k=3, p=0.9$)



The ‘success’ probability p can be defined via the Route Choice model dialog window when the Binomial model is chosen (see ‘Probability’ field in Figure 4-17).

4.2.2.3 Proportional

The choice probability P_k of a given alternative path k , where $k \in K_i$, can be expressed as:

$$P_k = \frac{CP_k^{-\alpha}}{\sum_{l \in K_i} CP_l^{-\alpha}}$$

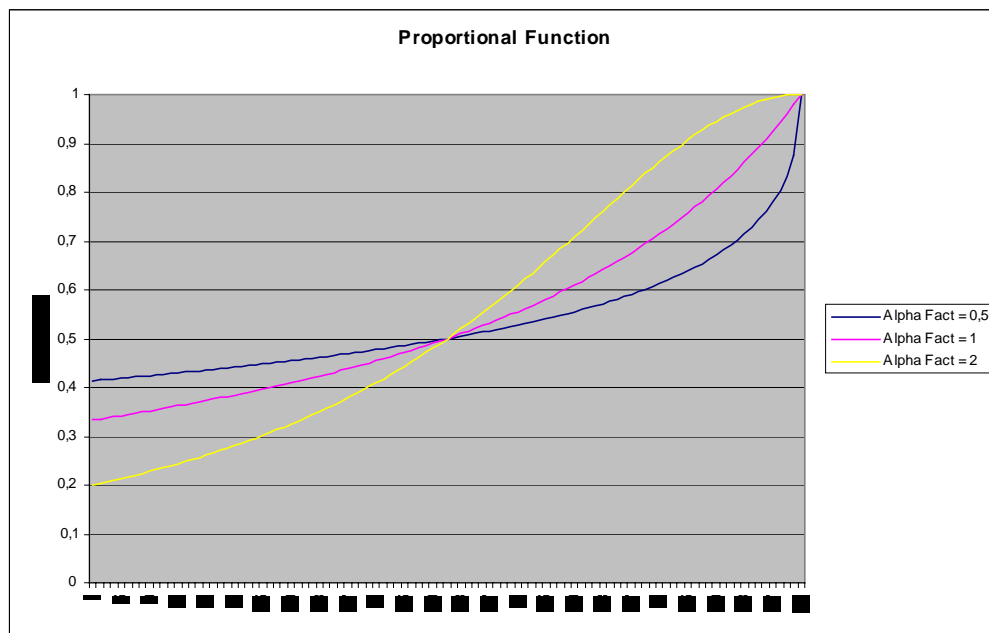
where CP_i , the cost of path i .

When $\alpha=1$, then the probability is inversely proportional to path costs. The example in Table 4-1 illustrates how the Alpha Factor (α) influences the probability of choosing a path in the case of two alternative paths with travel times 5 minutes and 4 minutes respectively.

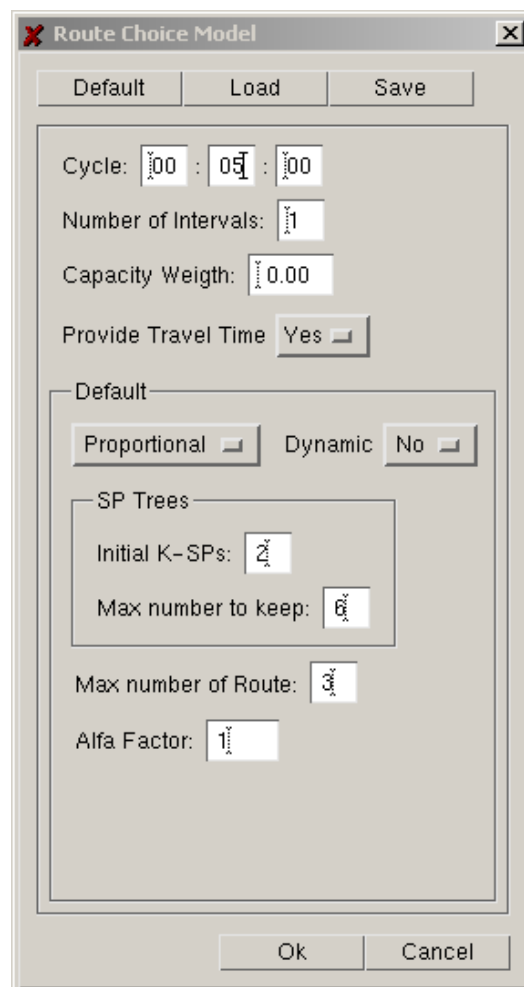
Figure 4-17: Route choice model window. Binomial Model.
Table 4.1: Alpha factor in Proportional Route Choice Model

Path Costs	Path 1	Path 2	
	300	240	seconds
	0,08333333	0,06666667	hours
Alpha	P(1)	P(2)	
0	0,5	0,5	
0,2	0,4888447	0,5111553	
0,4	0,4777004	0,5222996	
0,6	0,4665784	0,5334216	
0,8	0,4554894	0,5445106	
1	0,4444444	0,5555556	
1,2	0,4334541	0,5665459	
1,4	0,4225288	0,5774712	
1,6	0,4116788	0,5883212	
1,8	0,4009140	0,5990860	
2	0,3902439	0,6097561	
2,2	0,3796778	0,6203222	
2,4	0,3692246	0,6307754	
2,6	0,3588927	0,6411073	
2,8	0,3486901	0,6513099	
3	0,3386243	0,6613757	
3,2	0,3287025	0,6712975	
3,4	0,3189312	0,6810688	
3,6	0,3093165	0,6906835	
3,8	0,2998640	0,7001360	
4	0,2905789	0,7094211	

The Figure 4-18 depicts the role of the alpha factor, as a function of the different path costs. The parameter, or alpha factor, α , is a user-defined parameter that can consequently be used to adjust the effect that small changes in the travel times may have on the driver's decisions

Figure 4-18: Proportional function shape

The alpha factor parameter, α can be defined via the Route Choice model dialog window when the Proportional model is selected (see Figure 4-19).

Figure 4-19: Route choice model window. Proportional Model.

4.2.2.4 Multinomial Logit

The choice probability P_k of a given alternative path k , where $k \in K_i$, can be expressed as a function of the difference between the measured utilities of that path and all other alternative paths:

$$P_k = \frac{e^{v_k \theta}}{\sum_{l \in K_i} e^{v_l \theta}}$$

or its equivalent expression:

$$P_k = \frac{1}{1 + \sum_{l \neq k} e^{(v_l - v_k) \theta}}$$

where V_i is the perceived utility for alternative path i and θ is a shape or scale factor. We have taken $V_i = -CP_i/3600$. (function V_i is minus the cost of path i , measured in hours).

We assume that the utility U_k^i of path k between O-D pair i is given by:

$$U_k^i = -\theta t_k^i + \varepsilon_k^i$$

Where:

θ is a shape or scale factor parameter

t_k^i is the expected travel time on path k of O-D pair i , and

ε_k^i is a random term

The underlying modelling hypothesis is that random terms ε_k^i are independent identically distributed GUMBEL variates¹. Under these conditions, the probability of choosing path k amongst all alternative routes of O-D pair i is given by the logistic distribution:

$$P_k^i = \frac{e^{-\theta t_k^i}}{\sum_l e^{-\theta t_l^i}} = \frac{1}{1 + \sum_{l \neq k} e^{-\theta(t_l^i - t_k^i)}} \quad (1)$$

The scale factor θ plays a two-fold role, making the decision based on differences between utilities independent of measurement units, and influencing the standard error of the distribution of expected travel times:

$$Var(t_k^i) = \frac{\pi^2}{6\theta^2}$$

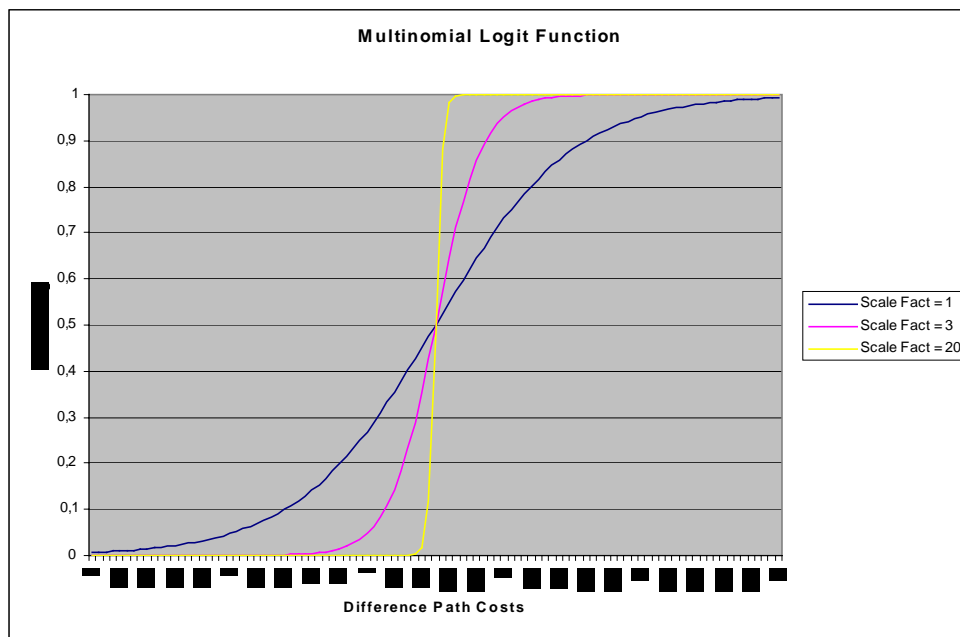
that is:

$\theta < 1$ high perception of the variance, in other words a trend towards utilizing many alternative routes

$\theta > 1$ alternative choices are concentrated in very few routes

For example, given four alternative path with expected travel time (cost path) of $T_1=12$ minutes, $T_2=15$ minutes, $T_3=16$ minutes and $T_4=18$ minutes, the corresponding probabilities according to equation (1) when $\theta = 1$ are: $P_1=0.93407$, $P_2=0.04650$, $P_3=0.01710$ and $P_4=0.00231$, whereas if $\theta = 0.5$ the probabilities are: $P_1=0.71009$, $P_2=0.15844$, $P_3=0.09610$ and $P_4=0.03535$. The Figure 4- depicts the role of the scale factor, as a function of the difference between path costs:

¹ A random variable ε is GUMBEL distributed if $F(\varepsilon) = \exp[-e^{-\mu(\varepsilon-\eta)}]$, with $\mu > 0$ and η a location parameter. If $\varepsilon_i, \varepsilon_j$ are independent GUMBEL variates with parameters (η_1, μ) and (η_2, μ) respectively, then $\varepsilon^* = \varepsilon_i - \varepsilon_j$ is logistically distributed, i.e. $F(\varepsilon) = [1 + e^{\mu(\eta_2 - \eta_1 - \varepsilon^*)}]^{-1}$ and if they are identically distributed ($\eta_1 = \eta_2$) then $F(\varepsilon) = [1 + e^{-\mu\varepsilon^*}]^{-1}$

Figure 4-20: Logit function shape

The parameter, or scale factor, θ , is a user-defined parameter that can consequently be used to adjust the effect that small changes in the travel times may have on the driver's decisions. The example in Table 4-2 illustrates how the Scale Factor (θ) influences the probability of choosing a path if there are two alternative paths with travel times 5 minutes and 4 minutes respectively.

Table 4.2: Multinomial Logit Example

Path Cost	Path 1	Path 2	
	300	240	Seconds
	5	4	Minutes
	0.083333	0.066667	Hours
Scale Factor	P(1)	P(2)	
1	0.495833	0.504167	
10	0.458430	0.541570	
20	0.417430	0.582570	
30	0.377541	0.622459	
40	0.339244	0.660756	
50	0.302941	0.697059	
60	0.268941	0.731059	
100	0.158869	0.841131	
500	0.000240	0.999760	
1000	5.7777E-08	1.0000E+00	
2000	3.3382E-15	1.0000E+00	
3600	8.7565E-27	1.0000E+00	

The scale factor parameter, θ can be defined via the Route Choice model dialog window when the Logit model is selected (see Figure 4-21).

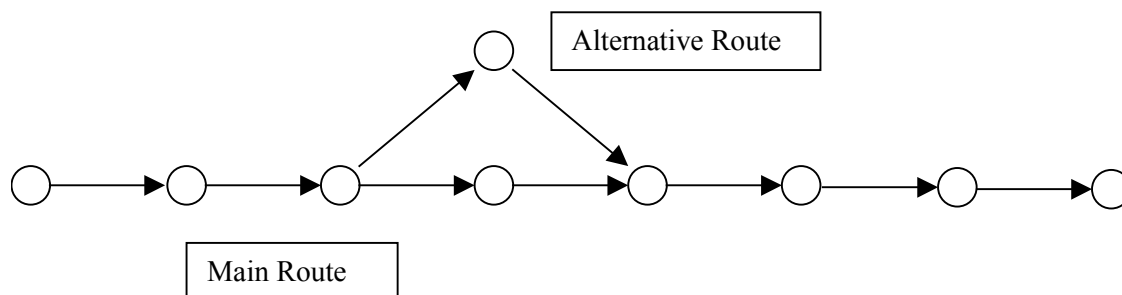
Figure 4-21: Route choice model window. Logit Model.

The screenshot shows a window titled "Route Choice Model" with a standard Windows interface. At the top are three buttons: "Default", "Load", and "Save". Below these are several input fields: "Cycle:" with a range of 00 to 00, "Number of Intervals:" set to 1, "Capacity Weigh:" set to 0.00, and "Provide Travel Time" with a "Yes" button. A "Default" section contains three radio buttons: "Logit" (selected), "Dynamic", and "No". Below this is an "SP Trees" section with "Initial K-SPs:" set to 1 and "Max number to keep:" set to 10. Further down are "Max number of Route:" set to 10 and "Scale Factor:" set to 10.0. At the bottom are "Ok" and "Cancel" buttons.

4.2.2.5 C Logit

The Logit models exhibit a tendency towards route oscillations in the routes used, with the corresponding instability generating a kind of flip-flop process. According to our experience there are two main reasons for this behaviour. The properties of the Logit function, and the inability of the Logit function to distinguish between two alternative routes when there is a high degree of overlapping. Some researchers Cascetta 1996 have recently reported these drawbacks.

The instability of the routes used can be substantially improved when the network topology allows for alternative paths with little or no overlapping at all, changing the shape factor θ and re-computing the path very frequently. However, in large networks where many alternative paths between origin and destinations exist and some of them exhibit a certain degree of overlapping (see Figure 4-22), the use of the Logit function may still exhibit some weaknesses.

Figure 4-22: Overlapping Paths

To avoid this drawback the C-Logit model Cascetta 1996 has been implemented. In the C-Logit model, which is, in fact, a variation of the Logit model, the choice probability P_k , of each alternative path k belonging to the set $\in K_i$ of available paths connecting an O/D pair, is expressed as:

$$P_k = \frac{e^{\theta(V_k - CF_k)}}{\sum_{l \in K_i} e^{\theta(V_l - CF_l)}}$$

where V_i is the perceived utility for alternative path i . and θ is the scale factor, as in the case of the Logit model.

The term CF_k , denoted as ‘commonality factor’ of path k , is directly proportional to the degree of overlapping of path k with other alternative paths. Thus, highly overlapped paths have a larger CF factor and therefore smaller utility with respect to similar paths. CF_k is calculated as follows:

$$CF_k = \beta \cdot \ln \sum_{l \in K_j} \left(\frac{L_{lk}}{L_l^{1/2} L_k^{1/2}} \right)^\gamma$$

where L_{lk} is the length of links common to paths l and k , while L_l and L_k are the length of paths l and k respectively. Depending on the two factor parameters β and γ , a greater or lesser weighting is given to the ‘commonality factor’. Larger values of β means that the overlapping factor has greater importance with respect to the utility V_i ; γ is a positive parameter, usually taken in the range $[0, 2]$, whose influence is smaller than β and which has the opposite effect.

As a rule of thumb, one can suggest to take factor β in the range $[t_{\min}, t_{\max}]$, being $t_{\min} = \text{Min}_{k \in K_i} [CP_k]$ and $t_{\max} = \text{Max}_{k \in K_i} [CP_k]$. Then β will become a kind of scaling factor for CF_k , that translates it into an order of magnitude similar to V_k in the formula $V_k - C_k$ used for the exponential. And thus, when using larger values for β , it is possible that the ‘commonality factor’, CF_k , will have a greater influence on the choice probability than the utility (i.e. the travel time) itself, thus giving higher probability of choosing non-overlapped longer paths than heavy overlapped shortest paths.

The following is an example to illustrate the use of C-Logit, comparing it with the Multinomial Logit. Figure 4-23 shows an example network with four alternative paths between origin O and destination D. Table 4-3 represents the resulting choice probabilities of both models.

Figure 4-23: Example of network with overlapped paths

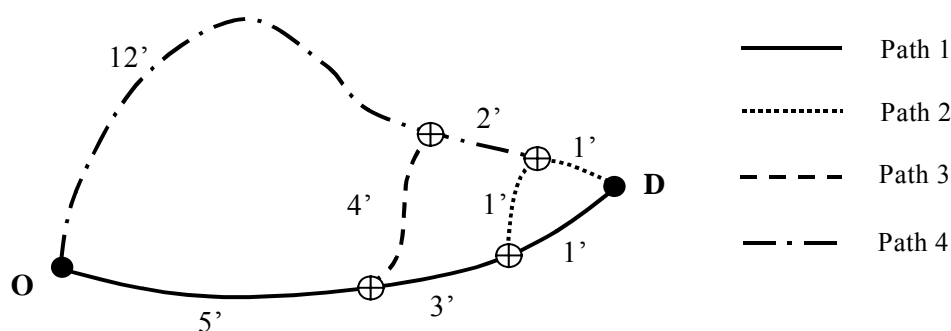


Table 4.3: Comparison between Logit and C-Logit

Travel Times	Path 1	Path 2	Path 3	Path 4	
	540	600	720	900	seconds
	9	10	12	15	minutes
	0.15	0.166666667	0.2	0.25	hours
Overlapping (lk)					
1 / k	1	2	3	4	
1	0.1500	0.1333	0.0833	0.0000	
2	0.1333	0.1667	0.1000	0.0167	
3	0.0833	0.1000	0.2000	0.0500	
4	0.0000	0.0167	0.0500	0.2500	
CFk	0.007029	0.007544	0.006767	0.002220	Beta 0.15 Gamma 1
Example of LOGIT					
Scale Factor	P(1)	P(2)	P(3)	P(4)	
1	0.260448	0.256143	0.247746	0.235663	
10	0.354498	0.300076	0.215014	0.130412	
20	0.450502	0.322799	0.165730	0.060969	
30	0.532071	0.322717	0.118721	0.026490	
40	0.599856	0.307976	0.081182	0.010987	
50	0.656417	0.285278	0.053882	0.004423	
60	0.704153	0.259044	0.035058	0.001745	
100	0.836359	0.157968	0.005635	0.000038	
500	0.999760	0.000240	1.3885E-11	1.9283E-22	
1000	1.0000E+00	5.7777E-08	1.9287E-22	3.7201E-44	
2000	1.0000E+00	3.3382E-15	3.7201E-44	1.3839E-87	
3600	1.0000E+00	8.7565E-27	6.7142E-79	4.5080E-157	
Example of C-LOGIT					
Scale Factor	P(1)	P(2)	P(3)	P(4)	
1	0.253734	0.247897	0.246238	0.252131	
10	0.288035	0.228222	0.213405	0.270338	
20	0.327050	0.205324	0.179528	0.288098	
30	0.366177	0.182150	0.148925	0.302747	
40	0.404621	0.159478	0.121923	0.313978	
50	0.441725	0.137948	0.098616	0.321711	
60	0.477006	0.118032	0.078900	0.326062	
100	0.596019	0.058128	0.029706	0.316146	
500	0.959694	0.000008	0.000000	0.040297	
1000	0.998240	0.000000	0.000000	0.001760	
2000	0.999997	0.000000	0.000000	0.000003	
3600	1.000000	0.000000	0.000000	0.000000	

The scale factor parameter, θ , and the commonality factor parameters, β and γ , can be defined via the Route Choice model dialog window when the C-Logit model is selected (see Figure 4-24)

Figure 4-24: Route choice model window. C-Logit Model.

Route Choice Model

Default Load Save

Cycle: 00 : 10 : 00

Number of Intervals: 1

Capacity Weight: 1.50

Provide Travel Time Yes

Default

C-Logit Dynamic No

SP Trees

Initial K-SPs: 1

Max number to keep: 50

Max number of Route: 50

Scale Factor: 10

Beta Factor: 0.15

Gamma Factor: 1

Ok Cancel

4.2.2.6 User-defined Route Choice Model

As an alternative to the default Route Choice Models, the user can define his/her own Route Choice Models. This is done using the Function Editor.

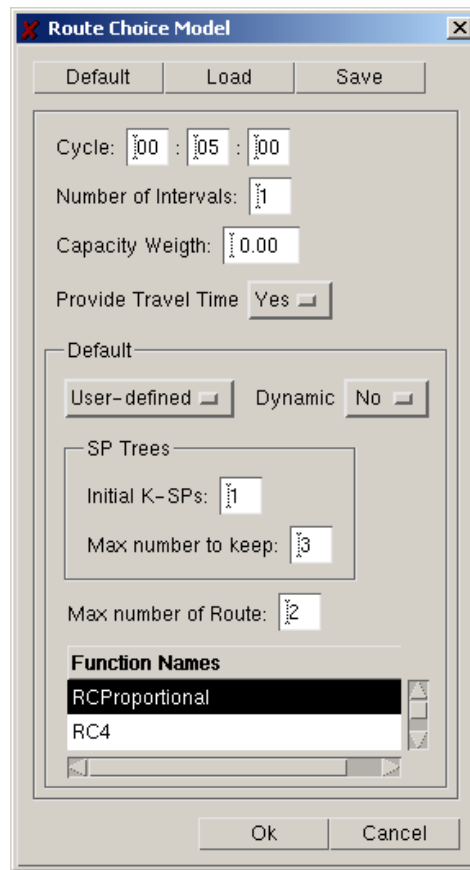
The user can select to use one of the User-Defined Route Choice Models via the Route Choice model dialog window (see Figure 4-25). This dialog window displays a list of User-Defined Route Choice Functions available is displayed. The user can select one of them by clicking on the list box.

4.2.3 Determine the finite set of path in the decision process

The path selection based on discrete route choice models, estimates the path flow rates of a finite set of alternative paths K_i , connecting the O-D pair i -th and this set of alternative paths is updated considering the shortest path trees calculated every interval of the Dynamic Traffic Assignment.

At time interval t , the shortest path algorithm is applied and it generates the shortest path tree $SPT_d(t)$ per each destination centroid $d \in D$. Then per each O-D pair $i \in I$ (with origin centroid o and destination centroid d) and taking $SPT_d(t)$ it generates as many paths $SP_{con}(t)$ as connectors **con** has the origin centroid.

The process to determine the finite set of paths to be used in the decision process receives as parameter the *MaxNumberSPT* (which represents the last number of shortest path tree to consider) and *MaxNumberPaths* (which represents the maximum number of different paths).

Figure 4-25: Route choice model window. User-Defined Model.

The algorithm is the following:

```

for each O-D pair  $i$  (from origin centroid  $o$  to destination  $d$ ) :
     $K_i = \emptyset$ 
    for  $con \in$  Connectors of origin centroid  $o$  :
         $K_i(con) = \emptyset$ 
        for  $j \in 0 \dots (MaxNumberSPT-1)$ :
            if  $SP_{con}(t-j) \notin K_i$ 
                Add  $SP_{con}(t-j)$  to  $K_i$ 
            endif
        endfor
        Order  $K_i(con)$  by cost ascendant: cost of
        for  $m \in 1 \dots MaxNumberPaths$  :
            Add element  $m$  of  $K_i(con)$  to  $K_i$ 
        endfor
    endfor
endfor

```

4.2.4 Initial Assignment

The Initial Assignment is the path selection process when a vehicle enters the system. And it corresponds to Step 1 in the generic algorithm of Dynamic Traffic Assignment.

A vehicle of vehicle type vt traveling from O-D pair i , can choose one path according to the user-defined assignment or as a result of a Route Choice model from the set of alternative paths K_i :

N User-defined Paths: $UdP_n^i, n=1..N$
 M Initial Shortest Paths: $ISP_m^i, m=1..M$
 P Timely Updated Shortest Paths: $DSP_p^i, p=1..P$

The Initial assignment algorithm has two parts depending on whether the vehicle v is guided or not:

when vehicle v is guided (an attribute of the vehicle) then assigns directly to the all alternatives paths k , $k \in K_i$, the probability calculated using the Route Choice Model, and

when vehicle v is non-guided, if the sum of probabilities $PUDa = \sum_{n=1}^N P(UdP_n^i, vt) + \sum_{m=1}^M P(ISP_m^i, vt)$

is equal to 1.0 then it assigns to the user-defined paths the probabilities defined with user-defined assignment. If this sum is less than 1.0, then $PUDa$ will represent the probability that the vehicle will uses the user-defined assignment and the rest $(1.0 - PUDa)$ will apply the Route Choice Model, assigning to the all alternatives paths k , $k \in K_i$, the probability calculated using the Route Choice Model.

Then the vehicle is assigned to one path randomly according to all path probabilities assigned.

The following algorithm defines the Initial assignment path of a vehicle v , which belongs to vehicle type vt , and which belongs to O-D Pair i (from origin O_i to destination D_j). The algorithm takes into account when a vehicle has a guided behaviour or not guided. A vehicle has a guided behaviour when is guided (the percentage of guided vehicles is a vehicle type parameter) and follows the recommendations using the Guidance Acceptance as a probability of use (proportion defined as vehicle type parameter).

if vehicle v has a guided behaviour then

Apply Route Choice Model:

Generate Random number X inside of interval defined between $[0..1]$

Accumulated = 0

for each k , $k \in K_i$:

if $X \leq \text{Accumulated} + P_k$ then

Assign path k to vehicle v and **STOP**

else

Accumulated = Accumulated + P_k

endif

else {the vehicle is not guided behaviour}

Generate Random number X inside of interval defined between $[0..1]$

if $\sum_{n=1}^N P(UdP_n^i, vt) + \sum_{m=1}^M P(ISP_m^i, vt) \leq X$ then

Apply User-defined Assignment:

Accumulated = 0

for each $UdP_n^i, n \in 1..N$:

if $X \leq \text{Accumulated} + P(UdP_n^i, vt)$ then

Assign path UdP_n^i to vehicle v and **STOP**

else

Accumulated = Accumulated + $P(UdP_n^i, vt)$

endif

for each $ISP_m^i, m \in 1..M$:

if $X \leq \text{Accumulated} + P(ISP_m^i, vt)$ then

Assign path ISP_m^i to vehicle v and **STOP**

else

Accumulated = Accumulated + $P(ISP_m^i, vt)$

endif

```

else
  Apply Route Choice Model:
    Generate Random number  $X$  defined between  $[0...1]$ 
    Accumulated = 0
    for each  $k, k \in K_i$ :
      if  $X \leq \text{Accumulated} + P_k$  then
        Assign path  $k$  to vehicle  $v$  and STOP
      else
        Accumulated = Accumulated +  $P_k$ 
      endif
    endif
  endif
endif

```

4.2.5 En-Route Assignment

Vehicles are initially assigned to a path from a set of available paths based on probability. Apart from the Initial assignment, which is made at the time of the vehicle's departure, there is the possibility of making a path reassignment during the trip (En-Route Assignment).

The En-Route Assignment, corresponds to Step 3 in the generic algorithm of Dynamic Traffic Assignment.

A guided vehicle can make a new decision about which path to follow at any time in a trip, whenever there are new shortest paths available. The algorithm is defined as:

```

for all vehicle  $v$  guided then
  Generate Random number  $X$  defined between  $[0...1]$ 
  Accumulated = 0
  for each  $k, k \in K_i$ :
    if  $X \leq \text{Accumulated} + P_k$  then
      Assign path  $k$  to vehicle  $v$  and STOP
    else
      Accumulated = Accumulated +  $P_k$ 
    endif
  endif

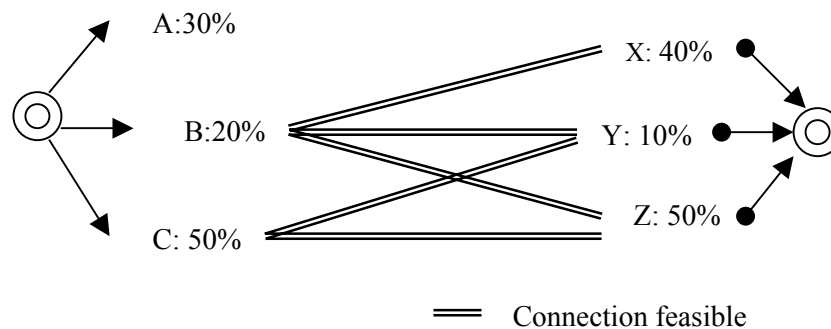
```

4.2.6 Entrance/Exit Section decision

When the route choice model is applied to a vehicle on the Initial assignment, the first step is to decide the entrance section at which the vehicle will enter the system and the exit section at which it will exit. This decision is made considering the use of percentages on the origin and destination centroid (attribute of centroid). Considering this we have 4 different situations:

Origin Centroid considers Percentages / Destination Centroid considers Percentages

To demonstrate the decision process, we will use the example in Figure 4-26, where the origin centroid has 3 connections (A, B and C) with 30%, 20% and 50% respectively and the destination centroid has 3 exit connections (X, Y and Z) with 40%, 10% and 50%. The only feasible combinations that the connectivity of the network allows are that from connection B, it is possible to reach connection X, Y and Z, while from connection C, it is possible to reach connection Y and Z. Other possible combinations are unfeasible.

Figure 4-26: Example of Origin/Destination Considers Percentages

The first step is to decide the entrance section. This process takes into account only the feasible connections available to reach the destination and recalculates the percentages considering the original percentage over the sum of percentages of all feasible connections. In our example, the percentages of A is 0% (not feasible), B is 20%/70% (70% is the sum of 20% and 50%) and C is 50%/70%. Considering these calculated percentages, the vehicle chooses the connector randomly (i.e., the entrance section).

After the entrance section has been determined, it is necessary to determine the exit section. This is done by considering only the feasible connections from the determined entrance section and recalculating the percentages as the original percentage over the sum of percentages of all feasible connections. In our example, if the entrance section chosen is using connection C, then the calculated probability of connection X will be 0% (not feasible from connection C), Y will be 10%/60% (60% is the sum of 10% and 50%) and Z will be 50%/60%. The vehicle then chooses the exit section randomly.

Origin Centroid considers Percentages / Destination Centroid does not consider Percentages

In this case the entrance section is selected as in case a) and the exit section depends on the shortest path.

Origin Centroid does not consider Percentages / Destination Centroid considers Percentages

In this case the exit section is selected as in case a) considering only the feasible connectors from the origin. The entrance section is selected by considering the shortest path.

Origin Centroid does not consider Percentages / Destination Centroid does not consider Percentages

The entrance section and the exit section are selected by considering the shortest path.

4.3 PATH ANALYSIS TOOL

4.3.1 Path Analysis

To get the insight on what is happening in a heuristic dynamic assignment for the proper calibration and validation of the simulation model the user should have access to the analysis of the used routes. The path information available is:

- Shortest Path Information: the user can view all shortest path information that are being used by vehicles during simulation
- User-defined Path Information: the user can view all user-defined path information.
- Shortest Path Display: the user can view simultaneously different shortest path, displaying their links on the network
- Initial Path Assignment: the user can view all probabilities considered when a vehicle enters the system.
- Dynamic Path Assignment: the user can view all probabilities considered when a vehicle makes a path reassignment during the trip.

4.3.1.1 Shortest Path Information

Paths are always accessed via a destination centroid. Consequently, a destination centroid has to be selected first. This can be done either by typing the destination centroid identifier in the 'Dest. Centroid' dialog window, by using the Browse button, or by clicking on the centroid icon directly in the network. When the destination centroid has more than one "from" connection and it uses destination percentages, a list of these connections are displayed and one has to be selected. Once the destination centroid and the connection have been selected, a list of intervals will be displayed in the 'Interval' list box and a list of vehicle types will be displayed in the 'Vehicle Type' list box. Each time interval corresponds to a tree of paths. For each one, the total number of vehicles that are currently following this tree is shown in the 'Total Veh.' column.

Select one path tree by clicking on a time interval in the 'Interval' list box, and the number of vehicles per type following the path is shown in the 'Vehicle Type' list box. Then, as different path trees are calculated for each vehicle type, select one of them by clicking on the 'Vehicle Types' list box. Now we have fully identified the path tree by the destination centroid, the time interval, and the vehicle type.

The next step is to select an origin section to obtain the particular path from that section to the destination centroid. This can be done by typing the identifier in the 'Origin Section' dialog window, using the browse button, or by clicking directly on the section image in the network. The shortest path from that section to the destination centroid will appear in the list box at the bottom of the window and will also be highlighted on the screen.

The path list box contains the list of section identifiers composing the path and the following information is displayed for each section:

The cost in time (seconds) from each of the sections in the path to the destination centroid. This can be calculated as either the sum of $\text{IniCost}(a)$ or $\text{Cost}(a, vt)$ of all the arcs composing the path.

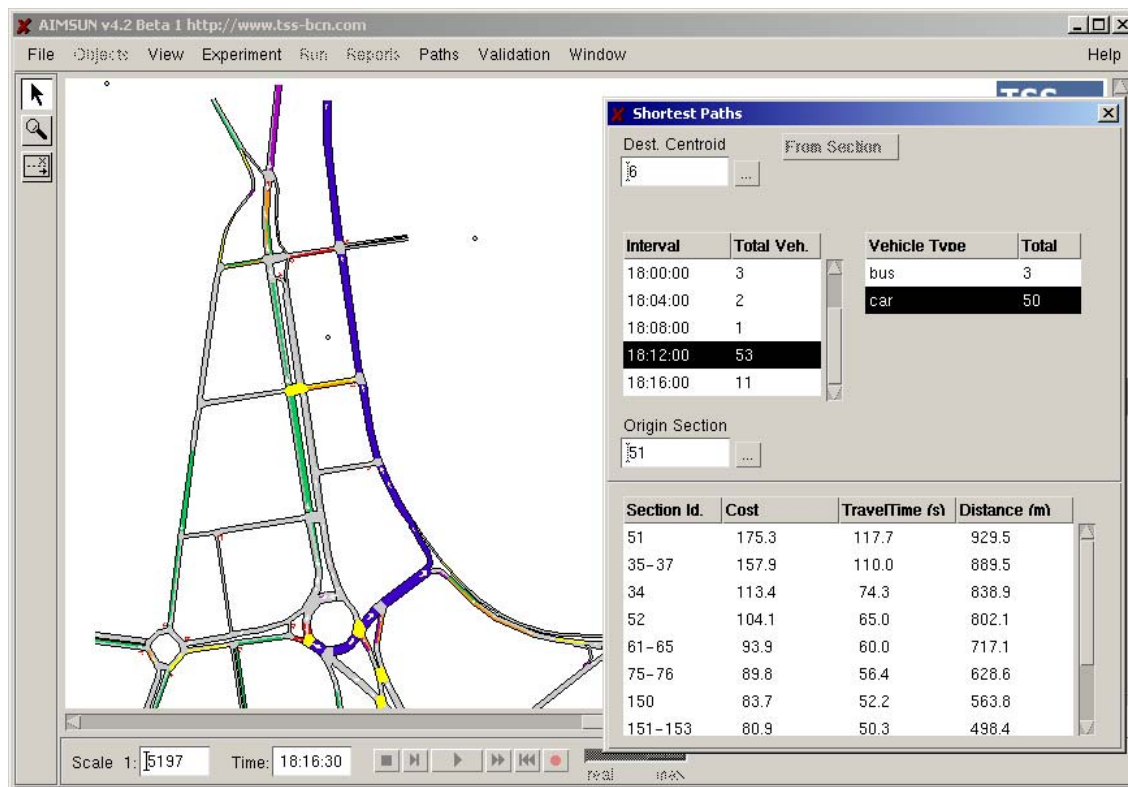
The travel time in seconds from each of the sections in the path to the destination centroid. This is equal to the cost only if the capacity weight parameter is set to zero.

The distance (metres) from each of the sections in the path to the destination centroid.

In the example shown in Figure 4-27, 53 vehicles are currently using the tree of paths calculated at time 08:12:00 (3 buses and 50 cars), 11 vehicles are using the one calculated at time 08:16:00, and so on. In the example, the shortest path from section 51 to centroid 1 goes through sections 51, 35 - 37, 38 - 40,

etc. (displayed with a different colour on the network). The cost of the whole path is 173.3 seconds, the travel time is 117.7 seconds and the distance is 929.5 meters.

Figure 4-27: Shortest Path Information



4.3.1.2 User-defined Path Information

User-defined paths are always accessed by origin and destination centroid. Consequently, an origin and a destination centroid have to be selected first. This can be done either by typing the centroid identifier and Enter key in the 'Origin' or 'Destination' dialog window, by using the Browse button, or by writing on the identifier directly. Once the origin and destination centroids have been selected, a list of user-defined paths with identifier and name will be displayed in 'Path Id/Name' list box and a list of vehicle types will be displayed in the 'Vehicle Type' list box.

The user-defined path will appear in the list box at the bottom of the window and will also be highlighted on the screen.

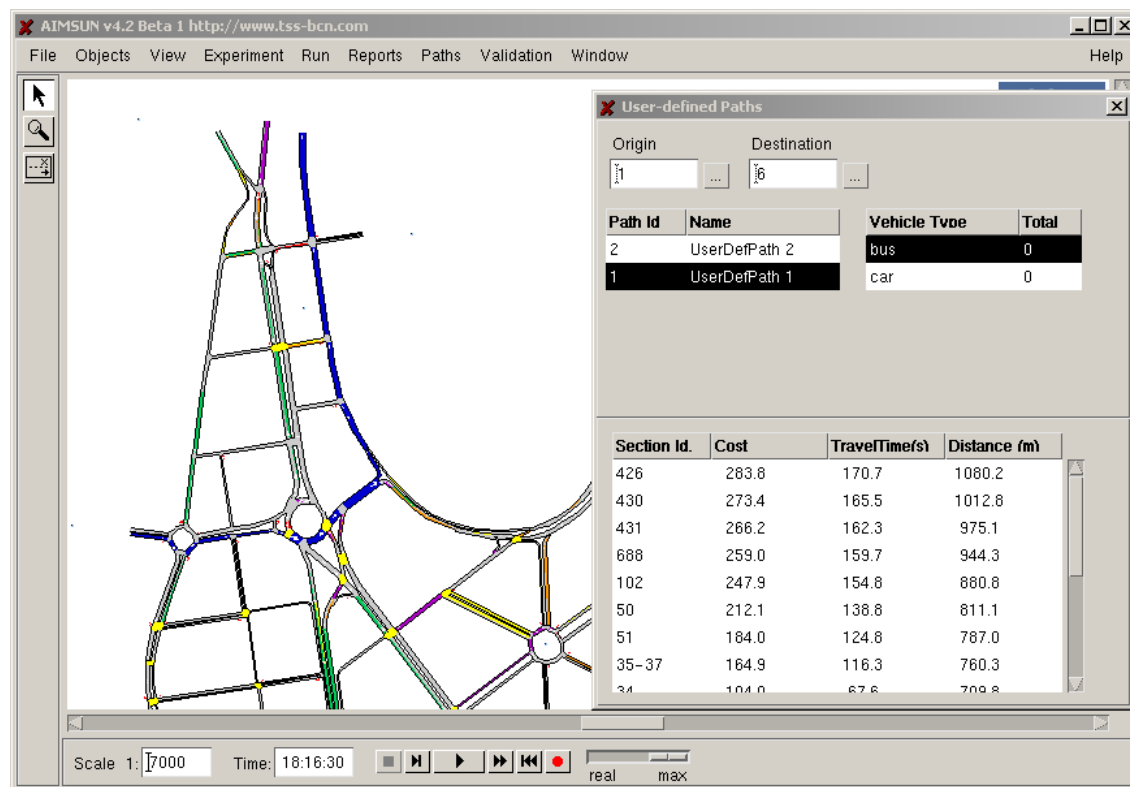
The user-defined path list box contains the list of section identifiers composing the path and the following information is displayed for each section:

The cost in time (seconds) from each of the sections in the path to the destination centroid. This can be calculated as either the sum of $IniCost(a)$ or $Cost(a, vt)$ of all the arcs composing the path.

The travel time in seconds from each of the sections in the path to the destination centroid. This is equal to the cost only if the capacity weight parameter is set to zero.

The distance (metres) from each of the sections in the path to the destination centroid.

In the example shown in Figure 4-28, from Origin 1 to Destination 6, there are two user-defined paths: *UserDefPath1* and *UserDefPath2*. The user-defined path *UserDefPath1* has a total cost of 283.8 seconds, a travel time of 170.7 seconds and a distance of 1080.2 meters.

Figure 4-28: User-defined Path Information

4.3.1.3 Shortest Path Display

The Shortest Path Display allows display, simultaneously, different shortest path calculated at different time intervals. This output allows to know the evolution of the shortest paths calculated in different time intervals. The paths displayed could be displayed defining:

- One destination centroid from one origin section.
- One destination centroid from one origin centroid.
- One destination centroid from all origin centroid.

Paths are always accessed via a destination centroid. Consequently, a destination centroid has to be selected first. This can be done either by typing the destination centroid identifier in the 'Dest. Centroid' dialog window, by using the Browse button, or by clicking on the centroid icon directly in the network. When the destination centroid has more than one "from" connection and it uses destination percentages, a list of these connections are displayed and one has to be selected. Once the destination centroid and the connection have been selected, a list of intervals will be displayed in the 'Interval' list. Each time interval corresponds to a tree of paths.

Select one path tree (or more than one) by clicking on a time interval in the 'Interval' list box, and an arrow identifies the path tree as selected to be displayed. A path tree could be deselected clicking on and then the arrow disappears. Then, as different path trees are calculated for each vehicle type, select one of them by clicking on the 'Vehicle Types' list box. Now we have fully identified the path tree by the destination centroid, the time interval, and the vehicle type.

The next step is to select either select *Origin Section* to obtain the particular path from that section to the destination centroid (this can be done by typing the identifier in the 'Origin Section' dialog window, using the browse button, or by clicking directly on the section image in the network) or select *Origin Centroid* to obtain the particular path from that origin to the destination centroid (this can be done by typing the identifier in the 'Origin Centroid' dialog window, using the browse button and when the

origin has more than one “from” connection and it uses origin percentages, a list of these connections are displayed and one has to be selected) or select From All origins to obtain the path tree from all origins to the destination centroid.

Finally press the button “Display” and all paths selected will be highlighted on the screen with a different colour assigned to each path. The colour assigned to each path is displayed in the ‘Interval’ list box. When a section belongs to different paths, that is there is overlapping among paths, then the section is highlighted using black colour.

Figure 4-29 shows the shortest path calculated at time 18:16:00, from all origin centroids to destination centroid 6, where the all links are displayed in red (or the color displayed in the window dialog), except the links shared by different paths, which are displayed using black colour. Figure 4-30 shows, displaying all links on red, the shortest path from origin centroid 22 to destination centroid 1 calculated at 4 minutes after the beginning of the simulation during the warm-up period and the shortest path calculated at 18h16m with all links displayed on green.

Figure 4-30: Shortest Path Display from all origins to one destination

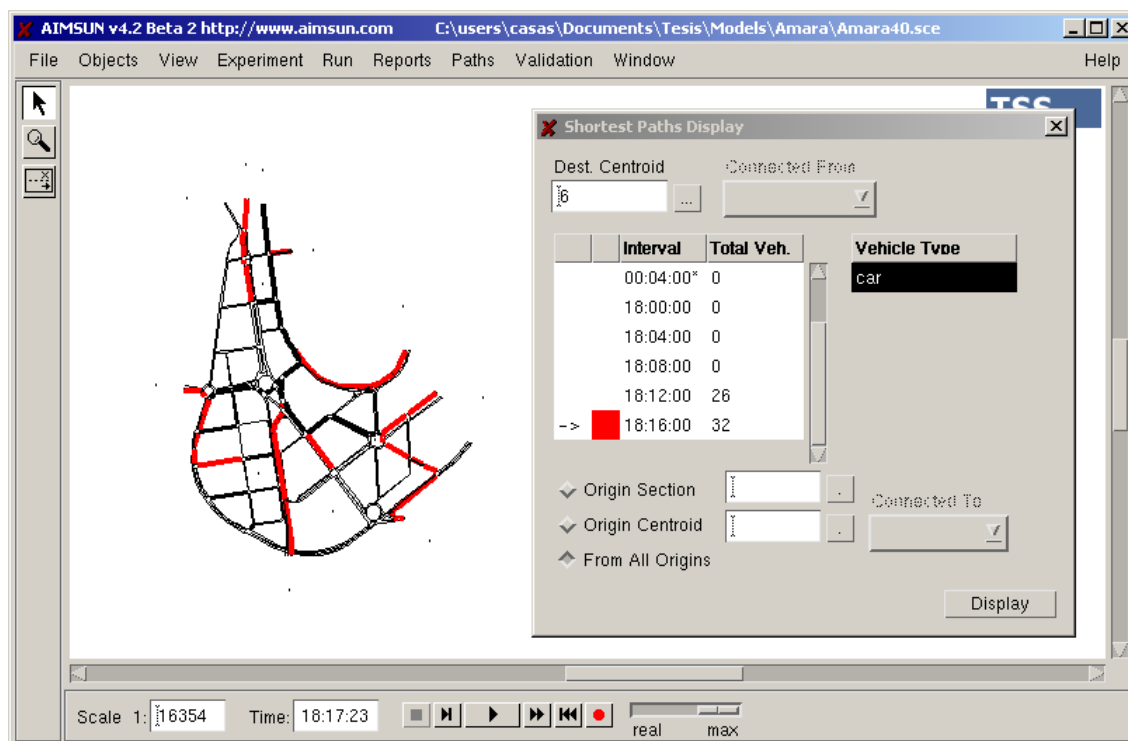
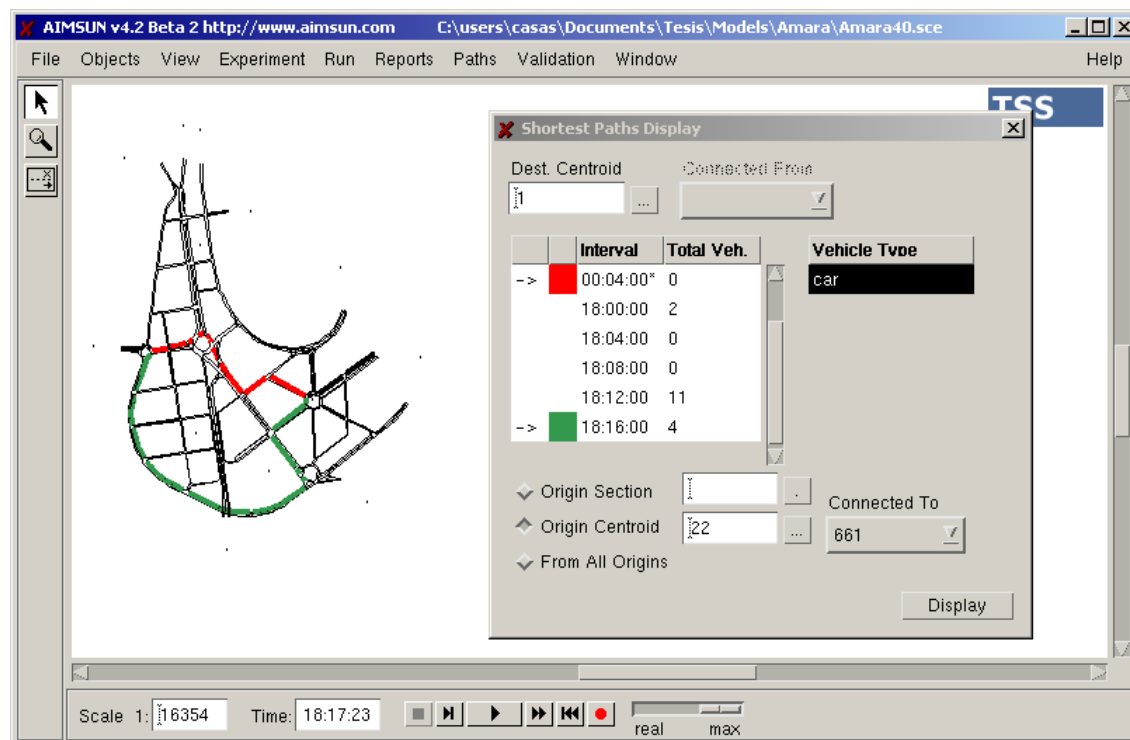


Figure 4-30: Shortest Path Display from one origin to one destination

4.3.1.4 Initial Path Assignment Information

When a vehicle decides to enter the system, it has to decide which path it will follow during its trip. This decision is based using probabilities of user-defined assignment and the probabilities calculated using the Route Choice model, explained in section 4.2.4.

Initial Path Assignment is always accessed by origin and destination centroid. Consequently, an origin centroid and a destination centroid have to be selected first. This can be done either by typing the centroid identifier in the 'Origin' or 'Destination' dialog window, by using the Browse button, or by writing on the identifier and the enter key directly. Once the origin and destination centroids have been selected, a list of user-defined paths with identifier and usage probability per vehicle type are displayed in the 'User-defined Assignment' list box. The use probabilities of all shortest paths per vehicle type and per entrance section and per exit section calculated considering the route choice model are displayed in the Estimated Assignment section. The Estimated Assignment requires determination of the entrance section when the origin centroid considers probabilities and the exit section when the destination centroid considers probabilities. In the event that the destination doesn't consider the probabilities, the exit section displays 'All' and the exit section will consider the shortest path.

If a path is not feasible, either a User-defined path or a calculated shortest path, it shows 'N/F' as a probability value instead of 0.

In the example shown in Figure 4-31, the initial path assignment for an 'unguided' bus that goes from centroid 10 to centroid 11 has a probability of 0.3 to choose user-defined path 1, 0.2 to choose user-defined path 2 and 0 to choose user-defined path 3 and the initial shortest path (ICSP). The 50% of buses will follow either user-defined path 1 or 2 (user-defined assignment) and the other 50% will consider the probabilities calculated using the route choice model (estimated assignment). Continuing with the example, one bus has a probability of 0.33 to take path calculated at 00:05:00, 0.33 to take path calculated at 00:10:00 and so on. The Initial assignment for a 'guided' bus considers only the probabilities calculated using the route choice model.

Figure 4-31: Initial Assignment Information

Initial Path Assignment

Origin: 10 Destination: 11 [Close]

User-defined Assignment

Path Id	bus	car
1	0.30	0.30
2	0.20	0.00
3	0.00	0.00
ICSPT	0.00	0.00

Estimated Assignment

Origin Connect To: 1 Dest. Connect From: 12

Path Id	Ent. Sect.	Exit Sect.	bus	car
00:00:00	1	12	0.00	0.00
00:05:00	1	12	0.33	0.33
00:10:00	1	12	0.33	0.33
00:15:00	1	12	0.33	0.33

4.3.1.5 En-Route Path Assignment Information

When a vehicle decides to make a reassignment of the path during its trip, the decision is taken using probabilities calculated by the Route Choice model.

Dynamic Path Assignment is always accessed via the destination centroid. Consequently, a destination centroid must be selected first. This can be done either by typing the centroid identifier in the 'Origin' or 'Destination' dialog window, by using the Browse button, or by writing on the identifier and the enter key directly. Once the destination centroid is selected, the next step is to select an origin section to obtain the particular path from that section to the destination centroid. This can be done by typing the identifier in the 'Origin Section' dialog window, using the Browse button, or by writing the section image in the network directly or by writing the section identifier directly. For the shortest path, a list of probabilities of all shortest paths per vehicle type and per exit section calculated by considering the route choice model is displayed. The Dynamic Path Assignment requires determination of the exit section in the case of the destination centroid by considering the probabilities. If the destination doesn't consider the probabilities, the exit section displays 'All' and the exit section will be determined by considering the shortest path.

If a path is not feasible from the origin section, 'N/F' is displayed as a probability value instead of 0. In the example shown in Figure 4-32, the En-Route path assignment for a bus that goes to centroid 11 and is located in section 14 has a probability of 0.33 to take path calculated at 00:05:00, 0.33 to take path calculated at 00:10:00 and so on.

Figure 4-32: En-Route Path Assignment Information

Path Id	Exit Sect.	bus	car
00:00:00	12	0.00	0.00
00:05:00	12	0.33	0.33
00:10:00	12	0.33	0.33
00:15:00	12	0.33	0.33

4.3.2 Simulation Output

The Simulation output for validating the Dynamic Traffic Assignment is: display user-defined paths and calculated shortest path, display Initial and En-Route path assignment. These capabilities are complemented by generating the path information output, generating link costs information and colouring all vehicles per origin, per destination and both, per origin and destination at the same time.

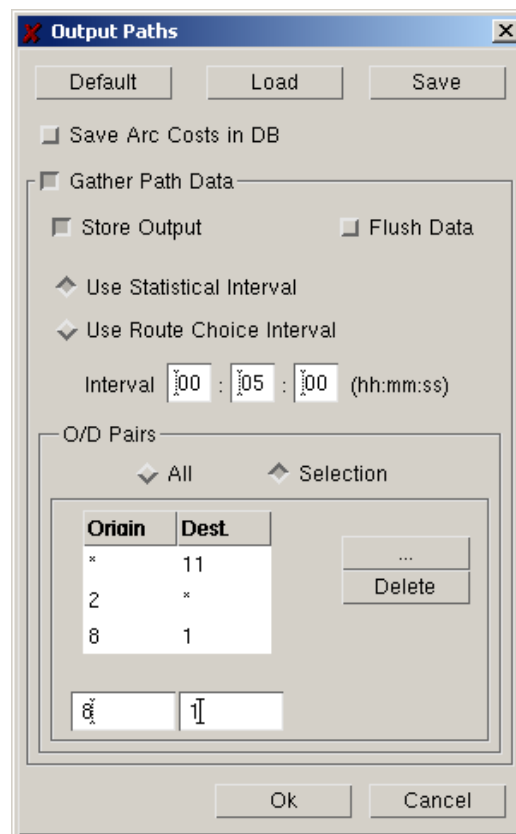
4.3.2.1 Path Information Output

For each calculated shortest path, the path definition (identifier, its calculation time, all links that compose it, etc) and the path statistics (the number of vehicles that have arrived and the average travel time) are output. When the output is based in ASCII files, the path definition is stored in a file named 'PathDefinition.PAT' and the statistical data for each statistical interval is stored in text files named 'HHhMMmSS.PAT', where HH:MM:SS corresponds to the simulation time at which the report was produced. When the output is stored in a database, the tables filled are: 'PathDef', 'PathSect' and 'PatSta'. (see Appendix 2: Output Database Definition)

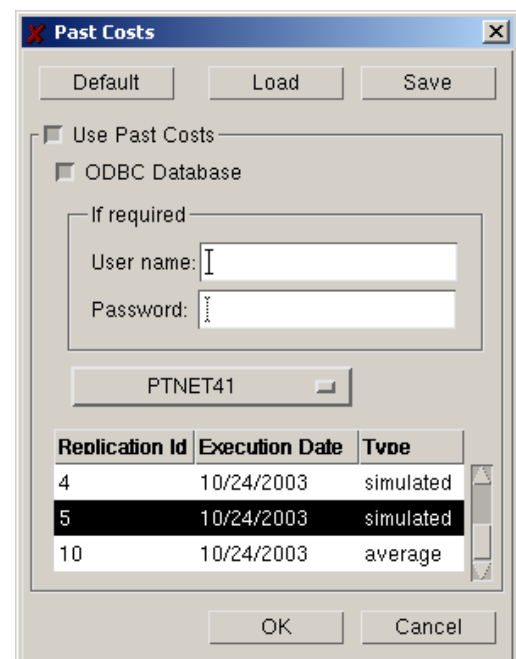
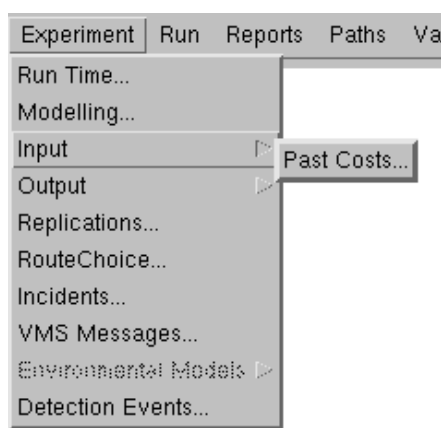
4.3.2.2 Link Costs Output

For each link and time interval used in the Dynamic Assignment, the costs are stored in a database in a table named PastCost. The name of the table is because all these link costs then can be used as the concept of Past Costs in a learning mechanism. (see Appendix 2: Output Database Definition)

This information may be produced automatically during simulation if the user has activated the 'Save Arc Costs in DB' toggle button or 'Gather Path Data' (via the 'Experiment / Output /Paths' command), prior to starting a simulation run. The Path statistics could be generated considering the route choice interval or the statistical interval, depending on the toggle button selected. Once the interval is selected, the user has to select all OD pairs or a selection of them. This selection can be made by writing the centroid identifier or using the Browse button. '*' is treated as a special identifier, indicating all centroids, either origin or destination. For example, in Figure 4-33, the user has selected '*/11', '2/*' and '5/3'. In this case the user wants to store all paths that have any centroid as their origin and centroid 11 as their destination; all paths that have centroid 2 as their origin and any centroid as their destination; and all paths from centroid 5 to centroid 3.

Figure 4-33: Output Paths Dialog Window

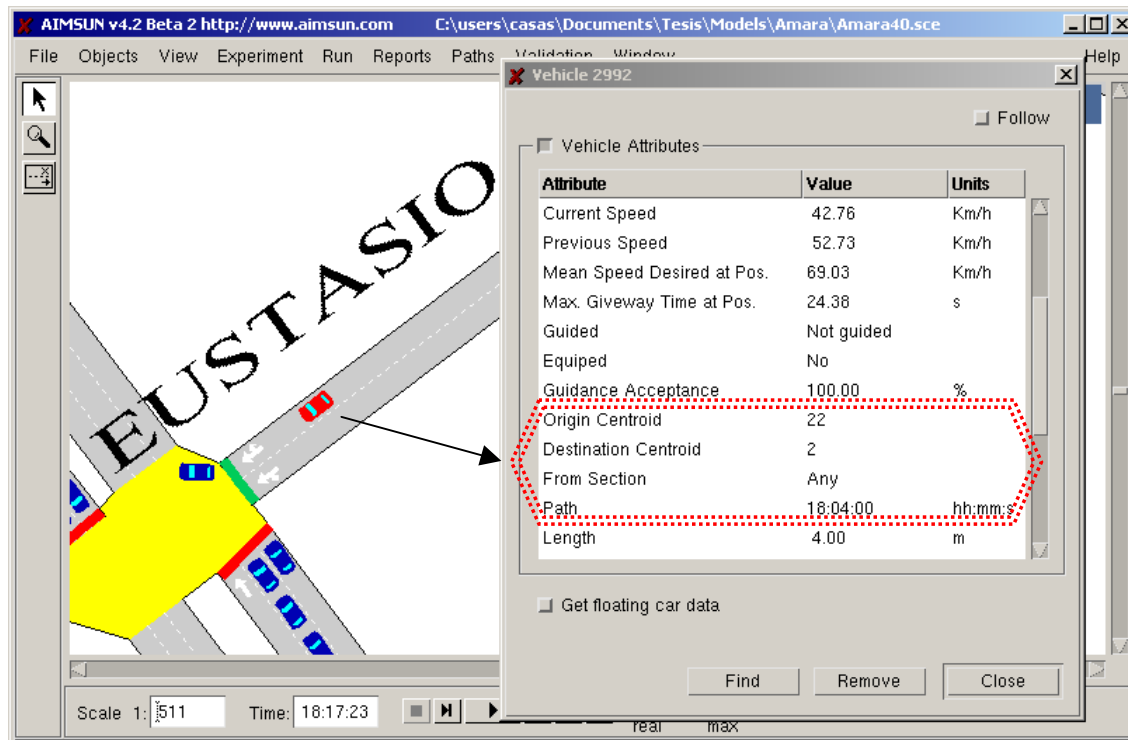
The information from this dialog window is kept in file: *odmatrix path\AIMSUN\ OutputPathPar*. By default, neither cost nor path statistics are calculated. The user may decide to load previously saved Past Costs to be used in a new simulation experiment. This is done via the Experiment/Input/Past Costs option. By default, Past Cost is not used. This information is kept on the hard disk in file: *odmatrix path\AIMSUN\InHistTTPar*

Figure 4-34: Past Costs Dialog Window

4.3.2.3 Vehicle Information

During the simulation, the information of each individual vehicle can be accessed and help to the user identify its origin, destination and the path assigned. In the example shown in Figure 4-35, the vehicle 2992 goes from origin centroid 22 to destination centroid 2 following the path calculated at 18h04m and the exit section will be 'Any', that is the exit section is determined by the shortest path (in the case of destination centroid fixes a percentage to each exit section, then to each vehicle is assigned to one exit section, see section 4.2.6 for details).

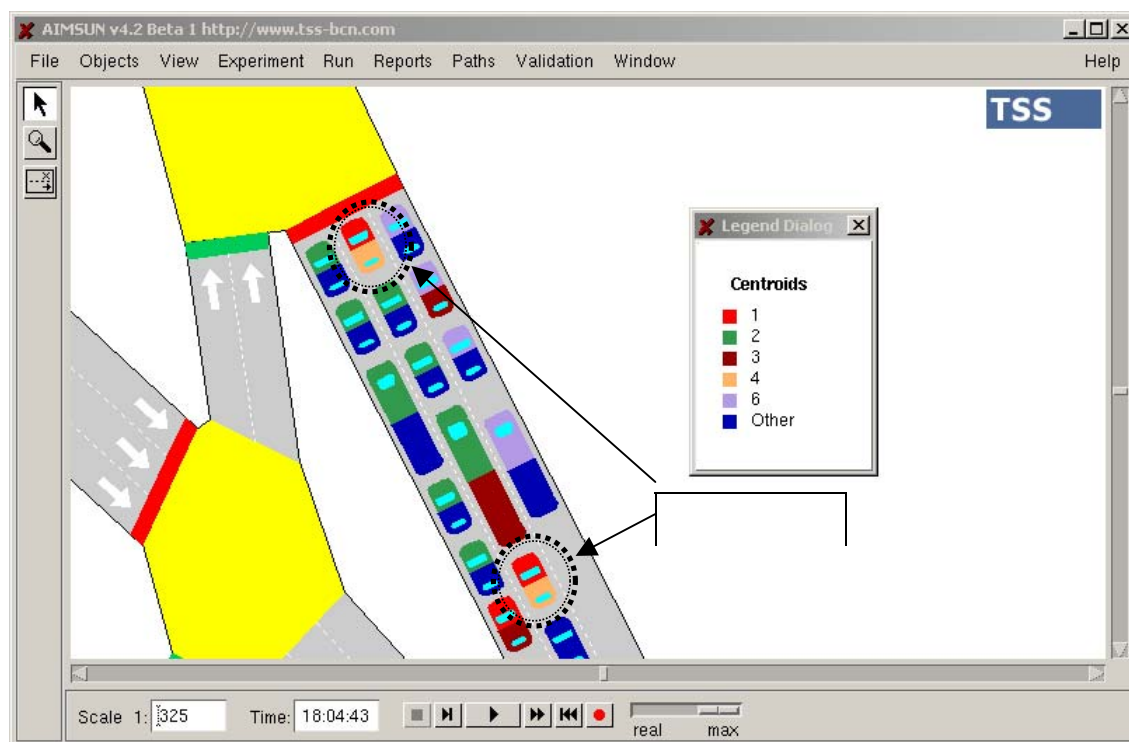
Figure 4-35: Vehicle Information



4.3.2.4 Vehicle Colouring

During the simulation, the vehicles can be coloured either by its origin or its destination or both. This help, during the calibration and validation process, identify quickly the origin and destination and understand the path assigned to each in individual vehicle.

Figure 4-36 shows a network, where all vehicle are coloured by O-D pair, displaying the colour assigned to their destination in the front half of the vehicle and the colour assigned to their origin to the back half of the vehicle.

Figure 4-36: Vehicle Colouring per O-D pair

5. MODELLING TRAFFIC CONTROL AND MANAGEMENT

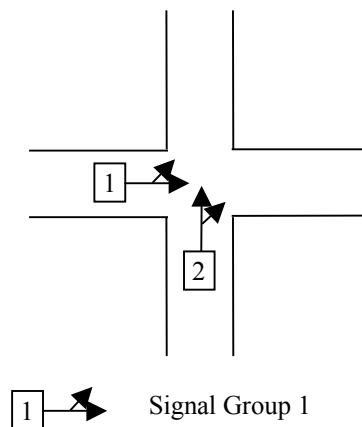
AIMSUN takes into account different types of traffic control: traffic signals, give-way signs and ramp metering. The first and second types are considered for junction nodes, while the third type is for sections that end up in join nodes. On the other hand, AIMSUN also includes the possibility of using Variable Message Signs (VMS) in order to implement Management actions that may somehow affect the traffic behaviour.

5.1 TRAFFIC SIGNAL CONTROL

5.1.1 Signal Groups and Phases

For intersection control, a phase-based approach is applied in which the cycle of the junction is divided into phases, where each has a particular set of signal groups with right of way at the same time. Figure 5-1 shows an example of signal groups for a simple junction.

Figure 5-1: Example of a simple junction, with signal groups.



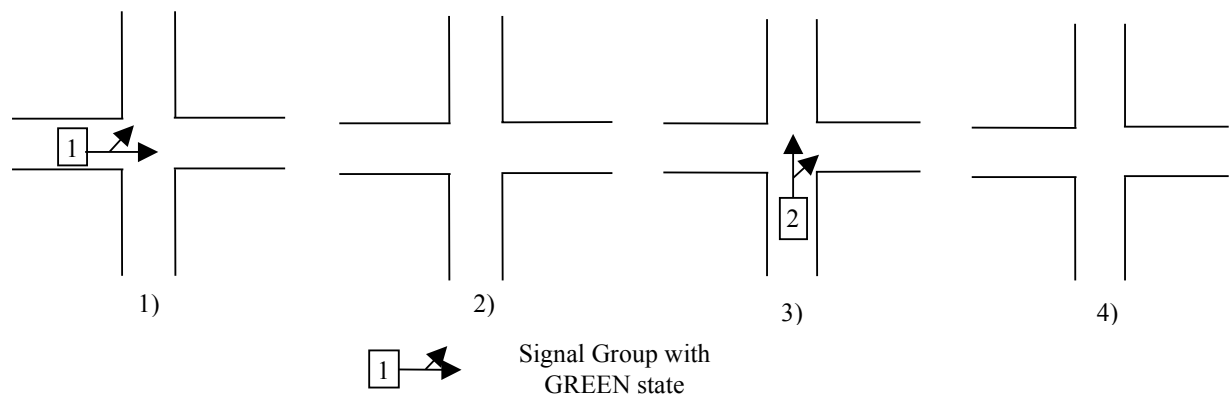
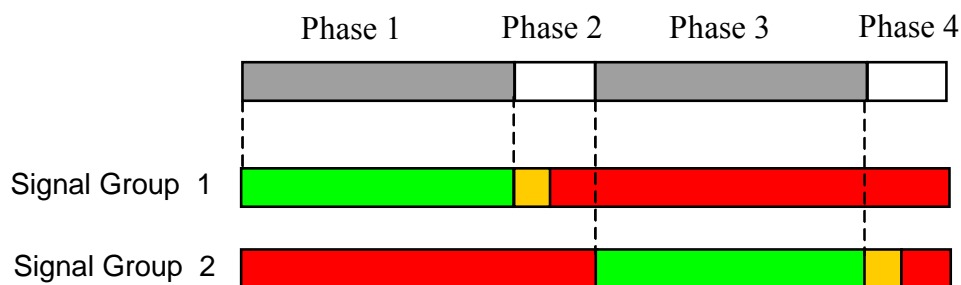
A signal group is considered equivalent to one traffic light. Therefore, all the turning movements that are controlled by the same traffic light and have right of way simultaneously can be grouped into one signal group. Then, a sequence of phases is defined for the whole junction. Each phase has a set of signal groups associated with it. Figure 5-2 shows an example of a right of way sequence for the junction in Figure 5-1, while Figure 5-3 shows phase modelling for the same junction.

In the example, there is a first phase during which turning movements belonging to signal group 1 have a green light. Secondly, there is a clearing-up phase when no movement is allowed. The third phase gives right of way to the movements associated with signal group 2, while the last phase is again another clearing-up phase.

During the simulation of a scenario, AIMSUN executes a fixed control plan taking into account the phase modelling for each junction. However, this fixed control definition can be variable over the simulation period. The user can employ different fixed plans that will be activated during the simulation at the specified time.

On the other hand, an External Traffic Control and Management System (ETCMS) can modify this execution by means of different actions, such as changing the duration of a phase or jumping directly from one phase to another. ETCMS interfacing is available via the GETRAM Extension Module and is explained in detail in the GETRAM Extensions User Manual.

At any point during a simulation you are not allowed to modify the traffic control plan structure (i.e. the definition of signal groups), but it is possible to change the allocation of signal groups to phases or the duration of any phase.

Figure 5-2 Example of rights of way sequence for the junction in Figure 5.1**Figure 5-3: Phase modelling for the junction in Figure 5-1**

The yellow (or amber) time, which is the same for all the traffic lights of the network, is considered as being part of the red time period. In the example of Figure 5-2, signal group 1 has right of way during phase 1, which means that green period starts in phase 1 and red period starts in phase 2, thus changing to yellow, and lasts during the rest of phases.

Traffic signal modelling is implemented using fictitious stopped vehicles, which are created and located at the stop line when the light turns red and eliminated when it turns green. In this way, the car-following model can be used to model braking to stop in front of a red light. If there is non-zero yellow time, the fictitious stopped vehicle is created in the middle of the yellow time. This means that vehicles consider half of the yellow time as green and half as red.

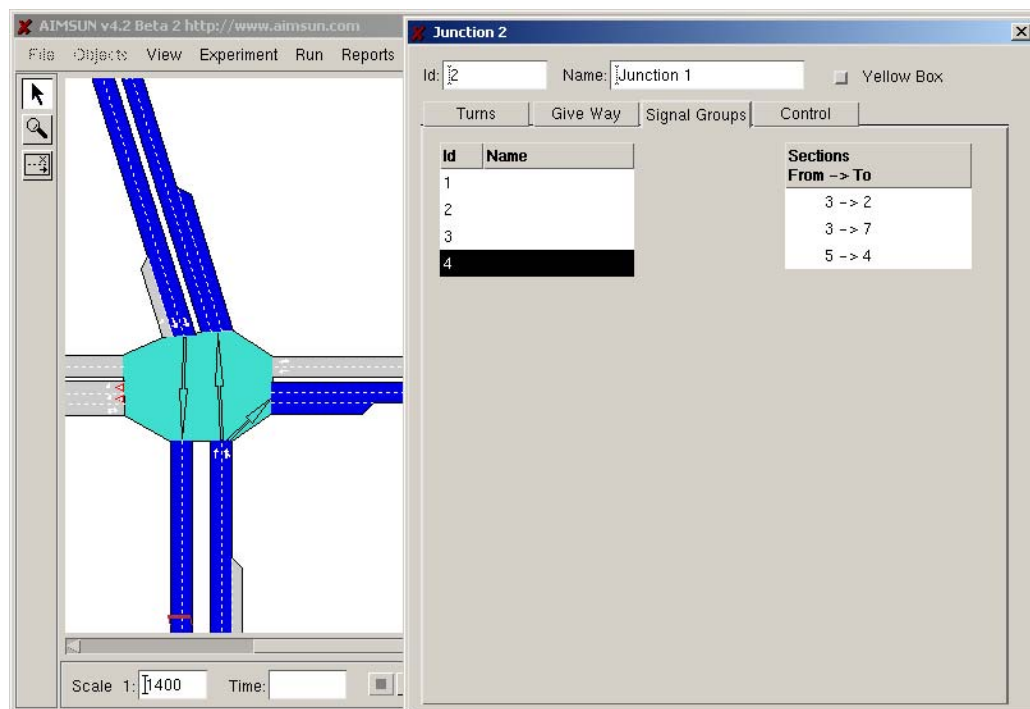
Conflicts between turning movements belonging to the same group may arise in the context of a junction, such as turning while opposing through traffic is crossing. For this purpose, it is possible to define a yield or stop sign for the lower priority turning movements, so they will have to give way to those with higher priority.

5.1.2 Displaying Signal Control at Junctions

The Junction dialogue window contains two tab folders related to the Traffic Signal Control of the Junction: 'Signal Groups' and 'Control'. When you select the 'Signal Groups' tab folder, the information about Signal Groups definition is displayed, as shown in Figure 5-4. Information about both Signal Groups and Control is only available when a Control Plan has been loaded into AIMSUN.

5.1.2.1 Signal Groups

The 'Signal Groups' folder contains two list boxes. The first list box contains the list of Signal Groups defined for the junction. These are numbered from 1 to N. When you select a Signal Group by clicking on it, the set of turning movements (represented by the 'Section From' and 'Section To' identifiers) associated with that Signal Group is displayed in the second list box. Also, the arrows representing all turning movements for the signal group and the lanes involved in the turns are displayed in the network.

Figure 5-4: Junction-signal group information window

5.1.2.2 Control Plan

Selecting the 'Control' tab folder displays information on the phases, as shown in Figure 5-5. A list box contains the Control Plans that have been loaded into the current simulation experiment. This list provides the following information for each Control Plan:

- Control: name of the control plan.
- Interval: Starting time, i.e. time at which the control plan will become active.
- Offset: time offset for the plan, i.e. the time origin for all time settings.

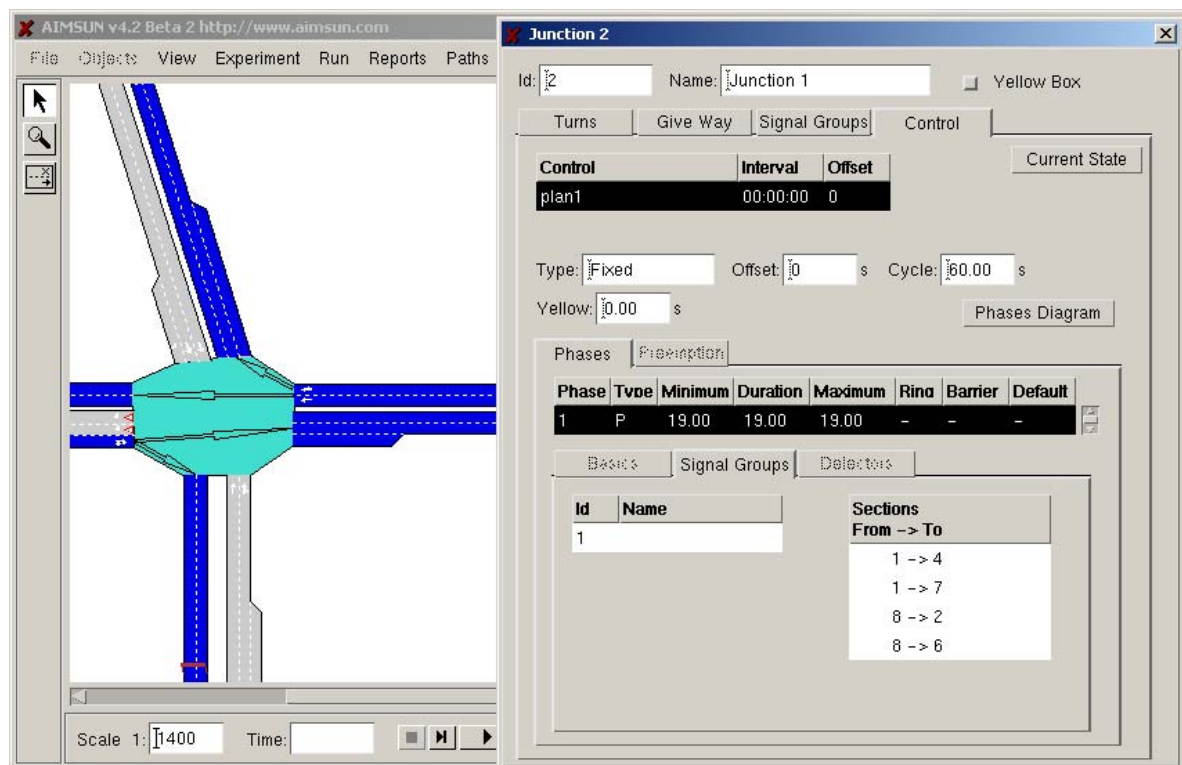
You select a control plan from the list box by clicking on it and the following information is now displayed:

- Type of control: Uncontrolled, Fixed, External or Adaptive
- Time offset: time at which the first phase of the junction starts. The junction offset must be added to the plan offset.
- Yellow Time: duration of yellow or amber time, which will be used for all the traffic lights.
- Cycle: duration of control cycle in seconds, equal to the sum of durations of all the phases.

There is also a list box containing the individual phase information. The following data is displayed for each phase:

- Phase number (from 1 to n)
- Minimum, initial and maximum duration of the stage, in seconds. When simulating with fixed control, all three values would be the same. When simulating with an External Adaptive Control, these parameters provide the initial duration of each phase and the range of feasible variation for the phase duration.

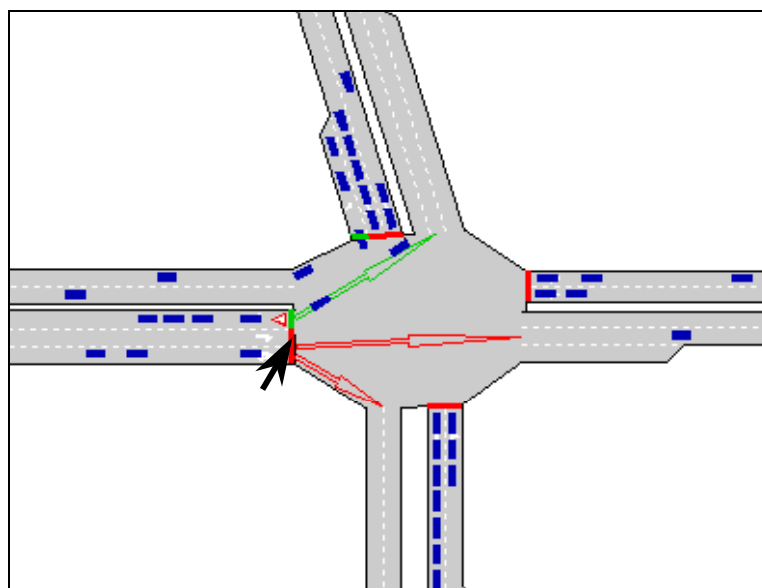
Clicking on a phase, a list of Signal Group identifiers that have right of way during this phase are displayed in the 'Signal Group' tab folder, and the corresponding list of turns is displayed in the 'Sections From->To' list box. Also, the arrows representing all the turning movements of the phase and the lanes involved in the turns are displayed in the network.

Figure 5-5: Junction-Control information window

5.1.2.3 Traffic Lights of a Section

For each section approaching a signalised intersection it is possible to view the status of the traffic light corresponding to every signalised turning movement of that section. The status of the traffic lights can be red, yellow or green.

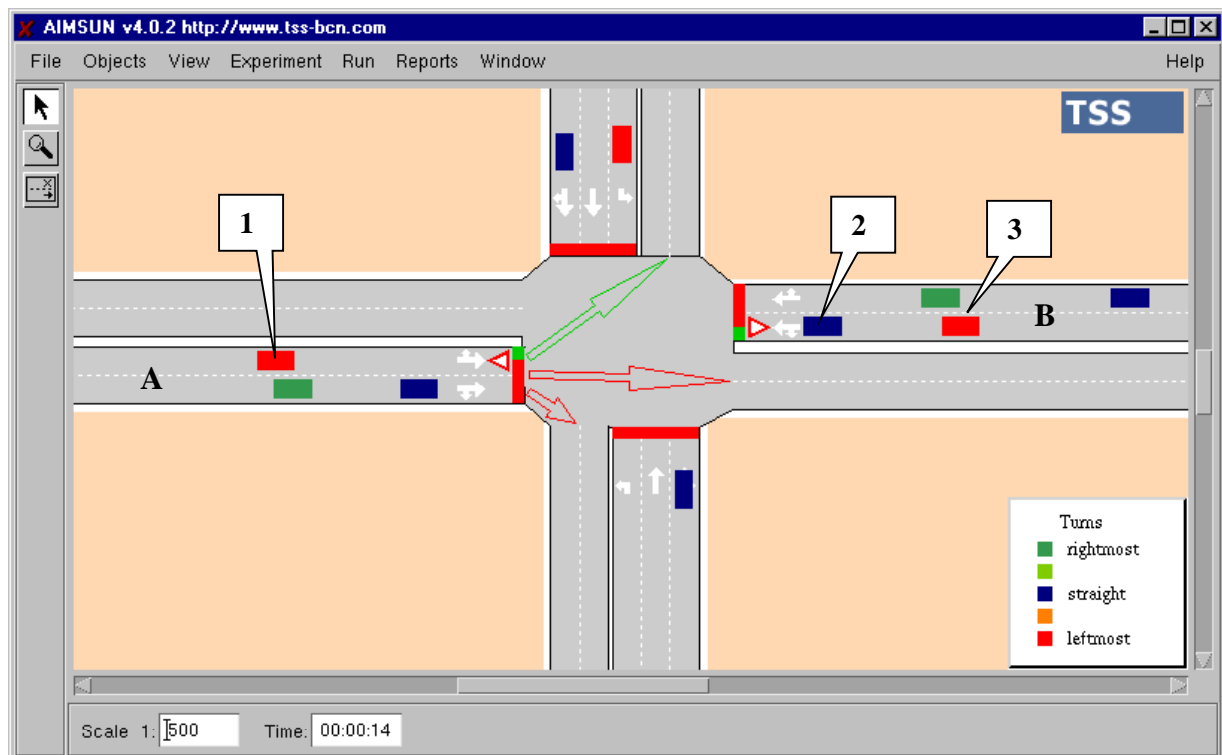
During a simulation run, but only while the simulation is stopped, double-click on the stop line at the end of the section (Figure 5-6). Then, an arrow for each turning movement will be displayed in a different colour, green, yellow or red according to the status of the corresponding traffic light. You can then continue the simulation run and the colours of the arrows will be updated continuously, thus representing the change of traffic light colours.

Figure 5-6: Traffic Lights of a Section

If a particular lane has different turning movements allocated to separate signal groups that have right of way during different phases, it means that two different traffic lights may be valid for the same lane at the same time. This is represented by dividing the stop line for that lane into two portions, each portion painted in a different colour (see Figure 5-7). Vehicles will be allowed to cross or not depending on the turning movement they intend to make.

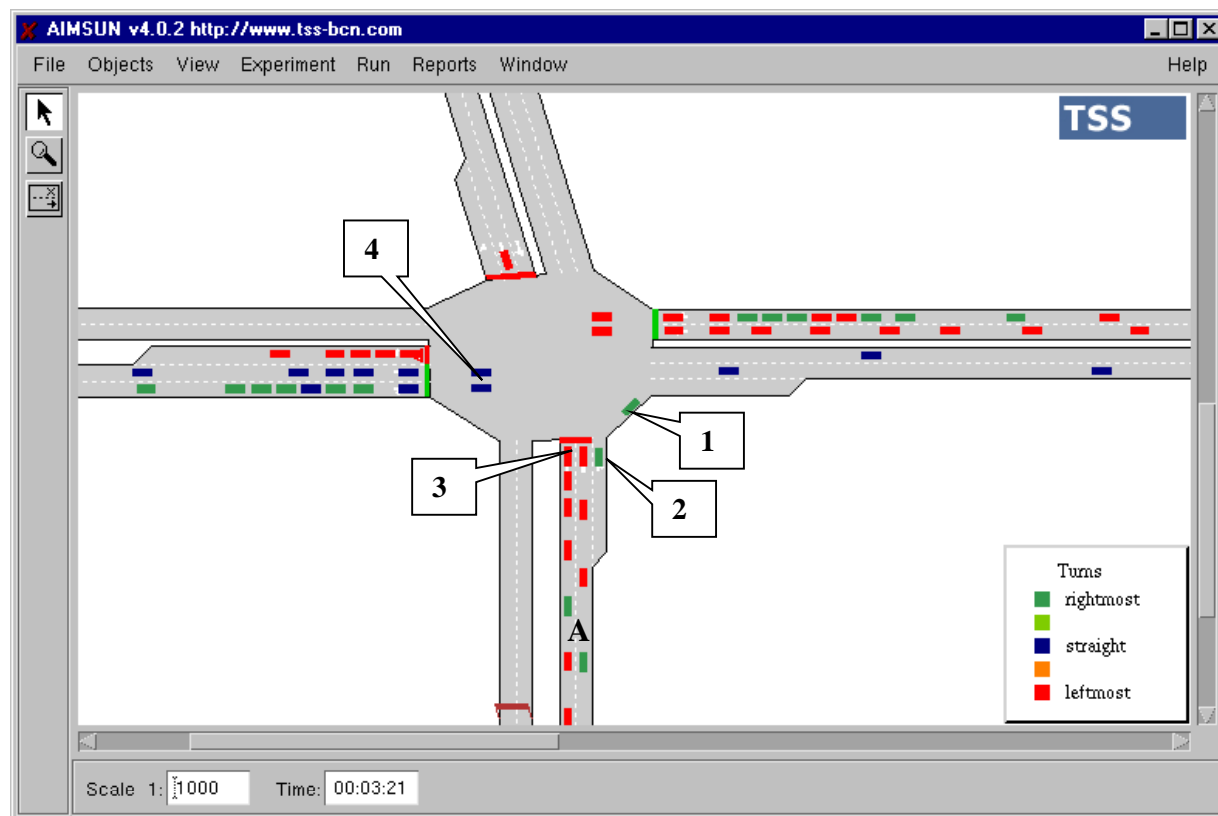
In the example displayed in Figure 5-7, vehicles in the leftmost lane of sections A and B can either go straight or turn left. The traffic light for a straight turn is red while it is green for a left turn. Vehicle 1 wishes to turn left and the traffic light for a left turn is green, therefore it will cross. On the other hand, vehicle 2 wants to go straight, therefore it will stop at the stop line, thus blocking vehicle 3 that wishes to turn left.

Figure 5-7: Traffic Lights sharing a lane



It is also possible to define turning movements that have right of way during the whole cycle. This would therefore be a lane that has no traffic light (or a permanent green light). This is achieved by not assigning the turning movement to any signal group. Figure 5-8 shows an example of a right turn that can always be taken, so no stop line is painted for the rightmost lane of section A. Vehicles 1 and 2 are crossing while the traffic light for section A is red and vehicles 3 are stopped.

Take into account that in the junction control illustrated in Figure 5-8, vehicle 4 does not have priority defined over vehicles 1 and 2. To properly model turns with permanent green or flashing yellow, it would be convenient to include a Yield sign in the right turning movement.

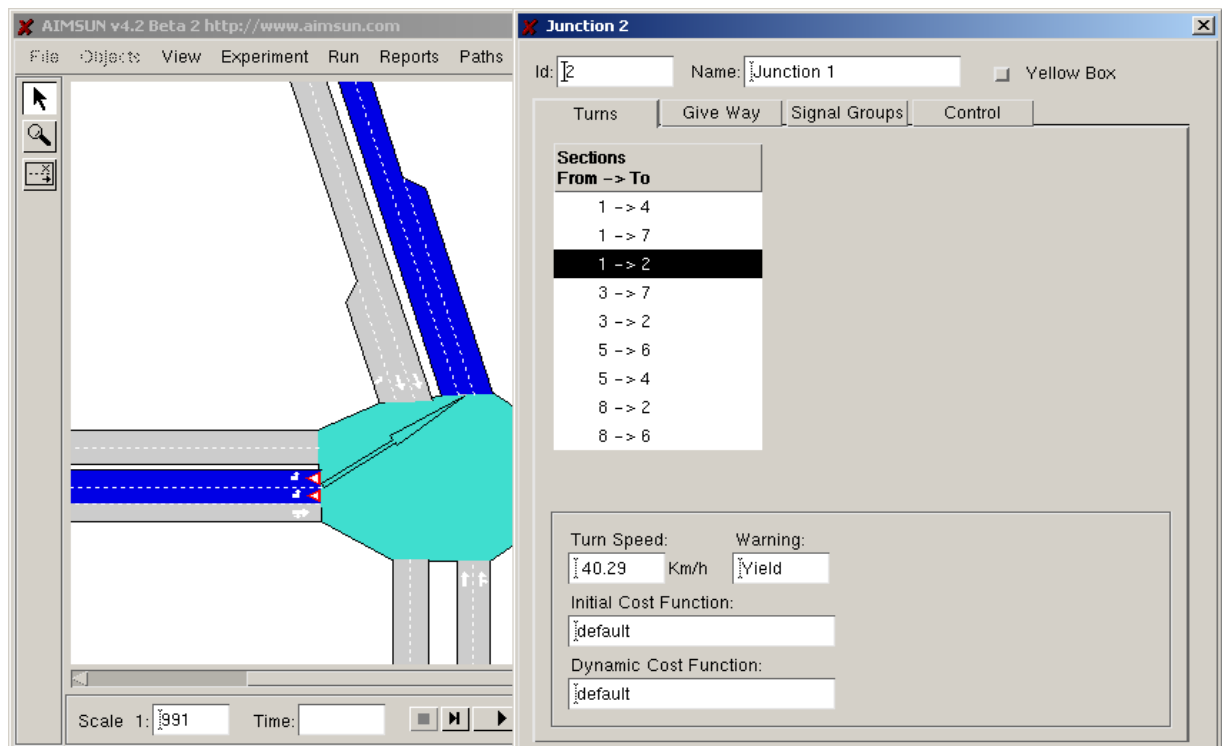
Figure 5-8: Turning movement with permanent right of way

5.2 YIELD AND STOP SIGNS

A yield (give way) or stop sign can be defined for any turning movement at a junction node. Vehicles that intend to cross the junction through a turning movement affected by a sign will have to give way to any other vehicle approaching through any other turn without a sign. Figure 5-9 shows a signalised intersection where a Yield (Give Way) sign has been defined for the left turn (from section 1 to 2), as it takes place while opposing through traffic is crossing.

For both types of give-way signs, Yield and Stop, the vehicle's decision on whether to cross or yield is modelled using the same Gap-Acceptance Model, although the decision is taken in different situations. When it encounters a Stop sign, the vehicle will come to a complete stop before making the decision about whether or not to cross. When it encounters a Yield sign, the vehicle starts to apply the Gap-Acceptance model as soon as it is nearer than 'Visibility Distance at Junction' from the stop line. As the vehicle approaches the stop line it will start looking for a gap. If it finds one, it will accelerate and cross. If it does not find a safety gap, it will keep decelerating and looking for a gap until it can cross or until it reaches the stop line and halts.

Figure 5-9: Yield sign for left turn in a junction



5.2.1 Gap-Acceptance Model

A Gap-Acceptance model is used to model give way behaviour. This model determines whether a lower priority vehicle approaching a junction can or cannot cross depending on the circumstances of higher priority vehicles (position and speed). This model takes into account the distance of vehicles from the hypothetical collision point, their speeds and their acceleration rates. It then determines the time needed by the vehicles to clear the junction and produces a decision to cross or not which is also a function of the level of risk for each driver.

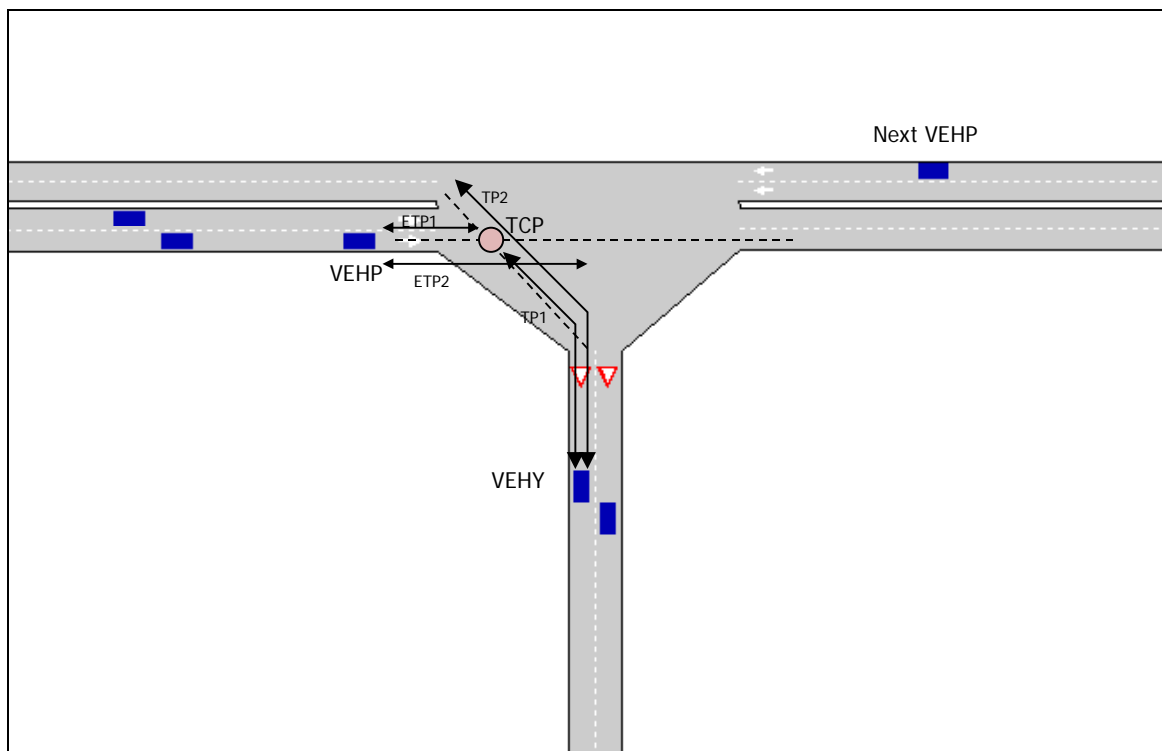
Several vehicle parameters may influence the behaviour of the gap-acceptance model: acceleration rate, desired speed, speed acceptance and maximum give-way time. Other parameters, such as visibility distance at the junction and turning speed, which are related to the section, may also have an effect. Among these, the acceleration rate, the maximum give-way time and the visibility distance at junctions are the most important.

The acceleration rate gives the acceleration capability of the vehicle and therefore has a direct influence on the required safety gap. The maximum give-way time is used to determine when a driver starts to get impatient if he/she cannot find a gap. When the driver has been waiting for more than this time, the safety margin (normally two simulation steps) is reduced by half (only one step).

The following algorithm is applied in order to determine whether a vehicle approaching a give-way sign can cross or not (see Figure 5-10):

Given a vehicle (VEHY) approaching a Yield (Give Way) junction,
 Obtain the closest higher priority vehicle (VEHP),
 Determine the Theoretical Collision Point (TCP),
 Calculate time (TP1) needed by VEHP to reach TCP,
 Calculate estimated time (ETP1) needed by VEHP to reach TCP,
 Calculate time (TP2) needed by VEHP to cross TCP,
 Calculate estimated time (ETP2) needed by VEHP to clear the junction,
 If TP2 (plus a safety margin) is less than ETP1, vehicle VEHP has enough time to cross, therefore it will accelerate and cross,
 Else, if ETP2 (plus a safety margin) is less than TP1, vehicle VEHP will have already crossed TCP when VEHP reaches it, then search for the next closest vehicle with a higher approach, Next VEHP and go to step 2.
 Else, vehicle VEHP must give way, decelerating and stopping if necessary.

Figure 5-10: Gap-acceptance model



5.2.2 Give Way Priority Definition

For the turning movements with yield sign defined (Give Way or Stop sign) the user can define what are the turning priorities through the Give Way folder which is in the Junction dialog.

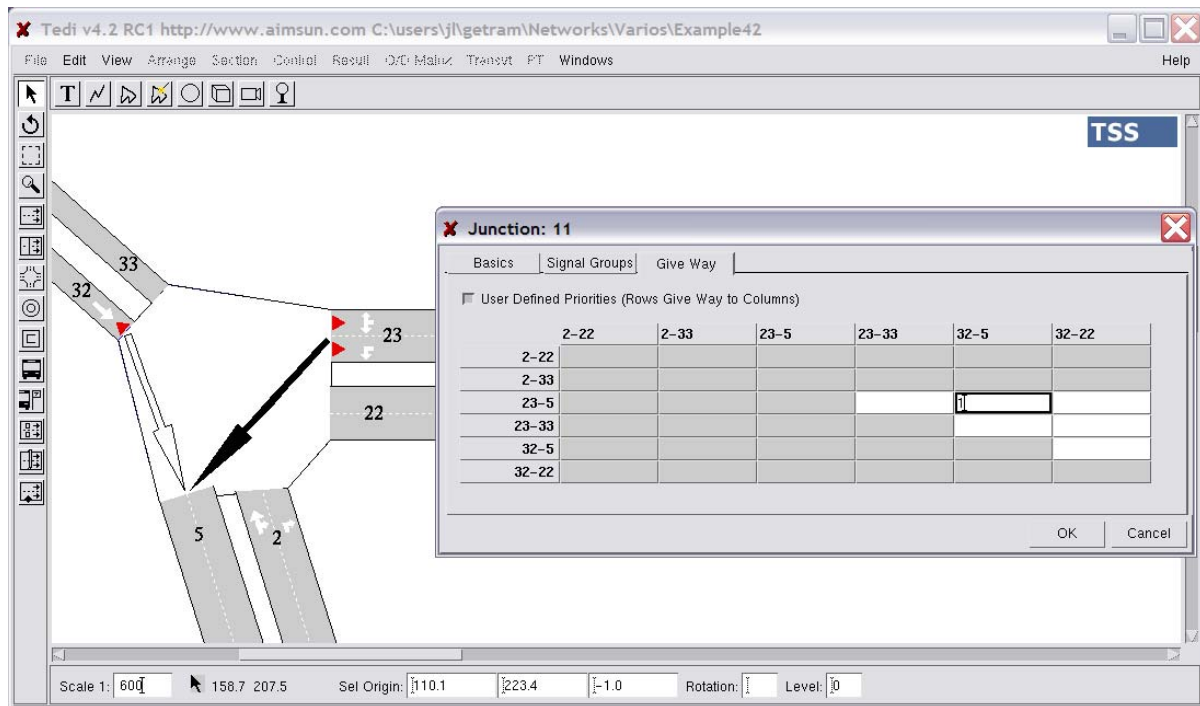
A matrix with all turning movements of the junction appears in the Give Way folder. Only the cells corresponding to pair of movements both having yield sign, are editable. It is assumed that any turn with

yield sign will give way to all turns without yield sign. Priority definition only affects to the relations between pairs of movements with yield sign.

To set a priority, put a 1 in the corresponding cell, taking into account that the turns in the rows of the matrix give way to the turns in the columns where there is a 1 in the cell.

In the figure 5-10b, turn 23-5 will give way to turn 32-5. Of course that turn 23-5 will also give way to turn 2-33, as this turn has not yield sign, so no need to define this priority.

Figure 5-10b: Give Way Priority Dialog



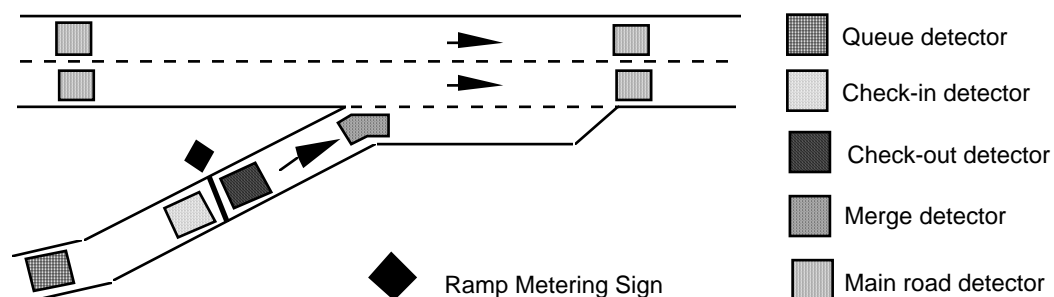
5.3 RAMP METERING

AIMSUN incorporates ramp-metering control. This type of control is used to limit the input flow to certain roads or freeways in order to maintain certain uninterrupted traffic conditions. The objective is to make sure that entrance demand never surpasses the capacity of the main road. Ramp meters are located at the downstream end of a section approaching a node type juncture and affect all the lanes of the section. Figure 5-11 displays a ramp-metering layout.

AIMSUN considers three types of ramp metering depending on the implementation and the parameters that characterise it:

- 1) Green time metering. Parameters are green time and cycle time. This is modelled as a traffic signal.
- 2) Flow metering. Parameters are platoon length and flow (veh/h). The meter is automatically regulated in order to permit the entrance of a certain maximum number of vehicles per hour. This model is currently under development.
- 3) Delay metering. Parameters are the mean delay time and the standard deviation. This is used to model the stopping of vehicles due to some control facility, such as a toll or customs barrier.

Figure 5-11: Ramp metering layout



Ramp meters may be located at any point of a section. Selecting the 'Equipment' tab folder in the 'Section' window will show three additional new tab folders. Select the 'Metering' tab folder to display the Metering Information, as displayed in Figure 5-12. This folder contains the Metering Identifier (a string of characters), a list of Traffic Control Plans, and the metering parameters, which depend on the type of metering. Selecting a Control Plan from the list box will display the corresponding control parameters for that meter.

Another way of accessing ramp metering information is by using the 'Objects / Metering' command. This displays the 'Metering Dialog' search window displayed in Figure 5-13. The user can search for a meter by clicking on the name in the list box and pressing the 'Find' button. Pressing the 'Open' button, opens the 'Metering' information window (see Figure 5-14). Double clicking directly on the ramp-metering image in the network has the same effect.

This window provides not only information about the ramp metering parameters but also the section identifier for its location and its position in the section (measured from the beginning of the section). Also, depending on the type of metering, another set of parameters is displayed in the 'Metering Dialog' information window.

Figure 5-12: Section-Metering Information Window

Section 11

Basics Lanes Traffic Equipment

Detector Metering VMS PT Stops

Id: 1 Name: Metering-11

Control	Interval	Offset
plan1	00:00:00	0

Type: Fixed

Green-time Metering (sec)

Cycle: 60.00 Offset: 0.00 Yellow: 0.00

min current max

30.00 <= 30.00 <= 30.00

Figure 5-13: Metering Dialog Search Window

Meterings Dialog

Id

Metering-11

Metering-16

Find Open Close

5.3.1 Green Time Metering

The ramp meter is controlled by a traffic signal that turns red and green on a cyclical basis. The control cycle for the metering and the green time values are displayed (as shown in Figure 5.14). If we are using fixed traffic control here, only the 'current' green time is used. If we are simulating in conjunction with an External Adaptive Traffic Control System, the 'min' and 'max' fields show the minimum and maximum values for the acceptable range of green time variation. For the rest of the cycle time, the traffic signal will be red. Yellow time is also modelled during Green Time Metering.

Figure 5-14: Green Time Metering Dialog Window

Metering 1

Id: 1

Name: Metering-11

Section Id: 11 Position: 63.2 m

Control	Interval	Offset
congested	00:00:00	0

Type: Fixed

Green-time Metering (sec)

Cycle	Offset	Yellow
60.00	0.00	0.00

min 33.00 ≤ current 33.00 ≤ max 33.00

5.3.2 Flow Metering

In this case, the ramp-metering objective is to let a certain number of vehicles go past the meter per hour. Now the parameters displayed are the Platoon Length and the Traffic Flow (see Figure 5-15). Each time the meter is opened to release vehicles, it is done in such a way that platoons, whose length is 'Platoon Length' parameter, can pass. On average, the 'current' number of vehicles per hour will be released. If simulating is being carried out in conjunction with an external Adaptive Traffic Control System, the 'min' and 'max' fields show the minimum and maximum values for the acceptable range of flow variation.

Figure 5-15: Flow Meter Dialog Window

Metering 2

Id: 2

Name: Metering-16

Section Id: 16 Position: 89.5 m

Control	Interval	Offset
congested	00:00:00	0

Type: Fixed

Flow Metering (veh/h)

Platoon Length	Yellow
2	

min 500.00 ≤ current 500.00 ≤ max 500.00

Detail of Traffic Control Events in the modelling of a Flow Ramp Metering

Let's assume a simulation step ($dt = 0.75$). Let's assume a Flow Metering of 750 veh/hour, which means that the meter will release a vehicle every $3600/750 = 4.8$ sec., which is not an integer number of dt .

Events 'change to green' should theoretically take place at times:

$$t_1=4.8, t_2=9.6, t_3=14.4, t_4=19.2, \text{ etc.}$$

which means a constant time interval between events of $I_i=4.8$ sec.

In the simulation, Events 'change to green' will effectively take place at times:

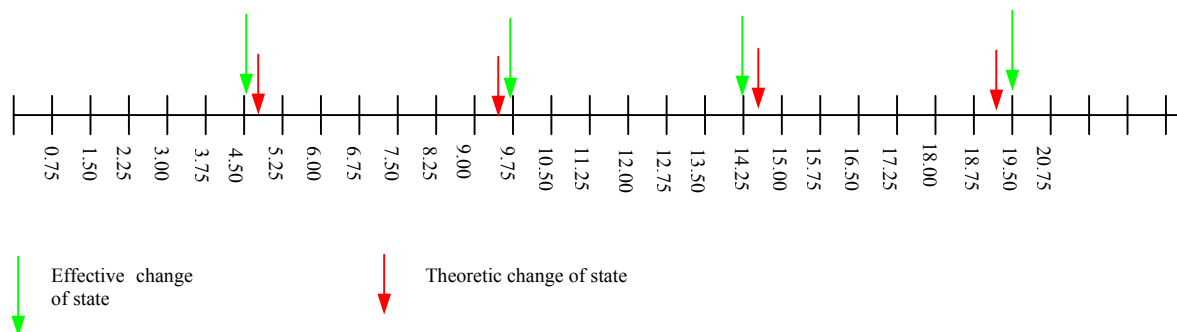
$$t_1'=4.50, t_2'=9.75, t_3'=14.25, t_4'=19.50, \text{ etc}$$

which are the times multiple of dt that are closest to the theoretical times. It means that the time intervals between events are now variable:

$$I_1=4.50, I_2=5.25, I_3=4.5, I_4=5.25, \text{ etc}$$

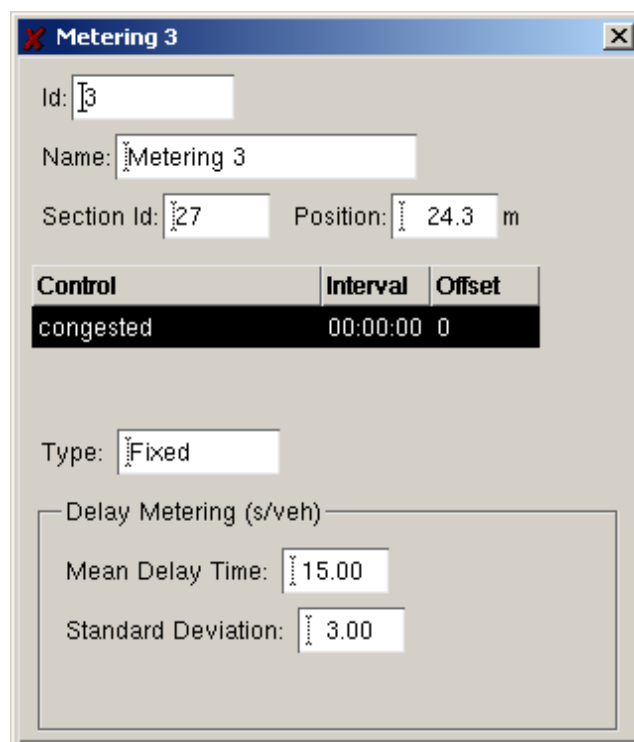
On average, however, the time interval between events would be 4.8 sec.

This mechanism ensures that, on average, 750 veh/hour will be released. Moreover, no error is carried on, as we always calculate the next event adding the theoretical time intervals between events to the last theoretical time event and finally adjusting the time to the closest integer of dt .



5.3.3 Delay Metering

This type of metering is used to model tolls, customs, checkpoints or any other type of individual control. It is assumed that every vehicle will have to stop at the control point (the ramp meter stop line) for a certain time. This time is assumed to be a normally distributed random variable with a mean of 'Mean Delay Time' and a standard deviation of 'Standard Deviation'. These parameters are displayed in the Delay Metering Window in Figure 5-16.

Figure 5-16: Delay metering Dialog Window

Metering 3

Id:

Name:

Section Id: Position: m

Control	Interval	Offset
congested	00:00:00	0

Type:

Delay Metering (s/veh)

Mean Delay Time:

Standard Deviation:

5.4 TRAFFIC MANAGEMENT WITH VMS's

Information to the drivers is considered as a possible result of the effect of a Traffic Management System on a network containing Variable Message Signs (see section 2.5.3 for a description of VMS's). Messages may inform the drivers about incidents, congestion or suggest alternative routes. AIMSUN takes into account the modelling of VMS and their influence on driver behaviour.

In AIMSUN, messages in a VMS may be activated in different ways:

1. Directly via the user interface by clicking on the VMS and pressing the 'Activate Message' button. Any message from the messages list box can be scheduled or activated at any time during simulation.
2. Loading a log file of messages (saved in a previous simulation via the '*Experiment/VMS Messages*' command) to be activated during the simulation.
3. Via the GETRAM Extensions, any external system can activate a message on any VMS in the network, by sending the corresponding command, consisting of the VMS identifier and the message text.

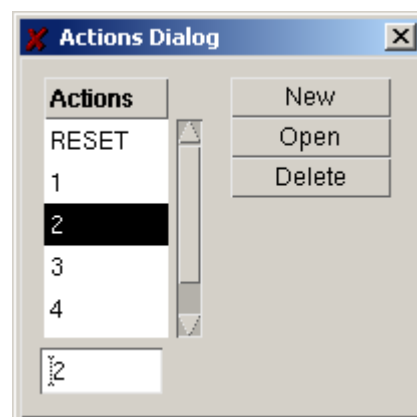
Options 1 and 2 have been already explained in section 2.5.3, while option 3 is explained in the GETRAM Extensions Manual.

In all cases, the message will be displayed as an Activated Message at the appropriate time and the Actions associated with it will be implemented. Each message has a list of Actions associated with it, which appear in the list box named 'Mess Actions' in the VMS Information Window (see Figure 2-25). The list box named 'Actions' contains all actions available for this network. The user may add/remove actions to/from the list of actions associated with a message by using the bottom part of the window.

An Action represents the impact that a message has on driver behaviour. Different types of actions are considered depending on whether the simulation is carried out on a Traffic Result basis (Input flows and turning proportion) or in the Route-Based simulation mode.

The user can access the Actions information in two ways, either by selecting the 'Objects / Actions' command from the menu bar, or by using the Actions area in the 'VMS's' information window (Figure 2-23). Selecting the 'Objects / Actions' command will display the 'Actions' selection window (see Figure 5-17). The user can create, delete or edit actions in this window.

Figure 5-17: Action selection window

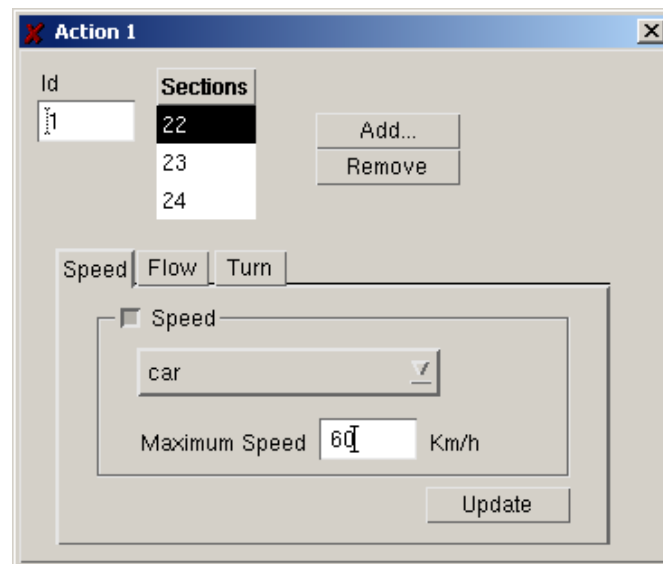


To create a new action, click on the 'New' and 'Add' buttons. To delete an existing action, select it by clicking on the list box and then click on the 'Delete' button. To edit or view an existing action, click the 'Open' button after selecting the action in the list box, either in the VMS information window or in the Actions Selection window. In any case, when opening an action the 'Action' information window will be displayed. The type of window displayed will depend on whether the simulation is Traffic Result based or Route-Based.

5.4.1 Actions for a Traffic Result Based Simulation

When the simulation is carried out on a Traffic Result basis (Input flows and turning proportion), the 'Action' information window shown in Figure 5-18 is displayed. This includes an area containing the list of sections where the action will have any impact and three tab folders: Speed, Flow and Turn. The user can add sections to the list with the 'Add...' button and can then click on the section in the network display. To remove sections, just select the section from the list box and press the 'Remove' button.

Figure 5-18: Actions-Speed Information Dialog Window (Result-Based)



Three types of actions can be defined: modifications to the speed limit, modifications to the input flow and modifications to the turning proportions.

Modifications to the Speed Limit

Select the Speed tab folder from the Actions dialog window (see Figure 5-18). Click on the Speed toggle button, select the vehicle type or 'all', and type the new speed limit for that set of sections into the 'Maximum Speed' field. When a message containing this action is displayed on a VMS, it will automatically modify the Speed limit for the selected sections and all vehicles of the selected type driving along these sections will therefore be affected.

Modification to the Input Flow

Select the Flow tab folder from the Actions dialog window (see Figure 5-19). Then click on the Flow toggle button and the list of vehicle types will be displayed. For each vehicle type, the user can select the increase or decrease to this flow as a percentage of the current flow. To do so, click on the vehicle type list box, select increase or decrease using the '+, -' option menu and type the required amount into the 'Flow' field. For example, in Figure 5-19, 'Car +0.20' means an increment of 20% in Car input flow for section 13, while 'hgv -0.15' means a 15 % reduction in the hgv input flow. Input flow modifications can only affect input sections where traffic is generated and injected into the network.

Modifications to Turning Proportions

Select the Turn tab folder from the Actions dialog window (see Figure 5-20). Then click on the 'Turn' toggle button and the list of selected turnings, if any, for the current section (the one selected in sections list box at the top of the window) will be displayed. The user can add or remove turnings to this list using the 'Add' or 'Remove' buttons. Selecting one of the turnings and one of the entrances, the user can define an increase or decrease to the proportion of vehicles having entered the section through that entrance that will follow a turn. The value defined in the 'Prob.' field represents the percentage of increase or decrease over the current turning probability. For instance, in Figure 5-20, the turning probability for traffic entering section 1 from section 11 and willing to turn into section 14 will increase by 20% for all vehicle types.

Figure 5-19: Action Flow Information Dialog Window (Result-Based)

Action 1

Id: 1

Sections: 24, 25

Buttons: Add..., Remove

Tabs: Speed, Flow, Turn

☒ Flow

Vehicle Type	Flow
bike	-0.10
car	+0.20
truck	+0.15

Input: car, +, 0.20

Update

Figure 5-20: Actions-Turn Information Dialog Window (Result-Based)

Action 2

Id: 2

Sections: 23, 32

Buttons: Add..., Remove

Tabs: Speed, Flow, Turn

☒ Turn

Turns	Entrances
33	28
5	

Buttons: Add..., Remove...

Vehicle Type	Prob.
bike	+0.20
car	+0.20
truck	+0.20

Input: car, +, 0.20

The following section contains an explanation and some examples of how the new turning proportions of a section are calculated whenever an action that increases or decreases a percentage of the turning proportion is applied.

When an increase or decrease over a turning proportion is defined, it is carried out in relation to the original percentage and will be subtracted from or added to the other turns of the section for which increases or decreases are defined. Therefore, when an increase is defined for a/some turn(s), it is necessary to define a corresponding decrease for other turn(s) in the same section.

The procedure is as follows:

1. Calculate the modifications of the percentages in all the affected turns.
2. Adjust the resulting percentages of the affected turns in order to keep the original total of all proportions

5.4.2 Examples of Actions

Example 1

Let us assume that we have a section 1 with three possible turnings (to sections 2, 3 and 4) with the following initial turning proportions:

1 → 2	0.30
1 → 3	0.50
1 → 4	0.20

And now suppose that we have defined an Action as follows:

1 → 2	+ 50%
1 → 3	– 40%

When this action is applied the following calculations are made:

1. Calculate the modifications to the percentages in the affected turns

$$\begin{aligned}
 1 \rightarrow 2 & \quad 0.30 + (0.30 \cdot 50\%) = 0.30 + 0.15 = 0.45 \\
 1 \rightarrow 3 & \quad 0.50 - (0.50 \cdot 40\%) = 0.50 - 0.20 = 0.30
 \end{aligned}$$

2. Adjust the resulting percentages in order to keep the same sum of proportions for the affected turns as the original:

	<u>Original</u>	<u>New</u>	
1 → 2	0.30	0.45	
1 → 3	0.50	0.30	
	-----	-----	
	0.80	0.75	=> There is a difference of 0.05

It is therefore necessary to add 0.05 to the resulting turning proportions (0.45 and 0.30). This is carried out proportionally for each one:

$$\begin{aligned}
 1 \rightarrow 2 & \quad 0.45 + (0.45/0.75) \cdot 0.05 = 0.45 + 0.03 = 0.48 \\
 1 \rightarrow 3 & \quad 0.30 + (0.30/0.75) \cdot 0.05 = 0.30 + 0.02 = 0.32 \\
 & \quad \text{Sum} = 0.80
 \end{aligned}$$

It is important to realise that only affected turns (for which increases or decreases have been defined in the action) are taken into account in the calculations. Other turnings remain the same.

Example 2

Consider the same original section and turnings as in Example 1:

1 → 2	0.30
1 → 3	0.50
1 → 4	0.20

Let us now assume an action defined as follows:

1 → 2	+ 50%
-------	-------

When this action is applied the following calculations are carried out:

1. Calculate the modifications to the percentages in the affected turns

$$1 \rightarrow 2 \quad 0.30 + (0.30 * 50\%) = 0.30 + 0.15 = 0.45$$

2. Adjust the resulting percentages in order to keep the same sum of proportions for the affected turns as the original:

	<u>Original</u>	<u>New</u>	
1 → 2	0.30	0.45	
	-----	-----	
	0.30	0.45	=> There is a difference of 0.15

We now have to subtract 0.15 from the resulting turning proportion (0.45). This is carried out proportionally for each of the affected turns. (In this case there is only one.):

$$1 \rightarrow 2 \quad 0.45 - (0.45/0.45) * 0.15 = 0.45 - 0.15 = 0.30$$

Sum = 0.30

Consequently, there has been no change in the turning proportion as a result of this action. This is because we have not defined from what alternative turn to subtract any proportion.

Example 3

Consider the same original section and turnings as in the previous examples:

1 → 2	0.30
1 → 3	0.50
1 → 4	0.20

And now suppose that we have defined an Action as follows:

1 → 2	+ 50%
1 → 3	+ 0%

When this action is applied the following calculations are carried out:

1. Calculate the modifications to the percentages in the affected turns

$$1 \rightarrow 2 \quad 0.30 + (0.30 * 50\%) = 0.30 + 0.15 = 0.45$$

$$1 \rightarrow 3 \quad 0.50 + (0.50 * 0\%) = 0.50 + 0 = 0.50$$

2. Adjust the resulting percentages in order to keep the same sum of proportions for the affected turns as the original:

	<u>Original</u>	<u>New</u>	
1 → 2	0.30	0.45	
1 → 3	0.50	0.50	
	-----	-----	

0.80 0.95 => There is a difference of 0.15

It is therefore necessary to subtract 0.15 from the resulting turning proportions (0.45 and 0.50). This is carried out proportionally for each one:

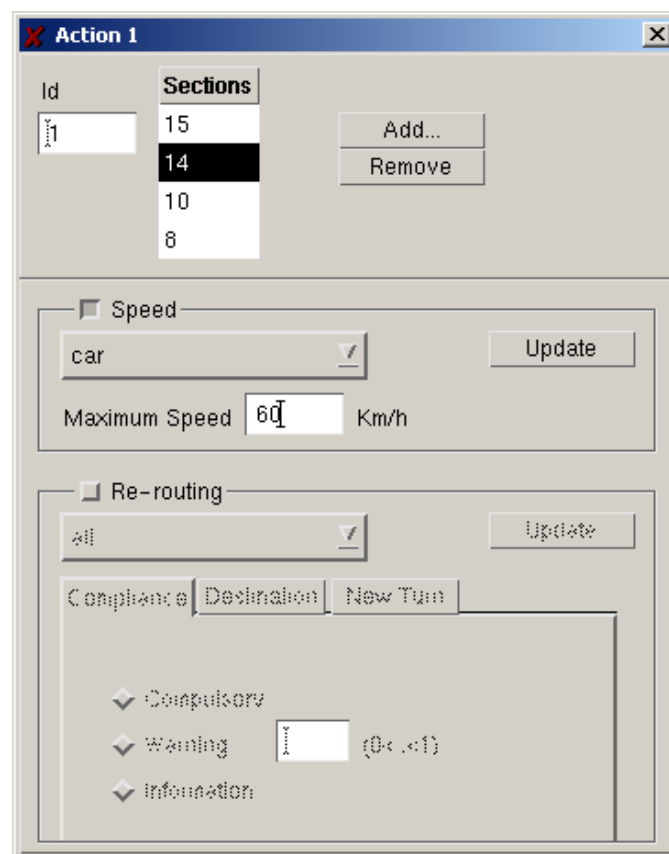
$$\begin{aligned}
 1 \rightarrow 2 & \quad 0.45 - (0.45/0.95)*0.15 = 0.45 - 0.07 = 0.38 \\
 1 \rightarrow 3 & \quad 0.50 - (0.50/0.95)*0.15 = 0.50 - 0.08 = 0.42 \\
 & \quad \text{Sum} = 0.80
 \end{aligned}$$

Examples 2 and 3 show that we need to define which turns are going to be taken into account in the transfer of flow from one to the other. If only one turn is considered (as in example 2) there is no possibility of moving flow to or from another turn. On the other hand, when two turns are considered in the action (although one of them has 0% increase) then the required increase for one turn can be taken from the other.

5.4.3 Actions for a Route-Based Simulation

When the simulation is Route-Based (using O/D matrices and route choice models) the 'Action' definition window is displayed (see Figure 5-21). This dialog window consists of three areas. The first area contains the list of sections impacted by the action. The second area is for defining the speed limit modifications, and the third area, at the bottom, is for defining re-routing actions. The re-routing area contains an option menu for selecting vehicle types and three tab folders: Compliance, Destination and New Turn.

Figure 5-21: Action-Speed Information Dialog Window (Route-Based)



The user can add sections to the list using the 'Add' button and then clicking on the section in the network display. To remove sections, just select the section from the list box and press the 'Remove' button.

Two types of actions can be defined: modifications of the speed limit, and re-routing actions, which can either be modifications to the destination centroid or modifications to the next turn to take place.

Modifications to the Speed Limit

Click on the Speed toggle button (see Figure 5-21) and type the new speed limit for that set of sections in the 'Maximum Speed' field. When a message containing this action is displayed in a VMS, it will automatically modify the Speed limit for the selected sections and therefore all vehicles driving along these sections will be affected.

Re-routing Actions

Re-routing means altering the vehicle's path. This is accomplished by defining the next turn and/or defining a new destination. To define re-routing actions click on the Re-routing toggle button (see Figure 5-22).

Re-routing can be defined for each vehicle type independently or for all types in the same way. The first thing to do is to define *Compliance* (δ). This parameter gives the compliance level of the action, i.e. the percentage of vehicles accepting the recommendation. To define it, select the Compliance tab folder (see Figure 5-22) and then select one of the options: Compulsory, Warning or Information.

Compulsory means $\delta=1$, which implies that the re-routing will be followed by everybody (i.e. it is compulsory). Information means $\delta=0$, where the action's success will depend on the driver's behaviour (Guidance acceptance λ , a vehicle attribute). In the Warning option the user can define δ ($0 < \delta < 1$), which is the level of acceptance, i.e. it is advice, not compulsory.

Figure 5-22: Actions-Re-routing Compliance Dialog Window (Route-Based)

The first type of re-routing action is the modification of the destination centroid. This is done by selecting the Destination tab folder from the Actions dialog window (see Figure 5-23). Click on the Destination toggle button and then select a set of pairs. The Previous destination centroid and the New destination centroid compose each pair. This means that vehicles going to a Previous destination will change to New destination as soon as they enter any of the sections affected by this action.

The second type of Re-routing action is modification of the next turning. This is done by selecting the New Turn tab folder from the Actions dialog window (see Figure 5-24). Click on the New Turn toggle button and type in the new Next Section. Then choose between All or Selected Destinations. All destinations means that

all vehicles entering the affected sections will take as the next turning the one specified independently of its destination. Selected destinations means that only vehicles going to those destinations will be affected by the new turn. In both situations the original destination may be lost if there is no path through the specified next turn.

If the conflicting re-routing actions are defined, the user must be aware that actions defined in the Destination folder are applied first, before those defined in the New Turn tab folder. By conflicting actions, we mean actions that are addressed to the same set of drivers. For instance, we may define an action for a section that changes the destination of drivers going to centroid 1, re-routing them to centroid 7. We may then define an action for the same section that changes the next turning for all vehicles going to any destination, to turn to section 25. When the action is activated, first the destination centroid is changed, and then the next turn is assigned. It is possible that there is no path from section 25 to destination 7 and some vehicles may get lost as a result of inconsistent actions.

Figure 5-23: Actions-Re-routing Destination Dialog Window (Route-Based)

Action 1

Id: 1

Sections: 15, 14 (selected), 10, 8

Buttons: Add..., Remove

Speed

car

Maximum Speed: 60 Km/h

Update

Re-routing

all

Update

Destination

Compliance | **Destination** | New Turn

Previous	New
1	7
6	8
8	9

Text boxes: ... 8 9 ...

Figure 5-24: Actions-Re-routing New Turn Dialog Window (Route-Based)

Action 1

Id: 1

Sections: 15, 14, 10, 8

Add... Remove

☐ Speed

car / Update

Maximum Speed: 60 Km/h

☐ Re-routing

all / Update

Compliance Destination New Turn

☐ New Turn

Next Section: 10 ...

Select

Destinations: 2

6. PUBLIC TRANSPORT MODELLING

The main difference between public transport vehicles and other road traffic is that they follow fixed routes and that they also try to adhere to a pre-defined timetable. Public Transport Modelling is applied in AIMSUN, whenever a Public Transport Plan is loaded. A Public Transport Plan in AIMSUN consists of Public Transport Lines (routes, reserved lanes and bus stops), the Timetables for each line (departures schedules, stop times and type of vehicles) and the Public Transport Vehicles (buses, minibus, trams, etc.).

6.1 PUBLIC TRANSPORT LINE

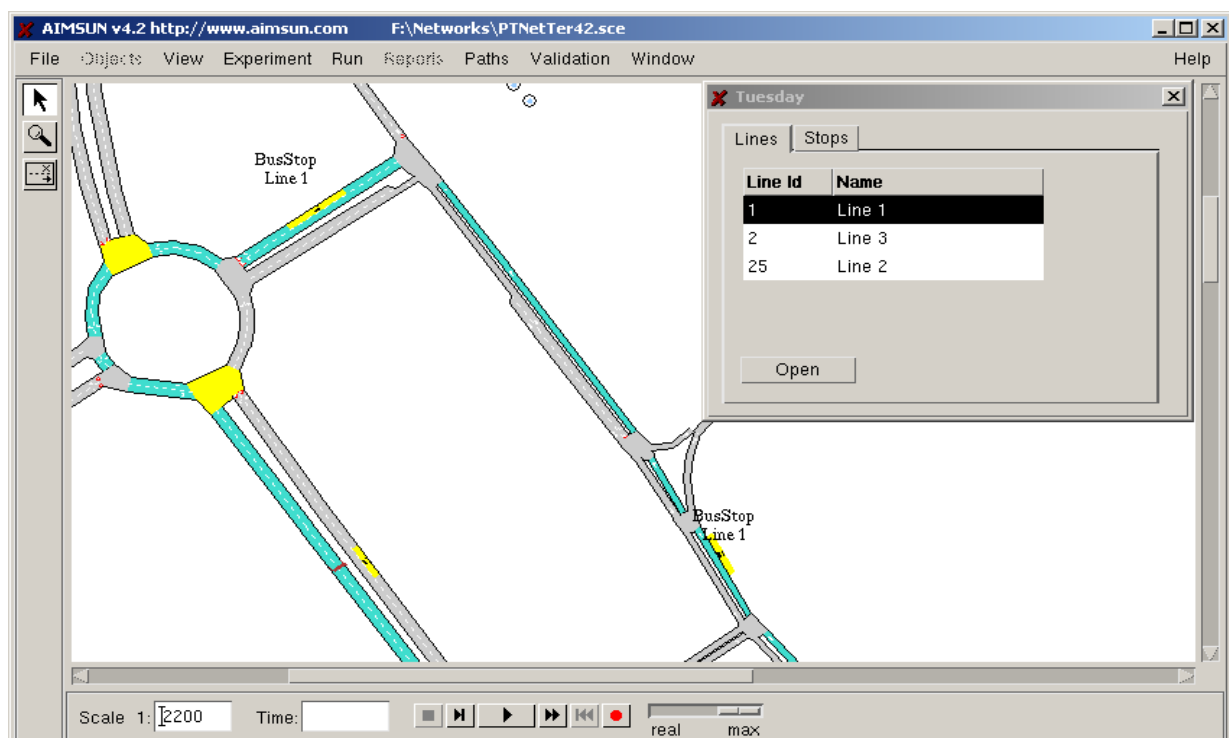
A Public Transport (PT) Line is composed of a route and a set of public transport stops. Notice that PT Lines does not appear in the simulation model unless a Public Transport Plan is loaded in AIMSUN. These objects are explained in the following sections.

6.1.1 Public Transport Route

Public transport Routes are a fixed series consecutive sections (or polysections) through which each PT vehicle of the line will have to pass, from an origin to a destination. The first section of the PT route is the origin, where the PT vehicles are input into the network. The frequency of departures is defined in the Timetable. The last section of the PT route is the destination, where the vehicles are removed from the network.

A network can contain as many PT Lines as required. To view the list of all the existing bus lines, use the 'Objects/Public Transport' command, and the Public Transport dialog window displayed in Figure 6-1 will appear. Clicking on a line name in the Lines list box will cause the sections composing the line to be highlighted in a different colour in the network.

Figure 6-1: Public Transport Dialog Window

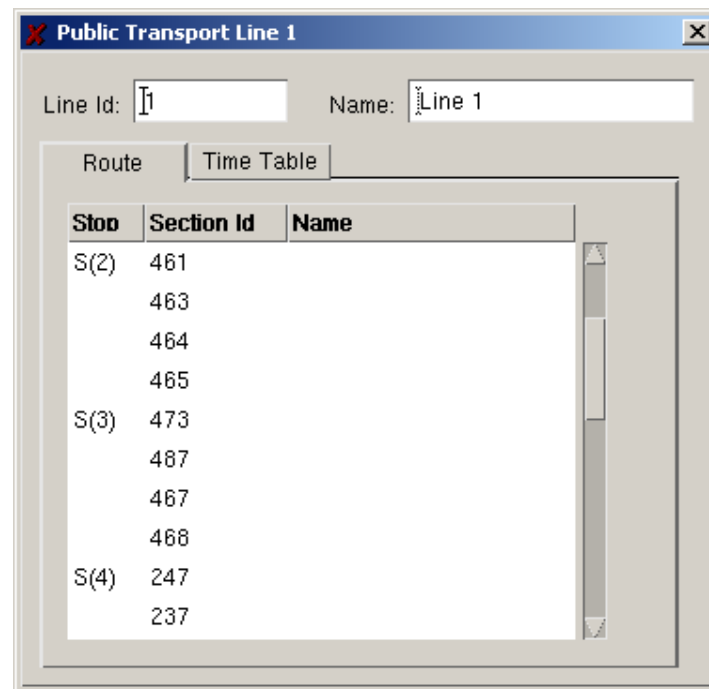


The Public Transport dialog window contains two columns:

- Line Id.: identifier number of the bus line.
- Name: name of the bus line

After selecting a line, click on the 'Open' button or simply double-click on the lines list box and the Public Transport Line dialog window displayed in Figure 6-2 will appear. This dialog window contains the list of sections comprising the route.

Figure 6-2: Public Transport Line Dialog Window



The PT Line definition is completed with the PT stops. Each route may have a number of PT stops, at which the PT vehicles will have to stop along their trip for a certain amount of time. The stop times are defined in the Timetable. PT lines with no PT stop are also accepted and in that case PT vehicles will not stop at any PT stop during their trip. The Public Transport Line dialog window displayed in Figure 6-2 also indicates in which sections PT stops are allocated to the line. For instance, in figure 6-2, PT Line 1 has stops numbers 2, 3 and 4 respectively located in sections 461, 473 and 247.

6.1.2 Public Transport Stops

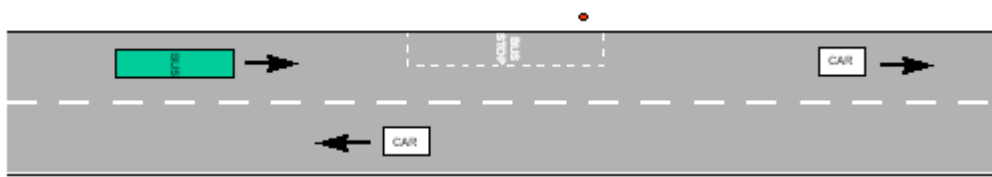
Public Transport Stops (or bus stops) are the points along the Public Transport Line at which public transport vehicles must stop for a certain amount of time in order to pick up or drop off passengers. Each PT stop belongs to only one section, while a section may have several PT stops. However, for a particular PT Line, no more than 1 PT stop can be located in the same section. Vehicles belonging to different bus lines can use the same PT stop.

Public Transport stops are common within road networks and they can influence the behaviour of traffic in the nearby. A critical element in the efficiency of public transport operations is also the behaviour of passengers boarding and alighting at stops. For that reason AIMSUN is able to model various types of PT stops that can be found in road networks throughout the world.

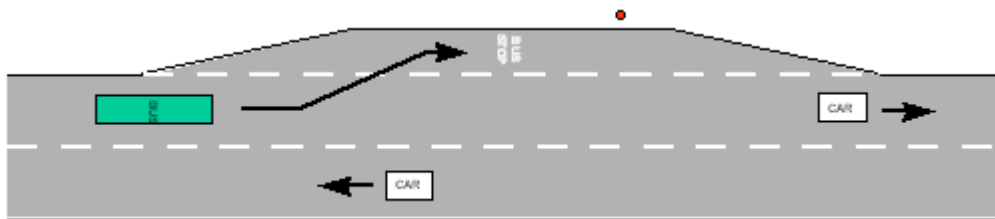
Three types of PT stops are considered in AIMSUN: Normal, Bus Bay and Terminal.

Normal Bus Stop

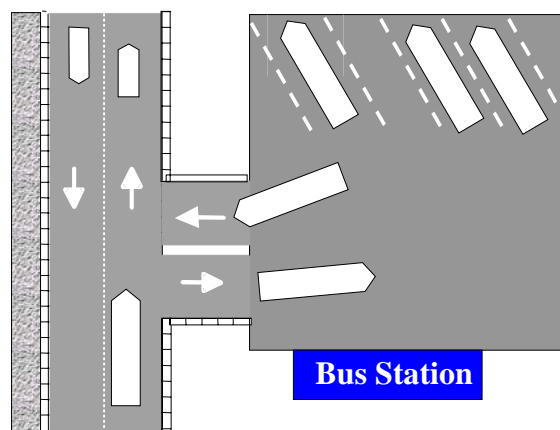
These are the simplest types of PT Stops. They are just indicated by a sign on the roadside and the public transport vehicle stops alongside it to pick up or put down passengers. Figure 6-3 shows the layout of this type of PT stop.

Figure 6-3: Normal Bus Stop**Bus Bay Stop**

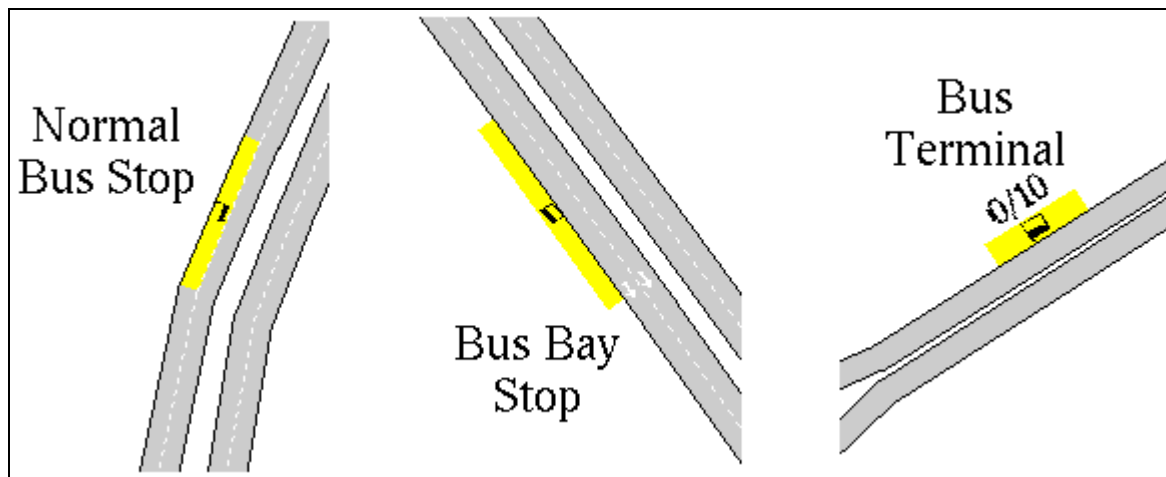
The Bus Bay stop provides a space for a bus to pull into at the stop and thus allows following traffic to pass the bus after it stops. The capacity of the bus bay stop depends on the length of the bay (the short lane) and the length of the buses that stop there. Figure 6-4 illustrates a bus bay stop.

Figure 6-4: Bus Bay Stop**Bus Terminal Stop**

Finally, the bus terminal stop is used to model bus stations or bus parking. They have room for a limited number of buses, which is a capacity defined as an attribute of the stop. A picture of what is considered as a Bus Terminal is shown in Figure 6-5.

Figure 6-5: Bus Terminal Stop

The graphical representation of the different types of Public Transport Stops considered in AIMSUN is shown in figure 6-6. To obtain basic information about a particular PT Stops, double-click on the bus stop icon in the network display and the Bus Stop dialog window shown in Figure 6-7 will appear. It contains information such as the type, position and length of the bus stop.

Figure 6-6: Public Transport Stops in AIMSUN**Figure 6-7: Bus Stop Dialog Window**

Id:	1
Name:	Main Stop
Type:	Terminal
Section:	431
Lane:	1
Position:	26.139 m
Length:	22.209 m
Distance to Stop:	50.000 m

On the other hand, a list of all stops in the network is displayed in the 'Stops' tab folder of the Public Transport dialog window, which appears when you select the 'Objects/Public Transport' command (see Figure 6-8). For each PT stop, the following data is provided:

- PT Stop Identifier number
- PT Stop name
- Type of PT Stop
- Identifier of the section where the stop is located
- Lane of the section where the stop is located (numbered from 1, the rightmost to n, the leftmost)
- Position of the stop in meters, measured from the beginning of the section
- Length of the stop in meters
- Distance to Stop parameter (see section 6.3.2)

Clicking on a Stop name in the list box causes the Identifiers of the lines whose PT vehicles have to stop in this stop to be displayed in another list box. In the example of Figure 6-8, lines 25 and 1 are supposed to stop at Bus Bay Stop number 27, which is located in section 182. Clicking on the 'Find' button causes the display to pan to the selected PT Stop.

Figure 6-8: Public Transport Stops Details

Stop Id	Name	Type	Section	Lane	Position	Length	Line Id
4		Normal	247	1	26.25	28.50	25
3		Bay	473	1	19.89	20.28	1
28		Normal	424	1	81.14	16.65	
2		Terminal	461	1	32.94	25.92	
27		Bay	182	1	110.47	34.58	
1	Main Stop	Terminal	431	1	26.14	22.21	
26		Terminal	211	1	159.93	13.39	

6.1.3 Reserved Public Transport Lanes

Reserved PT Lanes can be optionally defined along a PT Route. A reserved PT lane is an area of carriageway reserved for the use of PT Vehicles and occasionally other permitted vehicles, for all or part of the day. They allow buses to bypass traffic queues, usually on approaches to signalised junctions or roundabouts. The location of the start and finish of bus lanes within a link are crucial.

Reserved Lanes have been previously described in section 2.3.3. To define a reserved lane for PT vehicles only, it is required that the PT vehicles belong to the class allowed in the reserved lane and it is advised that the reserved lane is defined as 'Compulsory'.

6.2 TIMETABLES

The passage of public transport vehicles along a route is usually governed by a timetable. The bus starts at its origin at a given time and is expected to arrive at stops along the route at predicted times. This allows passengers along the route to know when they should arrive at the stop in order to catch the vehicle they require.

Timetable data is presented in the 'Time Table' tab folder of the Public Transport Line dialog window shown in Figure 6-9. This dialog window is called either by clicking on the Line icon in the network or via the 'Objects/Public Transport' command.

A Timetable consists of a set of Time Slices, each one indicating the Bus Departures Schedule and the Stop Times at each bus stop allocated to the line.

Figure 6-9: Public Transport Line. Timetables.

Stop Id	Mean	Deviation
1	40.00	5.00
2	30.00	5.00
3	40.00	0.00

The Bus Departures Schedule can be defined in terms of Frequency of Departure or by a set of Fixed Departure Times. In both cases the user can define a Deviation value, which will be used as standard deviation to sample the PT vehicles departures from a Normal Distribution.

Finally, for each PT Stop along the route, the mean stop time and deviation are defined. The time that the PT vehicle will be stopped at each PT stop is also sampled from a Normal distribution. A passenger generation model is not implemented. Only the waiting time of public transport vehicles at stops is used to model the passengers boarding and alighting times. The user can model different passengers rates just by ensuring that public transport vehicles stop for a suitable length of time at each stop.

6.3 PUBLIC TRANSPORT VEHICLE'S MODELING

Buses provide the most common form of public transport in urban areas, but other type of vehicles can be considered, as guided buses, trolley buses, trams or Light Rapid Transit (LRT). They come in many different sizes and shapes. Public Transport Vehicles are defined as any other vehicle type. Therefore the vehicle attributes used in the vehicle modelling are the ones already described in section 3.4. Typical parameters for some types of bus are given in the Table 6-1.

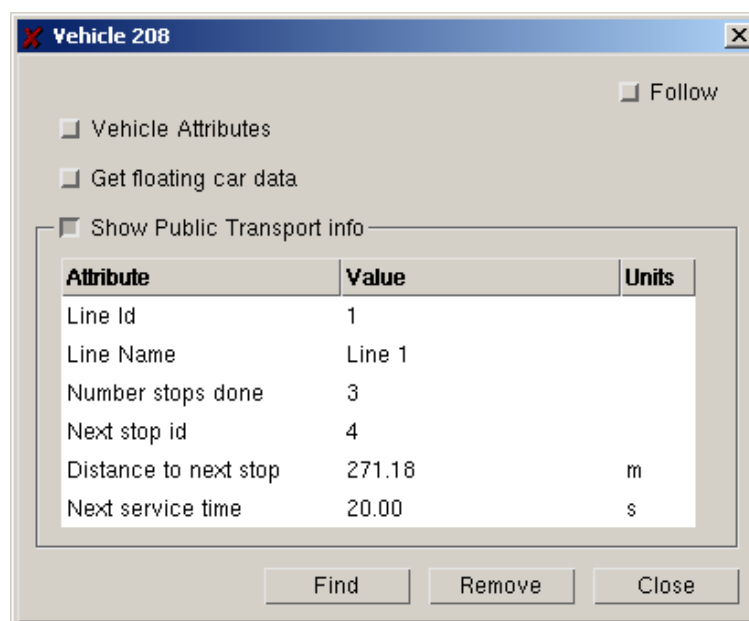
Table 6-1: Typical parameters for PT Vehicles (Source: US DOT, 1992)

	<i>Length (m)</i>	<i>Width (m)</i>	<i>Acceleration rate (m/s/s)</i>	<i>Deceleration rate (m/s/s)</i>
Buses				
Articulated	18.5	3.0	1.0	1.2
Double deck	10.0	4.2		
Single deck	8.5	3.4		
Minibus	7.0	2.4		
Trams				
Single unit	15.0	2.6	1.5	1.6
Articulated	22.0	2.7	1.3	1.6

The user can obtain detailed information on a particular PT vehicle just by double clicking on it while the simulation run has been halted. The Vehicle dialog window will then appear and the vehicle will be highlighted in a different colour (see Figure 6-10). Apart from the general Vehicle Attribute data and the floating car data, which are described later in section 8.2.3, the following Public Transport related information is provided:

- PT Line Identifier and Name of the vehicle.
- Number of stops that the vehicle has made since departure.
- Identifier of the next stop
- Distance to the next stop (in meters).
- Next expected stop time (in seconds).

Figure 6-10: Public Transport Vehicle Information.



Now, if the model is run after opening the Vehicle dialog window, the dynamic information contained in this dialog window is continuously updated while the simulation runs until the vehicle exits the network.

6.3.1 Vehicle Generation Model

Public Transport Vehicles are generated and input into the network via the first section of each bus line and they drive along the network following the bus line and making the corresponding stops. The arrival times are obtained according to the corresponding bus schedule, defined in the Timetable.

If there is a bus stop type Terminal in the first section of a bus line, buses will start their trip at this bus stop. Otherwise, they start their trip at the beginning of the first section of the line. Similarly, if there is a bus stop type Terminal in the last section of a bus line, buses will end their trip at this bus stop. Otherwise they will finish their trip at the end of the last section of the line.

6.3.2 Vehicle Movement Model

Public Transport vehicles follow a fixed route through the network. PT Vehicles normally behave as any other vehicle according to car-following and lane-changing models. When driving in a section (or polysection) containing a bus stop allocated to that line, they change their normal behaviour and act accordingly in order to stop in the appropriated place. This means that they will try to move to the appropriate lane before reaching the bus stop and they will decelerate before coming to a halt at the bus stop. The vehicle movement model:

- ensures that PT vehicles move into the lane to access the stop at a suitable time,
- makes PT vehicles pull into a bus bay or move out of the road if required to stop at a bus terminal stop,
- ensures that PT vehicles are able to exit from a bus bay either by waiting for a suitable gap in the traffic or by influencing the other traffic to give way to let the vehicle exit.

The Distance to Stop parameter governs the lane changing behaviour with respect to the Bus Stop. When the distance from the bus to the next stop is lower than Distance to Stop, the bus behaves as normal vehicles do in zone 3, it means that they are stressed to reach the appropriated stopping lane.

Reserved lanes in a network can be designated for use only by public transport. When such a lane is available in any section of the PT line, the PT vehicles will use it. In any case, turning movements and PT stops have more priority than reserved lanes. It means that a PT vehicle will abandon a reserved lane whenever it needs to reach an appropriated turning lane or make a particular PT stop.

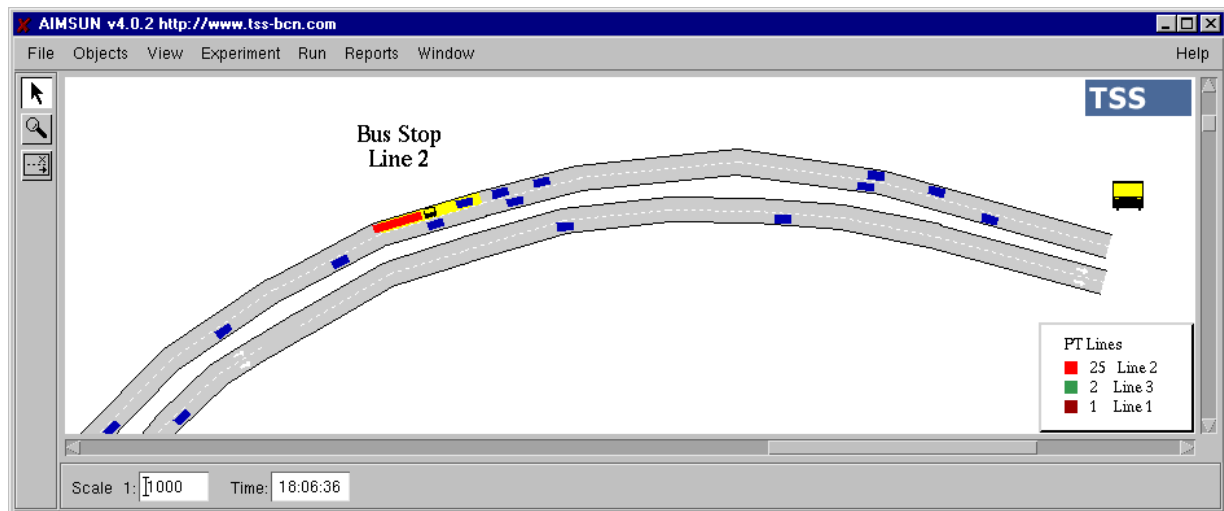
6.3.3 Stop Model

Public transport vehicles stop at fixed points along the route to pick up and set down passengers. The amount of time spent at the stop is determined by the Stop Time parameter. The stop time for a particular PT vehicle is sampled from a Normal distribution using the mean stop time and deviation.

The movement of the public transport vehicle in the vicinity of the stop is modelled. If the bus stop is full when a new bus arrives, i.e. another bus is already there and there is no room for additional buses, the bus will wait until that bus leaves and there is enough space available. A bus will spend a certain amount of time (defined in the timetable), stopped at the bus stop, and then it will start off again and continue its route following the bus line until the next bus stop.

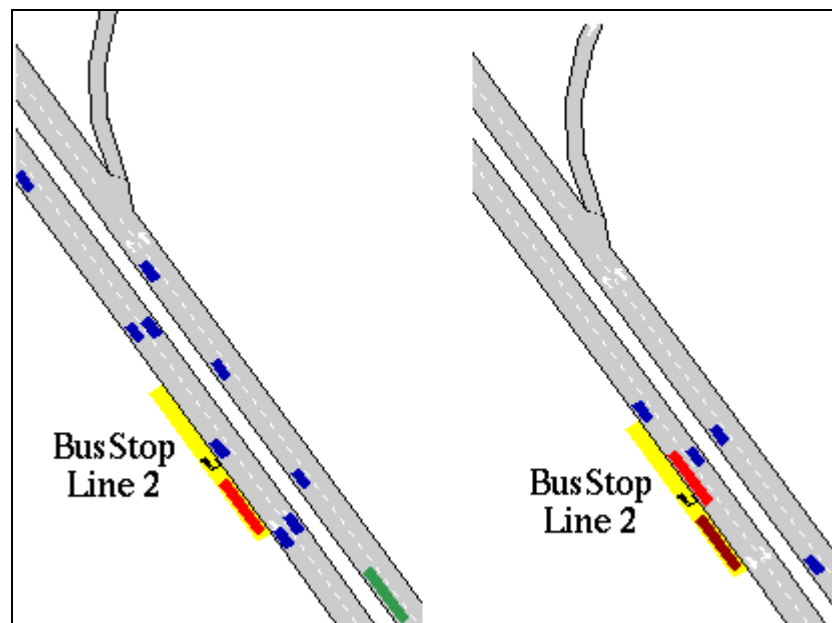
Normal Bus Stop

This is the type of bus stop that is located along the roadside. PT vehicles will stop on the street lane at the end of the bus stop, thus blocking the traffic in that lane (see figure 6-11). Other vehicles will overtake the PT vehicle (if there is more than one lane). The PT vehicle will be stopped during the stop time and then will normally start accelerating using the same lane, unless it needs to change lanes because of turning feasibility reasons or presence of reserved lanes in that section.

Figure 6-11: Stopping at a Normal Bus Stop**Bus Bay Stop**

This type of bus stop has a special short lane at the roadside for the bus to move out of the traffic stream, thus allowing following traffic to pass the bus after it stops. PT vehicles will stop on the street lane at the end of the bus stop, and then will pull into the bus bay, thus blocking the traffic in that lane but only during the lane changing manoeuvre (see figure 6-12). The PT vehicle will be stopped during the corresponding stop time and then will wait for a suitable gap in the traffic to go out from the bus bay. It applies a lane-changing model that can even influence the other traffic to give way to let the vehicle exit if necessary.

If a second PT vehicle arrives at the bus bay before the previous one has left, it will stop on the street lane just behind the first PT vehicle and then, if there is space available at the bus stop it will pull into the bus bay. Otherwise it will wait until the previous vehicle leaves the bus bay, thus blocking the traffic in that lane during the time that is waiting to enter.

Figure 6-12: Stopping at a Bus Bay Stop

7. SIMULATION EXPERIMENTS

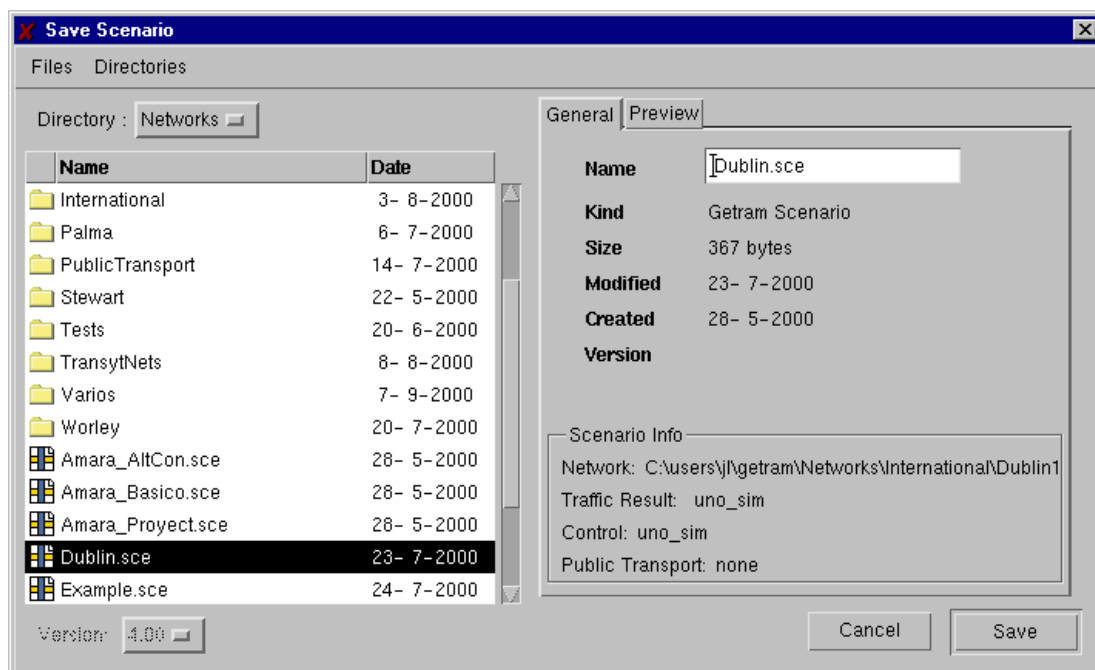
This chapter provides a detailed description of all the functions included in the AIMSUN user interface that are required for the preparation and running of simulation experiments. It goes through all the commands available in the menu bar, explaining how they work and providing all the related topics in order to provide an in-depth view of the tool.

7.1 LOADING SCENARIOS

An AIMSUN Scenario consists of up to five types of data: the network description, the traffic demand data, the traffic control plan, the public transport plan and the GETRAM Extensions. The network description contains the topology of the network, basically composed of sections and nodes. The traffic demand data can be defined in two different ways: 1) the traffic volumes at the input sections, the turning proportions at any node and, optionally the initial state of the network; 2) an O/D matrix that provides the traffic demand from every origin to every destination centroid. These are exclusive, so only one option can be selected. The traffic control plan consists of the description of signal groups, phases and their duration, for signal-controlled junctions, the priority definition for unsignalised junctions and any required ramp-metering information. A public transport plan consists of the definition of bus lines (routes and bus stops), and the timetables for each line, including stop times. Finally, the GETRAM Extensions are a set of user-defined Dynamic Link Libraries that will be run in the simulation experiment.

Some of the components of the scenario are compulsory, such as the network description and the traffic demand data, while others are optional. Therefore, loading an AIMSUN scenario consists of loading some of these five types of data. The File menu brings together a set of commands that allow you to load and unload each of the components of a scenario separately. Once the different components of a scenario have been loaded separately, the user may save this set of components as a new scenario using the command 'File/Save Scenario' or using the shortcut <Ctrl><S>. Using the Save Scenario dialog window (see Figure 7-1), the user can specify a particular name for the scenario that may be used for future simulation experiments.

Figure 7-1: Load/Save Scenario window



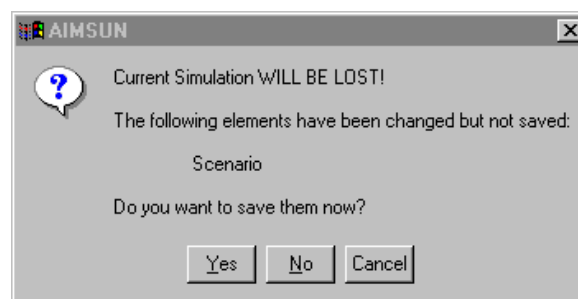
Scenarios are saved in ASCII files with the extension '.sce'. These files contain the components included in the scenario: network name, traffic result or O/D matrix name, traffic control plan, public transport plan and names of GETRAM Extension DLL's. The names of the components of a scenario are displayed in the General tab folder of the Load/Save Scenario window when a scenario is selected in the scenarios list box.

When including a traffic result in a scenario, the complete list of traffic states, the Load options (Initial State, Initial Messages and Incidents) and the Arrivals distribution selected are also included in the scenario definition. When including an O/D Matrix in a scenario, the Load options (Initial Messages and Incidents) and the Arrivals distribution selected are also included in the scenario definition.

Saved scenarios can be directly loaded using the 'File/Load Scenario' command, or using the shortcut <Ctrl><O>. In the Load Scenario dialog window, which is similar to the Save Scenario dialog window displayed in Figure 7-1, the user can browse to search for a particular scenario file. In this way the user does not need to define all components of a scenario every time a new simulation experiment has to be started.

To unload the scenario, use the command 'File/Unload Scenario' from the menu bar. Should the originally loaded scenario be modified and/or a simulation run started, a confirmation message (see Figure 7-2) will appear. If you press the 'Yes' button, or just press 'Enter', the scenario will be saved. If you press 'No', the scenario will not be saved. In both cases the network will be removed from the display window and all the components of the current scenario will be cleared from the system memory. If you press 'Cancel', nothing will happen.

Figure 7-2: Warning Message - Unload Scenario



7.1.1 Loading the Network

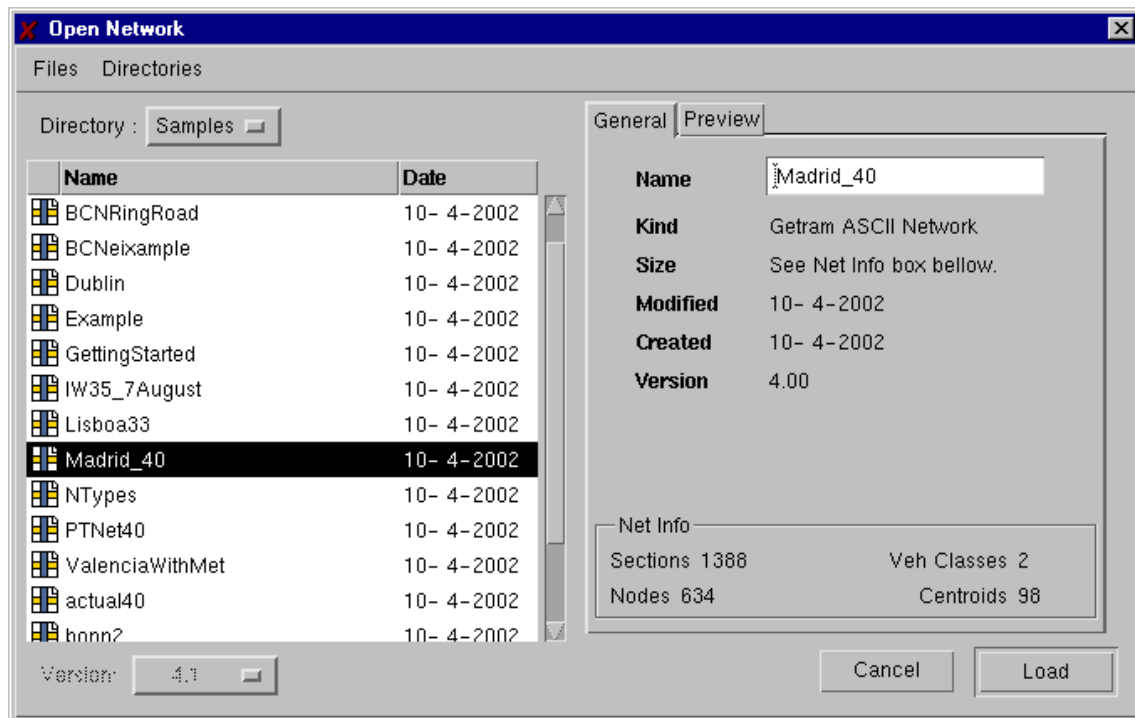
The first component of an AIMSUN scenario is the network description, which contains information about the geometry of the network, turning movements, layout of sections and junctions and the location of infrastructure objects throughout the network.

To load the network, select the command 'File/Load Network' from the menu bar. This can also be done by pressing <Ctrl><N>. The 'Open Network' window will appear (see Figure 7-3). Using this window the user can search for the directory containing the network.

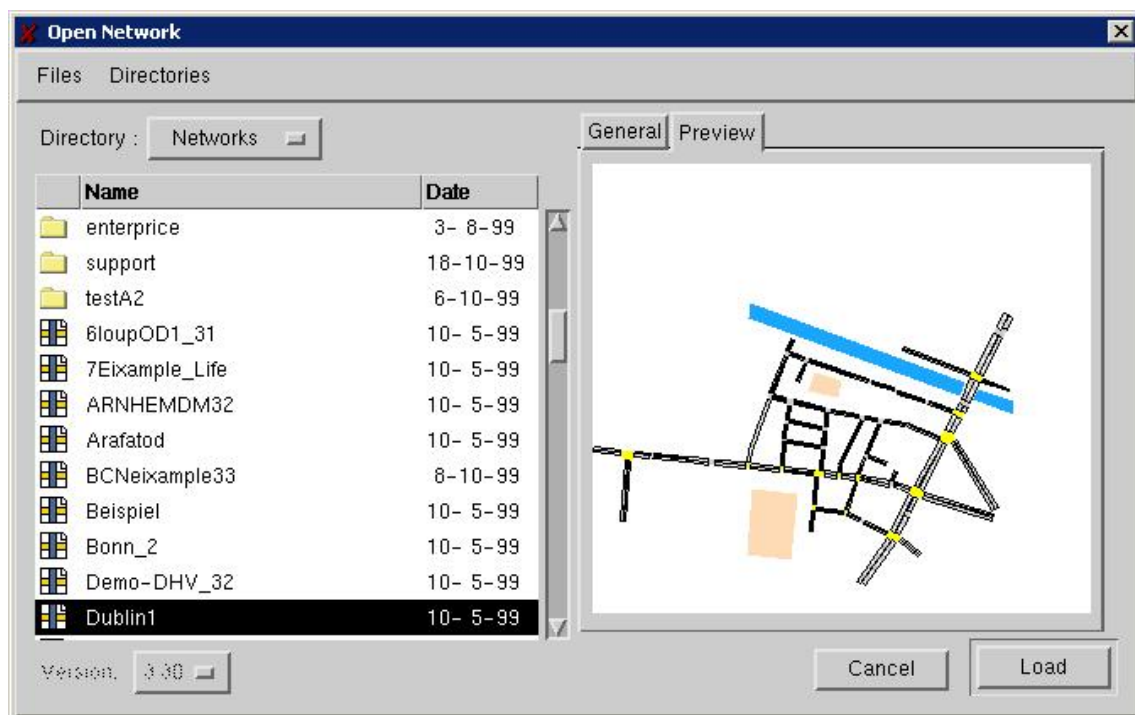
The 'Open Network' window has two areas. The left part of the window is for searching and selecting networks, while the two tab folders on the right contain information about the selected network. There is a menu bar situated at the top-left corner of the window, containing two commands: Files and Directories. Clicking on them will display a pull-down menu containing the latest networks and directories used respectively. There is also an option menu called 'Directory', located at the top-left corner of the window, via which the user can browse through the directory tree or even change to other disk drives.

The list box on the left displays the contents of the selected directory. Only directories and GETRAM networks are listed. The Type column identifies them by using a folder icon for directories and a file icon for networks. This list box also displays the latest modification date for the networks and directories.

Selecting a directory name from the list and clicking on the 'Open' button takes the user through that directory. Clicking on a network name and then clicking on the Load button will load the network. Double-clicking on the directory or network name has the same effect. The graphical representation of the network will appear in the network display area, while the name of the network is displayed in the information area.

Figure 7-3: Open network window

After selecting a network from the list box by clicking on it, select the tab folder 'General' to view the version and the size of the network, in terms of number of sections, nodes, etc. The user can also take a look at the network before loading it into AIMSUN if desired. Select the tab folder 'Preview' from the 'Open Network' window and a network preview will be displayed (see Figure 7-4).

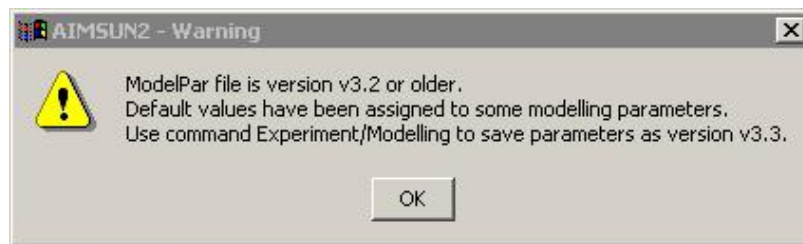
Figure 7-4: Open network preview

The 'File/Load Network' command loads the Network into the model, which means not only reading all the network information from the GETRAM Database, but also running the Pre-Simulation Process, which builds up the required structures for the AIMSUN simulation model (i.e. network decomposition into

entities). For this reason, loading big networks may take a certain time, depending on computer power and memory.

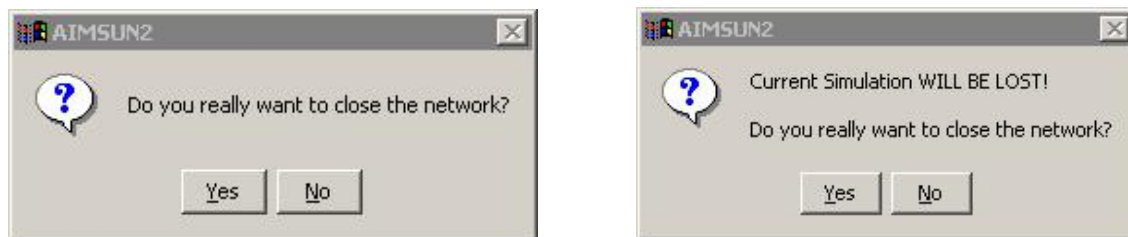
Because of the differences between v3.2 and v4.0 parameters, the warning message in Figure 7-5 may appear when loading a v3.2 network. This means that some Global Modelling Parameters (see section 3.4.3), may not have been defined and therefore they have been assigned using default values. To view or edit these parameters, use the Modelling dialog window by selecting the 'Experiment / Modelling' command. In order to avoid this warning message when loading this network in future, you must save the parameters using the 'Save' button in the Modelling dialog window.

Figure 7-5: Warning Message



To unload the network, use the 'File/Unload Network' command from the menu bar. One of the confirmation messages shown in Figure 7-6 will appear, depending on whether a simulation has been run or not. If you press the OK button, or just press 'Enter', the network will be removed from the display window and the simulation model of the current network will be cleared from the system memory.

Figure 7-6: Confirmation Messages



In order to avoid loss of data, a warning dialog window may appear when the user attempts to unload the network (see Figure 7-7). This window is only shown when modifications have been made to the current model and have not been saved. The modifications covered by this warning are: editing of new Streams for statistical data gathering (see section 9.1), creation of traffic incidents (section 3.6), definition of new Actions for Messages (section 5.4), and specification of Messages to be displayed on VMS's (section 2.5.3). The list of elements that have been modified but not saved is displayed in the warning dialog window.

Figure 7-7: Warning message (modified elements)



7.1.2 Loading the Traffic Demand Data

The second step in the process of loading a model is to load the traffic demand data to be used to feed the network with vehicles. This data can be defined either by the input flows and turning proportions (called

Traffic Result) or by an O-D matrix. They are exclusive, so only one option may be selected for a given simulation run, not both. Traffic Demand Data can only be loaded after a network has already been loaded.

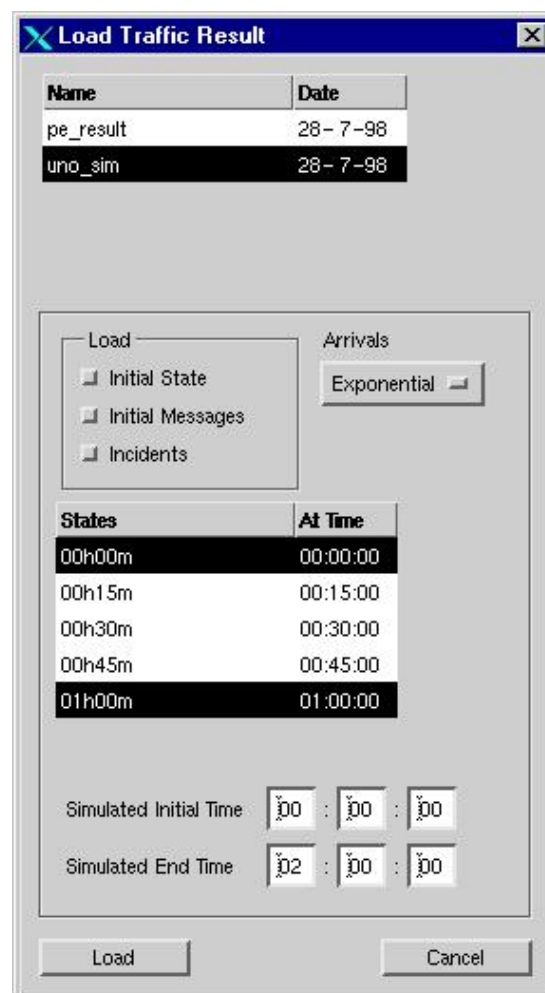
7.1.2.1 Loading a Traffic Result

To load a Traffic Result, select the 'File / Load Traffic Result' command from the menu bar or use the shortcut <Ctrl><T> and the 'Load Traffic Result' window will appear, as shown in Figure 7-8.

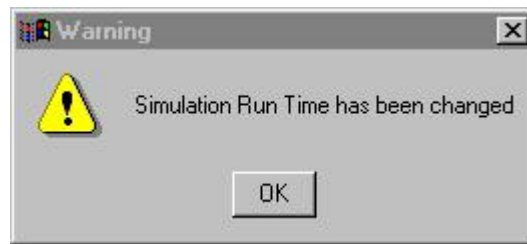
A network may have different Traffic Results. In the upper part of the window, a list of the results available for the current network is displayed in a list box. It contains the name and the date of creation or latest modification for each result. Select one of them by clicking on it and the list of states that are included in this result will appear below.

Each state corresponds to a certain time interval. Therefore the 'States' list box contains the name of the state and the time when this state begins. No states are selected by default. The user can select a set of states to be loaded by clicking on them in the 'States' list box. The user can also deselect them by clicking on them again. The selected states will be highlighted.

Figure 7-8: Load Traffic Result window



The Simulated Initial Time and Simulated End Time are displayed at the bottom of the window. By default, they are assigned according to the Run Time parameters of this network (see Section 7.2.2). When the user selects states, the Simulated Initial and End Times are updated accordingly. For example, if the 'At Time' of the first state selected is lower than the Simulated Initial Time, it will be reassigned to this value automatically. Alternatively, if the 'At Time' of the last state selected is greater than the Simulated End Time, that too will be reassigned. If this happens, the warning message displayed in Figure 7-9 appears. The user clicks on 'OK' button to continue.

Figure 7-9: Warning message for reassignment of Simulation Run Times

The system proposes the lowest 'At Time' for all the states selected as Simulated Initial Time. On the other hand, the Simulated End Time is taken as the maximum between the default Simulated End Time and the highest 'At Time' of the states selected. If the Simulated End Time is equal to the 'At Time' of the last selected state, a time interval is added to the Simulated End Time (when only one state is selected, an interval of one hour is taken as default).

The 'Simulated Time' dialog windows in the 'Load Traffic Result' window cannot be edited. The 'Experiment/Run Time' command (see section 7.2.2) must be used to modify any of these values.

Not all the states whose 'At Time' falls between those two values will be loaded, only the highlighted ones. For example, in Figure 7.8, although simulation goes from time 00:00:00 to time 2:00:00, only two states, 00h00m00 and 01h00m00 will be loaded for simulation. This means that the simulation will use the demand traffic data of state 00h00m00 since time 00:00:00 until time 01:00:00 and then it will take data from state 01h00m00 until the end of simulation at time 2:00:00.

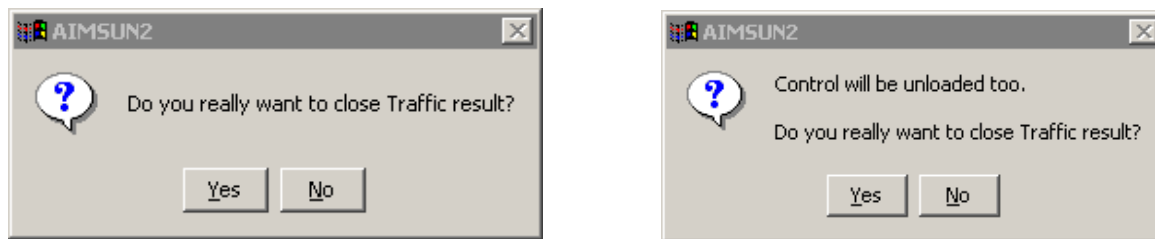
There are also three toggle buttons in the 'Load Traffic Result' window: 'Load Initial State', 'Load Initial Messages' and 'Load Incidents'. If the 'Load Initial State' toggle button is activated, not only the input flows and turning proportions will be taken from the selected states, but also the initial state of each section corresponding to the first state selected. This initial state is composed of the number of vehicles travelling along each section, their mean speed, and the number of vehicles at a standstill. Loading the Initial State of the network allows the user to start a simulation run from a previously saved simulation or initially fill the network up with vehicles without having to define and run a warm-up period.

The 'Load Initial Messages' toggle button triggers the loading of the Messages Log File and the initialisation of Variable Message Signs using the activated messages, if any, as they were saved from a previous run (section 5.4 explains how VMS's, messages and actions work). The 'Load Incidents' toggle button triggers the loading of the Incidents Log file, which may have been created and saved in a previous simulation experiment (a more detailed description of modelling incidents can be found in section 3.6).

Finally, there is an option menu to select the headway distribution to be used to generate the vehicle's inter-arrival times, i.e. the time gaps between two consecutive arrivals in an input section. The user may select from constant, uniform, normal, exponential, ASAP and external distributions. The default distribution when loading a Traffic Result is exponential.

Once a result, a set of states and the initialisation options have been selected, click on the 'Load' button. The result and states selected will be loaded into the AIMSUN simulation model and the result name will appear in the 'Traffic' dialog window in the information area located at the bottom of the main window.

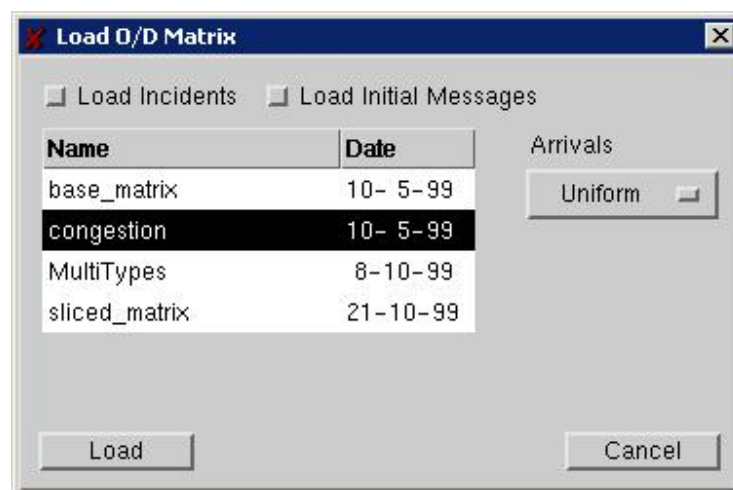
To unload the traffic result, use the 'File / Unload Traffic Result' command. If a Traffic Control Plan is also loaded, both the traffic result and the control plan will be unloaded, although a confirmation message will appear first (see Figure 7-10).

Figure 7-10: Unload Traffic Result Confirmation Dialog Windows

7.1.2.2 Loading an O/D Matrix

To load an O/D matrix, select the 'File / Load O/D Matrix' command from the menu bar or use the shortcut <Ctrl><M> and the 'Load O/D Matrix' window will appear (see Figure 7-11). As a network may have a number of O/D matrices, a list with the names and creation dates (or latest modification) for the available matrices for the current network is displayed. Select one of them by clicking on it.

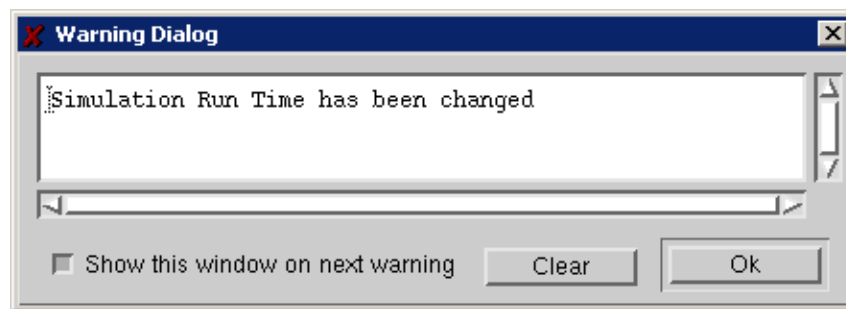
The 'Load Incidents' and 'Load Initial Messages' toggle buttons work in the same way as when loading a traffic result (explained in the previous section). No 'Load Initial State' toggle button appears in this window.

Figure 7-11: Load O/D Matrix window

An option menu is provided in which you can select the headway distribution to be used to generate the vehicle's inter-arrival times. The user may select from constant, uniform, normal, exponential, ASAP and external distributions. The default distribution when loading an O/D matrix is Uniform.

Finally, click on the 'Load' button to accept the selected O/D Matrix and it will be loaded into the AIMSUN simulation model. At the same time, Centroids information is loaded and the initial shortest paths from every section to every destination centroid are calculated. This may take a while depending on the size of the network and the number of centroids. Finally, the matrix name will be shown in the 'O/D Matrix' dialog window, which is the same 'Traffic' dialog window that becomes the 'O/D Matrix' dialog window. It is located in the information area at the bottom of the main window.

If the default 'Simulated Initial Time' and 'Simulated End Time' parameters of this network do not fit with the time scope of the O/D matrix loaded, they will be reassigned automatically, and the warning message shown in Figure 7-12 will be displayed. The user will have to click on the 'OK' button to continue. The Log Window is explained later in section 8.3.3.

Figure 7-12: Warning Message for reassignment of simulation run times

To unload the O/D matrix, use the 'File / Unload O/D Matrix' command. A confirmation message similar to those shown in Figure 7-10, will appear. If a Traffic Control Plan is also loaded, both the OD matrix and the control plan will be unloaded.

7.1.3 Loading a Traffic Control Plan

A traffic control plan is optional when running a simulation experiment. If traffic signals or ramp metering are defined in the network model, a Traffic Control Plan can be loaded. Otherwise it is not necessary. As discussed in chapter 5, the control plan basically consists of the Signal Group definition and the timings of every phase for all the controlled junctions and the timings or flows for all ramp meters.

To load a control plan, select the 'File / Load Control' command from the menu bar or use the shortcut <Ctrl><C> and the 'Load Control Plan' window will appear on the screen (see Figure 7-13).

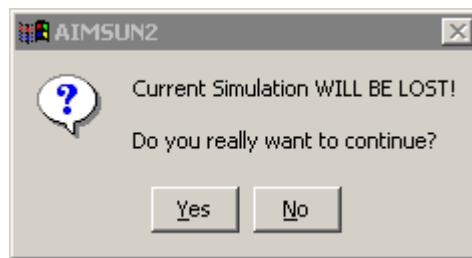
Figure 7-13: Load Traffic Control window

This window displays a list of control plans available for the current network, including the name of the plan, the date of creation or latest modification and the initial time. This initial time, which is the time at which the control plan will become active, can be modified by selecting the plan, typing the new time in the corresponding dialog windows and pressing Return.

The user can select one or several control plans for the same simulation experiment, taking care not to load more than one control plan due to be activated at the same time. Select one or more plans by clicking on them and then press the 'Load' button. The control plans will be loaded into the model and the name of the first one will appear in the information area at the bottom of the main window. For instance, Figure 7-13 shows an example where two plans are to be loaded into AIMSUN. Plan 'congested' will be applied from 7:30 to 9:00, when plan 'plan1' will be activated

To unload the traffic control plan, use the 'File / Unload Control' command. If a simulation run has already been launched, a confirmation message as shown in Figure 7-14, will appear.

Figure 7-14: Confirmation Message

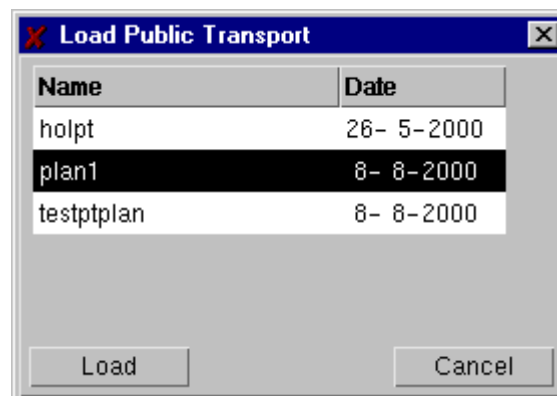


The Public Transport Plan is optional when running a simulation experiment. If public transport lines and bus stops are defined in the network model, a Public Transport Plan can be loaded. Otherwise it is not necessary.

7.1.4 Loading a Public Transport Plan

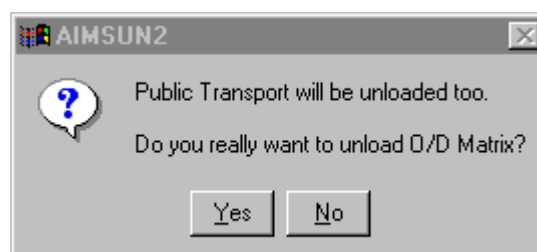
To load a Public Transport Plan, the traffic demand data (either Traffic Result or O/D Matrix) has to be loaded first. Then select the 'File / Load Public Transport' command from the menu bar or use the shortcut <Ctrl><P>, and the 'Load Public Transport' window will appear on the screen (see Figure 7-15). This window displays a list of available public transport plans for the current network, containing the names of the plans and the dates of creation or latest modification. Select one of them by clicking on it and then press the 'Load' button.

Figure 7-15: Load Public Transport window



To unload the Public Transport Plan, use the 'File / Unload Public Transport' command. If a simulation run has already been launched, a confirmation message as shown in Figure 7-14, will appear. If you attempt to unload the traffic demand data while a public transport plan is loaded, a confirmation message such as the one shown in Figure 7-16 will appear.

Figure 7-16: Confirmation Message



7.2 RUNNING THE SIMULATION

7.2.1 Parameters of the Experiment

Prior to running the simulation, sets of parameters that characterise a simulation experiment have to be defined. These parameters are grouped into six categories: simulation timing, modelling, reporting, random, detection and route choice aspects. They are assigned default values, which the user can view and change with the commands grouped under the 'Experiment' menu. 'Run Time' command is explained in this section (7.2), but the rest of the commands included in the 'Experiment' menu are explained in other sections.

'Modelling' command has already been explained in section 3.4. 'Output / Output Location' and 'Output / Statistics' are explained in section 9.1, while 'Output / Detection' is explained in section 9.2. 'Replications' is covered later in this chapter, in section 7.4. 'Route Choice' has already been covered in section 4.6. 'Incidents' is also included in this chapter, in section 3.6, 'VMS Messages' was covered in section 2.5.3 and finally, 'Environmental Models' is described in chapter 10.

All the dialog windows described in this set of 'Experiment' menu commands contain the following buttons:

- Default: Loads the default parameters common for all networks.
- Load: Loads the last saved parameters for the current network.
- Save: Saves the parameters displayed in the dialog window for the current network.
- OK: Accepts the parameters displayed in the dialog window as input for the current experiment.
- Cancel: Closes the dialog window without accepting modifications.

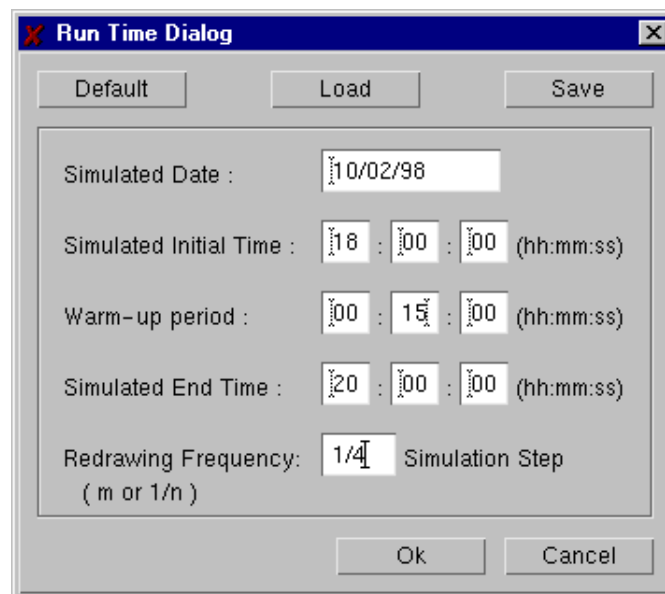
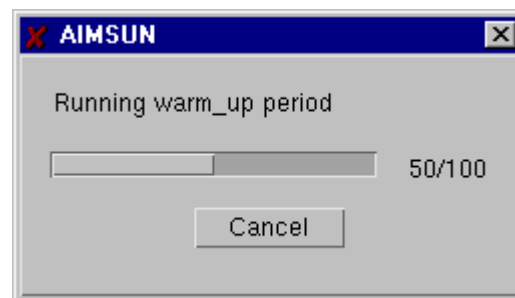
When loading a network, all six categories of parameters, Run Time, Modelling, Input (Past Costs), Output (Statistics, Detection and Paths), Replications and Route Choice, are loaded. If the network has previously saved parameters, these are used. Otherwise, the common default parameters are loaded. These parameters are stored in ASCII files, either at the network level or at a global level. The location, name and format of these files are described in Appendix 1.

Incidents and VMS Messages are only loaded if the corresponding toggle button has been activated in the Load Traffic Result or Load O/D Matrix dialog window, as explained in section 7.1.2. The Particular Models, Fuel Consumption and/or Pollution Emission are automatically activated if the loaded model has defined the required vehicle consumption and/or emission parameters.

7.2.2 Simulation Run Time

When you select the 'Experiment / Run Time' option from the menu bar, the 'Run Time' parameters dialog window appears (see Figure 7-17). This dialog box brings together a set of parameters related to simulation timings, which are the following:

- Simulated Date: This is the date (in format day/month/year) that is being simulated. This is the date that will appear in the statistical reports.
- Simulated Initial Time: Initial time (hours: minutes: seconds) for the simulation clock. This has to be in concordance with the Traffic Result or O/D matrix loaded as explained in section 7.1.2.
- Warm-up Period: Duration (hours: minutes: seconds) of the warm-up period. The purpose of this period is to fill up the network with vehicles before starting the effective simulation. It can be thought of as a transition period during which there is no simulation displaying either statistical or detection data gathering, although the simulation is actually taking place. During Warm-up period run, dialog in figure 7-18 is shown. Warm-up simulation can be cancelled at any time pressing the 'Cancel' button.
- Simulated End Time: Time (hours: minutes: seconds) when the simulation will finish. This has to be in concordance with the Traffic Result or O/D matrix loaded as explained in section 7.1.2. The effective length of the simulation will be (Simulated End Time - Simulated Initial Time).
- Redrawing Frequency: This parameter has two meanings:
 - m (m integer number greater than or equal to 1): the network will be refreshed each m simulation steps (e.g., if it is equal to 1, the animation will be updated every simulation step).
 - $1/n$ (n integer number greater than 0): the network will be refreshed n times per simulation step (e.g., if it is equal to 4, the animation will be updated 4 times per simulation step).

Figure 7-17: Run Time Parameters window**Figure 7-18: Warm-up period running**

7.2.3 Running Modes

There are three different ways of running the simulation: Run, Batch and Step. All these modes can be found in the 'Run' command menu. This option of the menu will be greyed out until a network and traffic demand data (traffic result or O/D matrix) have been loaded (the control plan is optional), otherwise the simulation cannot be run. During the simulation, the user can switch from one mode to another as often as desired.

Run mode can be selected using the command 'Run / Run' (or the shortcut <Ctrl>-<R>). This starts the simulation of the current scenario (network, traffic demand data, control and simulation parameters). This mode runs the simulation with the animated graphical representation of the traffic. Depending on the drawing scale, either the individual vehicle movements or the lane density are shown (see section 7.1 for more details on animation). Depending on the size of the network, the number of vehicles being simulated and the computer power, the simulation can be run faster or slower than real time. If it runs faster than real time, the user may wish to slow down the simulation in order to get the animation in real time. This is done using the Scale Bar (see section 7.2.5).

Batch mode is selected using the 'Run / Batch' command (or the shortcut <Ctrl>-). This simulation mode does not refresh the graphical display during the simulation. Therefore updates to the network status cannot be followed on the screen. In any case, the user can switch from one mode to another at any time by selecting the appropriate alternative menu command or by pressing <Ctrl>-<R> for Run or <Ctrl>- for Batch.

Run mode is slower than Batch mode, given that it has to draw all vehicles (or lane density colours) for every 'refreshment step' which is equal to the 'simulation step' multiplied by the 'redrawing frequency'. Run mode is useful for verifying that the simulation is running satisfactorily, and for understanding the behaviour of traffic in the network. On the other hand, Batch mode is mainly used when massive experimentation has

to be carried out, either for calibration purposes or for simulation results production, given that it is the fastest mode of simulation.

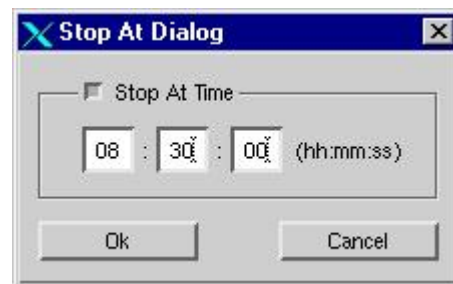
A third mode of simulation, the Step mode, is used only for debugging purposes. It is selected using the 'Run / Step' command (or the shortcut <Ctrl>-<T>). It runs one simulation step with graphical refreshment so the user can follow, for instance, the behaviour of a single vehicle at a junction, step by step (e.g. second by second).

7.2.4 Stopping the Simulation

The user can stop the simulation at any time, regardless of the simulation mode, using the 'Run / Stop' command from the menu bar (or the shortcut <Ctrl>-<T>). In any case, the simulation will finish when the simulation clock reaches the end of simulation time.

The user can also define simulation breakpoints (i.e. points in the simulation time where the run will be stopped automatically). A simulation breakpoint can be defined at any time, either while the simulation is running or when it is stopped. Select the 'Run / Stop at' command to display the window shown in Figure 7-19. Click on the 'Stop at Time' toggle button, set a stop time (hours: minutes: seconds) and press OK. The simulation will stop when the clock reaches this time.

Figure 7-19: "Stop at" Dialog Window



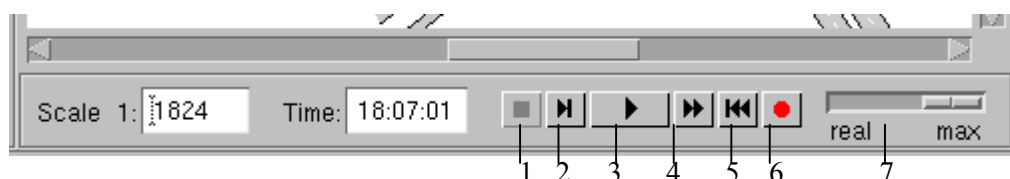
After a simulation stop, the simulation can be resumed by selecting commands Run, Real Run, Batch or Step, provided that the Simulation End Time has not been reached. However, even though the simulation end time has been reached, the user can extend this time using the 'Experiment / Run Time' command, but only if the simulation is Result-Based.

To initialise the simulation experiment, select the 'Run / Begin' option from the menu bar at any time. This will clear the current simulation experiment, statistical data areas, detection measures and all current vehicles in the network. If the 'Load Initial State' option was used when loading the Traffic Result, the initial state will be used again to initialise the network. If a new run is carried out without changing any parameter (random seed, modelling parameters, etc.), the results will be exactly the same.

7.2.5 Simulation Buttons

The simulation run can also be controlled using the Simulation Buttons from the Status Bar. The buttons are the following:

Figure 7-20: Simulation Buttons



1. Stop button: stops the simulation
2. Step button: run 1 simulation step
3. Run button: run mode

4. Batch button: batch mode
5. Begin button: initialise simulation experiment
6. Recorder button: to start/stop recording a simulation run for playing in AIMSUN3D offline.
7. Scale bar lets the user choose the desired speed of the simulator. If the scale is on the most left position the simulator works in real time; if it is on the most right position the simulator works as fast as possible. If different settings along the scale between real time and maximum run the simulator at one, two, three, four or five times real speed.

7.2.6 Running AIMSUN from a DOS Command line

AIMSUN can be directly run from a MS-DOS Command Prompt, just locating into the GETRAM folder and typing the command 'aimsun' or directly `c:\...\GETRAM v4.1\aimsun`. This command starts the AIMSUN Graphical User Interface.

The 'aimsun' command accepts some parameters to speed up the model loading and running process. There are the following options:

- 1) Just starts up AIMSUN and load a scenario
`c:\...\GETRAM v4.1\aimsun c:\scenario_path\scenario.sce`
- 2) Start up AIMSUN, load a scenario and run simulation in Run mode
`c:\...\GETRAM v4.1\aimsun -run c:\scenario_path\scenario.sce`
- 3) Start up AIMSUN, load a scenario and run simulation in Batch mode
`c:\...\GETRAM v4.1\aimsun -batch c:\scenario_path\scenario.sce`

Taking into account that `c:\...\GETRAM v4.1\` means the directory where `aimsun.exe` is located, and `c:\scenario_path\` is the folder where the scenario named 'scenario.sce' is located.

When executing the AIMSUN simulator in batch mode from the MS-DOS console, if the simulator detects that in parameter files multiple replications are defined they will be executed automatically.

7.2.7 Running AIMSUN without GUI: AConsole.

AConsole (AIMSUN Console) is an application used to run any number of AIMSUN simulations from a MS-DOS window without going through the Graphical User Interface.

The format of the command is:

```
aconsole <scenario_path1> [single | multiple] ... <scenario_pathN> [single | multiple]
```

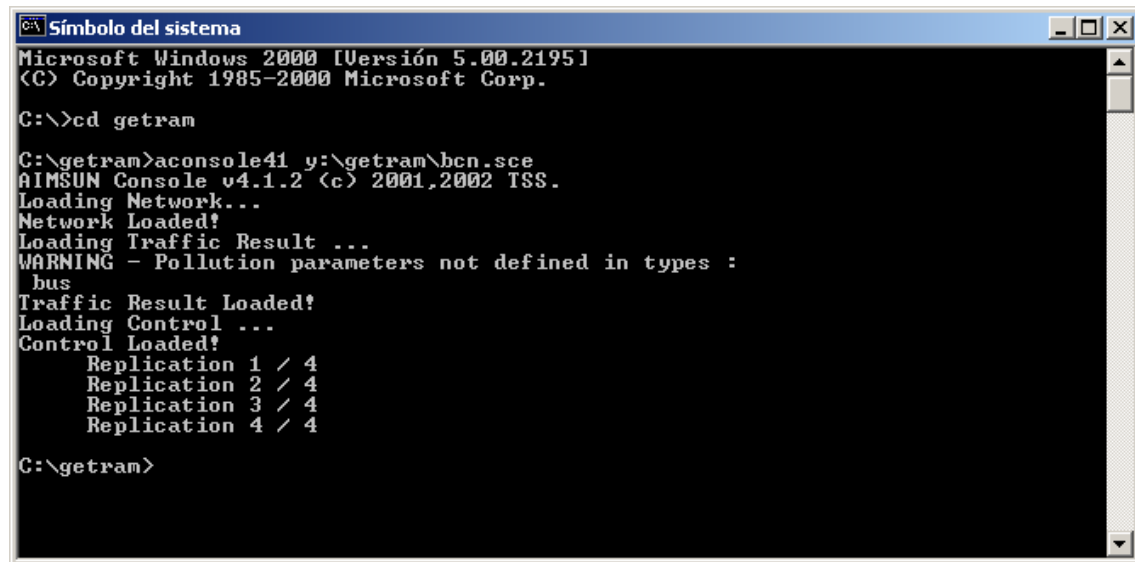
If you want to run a multiple number of replications, which should be previously defined (and saved) in the scenario through AIMSUN Replication Dialog, then the parameter 'multiple' has to follow the scenario path. In that case AIMSUN will execute 'Number of Replications' runs using the Random Seeds as defined in the scenario. Simulation Outputs of each replication will be stored according to the Output Location defined in the scenario. In case that the user has selected 'Calculate Average' in the scenario then the average replication will be created (only in ODBC and Access database output formats).

On the other hand, if you only want to simulate one replication of the simulation, despite of what you have defined in the scenario, the parameter 'single' has to be used. By default, if no 'single/multiple' parameter is used when running an experiment then the scenario will be simulated as multiple.

Examples:

- *aconsole c:\bcn.sce single x:\amara.sce multiple*
- *aconsole z:\networks\diagonal.sce*
 which is the same as *aconsole z:\networks\diagonal.sce multiple*

The next figure shows an example of the use of AConsole in a MS-DOS window.



```
Símbolo del sistema
Microsoft Windows 2000 [Versión 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\>cd getram

C:\getram>aconsole41 y:\getram\bcn.sce
AIMSUN Console v4.1.2 (c) 2001,2002 TSS.
Loading Network...
Network Loaded!
Loading Traffic Result ...
WARNING - Pollution parameters not defined in types :
bus
Traffic Result Loaded!
Loading Control ...
Control Loaded!
    Replication 1 / 4
    Replication 2 / 4
    Replication 3 / 4
    Replication 4 / 4

C:\getram>
```

If you want to simulate different scenarios of the same network, remember that the replications that will be simulated and the database where the simulation output will be stored are the same for all the scenarios of the same network. As they are saved at the network level, simulating the same network with different scenarios one after the other will overwrite the output data stored in the previous scenarios and you will only get the statistics of the last scenario. To solve this you can make as many copies of your network as scenarios you want to simulate. Afterwards, you can define the same ODBC or Access database to store the statistics in all the networks, but at the same time use different replication ID's in each copy of the network. Doing this will allow you to have all the simulation output data of all the replications in the same database.

7.3 SAVING THE SIMULATION STATE

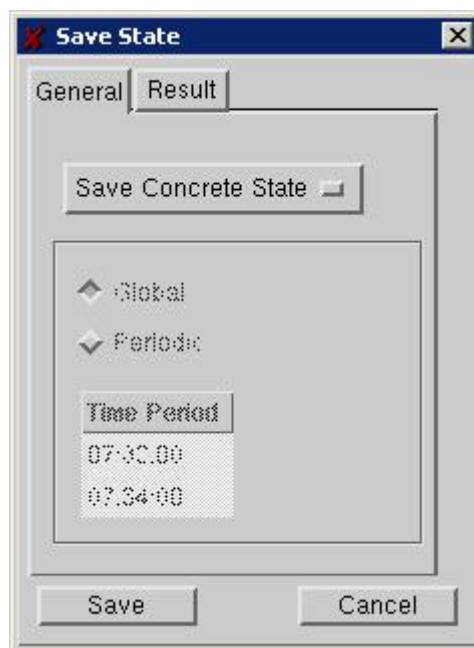
At any time during a simulation the user may wish to save the current state of the network, which may be retrieved and used in future simulation experiments. One reason for this might be to use this state as an initial state for continuing the simulation experiment later or under different conditions. Another reason might be to produce a Traffic Result from an O/D Matrix. This is the case when a state is saved while running in Route-Based mode.

The network state is defined by:

- Flows at the input sections (vehicles/hour).
- Turning proportions for all sections.
- Number of vehicles travelling along each section for each turning.
- Number of vehicles stopped at every section for each turning.
- Mean speed for vehicles travelling at each section.

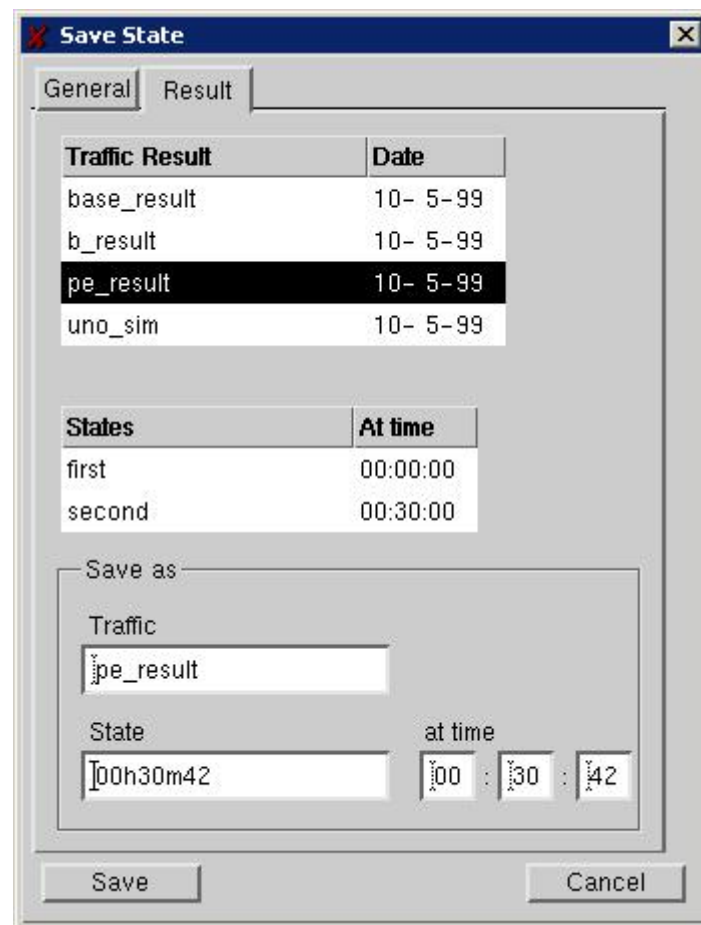
Saving the current state can be done using the command 'File / Save Current State'. This command can only be used when the simulation is stopped. Figure 7-19 shows the 'Save State' window. This window has two tab folders named 'General' and 'Result'.

Figure 7-19: Save State Dialog Window General tab folder



The 'General' tab folder contains an option menu for selecting between 'Save Concrete State' and 'Save Average State'. They differ in terms of the data used to create the state. The first can only be selected when simulating with a traffic result, while the second can be selected either when simulating with a traffic result or with an O/D matrix.

Once the save mode has been selected in the General tab folder, the user can use the via the 'Result' tab folder to specify into what Traffic Result to save the state, either as a Current state or an Average state (see Figure 7-20). The name and time of the state can also be defined here. By default, the system provides the name as the current simulation time (e.g. at time 00:30:42 it suggests the name 00h30m42).

Figure 7-20: Save State window Result tab folder.

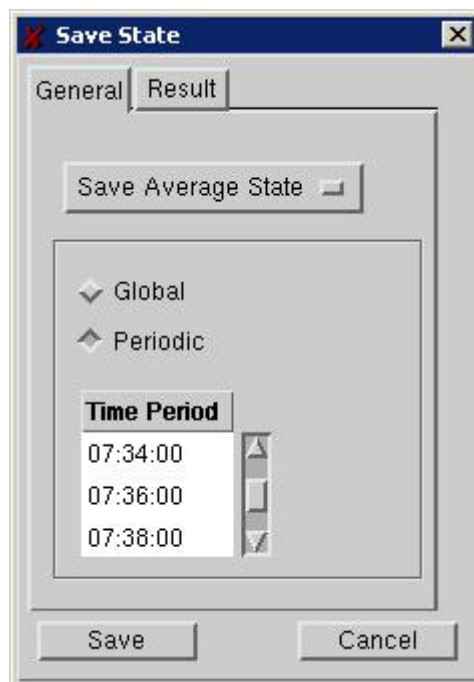
7.3.1 Save Concrete State

The flows at the input sections and the turning proportions for all sections are taken directly from the input flows and turning proportions defined in the state which is active at this time. Therefore, they will be the same as the traffic result used as input. The number of vehicles travelling, their mean speed and the number of vehicles stopped at every section for each turning are taken from the current simulated situation, not from the input state, nor from the statistical data gathered during the simulation.

7.3.2 Save Average State

The flows at the input sections and the turning proportions for all sections are taken from the statistical results of the current simulation. The user may then define whether to use Global or Periodic statistical data. Global means that data gathered during the whole simulation period is considered. Periodic means that only the data corresponding to certain time periods will be used.

The user can select one or various periods by clicking on the 'Time Period' list box (see Figure 7.21). The number of vehicles travelling, their mean speed and the number of vehicles stopped at every section for each turning movement are also calculated from the statistical data, not from the input state or from the current situation.

Figure 7-21: Save Average State

7.4 RUNNING DIFFERENT REPLICATIONS OF THE SIMULATION

A simulation model does not provide a unique solution to a given problem, it just tries to emulate the behaviour of a complex system in which randomness is involved. Each run of a simulation program, called a replication, provides a possible behaviour for the modelled system, which means a point in a sample of feasible results of the model. The final result is obtained through the statistical processing of simulation results from different replications. Therefore, a simulation study requires the run of a number of replications of the same model, using different random seeds.

The Random Seed is the only parameter related to randomisation. This parameter must be an integer number and it is used as the initial seed for the pseudo-random number generator. The pseudo-random generator is used to sample real numbers uniformly distributed between 0 and 1. These numbers are used to produce the different random distributions used in the model, such as the Shifted Exponential distribution for vehicle arrivals, the Normal distributions used to generate vehicle characteristics, or the Uniform distribution used to assign the next turning movements to vehicles. Using the same seed with the same input data (network, traffic demand data, control plan and simulation parameters) creates identical simulation results.

There are three types of seed in AIMSUN. The global seed defined by the user, which is used to generate randomly the rest of local seeds, N section-seeds, and M-centroid seeds. The section seeds are used in the generation of vehicle arrivals and the generation of the next turning (for Result based simulation). The centroid seed are used for the generation of vehicles and decision of route to follow in case of variable route choice model. Also when a vehicle is re-routed during the trip and has to apply the route choice model again, it uses the seed of its origin centroid, and not the seed of the section.

A simulation experiment should consist of a set of replications of the same scenario, composed of the traffic network, traffic demand, traffic control plan and a set of global modelling parameters. The user can define the number of replications to perform and a different seed for each replication. Then the whole experiment can be run in Batch mode and the results of each replication are stored. The Replication Identifier identifies each replication result. Chapter 7 explains how different replication results are stored. The whole experiment simulation results should be taken as the mean of all replication results, and variances and confidence intervals have to be calculated.

Replications are defined via the Replications dialog window (see Figure 7-22) that appears when you select the 'Experiment / Replications' command. In this dialog window, the user may choose to perform either a single or a multiple replication run by selecting the corresponding toggle button.

If the Single replication run is being used, the user must provide an initial random seed and a replication identifier. Then, to run the simulation, close the Replication dialog window by clicking on 'OK' button and select any of the available running modes from the 'Run' command.

If the Multiple replications run is being used, the user must type in the number of replications to run and must then choose between typing the set of pairs (replication identifier, random seed) or simply pressing the 'Create' button so that the program does it automatically. Optionally, the average of all replications can be also automatically calculated and stored, either in ODBC or Access database. This option is not available in case that the simulation output was stored in ASCII files. Average calculation is activated clicking on 'Calculate Average' toggle button. The average of all replications is stored as a new replication, therefore a replication identifier has to be defined.

Multiple replications can only be run in Batch mode. This is done via the Replications dialog window by clicking on 'Simulate' button. When running multiple replications, the 'Replicator' dialog window appears (see Figure 7-23). It provides information on which replication number is running, the percentage of run time covered, and the total number of replications to be run. The user may cancel the multiple replication run at any time by pressing the 'Cancel' button.

Figure 7-22: Replications Dialog Window

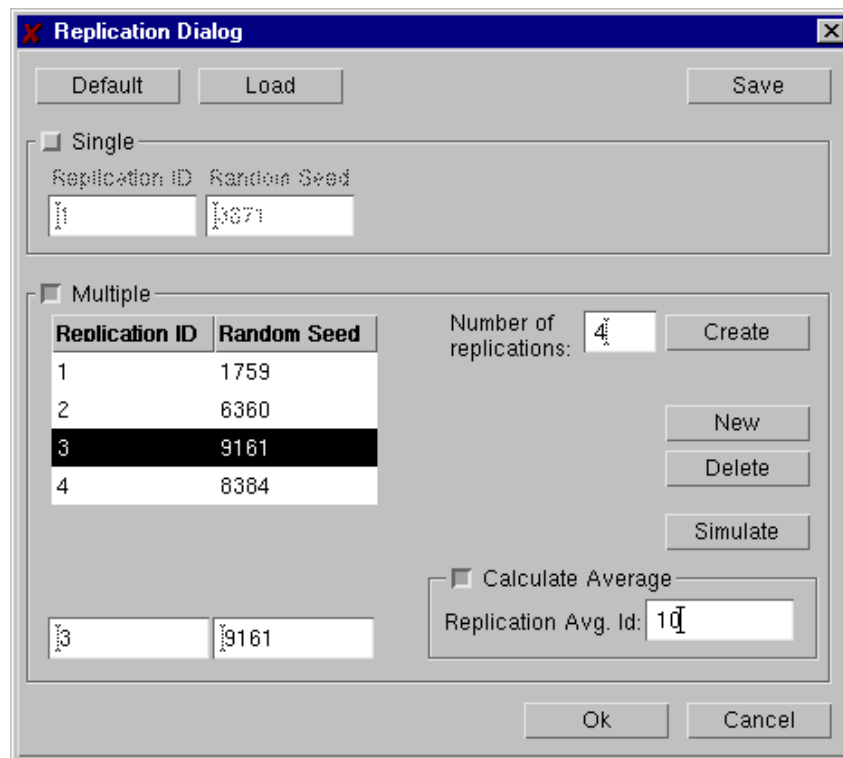
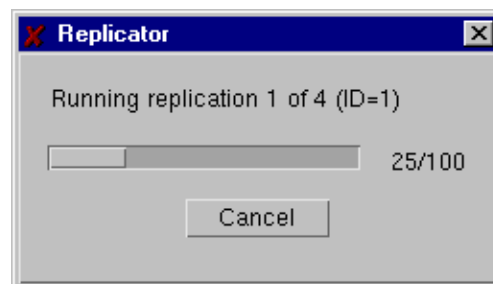


Figure 7-23: Running multiple replications



7.4.1 Calculation of the number of replications

This method [LAW91] provides a number of replications required in order to obtain a value within the limits of $\kappa\%$ of the mean of our sample with a $\alpha\%$ level of confidence.

First, we need to carry out a pilot experiment with a predefined number of replications (i.e. 4-6). Using the results of this experiment we calculate a confidence interval and check whether the above condition is satisfied. If not, we calculate:

$$n^* = n \left(\frac{h}{h^*} \right)^2$$

where:

n^* : total number of replications required

n : number of replications of the pilot experiment

h : confidence interval of the pilot experiment

h^* : confidence interval of the whole experiment.

Example:

Let us assume that we are studying the output variable 'travel time', and that we want to obtain a value within the limits of 5% of the mean of our sample with a 95% level of confidence.

We define a pilot experiment with $n=10$ replications and the resulting mean travel time (\bar{x}) is 32.4818 min. and the standard deviation of the sample (S) is 3.5149 min. We calculate the 95% confidence interval, where h is calculated as:

$$h = t_{n-1; 1-\alpha/2} \frac{S}{\sqrt{n}}$$

$$t_{9,975} = 2.26$$

$$h = 2.512$$

The 95% confidence interval ($\bar{x} \pm h$) is $(32.4818 - 2.512, 32.4818 + 2.512) = (29.9698, 34.9938)$, so we conclude that this is not an acceptable result, as 5% of the mean would be $0.05(32.4818) = 1.624$, which is smaller than $h=2.512$.

We now apply the above equation to calculate the required number of replications

$$n^* = 10(2.512/1.62)^2 = 24.04$$

We recommend that you round up n^* to the next integer, so the estimated number of replications required would be 25. However, we recommend taking this as an upper limit.

If we try 15 replications we will get a mean of 32.1094 and a standard deviation of 3.1903. This gives a value of $h = 1.3144$. Now, $0.05(32.1094) = 1.61 > 1.3144$, so the confidence interval is $(32.1094 - 1.3144, 32.1094 + 1.3144) = (30.795, 33.4238)$. Although it is still greater than 5% of the mean, we might think that this is an acceptable result, as we are saving 10 replication runs.

7.4.2 Replication Information Output

In each simulation run or replication, AIMSUN v4.1 generates the output of all information related to its scenario and all experiment parameters. This information is:

- Replication Information: Identifier, Seed, Simulated date, Initial Simulation Time, End Simulation Time, Warm-up, Execution Date, Execution Time, Statistics Interval, Detection Interval and etc.
- Scenario Information: Network, Traffic Demand (either traffic result with its states or OD Matrix), Arrival Distribution, Control Plans, etc.
- Experiment Parameters: Simulation Step, Reaction Time at Stop, Queuing Up Speed, Queuing Leaving Speed, Applying 2 Lanes Car Following, Percentage Overtake, Percentage Recover, On Ramp Model, Route Choice parameters, etc.

This information, when the output is based in ASCII files, is stored in a text file named 'ReplicationInf.txt' in the output directory, and, when the output is in a Data Base, it fills the tables Replicat, RepScen, RepState, RepCtrl, RepExp and RepExt (see Appendix 2: Output Data Base Definition to see all attributes associated to each table).

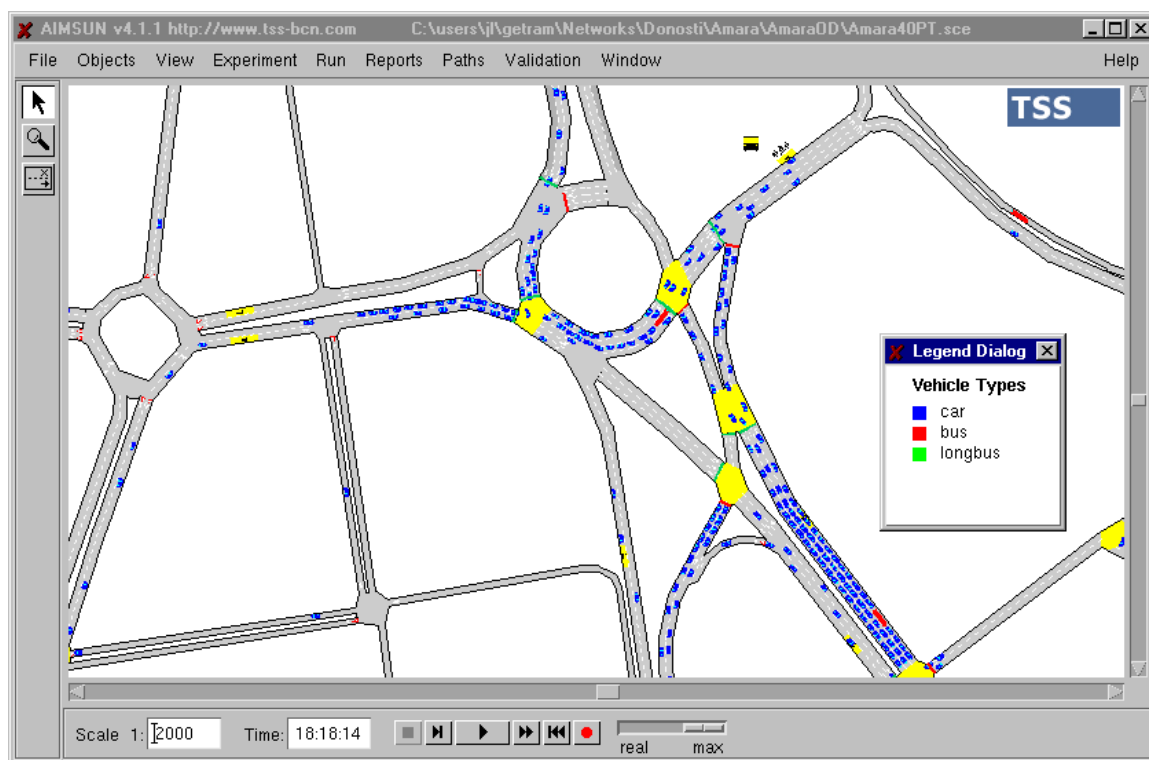
8. GRAPHICAL USER INTERFACE

This chapter provides a detailed description of all output provided by the AIMSUN microsimulator: Animation of the Simulation, Statistical Results and Detection data.

8.1 ANIMATION OF THE SIMULATION

AIMSUN provides as output a graphical animation of the simulation (see Figure 8-1). This means that, depending on the scale selected, the vehicles moving are drawn on the traffic network while the simulation runs and their positions are updated at every animation step, which is calculated as the simulation step multiplied by the redrawing frequency (see section 7.2.2). When simulating using the Run option, the animation goes as fast as possible. Simulating using the Real Run option slows the animation down to real time. Batch simulation mode does not produce simulation animation at all, and is therefore the fastest simulation mode.

Figure 8-1: Simulation animation. Vehicles detail



8.1.1 Vehicle Detail

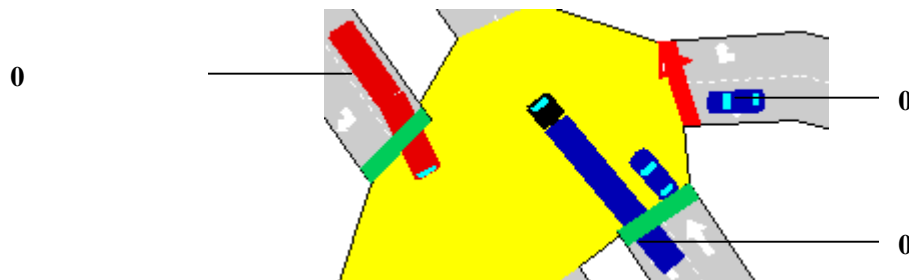
In 2D animation, vehicles can be drawn at two different levels of detail: Low Detail and High Detail. Drawing vehicles in High Detail decreases the animation speed so we have added a new option in the View Menu, Vehicle low detail / Vehicle high detail, allowing the user to select the level of detail. If you choose low detail, vehicles will be drawn as rectangles, otherwise they will be drawn as more realistic vehicles (see figure 8-2).

There are six models of realistic vehicles: car, van, truck, bus, bicycle and pedestrian. Each vehicle type has a model assigned. By default, if the vehicle is typed as public transport we use the bus model; in the other cases we assign the model according to the length of the vehicle:

- $\geq 10\text{m}$ truck
- $5\text{m} \leq \dots < 10\text{m}$ van
- $1.5\text{m} < \dots < 5\text{m}$ car
- $0.5\text{m} < \dots \leq 1.5\text{m}$ bicycle or motorbike
- $\leq 0.5\text{m}$ pedestrian

The user can change it and make his/her own assignation in the Preferences Dialog. See the Preferences Dialog section (8.2.9.3) for more information on how it works.

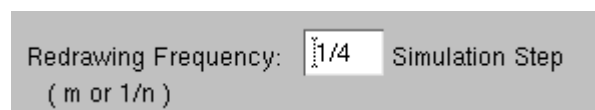
Figure 8-2: Simulation animation. Vehicles High Detail



Van, truck and bus vehicles models can be defined as articulated.

8.1.2 Smoother Animation

In AIMSUN v4.1 a Smoother animation option has been implemented. Consequently the Redrawing frequency parameter (Run Time Dialog) has two interpretations:



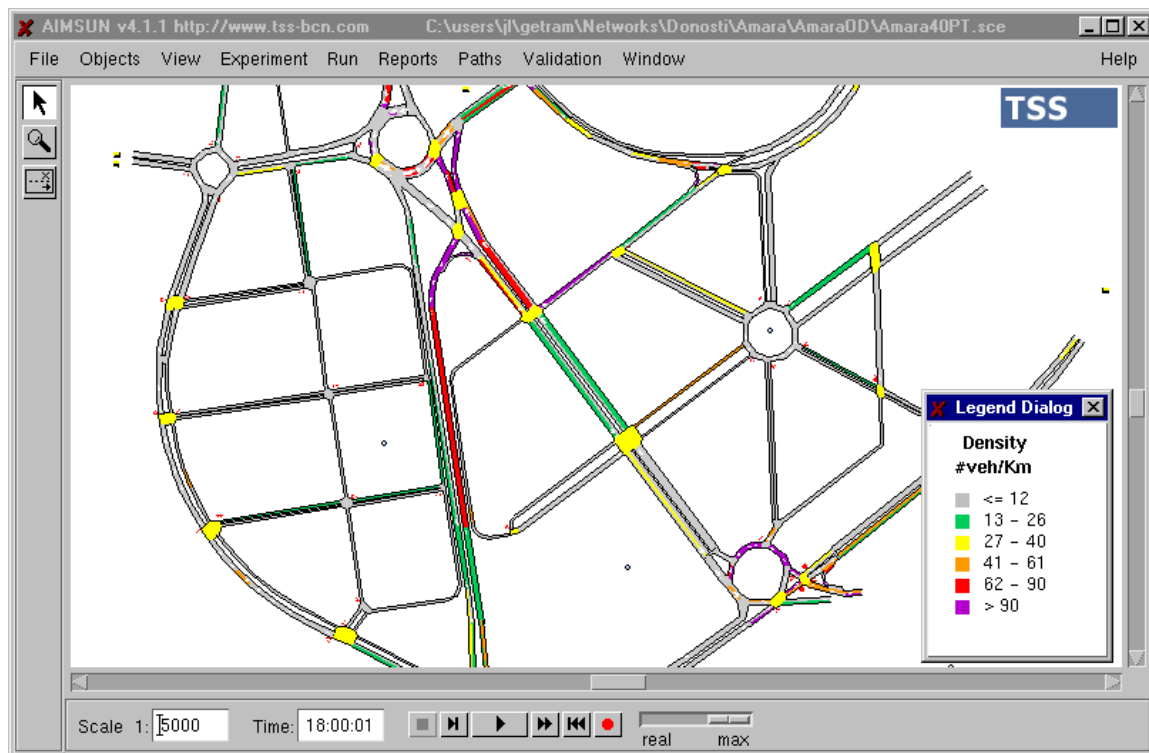
If you introduce an integer greater than zero (m), the network will be refreshed each m simulator cycles (as in previous versions), but if you introduce $1/n$ ($n > 0$), the network will be refreshed n times per simulator cycle.

Keep in mind that updating each vehicle state is carried out at every simulation step, not at each animation step. When Smoother animation is applied, positions of vehicles at each animation step are interpolated from the positions at each simulation step.

8.2 VIEW COMMAND MENU

The 'View' command menu groups a set of display functions. The first two options are for scaling purposes. 'View / Whole Network' automatically sets the scale to a value so that the entire network fits into the display area. 'View / Minimum Cars Scale' automatically sets the scale to a value such that the individual vehicles can be seen when running in Run, Real Run or Step modes (1:3000). When the scale is larger than this value, no cars are drawn but the sections are coloured according to a range of colours that represents the traffic density of the section (as shown in Figure 8-3). These colours are also updated for every step of the simulation run.

Figure 8-3: Simulation animation: Densities



8.2.1 Vehicle Colouring

By default, vehicles are coloured in the colour selected as Generic Vehicle in the 'View / Preferences' dialog (see section 8.2.9). The user may wish to display the vehicles in different colours to identify certain features. This is done using the 'View/Vehicle Colouring' option. The user can select from a range of colours to distinguish vehicle types, or use different colours to identify the vehicle origin, or the vehicle destination centroid. The different options are exclusive. Only one can be active at any given time. For the first option, select the 'View/Vehicle Colouring/Vehicle Type' command and each vehicle type will be automatically assigned to a particular default colour. The user may define this set of colours through the 'View / Preferences' command.

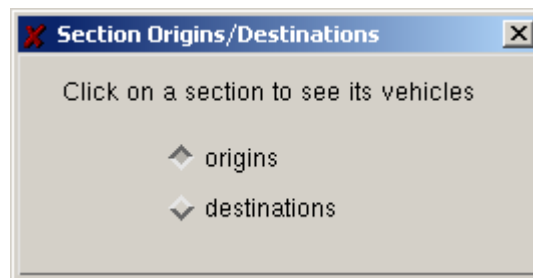
A second option is to colour vehicles by Turn. This is done by selecting the 'View/Vehicle Colouring/Turn' command. Vehicles at each section will be coloured depending on the next intended turning movement. Vehicles turning on the far right will be coloured green while vehicles turning to the far left will be coloured in red. A range of different colours will be used in between.

A third option is to colour vehicles by origin or destination centroid. This is done by selecting the 'View/Vehicle Colouring/Origin' and 'View/Vehicle Colouring/Destination' commands respectively. Vehicles coming from or going to certain centroids will be drawn in a different colour. In both cases the set of centroids to be taken into consideration must have been defined previously using the 'View / Preferences' dialog (see section 8.2.9).

The 'View/Vehicle Colouring/Lost' option will colour lost vehicles in the colour selected as Lost Vehicle in the 'View / Preferences' dialog, while the rest are coloured in the colour selected as Generic Vehicle. Lost vehicles are vehicles that have missed a turn and cannot reach their desired destination.

The 'View/Vehicle Colouring/Sections' option is used to view the origin or destination centroids for all the vehicles that are in a specific section or polysection. A window requesting the user to click on the desired section will appear (see figure 8-4). Select the desired toggle button, origin or destination, and then click on a section and the vehicles present in that section will be coloured according to their origin or destination centroids. In case that the user selects a section that pertains to a polysection, vehicles of all the polysection are coloured by their origins/destinations.

Figure 8-4: Colour vehicles in a section by origin or destination



These four last options, View/Vehicle Colouring/Origin, Destination, Lost, and Sections are only possible when the simulation is Route-Based, i.e., when traffic demand is defined as an O/D Matrix.

To colour Public Transport Vehicles, select the 'View/Vehicle Colouring/Public Transport' option. This causes vehicles to be coloured by bus line. Lines to be coloured are selected via the 'View / Preferences' dialog (see section 8.2.9). Vehicles belonging to the selected lines will be displayed in a different colour.

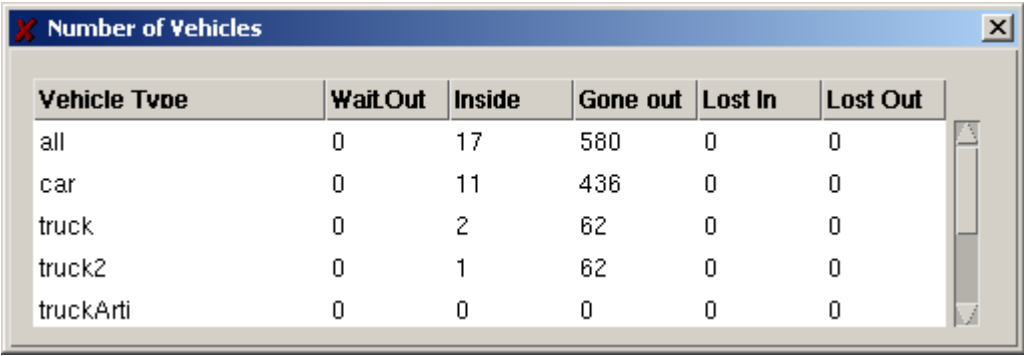
Finally, using the 'View/Vehicle Colouring/Vehicle Tracking' option, the user can choose to highlight the vehicles that are being tracked through a GETRAM Extension application in a different colour (see the GETRAM Extension User Manual for information on how to track a vehicle).

8.2.2 Displaying Number of Vehicles

During the simulation run the user can view information about the number of vehicles of each type that are currently driving in the network. When you select the 'View / Number of Vehicles' option, the dialog window shown in Figure 8-5 is displayed. If this window is left open while the simulation is running, the data in the dialog window will be updated continuously.

The Number of Vehicles dialog has the following information:

- **Wait. Out:** Number of Vehicles that are waiting to enter in the network.
- **Inside:** Number of Vehicles that are currently in the network.
- **Gone Out:** Number of Vehicles that have left the network.
- **Lost In:** if the route-based mode is being used, the number of lost vehicles that are currently in the network.
- **Lost Out:** if the route-based mode is being used, the number of lost vehicles that have left the network since the beginning of the simulation until the current time.

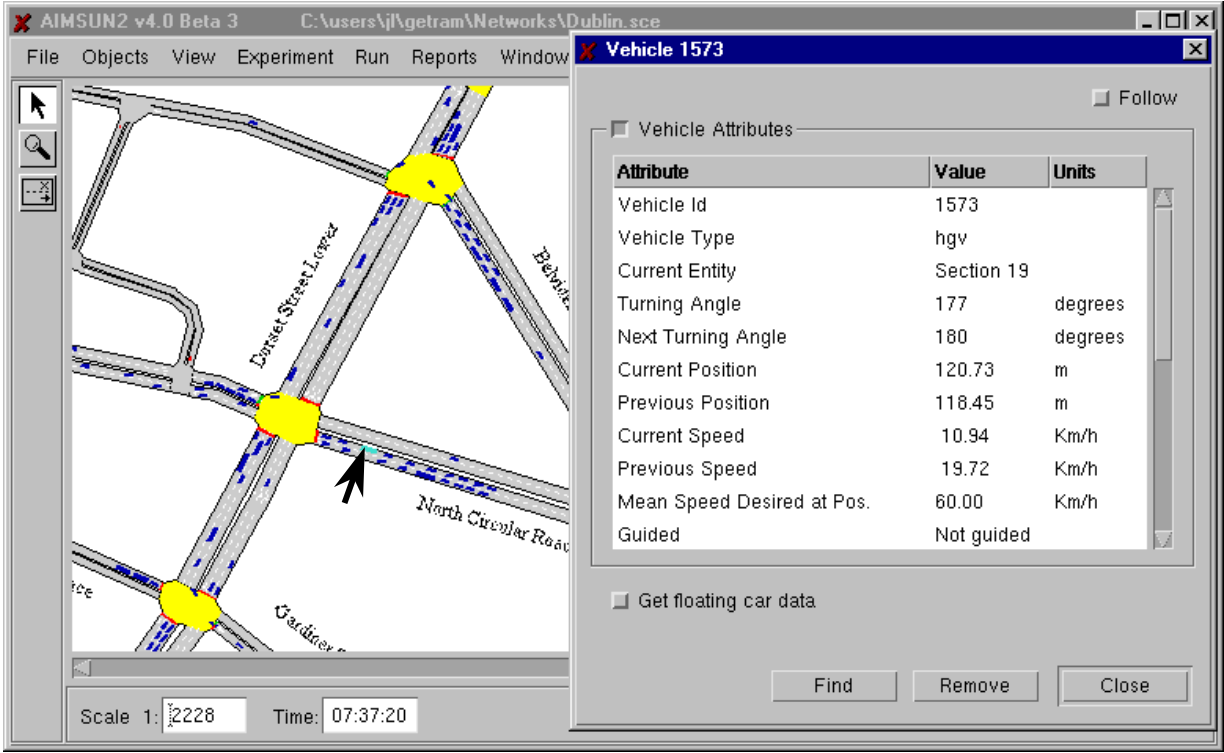
Figure 8-5: Number of Vehicles Dialogue Window


The image shows a window titled 'Number of Vehicles' with a table containing vehicle statistics. The table has six columns: Vehicle Type, Wait Out, Inside, Gone out, Lost In, and Lost Out. The rows represent different vehicle types: all, car, truck, truck2, and truckArti.

Vehicle Type	Wait Out	Inside	Gone out	Lost In	Lost Out
all	0	17	580	0	0
car	0	11	436	0	0
truck	0	2	62	0	0
truck2	0	1	62	0	0
truckArti	0	0	0	0	0

8.2.3 Displaying Vehicles Attributes

The user can also obtain detailed information on a particular vehicle just by double clicking on it. The Vehicle dialog window will then appear and the vehicle will be highlighted in a different colour (see Figure 8.6). This can only be done when the simulation run has been halted. Now, if the model is run after opening the Vehicle dialog window, the dynamic information contained in this dialog window is continuously updated while the simulation runs until the vehicle exits the network.

Figure 8-6: Vehicle Information Dialogue Window


The image shows the 'Vehicle 1573' dialog window. On the left is a map view of a road network with a yellow circle highlighting a specific location. On the right is a table of vehicle attributes. The 'Vehicle Attributes' checkbox is checked. The table lists various attributes such as Vehicle Id, Vehicle Type, Current Entity, Turning Angle, etc., along with their values and units. At the bottom, there are buttons for 'Find', 'Remove', and 'Close', and a checkbox for 'Get floating car data'.

Attribute	Value	Units
Vehicle Id	1573	
Vehicle Type	hgv	
Current Entity	Section 19	
Turning Angle	177	degrees
Next Turning Angle	180	degrees
Current Position	120.73	m
Previous Position	118.45	m
Current Speed	10.94	Km/h
Previous Speed	19.72	Km/h
Mean Speed Desired at Pos.	60.00	Km/h
Guided	Not guided	

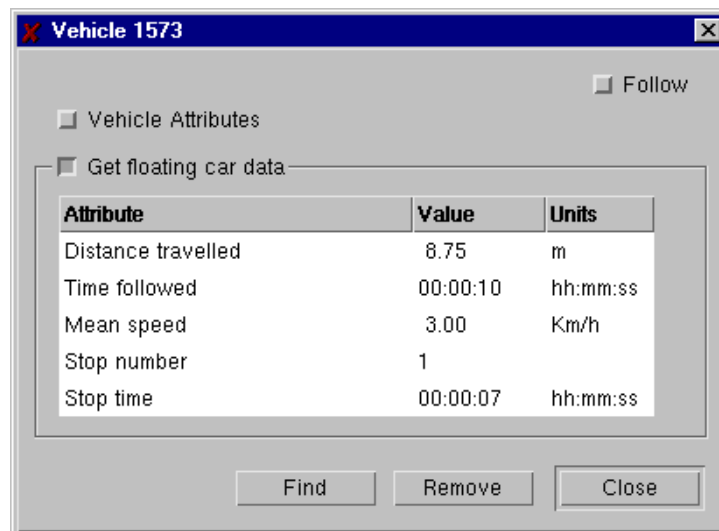
The Vehicle dialog window provides two types of information: Vehicle Attributes and Floating Car Data. By default, vehicle attributes that describe the current state of the vehicle are displayed. The user may select whether or not to display this information by clicking on the 'Vehicle Attributes' toggle button.

The data displayed are vehicle identifier and vehicle type, current entity (section or junction), turning angle (indicating the next turn), current and previous positions in the entity, current and previous speed, desired speed at position. Also, a flag indicating whether the vehicle is guided or not, the guidance acceptance level, the origin and destination centroids and the path tree being used are given when running in the Route-Based mode. Other vehicle static data provided are length, width, acceleration and deceleration rates, vehicle desired speed, speed acceptance, minimum distance to keep from previous vehicle and maximum give way time (see section 3.4.1 for a description of these parameters).

On the other hand, Floating Car Data can be obtained via the Vehicle dialog window. Clicking on the 'Get floating car data' toggle button causes the dialog window to expand as displayed in Figure 8-7. From that point, the distance travelled, the time being followed, the mean speed, the number of stops and the total stop time of the floating car are displayed in the dialog window and continuously updated while the simulation runs until the vehicle exits the network.

It is also possible to track the vehicle using an automatic, continuous scrolling network view. This is done when you click on the 'Follow' toggle button. The 'Find' button can be used to locate a vehicle in the network.

Figure 8-7: Vehicle Dialogue Window: Get floating car data

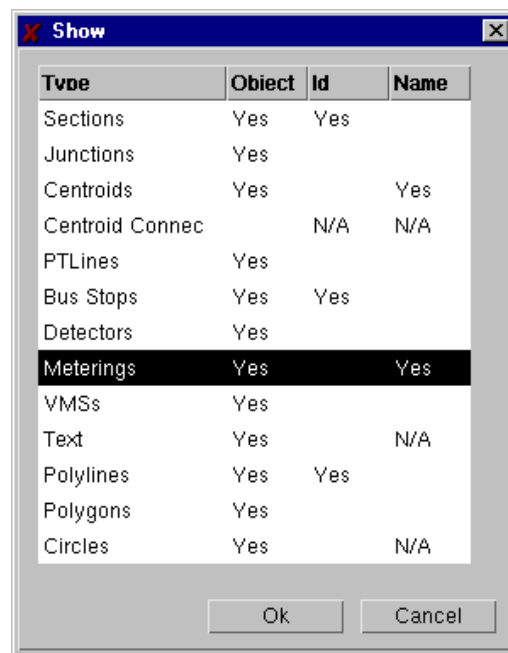


The last button in the Vehicle dialog window is the 'Remove' button. Clicking on this button causes the vehicle to be removed from the network. The removed vehicle will not be considered in the statistical data gathered for the current section.

8.2.4 Show Objects

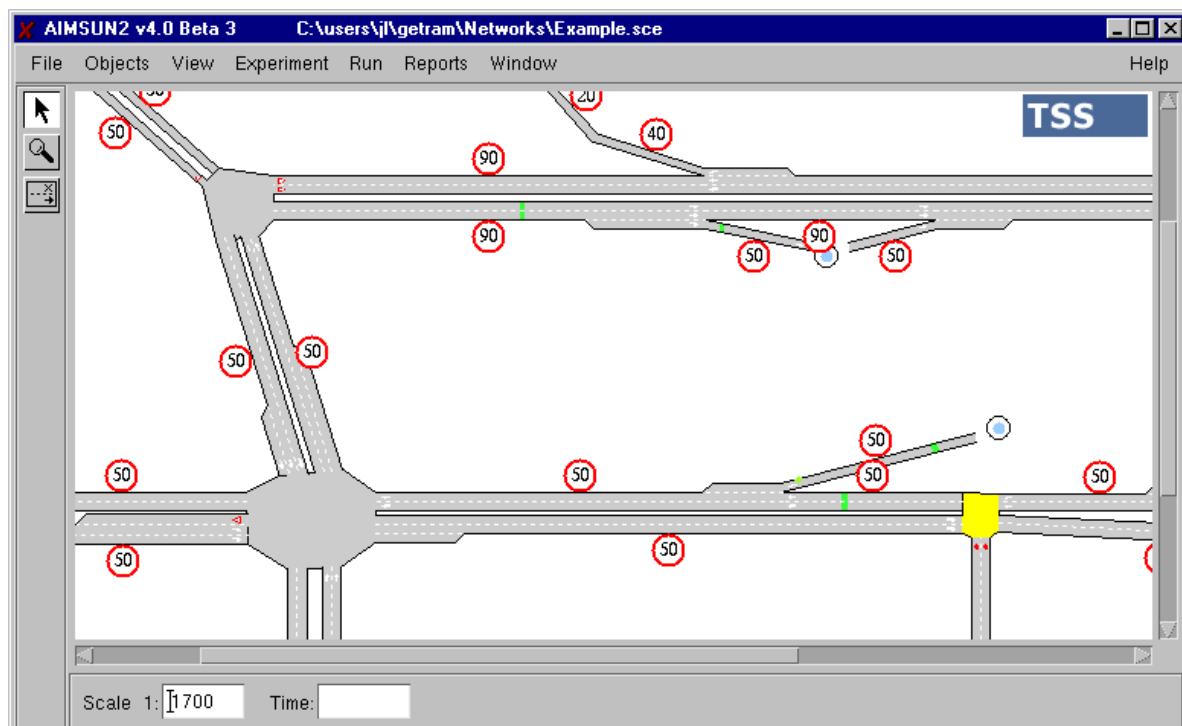
The 'View / Show Objects' command is for selecting which objects to display on the screen and whether or not to display their identifiers and or their Names. Figure 8-8 displays the 'Show Objects' dialog window. It consists of a column for each object characteristic to the display: object type, identifier and name. The user may activate the display of each characteristic of each object by clicking on the corresponding column. A 'Yes' means it will be displayed, empty means that it will not.

This dialog window looks similar to that in Tedi. It is now possible to visualise the vehicles over the background, with the sections and junctions hidden.

Figure 8-8: Show Objects Dialogue Window

8.2.5 Show Speed Limits

The 'View / Show Speed Limits' command displays speed limit traffic signs for each section, as shown in Figure 8-9. In a polysection, one speed limit sign is shown per section composing the polysection. To hide the speed limit signs, select the 'View / Hide Speed Limits' command.

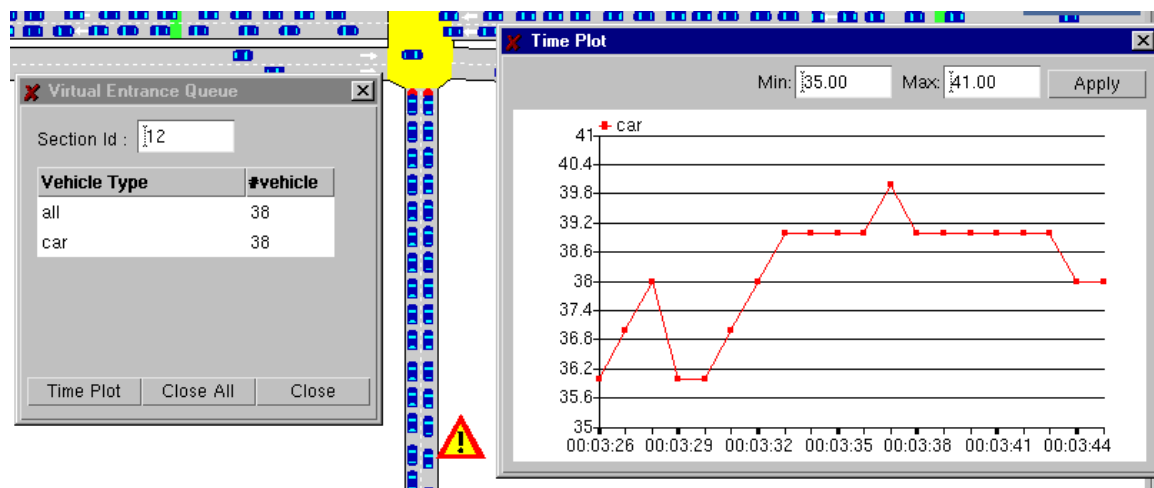
Figure 8-9: Speed Limits display

8.2.6 Show Virtual Queues

Selecting option Show Virtual Queue from the View menu, a warning traffic signal will appear over all those sections that have entrance queues (see figure 8-10).

Double clicking over one of these triangles opens a dialog showing the number of vehicles, detailed per vehicle type, of the queue of the correspondent section. The user has the option of displaying a Time Plot showing the evolution of the queue during the last 20 simulation steps. If you have more than one vehicle type you will see a chart for the total number of vehicles and a chart for each vehicle type.

Figure 8-10: Virtual Queues Display



The warning triangles are only shown if the network scale is greater than or equal to the Minimum car scale.

8.2.7 Show Occupied Detectors

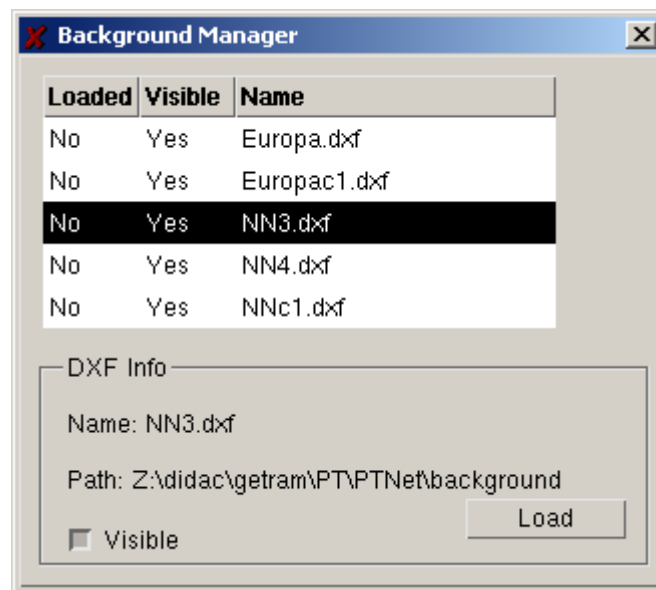
There is an option in the AIMSUN Interface to able/disable the visualisation of occupied detectors. This option, which can be found in the 'View' menu, appears as an alternative command: 'Show Occupied Detectors' vs. 'Hide Occupied Detectors'

When enabled, the detectors will be highlighted in a different colour whenever a vehicle passes it. The default colour is red, but this colour can be defined by the user via the Preferences dialog window. Only full simulation steps are considered. Therefore, if a vehicle passes the detector during part of a simulation step, the detector will be highlighted during the whole simulation step.

This works both for normal vehicle detection and Clickable Detection (see section 8.4). Visualisation of occupied detectors is automatically activated when Clickable Detection is activated.

8.2.8 Background

The 'View / Background' command opens the 'Background Manager' dialog window, which is used to load background images (see Figure 8-11). This dialog window is similar to the one described in the Tedi User Manual except that it is not possible to register a background in AIMSUN. The backgrounds that have been registered via Tedi are listed in the dialog window and these are the only ones that can be loaded into AIMSUN. The rest of the dialog window is the same as its counterpart in Tedi.

Figure 8-11: Background Manager Dialog Window

8.2.9 Preferences

The 'View / Preferences' dialog has three categories: Colours, Backgrounds and Shapes (see Figure 8-12). 'Colours' allows the user to customise the colours to be used for drawing certain objects. 'Backgrounds' is used to select preferences for background images. Finally, 'Shapes' is used to select what shape to use to draw each vehicle type. When selecting an element of the Category area, the preferences of this element are shown.

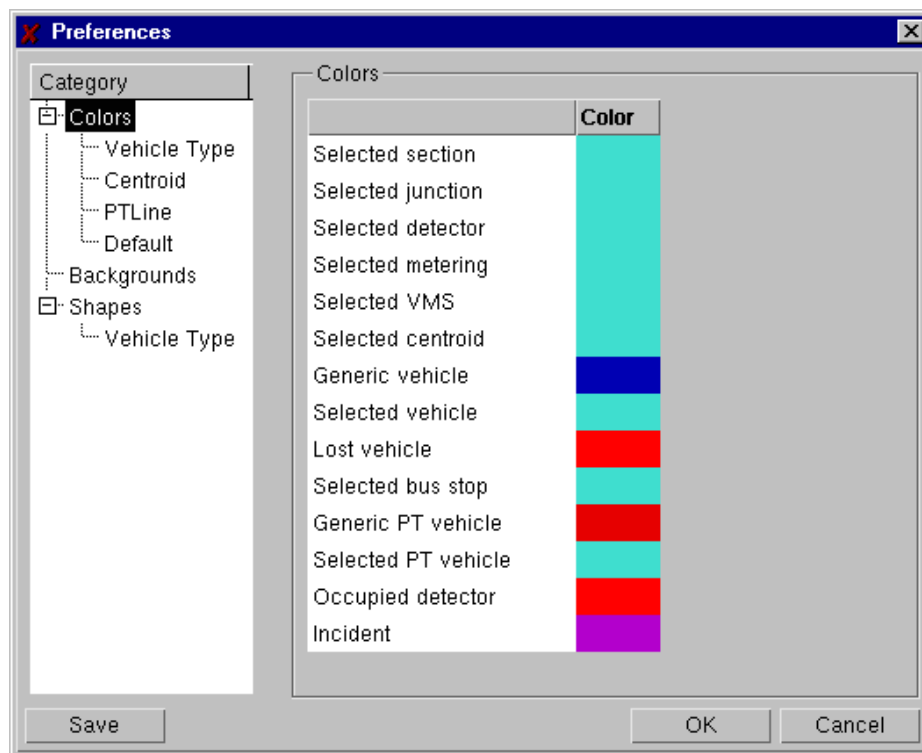
All the information of this dialog is kept on the hard disk, in several files:

- The preferences of background : *network path\BckPreferences*
- The generic preferences of the network: *network path\Preferences*
- The colour palette: *getram install dir\getram\Aimsun\DefaultColors*
- The colours of the vehicle types and public transport vehicle types:
network path\odmatrix o traffic res.\AIMSUN\VehTypesColors
network path\PTVehTypesColors
- The colours of the centroids: *network path\CentroidsColors*
- The colours of the public transport lines: *network path\PTLinesColors*
- The shapes of the vehicle types and public transport vehicle types:
network path\odmatrix o traffic res.\AIMSUN\VehTypeShapes
network path\PTVehTypeShapes

8.2.9.1 Colours

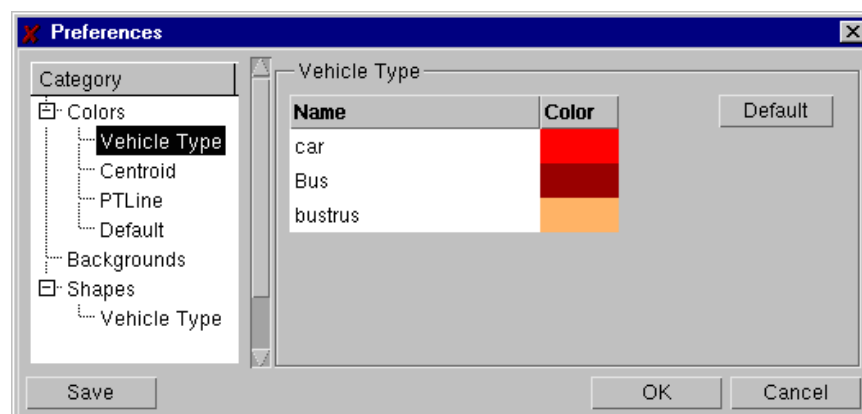
This folder corresponds to the 'Colours' category (see Figure 8-12) and is used to select the colours for highlighted objects when the user selects them: section, junction, detector, metering, VMS, centroid and vehicle. Also, the colour to be used for drawing a vehicle that is not distinguished by vehicle type can be selected here. Finally, the colour for lost vehicles is also defined in this folder.

We have removed the limitation of twenty predefined colours that we had in previous versions. Now the user can define his/her own colour palette. A minimum of 15 colours is required. By default, the simulator proposes a palette of 20 colours that the user can change at any time.

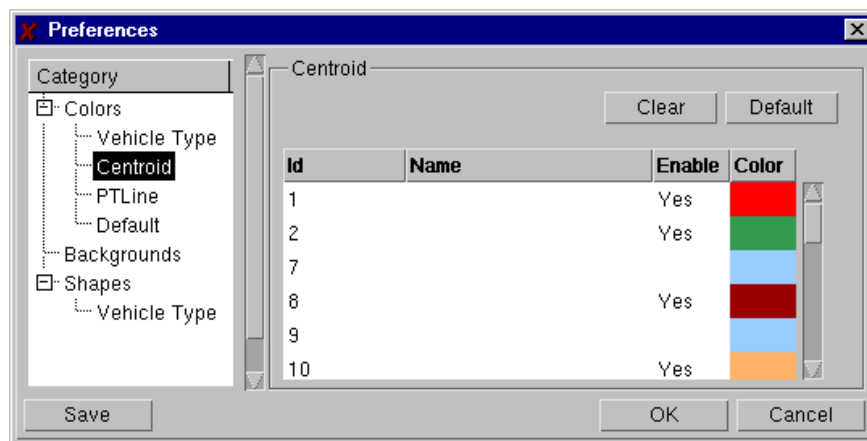
Figure 8-12: Preferences: Colours Category

The 'Colours' category is subdivided into four subcategories:

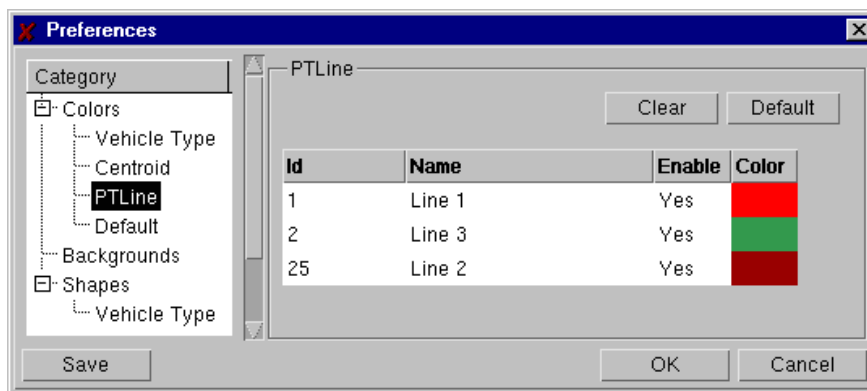
- **Vehicle Type:** to select the colours associated with each vehicle type. These colours will be used whenever the user has selected to colour different vehicle types via the 'View / Vehicle Colouring / Vehicle Type' option.

Figure 8-13: Preferences: Vehicle Types Colouring

- **Centroid:** to select the set of colours associated to each centroid and to select centroids for displaying vehicles by origin/destination.

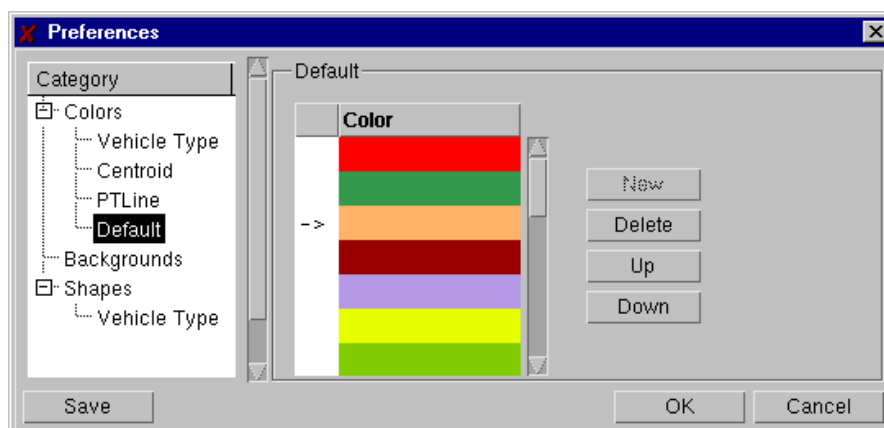
Figure 8-14: Preferences: Centroids Colouring

- **PT Line:** to select the set of colours associated to each PT Line and to select PT Lines for displaying PT Vehicles.

Figure 8-15: Preferences: PT Lines Colouring

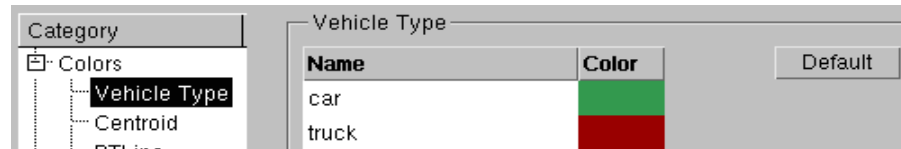
- **Default:** to define a list of default colours to be automatically assigned to objects, and the order in which they will be used.

The Default Category allows the user to define a palette of colours. With the New button is used to add colours. This button is activated if no colour is marked. To eliminate a previously selected colour you select it (click on the white column) and then press Delete

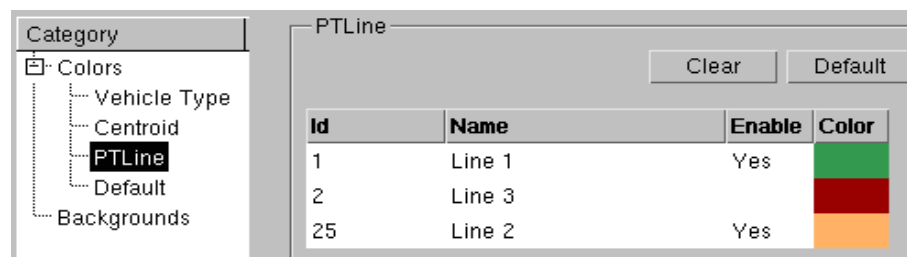
Figure 8-16: Preferences: Default Colours

The Up and Down buttons are used to change the list order. This order is the order that the application will take into account when making a default assignment of colours to the vehicle types, centroids or public transport lines. This action is activated with the Default button that you can find in the following areas:

Vehicle Type,



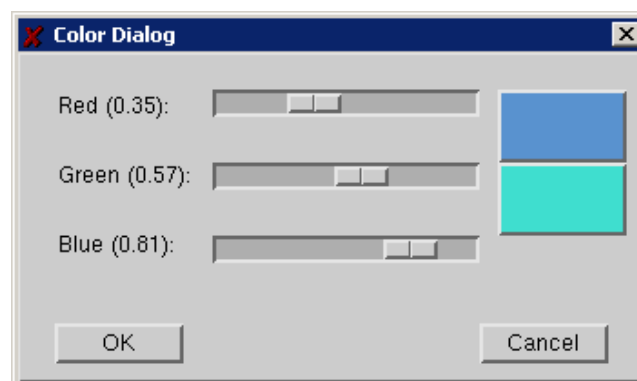
and Centroid or PTLine:



The Clear button deletes the assignments made and returns the generic centroid or public transport colour. When the user wants to assign colours by default, this will be made only to objects with the column Enable set to Yes. The objects selected (i.e. set to Yes), are the ones that will be used when selecting Vehicle Colouring per origin, destination or public transport.

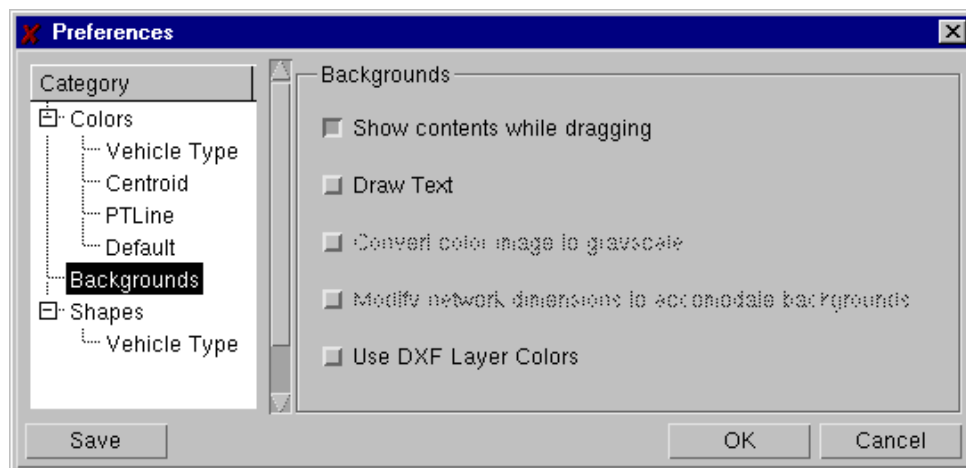
In all these dialog windows, you can modify a colour by clicking on the coloured rectangle. This will display the Colour dialog window (see Figure 8-17). Select the desired colour by moving the Red, Green and Blue bars left or right. Click on OK to accept the new colour, (as displayed in the upper rectangle). Click on Cancel to retain the previous colour, (displayed in the lower rectangle).

Figure 8-17: Colour Dialog Window



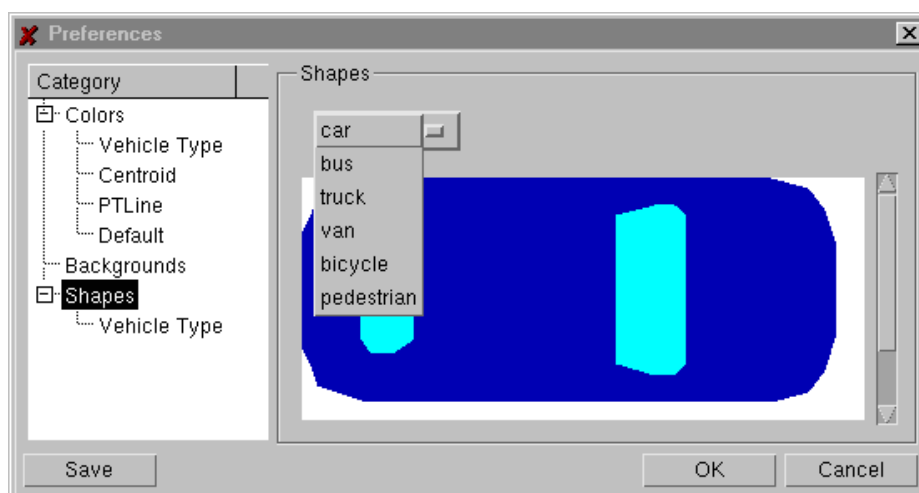
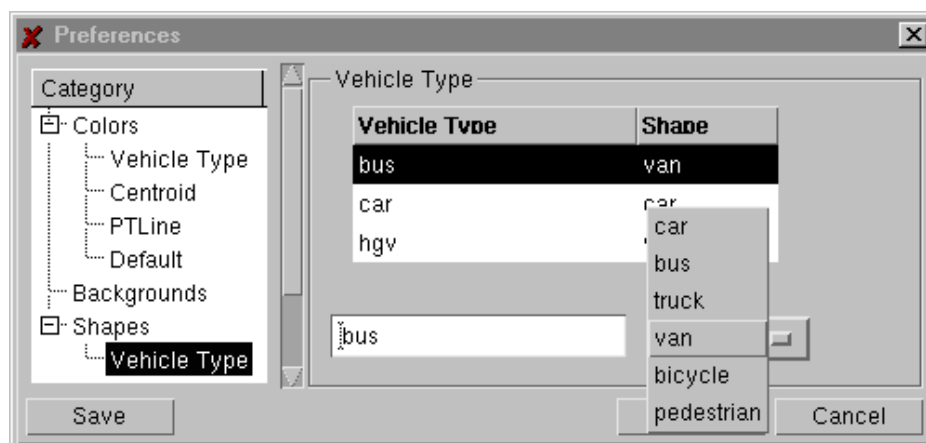
8.2.9.2 Backgrounds

Using the 'Backgrounds' the user can activate/deactivate a set of options related to the Backgrounds display. This dialog window is similar to the one described in the Tedi User Manual.

Figure 8-18: Preferences: Backgrounds

8.2.9.3 Shapes

The 'Shapes' category shows all vehicle icons available for representing cars, bus, trucks, vans, bicycles and pedestrians. Then, through the folder Shapes/Vehicle Type the user can associate each vehicle type in the model with a particular shape.

Figure 8-19: Preferences: Shapes**Figure 8-20: Preferences: Shapes. Vehicle Type.**

8.3 THE 'WINDOW' COMMAND MENU

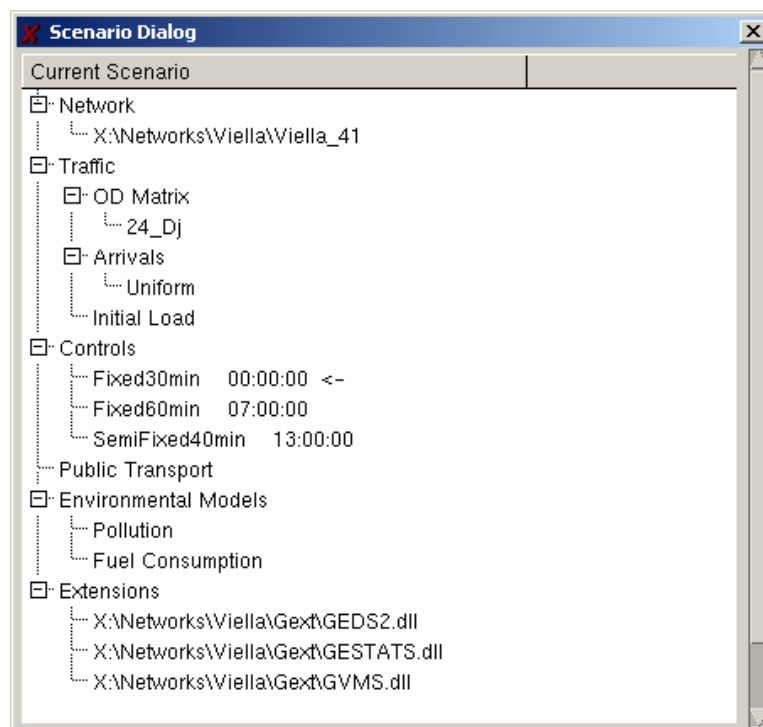
The 'Window' menu contains a set of commands used to manage additional information windows. These are Show Scenario Info, Show Legend, Show Log Window, Show/Hide Status Bar and Show/Hide Toolbar.

8.3.1 Show Scenario Info

When you select this option, the Scenario Information window is displayed (see Figure 8-21). This window contains the names of the components included in the current scenario: network, traffic result (including the states) or O/D matrix, arrivals distribution, initialisation options, list of control plan and schedule, public transport plan, environmental models and GETRAM Extension DLL's.

During simulation, an arrow indicates the state and the traffic control plan that is active at that time.

Figure 8-21: Scenario Information window



8.3.2 Show Legend

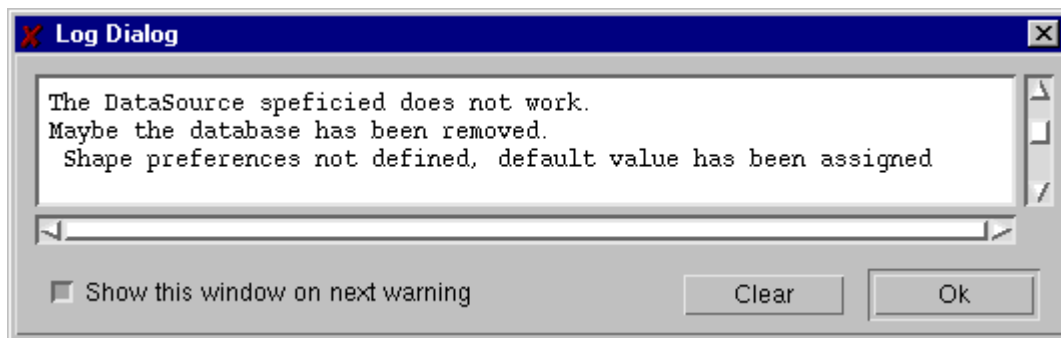
A legend can be displayed using the 'Window / Show Legend' command. The legend has its own window, with scroll bar (previously it was drawn in the same window area as the network), so you can place it anywhere on the screen and leave it open all the time if you wish.

Depending on the display option selected, the legend will display the range of colours for the densities, the colours of the vehicle types, or the colour identifying the origin or destination centroids. Figure 8-22 shows the different types of legends. To remove the Legend displayed from the window, select the 'Window / Hide Legend' command.

Figure 8-22: Legend Window

8.3.3 Show Log Window

Selecting the 'Window / Show Log Window' command displays the Log dialog window displayed in Figure 8-23. This dialog window contains the warning messages that have been issued during the current AIMSUN session. Press the 'OK' button to close the window and continue. Press the 'Clear' button to clear the window contents. To prevent the warning dialog window from appearing automatically whenever a warning message is displayed, deselect the 'Show this window on next warning' toggle button.

Figure 8-23: Log Dialog Window

8.3.4 Hide / Show Status Bar

Selecting the 'Hide / Show Status Bar' command determines whether or not the Scale and Time area located at the bottom of the AIMSUN main window is displayed. This option can be used to provide the maximum network display area.

8.3.5 Hide / Show Toolbar

Selecting the 'Hide / Show Toolbar' command determines whether or not the Tool Bar located on the leftmost side of the AIMSUN main window is displayed. This option can be used to provide the maximum network display area.

8.4 CLICKABLE DETECTION

This feature is intended to enable the user to manually activate detectors just by clicking on them during a simulation run. It is assumed that manually clicking detectors will be a method mainly used by developers of traffic control programs in the test/debug stage. For that reason it is assumed that this feature will only be available whenever the user is running a GETRAM Extension application which is performing an adaptive traffic control making use of the simulated detection.

The detection measures produced by the Clickable Detection Feature will not be considered in Aggregated Detection (detection aggregated during a time period) but only in the Cycle Detection (detection done each simulation step), which is basically used in combination with external applications. If a detector is both activated by simulated vehicles and manual clicks during a simulation step, the measures corresponding to the clicks will be disregarded and only the measures from the simulated vehicle will be considered.

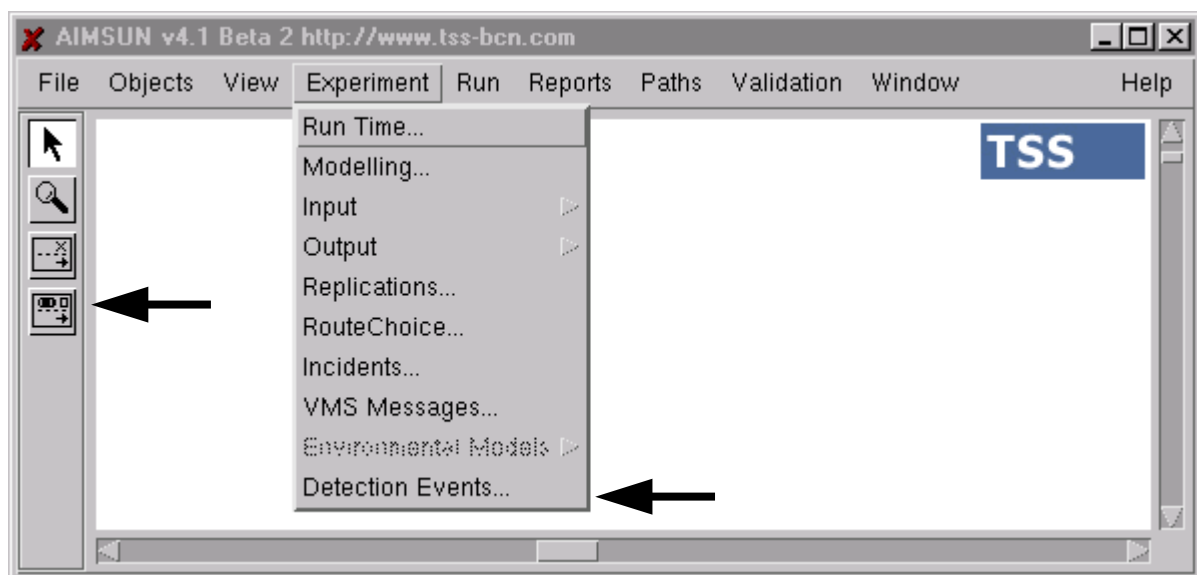
8.4.1 Enabling Clickable Detection Feature

The Clickable Detection Feature will be enabled by a GETRAM Extension Function. By default, this feature is disabled. The user can enable this feature by calling a particular function from a GETRAM Extension Application, as it is assumed that this function is only practical if there is some external control system which is able to deal with these detection events. See the document 'Detector Clickable Getram Extensions' for details of the GETRAM Extensions Functions related to clickable detection.

The Clickable Detection Feature is enabled as soon as the corresponding GETRAM Extension is loaded and is disabled when the GETRAM Extension is unloaded.

Once the Clickable Detection Feature is enabled, a new icon, called the Detection icon, will appear in the tools bar of the AIMSUN Interface, just below the Incidents icon. Also, a new command, 'Detection Events', will be included in the Experiment menu (see Figure 8-24).

Figure 8-24: Detection Icon and Detection Events command



8.4.2 Activating Clickable Detection Capability

The Detection icon is used to activate the Clickable Detection Capability during the simulation run. To deactivate Clickable Detection Capability, select any other icon. When activated, (i.e. the icon is pressed), the cursor arrow will change to a different colour to inform the user that now clicking on detectors, both single clicks and double clicks, will represent detection events. The default colour is red, but this colour can be defined by the user via the Preferences dialog window, as it is the same colour used for highlighting

occupied detectors (see Visualisation of Occupied Detectors Function). While Clickable Detection is activated, i.e. the cursor arrow is red, clicking on any network object other than detectors will have no effect.

8.4.3 Detection Events

A Detection Event is defined by the following values:

- Begin-time: time at which the detector becomes occupied
- End-time: time at which the detector becomes non-occupied
- Vehicle type
- Public Transport Line, in case the vehicle type is a PT vehicle.
- Speed of vehicle crossing the detector
- Vehicle length

Begin and End time appear in the detection events of all type of detectors, while the rest of measures are only for some detectors, depending on the detection capabilities.

Detection Events can be defined via the AIMSUN Graphical User Interface in two different ways:

- 1) by one double click on the detector, or
- 2) by two consecutive single clicks on the detector.

8.4.3.1 Double Clicks

Double-clicking on a detector opens a window (see Figure 8-25) in which the user may edit a Detection Event consisting of the following data (default values in brackets):

- 1) For all types of detectors
 - Begin-time: time at which the detector is occupied [current simulation time]
 - End-time: time at which the detector becomes non-occupied [current simulation time plus simulation step]
- 2) For some detectors, depending on the detection capabilities
 - Vehicle type, which can be selected from a list of available types that contains all types in the network including PT vehicles [first vehicle type alphabetically]
 - PT Line, in case the vehicle type is a PT vehicle, it can be selected from a list of PT Lines that crosses the detector [first PT Line in the list]
 - Speed of vehicle crossing the detector [section speed limit]
 - Vehicle length [the mean length of the selected vehicle type]

The above default values in brackets will be changed to the last defined values for each detector. Therefore, when a new detection event is defined for a particular detector, the values of vehicle type, PT Line, Speed and Length will be taken from the last detection event defined for that detector.

Figure 8-25: New Detection Event Dialog

The detection measures produced, which will depend on the detector capabilities, are as follows:

- Count: each Detection Event corresponds to 1 vehicle
- Presence: set to 1 from Begin-time to End-time
- Occupancy: set to 100% from Begin-time to End-time
- Speed: speed value defined in the dialog
- Headway: time between previous End-time and Begin-time
- Vehicle Type: vehicle type defined in the dialog
- PT Line: PT Line defined in the dialog
- Density: calculated using the length defined in the dialog

The user can define the Detection Event to occur either at the current simulation time or at a certain future simulation time. Begin and End times do not need to be multiples of the simulation step. In this way, the dialog informs the user about what is being measured and the user can either accept the default values or modify any of them. The OK button activates the detection with the selected parameters and closes the window again.

While a Detection Event is being edited, the corresponding detector is highlighted with the 'selection' colour. Double-clicks are only accepted while the simulation is paused; therefore the user should stop the simulation before opening the detection event dialog

8.4.3.2 Single Clicks

Left clicking on a detector changes the status of the detector. The first left click sets the detector status as occupied, while the second left click will change the status to non-occupied. The occupation of the detector starts or ends as soon as a new cycle step starts after the click. Minimum occupation time is one cycle step. Therefore, the single click option only allows the user to create periods of occupation that are multiples of the simulation step.

Single clicks can be made either while the simulation is stopped or running.

When you left click on a detector for the first time, a Detection Event is created. The End-Time of this detection event is undefined until you left click a second time.

- Begin-time: time of first single-click
- End-time: time of second single-click
- Vehicle type: default vehicle type
- Public Transport Line: default PT Line
- Speed: default speed
- Vehicle length: default length

The detection measures produced, which will depend on the detector capabilities, are as follows:

- Count: each Detection Event corresponds to 1 vehicle
- Presence: set to 1 during the occupied period
- Occupancy: set to 100% during the occupied period
- Speed: default speed limit
- Headway: time between previous End-time and Begin-time
- Vehicle Type: default vehicle type
- PT Line: default PT Line
- Density: calculated using the length of default vehicle type

8.4.3.3 Combining single and double clicks

A Detection Event is said to be active when the Begin-Time is smaller than current time and End-Time is either undefined or greater than current time. Therefore, an active Detection Event corresponds to a currently occupied detector.

An active Detection Event is said to be open when it has the Begin-Time defined but not the End-Time, and closed when both Begin-Time and End-Time are defined.

When you click once on a detector, either with the simulation stopped or running:

- If the detector is active and open, it is closed with End-Time set to current time.
- If the detector is active and closed, the detection event is finished: the previous value of End-Time is changed to current time.
- If the detector is not active, a new open detection event is defined, with Begin-Time equal to current time and leaving End-Time undefined. The rest of parameters take the default values

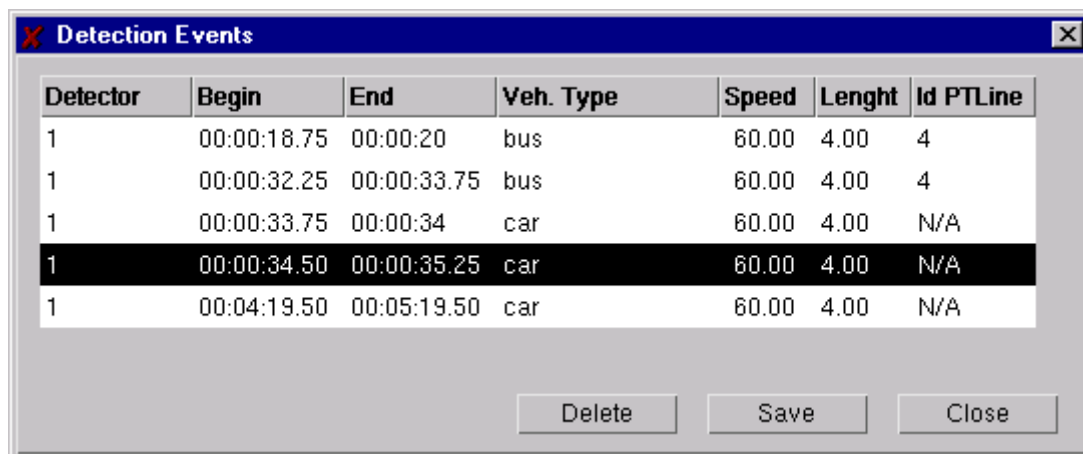
When you double click on a detector:

- If simulation is running (run or batch), it works as a single click.
- If simulation is stopped
 - If the detector is active and open, the Detection Event Dialog is open. The user can only edit the End-Time, which in any case can not be smaller than current time.
 - If the detector is either not active or active and closed, the New Detection Event Dialog is open. The user can define a new complete Detection Event.

8.4.3.4 Detection Events Log File

All detection events defined, both single-click and double-click events, can be stored in a Detection Events Log File. The idea is similar to the Incident Generation feature. AIMSUN is also able to read a Detection Events Log File, so previously stored detection events can be automatically reproduced in future simulations (similar to the incidents log file). They are ASCII files that can also be edited directly by the user if necessary.

During simulation, all Detection Events are stored in memory and they can be viewed via the Detection Events dialog. This dialog is similar to the Incidents and VMS Messages dialogs. It contains a list of all defined Detection Events and it is retrieved via the 'Experiment / Detection Events' menu command. This command will only appear in the menu if the user has enabled the Clickable Detection Feature.

Figure 8-26: Detection Events List


Detector	Begin	End	Veh. Type	Speed	Lenght	Id PTLine
1	00:00:18.75	00:00:20	bus	60.00	4.00	4
1	00:00:32.25	00:00:33.75	bus	60.00	4.00	4
1	00:00:33.75	00:00:34	car	60.00	4.00	N/A
1	00:00:34.50	00:00:35.25	car	60.00	4.00	N/A
1	00:04:19.50	00:05:19.50	car	60.00	4.00	N/A

The Detection Events dialog contains all events defined either by single or double clicking. The list is refreshed as soon as a new event is defined. Detection Events can be deleted from the list with the Delete button, but only when the simulation is stopped.

The user can save the list of Detection Events in a Log File. This can be done either using a 'Save' button that will be in the Detection Events dialog or via a call to a particular GETRAM Extension Function. At the beginning of a simulation run, the user can load a Detection Events Log File. This will only be done via a call to a particular GETRAM Extension Function.

8.4.3.5 Detection events visualisation

The detectors will be coloured with a different colour than the generic when the user selects one of the two next options:

- *Show occupied detectors.* Will be coloured differently from the detectors that are detecting a vehicle at current time and, only if there is a GETRAM extension loaded that allows detection events, those detectors of the events list that accomplish Initial Time \leq current simulation time $<$ End Time.
- With the detection event toggle activated. In this case all the detectors from the list events that accomplish Initial Time \leq current simulation time $<$ end Time will be displayed with a different colour.

9. SIMULATION OUTPUTS

This chapter provides a detailed description of all output provided by the AIMSUN microsimulator: Statistical Results and Detection data.

9.1 STATISTICAL SIMULATION RESULTS

In addition to the animated display of the simulation, AIMSUN also provides as output some statistical measures, such as Flow, Speed, Density, Travel and Delay Time. Prior to a simulation experiment, the user may select which statistics are required and how they are to be gathered via the 'Experiment / Output / Statistics' command. The user can also specify how and where to store the results via the 'Experiment / Output / Output Location' command.

The 'Reports' command from the menu bar has three options for visualising statistical results obtained from current or previous simulation experiments. The options available are 'Reports / Files Reports', 'Reports / Current Report' and 'Reports / Current Graphics'.

9.1.1 Statistical Traffic Measures

The statistical traffic measures provided by AIMSUN can be specified at different levels of aggregation: for the whole system, for each section, for each turning movement, for every stream (set of consecutive sections) defined by the user. In the Route-Based model, information by origin, destination or O/D pair can be also obtained. If the public transport model is being used, it provides statistical traffic measures for each public transport line.

The statistical measures can be presented according to two time scopes:

- Global: Statistical data gathered from the beginning to the end of the simulation experiment.
- Periodic: Statistical data gathered during certain time periods (defined by the user). After each period the statistics area is cleared.

N.B. In the following descriptions, all measurements have been expressed using the Metric system. If the network has been defined as using English system units, the user should read miles instead of kilometres and feet instead of meters.

System Statistics

The statistics gathered by AIMSUN at the system level, (i.e. referring to the entire network), are as follows:

- Mean Flow: average number of vehicles per hour that have passed through the network during the simulation period. The vehicles are counted when leaving the network via an exit section.
- Density: average number of vehicles per kilometre for the whole network.
- Mean Speed: average speed for all vehicles that have left the system. This is calculated using the mean journey speed for each vehicle.
- Harmonic Mean Speed: harmonic mean speed for all vehicles that have left the system.
- Travel Time: average time a vehicle needs to travel one kilometre inside the network. This is the mean of all the single travel times (exit time - entrance time) for every vehicle that has crossed the network, converted into time per kilometre.
- Delay Time: average delay time per vehicle per kilometre. This is the difference between the expected travel time (the time it would take to traverse the system under ideal conditions) and the travel time. It is calculated as the average of all vehicles and then converted into time per kilometre.
- Stop Time: average time at standstill per vehicle per kilometre.
- Number of Stops: average number of stops per vehicle per kilometre.
- Total Travel: total number of kilometres travelled by all the vehicles that have crossed the network.
- Total Travel Time: total travel time experimented by all the vehicles that have crossed the network.
- This is only provided when an environmental model is set to ON.

- Fuel Consumed: total litres of fuel consumed by all the vehicles that have crossed the network. This is only provided when the particular model 'Fuel Consumption' is set to ON.
- Pollution Emitted: for each pollutant, total kilograms of pollution emitted by all the vehicles that have crossed the network. It is only provided when the particular model 'Pollution Emission' is set to ON.

Section and Turn Statistics

At the section level, the vehicle data is gathered when they a vehicle leaves a section, after it has finished the turning movement and entered the next section. The statistics provided by AIMSUN at the section level and turning level are the following:

- Mean Flow: average number of vehicles per hour that have crossed the section during the simulation period.
- Density: average number of vehicles per kilometre in the section. No density measure is provided for turnings.
- Mean Speed: average speed for all vehicles that have traversed the section. This is calculated using the mean speed for the section journey for each vehicle.
- Harmonic Mean Speed: harmonic mean speed for all vehicles that have traversed the section.
- Travel Time: average time a vehicle needs to cross the section. This is the mean of all the single travel times (section exit time - section entrance time) of every vehicle that has left the section.
- Delay Time: average delay time per vehicle. This is the difference between the expected travel time (time it would take to traverse the section under ideal conditions) and the travel time. It is calculated as the average of all vehicles.
- Stop Time: average time at a standstill per vehicle while travelling in the section.
- Number of Stops: average number of stops per vehicle while travelling in the section.
- Mean Queue Length: average length of the queue in that section, expressed as the number of vehicles per lane. It is calculated as a time average.
- Maximum Queue Length: maximum length of the queue in this section, expressed as number of vehicles per lane.
- Total Travel: total number of kilometres travelled by all the vehicles in this section.
- Total Travel Time: total travel time experimented by all the vehicles in this section.
- Fuel Consumed: total litres of fuel consumed inside the section by all the vehicles that have crossed it. This is only provided when the particular 'Fuel Consumption' model is set to ON.
- Pollution Emitted: for each pollutant, total kilograms of pollution emitted inside the section by all the vehicles that have crossed it. This is only provided when the particular 'Pollution Emission' model is set to ON.

If turning information has to be generated, the vehicle data gathered is dissociated for each turning to calculate the mean flow, mean speed, harmonic mean speed, travel time, delay time, stop time, number of stops, queue length, total travel, fuel consumption and pollution emission per turning.

If the model contains polysections, the information could be generated either considering each individual section or aggregating by polysection, that is aggregate the information of all sections that belongs to the same polysection. When this information is aggregated by polysection, the polysection is identified by the last section identifier downstream.

Stream Statistics

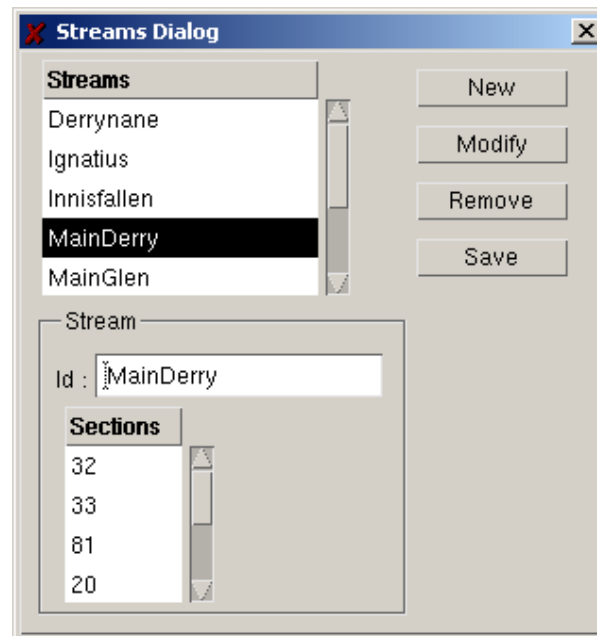
A *Stream* is a set of sections that are consecutive. This means a chain of sections that are connected either through junction turnings or through joins. *Streams* are not to be confused with *Paths*. The only purpose of a Stream is to provide a set of aggregated statistical measurements, although these measurements are just estimates, given that no path information is stored in the vehicles, so no track is kept of which particular vehicles have followed the whole stream. Therefore, the stream measurements are calculated as an average of the statistical data per turn of the sections composing it, and not using the data gathered from each vehicle that crosses the stream, as is done for section statistics.

The user can define Streams in the network via the AIMSUN User Interface. To view or edit streams, select the 'Objects / Streams' command and the 'Streams' definition window is displayed (see Figure 9.1). The list

of Stream names defined, if any, will be shown. Clicking on a stream name causes the sections composing the stream to be displayed in the Sections list box and also to be highlighted in the network.

The user can define new streams by pressing the 'New' button, typing in the Stream Identifier (or name) and selecting the sections composing the stream by clicking on them directly in the network. The sections must be selected in the correct order, indicating the stream direction. After selecting the sections, click on the 'Add' button to include the stream in the list box. The user may also remove or modify the streams and save them to the disk if required for future experiments.

Figure 9-1: Streams Definition window



N.B. To calculate the averages for the stream, statistical data per turn is used instead of statistical data per section. Statistical data per turn is calculated separately for each turning movement of the section, taking into account the data gathered for the vehicles that perform each of the turns. Statistical data per section is calculated taking into account the data gathered for all the vehicles that cross the section, independently of the turn they make.

The statistics provided by AIMSUN at the Stream level are the following:

- Mean Flow: average number of vehicles per hour that have crossed the stream during the simulation period. This is obtained using the flow of the first section of the stream and applying the corresponding turning proportions following the sections composing the stream.
- Density: average number of vehicles per kilometre for the stream. This is calculated as the average of densities of all the sections composing the stream.
- Mean Speed: average speed for the stream. This is calculated as the average of the mean speeds for the corresponding turns of all the sections composing the stream.
- Harmonic Mean Speed: harmonic mean speed.
- Travel Time: average time it takes a vehicle to cross the stream. This is the sum of mean travel times for the corresponding turns of all the sections composing the stream.
- Delay Time: average delay time per vehicle. This is the difference between the expected travel time (time it would take to traverse the stream under ideal conditions) and the travel time. It is calculated as the sum of mean delay times for the corresponding turns of all the sections composing the stream.
- Stop Time: average time spent at a stop per vehicle while travelling along the stream. This is calculated as the sum of mean stop times for the corresponding turns of all the sections composing the stream.
- Number of Stops: average number of stops per vehicle while travelling along the stream. This is calculated as the sum of the number of stops for the corresponding turns of all the sections composing the stream.

- Mean Queue Length: average length of the queue in number of vehicles per lane. This is calculated as the sum of the mean queue length for the corresponding turns of all the sections composing the stream.
- Maximum Queue Length: maximum length of the queue in number of vehicles per lane. This is calculated as the sum of the maximum queue length for the corresponding turns of all the sections composing the stream.
- Total Travel: total number of kilometres travelled by all the vehicles in this stream.
- Total Travel Time: total travel time experimented by all the vehicles in this stream.
- Fuel Consumed: total litres of fuel consumed along the stream by all the vehicles that have crossed it. This is only provided when the particular model 'Fuel Consumption' is set to 'ON'. It is calculated as the sum of Fuel Consumed by vehicles in the corresponding turns of all the sections composing the stream.
- Pollution Emitted: for each pollutant, total kilograms of pollution emitted inside the stream by all the vehicles that have crossed it. It is only provided when the particular model 'Pollution Emission' is set to 'ON'. It is calculated as the sum of Pollution Emitted by vehicles in the corresponding turns of all the sections composing the stream.

O/D Matrix Statistics

The statistics provided either by Origin centroid, Destination centroid or O/D pair are the following:

- Flow: number of vehicles that have reached the destination during the simulation period. The vehicles are counted when leaving the network via an exit section.
- Mean Speed: average speed for all vehicles that have reached the destination. This is calculated using the mean journey speed for each vehicle.
- Harmonic Mean Speed: harmonic mean speed for all vehicles that have reached the destination.
- Travel Time: average time it takes a vehicle to travel from the origin to the destination. This is the mean of all the single travel times (exit time - entrance time) for each vehicle.
- Delay Time: average delay time per vehicle. This is the difference between the expected travel time (time it would take to go from the origin to the destination under ideal conditions) and the actual travel time.
- Stop Time: average time spent at a stop per vehicle during the trip from origin to destination.
- Number of Stops: average number of stops per vehicle during the trip.
- Total Travel: total number of kilometres travelled by all the vehicles that have done the trip.
- Total Travel Time: total travel time experimented by all the vehicles that have done the trip.
- Fuel Consumed: total litres of fuel consumed by all the vehicles that have done the trip. This is only provided when the particular model 'Fuel Consumption' is set to 'ON'.
- Pollution Emitted: for each pollutant, total kilograms of pollution emitted by all the vehicles that have done the trip. This is only provided when the particular model 'Pollution Emission' is set to 'ON'.
- Lost Vehicles: total number of vehicles that have been lost while trying to do the trip from origin to destination, and which have therefore not reached the correct destination.

Public Transport Statistics

The statistics provided for each public transport line are the following:

- Flow: number of vehicles that have reached the end of the public transport line during the simulation period. The vehicles are counted when going out of the network via the last section line.
- Mean Speed: average speed for all vehicles that have reached the end of the public transport line. This is calculated using the mean journey speed for each vehicle.
- Harmonic Mean Speed: harmonic mean speed for all vehicles that have reached the end of the public transport line.
- Travel Time: average time it takes for a vehicle to travel along a public transport line. This is the mean of all the single travel times (exit time - entrance time) for each vehicle.
- Delay Time: average delay time per vehicle to make the trip. This is the difference between the expected travel time (time it would take to go from the origin to the destination under ideal conditions) and the actual travel time.
- Stop Time: average time spent at a stop per vehicle during the trip.
- Number of Stops: average number of stops per vehicle during the trip.
- Total Travel: total number of kilometres travelled by all the vehicles that have made the trip.
- Total Travel Time: total travel time experimented by all the vehicles that have made the trip.

- Fuel Consumed: total litres of fuel consumed by all the vehicles that have done the trip. This is only provided when the particular model 'Fuel Consumption' is set to 'ON'.
- Pollution Emitted: for each pollutant, total kilograms of pollution emitted by all the vehicles that have made the trip. This is only provided when the particular model 'Pollution Emission' is set to 'ON'.

9.1.2 Calculation of Traffic Statistics

This section describes in detail the procedures applied to calculate the different traffic statistical measures. The individual vehicle data that needs to be gathered in order to produce the traffic statistics is also described.

9.1.2.1 System Statistics

Vehicle Data Gathering

The information gathered from an individual vehicle is:

- TEN_i = Entrance time of vehicle i-th in the system (seconds).
- TEX_i = Exit time of vehicle i-th from the system (seconds).
- D_i = Total Distance travelled by vehicle i-th in the system (meters).
- TDT_i = Total Delay Time accumulated in each section by vehicle i-th (seconds).
- TST_i = Total Stop Time accumulated in each section by vehicle i-th (seconds).
- TNS_i = Total Number of Stops accumulated in each section by vehicle i-th .
- TFC_i = Total fuel consumed by vehicle i-th (litres).
- TPE_{ij} = Total emission of pollutant j-th by vehicle i-th (gr).

Taking this information, when a vehicle exits the system, the following variables are calculated:

TT_i = Average travel time per Km of vehicle i-th (seconds).

$$TT_i = \frac{TEX_i - TEN_i}{D_i} * 1000$$

DT_i = Average delay time per Km of vehicle i-th (seconds).

$$DT_i = \frac{TDT_i}{D_i} * 1000$$

S_i = Average speed of vehicle i-th (m/s).

$$S_i = \frac{D_i}{TEX_i - TEN_i}$$

HS_i = Inverse of S_i (s/m), used for calculating the harmonic mean speed

$$HS_i = \frac{1}{\frac{D_i}{TEX_i - TEN_i}}$$

ST_i = Average stop time per Km of vehicle i-th (seconds).

$$ST_i = \frac{TST_i}{D_i} * 1000$$

NS_i = Average number of stops per Km of vehicle i-th .

$$NS_i = \frac{TNS_i}{D_i} * 1000$$

Traffic Statistics

The previous Vehicle Variables are used for calculating the simulation statistical output. The following traffic statistics are calculated for every statistical interval and for the whole simulation period:

I = Interval of statistics (seconds).

N_{sys} = Number of vehicles that exit the system during period I

F_{sys} = Mean Flow

$$F_{sys} = \frac{N_{sys}}{I} * 3600$$

TT_{sys} = Average Travel Time per vehicle per Km

$$TT_{sys} = \frac{\sum_{i=1}^{N_{sys}} TT_i}{N_{sys}}$$

S_{sys} = Average Speed per vehicle (Km/h)

$$S_{sys} = \frac{\sum_{i=1}^{N_{sys}} S_i}{N_{sys}} * 3.6$$

HS_{sys} = Harmonic mean Speed per vehicle (Km/h)

$$HS_{sys} = \frac{N_{sys}}{\sum_{i=1}^{N_{sys}} \frac{1}{S_i}} * 3.6$$

DT_{sys} = Average Delay Time per vehicle per Km (seconds/Km)

$$DT_{sys} = \frac{\sum_{i=1}^{N_{sys}} DT_i}{N_{sys}}$$

ST_{sys} = Average Stop Time per vehicle per Km (seconds/Km)

$$ST_{sys} = \frac{\sum_{i=1}^{N_{sys}} ST_i}{N_{sys}}$$

NS_{sys} = Average Number of Stops per vehicle per Km

$$NS_{sys} = \frac{\sum_{i=1}^{N_{sys}} NS_i}{N_{sys}}$$

$TotalTrav_{sys}$ = Total number of kilometres travelled by all the vehicles that have crossed the network (Km)

$$TotalTrav_{sys} = \sum_{i=1}^{N_{sys}} D_i / 1000$$

$TotalTravTime_{sys}$ = Total travel time experimented by all the vehicles that have crossed the network (seconds)

$$TotalTravTime_{sys} = \sum_{i=1}^{N_{sys}} TEX_i - TEN_i$$

$FuelCon_{sys}$ = Total fuel consumed by all the vehicles that have crossed the network (litres)

$$FuelCon_{sys} = \sum_{i=1}^{N_{sys}} TFC_i$$

$PollEm_{sys,j}$ = Total pollution of pollutant j-th emitted by all the vehicles that have crossed the network (Kg)

$$PollEm_{sys,j} = \sum_{i=1}^{N_{sys}} TPE_{i,j}$$

The lane density of the system is calculated as follows:

L = Total length of all lanes of all sections of the network (m)

$NVeh_t$ = Number of vehicles in the system at time t

I = Interval of statistics (seconds).

$T = (0, t_1, \dots, t_m, I)$: instants when the number of vehicles in the system changes

$$DEN_{sys} = \frac{\sum_{t_i \in T} NVeh_{t_{(i-1)}} * (t_i - t_{(i-1)})}{L} * 1000$$

9.1.2.2 Turning and Section Statistics

Vehicle Data Gathering

The information gathered from an individual vehicle at every section is:

TEN_{s_i} = Entrance time of vehicle i-th in the section (seconds).

TEX_{s_i} = Exit time of vehicle i-th from the section (seconds).

TST_{s_i} = Total Stop Time accumulated in a section by vehicle i-th (seconds).

TNS_{s_i} = Total Number of Stops accumulated in a section by vehicle i-th.

TFC_{s_i} = Total fuel consumed accumulated in a section by vehicle i-th (litres).

$TPE_{s_i,j}$ = Total emission of pollutant j-th accumulated in a section by vehicle i-th (gr).

Taking this information, when a vehicle enters into a new section the following variables are calculated:

TT_i = Average section travel time of vehicle i-th (seconds).

$$TT_i = TEX_{s_i} - TEN_{s_i}$$

DT_i = Average section delay time of vehicle i-th (seconds).

$$DT_i = TT_i - \left[\frac{L_s}{\text{Min}(S_{Max_i}, S_s * \theta_i)} + \frac{L_t}{\text{Min}(S_{Max_i}, S_t * \theta_i)} \right]$$

where S_s = Speed limit of section s (m/s).

S_t = Speed limit of turning t (m/s).

θ_i = Speed acceptance of vehicle i

S_{Max_i} = Maximum desired Speed of vehicle i (m/s).

L_s = Distance of section s (meters).

L_t = Distance of turning t (meters).

S_i = Average section speed of vehicle i-th (m/s).

$$S_i = \frac{L_s + L_t}{TT_i}$$

HS_i = Inverse of S_i (s/m), used for calculating the harmonic mean speed.

$$HS_i = \frac{1}{\frac{L_s + L_t}{TT_i}}$$

ST_i = Stop time in the section of vehicle i-th (seconds).

$$ST_i = TSTs_i$$

NS_i = Number of stops of vehicle i-th .

$$NS_i = TNSs_i$$

Turning Traffic Statistics

The previous Vehicle Variables are used for calculating the turning statistical output. The following traffic statistics are calculated for every statistical interval and for the whole simulation period:

I = Interval of statistics (seconds).

N_{tur} = Number of vehicles that exit a turning

F_{tur} = Mean Flow of a turning (veh/h)

$$F_{tur} = \frac{N_{tur}}{I} * 3600$$

TT_{tur} = Average Travel Time per vehicle of a turning

$$TT_{tur} = \frac{\sum_{i=1}^{N_{tur}} TT_i}{N_{tur}}$$

S_{tur} = Average Speed per vehicle of a turning (Km/h)

$$S_{tur} = \frac{\sum_{i=1}^{N_{tur}} S_i}{N_{tur}} * 3.6$$

HS_{tur} = Harmonic mean Speed per vehicle of a turning (Km/h)

$$HS_{tur} = \frac{N_{tur}}{\sum_{i=1}^{N_{tur}} HS_i} * 3.6$$

DT_{tur} = Average Delay Time per vehicle of a turning

$$DT_{tur} = \frac{\sum_{i=1}^{N_{tur}} DT_i}{N_{tur}}$$

ST_{tur} = Average Stop Time per vehicle of a turning

$$ST_{tur} = \frac{\sum_{i=1}^{N_{tur}} ST_i}{N_{tur}}$$

NS_{tur} = Average Number of Stops per vehicle of a turning

$$NS_{tur} = \frac{\sum_{i=1}^{N_{tur}} NS_i}{N_{tur}}$$

$TotalTrav_{tur}$ = Total distance travelled by all the vehicles in the turning (Km)

$$TotalTrav_{tur} = \sum_{i=1}^{N_{tur}} D_i / 1000$$

$TotalTravTime_{tur}$ = Total travel time experimented by all the vehicles in the turning (seconds)

$$TotalTravTime_{tur} = \sum_{i=1}^{N_{tur}} TT_i$$

$FuelCon_{tur}$ = Total fuel consumed by all the vehicles in the turning (litres)

$$FuelCon_{tur} = \sum_{i=1}^{N_{tur}} TFC_i$$

$PollEm_{tur,j}$ = Total pollution of pollutant j-th emitted by all the vehicles inside the turning (Kg)

$$PollEm_{tur,j} = \sum_{i=1}^{N_{tur}} TPE_{i,j}$$

To calculate the average and maximum queue length of a turning (veh):

$QL_{l,t}$ = Queue Length in the lane l at time t

$MaxQL_l$ = Maximum Queue Length in the lane l

$NBTurns_l$ = Total number of allowed turnings movements of the lane l

I = Interval of statistics (seconds).

$T_l = (0, t_{l,1}, \dots, t_{l,m}, I)$: instants when the queue length in lane l changes

$NBLanes_{tur}$ = Number of lanes that belong to turning tur

$$AQL_{tur} = \frac{\sum_{l \in tur} \left[\left(\sum_{t_i \in T_l} [QL_{l,t(i-1)} * (t_{l,i} - t_{l,(i-1)})] / I \right) / NBTurns_l \right]}{NBLanes_{tur}}$$

$$MaxQL_{tur} = \frac{\sum_{l \in tur} (MaxQL_l / NBTurns_l)}{NBLanes_{tur}}$$

Section Traffic Statistics

The previous Vehicle Variables are used for calculating the section statistical output. The following traffic statistics are calculated for every statistical interval and for the whole simulation period:

I = Interval of statistics (seconds).

N_{sec} = Number of vehicles that exit a section

F_{sec} = Mean Flow of a section (veh/h)

$$F_{sec} = \frac{N_{sec}}{I} * 3600$$

TT_{sec} = Average Travel Time per vehicle of a section

$$TT_{sec} = \frac{\sum_{i=1}^{N_{sec}} TT_i}{N_{sec}}$$

S_{sec} = Average Speed per vehicle of a section (Km/h)

$$S_{sec} = \frac{\sum_{i=1}^{N_{sec}} S_i}{N_{sec}} * 3.6$$

HS_{sec} = Harmonic mean Speed per vehicle of a section (Km/h)

$$HS_{sec} = \frac{N_{sec}}{\sum_{i=1}^{N_{sec}} HS_i} * 3.6$$

DT_{sec} = Average Delay Time per vehicle of a section

$$DT_{sec} = \frac{\sum_{i=1}^{N_{sec}} DT_i}{N_{sec}}$$

ST_{sec} = Average Stop Time per vehicle of a section

$$ST_{sec} = \frac{\sum_{i=1}^{N_{sec}} ST_i}{N_{sec}}$$

NS_{sec} = Average Number of Stops per vehicle of a section

$$NS_{sec} = \frac{\sum_{i=1}^{N_{sec}} NS_i}{N_{sec}}$$

$TotalTrav_{sec}$ = Total distance travelled by all the vehicles in the section (Km)

$$TotalTrav_{sec} = \sum_{i=1}^{N_{sec}} D_i / 1000$$

$TotalTravTime_{sec}$ = Total travel time experimented by all the vehicles in the section (seconds)

$$TotalTravTime_{sec} = \sum_{i=1}^{N_{sec}} TT_i$$

$FuelCon_{sec}$ = Total fuel consumed by all the vehicles in the section (litres)

$$FuelCon_{sec} = \sum_{i=1}^{N_{sec}} TFC_i$$

$PollEm_{sec,j}$ = Total pollution of pollutant j-th emitted by all the vehicles inside the section (Kg)

$$PollEm_{sec,j} = \sum_{i=1}^{N_{sec}} TPE_{i,j}$$

To calculate the density of a section:

L_l = Length of lane l (m)

$NVeh_{l,t}$ = Number of vehicles in the lane l at time t

I = Interval of statistics (seconds).

$T_l = (0, t_{l,1}, \dots, t_{l,m}, I)$: instants when the number of vehicles in lane l changes

$$DEN_{sec} = \frac{\sum_{l \in sec} \left(\sum_{t_i \in T_l} \left[NVeh_{l,t(i-1)} * (t_{l,i} - t_{l,(i-1)}) \right] / I \right)}{\sum_{l \in sec} L_l} * 1000$$

To calculate the average and maximum queue length of a section (veh):

$QL_{l,t}$ = Queue Length in the lane l at time t

$MaxQL_l$ = Maximum Queue Length in the lane l

I = Interval of statistics (seconds).

$T_l = (0, t_{l,1}, \dots, t_{l,m}, I)$: instants when the queue length in lane l changes

$NBLanes_{sec}$ = Number of lanes of section *sec*

$$AQL_{sec} = \frac{\sum_{l \in sec} \left(\sum_{t_i \in T_l} \left[QL_{l,t(i-1)} * (t_{l,i} - t_{l,(i-1)}) \right] / I \right)}{NBLanes_{sec}}$$

$$MaxQL_{sec} = \frac{\sum_{l \in sec} MaxQL_l}{NBLanes_{sec}}$$

9.1.2.3 Streams Statistics

Data Gathering

The stream measurements are calculated as an average of the statistical data per turn of the sections composing it, and not using the data gathered from each vehicle that crosses the stream, as is done for all other statistics.

Stream Statistics

I = Interval of statistics (seconds).

Turns_{str} = Set of turnings composing the stream

NbTurns_{str} = Number of turnings composing the stream

Sects_{str} = Number of sections composing the stream

NbSects_{str} = Number of sections composing the stream

F_{str} = Mean Flow of a stream (veh/h), where F_{tur(i)} is the flow of turning i-th and F_{sec(i)} is the flow of section i-th

$$\begin{aligned} F_{str} &= F_{tur(0)} \\ \text{for } i &= 1 \text{ to } NbSects_{str} - 1 \\ F_{str} &= F_{tur(i)} * (F_{str} / F_{sec(i)}) \\ \text{endfor} \end{aligned}$$

TT_{str} = Average Travel Time per vehicle of a stream (seconds), where TT_{tur(i)} is the travel time of turning i-th and TT_{sec(i)} is the travel time of section i-th

$$TT_{str} = \frac{NbTurns_{str}}{\sum_{i=1} TT_{tur(i)} + TT_{sec}(NbSects_{str})}$$

S_{str} = Average Speed per vehicle of a stream (Km/h), where S_{tur(i)} is the average speed of turning i-th, S_{sec(i)} is the average speed of section i-th, L_{tur(i)} is the length of the turning i-th including the origin section of the turning and L_{sec(i)} is the length of the section i-th

$$S_{str} = \frac{\sum_{i=1}^{NbTurns_{str}} (S_{tur(i)} * L_{tur(i)}) + S_{sec}(NbSects_{str}) * L_{sec}(NbSects_{str})}{\sum_{i=1}^{NbTurns_{str}} L_{tur(i)} + L_{sec}(NbSects_{str})}$$

HS_{str} = Harmonic mean Speed per vehicle of a section (Km/h), where $HS_{tur(i)}$ is the harmonic mean speed of turning i-th and $HS_{sec(i)}$ is the harmonic mean speed of section i-th.

$$HS_{str} = \frac{NbSects_{str}}{\sum_{i=1}^{NbTurns_{str}} \left(\frac{1}{HS_{tur(i)}} \right) + \frac{1}{HS_{sec(NbSects_{str})}}}$$

DT_{str} = Average Delay Time per vehicle of a stream (seconds), where $DT_{tur(i)}$ is the delay time of turning i-th and $DT_{sec(i)}$ is the delay time of section i-th

$$DT_{str} = \sum_{i=1}^{NbTurns_{str}} DT_{tur(i)} + DT_{sec(NbSects_{str})}$$

ST_{str} = Average Stop Time per vehicle of a stream (seconds), where $ST_{tur(i)}$ is the delay time of turning i-th and $ST_{sec(i)}$ is the delay time of section i-th

$$ST_{str} = \sum_{i=1}^{NbTurns_{str}} ST_{tur(i)} + ST_{sec(NbSects_{str})}$$

NS_{str} = Average Number of Stops per vehicle of a stream (seconds), where $NS_{tur(i)}$ is the average number of stops of turning i-th and $NS_{sec(i)}$ is the average number of stops of section i-th

$$NS_{str} = \sum_{i=1}^{NbTurns_{str}} NS_{tur(i)} + NS_{sec(NbSects_{str})}$$

$TotalTrav_{str}$ = Total distance travelled by all the vehicles in the stream (Km)

$$TotalTrav_{str} = \sum_{i=1}^{NbTurns_{str}} TotalTrav_{tur(i)} + TotalTrav_{sec(NbSects_{str})}$$

$TotalTravTime_{str}$ = Total travel time experimented by all the vehicles in the stream (seconds)

$$TotalTravTime_{str} = \sum_{i=1}^{NbTurns_{str}} TotalTravTime_{tur(i)} + TotalTravTime_{sec(NbSects_{str})}$$

$FuelCon_{str}$ = Total fuel consumed by all the vehicles in the stream (litres)

$$FuelCon_{str} = \sum_{i=1}^{NbTurns_{str}} FuelCon_{tur(i)} + FuelCon_{sec(NbSects_{str})}$$

$PollEm_{str,j}$ = Total pollution of pollutant j-th emitted by all the vehicles in the stream (Kg)

$$PollEm_{str,j} = \sum_{i=1}^{NbTurns_{str}} PollEm_{tur(i),j} + PollEm_{sec(NbSects_{str}),j}$$

To calculate the density of a stream (veh/Km):

$L_{sec(i)}$ = Length of section i-th (m)

$$DEN_{str} = \frac{\sum_{i=1}^{NbSects_{str}} DEN_{sec(i)} * L_{sec(i)}}{\sum_{i=1}^{NbSects_{str}} L_{sec(i)}}$$

To calculate the average and maximum queue length of a stream (veh):

$L_{sec(i)}$ = Length of section i-th (m)

$$AQL_{str} = \sum_{i=1}^{NbTurns_{str}} AQL_{tur(i)} + AQL_{sec}(NbSects_{str})$$

$$MaxQL_{str} = \sum_{i=1}^{NbTurns_{str}} MaxQL_{tur(i)} + MaxQL_{sec}(NbSects_{str})$$

9.1.2.4 OD Statistics

Vehicle Data Gathering

The information gathered from an individual vehicle is as follows:

TEN_i = Entrance time of vehicle i-th in the system (seconds).

TEX_i = Exit time of vehicle i-th from the system (seconds).

D_i = Total Distance travelled by vehicle i-th in the system (meters).

TDT_i = Total Delay Time accumulated in each section by vehicle i-th (seconds).

TST_i = Total Stop Time accumulated in each section by vehicle i-th (seconds).

TNS_i = Total Number of Stops accumulated in each section by vehicle i-th .

TFC_i = Total fuel consumed accumulated in each section by vehicle i-th (litres).

TPE_{ij} = Total emission of pollutant j-th accumulated in each section by vehicle i-th

Taking this information, when a vehicle exits the system the following variables are calculated:

TT_i = Average travel time of vehicle i-th (seconds).

$$TT_i = TEX_i - TEN_i$$

DT_i = Average delay time of vehicle i-th (seconds).

$$TT_i = TDT_i$$

S_i = Average speed of vehicle i-th (m/s).

$$S_i = \frac{D_i}{TEX_i - TEN_i}$$

HS_i = Inverse of S_i (s/m), used for calculating the harmonic mean speed

$$HS_i = \frac{1}{\frac{D_i}{TEX_i - TEN_i}}$$

ST_i = Average stop time of vehicle i-th (seconds).

$$ST_i = TST_i$$

NS_i = Average number of stops of vehicle i -th .

$$NS_i = TNS_i$$

OD Pair Statistics

The previous Vehicle Variables are used for calculating the OD pair statistical output. The following traffic statistics are calculated for every statistical interval and for the whole simulation period:

I = Interval of statistics (seconds).

$N_{(orig,dest)}$ = Number of vehicles that have reached its destination *dest* from its origin *orig*.

$TT_{(orig,dest)}$ = Average Travel Time per vehicle (seconds)

$$TT_{(orig,dest)} = \frac{\sum_{i=1}^{N_{(orig,dest)}} TT_i}{N_{(orig,dest)}}$$

$S_{(orig,dest)}$ = Average Speed per vehicle (Km/h)

$$S_{(orig,dest)} = \frac{\sum_{i=1}^{N_{(orig,dest)}} S_i}{N_{(orig,dest)}} * 3.6$$

$HS_{(orig,dest)}$ = Harmonic mean Speed per vehicle (Km/h)

$$HS_{(orig,dest)} = \frac{N_{(orig,dest)}}{\sum_{i=1}^{N_{(orig,dest)}} \frac{1}{S_i}} * 3.6$$

$DT_{(orig,dest)}$ = Average Delay Time per vehicle (seconds)

$$DT_{(orig,dest)} = \frac{\sum_{i=1}^{N_{(orig,dest)}} DT_i}{N_{(orig,dest)}}$$

$ST_{(orig,dest)}$ = Average Stop Time per vehicle (seconds)

$$ST_{(orig,dest)} = \frac{\sum_{i=1}^{N_{(orig,dest)}} ST_i}{N_{(orig,dest)}}$$

$NS_{(orig,dest)}$ = Average Number of Stops per vehicle

$$NS_{(orig,dest)} = \frac{\sum_{i=1}^{N_{(orig,dest)}} NS_i}{N_{(orig,dest)}}$$

TotalTrav_(orig,dest) = Total distance travelled (Km)

$$TotalTrav_{(orig,dest)} = \sum_{i=1}^{N_{(orig,dest)}} D_i / 1000$$

TotalTravTime_(orig,dest) = Total travel time experimented(seconds)

$$TotalTravTime_{(orig,dest)} = \sum_{i=1}^{N_{(orig,dest)}} TT_i$$

FuelCon_(orig,dest) = Total fuel consumed (litres)

$$FuelCon_{(orig,dest)} = \sum_{i=1}^{N_{(orig,dest)}} TFC_i$$

PollEm_{(orig,dest),j} = Total pollution emitted of pollutant j-th (Kg)

$$PollEm_{(orig,dest),j} = \sum_{i=1}^{N_{(orig,dest)}} TPE_{i,j}$$

Origin Centroid Statistics

The previous Vehicle Variables are used for calculating the origin centroid statistical output. The following traffic statistics are calculated for every statistical interval and for the whole simulation period:

I = Interval of statistics (seconds).

N_{orig} = Number of vehicles that have reached its destination from its origin *orig*.

TT_{orig} = Average Travel Time per vehicle (seconds)

$$TT_{orig} = \frac{\sum_{i=1}^{N_{orig}} TT_i}{N_{orig}}$$

S_{orig} = Average Speed per vehicle (Km/h)

$$S_{orig} = \frac{\sum_{i=1}^{N_{orig}} S_i}{N_{orig}} * 3.6$$

HS_{orig} = Harmonic mean Speed per vehicle (Km/h)

$$HS_{orig} = \frac{N_{orig}}{\sum_{i=1}^{N_{orig}} HS_i} * 3.6$$

DT_{orig} = Average Delay Time per vehicle (seconds)

$$DT_{orig} = \frac{\sum_{i=1}^{N_{orig}} DT_i}{N_{orig}}$$

ST_{orig} = Average Stop Time per vehicle (seconds)

$$ST_{orig} = \frac{\sum_{i=1}^{N_{orig}} ST_i}{N_{orig}}$$

NS_{orig} = Average Number of Stops per vehicle

$$NS_{orig} = \frac{\sum_{i=1}^{N_{orig}} NS_i}{N_{orig}}$$

TotalTrav_{orig} = Total distance travelled (Km)

$$TotalTrav_{orig} = \sum_{i=1}^{N_{orig}} D_i / 1000$$

TotalTravTime_{orig} = Total travel time experimented (seconds)

$$TotalTravTime_{orig} = \sum_{i=1}^{N_{orig}} TT_i$$

FuelCon_{orig} = Total fuel consumed (litres)

$$FuelCon_{orig} = \sum_{i=1}^{N_{orig}} TFC_i$$

PollEm_{orig,j} = Total pollution emitted of pollutant j-th (Kg)

$$PollEm_{orig,j} = \sum_{i=1}^{N_{orig}} TPE_{i,j}$$

Destination Centroid Statistics

The previous Vehicle Variables are used for calculating the destination centroid statistical output. The following traffic statistics are calculated for every statistical interval and for the whole simulation period:

I = Interval of statistics (seconds).

N_{dest} = Number of vehicles that have reached their destination *dest* from their origin.

TT_{dest} = Average Travel Time per vehicle (seconds)

$$TT_{dest} = \frac{\sum_{i=1}^{N_{dest}} TT_i}{N_{dest}}$$

S_{dest} = Average Speed per vehicle (Km/h)

$$S_{dest} = \frac{\sum_{i=1}^{N_{dest}} S_i}{N_{dest}} * 3.6$$

HS_{dest} = Harmonic mean Speed per vehicle (Km/h)

$$HS_{dest} = \frac{N_{dest}}{\sum_{i=1}^{N_{dest}} \frac{1}{S_i}} * 3.6$$

DT_{dest} = Average Delay Time per vehicle (seconds)

$$DT_{dest} = \frac{\sum_{i=1}^{N_{dest}} DT_i}{N_{dest}}$$

ST_{dest} = Average Stop Time per vehicle (seconds)

$$ST_{dest} = \frac{\sum_{i=1}^{N_{dest}} ST_i}{N_{dest}}$$

NS_{dest} = Average Number of Stops per vehicle

$$NS_{dest} = \frac{\sum_{i=1}^{N_{dest}} NS_i}{N_{dest}}$$

$TotalTrav_{dest}$ = Total distance travelled (Km)

$$TotalTrav_{dest} = \sum_{i=1}^{N_{dest}} D_i / 1000$$

TotalTravTime_{dest} = Total travel time experimented (seconds)

$$TotalTravTime_{dest} = \sum_{i=1}^{N_{dest}} TT_i$$

FuelCon_{dest} = Total fuel consumed (litres)

$$FuelCon_{dest} = \sum_{i=1}^{N_{dest}} TFC_i$$

PollEm_{dest,j} = Total pollution emitted of pollutant j-th (Kg)

$$PollEm_{dest,j} = \sum_{i=1}^{N_{dest}} TPE_{i,j}$$

9.1.2.5 Public Transport Statistics

Vehicle Data Gathering

The information gathered from an individual public transport vehicle is as follows:

TEN_i = Entrance time of vehicle i-th in the system (seconds).

TEX_i = Exit time of vehicle i-th from the system (seconds).

D_i = Total Distance travelled by vehicle i-th in the system (meters).

TDT_i = Total Delay Time accumulated in each section by vehicle i-th (seconds).

TST_i = Total Stop Time accumulated in each section by vehicle i-th (seconds).

TNS_i = Total Number of Stops accumulated in each section by vehicle i-th .

TFC_i = Total fuel consumed accumulated in each section by vehicle i-th (litres).

TPE_{i,j} = Total emission of pollutant j-th accumulated in each section by vehicle i-th

Taking this information, when a public transport vehicle exits the system the following variables are calculated:

TT_i = Average travel time of vehicle i-th (seconds).

$$TT_i = TEX_i - TEN_i$$

DT_i = Average delay time of vehicle i-th (seconds).

$$TT_i = TDT_i$$

S_i = Average speed of vehicle i-th (m/s).

$$S_i = \frac{D_i}{TEX_i - TEN_i}$$

HS_i = Inverse of S_i (s/m), used for calculating the harmonic mean speed

$$HS_i = \frac{1}{\frac{D_i}{TEX_i - TEN_i}}$$

ST_i = Average stop time of vehicle i-th (seconds).

$$ST_i = TST_i$$

NS_i = Average number of stops of vehicle i -th .

$$NS_i = TNS_i$$

Public Transport Line Statistics

The previous Vehicle Variables are used for calculating the public transport line statistical output. The following traffic statistics are calculated for every statistical interval and for the whole simulation period:

I = Interval of statistics (seconds).

N_l = Number of vehicles that have completed the public transport line l -th.

TT_l = Average Travel Time per vehicle (seconds)

$$TT_l = \frac{\sum_{i=1}^{N_l} TT_i}{N_l}$$

S_l = Average Speed per vehicle (Km/h)

$$S_l = \frac{\sum_{i=1}^{N_l} S_i}{N_l} * 3.6$$

HS_l = Harmonic mean Speed per vehicle (Km/h)

$$HS_l = \frac{N_l}{\sum_{i=1}^{N_l} \frac{1}{S_i}} * 3.6$$

DT_l = Average Delay Time per vehicle (seconds)

$$DT_l = \frac{\sum_{i=1}^{N_l} DT_i}{N_l}$$

ST_l = Average Stop Time per vehicle (seconds)

$$ST_l = \frac{\sum_{i=1}^{N_l} ST_i}{N_l}$$

NS_l = Average Number of Stops per vehicle

$$NS_l = \frac{\sum_{i=1}^{N_l} NS_i}{N_l}$$

TotalTrav_l = Total distance travelled (Km)

$$TotalTrav_l = \sum_{i=1}^{N_l} D_i / 1000$$

TotalTravTime_l = Total travel time experimented (seconds)

$$TotalTravTime_l = \sum_{i=1}^{N_l} TT_i$$

FuelCon_l = Total fuel consumed (litres)

$$FuelCon_l = \sum_{i=1}^{N_l} TFC_i$$

PollEm_{l,j} = Total pollution emitted of pollutant j-th (Kg)

$$PollEm_{l,j} = \sum_{i=1}^{N_l} TPE_{i,j}$$

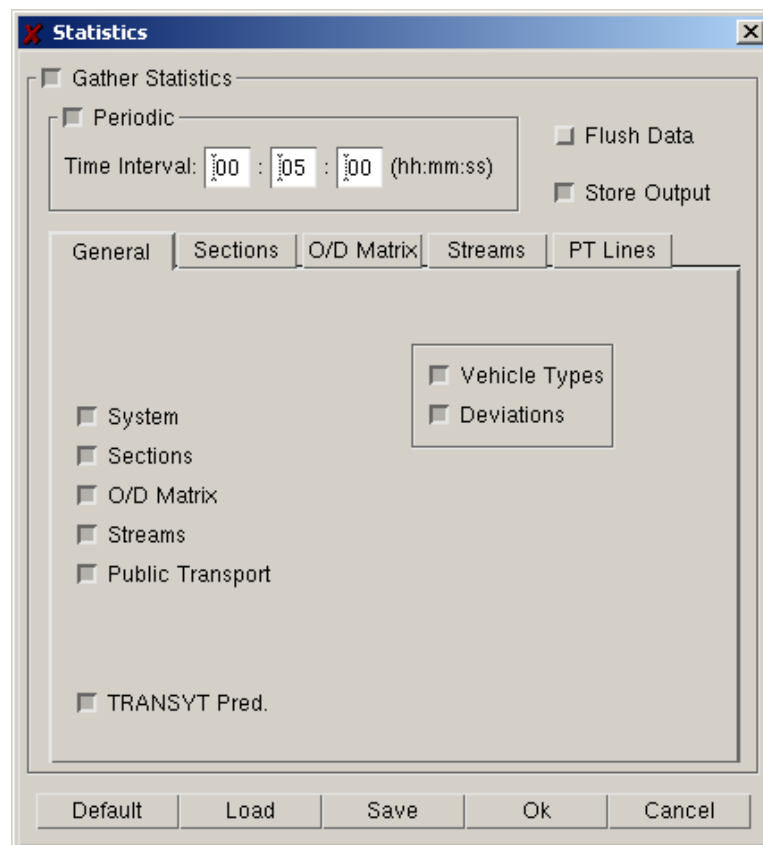
9.1.3 Customisation of Statistical Output

The ‘Experiment / Output / Statistics’ command can be used to define the type of statistics the user wishes to produce as simulation output and whether or not he/she wants them to be stored in files. This must be carried out before running a simulation experiment, since the report parameters cannot be modified once a simulation experiment has been started.

The Statistics Parameter window (see Figure 9-2), contains a ‘Gather Statistics’ toggle button that can be used to indicate whether or not the user wants statistics to be gathered during the simulation run. If this button is selected, the next two toggle buttons are activated and may be selected: ‘Periodic’ and ‘Report Print Out’.

The user can select the type of time scope for the statistical data gathering. It may be Global or Periodic, as described in section 9.1.1. If the ‘Periodic’ toggle button is pressed, periodical statistics will be gathered. If not, only global statistics will be collected. When selecting the Periodic option, the time interval has to be defined (hours: minutes: seconds). This definition has to be made before running the simulation. Otherwise, statistical data may not be gathered and no statistical result would be produced. In Periodic mode, Global statistics are also provided at the end of the simulation.

Also, when selecting the Periodic option, the ‘Flush’ toggle button becomes active. If the Flush button is selected, the periodic statistical data will be cleared from the memory at every time interval. It is assumed that the user has selected to store the necessary periodic statistics in some other way, such as into files or a database, or by activating the Report PrintOut button. The purpose of the Flush option is to save memory during the simulation run. However, when the flush option has been activated, it is not possible to view the current periodic statistics via the ‘Reports/Current Report/Statistics’ command, nor to display time plots via the ‘Reports/Current Report/Time Plot’ command.

Figure 9-2: Statistics Definition Dialog Window: General

With the 'Report PrintOut' toggle button, the user can specify whether or not an Output File or database containing the statistical results is to be produced automatically during the simulation run. The statistic measures are presented in several levels of aggregation:

- System: statistics relating to the whole network.
- Sections: statistics for each individual section and optionally for each turning movement.
- O/D Matrix: statistics for O/D centroid pairs
- Streams: statistics for each stream (set of consecutive sections).
- Public Transport: statistics for each Public Transport Line.

The user can select the desired level of aggregation by clicking on the appropriate toggle buttons, 'System', 'Sections', 'O/D Matrix', 'Streams' and 'Public Transport'. There is also the option to produce the standard TRANSYT/10 Performance Indexes.

The user can also specify whether or not the statistics gathered are to be broken down by vehicle type. This is done by clicking on the 'Veh. Types' toggle button. Finally, the user can request deviation data for all measurements by clicking on the 'Deviations' toggle button. If 'Deviations' is not selected, only the mean values for all measurements are provided.

Note: The difference between version 4.2 and previous versions is that version 4.2 the user defines the level of aggregation gathered during the simulation (system, sections, od matrices, streams and public transport) and selects whether want to store in output files or databases with the 'Store Ouput' toggle button, while in previous versions all statistical measures were gathered and the user selected the level of aggregation to create the output files or database. This means that in version 4.2 the user can view using the graphical interface only the statistical results (during or after the simulation run) of the levels of aggregation selected, while in previous versions was possible to view all statistical results and the 'Report PrintOut' button was only used for the automatic creation of output files or databases.

Sections

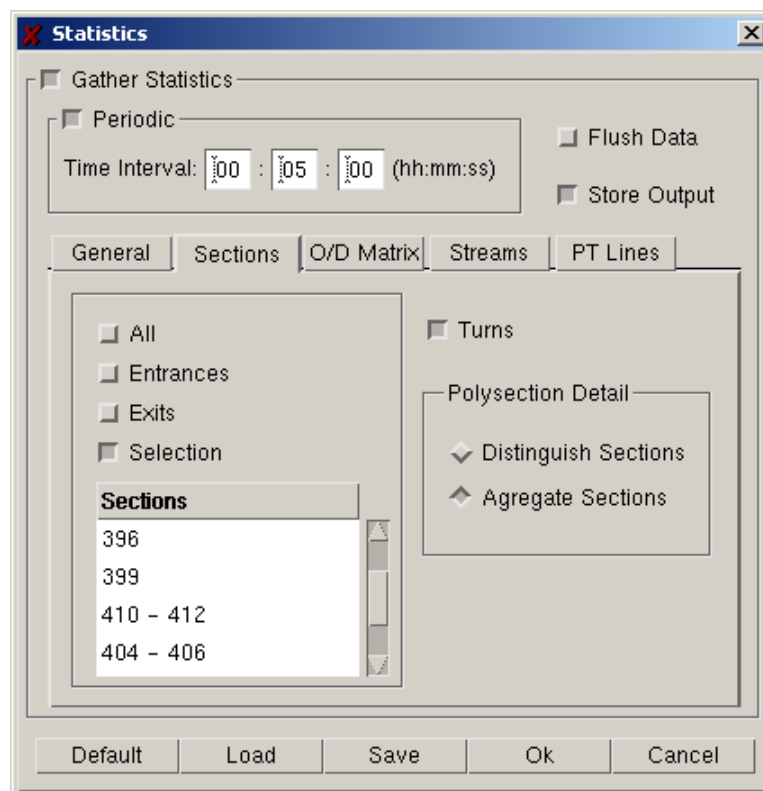
When selecting the 'Sections' level, the user must specify the desired sections in the 'Sections' tab folder (see Figure 9-3) using any of the following toggle buttons:

- All: statistics for all the sections of the network will be printed.
- Entrances: statistics for the input sections only will be printed.
- Exits: statistics for the output sections only will be printed.
- Selection: the user can select a list of sections by clicking on them directly in the graphical representation of the network. Only statistics for these sections will be printed.

Section information can be detailed for each turning movement. Clicking on the 'Turns' toggle button does this.

The polysection information can be detailed for each section, clicking on 'Distinguish Sections' toggle button, or can be aggregated per polysection clicking on 'Aggregate Sections' toggle button. If the level of aggregation is at level of polysection, the statistical measures of polysections in the output files or database are identified by the last section identifier downstream.

Figure 9-3: Statistics Definition Dialog Window: Sections



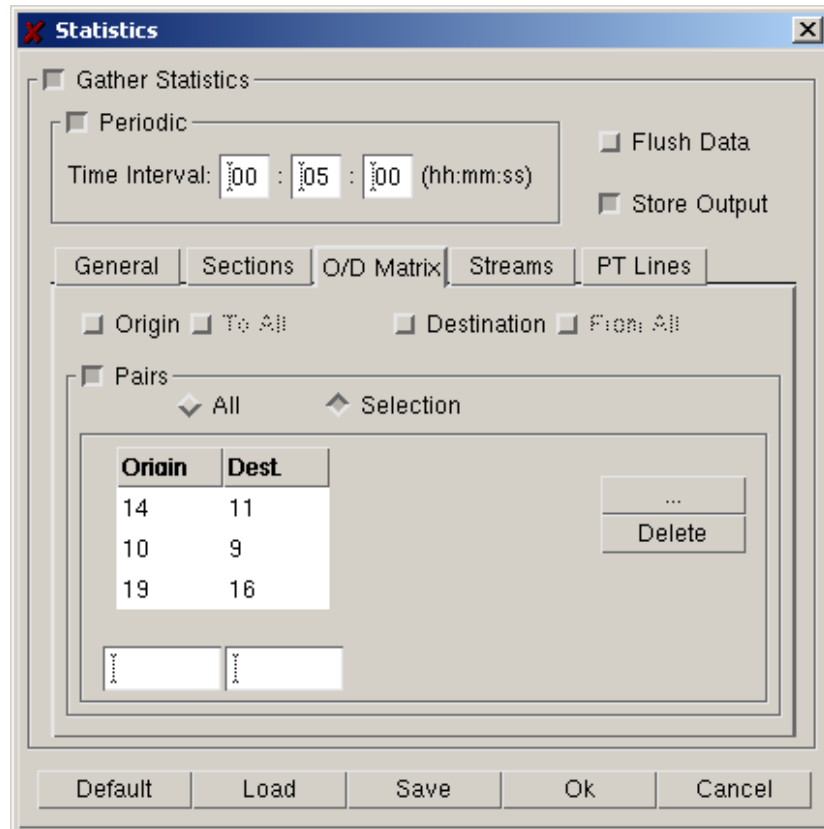
O/D Matrix

When selecting the 'OD Matrix' level, the user can specify the Origin and/or Destination centroids for which to generate statistical data via the 'O/D Matrix' tab folder (see Figure 9-4). The following four toggle buttons are provided:

- Origin: O/D statistics are listed by Origin centroid.
- To All: this is used together with the Origin toggle button. For each Origin centroid, data is broken down into all possible destination centroids.
- Destination: O/D statistics are listed by Destination centroid.
- From All: this is used together with the Destination toggle button. For each Destination centroid, data is provided for all possible origin centroids.

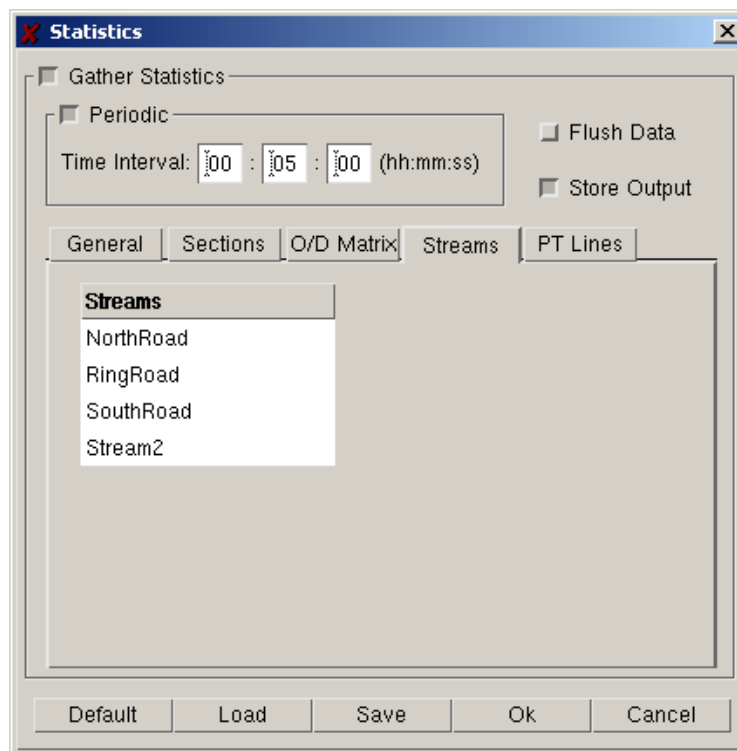
It is also possible to select statistics gathering for a set or all possible O/D pairs. Clicking on the pairs toggle button and on the 'All' toggle button will provide statistical measurements for every O/D pair. However, the user has to be careful when selecting this option because of the huge amount of data this may produce. To avoid this, the user can select a set of O/D pairs by clicking on the Selection toggle button and then choosing a set of pairs.

Figure 9-4: Statistics Definition Dialog Window: O/D Matrix.

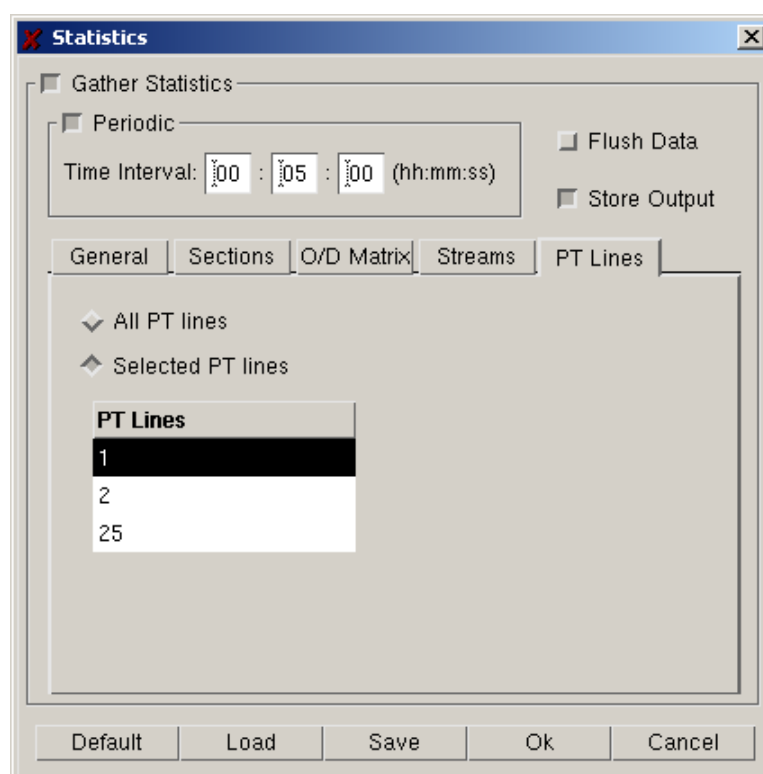


Streams

When selecting the Streams level, the user can specify a set of streams via the 'Streams' tab folder (see Figure 9-5), by first clicking on the list box in which the names of the available streams are presented. Doing this highlights the name. Only statistics for the selected streams will be printed.

Figure 9-5: Statistics Definition Dialog Window: Streams**Public Transport**

The 'PT Lines' tab folder (see Figure 9-6) allows the user to specify whether to print statistics for all the Public Transport Lines, or to print statistics for a selected subset only, just by clicking on the appropriate toggle button. When 'Selected PT Lines' is chosen, select the lines by clicking on the list box containing the names of the available PT Lines. Doing this highlights the name. Only statistics for the selected Public Transport Lines will be printed.

Figure 9-6: Statistics Definition Dialog Window: Public Transport

TRANSYT/10 Performance Indexes

When the 'TRANSYT Pred.' Toggle button is selected, the TRANSYT/10 Performance Indexes are provided as simulation output. See the GETRAM-TRANSYT/10 Interface User Manual for details.

9.1.4 Statistics File Reports

At any time, the user may load statistics reports stored in files that have been saved from previous or current simulation runs. These files may be produced automatically during simulation if the user has activated the 'Report PrintOut' toggle button (carried out in the Statistics Parameter Window, via the 'Experiment / Output / Statistics' command), prior to start a simulation run (see section 9.1.3). Selecting the 'Reports / File Reports' option displays the Choose Files Window (see Figure 9-7). Using this dialog window, the user can browse throughout the directories to find a particular Statistics Files.

The name that AIMSUN assigns to the report files produced automatically during simulation have the format 'time'.EST, where 'time' corresponds to the simulation time when the report was produced with the format HHhMMmSS.EST. Select a file, press the 'Load' button and the file will be loaded and displayed as shown in Figure 9-8. Only Statistics and Detection data files can be loaded via the AIMSUN File Report Window. The contents of the file will be those previously defined in the Statistics Window.

Physically, the Statistics files are located in the directory selected by the user via the 'Experiment / Output / Output Location' command (see Figure 9-9). First, the user must specify that the output is to be produced as ASCII files by selecting the 'To ASCII Files' toggle button. Then, in the 'Directory' dialog window, type in the folder name to which the output files are to be saved or find the folder using the browse button. The ASCII files can be edited directly using any text editor, not necessarily via the AIMSUN user interface.

Figure 9-7: Choose File window

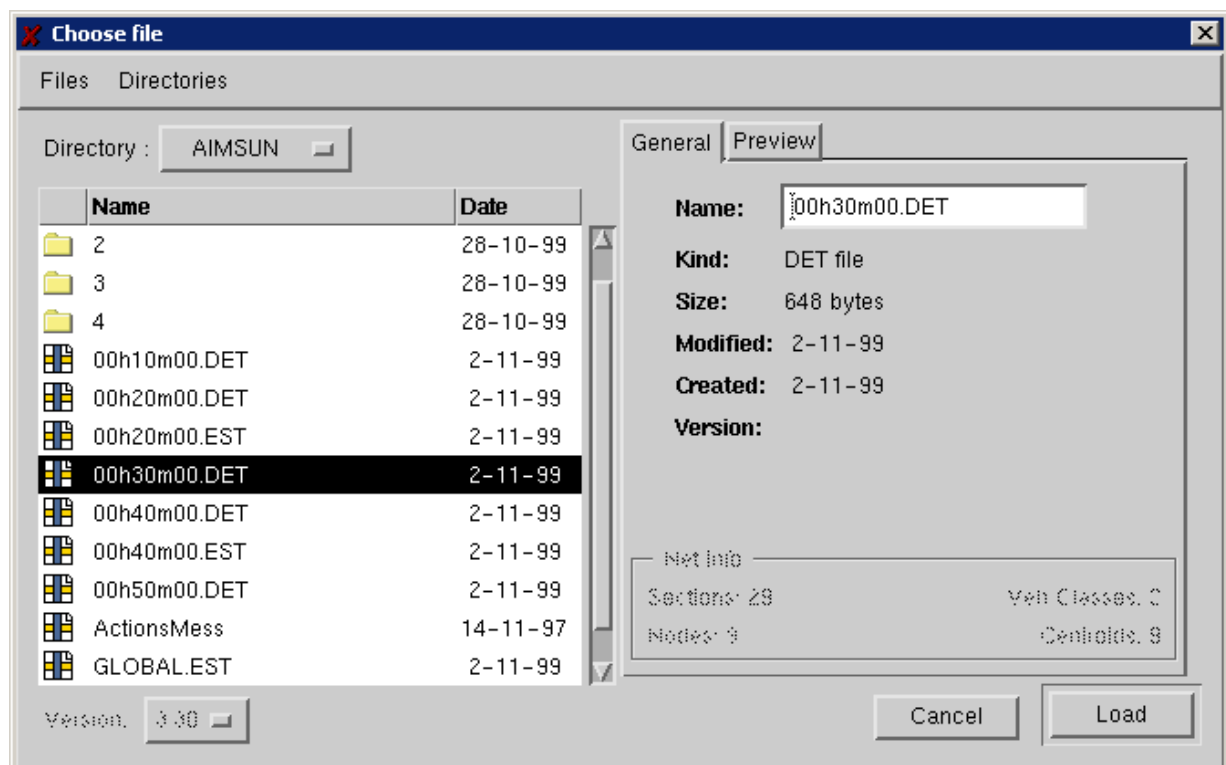


Figure 9-8: File Report window

File Report "F:\Networks\Dublin_40\Peak4_5pm\AIMSUN\GLOBAL.ES1"

System Global Statistics										
Time: 18:00:00					Date: 29/12/94					
FLOW	DENSTY	SPEED	H. SPEED	TRAVEL T.	DELAY T.	STOP T.	STOPS	TRAVEL	TOT. TRAVEL T.	
Veh/h	veh/Km	Km/h	Km/h	h:mm:ss	h:mm:ss	h:mm:ss	#/Veh	Km	h:mm:ss	
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	
+ System	5335	15.5	26.6	16.9	0:03:32	0:02:27	0:02:07	3.2	5713.0	339:57:59

All Sections											
Global Statistics of Sections and Turnings											
Time: 18:00:00						Date: 29/12/94					
FLOW	DENSTY	SPEED	H. SPEED	TRAVEL T.	DELAY T.	STOP T.	STOPS	QUEUE LENGTH	TRAVEL	TOT. TRAVEL T.	
Veh/h	veh/Km	Km/h	Km/h	h:mm:ss	h:mm:ss	h:mm:ss	#/Veh	Mean / Max	Km	h:mm:ss	
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	
* Dorset (N)	897	17.4	29.3	14.9	0:00:29	0:00:21	0:00:18	0.6	1.4 / 10.0	216.4	14:32:58
- TO Gardiner	47		21.5	12.4	0:00:35	0:00:26	0:00:22	1.1	0.8 / 5.5	11.7	00:56:21
- TO Dorset (N)	828		29.8	15.1	0:00:28	0:00:21	0:00:17	0.6	0.9 / 6.7	199.8	13:16:40
- TO Synnot Pl.	22		26.6	14.6	0:00:27	0:00:20	0:00:16	0.6	0.7 / 4.5	4.9	00:19:55
* Dorset (N)	1276	18.5	33.1	17.5	0:00:19	0:00:13	0:00:10	0.4	1.1 / 10.0	243.2	13:54:29
- TO N. C. R. (E)	68		25.4	16.7	0:00:21	0:00:13	0:00:10	0.6	0.4 / 5.0	13.3	00:47:51
- TO Dorset (N)	1154		33.6	17.6	0:00:19	0:00:13	0:00:09	0.4	0.8 / 7.0	220.9	12:33:57
- TO N. C. R. (W)	54		33.1	16.6	0:00:18	0:00:12	0:00:10	0.4	0.3 / 4.0	9.0	00:32:39

Figure 9-9: Output Location Dialog Window

Output location

Stats. & Detection | Recorder

☐ To ASCII Files

Directory: F:\Networks\PTNetTer42\AIMSUN

☐ To ODBC Database

PTNET41

User: Password:

☐ To Access Database

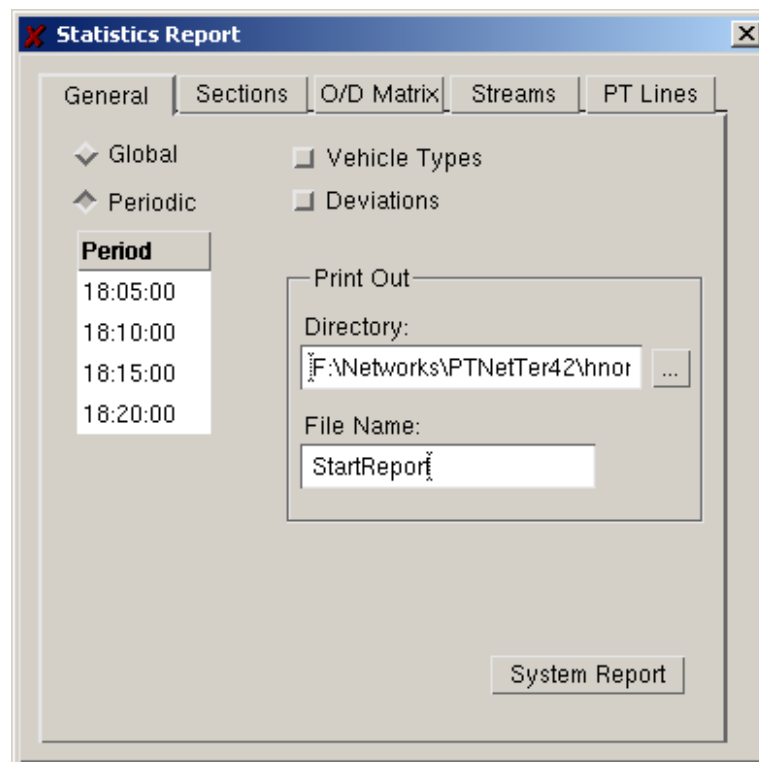
Database:

Default Load Save Ok Cancel

9.1.5 Statistics Current Reports

To view statistics gathered during the current simulation experiment, use the 'Reports / Current Report / Statistics' command. This can only be done while the simulation is not running. The 'Statistics Report' window will appear as displayed in Figure 9-10.

The user may select the time scope, Global or Periodic, by clicking on the corresponding toggle button. When the 'Periodic' level is selected, the user must also choose a period by clicking on the periods list box. The toggle buttons allow the user to specify whether or not to have the statistics reports detailed by vehicle type and whether to include deviations for all measurements, or only the mean values.

Figure 9-10: Report Information window

Each level of aggregation corresponds to a tab folder: sections, O/D matrices, streams and Public Transport. Pressing the System button, opens the System Global Report Window (see Figure 9-11). A printout of the contents of this window may be obtained by pressing the 'Print Out' button. A Print Out File name must first be defined in the General folder of the 'Statistics Report' window, however.

Figure 9-11: System Global Report window

System Global Report from 18:00:00 to 20:00:00 Date:11/16/00										
	FLOW veh/h	DENSITY veh/Km	SPEED Km/h	H.SPEED Km/h	TRAVEL T. hh:mm:ss/Km	DELAY T. hh:mm:ss/Km	STOP T. hh:mm:ss/Km	STOPS #/veh/Km	TRAVEL Km	T.T.TRAVEL hh:mm:ss
+ System	1166	2.1	51.5	49.9	00:01:12	00:00:07	00:00:02	0.2	2287.5	46:50:14

When selecting Section level by choosing the 'Sections' tab folder the user specifies the set of sections to take into account as explained in section 9.1.3.

For example, selecting 'Periodic', period 00:15:00, 'Sections', 'All', 'Turns' and pressing the 'OK' button will display the window in Figure 9-12. As for the System level, a printout of the contents of this window may be obtained by pressing the 'Print Out' button. A Printout File name must first be defined in the General folder of the 'Statistics Report' window, however. Several Statistics Report windows may be open at the same time, so that the user can compare results from different periods, for example.

Figure 9-12: Sections and Turnings Periodic Report window

Sections & Turns Report at 18:20:00 Date:11/16/00													
	Id	FLOW veh/h	DENSITY veh/Km	SPEED Km/h	H.SPEED Km/h	TRAVEL T. hh:mm:ss	DELAY T. hh:mm:ss	STOP T. hh:mm:ss	STOPS #/veh	QUEUE Mean / Max	LENGTH	TRAVEL Km	T.T.TRAVEL hh:mm:ss
* SECTION	176	348	3.0	39.3	38.6	00:00:03	00:00:00	00:00:00	0.0	0.0 / 0.0		1.2	00:01:47
- TO	181	264		39.3	38.3	00:00:03	00:00:00	00:00:00	0.0	0.0 / 0.0		0.8	00:01:18
- TO	243	84		39.5	39.5	00:00:04	00:00:00	00:00:00	0.0	0.0 / 0.0		0.3	00:00:29
* SECTION	179	228	2.5	55.4	45.9	00:00:15	00:00:04	00:00:02	0.2	0.0 / 0.0		3.7	00:04:48
- TO	243	228		55.4	45.9	00:00:15	00:00:04	00:00:02	0.2	0.0 / 0.0		3.7	00:04:48
* SECTION	181	252	1.9	66.9	66.7	00:00:09	00:00:01	00:00:00	0.0	0.0 / 0.0		3.6	00:03:12
- TO	182	252		66.9	66.7	00:00:09	00:00:01	00:00:00	0.0	0.0 / 0.0		3.6	00:03:12
* SECTION	182	252	1.8	72.6	53.7	00:00:10	00:00:03	00:00:02	0.0	0.0 / 1.0		3.2	00:03:31
- TO	399	252		72.6	53.7	00:00:10	00:00:03	00:00:02	0.0	0.0 / 1.0		3.2	00:03:31

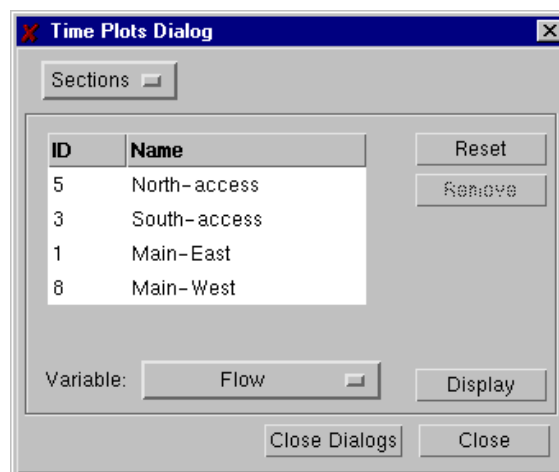
O/D Matrices, Streams and PT Lines tab folders work in the same way as described in section 9.1.3. The dialog windows are almost identical, apart from the option for defining the name of the Print Out file.

9.1.6 Statistics Current Graphics

Statistical data gathered during the current simulation experiment can be presented in a graphical form by means of time series plots or directly by representing data in the network using a range of colours. These functions are done via options included in the 'Reports / Current Graphics' command: Time Plots and Whole Network.

Time Plots

To access a Time Plot, use the 'Reports / Current Graphics / Time Plot' command in the menu bar. The Time Plots dialog window appears as displayed in Figure 9-13.

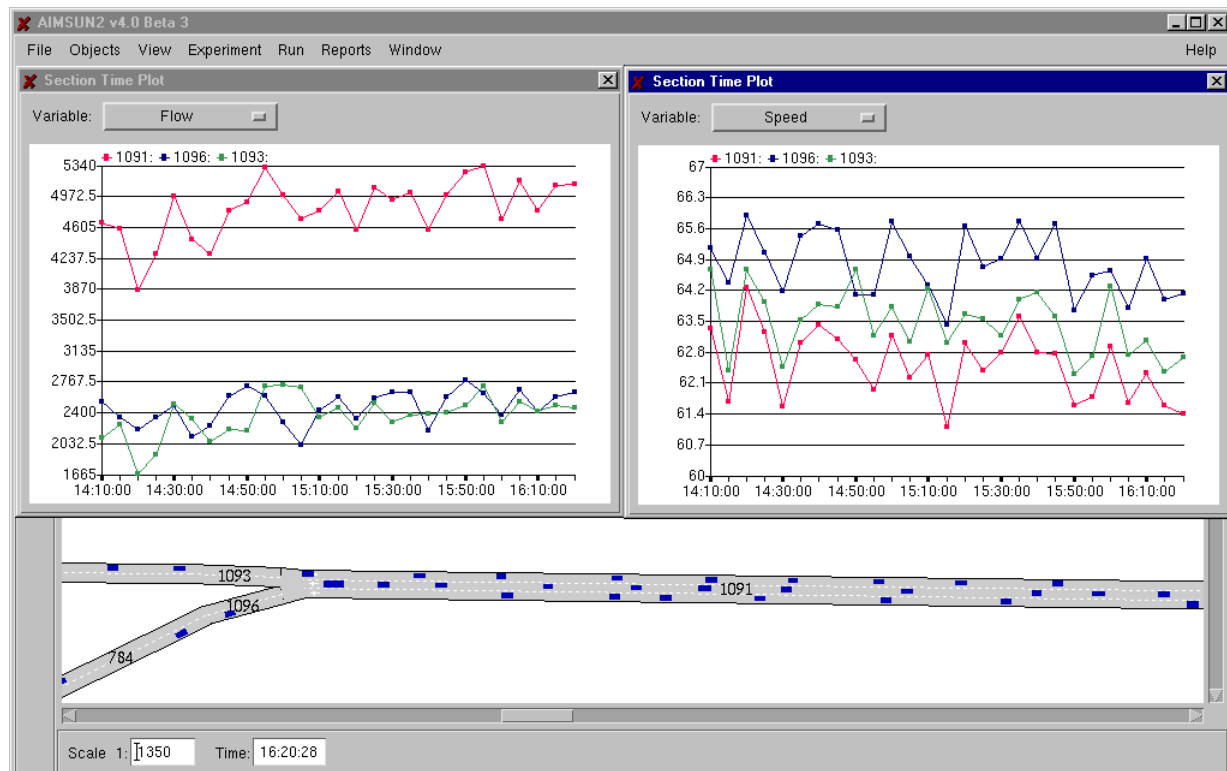
Figure 9-13: Statistics Time Plot Dialog Window

The user may select a set of sections whose statistical data is to be displayed by clicking directly on the section in the network display (the available sections, whose statistical data has been gathered, are coloured, while the non available sections are displayed using the contour). The identifiers for the selected sections will be included in the 'Section id's' list box. To remove a section from the list, click again on the section or use the Remove button.

The variable to be plotted can be selected using the 'Variable' option menu. The user can chose from Flow, Speed, Density, Travel Time, etc. Then click on the Display button and the time plot is drawn (see Figure 9-

14). Each Section Time Plot dialog window has its own 'Variable' option menu; thus the selected variable can be changed at any time.

Figure 9-14: Current Graphics: Section Time Series Plot. Flow and Speed



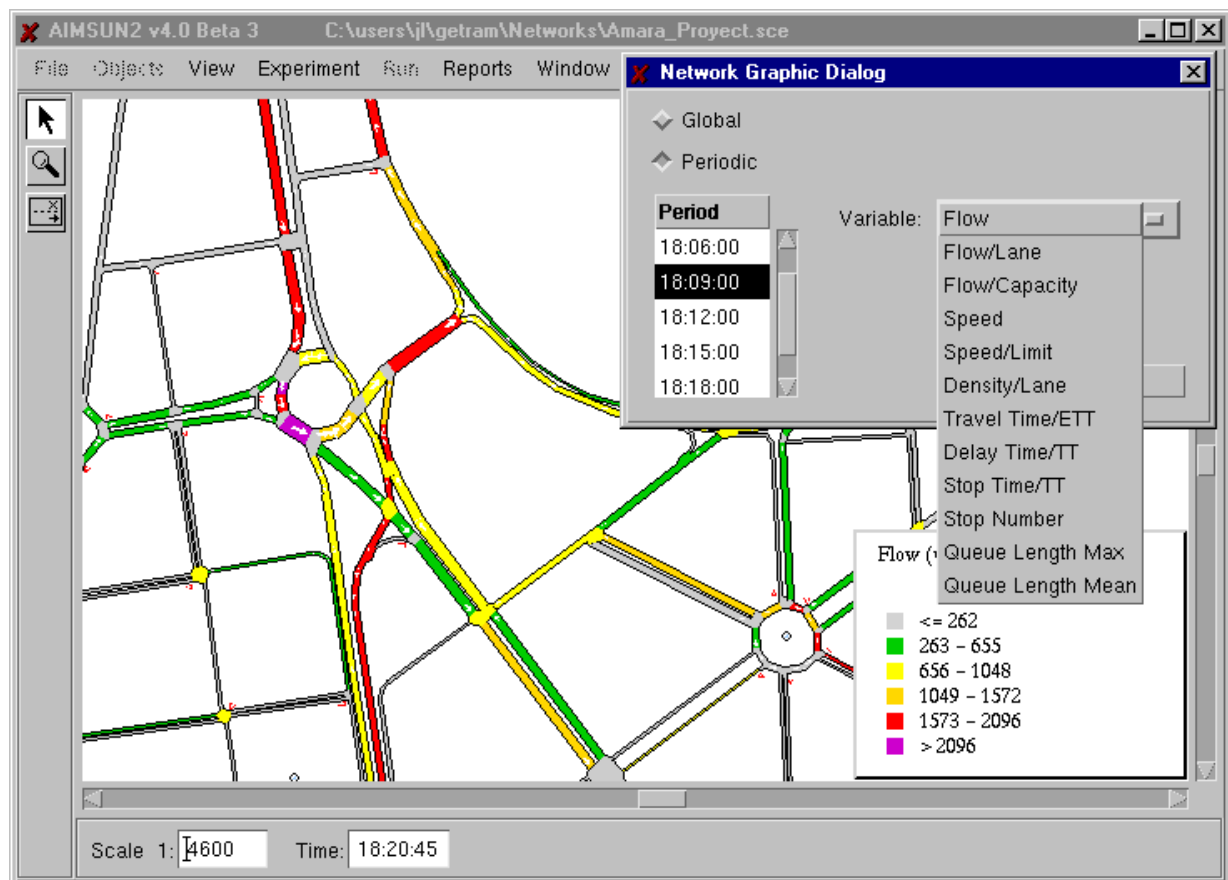
Each Section Time Plot dialog window can be closed individually. The 'Close Dialogs' button in the Time Plots dialog window causes all Section Time Plot windows to be closed simultaneously, while the 'Close' button keep all Time Plots open and only closes the Time Plots dialog window. The Section Time Plot window can be opened while the simulation is running, and the time plot will be updated continuously as new periodic statistical data becomes available.

Whole Network

The 'Reports / Current Graphics / Whole Network' option allows you to colour the sections of the network according to a range of colours that represent different values for a set of traffic measurements (see Figure 9-15).

The measures that can be displayed on the network, either at the global level or for certain time periods, are the following:

- Flow: mean number of vehicles per hour
- Flow / Lane: vehicles per lane per hour
- Flow / Capacity: percentage relationship between mean flow and section capacity (taking as capacity the one defined as input via Tedi)
- Speed: mean speed in km/h
- Speed / Speed Limit: percentage relationship between mean speed and section speed limit
- Density / Lane: number of vehicles per kilometre per lane
- Travel Time: percentage relationship between mean travel time and mean expected travel time
- Delay Time: percentage relationship between mean delay time and mean travel time
- Stop Time: percentage relationship between mean stop time and mean travel time
- Stop Number: mean number of stops per vehicle
- Maximum Queue Length: maximum length of the queue in number of vehicles per lane
- Mean Queue Length: average length of the queue in number of vehicles per lane.

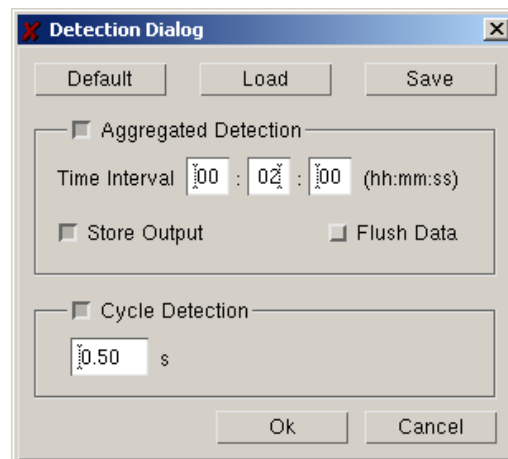
Figure 9-15: Displaying Statistical Data on the Network

9.2 OUTPUT DETECTION

If the network contains traffic detectors, the user can choose what detection model to apply by using the 'Experiment / Output / Detection' command. Two types of detection could be modelled: Common detection and Cycle detection.

To model Common Detection, click on the 'Aggregated Detection' toggle button in the 'Detection' window as displayed in Figure 9-16. A detection time interval can then be defined (hours: minutes: seconds). This parameter represents the frequency of Detection Output Files production.

Figure 9-16: Detection Model window



Finally, the detection output data can be printed to ASCII files or to a database, or not printed at all. This is also defined via the 'Detection' dialog window by clicking on 'Store Output' toggle button. If 'Store Output' toggle button is not selected, aggregated detection data will only be accessible via the GETRAM Extension functions. If the 'Store Output' toggle button is selected, the 'Flush Data' option is no longer greyed out. Activating 'Flush Data' means that no detection data is stored in memory, but only printed out. In this case, it would not be possible to view Detection Time Plots via the 'Reports / Current Graphics / Time Plot' command.

To model Cycle Detection, a detection for the ETCMS type Detection Model, click on the 'Cycle Detection' toggle button and define the time interval in seconds. By default, when the 'Cycle Detection' toggle button is selected, the time interval proposed is the same value as the simulation step, but it can be modified, that means the detection for the ETCMS type Detection Model is independent of the simulation step. This type of detection data will only be accessible through the GETRAM Extension functions.

9.2.1 Detection File Reports

Detection output files are produced periodically, provided that the user has

1. activated the Aggregated Detection toggle button in the Detection Parameter Window,
2. set the Print Out option to 'Yes' before running the simulation experiment, and
3. defined an output detection interval.

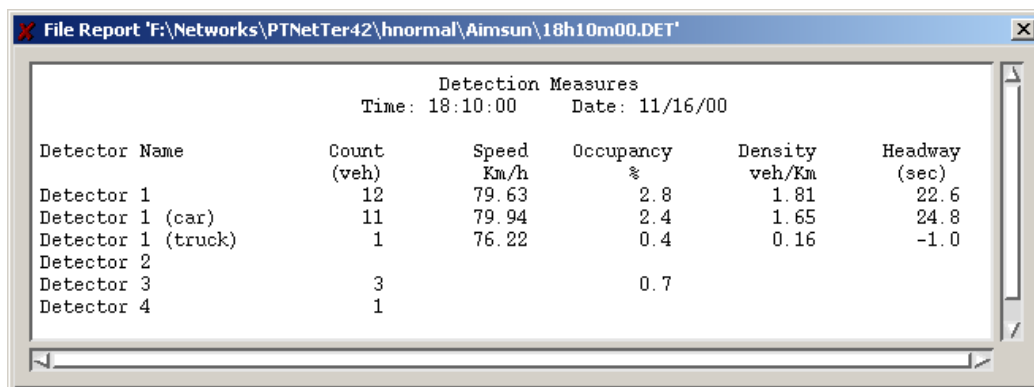
Physically, the Detection files are located in the directory selected by the user via the 'Experiment / Output / Output Location' command (see Figure 9-9). This is carried out simultaneously for Detection and Statistic Outputs. They can be viewed via the AIMSUN interface using the 'Output / Files Reports' command in the same way as for Statistics Output Files (see section 9.1.4). An example of a Detection file loaded via the AIMSUN interface is displayed in Figure 9-17. Since they are ASCII files, they can also be directly edited using any text editor, not necessarily via the AIMSUN user interface.

The content of detection output files depends on the measuring capabilities of the detectors. A data line exists for each detector, containing the detector identifier and the list of measurements gathered. The measurements that can be provided by detectors are the following:

- Count: number of vehicles that have passed through the detector during the interval (vehicles)
- Occupancy: percentage of time that the detector has been triggered during the interval (%)
- Speed: mean speed of the vehicles when crossing the detector (kilometres/hour)
- Density: calculated using the measured count and speed (vehicles/kilometres).
- Headway: mean headway of the vehicles when crossing the detector during the interval (seconds).

When the a detector has the 'Equipped' capability activated, AIMSUN produces automatically during simulation a list of all equipped vehicles that have passed through the detector during the interval. The name that AIMSUN assigns to these report files have the format Equipped'time'.DET, where 'time' corresponds to the simulation time when the report was produced with the format EquippedHHhMMmSS.DET. A data line exists for each equipped vehicle, containing the detector identifier, the vehicle identifier, the vehicle type name and the public transport line identifier in the case of a public transport vehicle.

Figure 9-17: Detection Output File Report



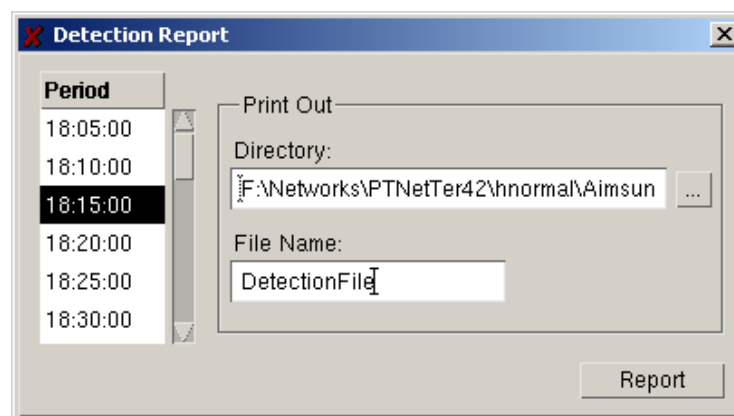
The screenshot shows a window titled 'File Report: F:\Networks\PTNetTer42\hnormal\Aimsun\18h10m00.DET'. It displays a table of detection measures for several detectors. The table has columns for Detector Name, Count (veh), Speed (Km/h), Occupancy (%), Density (veh/Km), and Headway (sec). The data is as follows:

Detector Name	Count (veh)	Speed (Km/h)	Occupancy (%)	Density (veh/Km)	Headway (sec)
Detector 1	12	79.63	2.8	1.81	22.6
Detector 1 (car)	11	79.94	2.4	1.65	24.8
Detector 1 (truck)	1	76.22	0.4	0.16	-1.0
Detector 2					
Detector 3	3		0.7		
Detector 4	1				

9.2.2 Detection Current Reports

To view detection data gathered during the current simulation experiment, use the 'Reports / Current Report / Detection' command. This can only be done when the simulation is not running. The 'Detection Report' dialog window is displayed as shown in Figure 9-18.

Figure 9-18: Detection Report Dialog Window



The screenshot shows the 'Detection Report' dialog window. It has a 'Period' list box on the left with the following times: 18:05:00, 18:10:00, 18:15:00 (selected), 18:20:00, 18:25:00, and 18:30:00. On the right, there is a 'Print Out' section with a 'Directory' text box containing 'F:\Networks\PTNetTer42\hnormal\Aimsun' and a 'File Name' text box containing 'DetectionFile'. A 'Report' button is at the bottom right.

To view the detection data for a given period, select a time period from the list box and click on the 'Report' button. The Detection Report window is then displayed as shown in Figure 9-19. A printout of the contents of this window can be obtained by pressing the 'Print Out' button. A Print Out File name and directory must first be defined in the 'Detection Report' window, however.

Figure 9-19: Detection Report window

Id	Count veh.	Speed Km/h	Occupancy %	Density veh/Km	Headway s
1	12	82.1	2.8	1.7	22.8
car	10	83.2	2.1	1.4	27.8
truck	2	78.6	0.8	0.3	195.6
2					
3	2		0.3		
4	0				

Print Out

9.2.3 Detection Current Graphics

Detection data gathered during the current simulation experiment can be presented in a graphical form by means of time series plots. This is done via the 'Reports / Current Graphics / Time Plot' command. The 'Time Plot' dialog window appears, as shown previously in Figure 9-13.

The window contains an option menu with two options: 'Sections' and 'Detectors'. By default, the 'Sections' option is active, so you must change it to the 'Detectors' option. It is then possible to select a set of detectors for which data has to be displayed, by clicking directly on them in the network display. The identifiers for the selected detectors will be included in the 'Detectors id's' list box (see Figure 9-20). To remove a detector from the list, click again on the detector or use the Remove button.

Figure 9-20: Detection Time Plot Dialog

Time Plots Dialog

Detectors ▾

ID	Name
	Detector-3
	Detector-1

Reset

Remove

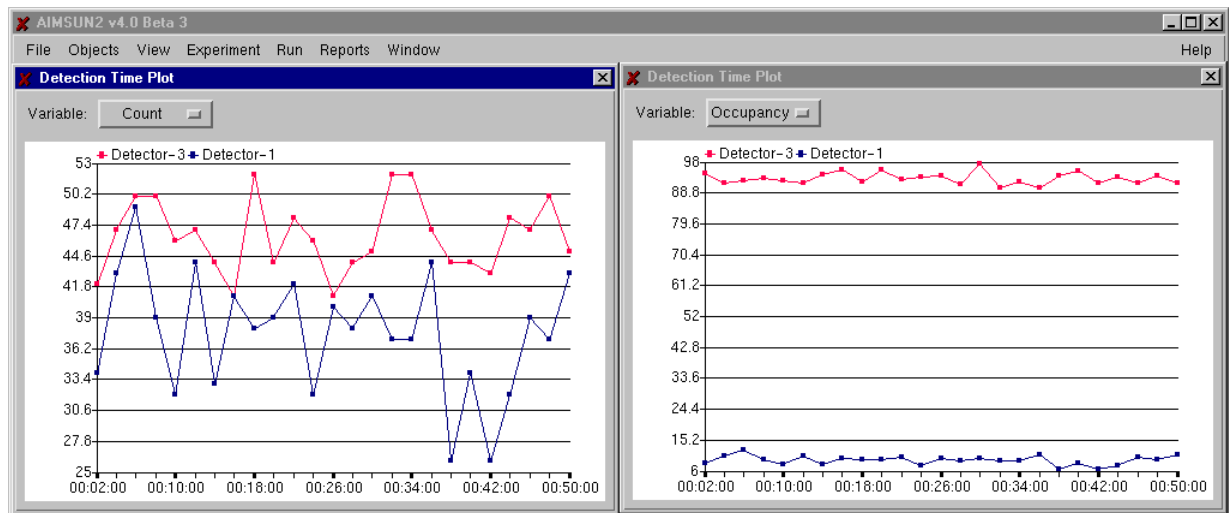
Variable: Count ▾

Display

Close Dialogs Close

The variable to be plotted may be selected using the 'Variable' option menu. The user may choose between Count, Speed, Density and Occupancy. Then click on the Display button and the time plot will be drawn (see Figure 9-21). Each Detection Time Plot dialog window has its own 'Variable' option menu. The selected variable can be changed at any time.

Each Detection Time Plot dialog can be closed individually. The 'Close Dialogs' button in the Time Plots dialog window causes all Detection Time Plot windows to be closed simultaneously. The 'Close' button keeps all Time Plots open and only closes the Time Plots dialog window. The Detection Time Plot window can be open while the simulation is running, and the time plot will be updated continuously as new detection data becomes available.

Figure 9-21: Current Graphics: Detection Time Series Plot. Count and Occupancy

9.2.4 Detection Data Gathering

This section describes in detail the procedures applied to produce the different traffic detection measures, as well as the individual vehicle data that is gathered.

Each Detector has two types of measures gathered:

- Aggregated detection: Detection is aggregated during a time period.
- Cycle Detection: Detection done each cycle of detection.

Considering these two types of detection, each detector stores the following variables:

- *Veh (cycle)*: Set of Vehicles that have entered during last Cycle.
- *NbVeh (cycle)*: Number of Vehicles that have entered during last Cycle.
- *Speed (v, cycle)*: Speed of Vehicle *v* that has entered during last Cycle.
- *EntranceTime (v, cycle)*: Time of Vehicle *v* that has entered during last Cycle.

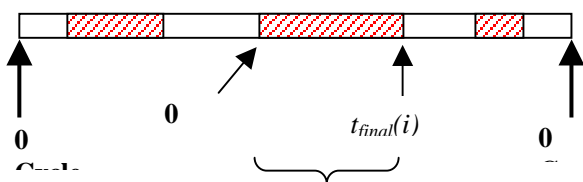
Cycle Detection Measures

- Count: number of vehicles that have passed through the detector during the last cycle (vehicles)

$$\text{Count}(\text{cycle}) = \text{NbVeh}(\text{cycle}).$$
- Speed: mean speed of the vehicles when crossing the detector during the last cycle (km/h or mph)

$$\text{Speed}(\text{cycle}) = \frac{\sum_{v \in \text{Veh}(\text{cycle})} \text{Speed}(v, \text{cycle})}{\text{NbVeh}(\text{cycle})}$$

- Presence: 1 if any vehicle is over the detector during the last cycle, 0 otherwise.
- Occupancy: percentage of cycle time that the detector has been pressed during last cycle (%).



Interval *i*th of the Cycle with vehicles pressing the detector. Each interval has an initial time $t_{init}(i, \text{cycle})$ and a final time $t_{final}(i, \text{cycle})$

$$Occupancy(Cycle) = \frac{\sum_{i=1}^{TotalNumberIntervals(cycle)} t_{final}(i, cycle) - t_{init}(i, cycle)}{Cycle} * 100$$

Interval Detection Measures

- Count: number of vehicles that have passed through the detector during the interval (vehicles)

$$Count(interval) = \sum_{cycle \in interval} Count(cycle)$$

- Speed: mean speed of the vehicles when crossing the detector during the interval (km/h or mph)

$$Speed(interval) = \frac{\sum_{cycle \in interval} \sum_{v \in Veh(cycle)} Speed(v, cycle)}{\sum_{cycle \in interval} NbVeh(cycle)}$$

- Presence: 1 if any vehicle is over the detector during the last interval, 0 otherwise.
- Occupancy: percentage of cycle time that the detector has been pressed during the interval (%).

$$Occupancy(interval) = \frac{\sum_{cycle \in interval} \sum_{i=1}^{TotalNumberIntervals(cycle)} t_{final}(i, cycle) - t_{init}(i, cycle)}{Interval} * 100$$

- Headway: mean headway of the vehicles when crossing the detector during the interval (seconds)

$$Headway(interval) = \frac{\sum_{\substack{v \in Veh(interval) \\ v-1 \text{ previous of vehicle } v}} (EntranceTime(v, interval) - EntranceTime(v-1, interval))}{NbVeh(interval) - 1}$$

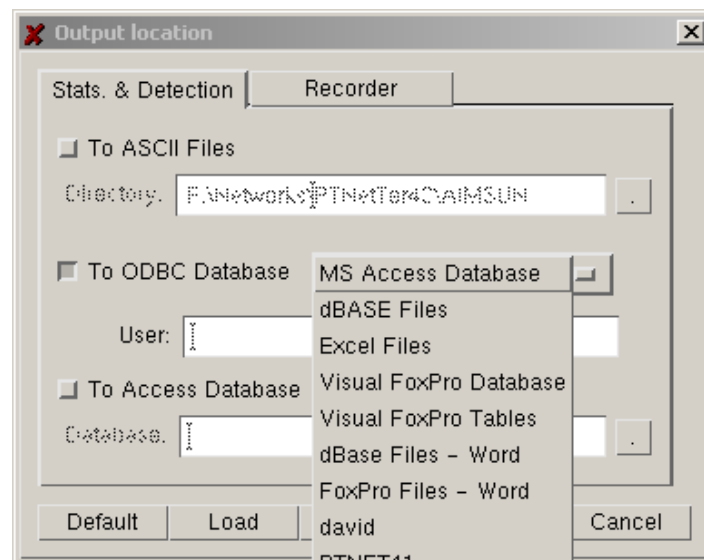
9.3 STORING SIMULATION RESULTS USING ODBC

A more flexible mechanism for storing simulation outputs has also been included. The user can opt to store the simulation outputs (statistics and detection) either as ASCII files, or as a database in an ODBC format. In both cases the user may select where to locate these data, making it possible to store the results of different runs or replications of the same model. The database format is described in detail in Appendix 2.

9.3.1 Output Location

To select the type of output support and its location, select the 'Experiment / Output / Output Location' command to display the Output Location dialog window (see Figure 9-22). Then select ASCII files, general ODBC Database or Access Database by clicking on the appropriate toggle button.

Figure 9-22: Definition of Output Location



When selecting 'To ASCII Files', the user can specify the directory in which to store the files, either by typing the whole path into the dialog box, or by browsing for the directory. If a Multiple Replication Experiment is run, different results corresponding to different replications are stored in different subdirectories using the replication identifier (1, 2, ... n) as the directory name.

When selecting 'To ODBC Database', the option menu containing the available Data Sources will become active, allowing the user can select one of them. Creation of Data Sources is explained in the next section. If the ODBC Database requires an username and a password then the 'User' and 'Password' fields has to be set.

When selecting 'To Access Database', the user can browse to search for the appropriate Access database, (file with the extension .mdb). The database can be an empty database, which can be created when the Data Source is defined or using Microsoft Access. If the database is not empty, the simulation output data having the same replications ID's will be override, otherwise it will be added to the existing data.

9.3.2 Creating a Data Source

In order to store the simulation results as a database, the user must first create a Data Source. An application named ODBC Data Sources is located in the Windows Control Panel folder. When it is run, the dialog window in Figure 9-23 appears.

Click on the 'Add' button and the 'Create New data Source' dialog window will appear, as shown in Figure 9-24. The user may then select between a Microsoft Access Driver (*.mdb) or an SQL Server. If you select Microsoft Access Driver and then click on the 'Finish' button, the ODBC Microsoft Access Setup dialog window is displayed (see Figure 9-25). Type in a Data Source Name as the logical name for your database. Then click on the 'Select' button to browse and search for a Microsoft Access file (*.mdb) or create an

empty database clicking on 'Create' button. Select or create a database and click on the 'OK' button. The new data source will then be added to the list of User Data Sources. From now on, the created data source, referenced by its logical name, can be used in AIMSUN to store simulation results.

When selecting the Database, the user can choose among the different Data Sources available in the computer. Only Data Sources that have been properly installed in the system will be shown in the option menu. If you are running Multiple Replications, results corresponding to different replications are stored in the same database, using the replication identifier as the primary key. The following section explains in detail the database structure and its contents.

Figure 9-23: ODBC Data Source Administrator

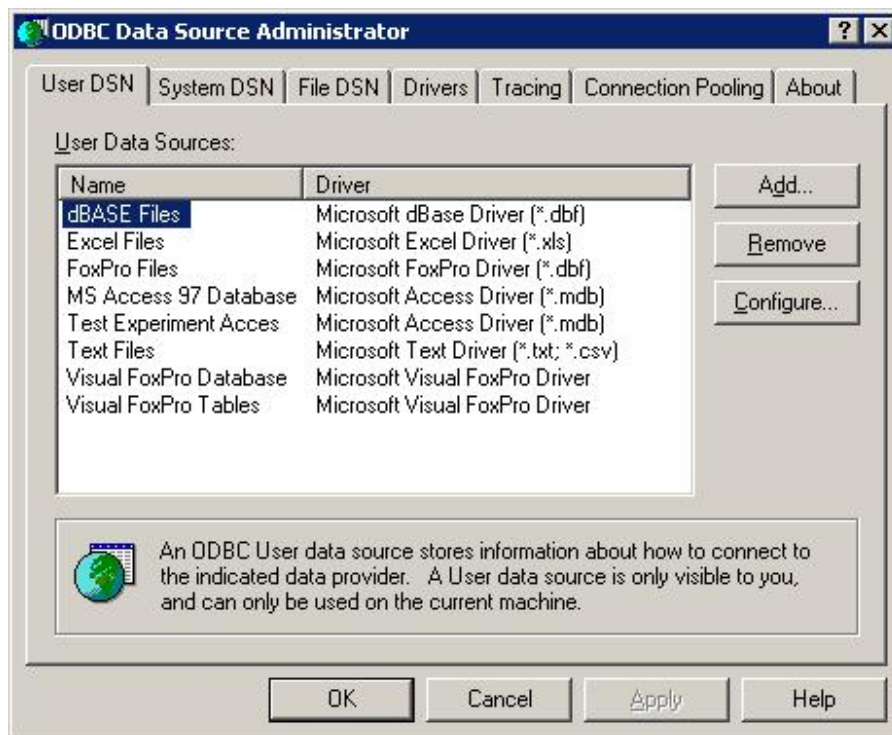
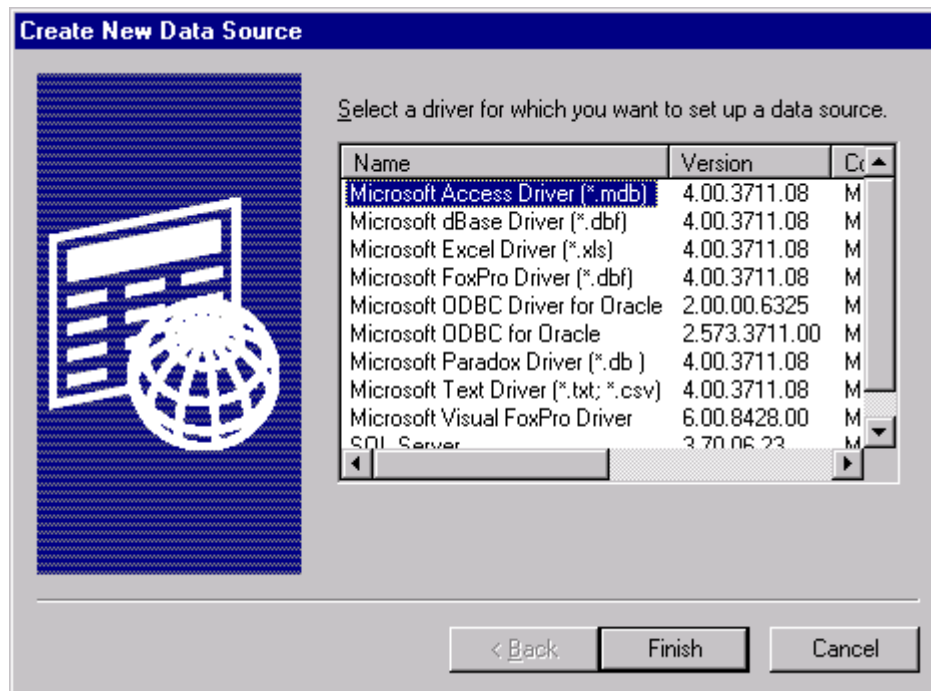
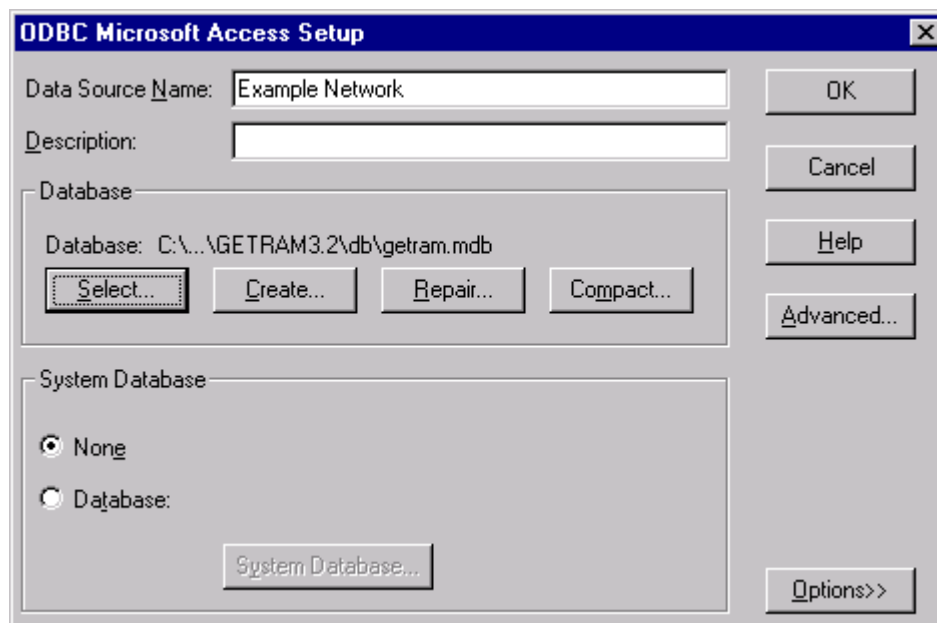
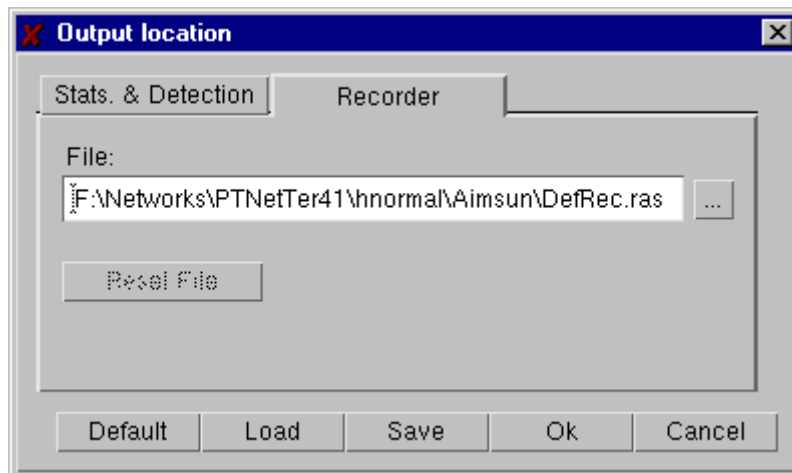


Figure 9-24: Create New Data Source dialog window**Figure 9-25: ODBC Microsoft Access**

9.4 RECORDED SIMULATION

It is now possible to record a running simulation and visualise it later using the AIMSUN 3D Application. The storage location for the recording is defined in the Output Location dialog window.

Figure 9-26: Output Location. Recorded Simulation



The Stats. & Detection folder already has the information defined in previous versions, and the new Recorder folder allows the user to define the recording file path. This file has the extension *.ras. Figure 9-26 shows the default path used (the Browse button allows the user to customise the path). The recorder information is kept in the same file as the other output location data.

The Reset File button is active when recording has started and the simulation is stopped. Pressing it deletes the current recording. When the user continues the simulation the recording is made to the same file.

To activate the recording press the REC toggle button (see 7.2.5), the simulation record also starts. If no recorder file is defined, the recorder toggle will not be available. During the simulation, the user can stop or start the recording again.

It is possible to change the file during the simulation. When recording over an existing file, previous recorded data will be lost.

10. ENVIRONMENTAL MODELS

AIMSUN provides two Environmental Models, namely the Fuel Consumption and Pollution Emission Models. The Environmental Models are automatically selected by default whenever the corresponding data is defined in the vehicle types library. If, for example, the user has defined Pollution data for any vehicle type, the Pollution Emission Model will be activated. The name of the model selected will be displayed in the Information Area at the bottom of the main window. The user can choose to activate or deactivate any model before running the simulation experiment. This is done via the extended menu showing the list of environmental models that appears when you select the 'Experiment / Particular Models' command. The following sections explain how these Particular Models work, describing their input requirements and the output generated.

10.1 FUEL CONSUMPTION MODEL

The AIMSUN Fuel Consumption Model assumes that each vehicle is either idling, cruising at a constant speed, accelerating or decelerating. The state of each vehicle is determined and the model then uses the appropriate formula to calculate the fuel consumed for this state.

For idling and decelerating vehicles the rate (in ml/s) can be assumed to be constant. For an accelerating vehicle it is given by the formula:

$$F_a = (c_1 + c_2 av)$$

where c_1 and c_2 are constants and a and v are the vehicle acceleration and speed respectively.

The following fuel consumption equation for a cruising vehicle moving at speed v , has been determined by Akcelic [AKC82]. It contains three constants: k_1 , k_2 , and v_m , which need to be determined empirically for each vehicle type.

$$\frac{dF}{dt} = k_1 \left(1 + \frac{v^3}{2v_m^3}\right) + k_2 v$$

v_m is the speed at which the fuel consumed per km is a minimum. Typically this is around 50 km/h.

The UK Department of Transport [DoT94] provides fuel consumption figures for all new cars. Amongst the figures given are the fuel consumption in litres per 100 km, for vehicles travelling at speeds of 90 km/h and 120 km/h. These figures can be used to determine the constants k_1 and k_2 above. It is easy to show that if F_1 and F_2 are the fuel consumption rates in litres per 100 km/h for a vehicle travelling at a constant speed of either v_1 or v_2 respectively, then:

$$k_1 = \frac{(F_1 - F_2)v_1v_2v_m^3}{180(2v_2v_m^3 - 2v_1v_m^3 + v_2v_1^3 - v_1v_2^3)}$$

$$k_2 = \frac{2F_2v_2v_m^3 - 2F_1v_1v_m^3 + F_2v_2v_1^3 - F_1v_1v_2^3}{360(2v_2v_m^3 - 2v_1v_m^3 + v_2v_1^3 - v_1v_2^3)}$$

Then, for each time step in the simulation, the state of each vehicle will be determined as either idling, accelerating, cruising or decelerating. The fuel consumed during the simulation time step, Δt , will then be calculated for each vehicle according to its state using the formulae given in Table 10.1

Table 10-1: Fuel consumed

Vehicle State	Fuel Consumed (ml) during Δt
Idling	$F_i \Delta t$
Accelerating with acceleration a (m/s/s) and speed v (m/s)	$(c_1 + c_2 a v) \Delta t$
Cruising at speed v (m/s)	$(k_1 (1 + (\frac{v}{v_m})^3) + k_2 v) \Delta t$
Decelerating	$F_d \Delta t$

where F_i and F_d are the fuel consumption rate in ml/s for idling and decelerating vehicles respectively and constants c_1 and c_2 need to be calibrated.

Input Parameters

For each vehicle type, the following additional six parameters, which specify the vehicle's fuel consumption rates, have to be specified:

- F_i : the fuel consumption rate for idling vehicles in ml/s
- c_1 and c_2 : the two constants in the equation for the fuel consumption rate for accelerating vehicles, F_a , in ml/s
- F_1 : the fuel consumption rate, in litres per 100 km, for vehicles travelling at a constant speed of 90 km/h
- F_2 : the fuel consumption rate, in litres per 100 km, for vehicles travelling at a constant speed of 120 km/h
- v_m : the speed at which the fuel consumption rate, in ml/s, is at a minimum for a vehicle cruising at constant speed
- F_d : the fuel consumption rate for decelerating vehicles in ml/s.

The following are example values of these input parameters, taken from [FER82] and the UK Department of Transport [DOT94].

The fuel consumption rate for idling vehicles in ml/s $F_i = 0.333$, c_1 and c_2 in the equation for the fuel consumption rate for accelerating vehicles: $c_1 = 0.420$, $c_2 = 0.260$. The fuel consumption rate for decelerating vehicles in ml/s $F_d = 0.537$. The fuel consumption rates for cruising vehicles for three different cars are:

Ford Fiesta: $F_1 = 4.7$ (l/100km at 90 km/h)
 $F_2 = 6.5$ (l/100km at 120 km/h)
 $v_m = 50$ km/h

Ford Escort: $F_1 = 5.4$ (l/100km at 90 km/h)
 $F_2 = 7.1$ (l/100km at 120 km/h)
 $v_m = 50$ km/h

Ferrari Testarossa $F_1 = 10.0$ (l/100km at 90 km/h)
 $F_2 = 11.4$ (l/100km at 120 km/h)
 $v_m = 70$ km/h

Output

Output produced by the Fuel Consumption model at the different levels of aggregation is as follows:

- For the entire network, the total distance travelled (in km) by all the vehicles having finished their trip and the total fuel consumed by all of them, in litres.
- For each section and turning, the total km travelled by all the vehicles that have crossed that section and the total fuel consumed by all of them, in litres.
- For each route, the total distance travelled (in km) by all the vehicles that have followed that route and the total fuel consumed by all of them, in litres.

10.2 POLLUTION EMISSION MODEL

AIMSUN can model the pollution emissions for all the vehicles in the simulation. As in the Fuel Consumption Model, the vehicle state (idling, cruising, accelerating or decelerating) and the vehicle speed / acceleration is used to evaluate the emission from each vehicle for each simulation step. This is done by referencing look-up tables for each pollutant, which give emissions (in g/s) for every relevant combination of vehicle behaviour, speed / acceleration. There are different sets of look-up tables for each vehicle type and for each pollutant.

At the moment, a maximum of three pollutants are considered, corresponding to the three most widely used pollutants (Carbon Monoxide, Nitrogen Oxides and unburned Hydrocarbons), but it is conceivable that additional pollutants might be modelled if data becomes available.

Input Parameters

The input required for the pollution emission model is as follows:

- For each vehicle type (i.e. cars, buses, trucks)
 - For each pollutant modelled (i.e. CO, NO_x, HC)
 1. Emission rate for accelerating vehicles in g/s
 2. Emission rate for decelerating vehicles in g/s
 3. Emission rate for idling vehicles in g/s
 4. A look-up table for vehicles cruising at a constant speed consisting of a set of pairs (speed break point (km/h), emission rate (g/s), for a maximum of 15 break points.

Emission values for petrol cars and buses, taken from QUARTET deliverable, are summarised in the following two tables:

Table 10-2: Pollution emission rates for cars

Emission rates for cars (g/s)	CO	NO _x	HC
Idling emission rate (g/s)	0.060	0.0008	0.0067
Accelerating emission rate (g/s)	0.377	0.0100	0.0200
Decelerating emission rate (g/s)	0.072	0.0005	0.0067
Cruising emission rate (g/s)			
10 km/h	0.060	0.0006	0.0063
20 km/h	0.091	0.0006	0.0078
30 km/h	0.130	0.0017	0.0083
40 km/h	0.129	0.0022	0.0128
50 km/h	0.090	0.0042	0.0097
60 km/h	0.110	0.0050	0.0117
70 km/h	0.177	0.0058	0.0136

Table 10-3: Pollution emission rates for buses

Emission rates for buses (g/s)	CO	NO _x	HC
Idling emission rate (g/s)	0.050	0.0050	0.0383
Accelerating emission rate (g/s)	0.377	0.0100	0.0200
Decelerating emission rate (g/s)	0.072	0.0005	0.0067
Cruising emission rate (g/s)			
10 km/h	0.097	0.018	0.078
20 km/h	0.056	0.020	0.044
30 km/h	0.050	0.023	0.042
40 km/h	0.069	0.036	0.056
50 km/h	0.056	0.067	0.078
60 km/h	0.042	0.083	0.067
70 km/h	0.000	0.133	0.067

Output

Output produced by the pollution model at the different levels of aggregation is:

- For the entire network, the total distance travelled (in km) by all the vehicles that have finished their trip, and the kilograms of each pollutant emitted by them all.
- For each section and turning, the total km travelled by all the vehicles that have crossed that section, and kilograms of each pollutant emitted by them all.
- For each route, the total distance travelled (in km) by all the vehicles that have followed that route, and kilograms of each pollutant emitted by them all.

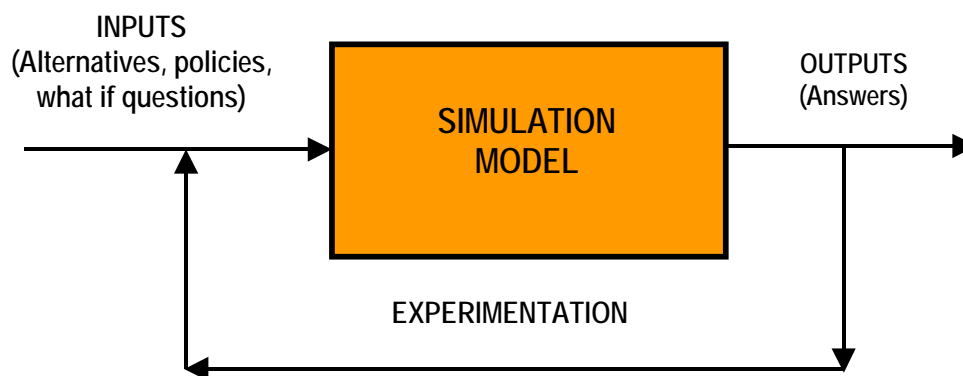
11. CALIBRATION AND VALIDATION OF AIMSUN MODELS

11.1 METHODOLOGY FOR BUILDING SIMULATION MODELS

From a methodological point of view it is widely accepted that simulation is a useful technique to provide an experimental test bed to compare alternate system designs, replacing the experiments on the physical system by experiments on its formal representation in a computer in terms of a simulation model. The outcomes of the computer experiment provide in this way the basis for a quantitative support to decision-makers. According with this conception the simulation model can be seen as computer laboratory to conduct experiments with the model of the system with the purpose of drawing valid conclusions for the real system. In other words, use the simulation model to answer what if questions about the system.

Simulation may be then seen as a sampling experiment on the real system through its model [PID92]. In other words, assuming that the evolution over time of the system model imitates properly the evolution over time of the modelled system, samples of the observational variables of interest are collected from which, using statistical analysis techniques, conclusions on the system behaviour can be drawn. Figure 11-1 illustrates conceptually this methodology.

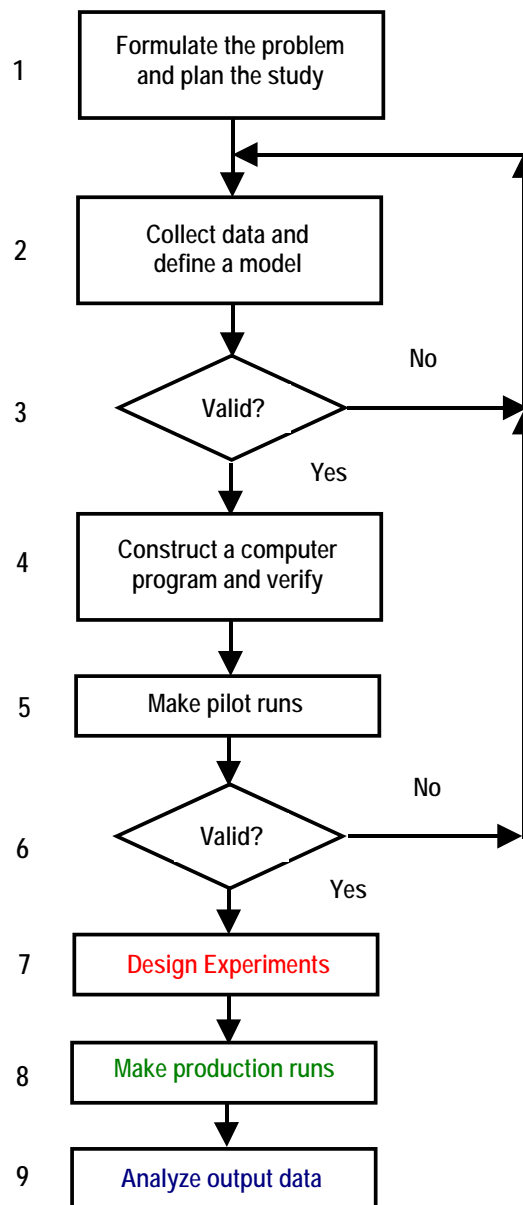
Figure 11-1. Experimental Nature of Simulation



The reliability of this decision making process depends on the ability to produce a simulation model representing the system behaviour closely enough for the purpose of using the model as a substitute of the actual system for experimental purposes. This is true for any simulation analysis in general and obviously for traffic simulation. The process of determining whether the simulation model is close enough to the actual system is usually achieved through the validation of the model, an iterative process involving the calibration of the model parameters and comparing the model to the actual system behaviour. The discrepancies between the two, and the insight gained, are then used to improve the model until the accuracy is judged to be acceptable. Validation of a simulation model is a concept that should be taken into account thorough the whole model building process.

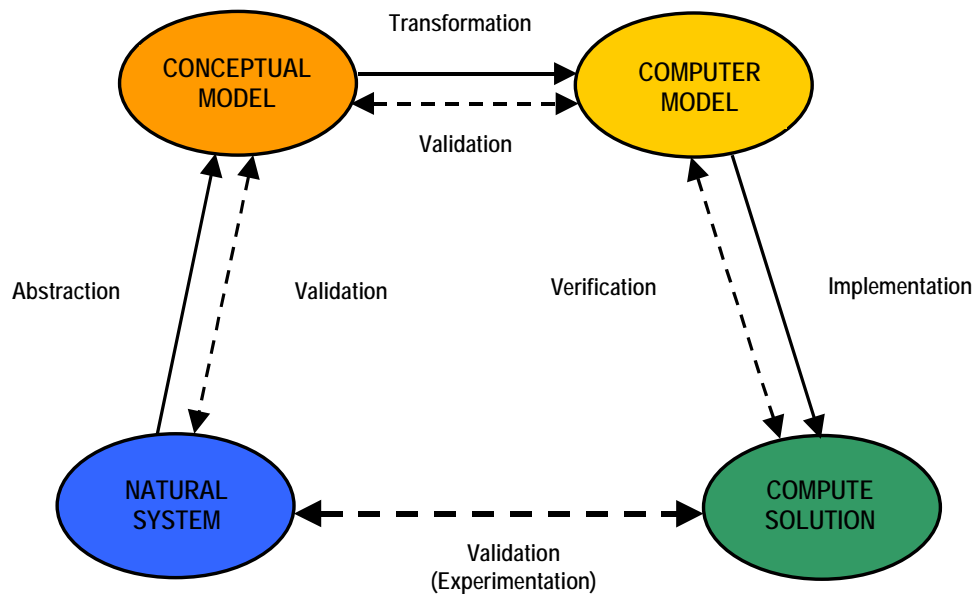
Figure 11-2 depicts the diagram of the methodological process of a simulation study [LAW91], which consists generally of the following steps:

1. Formulate the problem and plan the study: identify the nature of the problem and the requirements to find a solution.
2. Collect data and formulate the model: acquire empirical evidence (knowledge acquisition process) on the system's behaviour to formulate hypothesis, and translate them in terms of the formal representation that constitutes the model of the system.
3. Check whether the model built is a valid representation of the system for the purposes of the study, that is, verify that the answers provided by the model to the what if questions can be accepted as valid ones.
4. Translate the formal model in terms of a computer program

Figure 11-2: Steps in a simulation study

5. and 6. Check that the computer model performs correctly, that is, is error free and provides acceptable results.
7. Identify the design factors that translate the “what if” questions in terms of computer experiments. Specify the experimental sampling procedures to gather the data for the statistical analysis that will provide the expected answers.
8. And 9. Conduct the simulation experiments on the computer and analyse the outputs.

The main conceptual steps in the model building and use process are further detailed in the diagram in figure 11-3. The process of knowledge acquisition can also be interpreted in terms of an abstraction of the reality, the natural system under study, leading in first terms to a primary representation of the system or conceptual model. This conceptual model can be the object of a primary validation exercise in order to check that all main components of the system are being taken into consideration, and are suitably represented in terms of their attributes. This means a refinement in the meaning of validation. Validation is therefore an activity that should be realised at each step of the process, and not only at the end.

Figure 11-3: Detailed methodological steps of the model building and use process

Translating the conceptual model in terms of a mathematical representation for which a numerical algorithm is available, can also be understood in terms of building a suitable computer model, given that, for large systems, the modelling process is feasible only if suitable computing tools are available. Computer models should themselves be objects of verification, checking that the computer model is error free, and validation, that is, checking that the computer model does what is expected it should do.

The error free computer model can then be implemented and executed to provide the solutions that will be the object of the last verification. This last verification exercise consists very often of the comparison with the observed reality. The validated computer model will then become the “laboratory” to conduct the simulation experiments that suitably designed will answer the questions, very often “what if questions”, about the system behaviour under the various design alternatives that configure the experimental scenarios.

11.2 THE VALIDATION PROCESS: BUILDING VALID AND CREDIBLE SIMULATION MODELS

According to Law and Kelton [LAW91]:

- **Verification:** consists of determining that a simulation computer program performs as intended, i.e. debugging the computer program. Thus, verification checks the translation of the conceptual simulation model into a correctly working program.
- **Validation:** is concerned with determining whether the conceptual simulation model (as opposed to the computer program) is an accurate representation of the system under study. If a model is valid, then the decision made with the model should be similar to those that would be made by physically experimenting with the system (if this were possible)
- A model is **credible** when its results are accepted by the user, and are used as an aid in making decisions.

The importance of model credibility is the major reason for the widespread interest in animating simulation output, since animation is an effective way for an analyst to communicate the essence of a model to a client.

Verification usually implies running the simulation model under a variety of settings of the input parameters and check to see that the output is reasonable. In some cases, certain measures of performance may be computed exactly and used for comparison. Animation can also be of great help for this purpose, with some types of simulation model (traffic model are just a good example) it may be helpful to observe an animation of the simulation output to establish whether the computer model is working as expected.

In validating a simulation model analyst should not forget that:

- A simulation model of a complex system can only be an approximation to the actual system, regardless of how much effort is put into developing the model. There is no such thing as an absolutely valid model of a system.
- A simulation model should always be developed for a particular set of purposes.
- A simulation model should be validated relative to those measures of performance that will actually be representative of these purposes, and will therefore be used for decision making.
- Model development and validation should be done hand-in-hand thorough the entire simulation study,

Law and Kelton propose a three-step approach for developing valid and credible simulation models:

1. Develop a model with high face validity

A model that, on the surface, seems reasonable to people who is knowledgeable about the system under study.

- The modeller should work closely with people who are intimately familiar with the system.
- The analyst will have to be resourceful in order to obtain all of the required information.
- If data exist, then they should be obtained and used for building the model. These data may be available from historical records or may have to be collected during a time study.
- Care must be taken to ensure that data are correct and representative of what is being modelled.

2. Test the assumptions of the model empirically

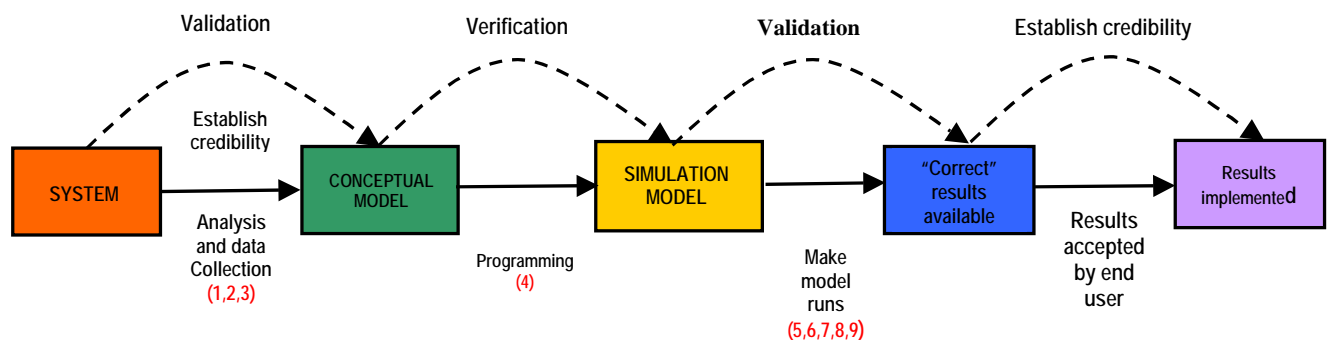
- If a theoretical probability distribution has been fitted to some observed data and used as input to the simulation model, the adequacy of the fit can be assessed.
- *Sensitivity analysis:* Determine if the simulation output changes significantly when the value of an input parameter is changed, when an input probability distribution is changed, or when the level of detail for a subsystem is changed.

3. Determine how representative the simulation output data are

- The most definitive test of a simulation model's validity is establishing that its output data closely resemble the output data that would be expected from the system. If the simulation output data compare favourably with the system output data then the model can be considered "valid".
- Since the model is only an "approximation" to the actual system, a null hypothesis that the model and the system are the "same" is clearly false. We believe that it is more useful to ask whether or not the differences between the system and the model are significant enough to affect the conclusions derived from the model.
- Animation may also be an effective way to evaluate (at least qualitatively) the validity of a simulation model.

The methodology proposed by Law and Kelton for validating, verifying and establishing the credibility of a simulation model is summarised in figure 11-4. Boxes in the diagram represent the states of the model and their relationships to the stages in a simulation study as depicted in figure 11-2. Curved dashed arrows show where the three concepts are employed. Although not represented in the figure one should not forget that the whole process must be understood as an iterative process that could eventually require the repetition of some steps, therefore backward arrows representing a feedback step should be added.

Figure 11-4. Timing and relationships of validation, verification and establishing credibility

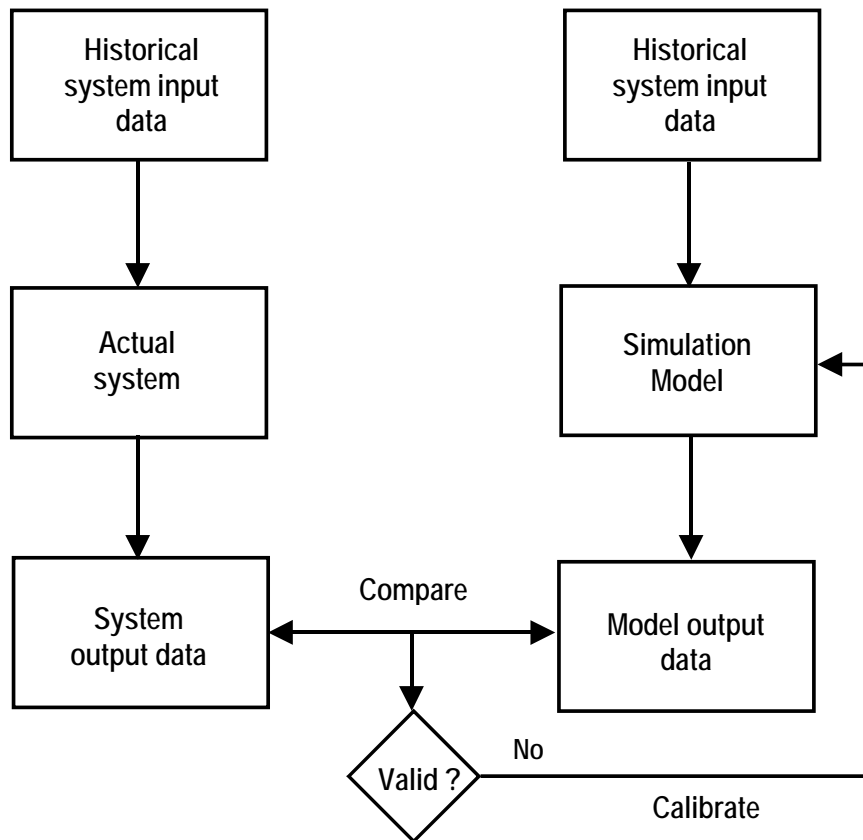


Understanding that validation means the process of testing that the model does actually represent a viable and useful alternative means to real experimentation. This requires the exercise of calibrating the model, that is adjusting model parameters until the resulting output data agree closely to the system observed data. We must ask whether this process produces a valid model for the system in general, or whether the model is only representative of the particular set of input data. To answer this question properly, the analyst must use two independent sets of data, one for calibration and another one for validation. The first set should be used for calibrating the model parameters and the second for running the calibrated model and then for validating the calibrated model. The resulting model output data are compared to the second set of system output data.

Validation of the simulation model will be established on the basis of the comparison analysis between the observed output data from the actual system and the output data provided by the simulation experiments conducted with the computer model.

The conceptual framework for this validation methodology is described in the diagram of figure 11-5 (adapted from [BAL98]). According to this logic when the results of the comparison analysis are not acceptable to the degree of significance defined by the analyst, the rejection of the simulation results implies the need of re-calibrating some aspects of the simulation model, adjusting some parameters etc. The process is repeated until a significant degree of similarity according to some statistical analysis techniques is achieved. An overview of such techniques is provided in Section 11.4.

Figure 11-5: Methodological framework for the validation of the simulation model by comparison with the observed system



11.3 SPECIFICS FOR THE VALIDATION OF TRAFFIC SIMULATION MODELS

In traffic systems, the behaviour of the actual system is usually defined in terms of the traffic variables, flows, speeds, occupancies, queue lengths, and so on, which can be measured by traffic detectors at specific locations in the road network. To validate the traffic simulation model, the simulator should be able to emulate the traffic detection process and produce a series of simulated observations. A comparison with the actual measurements will be used to determine whether the desired accuracy in reproducing the system behaviour is achieved. Statistical techniques will be used for such determination. In particular for the validation and calibration phases, the general methodology proposed in Section 11.2 according to Law and Kelton [LAW91] can be specialised as follows for traffic simulation systems. To produce input data for the simulation model of the quality required to conduct an accurate analysis, a careful data collection process is required. [HUG98] provides an excellent case study on how to proceed.

11.3.1 Develop a traffic simulation model with high face validity with GETRAM

The main components of a traffic micosimulation model are:

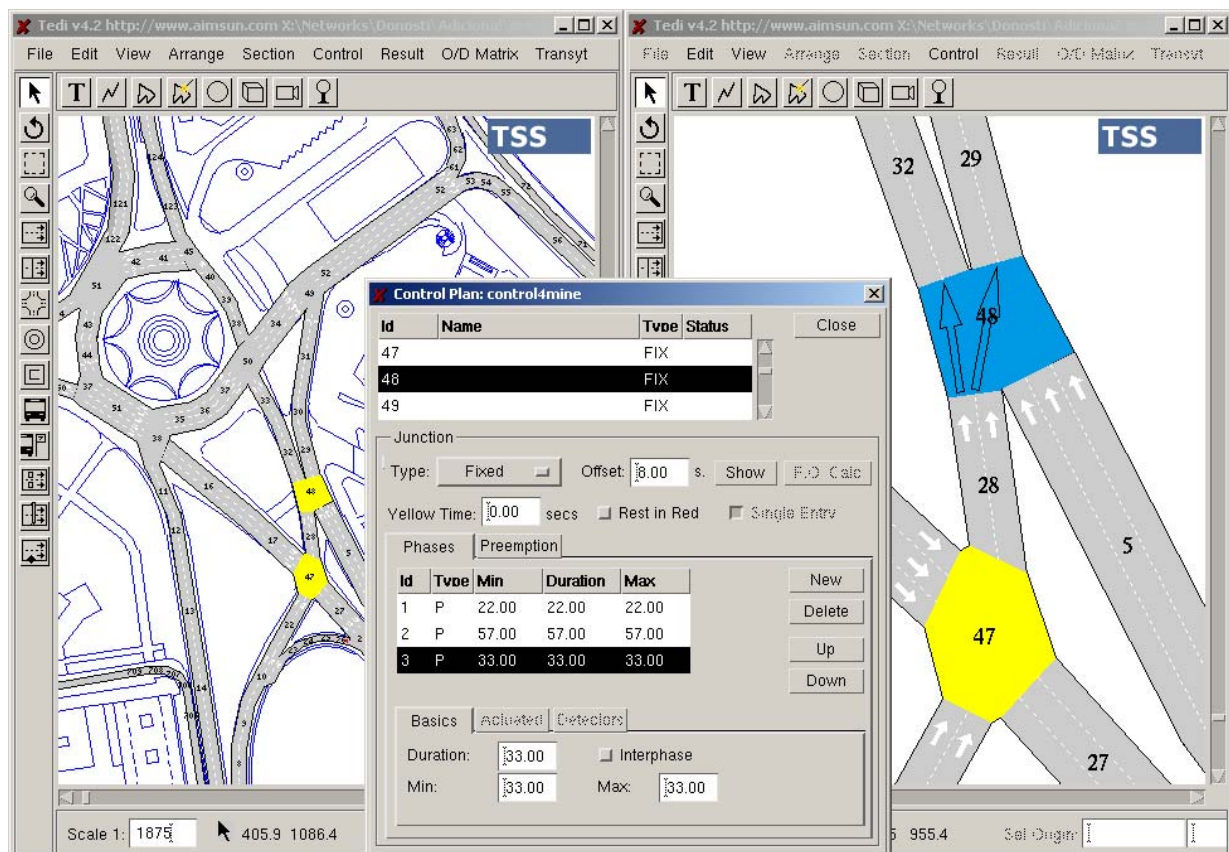
- a) The geometric representation of the road traffic network and related objects (traffic detectors, Variable Message Panels, traffic lights, etc.)
- b) The representation of traffic management schemes (directions of vehicle's movement, allowed and banned turnings, etc.), and of traffic control schemes (phasing, timings, offsets)
- c) The individual vehicles behavioural models: car following, lane change, gap acceptance, etc.
- d) The representation of the traffic demand:
 - d.1) Input flow patterns at input sections to the road model and turning percentages at intersections
 - d.2) Time-sliced OD matrices for each vehicle class
- e) The route choice models

The TEDI suite of graphic editors in GETRAM has been designed with the objective of supporting the user in tasks a) and b) of the process of building the road network model. To facilitate these tasks the editor accepts as a background a digital map of the road network, in terms of a DXF file from a GIS or an AutoCAD system, a JPEG or a bitmap file, etc. so sections and nodes can be built subsequently into the foreground. The editor supports both urban and interurban roads, which means that the level of detail covers elements such as surface roads, entrance and exit ramps, intersections, traffic lights and ramp metering. Figure 11-6 illustrates the process of using the graphical editor to build an urban model on the top of a background.

With respect to the methodological diagram in figure 11-3, TEDI provides the model builder with a software tool to go directly from the natural system, the road network object of study, to its computer representation in terms of the GETRAM model. The use of the graphic editors on the digital maps of the road networks provides the basis for a continuous visual validation of the quality of the geometric model. At the same time, the auxiliary on-line debugging tools in TEDI prevent the most blatant mistakes in building the geometric representation, warning the modeller when obvious inconsistencies may occur.

In other words, the TEDI model building process is assisted with validation tools to check the correctness of the geometric model of the road network within the limits of logic rules. Some aspects may lie beyond the analysis capabilities of the assistance software, i.e. whether banning a turning is correct or not. This decision may depend on the objectives of the traffic management scheme defined by the traffic manager. Something similar could be said regarding whether or not to include a movement in a phase. However, a different case might be that of a previously defined movement that was not included in any phase, this is something that can be checked by the assistance tools.

In this way, TEDI ensures a geometric model exhibiting a "high face validity" that could even be further validated by the modeller through visual inspection facilitated by the graphic display of the GETRAM model.

Figure 11-6. Building and checking the validity of a geometric road network model with TEDI

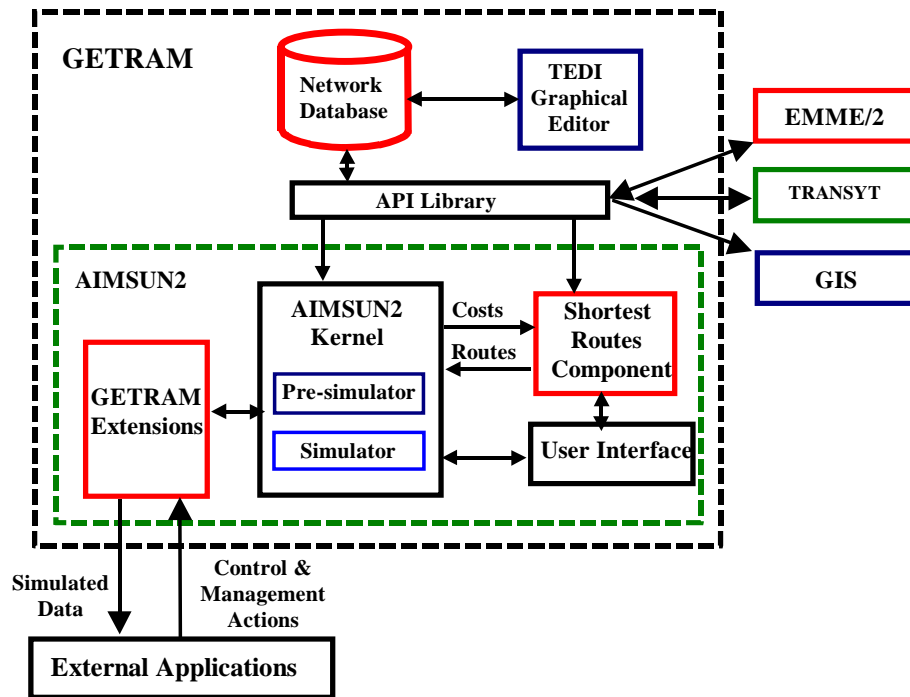
The validation of the geometric model should also take into account other aspects that may not be so obvious at a first glance, i.e. the connectivity of the network, the existence of paths connecting every origin to every destination in a route based simulation.

To assist the modeller in this additional validation task, before running the simulation the GETRAM/AIMSUN software system performs a pre-simulation step, as shown in the conceptual diagram of figure 11-7. When a model inconsistency is identified at this level a warning, or an error message, depending on the case, is issued informing the modeller of the type of inconsistency, and where in the model it is located.

11.3.2 Test the assumptions of the model empirically.

In the case of an AIMSUN traffic simulation model, the model behaviour depends on a wide variety of model parameters. In summary, if one considers the model to be composed of entities, (i.e. vehicles, sections, junctions, intersections, and so on), each of them described by a set of attributes, (i.e. parameters of the car-following, the lane change, gap acceptance, speed limits and speed acceptance on sections, and so on), the model behaviour is determined by the numerical values of these parameters. The calibration process will therefore have the objective of finding the values of these parameters that will produce a valid model.

Some examples may help to illustrate this dependency between parameter values and model behaviour. Vehicle lengths have a clear influence in flows: as the vehicle lengths increase flows decrease and queue lengths increase. In the AIMSUN car-following model the target speed, the section speed limit and the speed acceptance, among others, define the desired speed for each vehicle on each section. The higher the target speed, the higher the desired speed for any given section, resulting in an increase in flow according to the flow-speed relationships. In this way, as part of the calibration process one should establish for a particular model the influence of acceleration and braking parameters in the capacity of the sections, namely for weaving sections. Similarly, the effects of lengths of Zones 1 and 2 in the lane change model influence the capacity of the weaving sections as well as the percent overtake and percent recover parameters influence the lane distribution, and so on.

Figure 11-7: Conceptual architecture of GETRAM/AIMSUN: the pre-simulation function in AIMSUN

11.4 STATISTICAL METHODS FOR MODEL VALIDATION

The statistical methods and techniques for validating simulation models are clearly explained in most textbooks and specialised papers [LAW91], [BAL98] and [KLE95]. What follows is an adaptation of the general process to our problem of validating a microscopic simulation model. The measured data in the actual system should be split into two data sets: the data set that will be used to develop and calibrate the model, and a separate data set that will be used for the validation test.

At each step in the iterative validation process a simulation experiment will be conducted. Each of these simulation experiments will be defined by the data input to the simulation model and the set of values of the model parameters that identify the experiment. The output of the simulation experiment will be a set of simulated values of the variables of interest, in the case of our example the flows measured at each traffic detector in the road network at each sampling interval. For example, assuming that in the definition of the simulation experiment the sampling interval is five minutes, that is the model statistics are gathered every five minutes, and that the sampling variable is the simulated flow w , the output of the simulation model will be characterised by the set of values w_{ij} , of the simulated flow at detector i at time j , where index i identifies the detector ($i=1,2,\dots,n$, being n the number of detectors), and index j the sampling interval ($j=1,2,\dots,m$, being m the number of sampling intervals in the simulation horizon T). If v_{ij} are the corresponding actual model measures for detector i at sampling interval j , a typical statistical technique to validate the model would be compare both series of observations to determine if they are close enough. For detector i the comparison could be based on testing whether the difference.

$$d_i = w_{ij} - v_{ij}, j=1,\dots,m$$

has a mean \bar{d}_i significantly different from zero or not. This can be determined using the t-statistics:

$$\bar{t}_{m-1} = \frac{\bar{d}_i - \delta_i}{\bar{s}_d / \sqrt{m}}$$

where δ_i is the expected value of \bar{d}_i and \bar{s}_d the standard deviation of \bar{d}_i , for testing the null hypothesis:

$$H_0 : \delta_i = 0 \quad (|\bar{t}_{m-1}| > t_{m-1;\alpha/2})$$

- a) If for $\delta_i = 0$ the calculated value \bar{t}_{m-1} of the Student's t distribution is significant to the specified significance level α then we have to conclude that the model is not reproducing close enough the system behaviour and then we have to reject the model.
- b) If $\delta_i = 0$ gives a non-significant \bar{t}_{m-1} then we conclude that the simulated and the real means are “practically” the same so the simulation is “valid enough”.

This process will be repeated for each of the n detectors. The model is accepted when all detectors (or a specific subset of detectors, depending on the model purposes and taking into account that the simulation is only a model, and therefore an approximation, so δ_i will never be exactly zero) pass the test.

So far the statistical method, however, there is some special considerations to take into account [KLE95] namely in the case of the traffic simulation analysis.

1. The statistical procedure assumes identically and independently distributed (i.i.d) observations whereas the actual system measures and the corresponding simulated output for a time series. Therefore it would be desirable that at least the m paired (correlated) differences $d_i = w_{ij} - v_{ij}$, $j=1,\dots,m$ are i.i.d. This can be achieved when the w_{ij} and the v_{ij} are average values of independently replicated experiments.
2. The bigger the sample is, the smaller the critical value $\bar{t}_{m-1;\alpha/2}$ is, and this implies that a simulation model has a higher chance of being rejected at the sample grows bigger. Therefore the t statistics may be significant and yet unimportant if the sample is very large, and the simulation model be good enough for practical purposes.

These considerations lead us to recommend that you do not rely on only one type of statistical test for validating the simulation model. An alternative test is to check whether w and v are positively correlated, that is test the significance of the null hypothesis:

$$\mathbf{H}_0 : \rho > 0 \text{ } (\rho \text{ linear correlation coefficient})$$

This represents a less stringent validation tests accepting that simulated and real responses do not necessarily have the same mean and that what is significant is whether or not they are positively correlated. The test can be implemented using the ordinary least squares technique to estimate the regression model:

$$\mathbf{E}(v|w) = \beta_0 + \beta_1 w + \varepsilon \text{ } (\varepsilon \text{ random error term})$$

The test concerns then the one-sided hypothesis $\mathbf{H}_0 : \beta_1 \leq 0$. The null hypothesis is rejected and the simulation model accepted if there is strong evidence that the simulated and the real responses are positively correlated. The variance analysis of the regression model is the usual way of implementing this test. This test may be strengthened, becoming equivalent to the first test if this hypothesis is replaced by the composite hypothesis $\mathbf{H}_0 : \beta_0 = 0$ and $\beta_1 = 1$, implying that the means of the actual measurements and the simulated responses are identical and when a systems measurement exceeds its mean then the simulated observation exceeds its mean too.

A third family of statistical tests for the validation of simulation model is rooted in the former observation that the measured and the simulated series, v_{ij} and w_{ij} respectively, are time series. In this case the measured series could be interpreted as the original one and the simulated series the “prediction” of the observed series. In that case the quality of the simulation model could be established in terms of the quality of the prediction, and that would mean to resort to time series forecasting techniques for that purpose. If one considers that what is observed as output of the system as well as output of the model representing the system is dependent on two type of components: the functional relationships governing the system (the pattern) and the randomness (the error), and that the measured as well as the observed data are related to these components by the relationship:

$$\mathbf{Data} = \mathbf{pattern} + \mathbf{error}$$

This means that the critical task in forecasting can be interpreted in terms of separating the pattern from the error component so that the former can be used for forecasting. The general procedure for estimating the pattern of a relationship is through fitting some functional form in such a way as to minimise the error component. A way of achieving that could be through the regression analysis as in the former test.

If for detector i the error of the j -th “prediction” is $d_i = w_{ij} - v_{ij}$, $j=1, \dots, m$, then a typical way of estimating the error of the predictions for the detector i is “Root Mean Square Error”, \mathbf{rms}_i defined by:

$$\mathbf{rms}_i = \sqrt{\frac{1}{m} \sum_{j=1}^m (w_{ij} - v_{ij})^2}$$

This error estimate has been perhaps the most used in traffic simulation, and although obviously the smaller \mathbf{rms}_i is the better the model is, it has quite a significant drawback, as far as it squares the error, thereby emphasising large errors. Therefore, it would be helpful to have a measure that considers the disproportionate weight of large errors and also provides a basis for comparison with other methods.

Theil’s U-Statistic [THE66] is the measure achieving these objectives. In general, if X_j is the observed and Y_j the forecasted series, $j = 1, \dots, m$, then, if $\mathbf{FRC}_{j+1} = \frac{Y_{j+1} - X_j}{X_j}$ is the forecasted relative change, and

$\mathbf{ARC}_{j+1} = \frac{X_{j+1} - X_j}{X_j}$ is the actual relative change, Theil’s U-Statistic is defined as:

$$U = \sqrt{\frac{\sum_{j=1}^{m-1} (\text{FRC}_{j+1} - \text{ARC}_{j+1})^2}{\sum_{j=1}^{m-1} (\text{ARC}_{j+1})^2} \cdot \frac{(m-1)}{(m-1)}} = \sqrt{\frac{\sum_{j=1}^{m-1} \left(\frac{Y_{j+1} - X_{j+1}}{X_j} \right)^2}{\sum_{j=1}^{m-1} \left(\frac{X_{j+1} - X_j}{X_j} \right)^2}}$$

An immediate interpretation of Theil's U-Statistic, is the following:

$$\begin{aligned} U = 0 &\Leftrightarrow \text{FRC}_{j+1} = \text{ARC}_{j+1}, \text{ and then the forecast is perfect} \\ U = 1 &\Leftrightarrow \text{FRC}_{j+1} = 0, \text{ and the forecast is as bad as possible} \end{aligned}$$

In this last case the forecast is the same as that which would be obtained forecasting no change in the actual values. When forecasts Y_{j+1} are in the opposite direction of X_{j+1} then the U statistic will be greater than unity. Therefore the closer to zero the Theil's U-Statistic is, the better the forecasted series is or, in other words, the better the simulation model. When Theil's U-Statistic is close to or greater than 1, the forecasted series and therefore the simulation model, should be rejected.

When the forecast efficiency is based on the regression model $E(v|w) = \beta_0 + \beta_1 w + \varepsilon$ (ε random error term), the most efficient forecast would correspond to $\beta_0 = 0$ and $\beta_1 = 1$, that can be tested by the application of variance analysis to the regression model as indicated earlier. But taking into account that the average squared forecast error:

$$D_m^2 = \frac{1}{m} \sum_{j=1}^m (Y_j - X_j)^2$$

can be decomposed (Theil) in the following way:

$$D_m^2 = \frac{1}{m} \sum_{j=1}^m (Y_j - X_j)^2 = (\bar{Y} - \bar{X})^2 + (S_Y - S_X)^2 + 2(1 - \rho)S_Y S_X$$

where \bar{Y} and \bar{X} are the sample means of the forecasted and the observed series respectively, S_Y and S_X are the sample standard deviations and ρ is the sample correlation coefficient between the two series, the following indices can be defined:

$$\left. \begin{aligned} U_M &= \frac{(\bar{Y} - \bar{X})^2}{D_m^2} \\ U_S &= \frac{(S_Y - S_X)^2}{D_m^2} \\ U_C &= \frac{2(1 - \rho)S_Y S_X}{D_m^2} \end{aligned} \right\} \Rightarrow U_M + U_S + U_C = 1$$

U_M is the "Bias proportion" index and can be interpreted in terms of a measure of systematic error, U_S is the "variance proportion" index and provides an indication of the forecasted series ability to replicate the degree of variability of the original series or, in other words, the simulation model's ability to replicate the variable of interest of the actual system. Finally, U_C or "Covariance Proportion" index is a measure of the unsystematic error. The best forecasts, and hence the best simulation model, are those for which U_M and U_S do not differ significantly from zero and U_C is close to unity. It can be shown that this happens when β_0 and β_1 in the regression do not differ significantly from zero and unity respectively.

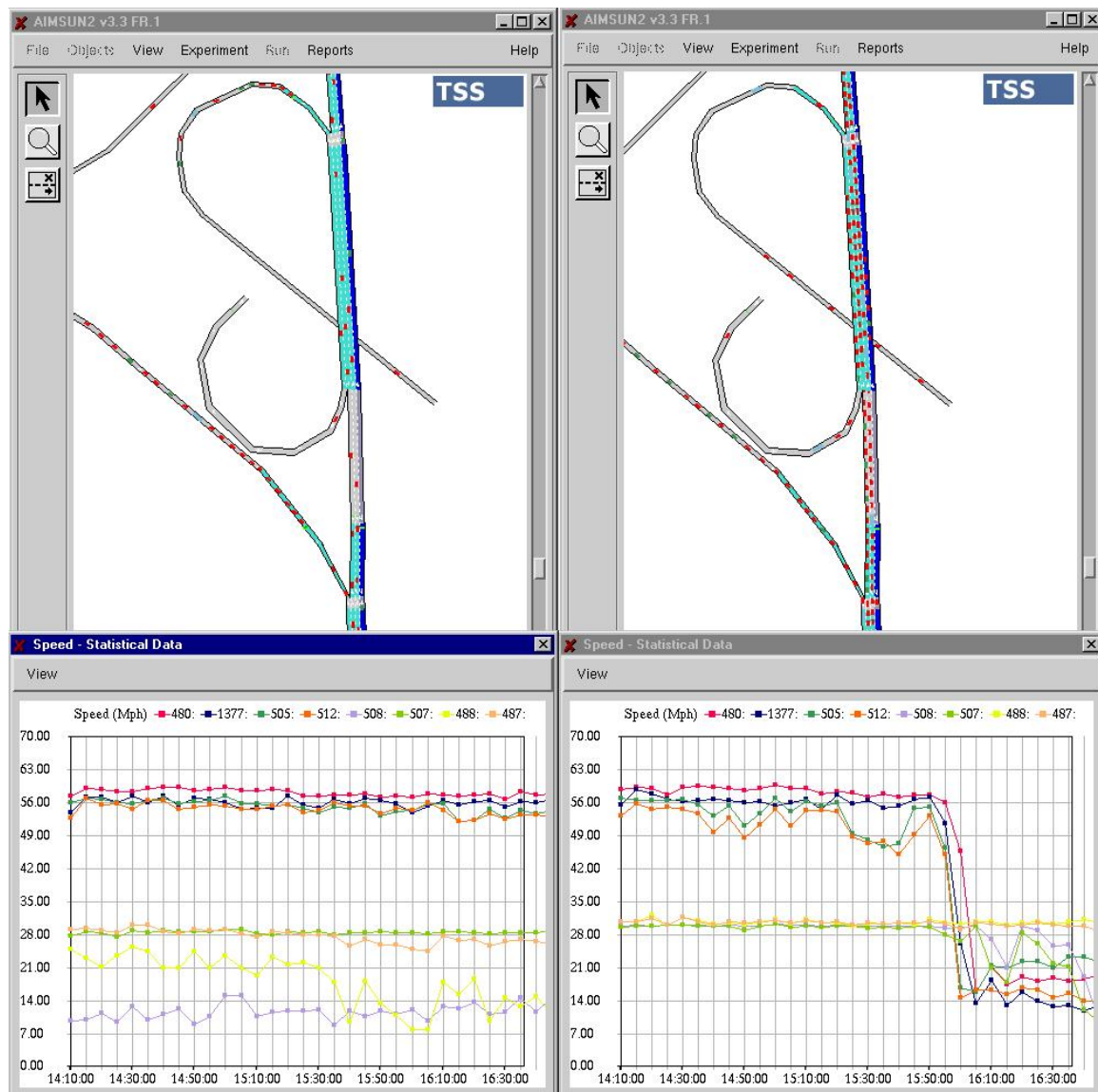
11.5 A CASE STUDY: THE I-35W FREEWAY IN MINNEAPOLIS

This is an example of a feasibility study on the suitability of integrated ramp control strategies for freeway corridors done by the simulation test of the ramp control logic developed by the Minnesota Department of Transportation (MnDOT) on a segment of the I-35W in Minneapolis. The work was done at the ITS Laboratory of the University of Minnesota.

11.5.1 Test Site Description

Following discussions with the MnDOT engineers in charge of the Transport Management Centre, a 24 km long section of I-35W going south was selected for testing purposes. This section was specifically chosen as it includes most of the common geometry configurations found in the Twin Cities. This section begins at Downtown Minneapolis and ends at the interchange with Highway 13. It includes 20 exit and 22 entrance ramps, which are controlled during PM peak hours. Four entrances are freeway to freeway ramps, carrying very high volumes in the range of 1200 veh/hr with long spill-back queues. The geometry includes 6 weaving sections and also has a lane drop section. The test site is divided into three zones and has three bottleneck locations. It also has a single HOV lane from I-494 interchange to Highway 13 that is about 10 km long. The total experiment was based on data collected during a 60 day period during May and June 1999. Most of this data was used to calibrate simulation model parameters.

Figure 11-8: Speeds at the interchange node in I-35W with ramp metering (left) and without (right)



11.5.2 Simulation Results

After it was deemed that the simulator was working as close as possible to real life conditions, one day's worth of data from the above period was used for the evaluation. The experiment consisted of two test cases, one involving normal congestion levels and the other where the previous demands were uniformly increased by 20%. In each case, two simulations were performed with and without ramp control. The Measures of Effectiveness (MOEs) collected included Total Travel Time (TTT) in veh-hrs and Total Delay (TD) in veh-hrs separately for the mainline and the ramps and Total Travel (TT) in veh-km for the whole network.

The effectiveness of ramp control for normal and heavy congestion was established. TTT in the mainline decreased by 46% when control was introduced under normal congestion. This can be explained by the fact that with ramp control density remains below critical at the bottleneck. As a result, higher speeds were achieved. Total ramp delays increased substantially as expected but overall system TTT and delays were reduced by 34.61% and 61.78% respectively. For the heavy congestion case, the system TTT decreased by 24.39% and TD by 39.41%. Similar improvements were also realised in the remaining MOE's. In general, in both cases with control, higher speeds were achieved and the flow was smoother throughout the freeway, as can be appreciated in figure 11-8. In it we can appreciate the evolution of average speeds during a 160 minute period with and without adaptive ramp metering.

The results of this testing simply confirmed that the ramp control strategy, improved the operating conditions on the freeway significantly on the overall system, especially with heavy congestion. This of course, was not unexpected. However, quantification of the results became a much easier task thanks to AIMSUN.

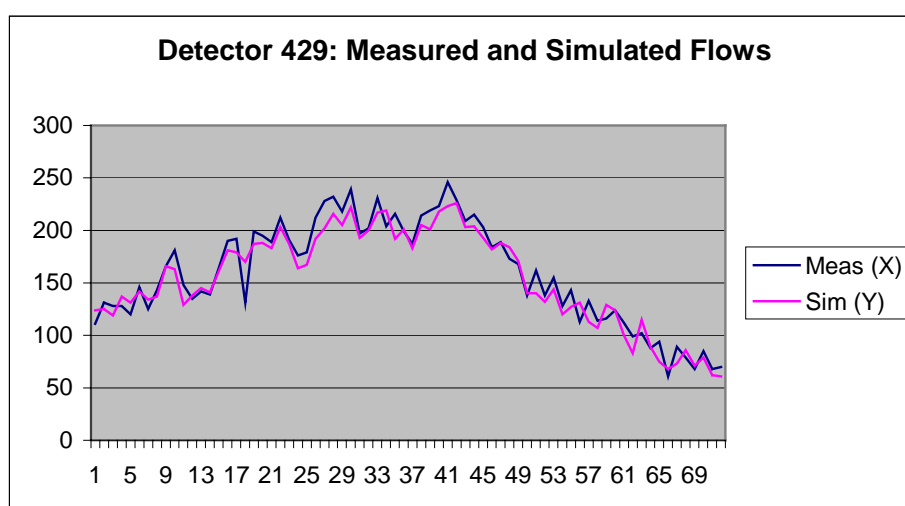
11.5.3 Model Validation

To validate the simulation model we should be able of using as input data the same input data as in the actual system. For example, in the case of the I-35W freeway in Minneapolis [KOK00], inputs are time dependent flows at input sections and turning percentages at exit sections. Assuming that these input data are known, and correspond to the actual data in the real system (i.e. measurements of input flows every five minutes) these will be the data used to define the inputs to the simulation model. Applying the validation criteria established in the above sections to the I-35W motorway, the final calibration process leads to a model for which the validation process gives the following results for some selected detectors.

11.5.3.1 Detector 429

The graphical comparison between the Observed and the Simulated flows measured every five minutes over the simulation horizon of six hours is depicted in Figure 11-9.

Figure 11-9: Observed and Simulated Flows. Detector 429



11.5.3.2 Paired T-Test and Confidence Interval for detector 429

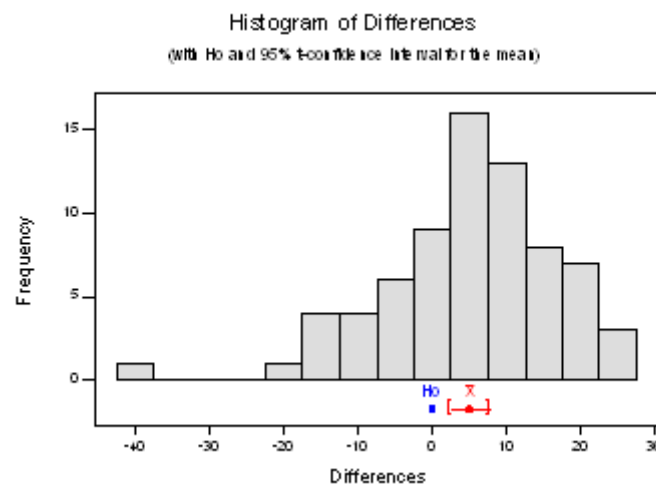
According with the procedures described in Section 11.4 this is the first statistical test. The results of the test are summarised in the table below for the Paired T for Observed – Simulated flows at detector 429

	N	Mean	StDev	SE Mean
Observed	72	159.40	49.20	5.80
Simulated	72	154.31	46.11	5.43
Difference	72	5.10	11.50	1.36

95% Confidence interval for mean difference is (2.39, 7.80), and the T-Test of mean difference = 0 (vs not = 0): gives a T-Value = 3.76 and a P-Value = 0.000.

The extremely small P-Value means that in this case the null hypothesis should be rejected and therefore conclude that both series of observations, measured and simulated are significantly different. This conclusion is also corroborated by the analysis of the histogram of differences in Figure 11-10.

Figure 11-10: Histogram of Differences. Detector 429



11.5.3.3 Two Sample T-Test and Confidence Interval for detector 429

Taking into account Kleijnen comments on this strict test, and looking how close both series appear in the graphics, one can wonder whether a less strict test would lead to the same conclusion. The table below summarises the results for the less demanding test “Two sample T for Observed versus Simulated”

	N	Mean	StDev	SE Mean
Observed	72	159.4	49.2	5.8
Simulated	72	154.3	46.1	5.4

95% Confidence interval for μ Observed - μ Simulate: (-10.6, 20.8), T-Test μ Observed = μ Simulate (vs not =): T = 0.64 P = 0.52 DF = 141. The P-Value tell us in this case that both series of measurements are comparable and the means of both samples could be considered similar.

11.5.3.4 Regression Analysis for detector 429

To corroborate these results the next methodological steps is the regression analysis of the observed versus the simulated measurements. The regression equation is:

$$y = -0.79 + 1.04 x$$

Analysis of Variance

Predictor	Coef	StDev	T	P	Source	DF	SS	MS	F	P
Constant	-0.787	4.742	-0.17	0.869	Regression	1	162705	162705	1241.64	0.000
x	1.03813	0.02946	35.24	0.000	Residual Error	70	9173	131		
					Total	71	171877			

S = 11.45 R-Sq = 94.7% R-Sq(adj) = 94.6%

The variance analysis show that the constant is no significant, but the both series are positively significantly correlated as the practically zero P-Value shows, and the high correlation coefficient. The additional analysis of residuals, which graphic is shown in Figure 11-11 reveals an almost linear behaviour as expected, and the presence of one anomalous observation, confirmed in the plot of the regression fits and its 95% prediction Interval depicted in Figure 11-12. A more complete analysis would require the identification of this outlayer and the search for an explanation.

Figure 11-11: Residual Model Diagnostics (Detector 429)

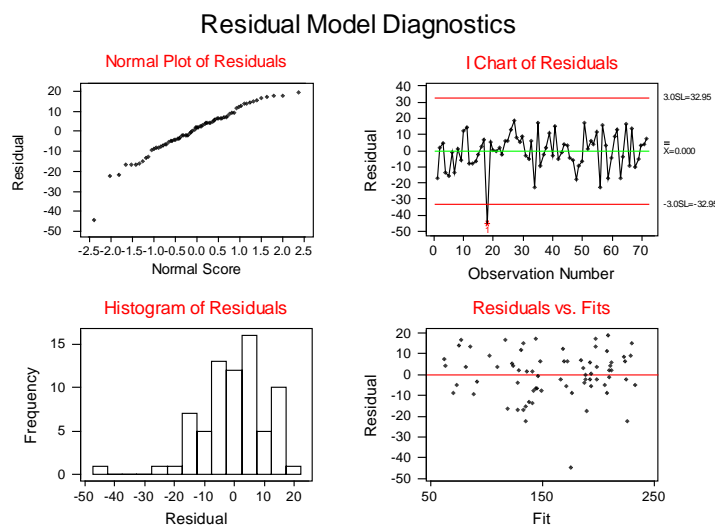
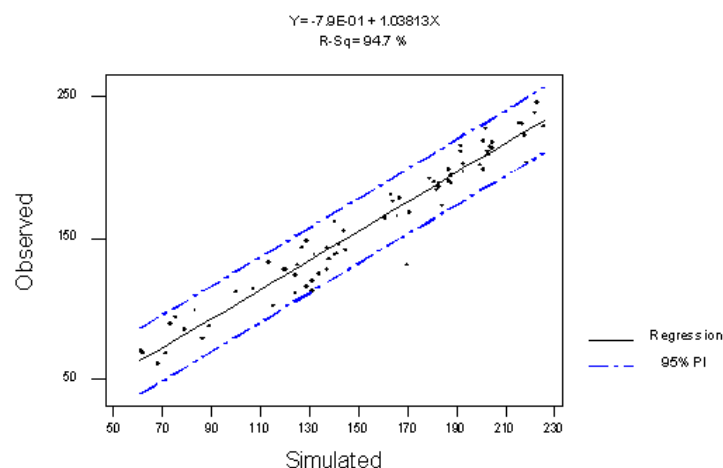


Figure 11-12: Observed-Simulated Regression Analysis (Detector 429)



11.5.3.5 How close the observed and simulated series are?

The visual inspection of both series reveals a high degree of agreement that has been partially confirmed by the above statistical tests. If all former tests would have given negative results then the simulated series should be discarded and a new simulation experiment should be conducted with new values for model parameters. But once we get to the conclusion that we could not reject the simulated series the key question,

as discussed in the methodological section is: are both series close enough? This should be corroborated by the Theil's coefficients:

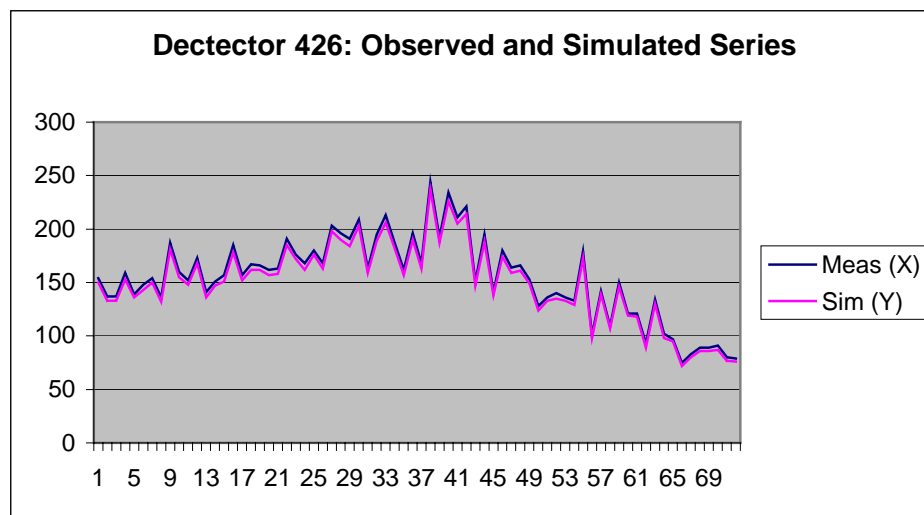
$$\begin{aligned}U &= 0.038169 \\UM &= 0.166091 \\US &= 0.061009 \\UC &= 0.784646\end{aligned}$$

The fact that the global coefficient U , and the variance portion coefficient US are significantly lower than 0.2 reveal a high degree of similarity between both series and a high ability of the simulated series for replicating the observed one. The value of UM , relatively close to 0.2 is undesirable but not enough to reject the simulated series. This could be related to the identified presence of outliers, and the fact that β_0 in the regression differs significantly from zero. We can conclude accepting the simulated series as a significant replication of the observed one.

11.5.3.6 Detector 426

This is an interesting case. The results for the plot of the observed and simulated series are shown in Figure 11-13. Visual inspection reveals a very good agreement between both series.

Figure 11-13: Observed and Simulated Flows. Detector 426



The statistical test "Paired T for Observed – Simulated" led to the conclusion that both series are almost identical (P-Value practically zero)

	N	Mean	StDev	SE Mean
Observed	72	154.32	38.50	4.54
Simulated	72	149.71	37.37	4.40
Difference	72	4.611	1.369	0.161

95% CI for mean difference: (4.289, 4.933) T-Test of mean difference = 0 (vs not = 0): T-Value = 28.58 P-Value = 0.000.

So we do not proceed to the relaxed two samples test. The regression equation is:

$$y = 0.123 + 1.03 x$$

Analysis of Variance

Predictor	Coef	StDev	T	P	Source	DF	SS	MS	F	P
Constant	0.1235	0.3885	0.32	0.752	Regression	1	105174	105174	167215.87	0.000
x	1.02998	0.00252	408.92	0.000	Residual Error	70	44	1		
					Total	71	105218			

S = 0.7931 R-Sq = 100.0% R-Sq (adj) = 100.0%

As before the regression is significant but not the constant. The regression plots and residuals analysis do not reveal the presence of outliers (Figures 11-14 and 11-15), but the analysis of residuals versus fits in Figure 11-15 reveals a very peculiar pattern.

Figure 11-14: Observed-Simulated Regression Analysis (Detector 426)

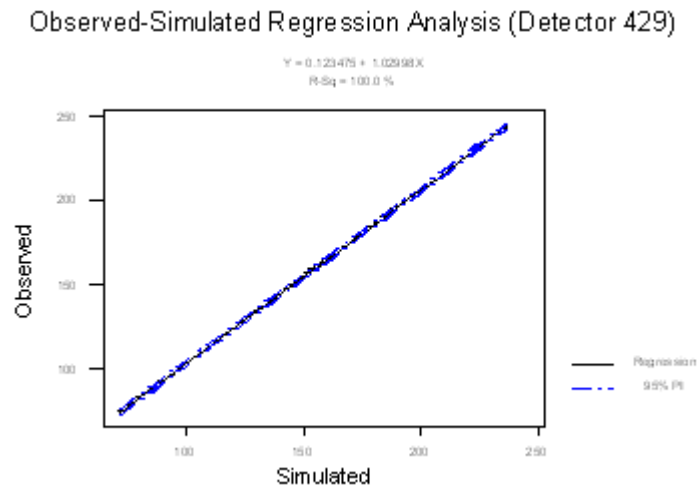
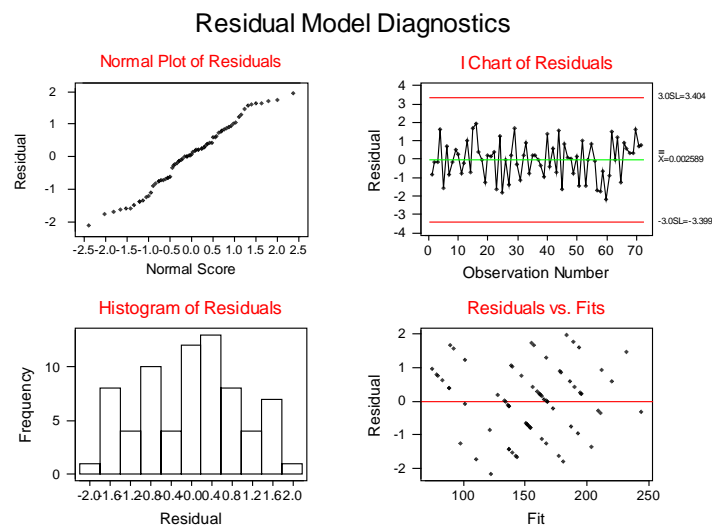


Figure 11-15: Residual Model Diagnostics (Detector 426)



The analysis of the Theil's coefficients corroborates the quality of the simulated (very low values of U and US), but the presence of a very high value of UM reveals the presence of a systematic bias, already identified by the fit of residuals. It can be identified that there is an almost constant difference of four units between the observed and the simulated series. That is, the simulated series is shifted 4 units with respect to the observed one.

U=	0.015348
UM=	0.920005
US=	0.055073
UC=	0.000362

We can also conclude accepting the simulated series in this case.

11.6 TUNNING VEHICLE MODELLING PARAMETERS IN AIMSUN

For calibration purposes, we can classify the AIMSUN vehicle modelling parameters according to their influence in the simulation outputs, into three groups:

- Global parameters
- Local Section Parameters
- Particular Vehicle Type Parameters

11.6.1 Influence of Global Parameters

Global parameters influence to all vehicles, regardless of the type, while driving anywhere in the network. Some examples of parameter influences are the following:

- Reaction Time influences:
 - Section Capacity (lower Reaction Time => higher section capacities)
 - On-Ramp Capacity (lower Reaction Time => higher On-ramp capacities)
- Reaction Time at Stop influences:
 - Stop and Go Capacity (lower Reaction Time at Stop => higher capacity).
 - Queue measures (higher Reaction Time at Stop => increases queue length and stop time)
- Queue Up and Leaving Speed influences:
 - Behaviour at Yellow Boxes
 - Queue Statistics
- Two Lane Car-Following (Number Veh., Distance, Max. Speed Differences) influences:
 - Smoothing Traffic
 - Merging situations
- Lane Changing parameters (% Overtake, % Recover) influences:
 - Distribution among of Lanes
 - Interurban situations
- Lane Changing: On-Ramp model
 - v3.2: less on-ramp capacity
 - v3.3: increase on-ramp capacity

11.6.2 Influence of Section Parameters

Section parameters influence all vehicles, regardless of type, when driving in a particular section of the network. Some examples of parameter influences include the following:

- Speed Limit influences:
 - Average Speed (higher speed limit => higher average section speed)
 - Travel Times: (higher speed limit => lower average section travel time)
- Turning Speed influences:
 - Turning Capacity (higher Turning speed => higher capacity)
 - Travel Times (higher Turning speed => lower travel times)
 - Average Speed (higher Turning speed => higher average speed)
- Visibility Distance influences:
 - Yield (Give Way) Sign Behaviour
- Distance Lane Changing Zones influences:
 - Turning proportions
 - Blocking situations

- Distance On-Ramp influences:
 - On-ramp Capacity
 - Use of lane as slow lane

11.6.3 Influence of Vehicle Parameters

Vehicle parameters influence all vehicles of a particular type when driving anywhere in the network. Some examples of parameter influences include the following:

- Maximum desired speed, Maximum acceleration, Normal and Maximum deceleration, Speed acceptance influence:
 - Speed, travel time, queue discharge, lane changing, etc
- Minimum distance between vehicles influences:
 - Capacity (smaller distances => higher capacity)
 - Queue lengths (smaller distances => lower queue lengths)
- Give-way time influences:
 - Yield and On-Ramp capacity
 - Lane Changing blockages
- Guidance acceptance influences:
 - Use of new routes

11.6.4 On-ramp Calibration

The parameters that are more relevant in the calibration of the on-ramps are the following:

- Reaction Time directly influences the on-ramp capacity. A lower Reaction Time gives a higher on-ramp capacity.
- 2-Lanes Car-Following parameters (Number of Vehicles, Distance, Maximum Speed Differences) influence the merging situations as it helps to make traffic smoother. The more equalised speeds among lanes provide more easiness to merge.
- Lane changing parameters related to overtaking manoeuvres (% Overtake, % Recover) influence distribution of vehicles among lanes in Interurban situations. Increasing the use of the speed lane will provide more opportunities to merge to vehicles entering from an on-ramp into the slow lane.
- Lane Changing: On-Ramp model version v3.2 results in a lower on-ramp capacity than v3.3 or higher.
- Distance On-Ramp influences the On-ramp Capacity as it determines the use of lane as slow lane or as on-ramp properly. The greater Distance On-Ramp provides a higher on-ramp capacity.
- Give-way time is the time a vehicle will accept to wait once it has reached a full stop, therefore it influences the on-ramp capacity in congested situations.
- Maximum acceleration influences the queue discharge and ability of lane changing. The higher acceleration rate will result in a higher on-ramp capacity as vehicles will accept smaller gaps.

11.7 AIMSUN VALIDATION TOOL

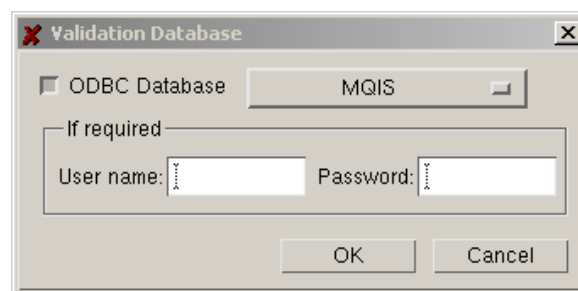
The Validation Tool can be found in the Validation command of the menu bar of the AIMSUN graphical interface. The Validation menu has two commands: Database and Charts.

11.7.1 Comparing two sets of data

The validation tool allows you to calculate the average of a set of replications and to compare statistical data and detector data gathered during previous replications in the form of time series plots and statistical variables. The replications can be as a result of one simulation, as a result of average calculation from a set of replications, or could be interpreted as the real data. The only requirement is all replications that the user wants to compare have to be stored in the same Data Base.

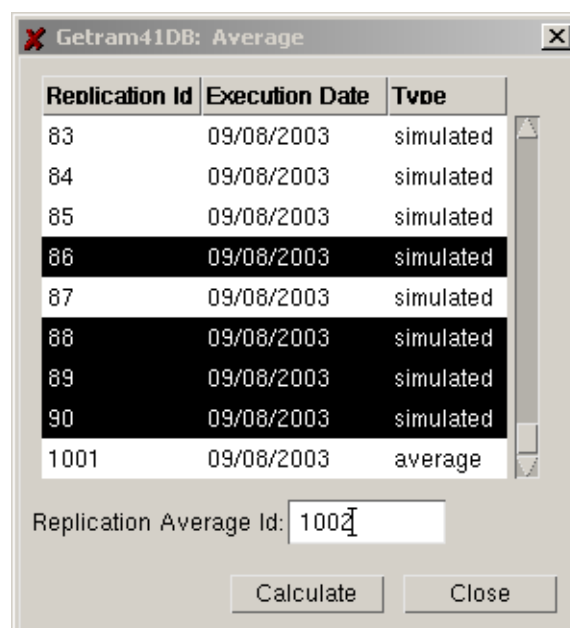
The first step is to select the database accessing to 'Validation/Database...' menu. The Validation Database selection dialog shown in figure 11-16 will appear.

Figure 11-16: Validation Database dialog



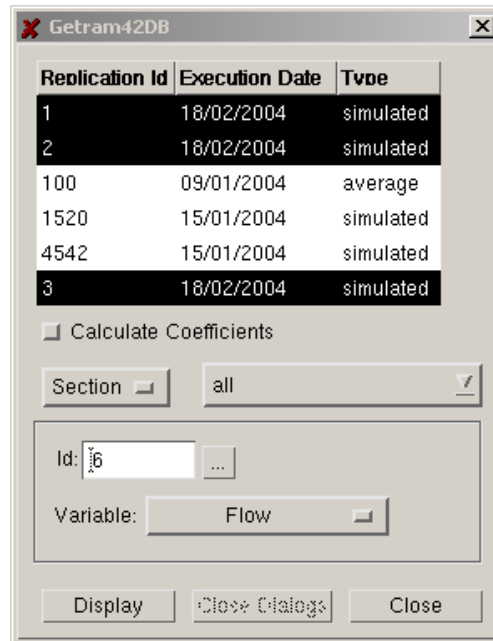
Once the database has been selected, the user can calculate the average of a set of replication and compare statistical detector data. To calculate the average of a set of replications, select the 'Averages' menu, where a list of all replications stored in the selected database appear. The user defines the replication average identifier writing an identifier in the 'Replication Identifier Average' field and then select or deselect the set of replications clicking on. Once all is defined, the user press 'Calculate' button and AIMSUN creates a new replication with the average of all selected replications. Figure 11-17 shows an example where replication 1002 will contain the average of replications 86, 88, 89 and 90. If the Replication average Identifier already exists, then AIMSUN gives a warning message that if the users ignores the data of the replication will be overwritten.

Figure 11-17: Average Calculation dialog



To compare statistical detector data, select the 'Charts' menu, where a list of all replications stored in the selected database. For each replication shows its identifier, the execution date and its type (could be simulated, average or real). The user may select a set of replications whose time plots will be displayed. After that, the user decides if the time plots are related to statistical data of sections or detectors and then selects the identifier of the element, either section or detector, clicking directly on the element in the network display or using the browser button or directly writing the identifier. If the database contains the statistical data per vehicle type then the user may select either 'all' to aggregate all vehicle types or one vehicle type.

Figure 11-18: Validation Tool



The variable to be plotted can be selected using the 'Variable' option menu. Then click on the Display button and the time plot is drawn. Each Time Plot dialog window has its own 'Variable' option menu; thus the selected variable can be changed at any time.

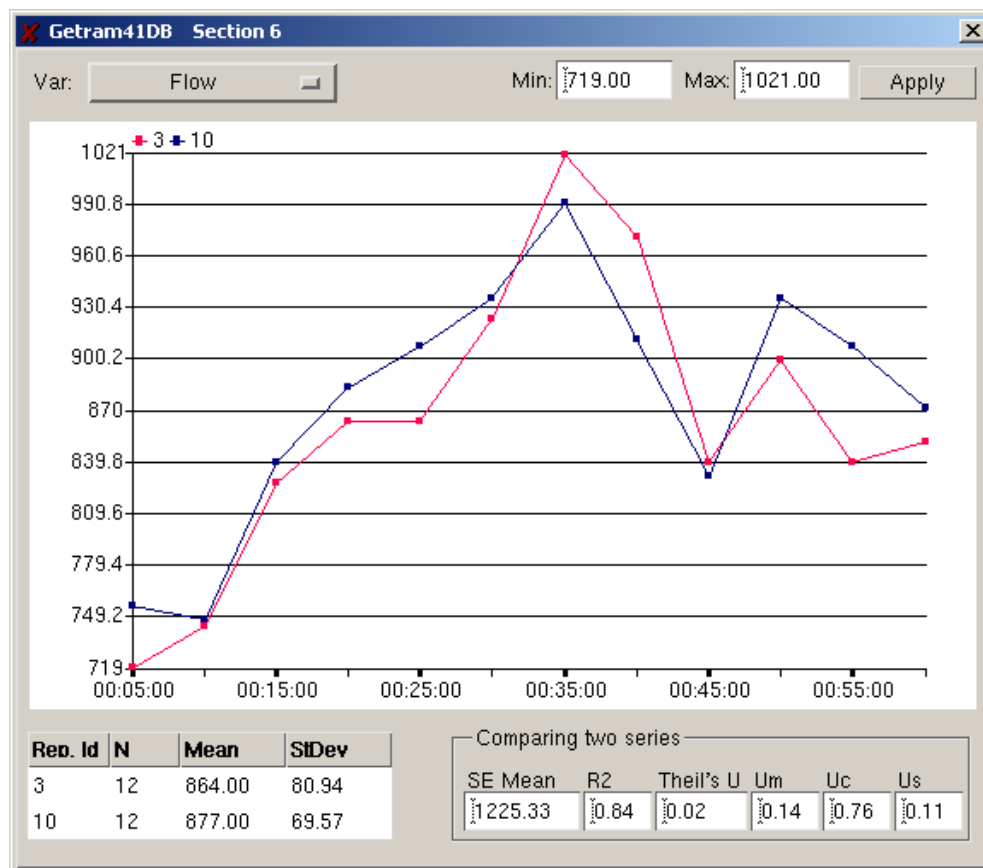
When the user selects 'Calculate coefficients' toggle button, for each replication shows the size of the replication (N), the mean and the standard deviation. And in the case to have only two replications it shows

the Mean Square Error($SEMean = \frac{1}{n} \sum_{i=1}^n (xi - yi)^2$), the coefficient of determination (ρ^2) ($\rho = Sxy / SxSy$)

where $Sxy = \frac{1}{n} \sum_{i=1}^n (xi - \bar{x})(yi - \bar{y})$ and $Sx = \sqrt{\frac{1}{n} \sum_{i=1}^n (xi - \bar{x})^2}$) and the Theil's coefficient U, Um, Uc and Us :

$$U = \frac{\sqrt{\frac{1}{n} \sum_{i=1}^n (xi - yi)^2}}{\sqrt{\frac{1}{n} \sum_{i=1}^n yi^2} + \sqrt{\frac{1}{n} \sum_{i=1}^n xi^2}}, 0 \leq U \leq 1$$

$$Um = \frac{(\bar{x} - \bar{y})^2}{SEm} \quad Us = \frac{(Sx - Sy)^2}{SEm} \quad Uc = \frac{2(1 - \rho)SxSy}{SEm} \quad Um + Us + Uc = 1$$

Figure 11-19: Comparing two sets of data

11.7.2 How to add Real Data in Output Data Base

To add the real data and then use the validation tool to compare with simulated data it is necessary to add the real data as a new replication in the Data Base, so the format of the real data has to match with the tables that AIMSUN generates. The steps are:

1. Add a new register or row in 'Replicat' table with at least the following fields

Attribute Name	Value
rid	Replication identifier
type	3 (Real Data)
iniTimeSimul	Initial Simulation Time (seconds)
endTimeSimul	End Simulation Time (seconds)
executionDate	Date of the execution of the simulation
periodSta	Statistics Interval (seconds)
periodDetec	Detection Interval (seconds)

Have the same iniTimeSimul, endTimeSimul, periodSta and periodDetec is the requirement to compare two replications.

2. In then case to have real section data, add all data in 'SectSta' table with the followin format:

Attribute Name	Description
rid	Replication identifier
sect_id	Section identifier

tfrom	Initial time of the period
tto	Final time of the period
ctype	Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
flow	Mean flow (veh/h)
density	Density (veh/Km or veh/Mile)
spd1	Mean speed (Km/h or Mph)
spd2	Speed deviation (Km/h or Mph)
hspd1	Harmonic mean speed (Km/h or Mph)
hspd2	Harmonic speed deviation (Km/h or Mph)
ttime1	Mean Travel Time (seconds)
ttime2	Travel Time deviation (seconds)
dtime1	Mean Delay Time (seconds)
dtime2	Delay Time deviation (seconds)
stime1	Mean Stop Time (seconds)
stime2	Stop Time deviation (seconds)
nstops	Number of stops per vehicle
qmean	Mean Queue Length
qmax	Maximum Queue Length
travel	Total number of km or miles travelled in the section1
fuelc	Total litres of fuel consumed in the section2

3. In the case to have real detection data, add all data in 'DetectMea' table with the following format:

Attribute Name	Description
rid	Replication identifier
iddet	Detector Identifier
namedet	Detector Name
tfrom	Initial time of the period
tto	Final time of the period
ctype	Identifier of vehicle type. = 0, aggregation of all vehicle type. >0, a specific vehicle type.
countveh	Number of vehicles
speed	Average Speed (Km/h or Mph)
occupancy	Percentage of occupancy
density	Density (Veh/Km or Veh/Mile)
headway	Average Headway between vehicles (sec)

APPENDIX 1: DESCRIPTION OF PARAMETER'S FILES

A simulation experiment is characterised by a set of parameters that are grouped into six categories: Run Time, Modelling, Statistics, Detection, Replications and Route Choice. When loading a network, all six categories of parameters are loaded. These parameters are stored in ASCII files, either at the network level or at a global level. The location, name and format of these files are the following:

RUN TIME

File Path

..network path../AIMSUN/RunTime

Contents and Format

Line	Description	Format	Type
1	simulated date	maximum length 11 characters	character
2	simulated initial time	hh:mm:ss	character
3	warm-up period	hh:mm:ss	character
4	simulated end time	hh:mm:ss	character
5	redrawing frequency		integer

MODELLING

File Path

..network path../AIMSUN/ModelPar

Contents and Format

Line	Description	Type	Value / Range
1	current version of the file	character	"V3.3"
2	simulated step (reaction time) in seconds	float	$0.5 \leq \dots \leq 1$
3	reaction time at stop in seconds	float	\geq simulation step
4	queuing up speed (m/s)	float	≥ 0
5	queuing leaving speed (m/s)	float	≥ 0
6	apply 2 lanes car following model	integer	0 or 1
7	number of vehicles	integer	> 0
8	maximum distance (m)	float	> 0
9	maximum speed difference (Km/h)	float	> 0
10	maximum speed difference on ramp (Km/h)	float	> 0
11	percent overtake	float	$0 < \dots \leq 1$
12	percent recover	float	$0 < \dots \leq 1$
13	on ramp model	integer	32 or 33

STATISTICS

In this case we have three files. The first one describes general statistics, the second one describes O/D Matrix statistics and the third one describes public transport lines statistics

File Path

..network path../AIMSUN/DefEstad
 ..network path../AIMSUN/DefODEstad
 ..network path../AIMSUN/DefPTEstad

Contents and Format of 'DefEstad'

Line	Description	Type	Value
1	gather statistics	integer	0 or 1
2	a comment that describes next parameter	character	"#TEMPS"
3	the interval period to calculate statistics (hh:mm:ss)	character	
4	the following integers must be introduced depending on your preferences:		
	Report printout. Setting this value report statistics will not be printed, otherwise introduce some of the next values that identifies the kind of statistics to be printed.	integer	0
	prints system statistics	integer	1
	prints sections statistics	integer	2
	prints sections and turnings statistics	integer	3
	prints routes statistics	integer	4
	statistical data flush	integer	5
	prints TRANSYT statistics	integer	6
5	a comment that describes next parameter, this value is expected if you have set print sections/turnings statistics	character	"#ARCS"
6	the following integers must be introduced depending on your preferences of section statistics		
	prints statistics of all sections	integer	0
	prints statistics only of all entrance sections	integer	1
	prints statistics only of all exit sections	integer	2
	prints statistics only of selected sections	integer	3
7	The number of selected sections if the previous option has been set to. The format is: "* %d"	integer	>0
8	The identifiers of the sections selected, separated by a white space.	integer	
9	a comment that describes next parameter, this value is expected if you have set print routes statistics	character	"#RUTES"
i	The identifiers of desired routes. Each identifier (the name of the route) in a new line.	character	
n	a comment that describes next parameter	character	"#ALLMODS"
n + 1	prints statistics distinguishing by vehicle type if set 1	integer	0 or 1
n + 2	a comment that describes next parameter	character	"#DEVIATIONS"
n+3	prints standard deviations if set 1	integer	0 or 1

An example of ‘DefEstad’ file

```

1
#TEMPS
00:02:00
1 3 4 6
#ARCS
3
* 4
330 331 396 399
#RUTES
stream1
#ALLMODS
1
#DEVIATIONS
1

```

Contents and Format of ‘DefODEstad’

Line	Description	Type	Value
1	prints OD matrix statistics	integer	0 or 1
1	prints statistics distinguishing by vehicle type	integer	2
1	prints standard deviations	integer	3
1	statistical data flush	integer	4
2	a comment that describes next parameter	character	“#ORIGIN”
3	prints origin statistics: not print statistics by each origin centroid “ ,distinguishing by to all centroids	integer	0 1 2
4	a comment that describes next parameter	character	“#DESTINATION”
5	prints destination statistics: not print statistics by each destination centroid “ ,distinguished by to all centroids	integer	0 1 2
6	a comment that describes next parameter	character	“#PAIRS”
7	prints statistics by pairs of centroids	integer	0 or 1
8	number of pairs of centroids	integer	> 0
i	for each pair introduce in a new line the identifiers of the centroids that define the pair: identifier 1 identifier 2	integer	> 0

Example of ‘DefODEstad’ file

```

123
#ORIGIN
0
#DESTINATION
0
#PAIRS
1
3
14 11
10 9
19 16

```

Contents and Format of ‘DefPTEstad’

Line	Description	Type	Value
1	prints public transport lines statistics	integer	0 or 1
1	prints statistics distinguishing by vehicle type	integer	2
1	prints standard deviations	integer	3
1	statistical data flush	integer	4
2	a comment that describes next parameter, this line is expected if print pt lines statistics is set to 1	character	“#PTLines”
3	For each pt line that you desire printed statistics introduce its identifier, the identifiers are separated by a white space.	integer	>0

Example of ‘DefPTEstad’ file

```
123
#PTLines
1 25
```

DETECTION**File Path**

..network path../AIMSUN/DefDetec

Contents and Format

Line	Description	Type	Format	Value
1	Aggregated detection time. If none write “00:00:00”	character	hh:mm:ss	
2	print out	integer		0 or 1
3	statistical data flush	integer		0 or 1

REPLICATIONS**File Path**

..network path../AIMSUN/Seed

Contents and Format

Line	Description	Type	Value
1	random seed	integer	> 0
2	replication id	integer	usually 1
3	If multiple replications, number of replications	integer	>= 0
i	For each replication a new line must be introduced. Each of this lines must have 2 integers: seed integer identifier replication integer	integer	> 0

ROUTE CHOICE

File Path

..network path../AIMSUN/RouteBasedPar

Contents and Format of 'RouteBasedPar'

Line	Description	Type	Value
1	current version of the file	character	"V4.0"
2	cycle, in seconds	float	≥ 0
3	number of intervals	integer	> 0
4	capacity	float	
5	Route based type. The number '1' before the type means that it is dynamic. Only one of the following values can be introduced:	integer	
	fixed distance		0
	fixed time		1
	binomial		2 or 12
	logit		3 or 13
	C_logit		4 or 14
	user defined		5 or 15
	If route based type > 1 next data lines are expected		
6	Maximum number of routes.	integer	> 0
7	If type is binomial, the probability " logit or C_logit, the scale " user defined, the function name	float float character	$0 < \dots \leq 1$ ≥ 0
8	If type is logit or C_logit, the beta parameter	float	≥ 0
9	If type is logit or C_logit, the gamma parameter	float	≥ 0

Example of 'RouteBasedPar' file

```
V4.0
120.000000
1
2.000000
2
2
0.75
```

OUTPUT LOCATION

File Path

..traffic result path../AIMSUN/Output
or
..od matrix path../AIMSUN/Output

Contents and Format of 'Output'

Line	Description	Type	Value
1	Type : ASCII Access Data Base (ODBC)	integer	0 1 2
2	If type ASCII, the path of the directory where to place the output. If type Access, The path of the database If type ODBC The name of the connection White spaces in the path must be represented with the character '#'. .	character	

Example of 'Output' file

0
Z:\olga\PTNet40\hnormal\Aimsun

APPENDIX 2: OUTPUT DATABASE DEFINITION

The Tables defined in the AIMSUN Results Database are Networks, Replicat, RepScen, RepState, RepExt, RepCtrl, RepExp, SysSta, SectSta, TurnSta, CentSta, ODPSta, StrmSta, PTSta, SysPoll, SectPoll, TurnPoll, CentPoll, ODPPoll, StrmPoll, PTPoll, DetecMea, DetEquipVeh, vehTypes, streamsDef, PathDef, PathSect, PathSta and PastCost. Each table has a candidate primary key, although these primary keys are not defined in the database.

Networks

This contains information about the definition of the network.

Attributes:

Attribute Name	Type	Size	Description
id	integer		Network identifier
path	char	254	Path where is located the getram model
idOffset	integer		Reserved attribute for internal use

Candidate Primary Key: id

Replicat : It contains information about every replication simulated.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
done	integer		0 = Not simulated; 1= Simulated
seed	integer		Replication seed used
type	integer		1: Simulated; 2: Average; 3:Real Data
simulatedDate	char	10	Date of the Simulation
iniTimeSimul	float		Initial Simulation Time (seconds)
endTimeSimul	float		End Simulation Time (seconds)
warmUp	float		Duration of the war-up period (seconds)
executionDate	char	10	Date of the execution of the simulation
executionTime	char	8	Time of the execution of the simulation
periodSta	float		Statistics Interval (seconds)
periodDetec	float		Detection Interval (seconds)
metricUnits	integer		1: Metric Units; 0: English Units

Candidate Primary Key: rid

RepScen : It contains information about the scenario of every replication simulated.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
network	char	254	Network Name
trafficRes	char	254	Traffic Result Name
initialState	integer		0: Initial State not Loaded; 1: Loaded
odMatrix	char	254	OD Matrix Name
ptPlan	char	254	Public Transport Name
initialMessages	integer		0: Initial Messages not Loaded; 1: Loaded

initialIncidents	integer		0: Initial Incidents not Loaded; 1: Loaded
arrivalType	integer		0:Exponential; 1:Uniform; 2:Normal; 3:Constant; 4:External; 5:ASAP
pollutants	integer		0: Pollutant Model Disabled; 1:Enabled
fuelConsum	integer		0: Fuel Consumption Model Disabled; 1:Enabled

Candidate Primary Key: rid

RepState : It contains information about every state loaded of a traffic result simulated in a replication.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
statePos	integer		Position of the state from 0 to N
trafficState	char	33	Traffic State Name

Candidate Primary Key: rid

RepCtrl : It contains information about every control plan loaded in a replication.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
controlPos	integer		Position of the control Plan from 0 to N
controlName	char	33	Control Plan Name
atTime	float		Initial Time of the Control Plan (seconds)

Candidate Primary Key: rid

RepExt : It contains information about every getram extensions loaded in a replication.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
extensionPos	integer		Position of the extension from 0 to N
extensionName	char	254	Name and Path of the getram extension

Candidate Primary Key: rid

RepExp: It contains information about every Experiment parameter used in a replication.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
simulStep	float		Simulation Step
reactionTimeAtStop	float		Reaction Time at Stop parameter

queueingUpSpeed	float		Queuing Up Speed parameter
queueingLeavingSpeed	float		Queuing Leaving Speed parameter
apply2LanesCarFollowing	integer		0:two lanes car following not applied; 1:applied
nbVehs2LanesCF	integer		Number of Vehicles of the two lanes car following model
maxDistance2LanesCF	float		Maximum Distance of the two lanes car following model
maxSpeedDiff2LanesCF	float		Maximum Speed Difference of the two lanes car following model
maxSpeedDiffOnRamp2LCF	float		Maximum Speed Difference On Ramp of the two lanes car following model
percentOvertake	float		Percentage of Overtake
percentRecover	float		Percentage of Recover
OnRampv33	integer		0: OnRamp model v3.3 not applied; 1:applied
genericTypeRB	integer		0:Fixed Distance; 1:Fixed Time; 2:Binomial; 3:Proportional 4:Logit; 5:C-Logit; 6:User Defined
cycleRB	float		RouteChoice Cycle
dynamic	integer		0: Dynamic Route Choice not applied; 1: applied
nbintervalsRB	float		Number of Intervals considered
weightCapacity	float		Capacity Weight parameter
scaleFactor	float		Scale Factor
pbinomialRB	float		“success” probability of Binomial Dist.
initialPathTreesRB	integer		The initial path trees calculated at the beginning of the simulation
maxPathTreesRB	integer		The maximum number of paths to be kept in the memory
maxAlterRoutesRB	integer		The maximum number of alternative paths used in the Route Choice model
betaFactor	float		Beta Factor of C-Logit model
gammaFactor	float		Gamma Factor of C-Logit model
userDefinedRCName	float		Name of the User Defined Route Choice Function
costPerVehType	integer		1: Arc Costs distinguished per Vehicle Type; 0: No

Candidate Primary Key: rid

SysSta: This contains statistical information of the whole system for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
flow	integer		Mean flow (veh/h)

density	float		Density (veh/Km or veh/Mile)
spd1	float		Mean speed (Km/h or Mph)
spd2	float		Speed deviation (Km/h or Mph)
hspd1	float		Harmonic mean speed (Km/h or Mph)
hspd2	float		Harmonic Speed deviation (Km/h or Mph)
ttime1	integer		Mean Travel Time (sec/Km or sec/Mile)
ttime2	integer		Travel Time deviation (sec/Km or sec/Mile)
dtime1	integer		Mean Delay Time (sec/Km or sec/Mile)
dtime2	integer		Delay Time deviation (sec/Km or sec/Mile)
stime1	integer		Mean Stop Time (sec/Km or sec/Mile)
stime2	integer		Stop Time deviation (sec/Km or sec/Mile)
nstops	float		Number of stops per vehicle (#/Km or #/Mile)
travel	float		Total number of km or miles travelled
travelttime	float		Total travel time experimented
fuelc	float		Total litres of fuel consumed ¹

Candidate Primary Key: rid, tfrom, tto, ctype

SectSta : This contains statistical information of the sections for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
sect_id	integer		Section identifier
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
flow	integer		Mean flow (veh/h)
density	float		Density (veh/Km or veh/Mile)
spd1	float		Mean speed (Km/h or Mph)
spd2	float		Speed deviation (Km/h or Mph)
hspd1	float		Harmonic mean speed (Km/h or Mph)
hspd2	float		Harmonic speed deviation (Km/h or Mph)
ttime1	integer		Mean Travel Time (seconds)
ttime2	integer		Travel Time deviation (seconds)
dtime1	integer		Mean Delay Time (seconds)
dtime2	integer		Delay Time deviation (seconds)
stime1	integer		Mean Stop Time (seconds)
stime2	integer		Stop Time deviation (seconds)
nstops	float		Number of stops per vehicle
qmean	float		Mean Queue Length
qmax	float		Maximum Queue Length
travel	float		Total number of km or miles travelled in the section
travelttime	float		Total travel time experimented in the section
fuelc	float		Total litres of fuel consumed in the section ¹

¹ This is only provided when the particular model 'Fuel Consumption' is set to 'ON'

Candidate Primary Key: sect_id, rid, tfrom, tto, ctype

TurnSta: This contains statistical information of the turnings for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
fromsect	integer		Identifier of Origin Section of the turning
tosect	integer		Identifier of Destination Section of the turning
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
flow	integer		Mean flow (veh/h)
spd1	float		Mean speed (Km/h or Mph)
spd2	float		Speed deviation (Km/h or Mph)
hspd1	float		Harmonic mean speed (Km/h or Mph)
hspd2	float		Harmonic speed deviation (Km/h or Mph)
ttime1	integer		Mean Travel Time (seconds)
ttime2	integer		Travel Time deviation (seconds)
dtime1	integer		Mean Delay Time (seconds)
dtime2	integer		Delay Time deviation (seconds)
stime1	integer		Mean Stop Time (seconds)
stime2	integer		Stop Time deviation (seconds)
nstops	float		Number of stops per vehicle
qmean	float		Mean Queue Length
qmax	float		Maximum Queue Length
travel	float		Total number of km or miles travelled in the turning
traveltime	float		Total travel time experimented in the turning
fuelc	float		Total litres of fuel consumed in the turning ¹

Candidate Primary Key: fromsect, tosect, rid, tfrom, tto, ctype

CentSta: This contains statistical information of the centroids for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
idcent	integer		Identifier of the centroid
centype	integer		Type of centroid (0: Origin, 1: Destination)
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
nbveh	integer		Number of Vehicles
spd1	float		Mean speed (Km/h or Mph)

¹ This is only provided when the particular model 'Fuel Consumption' is set to 'ON'

spd2	float		Speed deviation (Km/h or Mph)
hspd1	float		Harmonic mean speed (Km/h or Mph)
hspd2	float		Harmonic speed deviation (Km/h or Mph)
ttime1	integer		Mean Travel Time (seconds)
ttime2	integer		Travel Time deviation (seconds)
dtime1	integer		Mean Delay Time (seconds)
dtime2	integer		Delay Time deviation (seconds)
stime1	integer		Mean Stop Time (seconds)
stime2	integer		Stop Time deviation (seconds)
nstops	float		Number of stops per vehicle
vlost	integer		Number of vehicles lost
travel	float		Total number of km or miles travelled
traveltime	float		Total travel time experimented
fuelc	float		Total litres of fuel consumed ¹

Candidate Primary Key: idcent, rid, tfrom, tto, ctype

ODPSta: This contains statistical information of the O/D pairs for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
origin	integer		Identifier of the origin centroid
dest	integer		Identifier of the destination centroid
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
nbveh	integer		Number of Vehicles
spd1	float		Mean speed (Km/h or Mph)
spd2	float		Speed deviation (Km/h or Mph)
hspd1	float		Harmonic mean speed (Km/h or Mph)
hspd2	float		Harmonic speed deviation (Km/h or Mph)
ttime1	integer		Mean Travel Time (seconds)
ttime2	integer		Travel Time deviation (seconds)
dtime1	integer		Mean Delay Time (seconds)
dtime2	integer		Delay Time deviation (seconds)
stime1	integer		Mean Stop Time (seconds)
stime2	integer		Stop Time deviation (seconds)
nstops	float		Number of stops per vehicle
vlost	integer		Number of vehicles lost
travel	float		Total number of km or miles travelled
traveltime	float		Total travel time experimented
fuelc	float		Total litres of fuel consumed in the turning ¹

Candidate Primary Key: origin, dest, rid, tfrom, tto, ctype

¹ This is only provided when the particular model 'Fuel Consumption' is set to 'ON'

StrmSta: This contains statistical information of the streams for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
name_stream	char	20	Identifier of the stream
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
flow	integer		Mean flow (veh/h)
density	float		Density (veh/Km or veh/Mile)
spd1	float		Mean speed (Km/h or Mph)
spd2	float		Speed deviation (Km/h or Mph)
hspd1	float		Harmonic mean speed (Km/h or Mph)
hspd2	float		Harmonic speed deviation (Km/h or Mph)
ttime1	integer		Mean Travel Time (seconds)
ttime2	integer		Travel Time deviation (seconds)
dtime1	integer		Mean Delay Time (seconds)
dtime2	integer		Delay Time deviation (seconds)
stime1	integer		Mean Stop Time (seconds)
stime2	integer		Stop Time deviation (seconds)
nstops	float		Number of stops per vehicle
qmean	float		Mean Queue Length
qmax	float		Maximum Queue Length
travel	float		Total number of km or miles travelled in the stream
traveltime	float		Total travel time experimented in the stream
fuelc	float		Total litres of fuel consumed in the turning ¹

Candidate Primary Key: name_stream, rid, tfrom, tto, ctype

PTSta : This contains statistical information of public transport lines for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
idline	integer		Identifier of the public transport line
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicles type.
nbveh	integer		Number of Vehicles
spd1	float		Mean speed (Km/h or Mph)
spd2	float		Speed deviation (Km/h or Mph)
hspd1	float		Harmonic mean speed (Km/h or Mph)
hspd2	float		Harmonic speed deviation (Km/h or Mph)

¹ This is only provided when the particular model 'Fuel Consumption' is set to 'ON'

ttime1	integer		Mean Travel Time (seconds)
ttime2	integer		Travel Time deviation (seconds)
dtime1	integer		Mean Delay Time (seconds)
dtime2	integer		Delay Time deviation (seconds)
stime1	integer		Mean Stop Time (seconds)
stime2	integer		Stop Time deviation (seconds)
nstops	float		Number of stops per vehicle
travel	float		Total number of km or miles travelled.
traveltime	float		Total travel time experimented.
fuelc	float		Total litres of fuel consumed ¹

Candidate Primary Key: rid, idline, tfrom, tto, ctype

SysPoll: This contains pollution statistical information of the whole system for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
npollutant	char	20	Pollutant name
vpollutant	float		Value of pollutant (Kgr)

Candidate Primary Key: rid, tfrom, tto, ctype, npollutant

SectPoll: This contains pollution statistical information of the sections for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
sectid	integer		Section Identifier
npollutant	char	20	Pollutant name
vpollutant	float		Value of pollutant (Kgr)

Candidate Primary Key: sectid, rid, tfrom, tto, ctype, npollutant

TurnPoll: This contains pollution statistical information of the turnings for each period.

Attributes:

¹ This is only provided when the particular model 'Fuel Consumption' is set to 'ON'

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types..
fromsect	integer		Identifier of Origin Section of the turning
tosect	integer		Identifier of Destination Section of the turning
npollutant	char	20	Pollutant name
vpollutant	float		Value of pollutant (Kgr)

Candidate Primary Key: fromsect, tosect, rid, tfrom, tto, ctype, npollutant

CentPoll: This contains pollution statistical information of the centroids for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
idcent	integer		Centroid Identifier
centype	integer		Type of centroid (0: Origin, 1: Destination)
npollutant	char	20	Pollutant name
vpollutant	float		Value of pollutant (Kgr)

Candidate Primary Key: idcent, rid, tfrom, tto, ctype, npollutant

ODPPoll: This contains pollution statistical information of the O/D pairs for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types..
origin	integer		Origin Centroid Identifier
dest	integer		Destination Centroid Identifier
npollutant	char	20	Pollutant name
vpollutant	float		Value of pollutant (Kgr)

Candidate Primary Key: origin, dest, rid, tfrom, tto, ctype, npollutant

StrmPoll: This contains pollution statistical information of the streams for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, specific vehicle types.
name_stream	char	20	Origin Centroid Identifier
npollutant	char	20	Pollutant name
vpollutant	float		Value of pollutant (Kgr)

Candidate Primary Key: name_stream, rid, tfrom, tto, ctype, npollutant

PTPoll: This contains pollution statistical information of the public transport lines for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
idline	integer		Identifier of the public transport line
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle types. >0, a specific vehicle types.
npollutant	char	33	Pollutant name
vpolutant	float		Value of pollutant (Kgr)

Candidate Primary Key: rid, idline, tfrom, tto, ctype, npollutant

DetecMea: It contains the detection measures for each period.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
iddet	integer		Detector Identifier
namedet	char	33	Detector Name
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle type. >0, a specific vehicle type.
countveh	integer		Number of vehicles
speed	float		Average Speed (Km/h or Mph)
occupancy	float		Percentage of occupancy
density	float		Density (Veh/Km or Veh/Mile)
headway	float		Average Headway between vehicles (sec)

Candidate Primary Key: rid, iddet, tfrom, tto, ctype

DetEquipVeh: It contains a log of all equipped vehicles detected.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
iddet	integer		Detector Identifier
namedet	char	33	Detector Name
timedet	float		Detection time of the Equipped Vehicle
idveh	integer		Equipped Vehicle Identifier
vehetype	integer		Vehicle Type of the Equipped Vehicle
idptline	integer		Public Transport Line of the Equipped Vehicle, if it is a public transport vehicle.

Candidate Primary Key: rid, iddet, tfrom, tto, ctype

vehTypes: It contains the definition of vehicle types.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle type. >0, a specific vehicle type.
name	char	33	Name of the vehicle type

streamsDef: It contains the definition of streams.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
name_stream	char	33	Name of the stream
sect_pos	integer		Position of the section
sect_id	integer	33	Identifier of the section in position sect_pos

PathDef: It contains the definition of simulated paths.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
id	integer		Path identifier
origin	integer		Origin Centroid identifier
dest	integer		Destination Centroid identifier
nbsect	integer		Total number of sections
created	float		Path Creation Time (seconds)
distance	float		Path Distance (meters or feet)

Candidate Primary Key: rid, id

PathSect: It contains all section identifiers that belong to each path.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
id	integer		Path identifier
sect_pos	integer		Position of the section
sect_id	integer		Identifier of the section in position sect_pos

Candidate Primary Key: rid, id, sect_pos

PathSta: It contains statistics of each path.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
id	integer		Path identifier
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle type. >0, a specific vehicle type.
nbvehass	integer		Number of Vehicles assigned
nbveh	integer		Number of Vehicles arrived to its detination that followed the path
ttime1	integer		Mean Travel Time of all vehicles arrived to its destination that followed the path(seconds)

Candidate Primary Key: rid, id, tfrom, tto, ctype

PastCost: It contains statistics of each path.

Attributes:

Attribute Name	Type	Size	Description
rid	integer		Replication identifier
fromsect	integer		Origin Section of the arc
tosect	integer		Destination Section of the arc
tfrom	integer		Initial time of the period
tto	integer		Final time of the period
ctype	integer		Identifier of vehicle type. = 0, aggregation of all vehicle type. >0, a specific vehicle type.
cost	float		link cost used to calculate the paths
outputcost	float		experimented link travel time

Candidate Primary Key: rid, fromsect, tosect, tfrom, tto, ctype

APPENDIX 3: CALIBRATION OF AIMSUN CAR-FOLLOWING MODEL

1. BACKGROUND

Most of the currently existing microscopic traffic simulators are based on the family of car-following, lane changing and gap acceptance models to model the vehicle's behaviour. The most commonly used current models include Helly's model [HEL61], implemented in SITRA-B+ [GAB91], Herman's model [HER59], or its improved version by Wicks [WIC77], implemented in MITSIM [QIY96], the psycho-physical model of Wiedemann [WIE74], used in VISSIM [FEL94], or the ad hoc version of Gipps [GIP81], used in AIMSUN [BAR95], [BAR98]. Other microscopic simulators such as INTEGRATION [VAN96] and PARAMICS employ heuristic or other modelling not publicly available in analytic form.

Helly's model uses the following expression for the acceleration of the follower car $\ddot{x}_{n+1}(t+T)$:

$$\ddot{x}_{n+1}(t+T) = c_1[\dot{x}_n(t) - \dot{x}_{n+1}(t)] + c_2[x_n(t) - x_{n+1}(t) - D]$$

where T is the reaction time for the vehicle-driver system, c_1 and c_2 are the relative velocity and headway control parameters and D the desired headway, typically expressed as: $D = l_n + \tau_{n+1}\dot{x}_{n+1}(t)$ with l_n being the length of the leader vehicle n and τ_{n+1} the time headway for the follower $n+1$.

Herman's model assumes an acceleration rate given by

$$\ddot{x}_{n+1}(t) = \alpha^\pm \frac{\dot{x}_{n+1}^{\beta^\pm}(t)}{g_{n+1}^{\gamma^\pm}(t)} (\dot{x}_n(t) - \dot{x}_{n+1}(t))$$

where α^\pm, β^\pm and γ^\pm are model parameters. $\alpha^+, \beta^+, \gamma^+$ are used for acceleration ($\dot{x}_{n+1}(t) \leq \dot{x}_n(t)$), and $\alpha^-, \beta^-, \gamma^-$ for deceleration ($\dot{x}_{n+1}(t) > \dot{x}_n(t)$), and $g_{n+1} = x_{n+1} - x_n - l_n$ represents the gap distance from the leading vehicle.

The Gipps model, as described in section 3.4, consists of two components: acceleration and deceleration. The first represents the intention of a vehicle to achieve certain desired speed, while the second reproduces the limitations imposed by the preceding vehicle when trying to drive at the desired speed. This model states that, the maximum speed at which a vehicle (n) can accelerate during a time period ($t, t+T$) is given by:

$$V_a(n, t+T) = V(n, t) + 2.5a(n)T \left(1 - \frac{V(n, t)}{V^*(n)} \right) \sqrt{0.025 + \frac{V(n, t)}{V^*(n)}}$$

where: $V(n, t)$ is the speed of vehicle n at time t ; $V^*(n)$ is the desired speed of the vehicle (n); $a(n)$ is the maximum acceleration for vehicle n ; T is the reaction time.

On the other hand, the maximum speed that the same vehicle (n) can reach during the same time interval ($t, t+T$), according to its own characteristics and the limitations imposed by the presence of the leader vehicle is:

$$V_b(n, t+T) = d(n)T + \sqrt{d(n)^2 T^2 - d(n) \left[2\{x(n-1, t) - s(n-1) - x(n, t)\} - V(n, t)T - \frac{V(n-1, t)^2}{d'(n-1)} \right]}$$

where: $d(n) (< 0)$ is the maximum deceleration desired by vehicle n ; $x(n, t)$ is position of vehicle n at time t ; $x(n-1, t)$ is position of preceding vehicle ($n-1$) at time t ; $s(n-1)$ is the effective length of vehicle ($n-1$); $d'(n-1)$ is an estimation of vehicle ($n-1$) desired deceleration. The final speed for vehicle n during time interval ($t, t+T$) is the minimum of those previously defined speeds:

$$V(n, t + T) = \min \{ V_a(n, t + T), V_b(n, t + T) \}$$

The position of vehicle n inside the current lane is updated by taking the speed into the movement equation:

$$x(n, t + T) = x(n, t) + V(n, t + T)T$$

A common drawback of most of these models is that the model parameters are global i.e. constant for the entire network whereas it is well known that driver's behaviour is affected by traffic conditions. Therefore a more realistic car-following modelling for microscopic simulation should account for local behaviour. That implies that some of the model parameters must be local depending on local geometric and traffic conditions.

2. MODELING IN AIMSUN

The AIMSUN car following model evolved after the seminal Gipps model, which was improved to meet the local requirements, described earlier. Three main aspects of the model have been enhanced based on the empirical evidence gathered calibrating the model for observed data:

- The way in which is calculated the vehicle speed $V^*(n)$ used in the Gipps model
- How vehicles in adjacent lanes influence vehicle's behaviour
- Accounting for grade effects in car-following

Speed calculation

The first improvement is related to the vehicle speed $V^*(n)$ used in equation (1). In AIMSUN implementation $V^*(n)$ is the desired speed of vehicle n for the current section. In car-following a leading vehicle, attempts to drive to its maximum desired speed. Three parameters are used to calculate the maximum speed of leading vehicle n while driving on a particular section or turning:

1. Maximum desired speed of n : $v_{\max}(n)$. This is a vehicle parameter.
2. Speed acceptance of n : $\theta(n)$ This is a vehicle parameter measuring the driver's degree of compliance of the speed limits on the section. $\theta(n) = 1$, represents the perfect compliance. $\theta(n) < 1$, a driver driving below the speed limits, and $\theta(n) > 1$, faster than the speed limits. $\theta(n)$ is a vehicle parameter that in AIMSUN can be sampled from a probability distribution, when such information is available, modelling implicitly in that way the aggressiveness of the drivers.
3. Speed limit in section or turning s : $S_{\lim nt}(s)$. This is a section parameter.

The actual speed limit for a vehicle n on a section or turning s , $S_{\lim nt}(n, s)$, is given by:

$$S_{\lim nt}(n, s) = S_{\lim nt}(s) \cdot \theta(n)$$

The maximum desired speed of vehicle n on a section or turning s , $v_{\max}(n, s)$ is:

$$v_{\max}(n, s) = \min[S_{\lim nt}(n, s), v_{\max}(n)]$$

Thus the local maximum desired speed $v_{\max}(n, s)$ equals the desired speed $V^*(n)$ in Eq. (1).

Influence of adjacent lanes

When the leading vehicle is driving along a section, the AIMSUN car-following model takes into account the potential influence of certain number of vehicles (N_{vehicles}) driving slower in the adjacent right-side lane –or left-side lane, when driving on the left–. The model calculates first the mean speed for N_{vehicles} driving downstream of the vehicle in the adjacent slower lane ($\text{MeanSpeedVehiclesDown}$). Only vehicles within a certain distance (MaximumDistance) from the current vehicle are taken into account. We distinguish two cases: 1) the adjacent lane is an on-ramp, or acceleration lane, and 2) the adjacent lane is any other type of lane. Apart from N_{vehicles} and MaximumDistance parameters, the user can define two

additional parameters, MaximumSpeedDifference and MaximumSpeedDifferenceOnRamp. Then, the final desired speed of a vehicle on a section is given from the following logic:

if (the adjacent slower lane is an On-ramp)

{MaximumSpeed = MeanSpeedVehiclesDown + MaximumSpeedDifferenceOnRamp}

else {MaximumSpeed = MeanSpeedVehiclesDown + MaximumSpeedDifference}

DesiredSpeed = Minimum ($v_{\max}(n, s), \theta(n) * \text{MaximumSpeed}$)

This procedure ensures that the differences of speeds between two adjacent lanes will always be lower than MaximumSpeedDifference or MaximumSpeedDifferenceOnRamp, depending on the case.

Effects of Grades

The maximum desired acceleration for a vehicle (vehicle_acc) is a vehicle parameter defined by the modeller in AIMSUN for all vehicles belonging to the same class. The influencing of the section grade on the vehicle movement is taken into account by increasing or reducing the acceleration and deceleration rate. The maximum acceleration for a vehicle on a section is a function of the grade and the maximum desired acceleration for the vehicle i.e.:

$$\text{accel} = \text{Maximum}(\text{vehicle_acc} - \text{grade} * 9.81 / 100.0, \text{vehicle_acc} * 0.1)$$

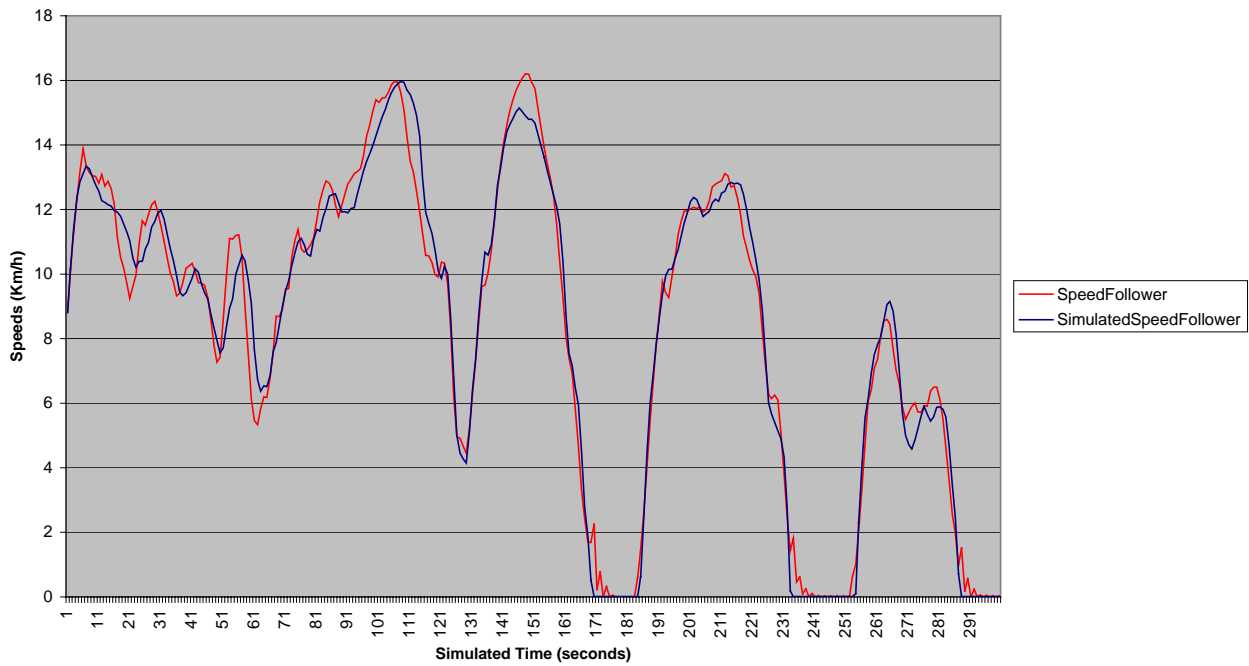
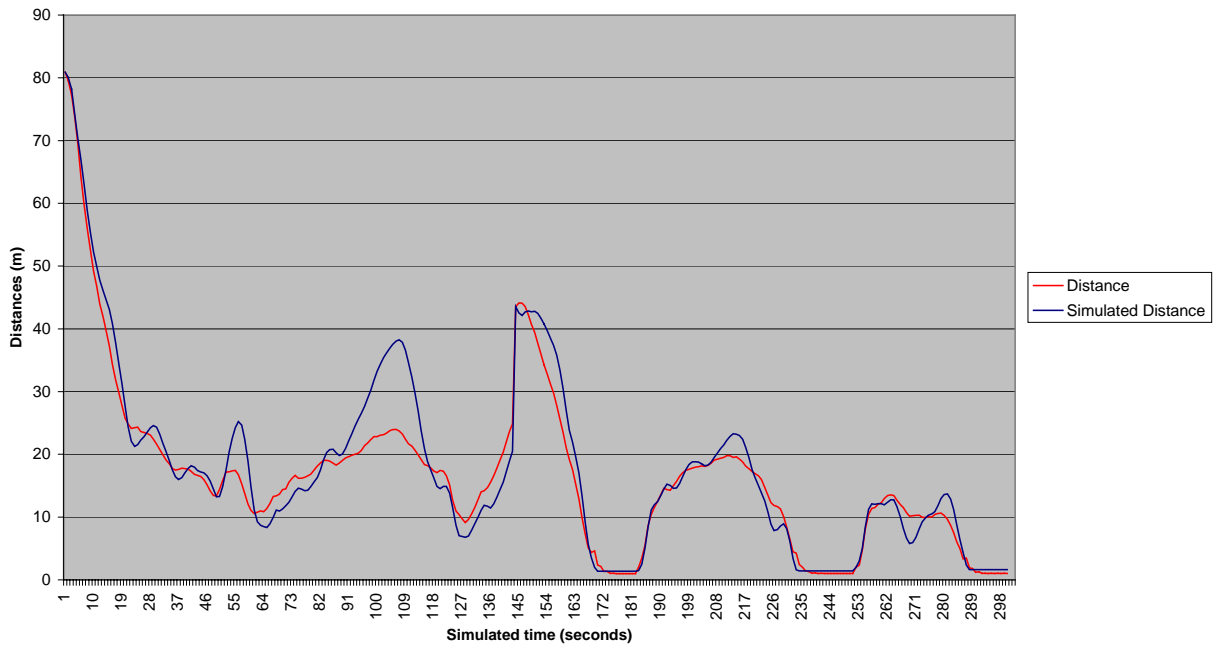
In order to avoid zero or negative acceleration values, a minimum value of 10% of the maximum desired acceleration for the vehicle is used.

3. MODEL CALIBRATION AND TESTING

In addition to numerous tests performed by the research team, the car-following model in AIMSUN has been tested and calibrated in various real life projects; we present here the benchmark test performed based on the data and the methodology supplied and proposed by a research group from Robert Bosch GmbH [MAN98]. This test employed a set of field data and most of the micro simulator developers in Europe and North America were invited to participate. The test for the car-following model had two parts: a direct test on the car-following model and a test on the ability of the micro model to reproduce macroscopic relationships between fundamental traffic variables.

3.1 Direct test of the car-following model

“The primary task of a car-following model is to reproduce realistic car-following behaviour. The reality has been measured with a radar sensor equipped vehicle recording distance and relative speed to the front car (additionally to the own car's speed) in a 100 ms cycle, see [BLE96] for further details. One specific sequence of 5 min length has been chosen to perform the comparison. This sequence has been recorded under stop&go conditions during an afternoon peak on a one-lane-per-direction fairly straight road in Stuttgart, Germany. Stop&go is most challenging to the models because the free flow behaviour is relatively easy reproducible by any model. During the 5-min sequence several decelerations and accelerations of the front car have been observed. At one moment after 144 sec the front car turned off resulting in a distance step of about 40 m. Because such a manoeuvre can always happen in real traffic the models have to be able to deal with. Note that it can't be the target of a simulation model to reproduce exactly the measured behaviour of this specific driver in the specific test vehicle. Driver and vehicle variations have to be respected. Hence, the main focus lies on qualitative differences. But the fairly good reproduction of the behaviour indicates a model's realism. To give an impression of similarity to the measured behaviour a quantitative error metric on the distance seems to be reasonable. To avoid overrating discrepancies for large distances a relative metric was chosen weighted by the logarithm and squared. Only the values after each second have been considered”.

Figure A3.1: Measured and Simulated Speeds**Figure A3.2: Measured and Simulated Distances**

The error metric used to measure the accuracy of the fitting between measured and simulated values was:

$$Em = \sqrt{\sum \left[\log \left(\frac{d_{sim}}{d_{meas}} \right) \right]^2}$$

where d_{sim} is the distance of the simulated vehicle, d_{meas} is the distance measured with the test vehicle, and \log denotes the logarithm base 10. Figures A3.1 and A3.2 display the curves for the observed versus simulated relative distance and speed between leader and follower. The numerical value for the error metric for the AIMSUN model was 3.4726. These results show that the AIMSUN car-following model is able of a fairly good reproduction of the observed values. The numerical value of the error metric outperforms those

provided for most of the currently used models (see [BLE96] for details). The table below extracted from [BLE96] shows the results for MITSIM, two versions of VISSIM (Pelops and Wiedemann) and other microsimulators.

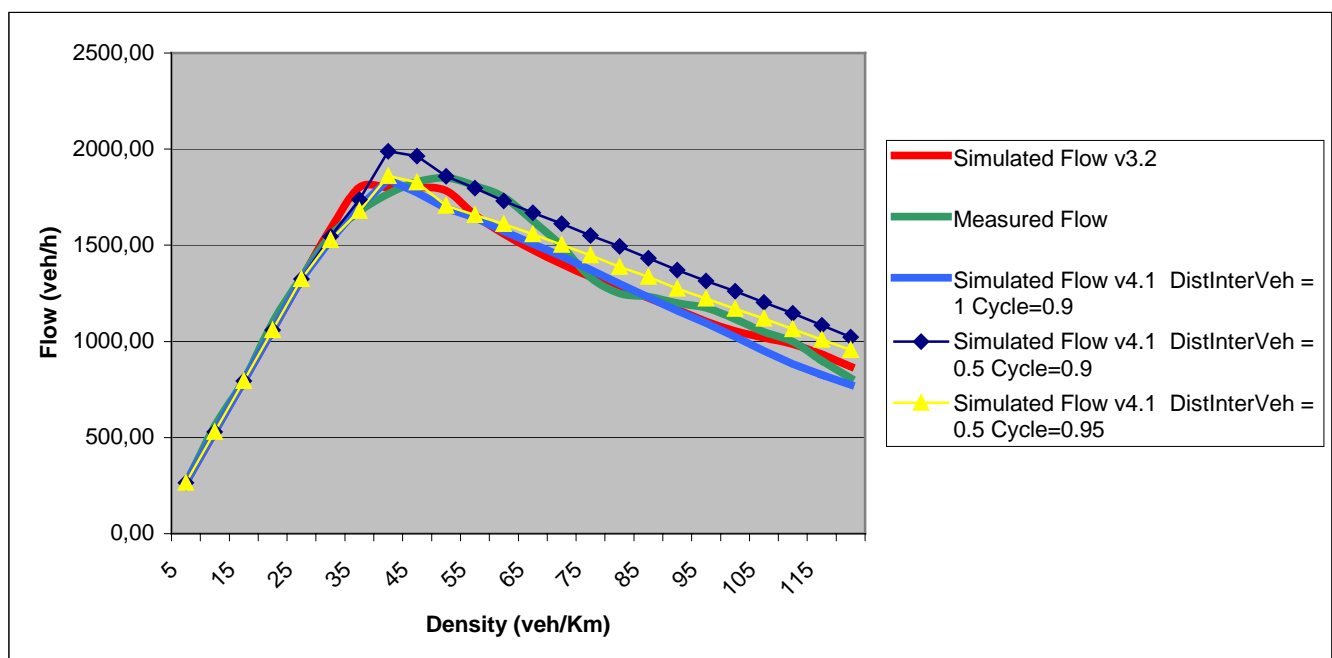
Model	MITSIM	Wied/Pel	Wied/Vis	NSM	OVM	T3M
Deviation	3.75	14.01	10.67	24.51	9.37	2.40

3.2 Testing the ability to reproduce macroscopic behaviour

An additional test to analyse the quality of the microscopic simulator is to check the ability to reproduce macroscopic behaviour. Also the research team at Bosch proposed in [BLE96] a test to compare various microscopic simulators: “The macroscopic behaviour of a microscopic model can be most easily tested by simulating the traffic on cyclic one lane roads. This excludes any effects of lane changes and node passing and concentrates on the car-following task. For this study, a cyclic road of 1000 Mts. length was used. A fixed number of vehicles has been initially set with speed value 0 km/h at randomly determined positions. All vehicles had the same length of 4.5 m and the drivers had the same free flow speed of 54 km/h. Starting with this initial situation a 10 minute time period was simulated without any measurements to reach traffic conditions which are achievable by the model's behaviour itself. After the starting phase the traffic behaviour has been recorded at one local measurement point during a simulation time of 2 hours (exact passing time and speed value of each vehicle). The fixed number of vehicles for the simulation run was varied in discrete steps to realise different traffic densities. To visualise the results the traffic flow has been drawn versus the density (given as the number of initially set vehicles on the 1 km ring). The maximal mean traffic flow value of about 1800 veh/h is known as a quite realistic value for longer periods of measurement time. Under urban traffic conditions this maximal flow is typically reached at higher density values than for freeway traffic”.

The results of AIMSUN for the simulated flow density curve versus the empirical one for the second test are displayed in figure A3.3, and they appear to be fairly reasonable. This subjective perception is confirmed by the values of the error metric to measure the fitting between the measured and simulated values as before, that in this case is $E_m=0.063381$. The graphics in figure A3 also show the sensitivity of the AIMSUN Car-following model to variations in the values of the model parameters. In absence of microscopic measurements the adjustment of model parameters to fitting macroscopic empirical curves for the relationships between the fundamental traffic variables could also be used as an alternative procedure for model calibration.

Figure A3.3: Empirical versus simulated flow-density curves



REFERENCES

- [AKC82] Akcelic (1982). Progress in Fuel Consumption Modelling for Urban Traffic Management. Australian Road Research Board Research Report ARR No. 124.
- [BAR94] J. Barceló, J.L. Ferrer and R. Grau. AIMSUN and the GETRAM Simulation Environment. Technical Report. Departamento de Estadística e Investigación Operativa. Universidad Politécnica de Cataluña (1994).
- [BAR95] J. Barceló, J.L. Ferrer, R. Grau, M. Florian, I. Chabini and E. Le Saux. A Route Based Variant of the AIMSUN Microsimulation Model. 2nd. World Congress on Intelligent Transport Systems, Yokohama, (1995).
- [BAR98] J. Barceló, J.L. Ferrer, D. García, M. Florian and E. Le Saux, Parallelization of Microscopic Traffic simulation for ATT Systems Analysis. In: P. Marcotte and S. Nguyen (Eds.), Equilibrium and Advanced Transportation Modeling, Kluwer Academic Publishers, 1998.
- [BAL98] Balci, O. Verification, Validation and Testing, in: Handbook of Simulation: Principles, Methodology, Advances, Applications and Practice, Ed. by J. Banks, John Wiley, 1998.
- [BLE96] T. Bleile, W. Krautter, D. Manstetten and T. Schwab, Traffic Simulation at Robert Bosch GmbH, Proc. Euromotor Seminar Telematic / Vehicle and Environment, Aachen, Germany, Nov. 11-12, (1996)
- [CAS96a] Cascetta, A. Nuzzolo, F. Russo and A. Vitetta. A modified logit route choice model overcoming path-overlapping problems. Specification and some calibration results for interurban networks. In Transportation and Traffic Theory, Proceedings of the 13th International Symposium on Transportation and Traffic Theory, ed. by J.B. Lesort, Pergamon press, 1996.
- [CAS96b] Cascetta, A. Nuzzolo, F. Russo and A. Vitetta. A modified Logit model for route choice with explicit path enumeration. 4th Meeting of the EURO Working Group on Transportation. University of Newcastle, 1996.
- [COW75] Cowan, R.J. (1975). Useful Headway Models. Transportation Research, Vol. 9, pp. 371-375.
- [DIJ59] Dijkstra, E.W., "A Note on Two Problems in Connection with Graphs", Numerische Mathematic, 1, 269-271, 1959.
- [DOT94] Department of Transport (1994), New Car Fuel Consumption: the official figures December 1994, UK DoT.
- [FEL94] M. Fellendorf, VISSIM: A microscopic simulation tool to evaluate actuated signal control including bus priority, Technical paper, Session 32, 64th ITE Annual Meeting, Dallas, 1994.
- [FER82] Ferrerira, L.J.A. (1982), Car Fuel Consumption in Urban Traffic: The Results of a Survey in Leeds using Instrumented Vehicles. ITS Working Paper N° 162. Institute for Transportation Studies. University of Leeds.
- [GAB91] J.F. Gabbard, Car-Following Models. In: M. Papageorgiou (Ed.), Concise Encyclopedia of Traffic and Transportation Systems, Pergamon Press, Oxford, (1991).
- [GIP81] Gipps, P.G. (1981). A behavioural car-following model for computer simulation. Transportation Research Board, Vol. 15-B, pp. 105-111.
- [GIP86a] Gipps, P.G. (1986). A model for the structure of lane-changing decisions. Transportation Research - B. Vol. 20-B, No. 5, pp. 403-414.

- [GIP86b] Gipps, P.G. (1986). MULTSIM: A Model for Simulating Vehicular Traffic on Multi-Lane Arterial Roads. *Mathematics and Computers in Simulation*, 28. 291-295.
- [GRA93] Grau, R. and J. Barceló (1993). The design of GETRAM: A Generic Environment for Traffic Analysis and Modeling. Research Report DR 93/02. Departament d'Estadística i Investigació Operativa. Universitat Politècnica de Catalunya.
- [HEL61] W. Helly, Simulation of Bottlenecks in Single-Lane Traffic Flow. In: R.C. Herman (Ed.), *Theory of Traffic Flow*, Proc. Symp. Theory of Traffic Flow, Elsevier, Amsterdam, pp. 207-238, (1961)
- [HER59] R. Herman, E. W. Montroll, R. Potts and R.W. Rothery, Traffic Dynamics: Analysis of Stability in Car-following. *Operations research*, 1(7), pp. 86-106, (1959).
- [HUG98] Hughes, J. AIMSUN Simulation of a Congested Auckland Freeway, Proceedings of the 6th EURO Working Group Meeting, Göteborg, 1998
- [KLE95] J.P.C. Kleijnen, Theory and Methodology: Verification and Validation of Simulation Models, *European Journal of Operational Research*, Vol. 82, pp. 145-162, 1995.
- [KOK00] Koka, M. J. Hourdakis, P. G. Michalopoulos, Computer Aided Testing and Evaluation of Adaptive Ramp Control Strategies, Paper presented at the 79th Annual Meeting of the Transportation Research Board, Washington, January 2000, accepted for publication in the TRB.
- [LAW91] Law, Averill M. and Kelton W. David, *Simulation Modeling and Analysis*. McGraw-Hill International Editions. Second Edition. 1991.
- [MAN98] D. Manstetten, W. Krautter and T. Schwab, Traffic Simulation Supporting Urban Control System Development, Robert Bosch GmbH, Corporate Research and Development, Information and Systems Technology, P.O. Box 10 60 50, 70049 Stuttgart, Germany, (1998).
- [PID92] Pidd, M. *Computer Simulation in Management Science*, John Wiley 1992.
- [QIY96] Qi Yang and H.N. Koutsopoulos, A Microscopic Traffic Simulator for Evaluation of Dynamic Traffic Management Systems, *Transp. Res. C*. Vol.4, No. 3, pp. 113-129, (1996).
- [THE66] Theil, H. *Applied Economic Forecasting*, North-Holland, 1966
- [QUA92] QUARTET Deliverable N°2 (1992), Assessment of current Tools for Environmental Assessment in QUARTET, DRIVE II Project V2018: QUARTET, September 1992.
- [VAN96] M. Van Aerde, B. Hellinga, M. Baker and H. Rakha, INTEGRATION: An Overview Traffic Simulation Features, paper presented at the 1996 Transportation Research Board Annual Meeting.
- [WIC77] D.A. Wicks, INTRAS-a microscopic freeway corridor simulation model. Overview of simulation in highway transportation, 1, pp.95-107, (1977).
- [WIE74] R. Wiedemann, Simulation des Verkehrsflusses. Schriftenreihe des Instituts für Verkehrswesen, Heft 8, Universität (TH) Karlsruhe, (1974).

Copyright

Copyright © 1992-2004 TSS-Transport Simulation Systems

All rights reserved. TSS-Transport Simulation Systems products contain certain trade secrets and confidential and proprietary information of TSS-Transport Simulation Systems. Use of this copyright notice is precautionary and does not imply publication or disclosure.

Trademark

AIMSUN and GETRAM are trademarks of TSS-Transport Simulation Systems.

Other brand or product names are trademarks or registered trademarks of their respective holders.