STS08 021
Examensarbete 30 hp
Maj 2008

Traffic route guidance using feedback of predicted travel times

Improving travel times in the Berlin traffic network

Arvid Bergsten
Daniel Zetterberg

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Traffic route guidance using feedback of predicted travel times

Arvid Bergsten and Daniel Zetterberg

Traffic congestions constitute a problem in many large cities. Congestions can be handled by reducing the network demand, expanding the infrastructure, or by utilizing the road network more efficiently. This master thesis presents a methodology for route guidance, based on automatic feedback control from the current traffic situation. Through variable direction signs or individual in-car devices, all vehicles with a certain origin and destination (which are both normally intermediate) are guided to take the currently fastest route. In this paper the traffic is guided over one of two alternative routes with the goal of a Nash equilibrium, i.e. that every guided agent travels the fastest path. Nash equilibrium occurs when the routes have equal travel times, or when all agents use a route with shorter travel time.

Predictive data describing how the system reacts to control measures is fundamental to control the traffic in an optimal way. Feedback of observed travel times results in a system with highly oscillating travel times. Given this background, the task of this thesis has been to present a model that predicts route travel times, with the purpose of improving the performance of traffic route guidance. The approach used is automatic feedback control, and therefore some basic terminology from control theory is used throughout the report.

The model introduced in this paper needs no parameter estimation but uses only static information about traffic network and on-line counting of vehicles. Simulations show that with reliable travel time predictions at hand, optimal control can be achieved by bang-bang control; a controller that needs no parameter estimation. The result is a guidance method that has the potential to work on any location without prior estimation of location specific parameters.

A microscopic traffic simulator, MATSim, is used for developing the prediction model and evaluating its system effect with route guidance. Simulated in-car COOPERS (Co-operative Systems for Intelligent Road Safety) devices are used for data collection and for transmitting the guidance to vehicles. The prediction models and the feedback control are evaluated in two different traffic networks; a topologically simple test network and a realistic location in the Berlin network. The results of the simulations are promising; guidance with predictive models results in significantly shorter average travel time than guidance based on observed travel times. Evaluation with an economic measure indicates that drivers benefit economically from predictive route guidance.

Handledare: Kai Nagel
Ämnesgranskare: Bengt Carlsson
Examinator: Elisabet Andrésdóttir
ISSN: 1650-8319, STS08 021
Sponsor: COOPERS

Populärvetenskaplig sammanfattning

Trafikköer är ett allt vanligare inslag i storstädernas vardag. Många timmar går åt till att sitta i köer på morgonen och kvällen. För att göra något åt problemen kan kapaciteterna på vägarna ökas genom utbyggnad, men ofta finns det också delar av trafiknätet som inte är maximalt utnyttjade. Dessa vägar är i allmänhet längre och tar längre tid än de vägar som utnyttjas flitigast, men då köer uppstår på huvudvägarna börjar andra alternativ bli intressanta. Det är redan vanligt förekommande med GPS-utrustning som kan föreslå den kortaste vägen till ett resmål. COOPERS är ett trafikprojekt som bland annat syftar till utvecklingen av informationsutbyte mellan fordon och infrastruktur. Genom att samla in data för hur trafiken ser ut, t.ex. antalet fordon och flöden till och från vägarna, så kan körtiderna för olika alternativa rutter beräknas och informationen använda till att guida fordon till den rutt som har den kortaste körtiden.

Ett annat sätt att guida fordon att ta snabbaste vägen mellan A och B är att mäta körtider vid B och guida fordonen vid A till den vägen vars uppmätta körtid är kortast. Denna sortens guidning är dock för sen, vilket blir ett problem ju längre rutterna är. Den vägen som ett fordon leds in på bör vara den väg som kommer att ha den kortaste körtiden, inte den som hade det vid ett tidigare tillfälle.

Målet med detta arbete är att utveckla generella prediktionsmodeller för körtidsberäkningar som är bättre än att guida med uppmätta värden. Styrmålet är att uppnå Nashjämvikt, dvs. att alla agenter guidas till den för tillfället snabbaste vägen. Modellerna och vägvisningen implementeras i JAVA i den agentbaserade trafiksimulatorn MATSim och testas dels på ett enkelt testnätverk och dels på ett scenario i ett Berlinnätverk.

Tester bekräftar entydigt att modellerna fungerar i testnätverket. Guidningen med prediktionsmodeller är bättre än guidning med uppmätta värden. Denna generella slutsats stärks av berlinsimuleringarna, men för att dra slutsatser om skillnader mellan de olika modellvarianterna krävs mer och tillförlitligare utvärderingsdata.

Table of Contents

Part I - Background

1. Introduction

The area of traffic guidance is currently developing fast. The background is both increasing problems with traffic congestions in major cities and introduction of new traffic management technology such as GPS devices. Congestions are naturally due to several causes. If the demand on the road network is much bigger than its capacity, the situation is clearly hard and cannot be solved without physical, and generally expensive, extension of the infrastructure. But this is not always the case. Often, only some of the streets are congested at a given time while the demand on other streets are far beneath the maximum capacity. In such cases, it can be fruitful to guide vehicles into less crowded streets in order to use the available network optimally.

It is a common assumption that people take the fastest route in a normal traffic situation. But when the traffic fluctuates due to incidents and unexpected demands, people often take the slower alternative.

How should this guiding be carried out? There are many ways of estimating the traffic situation. Public radio information about traffic is often based on phone calls, cameras or observing helicopters. Route guidance can also be given through variable road signs with direction advices or directly to in-car GPS devices. Measurements of travel times from point A to point B via two alternative routes can be used to guide vehicles at A to take the faster alternative. The theory applied is classic feedback control that is commonly used in many industrial processes to achieve a certain predefined control goal. The control goal in our case is to direct vehicles into the currently fastest route.

This approach has been tested with good results; see for example Diakaki (1997). However, measuring the travel times at B generally sends a control signal that is too late, since there is a time lag between A and B. What is really wanted, is a reliable prognosis of the time it will take a vehicle to travel to B, starting from A in this moment – not the travel time for the latest measured vehicle that reached B. These adequate travel times must be predicted. The idea of this master thesis is to calculate the travel times based on the present traffic situation. The development and testing of the prediction model were carried out in the traffic simulating software MATSim.

1.1 Aim and hypotheses

The overall aim of this thesis work is to develop travel time prediction models that improve the feedback control compared to using measured data. The models will be implemented and evaluated in MATSim. A general approach will be taken when developing the models so that they work independently from the guidance location, meaning that no parameter tuning will be necessary.

In focus are accident scenarios. When there is a queue under normal conditions – for example in the morning peak hours – the road network is normally used to its full capacity and no better alternatives can be found. Hence there is no potential for guidance control. But when there is a temporary capacity reduction caused by an accident or road maintenance work, the traffic can be rerouted on an alternative route that still has a capacity buffer.

1.2 Method

Our approaches for predicting travel times are based on physical principles of traffic flows. We have implemented the models in JAVA code using the MATSim structure, integrating them with the pre-existent classes for drivers, streets, intersections, etc. In the development phase and in the primary evaluation phase a small test network was used that consists basically of two routes between point A and point B. For a final and more realistic testing scenario we identified a location in Berlin that seemed suitable for this kind of guidance control, and evaluated the models further by running them on the large Berlin MATSim network on that location.

1.3 Structure of this report

Part I: Introduction, theory and reactive control
The first chapter introduces feedback control as a traffic guidance method and presents the aim of the thesis work. Chapter two introduces the theoretical framework and the problems with reactive feedback control, i.e. using a measured output. In Chapter three, the MATSim framework; the system studied, its dynamics and limitations are presented. Also, the networks for testing the models, an introduction to the COOPERS project and a brief overview of the JAVA implementation is found here.

Part II – The prediction models
Chapter four is the start of part two, and contains the basis for all predictive models in this thesis work. Chapter five to nine describes the models and the evaluations of them on the test network.

Part III - Evaluations on a Berlin scenario and conclusions
The last part includes chapter 10, that presents evaluations from simulations on a Berlin network, chapter eleven with conclusions from all evaluations and a last chapter with ideas for further research that would push the development forward.

2. Theory

To introduce the relevant theoretical framework, the basic guidance scenario is here presented. Basically, drivers want to go from the point A to the point B and there are two different routes available. In accordance with the MATSim nomenclature, drivers will hereafter be called *agents* and point A and point B will be referred to as the *sign link* and the *destination lin*k, respectively. The agents have learned from experience what route has the shortest travel time. Optimally this will result in one of two possible cases. The first case is that all agents take the fastest route. The second case is that both routes are equally fast, and agents travel over both routs. Two routes can be equally fast if they have the same length and speed limit, or if one of them is shorter but currently contains more agents that make it slower than normal. At this situation every agents takes the fastest alternative; no individual agents would travel faster if he had chosen the other route. We say that the system is at *Nash equilibrium*, with an expression from game theory. The Nash equilibrium is the control goal for our route guidance. As mentioned earlier, the interesting scenario for route guidance is when something extraordinary happens, for example an accident. Since the agents do not know about the accident further down the route, many of them will take the slower route.

2.1 Guiding the agents with feedback control

The route guidance idea used in the preceding thesis work of Carl Rommel (2007) was a classic feedback control. The system output $y(t)$, i.e. the difference in travel times for the routes, is called the *Nash time*. This was measured and used as feedback signal. The difference between this output and the wanted output y_{ref} is then sent to the controller that calculates the next input (guidance direction) to the system.
In other words, basic feedback control works by calculating the control input $u(t)$ based on the size of the control error $y - y_{ref}$. In our case the control error is the difference between current Nash time and the Nash time defined by the control goal, which is zero, corresponding to Nash equilibrium with equal travel time on both routes. A flowchart representation of the closed loop control system is displayed in Figure 1.

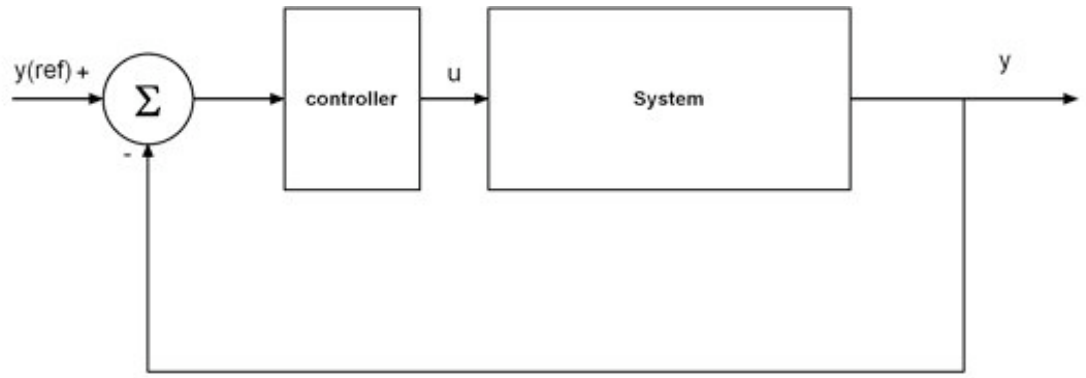

Figure 1. Feedback flowchart of classic feedback control.

2.2 The feedback controller

There are many different controllers that can be used for feedback control. In our thesis work, we will solely use bang-bang control, or relay control as it is also called. This is an all-or-nothing approach. In every time instance, all the passing traffic is directed into the faster of the two routes. Mathematically, one can write the strategy in the following way.

If $y(t) > 0$	*then* $u(t) = 1$	the traffic is directed into the alternative route
If $y(t) < 0$	*then* $u(t) = 0$	the traffic is directed into the main route
If $y(t) = 0$	*then* $u(t) = u_0$	no guidance is given, vehicles take either route

where $y(t) = TT_{main}(t) - TT_{alt}(t)$, where TT_i means travel time for route *i*.

A bang-bang controller is robust in terms of that it is system independent. It has no control parameters that must be tuned for the specific routes, unlike for example a PID-controller. This is a major advantage, especially when no system model is at hand. One drawback is that such control tends to be too strong. If the control output is very close to zero, the bang-bang controller still controls in the same manner as if the control error were very big. This leads to an oscillatory behaviour. The oscillations can be reduced with a dead-zone on the output signal so that no control signal is given when the Nash time is within an interval close to zero.

The oscillatory behaviour of a bang-bang controlled system also depends on how often a new control signal is applied. Assume that the control signal is updated every five minutes. If the demand is high and all agents are directed into the same route for five minutes, this route will have become considerably slower by the time the control signal changes. This obviously generates oscillations, which would be mitigated if the control signal would be proportional to the control error. However, if the control signal would change as soon as the system state changed, no oscillations (of considerable amplitude) would be generated in first place. But then one must have reliable non time-delayed information about when the sign of $y(t)$ changes. Having such information, and in-car devices instead of variable direction signs on the roads, every car can be guided individually according to the latest calculated control signal. We return to the practical aspects of the communication of the control signal in the next chapter.

2.3 Problem with reactive control based on measured system output

Observations of the time difference between the two routes do not give us reliable information about what travel time an agent at the sign link can expect. The time lag from input (the control signal) to output (the Nash time), results in oscillatory system behaviour and a guidance signal that will never be up to date. To illustrate the time lag and the oscillatory behaviour, two graphs are shown below. We can see the time lag in the travel time diagram in Figure 2. Here, no guidance control is applied. The dashed black graph is the measured travel time on which a reactive control signal would be based, if used. For example, at 4000 seconds simulation time the expected travel time for an agent entering the main route was 800 seconds. But when that vehicle reaches the destination link, its travel time is measured to 1000 seconds. The two curves contain the same values (in this test case all agents travels a complete route), but shifted depending on the length of the queue. If the travel time is 500 seconds, then the measured value will show up 500 seconds "too late". The longer the travel time, the larger is the time lag.

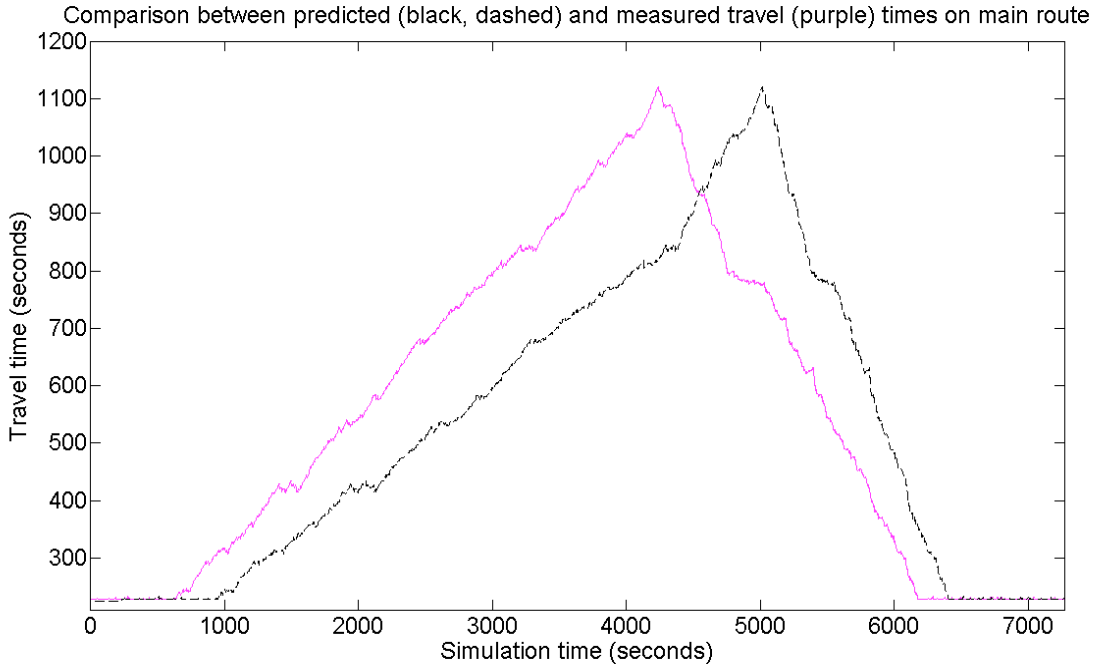

Figure 2. Illustration of the system time lag.

As the control signal is derived from "old" information, agents are directed into a route also after it has become the slower alternative, generating the oscillating Nash times showed in Figure 3. Unlike the last diagram, the Nash time plot was generated from a simulation with the route guidance activated.

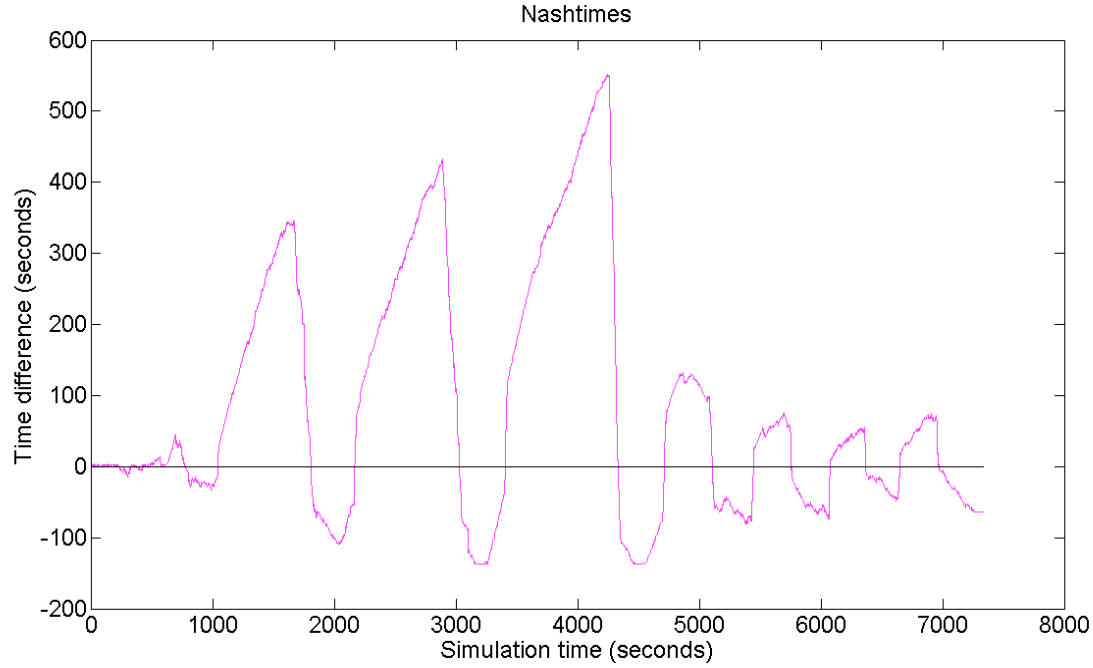

Figure 3. Nash times with reactive control. The asymmetry is due to the capacity reduction on the main route, which has longer queuing times.

The cause of the oscillating behaviour is the inadequate system output that we use for the control. As seen in Figure 2, there is a discrepancy between travel time values used

for control and the travel times that are later measured for the guided vehicle. Therefore, if a queue is discovered, i.e. the measured travel times are bigger on one route; the controller starts sending vehicles in the other direction. But it is not until these vehicles reach the end of the route that the controller can notice the change that the control achieved. Thus, the system output needed is not observable, as the measurements are too old.

Our primal idea was to compensate for this by identifying the time delay of the system; i.e. after how many seconds a change in the control signal causes a change in the measured Nash time. This is problematic as the time lag is not time invariant but depends on the queuing time (which in a controlled system is affected by the control signal). As showed in the travel time graph, the longer it takes to travel the route, the greater the difference between the current values and the measured. This is intuitively comprehensible as the time lag is determined completely by travel time on the routes, which of course depend on the traffic situation, e.g. queues.

2.4 Modelling theory – predicting the system output

Instead of using outputs that are "too old" we want to predict the output for the agents before they are guided. For that we need a model that predicts the travel times according to the present traffic situation. With predicted system output, the flow chart of the system will look like in Figure 4.

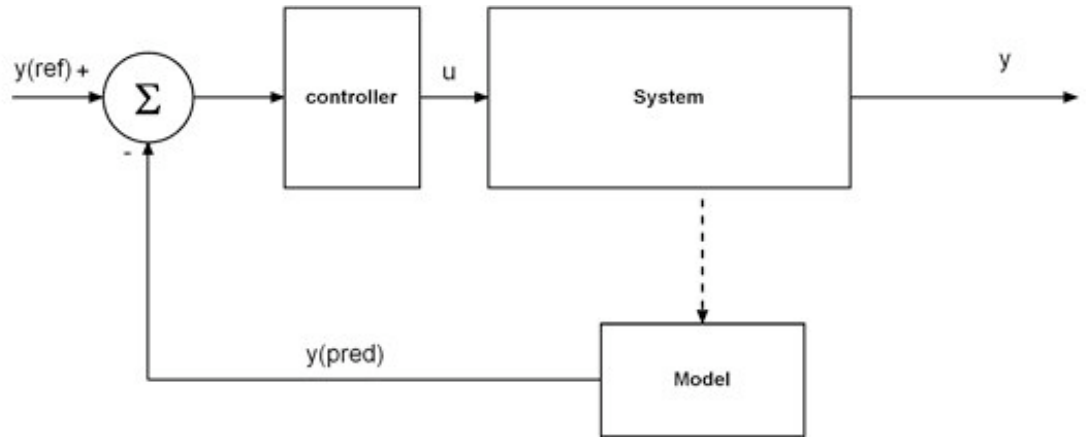

Figure 4. Feedback flowchart with prediction. The dotted arrow is the measurements from the system, i.e. number of agents on the links, etc.

There are different approaches to model dynamic systems. Physical modelling is often preferred for simple systems where the dynamics follow well-established physical theories. When no system knowledge is at hand, there are a number of black-box approaches. When identifying a system with such methods, parameters in differential equations are mathematically optimized to suite the relationship between input and output as good as possible. Time lags are often included in such models, but the problem for the traffic guidance is that the time lag between input (left or right for an agent) and the output (difference in travel time between that agent and another agent travelling on the other route) is varying over time, depending on current traffic conditions. If there is presently a long queue, it will take a long time before a change in guidance will be noticed at the end of the routes. Due to this and also for the reason that it is always a good idea to try the simplest things first, we choose the approach of physical modelling.

One could say that Figure 4 does not represent a feedback control loop, using a strict definition of "system feedback". There is no feedback from the system output *y(t)* – in fact the observed output plays no part at in the control. Nevertheless is there an obvious feedback from the system to the control signal, and therefore we will continue to use the term "feedback control" for the route guidance.

2.5 Guidance over more than two alternative routes

Our scenarios – and our modelling methodology – limit the controller to using two alternative routes for guidance. It is easy to see that a road network could be used more efficiently if the guidance control could direct traffic via several different routes. Later in this report we discuss the difficulties we had finding a suitable test location in Berlin. As we focus on highway accidents, it is hard to find one single alternative route that has capacity enough to handle a substantial part of a highway traffic flow – and still having a travel time short enough to be a reasonable alternative.

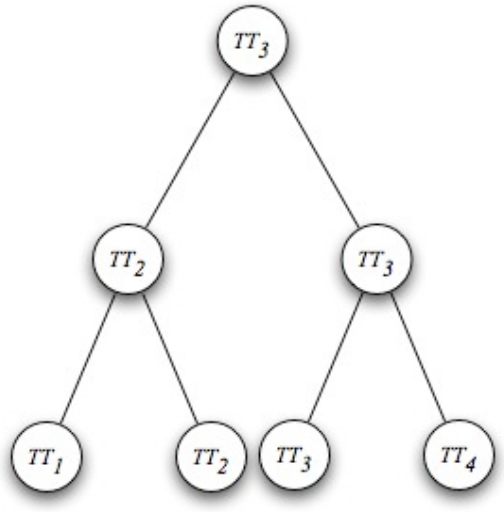

Figure 5. Diagram for multiple route guiding.

These limitations in the road network can be dealt with by using more than one alternative route to compensate for a capacity reduction on one big main route. It is possible and theoretically quite simple to connect many two-route controllers of the kind previously described, so that a traffic management system can guide traffic over many alternative routes. Figure 5 is a binary tree where every leaf represents the current travel time for a route, predicted with the models developed within this thesis work. Every parent displays the travel time for the fastest of its children routes. The current structure of the tree in Figure 5 informs us that route 2 is currently faster than route 1, but route 3 is the fastest of all four alternatives. All agents going from A to B are therefore directed into route 3. As soon as any of the other routes become faster route 3, that route travel time will become the root of the tree. The guidance will shift, keeping the system at Nash equilibrium.

2.6 Modelling traffic systems – current research

Looking at traffic theory in the literature and previous similar attempts gave a starting point to model the system physically. There are several different ways to model traffic. One is the cellular automata model, i.e. models that consist of a regular grid of cells, each in one of a finite number of states. The time in cellular automata is discrete and the state of a cell at time t is a function of the states of its neighbourhood at time $t - 1$ (Wikipedia, 2008). Each cell could be a part of a road.

Another way of modelling traffic is by microscopic, discrete models that handle individual vehicles; MATSim is an example of such an approach. Traffic systems can also be simulated in macroscopic, continuous fluid-dynamic traffic flow models (see for example Helbing, 2003). In the article *Real-time freeway traffic state estimation based on extended Kalman filter: a general approach* (Papageorgiou, 2005), Markos Papageorgiou, and Yibing Wang develop "a general approach to the real-time estimation of the

complete traffic state in freeway stretches […] based on the extended Kalman filter." A macroscopic traffic flow model is presented in a state-space form and tested with good results. However, this approach is computationally complex and hard to implement for testing in e.g. MATSim, especially when the freeway stretches are long and there are several intersections or on and off ramps.

The general idea though was adopted, i.e. to use physical conditions of the roads such as capacity, length, intersections, together with speed limits and sensors that can give information about current velocities, number of agents and traffic densities on a road segment. Traffic densities are not treated as explicitly as in the work of Papageorgiou and Wang. In their model, as well as in several other models (see for example Vandaele, 2000), road capacities are expressed as the maximum density of vehicles of a road. In this thesis, the maximum capacity of a road segment is defined as the maximum rate with which it can release its vehicles. A reasonable assumption used in our model development, is that in case of queue, agents are served in this maximum rate. One could say that the simple queuing model used in this work, is similar to classic queuing theory, involving servers and agents. Bottleneck links are the servers and vehicles are the agents. Little or no attention is paid to the agent-to-agent dynamics. The modelling method used in this thesis work is thus a macroscopic traffic flow model.

3. MATSim

The abbreviation MATSim stands for *Multi-Agent Transport Simulation Toolkit*. MATSim is a micro-simulator toolbox for demand modelling, simulation, iteration and analysis of transportation scenarios and is built on the queuing model introduced by Gawron (Gawron, 1998). "Micro-simulator" means that the simulation is agent-based, simulating on a per-car-basis, but the mobility simulation is sometimes labelled "mesoscopic" as some aspects of individual agent behaviour are described macroscopically. Unlike other traffic simulating software, MATSim generates individual activity plans for all agents, who in turn dynamically generate a demand on the road network.

3.1 The street network

A traffic network consists of links that represent roads, or parts of roads, tied to each other at nodes, which represent intersections. A link has three basic properties that constitute its traffic dynamics; its free speed velocity, its flow capacity (number of cars that can leave the link per time unit) and a storage capacity (the maximum number of cars that can occupy the link at the same time) (Cetin, 2003, page 8).

The intersections follow a three step logic: A car will move from one link to another if 1) it has arrived at the end of a link, 2) it can be moved according to the flow capacity of the link and 3) there is free space on the link that it is about to enter. If two cars from different links want to enter a third link at the same time step, the simulation will choose which car randomly proportional to the capacity of the incoming links (Cetin, 2003, page 9).

The links in MATSim have only one lane with a capacity that represents all lanes on the road segment in the real world. This simplification has become a problem on highway ramp locations in our simulations, a problem we will come back to in Section 3.6. Another simplification of the Berlin traffic network itself is that no traffic lights have been implemented.

3.2 Population, the agents

Simulations in MATSim need a number of interacting agents. The agents constitute a population that uses the network for different activities. Every agent has a plan that he tries to follow during the day. A plan is a sequence of activities such as home, work, leisure and travels that connect the different places where the activities take place. An activity has a start and end time and the agents know over which links they should travel to get to each activity. The plans are derived from real data. In other words, the population used in the Berlin simulations represents the real people in Berlin, having similar activities. To run the simulation on standard personal computers, the population file used in the Berlin simulations has approximately 350 000 agents, constituting a 10 percent sample of the total number of agents in Berlin. The capacity of the network is reduced to a corresponding level in order to keep the simulation realistic.

3.3 Replanning

To develop traffic situations that resemble real world scenarios, the agents can replan from day to day. In MATSim, altering an agent's behaviour is possible only through modification of its plan (Illenberger, 2007, page 2). The system remembers several

plans for every agent and measures the performance of each plan. In this way agents learn what routes are fast, and discard plans with slower routes. If the links that they travelled were congested yesterday, they might try another route today. After a number of iterations of this sort, the population learned to take the approximate best route alternatives if nothing unexpected happens, such as an accident. In the Berlin simulation, the population is the result of 80 iterations of the same day. After 80 opportunities to replan, it is expected that the agents take the optimal or close to optimal routes between their activities, and no further day-to-day replanning is made (Illenberger, 2007, page 5). This implies that there is no need to actively guide the traffic unless something extraordinary happens that causes queues to build up unexpectedly. As all Berlin simulations were made from the same iteration, the day-to-day replanning was never active within this thesis work.

Agents can also do within-day replanning. Under certain conditions, e.g. if a link ahead looks congested, the agent can modify its day plan spontaneously and choose a faster route. It is this feature, which makes it possible for agents to react to unforeseeable incidents, that is used for the online traffic guidance that we develop in this thesis work. According to predictions of the travel times on different alternative routes, the fastest way is provided to the agents, which will listen to the recommendation with a certain probability, a compliance rate. Other within-day replanning features than the route guidance subject to this report are deactivated in our simulations to increase transparency.

3.4 How MATSim is used in this project

The general process of simulating traffic in MATSim is illustrated in figure 6. We use MATSim for developing and evaluating prediction models, and for simulating feedback control with the output predicted from the models. From MATSim we measure number of agents; travel times on routes and on links. We also get information about links; capacities, lengths and speed limits, from the network data. MATSim allows us to set accidents on specific locations, which is vital for our purposes since this alter the traffic situation and change the best route choices for the agents. There is also a number of preset evaluation data that are given by MATSim, such as average travel times and a population score. Those have been modified to suite our purposes better and we will return to this in the evaluation chapter. For an implementation of this kind of route guidance in the real world, it would be necessary with a physical infrastructure for the online measuring of the traffic data mentioned. We return to this matter briefly in chapter 4 of this report.

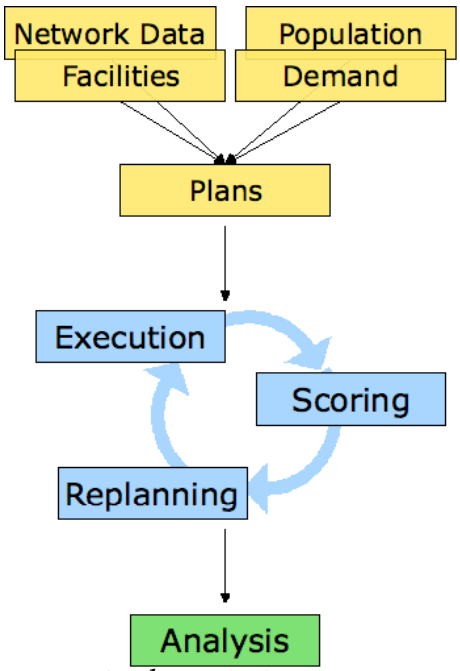

Figure 6. The MATSim simulation process.

As in any model development that employs a simulator for testing, the conclusions drawn from this thesis work must be limited. It is possible that the microscopic simulator MATSim and the prediction models that are developed can show similar qualities – traits that are not a true reflection of real traffic. Normally we assume that the results simulated in MATSim are representative for real traffic systems. Hence, the models are designed to take care of the flaws that the models show in the MATSim simulations that are being studied and, but this is if and only if the flaw can be associated with a

problem in the real world, i.e. no model or optimization has been done that did not have any foundation in the real world. In this matter, knowing more about the MATSim dynamics would not have been beneficial to improve the results of the models.

3.5 The Coopers-project and the Test Scenario.

The MATSim development team cooperates in some projects with COOPERS, which stands for **CO**-**OP**erative Syst**E**ms for Intelligent **R**oad **S**afety. It is an European research, development and innovation activity within the Call 4 (Co-operative Systems and in vehicle integrated safety systems) of the 6th Framework Programme by the European Commission - Information Society and Media (COOPERS, 2008). The COOPERS project focuses among other things on route guidance and is indirectly involved in this thesis work project and hence a short presentation is in place.

Aims of the COOPERS project

From its start in 2006, the COOPERS project has focused on developing telematic applications between vehicles and infrastructure in order to provide infrastructural and safety related status information. Two specific goals for the project are (Nemec, 2008):

- Selective traffic jam warning and guidance based on current traffic situation on highway segments
- Short term ETA (Estimated Time of Arrival) based on current traffic situation on the road network

3.6 Berlin test location

For the evaluation of route guidance using the prediction models that were to be developed, we chose a Berlin highway section relevant according to the aims with the COOPERS project. In addition to these we had the following criteria on the test locations:

- The main route must have a high demand during simulation hours. When an accident reduces the capacity of the road temporarily and partially (e.g. to 50 percent for an hour), this should generate a queue that increases the travel time on the main route so that the alternative route becomes faster.
- The alternative route should have capacity enough to handle the extra traffic that it receives due to the accident on the main route. If it is not big enough to take any additional traffic, then it will not be a real alternative.
- The alternative route is generally longer than the main route, but it should not be too long. Under normal conditions agents take the main route because its cost, in terms of travel time, is lower than for any of the alternatives. Nevertheless, the alternative should not be as costly that you stick to the main route; no matter how much time you will spend queuing.

The location finally chosen is on the northbound highway 111 in north-western Berlin, close to the Tegel airport. After analyzing MATSim simulations, this location seemed to fulfil the criteria above and to be suitable for our purposes. The highway constitutes the main route and has a capacity of 47 600 agents/hour, while the alternative route has a capacity of 22 300 agents/hour. To make the simulations run faster, the population of Berlin is reduced to 10%. This is compensated for as the capacities of the links are also reduced to 13%, which is something that has been proved by previous research to be realistic. This must be kept in mind when studying the tables that contains number of

guided agents. An accident that reduces the capacity by 50% is set far north on Highway 111, red on the map in Figure 7. The blue route is the alternative route. The traffic dynamics of the alternative route is quite different from that on the main route in terms of having many intersections and therefore a lot of additional traffic. Predictions on the alternative route should therefore be harder to do than for the main route that only has one additional on/off-ramp location.

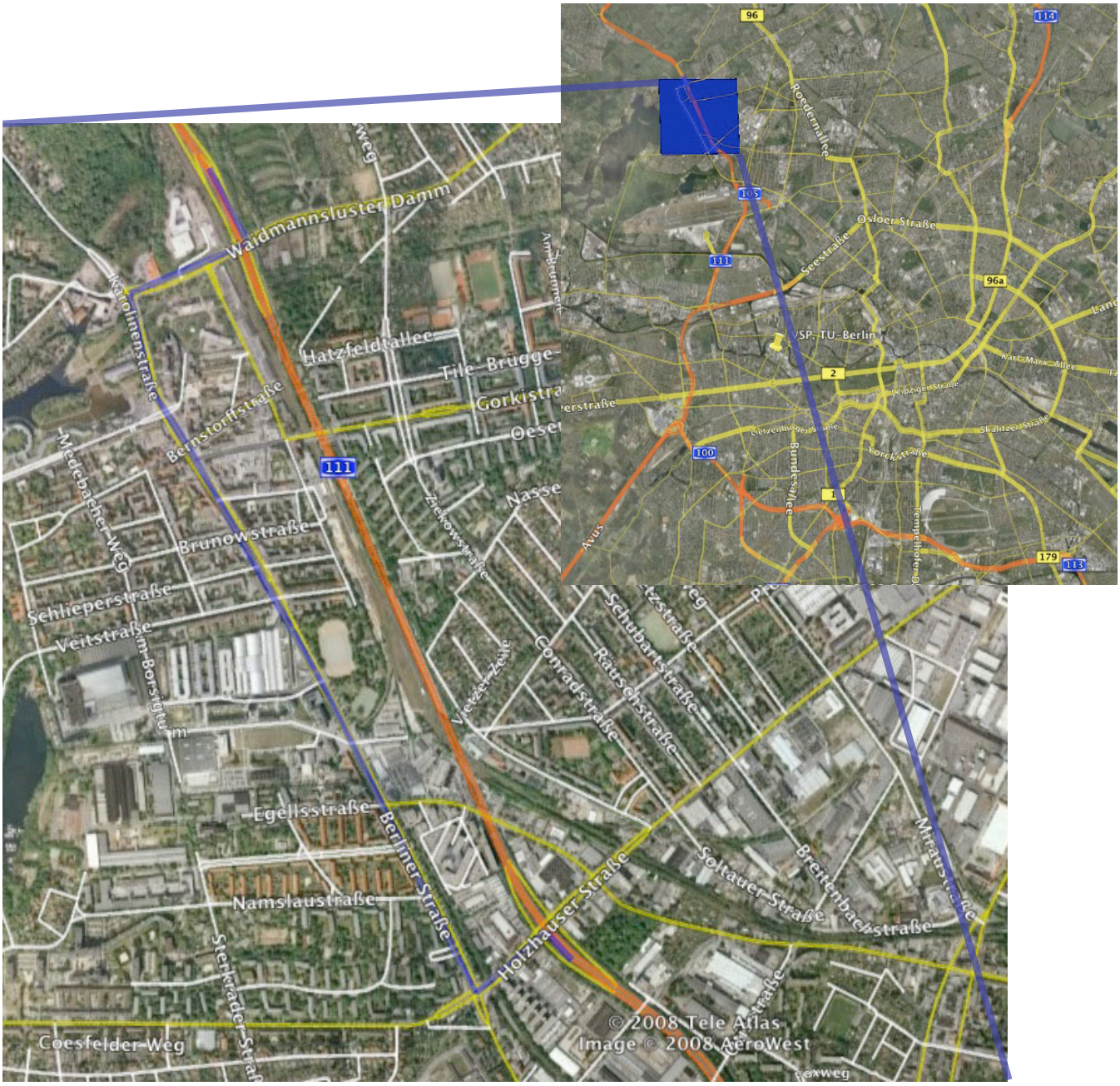

Figure 7. The main route (red) and the alternative route (blue) of the Tegel test location. (Google Maps)

We experienced some problems with the road network while trying the route guidance initially. The on and off ramps of the highway were congested immediately when just a few agents were directed off the highway and onto the alternative route. The flow capacities of these links were doubled, and in this way it was made possible to evaluate route guidance on the Tegel location. Increasing the capacities is not necessarily equivalent to widening the road in the real world. The effect can also be achieved by a

telematic modification of the intersection, i.e. giving the guided agents a special treatment (e.g. a green light).[1]

3.7 Infrastructure needed. Coopers-devices and VDS

Our models and implementations are fairly flexible. They require sensors that count vehicles at certain measurement points. Unlike when using reactive control based on measured travel times, the prediction models presented in this thesis do not require that the vehicles be tagged, just counted. The sensors must provide information about number of vehicles on a road segment ("links" in MATSim) as well as the current in- and outflows to the route. The route guidance sent to in-car devices, or to a stationary variable direction sign, a *VDS*, at the sign link. In this thesis work is assumed that individual in-car devices are available such as the RDS-function of the car radio or a GPS, and we refer to them as COO*PERS-devices*.

For the simulations we assumed that 80 percent of the receivers of the guidance actually follow the advice. The rest stick to their original plans, perhaps thinking that their own experience of the traffic is more reliable than the automatic traffic management system. The 80 percent compliance rate was used in the preceding work of Carl Rommel (2007), and to us it seemed to make the simulations more realistic. The positive effect of the guidance (measured in travel times, Nash times and also model fit) is expected to be lower when not all agents comply with the control signal. Still, the indications should point towards the same conclusions, regardless of the compliance rate.

COOPER-devices should be able to supply the relevant information, but if they only will be able to receive and not send information, sensors at certain locations could be used. Instead of using a COOPERS infrastructure, the models developed in this thesis work can be used with variable direction signs. The biggest difference would be that the direction sign could not be updated too often since that would confuse the drivers. Further discussion about VDS implementations can be found in the thesis work of Carl Rommel (2007).

[1] The capacity modifications (link number and modification factor) can be found in the file TrafficManagementConfiguration.xml. Naturally, the capacities are constant over all simulations.

3.8 The test network

The simulations on the Berlin network are very time consuming and the traffic situations are complex. To develop our models, a test network was designed together with populations suited for trying the features of the models. The test network is shown in figure 8.

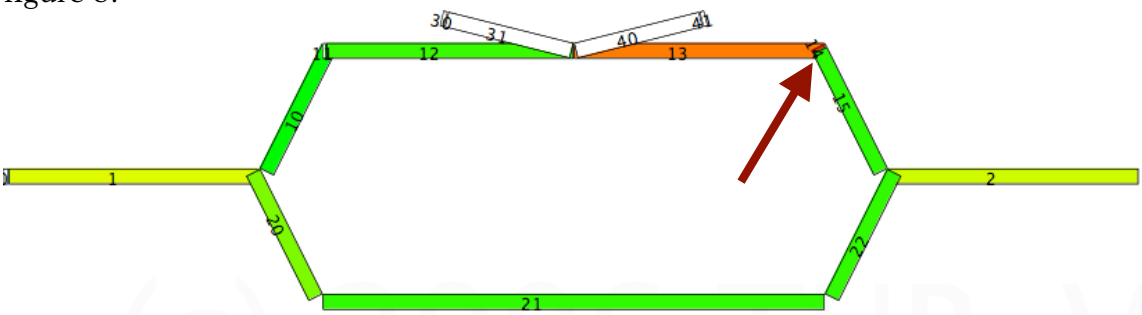

Figure 8. The test network. Green colour represents free flow conditions and red represents congestion. The accident location is indicated by the arrow.

The test network is designed so that the capacity of one link equals the sum of the capacities of the incoming links. This seemed like a reasonable assumption and suitable for testing the features of the models. Agents go from left to right and the guiding takes place at the first crossing. The upper route is referred to as the main route and the lower as the alternative route. The accident is set on link 14, which is the short link immediately after link 13, which is the long red link in figure 8. On the main route, there are two additional links, one for incoming and one for outgoing traffic if such scenarios are needed. Traffic from or to these link will not be guided and is looked upon as a disturbance. The colour scheme represents the traffic density on the links. White links are not trafficked at all, green links have low density, yellow represents medium density, and red means that the link is completely congested.

3.9 Implementation

MATSim is a large and complex simulator and during the work with this thesis, most of the architecture was perceived as a black box whose inner dynamics was known nor to us, nor to the models. To facilitate the reader's understanding of MATSim and how the prediction models were integrated, this chapter presents the JAVA-classes that are the closest and most relevant to the prediction models.

VDSSign: From prediction to guidance signal

The name `VDSSign` is a remnant from the earlier thesis work of Carl Rommel who initially used Variable Direction Signs to communicate with the agents. This class, as well as some other implementations, was refactored during our work, but the name remained.[2]

[2] As said before, a big difference between using a VDS-sign and in-car COOPER devices is that the guidance signals going out to the agents can be updated every second having COOPER devices, while updating the guiding on a big road sign every second will confuse the drivers quite severely. The java class VDSSign is still the class communicating with the agents, but in our implementation it can do it without paying any respect to the update intervals.

When the route guidance is activated as a traffic management module, an instance of `VDSSign` is created for every two-route guidance system. The task of `VDSSign` is to ask the `ControlInput` object for the current system output and transform this into a control signal. When the control input is an object of `ControlInputImpl1`, VDSSign gets the measured Nash time (reactive control). Otherwise it gets a system output predicted by either `ControlInputSB` or `ControlInputMB` (more about these classes in Part II).

The VDSSign sends the Nash time a `FeedbackControler`, which returns a control signal. The `BangBangControler` returns –1 or 1, indicating the fastest route. The `PControler` – which is not used as argued for earlier in this report – returns a value in the interval [–1, 1], depending not only of the sign of the Nash time but also the size. VDSSign then communicates this value to the MATSim system for withinday-replanning, which makes the agents replan and change their route. If the Nash time is zero, the VDSSign does not give any guidance and the agents stick to their original plans.

ControlInput: making the predictions

The Interface `ControlInputI` defines the different ControlInput implementations, i.e. the prediction models that we developed, by demanding some necessary methods like the `getNashTime()`. A more important class is `AbstractControlInput`, which contains the objects that are common for all prediction models. For instance, this is where all the information about the network is read when the simulation starts, like free speed travel times, inlinks and outlinks, natural bottlenecks, etc.

The abstract class also contains the important code that measures travel times. This is done by calling the MATSim method `handleEvent()`, which can be considered to represent sensors that detect when a vehicle leaves or enters a link. In this way flows and agents are measured and the models get the data they need for predicting the travel times for an agent at the sign link. In the ControlInput implementations, `getPredictedTravelTime()` predicts the travel time for every route, and the difference is being returned by `getPresictedNashTime()`. The class variables messageHoldTime defines how often a new guidance signal derived from the system output.

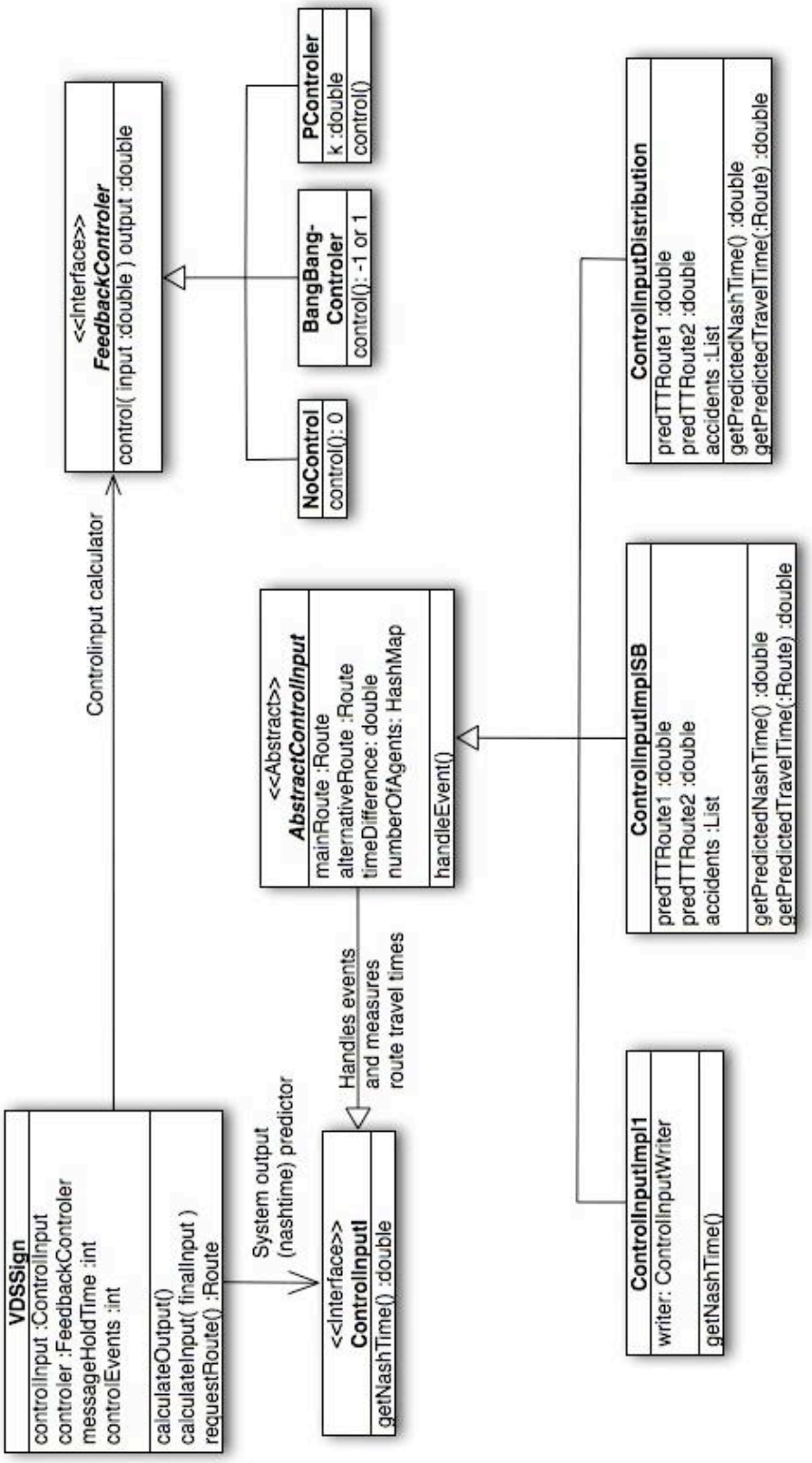

Figure 9, An UML–representation of the controller and prediction classes. The UML standard for illustrating class structures can be found for example on the web page: http://www.agilemodeling.com/artifacts/classDiagram.htm (29 Jan 2008).

Part II - Prediction Models

4. The basic theory used in the models

Starting off from a simple flow based queuing model, we need the number of cars on a road and the capacity of that road to calculate the travel time. As said earlier, in MATSim roads are represented by so called links. If the current link has a capacity of letting c vehicles out per second, a maximum of $c \cdot t$ vehicles can exit the road in t seconds. Thus, c is the outflow rate from a link when a queue is present. Assuming that all agents travel at the maximum allowed speed, called *free speed*, the time it takes to travel a link if no queuing is needed is the length of the link divided by the free speed. This time is called TT_{fs} , which strand for Travel Time at Free Speed.

If we count the number of vehicles that currently occupy a link, we can calculate the time it would take for all the agents to be released off the link, using the maximum outflow, c. This time could be called the queuing time, TT_q. In queuing theory terms, we calculate the total *serving time* for the agents. Comparing TT_{fs} and TT_q and choosing the greater of the two gives us an estimate of the link travel time. If $TT_q < TT_{fs}$, this can mean one of two cases, that we treat in the same way. Either there is a queue short enough to be completely served before an agent about to enter the segment arrives at its end. Else the agents presently on the road segment are even fewer, generating no queue at all. An agent entering the road segment will in both cases travel the whole segment by free speed, arriving at the end of the segment in TT_{fs} seconds.

Doing this kind of predictions for an entire route, consisting of several links, our first approach was to find one bottleneck link of the route and using this link's capacity for the predictions. This first model is called the *Single Bottleneck Model*, or just *the SB*, and will be described in detail in the next chapter. Focusing on one single bottleneck seemed as a good methodology a priori, based on the assumption that there are no big inflows or outflows that influence the traffic situation to great extent.

It is important to realize that TT_q is not just the time the queue needs to dissolve, but also our predicted travel time for an agent entering the route. Even if the agent spends most of the time on the route driving free speed and only a little time queuing at the bottleneck, his total travel time to the bottleneck will be determined by when the agent in front of him pass the bottleneck.

The assumption used above, that the maximal outflow of a link occurs when it has a queue might seem intuitively strange. One might think that a queue would mean lower velocity, and therefore that the outgoing flow would be beneath the maximum. Our definition of a queue states, however, that a queue arises when the flow of agents onto a link is higher than the outflow capacity of that link. This means that when agents are queuing on a link, the actual outflow has reached the maximum allowed by the physical characteristics of the link. Only when the capacity is reduced – as in the accident test case – will the presence of a queue imply that the outflow is beneath the normal maximum.

4.1 Evaluation measures

There are different ways of evaluating a model. The model itself can be evaluated through *fit* values. The fit value measures how much of the travel time variance that is explained by the model, in other words its prediction accuracy. It is calculated by

comparing predicted and measured travel times on the routes, as in the expression below. The prediction error $(y_i - \hat{y}_i)$ is calculated every second so index *i* corresponds to the simulation time. The simulation is set so that a new $\hat{y}_i$ is calculated every second. This should be compared to the travel time observed at the end of the route y_i seconds after $\hat{y}_i$ was calculated (when the agent entered the route). Therefore, the data set with predicted values was pre-processed so that $\hat{y}_i$ is the travel time predicted for the agent whose observed travel is y_i. Another interesting value is the Nash time, since minimizing them is the control goal. The Nash time is, as explained earlier, the travel time difference between the routes. Instead of "time difference" we will continue to use "Nash time" for the reason that this is the term used in the MATSim code. With the following formula we get a scalar value reflecting the size of the Nash deviation for an entire simulation (the value is hereafter referred to as "standard Nash deviation").

$$fit = 100 \cdot \sum_{i=1}^{N} \frac{(y_i - \hat{y}_i)}{y_i} \qquad \sigma_N = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}}$$

where is the measured Nash time
the predicted Nash time
the standard deviation of Nash times and
N the number of predictions.

It is worth mentioning that the predictions are done every time step, in our case every second, but there are not always agents passing and receiving the guidance. Also the measured times are extracted every second, which means that the latest measured travel time will be used in every step. In a time step when no car left the route, we compare the travel time measured for the latest vehicle that left the route with the travel time predicted *this time step*. If the traffic situation changes (e.g. number of agents on the route links) and the predictions reflect that, the fit value will be affected negatively even though the model is doing a good job, providing the controller with a more up to date input. This deviation is normally very small compared to the value of *y* and cannot even be seen in the travel time plots presented n the report. The error becomes clear in a scenario where only very few agents travel on one of the routes, since predictions still are made every second. Another interesting way of evaluating the models is to see how the *average travel times* are changed for the agents when using a certain model compared to using reactive control or no control. The travel times for every agent were extracted from the simulations. The average travel time for each of the two routes and the average travel time for both routes are calculated and used as measure of the system performance. Also the number of agents travelling on each route will be given as an evaluation measure. This measure allows us to compare the number of redirected agents for different models, and informs about how many agents "suffer" from travelling on the slower route, and respectively how many benefit from taking the faster alternative.

We use one more evaluation measure, the *score*. The score translates changes in travel time to economic effects (measured in euros per agents). This measure works best with a realistic population, and was there fore used only in the Berlin scenarios. The scoring functions will be explained further in Part III.

5. The Single Bottleneck Model

5.1 Description of the model

The model is based on the following strategy. We identify the bottleneck on the route, which is either the link with the lowest outflow capacity or, more interestingly, a link where an accident partially reduces the normal capacity. All the agents that are on the part of the route before the bottleneck are counted and so the travel time is predicted. The free speed time needed to reach the bottleneck is then compared with the time it takes for the cars on the route to pass the accident (calculated with the reduced capacity outflow in case of a known accident). If not all agents ahead of the agent currently being guided (the guidance object) will have passed the bottleneck within the free speed driving time the guidance object will have to queue. In this case the travel time for the guidance object is predicted to be the queuing time up to the bottleneck. This is added to the time it takes to finish the remaining part of the route, which is assumed to be travelled free speed. Finally, after the travel time has been predicted in this manner for both routes, the predicted time difference is returned to the guidance system.

In the end of this chapter we discuss some traffic dynamics that could cause problems for this rather simple prediction model. We expect, however, the route guidance to perform better with this model than with observed travel times. Recalling the background given in chapter 3, we can expect the effect to be greatest when the accident is far away from the sign link, with much congestion in between.
It is a prerequisite that the accident is "known" to the model, i.e. that the reduced capacity can be used in the calculations instead of the normal maximal capacity of the link. Naturally also the location of the accident must be known. This information is accessed easily in MATSim, but in the real world it is needed that the temporary capacity reduction is identified manually, for example by traffic management personnel. If the reduction is due to road work or similar, such information can be acquired beforehand. Later in the report a methodology for detecting capacity reductions automatically is presented.

It is important to realize why the travel time is predicted as the queuing time, and not the (free speed) travel time needed to reach the queue + queuing time. This would be incorrect because the queue will have partly dissolved when the agents reaches it. In fact, the predicted queuing time will be reduced with exactly as many seconds as it took for the agent to reach the queue, driving free speed. In other words, the TT_{fs} part is "eaten" by the TT_q part.

As discussed previously, when the traffic situation on the route is not affected significantly by inflows or outflows, i.e. additional traffic coming onto or going off the routes from intersections along the route between A and B, there is no reason to argue for that there is more than *one* bottleneck on a route. If the bottleneck is the narrowest part of the route, there must be significant additional inflows after it in order to build up a queue at a subsequent location. If there are two bottlenecks that are equally narrow, the model chooses the latter on the route since that bottleneck could swallow some of the time it takes to queue on the earlier, but never the other way around. When the background noise of inflows and outflows is large (in the Berlin scenario this background noise sometimes exceeds the proportion of controlled agents on the alternative route), it

becomes problematic and should be handled. The multi-bottleneck model presented later was developed for this reason.

Mathematical description

The notations used in the expressions are described below. The sums' index refers to the sequence of links on the route, where $i = 0$ refers the first link on the route.

tt_i	free speed travel time for link i
x_i	number of agents on link i
b	index of the bottleneck link (the link where the accident has occurred)
n	index of the last link on the route
c_b	bottleneck capacity (vehicles/second), assumed to be known to the model

The free speed travel time for the entire route is the sum of the free speed travel time of all the links that constitute a route:

$$TT_{fs} = \sum_{i=0}^{n} tt_i$$

The travel time in case of queue, TT_{queue}, is expressed by dividing the number of agents up to the bottleneck with the bottleneck capacity and last adding the free speed part.

$$TT_{queue} = \frac{\sum_{i=0}^{b} x_i}{c_b} + \sum_{i=b+1}^{n} tt_i$$

The largest value of the two calculations is taken as the travel time prediction of the route:

$$TT = \max\left(TT_{fs}; TT_{queue}\right)$$

These calculations are made every second for both routes. The Nash time is calculated as the difference between the two routes (called the "main route" and the "alternative route"):

$$\hat{y} = TT_{main} - TT_{alt}$$

This is the value used as input to the feedback controller, which guides the agents into the faster of the two routes.

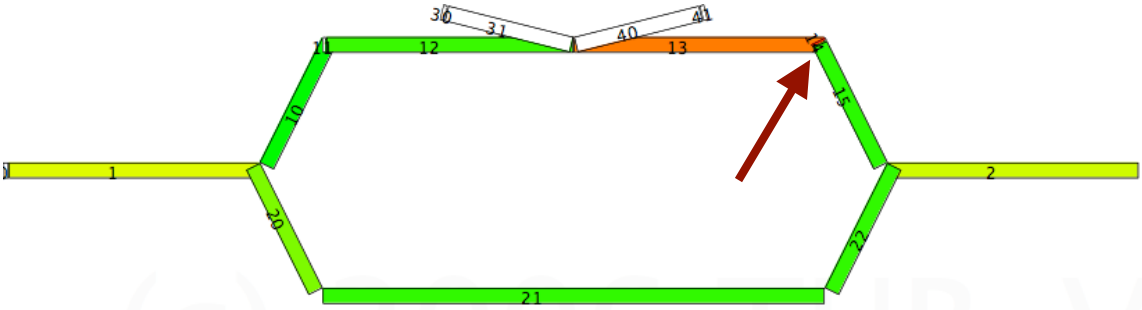

Figure 10. The test network used for the primary evaluation round.

5.2 Evaluation

In this chapter, the model performance will be evaluated on the test network described in section 3.8. Figure 10 helps us recall the scenario. According to the agents' plans, the traffic flows steadily and with equal density over the two alternative routes. The accident indicated by the arrow reduces the capacity, which the guidance system compensates for by directing a larger proportion of the agents into the alternative (the lower) route. An accident occurs at 7:15 in the morning and the road capacity is restored at 8:15. Between those hours, the capacity is reduced to 50%, which can illustrate roughly the real world case where one or two lanes are shut off due to the accident.

When switching on the control, the travel times decrease a lot and random noise is a big part of the differences in travel time between the main route and the alternative route. Therefore, a scenario with no control was run, but with the model predicting the travel times. In figure 11 we can compare the measured travel times (purple line) with the predicted (black, dashed line).

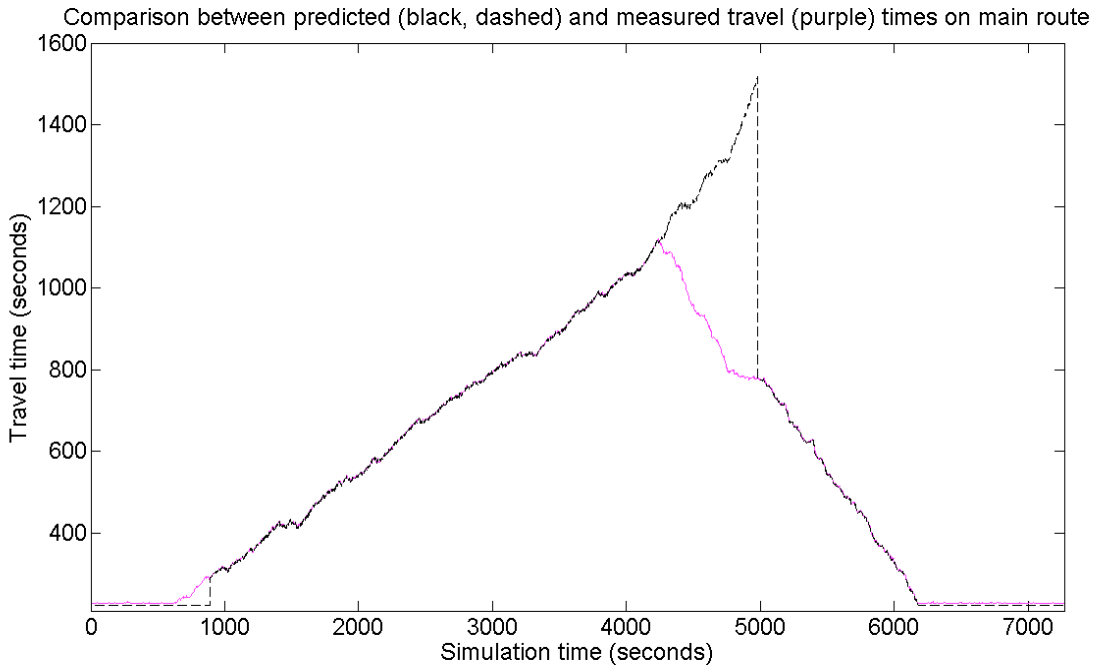

Figure 11: Comparison of the predictions of the Single Bottleneck Model and the actual travel times. In the diagram, the accident occurs at time t = 600 and ends at t = 4200. No guidance control is applied.

As shown, the model predicts almost perfectly except for when the accident starts and ends. To deal with the first error, one would need to know that an accident will occur at a certain time and where and how much it reduces the capacity, so that the model can predict the future travel times correctly and the guidance control can guide pro-actively. Similarly, the restoration back to normal capacity must be known some time before, so that one could guide the traffic as if the accident had already ended before it actually does.

Since the model predicts the travel times accurately according to figure 11, travel time improvements are expected when switching on the control. In table 1, the travel times achieved with the Single Bottleneck Model (abbreviated the "SB") are compared with the travel times of a non-accident case scenario, an accident scenario without control and with reactive control, i.e. the control using measured system outputs.

Table 1: Fit values and travel time improvements of the SB model. Accident reduces the normal capacity to 50 percent on link 14.

	Normal Case (no accident)	Accident Case, No Control	No Model (measured output)	Single Bottleneck Model
Main route:				
TT	229	568	359	232
Agents	3999	4000	3258	3405
Fit	-	-	58.8	96.8
Alternative route:				
TT	227	227	276	229
Agents	3999	3999	4741	4594
Fit	-	-	83.4	98.3
Total				
Std. Nash deviation	2.6	425.8	187.5	8.6
Average TT	228	397	310	230

The average travel time increases with 397 – 228 = 169 seconds when the accident occurs, which is a percentage increase of 74%. This increase in travel time is smaller with reactive control, which reduces it by 87 seconds. The percentage increase is thus 36%. Using the model, the average travel time decreases to 230 seconds, only 2 seconds worse than the case without an accident. From these runs, we can also see that the fit values are much higher using the model compared to using measured travel times. The fit on the main route improves from 58.8 to 96.8 on the main route, and from 83.4 to 98.3 on the alternative route.

In the table, we can also see the standard Nash deviation. Our control goal is a Nash time of zero, although in reality, a Nash deviation that is similar to the deviation without an accident is quite enough. The table shows that the reactive control cuts the standard Nash deviation in half compared to when no control is applied. The Single Bottleneck Model further mitigates the effects of the accident and brings the standard Nash deviation down to one fiftieth of the no control case deviation. As expected, it can be seen in figure 12 that the only severe deviation is in the beginning, when the accident occurs.

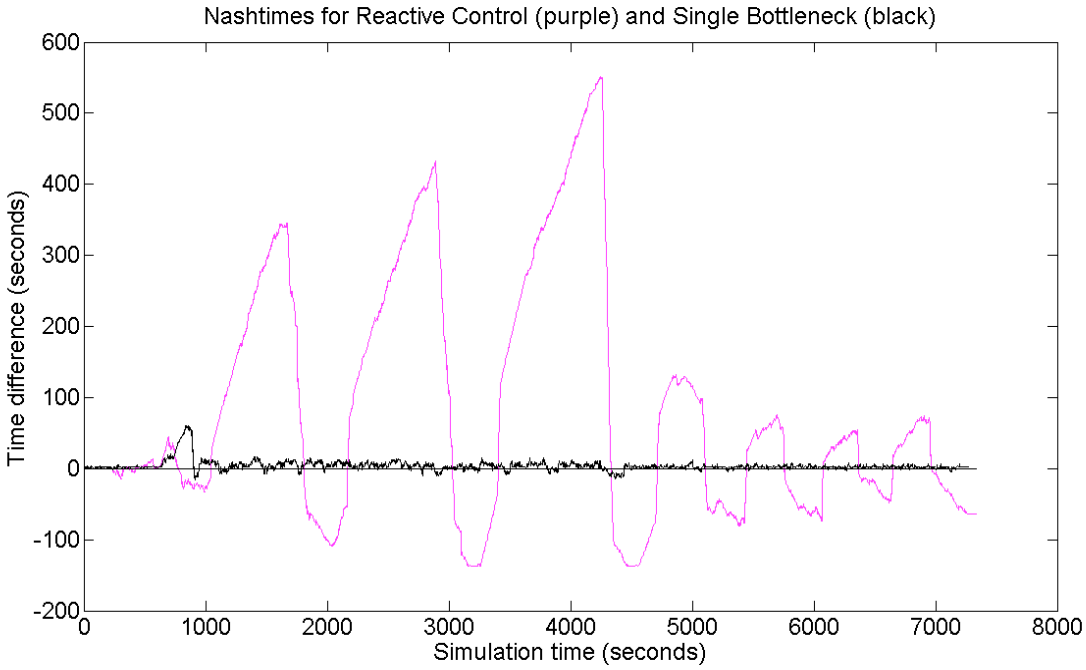

Figure 12: Nash times Comparison between reactive control and Single Bottleneck control

5.3 What does the Single Bottleneck model not handle?

Due to delays caused by technical difficulties, the performance of the prediction model was not evaluated on the Berlin scenario until in a late phase of the thesis work. Before that, however, the effect of some traffic dynamics on the prediction model were analysed theoretically. The following problems were pinpointed in order to develop the model further:

- Non-homogenous traffic density on routes. To the model, the route consists of two parts; the part before the bottleneck and the part after the bottleneck. When the traffic is concentrated to the beginning of the pre-bottleneck part, the Single Bottleneck model can falsely predict that an agent at the sign link will not queue at the bottleneck. The longer the route and the more heterogeneous distribution of agents, bigger will the prediction error be.
- Inflows and outflows during the prediction time horizon. The Single Bottleneck model only takes into account the amount of traffic that is *on* the route when the prediction is done. In case agents leave or enter the route between the sign link and the bottleneck, the queuing time will change.

Handling these problems is the subject of the two following chapters. Another feature that we wanted to develop in order to make the models less dependent of human operation was *automatic incident detection*. At a late stage, when the Berlin scenario was running, the effect of yet another model assumption was detected – there were two separate queues on the alternative route. In the Single Bottleneck model the travel time is assumed to be determined by not more than one queue. The topography of the test network had been too simple to generate a situation with multiple queues. Nevertheless, the last chapter in this part of the report is dedicated to the *Multi-Bottleneck model*.

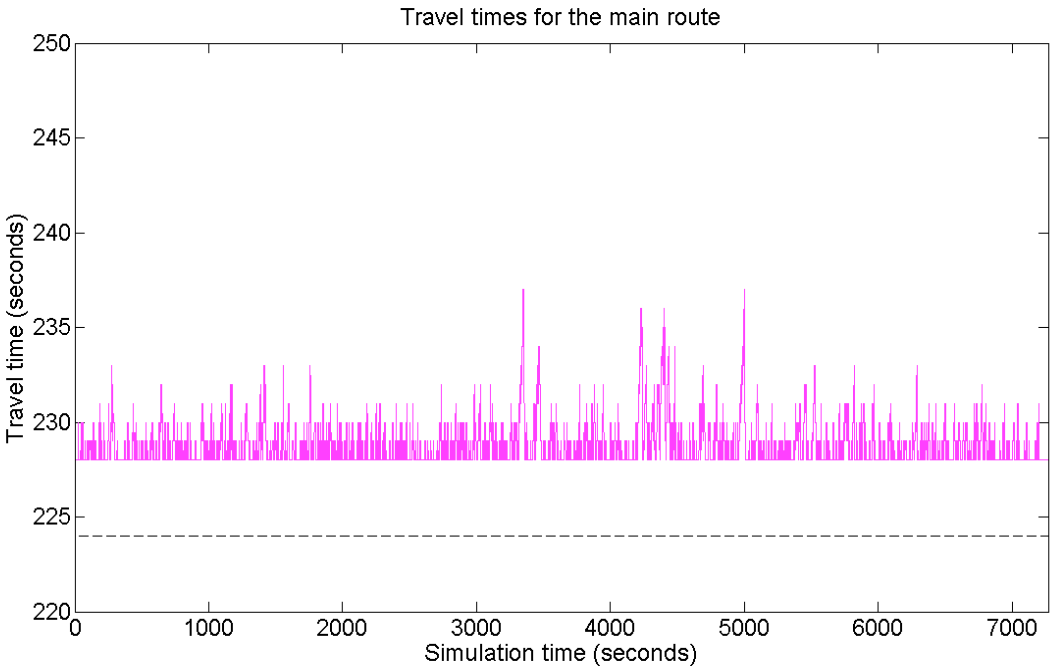

Figure 13. Travel times for the main route without an accident. It can be seen that not all agents travel with free speed even though there are capacity left. Maximum capacity of a route is 3000 agents/hour and in this run, there are 2000 agents/hour.

Figure 13 contains two prediction errors that will not be subject to further problem solving in this thesis work. The noise in the observed curve is due to non-homogenous traffic density on links. There are often micro-queues on single links that have much traffic, but not enough traffic to generate a "real" queue.[3] This randomness makes the simulation more realistic since not all vehicles in a real world scenario drive at the speed limit at all times (of course, some vehicles would also drive faster than that, which is not possible at all in MATSim). No stochastic influence on an agent's travel time will be modelled. However, one should have this random noise in mind when the prediction models are evaluated.

Apart from the noise, figure 13 also shows a static prediction error (four seconds in this plot). Knowledge about the MATSim dynamics tells us that this error stems from the fact that it takes one second for an agent to go over a node. This could of course be compensated for in the models to achieve better predictions, but since this simplification in the simulator does not reflects the intersection dynamics in real traffic very well, we chose not to compensate for such errors.

[3] On probable cause of the noise is that the plans for the agents are created with a randomized method. Two agents trying to leave the same place simultaneously will cause a instantaneous demand peak, even though the average demand is below the maximum capacity. On of the agent will have to wait and try again in the next simulation step (which is by default one second later).

6. The Distribution Model

6.1 The problem with unevenly distributed traffic

The problem with *heterogeneous or unevenly distributed of traffic* arises when there is much traffic on the beginning of the route and only sparse traffic on the rest. Consider the situation in figure 14. There is heavy traffic on the first part of the route and after that almost empty up to the bottleneck. The basic Single Bottleneck predicts that the agent now being guided will not have to queue, based on the number of agents on all links before the bottleneck. The sparsely trafficked part lowers the average below the critical number of agents needed for a queue to build according to the SB model. Nevertheless, when the "heavy charge" in the beginning of the route reaches the bottleneck there will instantly be a congestion containing all the vehicles in the charge. Thus, the guided vehicle will queue even though it was expected drive free speed. If the prediction was correct, the vehicle would not have been guided into the lower route.

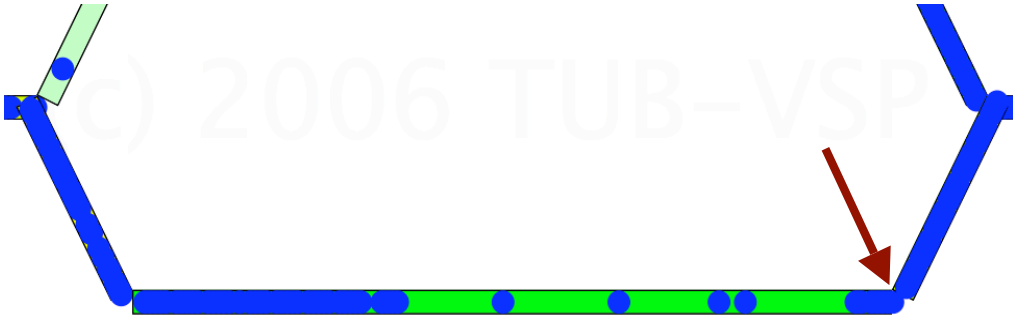

Figure 14. Sparse traffic on the part before the accident. The arrow indicates the bottleneck.

Because of the distribution of the traffic is not taken into account, the standard Single Bottleneck model makes prediction errors for the scenario described above. To detect heterogeneous traffic patterns we added a distribution check functionality to the Single Bottleneck model.

6.2 The SB model with distribution check

To solve the problem, multiple queuing calculations are done, i.e. the basic Single Bottleneck model approach is used a number of times, but for several subparts of the route. Important is that the same bottleneck capacity is used in the calculations for all parts. As in the SB model, agents are presumed to travel with free speed on all links after the bottleneck. Thus, the links after the bottleneck are added to what is called the *free speed part* of the route. The approach is then to go through the links, one by one, starting from the bottleneck and proceeding backwards against the beginning of the route. For every link, we check if the agents on the link are enough to cause congestion at the bottleneck. This is done with the basic SB method, i.e. the free speed travel time of the link is compared with the hypothetical queuing time; the time it will take for all agents on the link to pass the bottleneck using the number of agents and the bottleneck's outflow capacity. This is a way of finding out if a dense group of cars on a part of the route will queue at the bottleneck. If the calculations state that they will not queue, the link is added to the *free speed part*, but if they will, one more calculation must be done.

This time, the same basic Single Bottleneck calculation is done, but for the part of the route consisting of the links from the start of the route, up to the link in question; whose vehicles will make up the queue. Still, the same bottleneck is used. This calculation is done to see if that particular queue has vanished by the time the vehicle currently being guided has travelled the distance to the first vehicle of the dense group, which is the sum of the link lengths of that part of the route. Here, the first vehicle of that group is assumed to be at the end of the link that it is travelling.

Again, if the queue is predicted to have dissolved, we continue to check the next link. After a while, the entire route will belong to the *free speed part*, or there will be a *free speed part* and a *queuing part*. These are added to make up the prediction for the whole route.

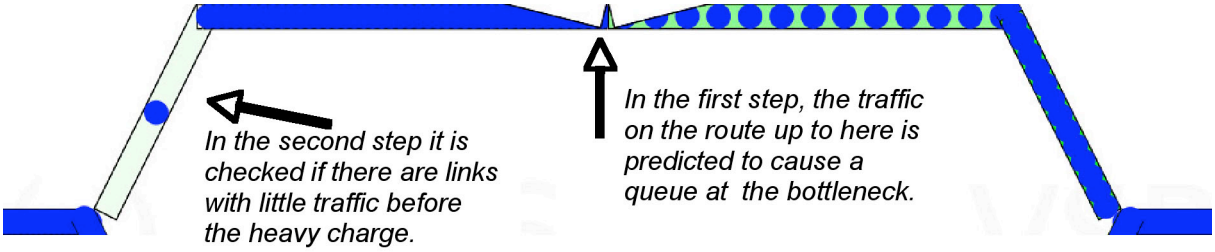

Figure 15. The two steps in the algorithm of the Distribution model.

6.3 Evaluation: Prediction abilities

When using the standard test population the problems described in the beginning of this chapter do not occur because the agents come in a steady flow. To evaluate the distribution model a new population was created which generates a steady base flow but also pulses that arise in reality, for example, because many agents leave home at the same time. Another example this can represent is the influence of traffic lights. In this evaluation scenario, the steady base flow is lower than it was with the normal population used for the basic SB evaluations, but pulses of heavy traffic comes on both routes approximately every ten minutes.

First, the new population was guided using the regular SB model. The graphs of the predictions of the travel times on both routes exemplify the problem faced by the basic Single Bottleneck model. The guiding is activated.

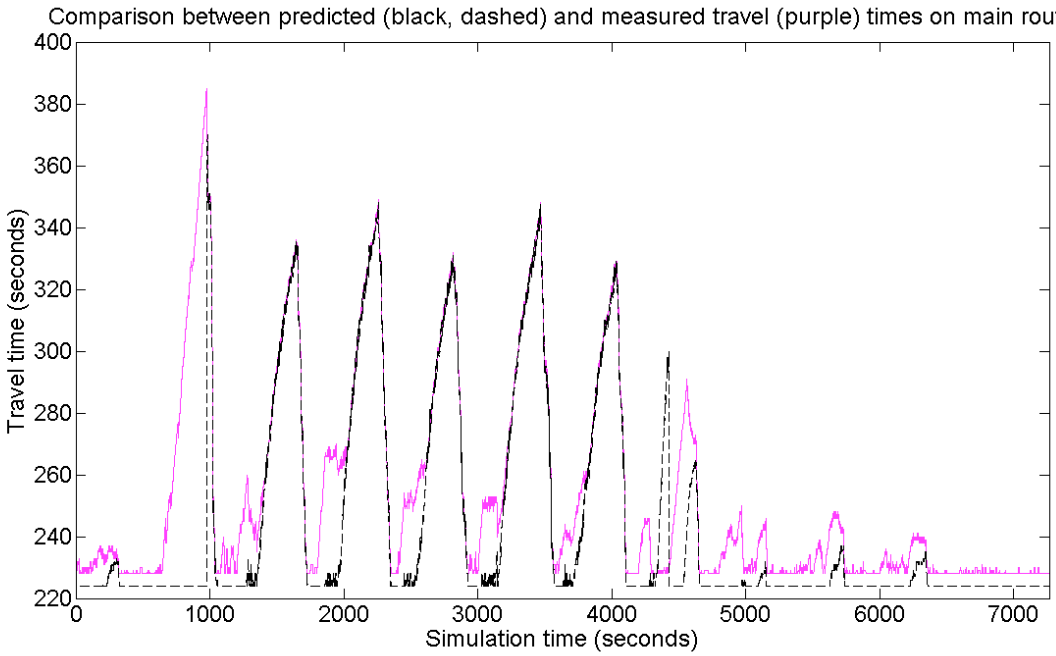

Figure 16: Main route predictions with basic SB-control of pulsating traffic.

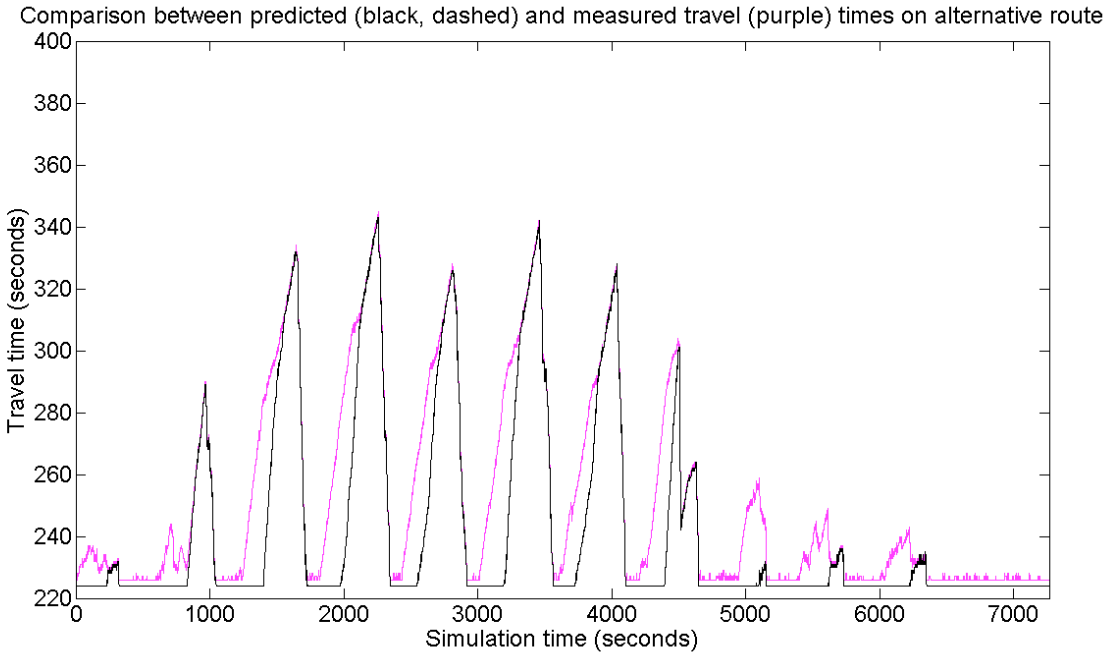

Figure 17: Alternative route predictions with basic SB-control of pulsating traffic.

In figures 16 and 17 above, we can see how the travel times increases every time a pulse of traffic comes on the route. It is obvious that the SB model is too late, predicting too low when a pulse comes. In these moments the traffic situation looks something like in figure 14.
Now, let us see what the predictions look like using the Distribution model.

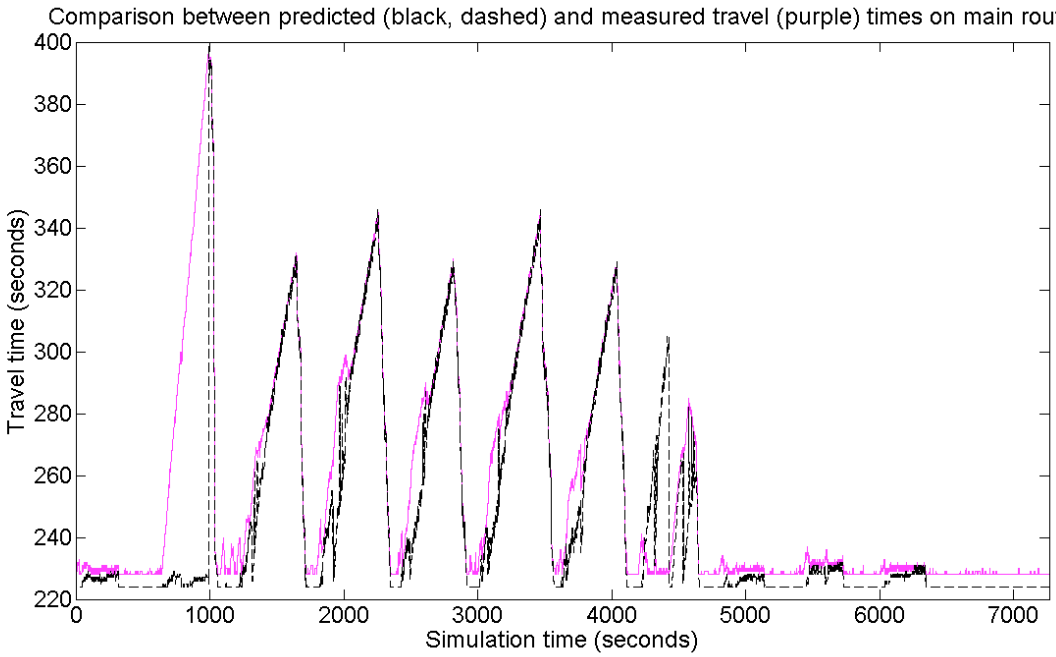

Figure 18: Alternative route predictions using Distribution Model on pulsating traffic.

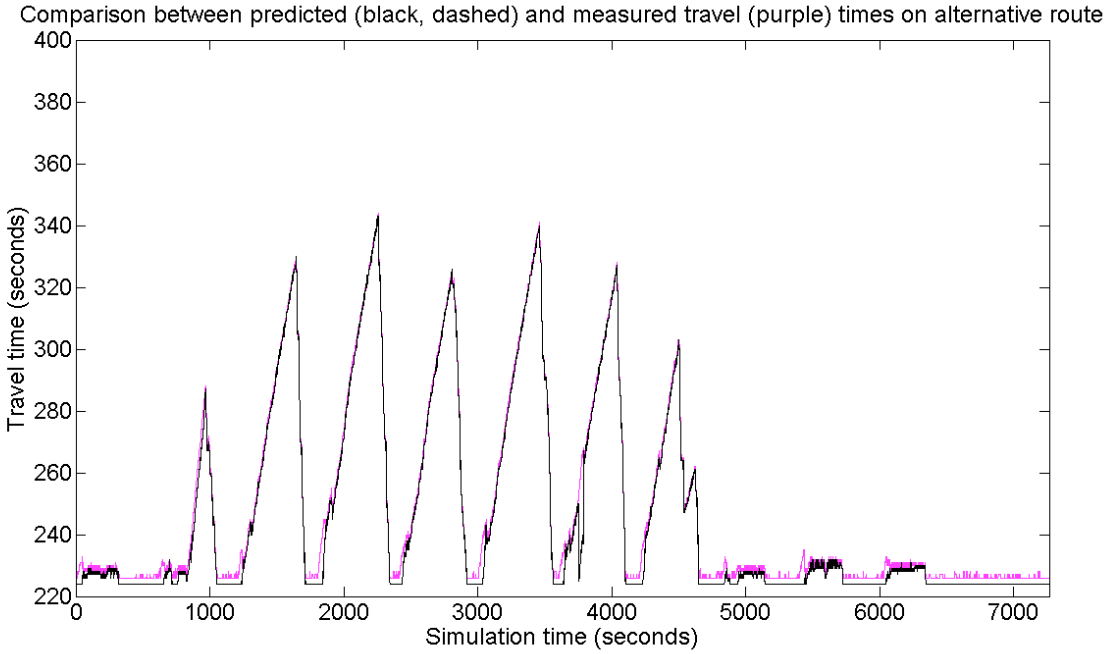

Figure 19: Main route predictions using Distribution Model on pulsating traffic.

The results are good. This model predicts the travel times much better. The prediction errors are reduced on the main route and virtually eliminated on the alternative route. As shown by the figure 18 and figure 19 and in the evaluation table below, adding the distribution-check increased the fit slightly on the main route, and considerably on the alternative route. The conclusion is that with this feature in the prediction model pulsating traffic patterns will be handled better. And more reliable information about the travel times make way for better system performance when using feedback control.

Table 2: Evaluation table for the scenario with pulsating traffic.

	Normal Case	Accident No Control	Single Bottleneck Model	Distribution Model
Main route:				
TT	233	683	253	254
Agents	3998	3998	3354	3352
Fit	-	-	95.6	96.1
Alternative route:				
TT	232	232	259	255
Agents	3998	3998	4642	4644
Fit	-	-	96.3	99.1
Total				
Std. Nash deviation	7.2	425.8	19.4	21.9
Average TT	232	539	257	255

6.4 Evaluation: Travel time improvements

More accurate travel time prediction means a more adequate control signal which means decreased travel times for the agents – but only if there is potential for more effective route guidance in the system. So is the system performance improved when the distribution check is added?

According to table 2, the distribution check improves the average travel time, but very slightly. The Nash deviation is a little bit worsened. The most probable reason for why the travel time only decrease by two seconds is that the usage of the routes' capacities are already close to the optimum. When a pulse of heavy traffic comes, the travel times increases on both routes, as shown in figures 16 and 17. But even with a standard SB-predicted output, the controller makes the best of the situation by letting the total travel times increase equally on both routes simultaneously. On the other hand, if the pulses would be only one of either route, the distribution check would detect them and compensate for the increased travel time on that route by redirecting a proportion of the traffic into the other route, whose capacity were not used fully. To create such a scenario in the test network, the agents in the pulses must not be object to route control, but must travel according to their own plan. Such agents are for example those who enter or leave the route between the sign and destination link. Then we would have a scenario in which the traffic flows cannot be regulated completely by the route guidance system.

The reason why the Nash values are worsened is probably the square factor of the standard Nash deviation calculations. As we can see in figure 20 below, the Nash Times for the Distribution Model has more peaks than the Single Bottleneck Model. One important thing to keep in mind is that the bang-bang controller does not care about the magnitude of the Nash difference; it works exclusively according to the sign of the difference. This means that improved model fit values does not necessarily imply improved travel times – the control signal, i.e. the guiding signal to the agents might be exactly the same. It should also be mentioned that small differences such as a few seconds are hard to analyze, since there might be micro queues or other noise affecting the difference between the models to a large extent. It is undeniable though, that better fit values would be statistically beneficial for the system in the long run.

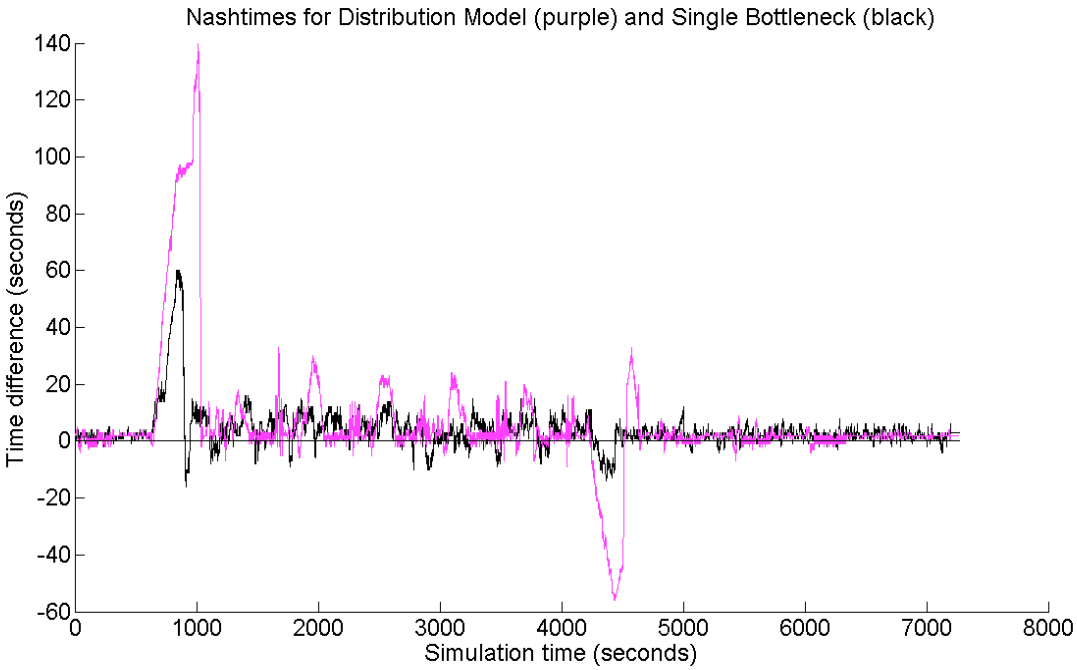

Figure 20: Nash time plot for the distribution model on a scenario with pulsating traffic.

7. Disturbance model

Inflows and outflows of vehicles on a route can alter the traffic situation so that the travel time predicted with the regular SB model increases or decreases. An outflow of agents means that a guided vehicle might experience a travel time shorter than the predicted, whereas an inflow can make the route slower than predicted. If the majority of the agents on the route do not enter via the sign link (the link where the guidance is given), then the regular SB prediction will be based on very inadequate information. In order to compensate for such errors, the in- and outflows must be taken into account when counting the number of cars on the route.

The non-deterministic character of such in and outflows constitute a fundamental problem since they are immeasurable during the prediction time and we therefore cannot know for sure how many vehicles that actually have to pass the bottleneck before the one being guided. The information at hand when giving the guidance is how many cars that is presently on the links, and present and historical inflow and outflow data.

7.1 The Disturbance model

Our approach to simulate a continuous in- or outflow is to measure the flow, and multiply that with the time it takes for the guided vehicle to get to the intersection where the in- or outflows take place. When this is done for all in- and outlinks, we calculate the total net change of agents caused by in- and outflows. The time factor is needed for calculating the number of agents that go on or off the route ahead of the agent now being guided, as a flow just before the bottleneck will result in a greater net change than a flow just after the sign link.

The true travel time to an intersection is obviously not known beforehand since there might be a queue starting at the bottleneck and stretching passed the intersection. However, to predict the queue length correctly, the net flows must be taken into account, which puts us in a complex situation. To keep things simple and avoid complex calculations, we decided to use the free speed travel time up to the intersections when calculating the net flows. The free speed travel time will indeed be the correct simulation time to use when there is no queue stretching back passed the intersection. As long as demand is not extremely high and the guiding works well, this is a reasonable approximation. In any case, the model will always compensate for additional traffic to a certain extent. When calculating the average flow values we use the free speed travel time on the route as time horizon.

Multiplying the additional flows with the travel time to each intersection, a net number of agents will be predicted to go off, or on the route ahead of the guided vehicle and thus the additional flows can be compensated for in the travel time prediction. The longer the queue gets, i.e. the more intersections the queue stretches passed, the worse the approximation will be. This is both because the travel time to the intersections will change and because the intersection dynamics will change when two flows interact, e.g. vehicles have to wait for each other.

Mathematical description

First, some notations are needed:

x_i number of agents on link i

f_i	flow to link *i* (positive value means inflow)
tt_j	free speed travel time for link *j*
$TT_{fs.}$	free speed travel time for a route; Link 0 to link n
b	bottleneck index
n	index of last link
c_b	bottleneck capacity (vehicles/second)

The total number of agents that have to pass the bottleneck before the guided vehicle, is given by the following formula:

$$X_{tot} = \sum_{i=0}^{b} x_i + \sum_{i=0}^{b} \left(f_i \cdot \sum_{j=0}^{i} tt_j \right)$$

This expresses the agents currently on the link (the first term) and the agents coming onto or going off the link calculated by the flows and the (free speed) travel time to the respective nodes (the second term). The queuing travel time is then:

$$TT_{queue} = \frac{X_{tot}}{c_b} + \sum_{i=b}^{n} tt_i$$

At last, this is compared with the free speed travel time for the route and the biggest value is taken as the prediction.

$$TT = \max\left(TT_{fs}; TT_{queue}\right)$$

7.2 Evaluation

To evaluate the Disturbance model, a new population was created that included an extra inflow to the main route before the accident location.[4] Vehicles coming onto the route there are not controlled by the route guidance but have to be compensated for in the model. One could also imagine a test scenario where there is outflow instead of an inflow. In analogy with the results presented in this section, an outflow would cause the regular SB model to make too high estimates of the travel time, and the disturbance model would compensate for it in the same manner as for the inflow. The inflow and the outflow case are theoretically similar and we will here present only the results from the inflow scenario, which in contrast to the outflow results in more congestion. As mentioned earlier, the test network was designed for the model development, while the Berlin network is used to evaluate the model. To evaluate the predictive strength of the model, the basic Single Bottleneck model is first run on the same population and the results are shown in figure 21.

[4] The test population is called PlansInflow. It is like PlansNormal but with an additional inflow on link 31 of 500 agents per hour.

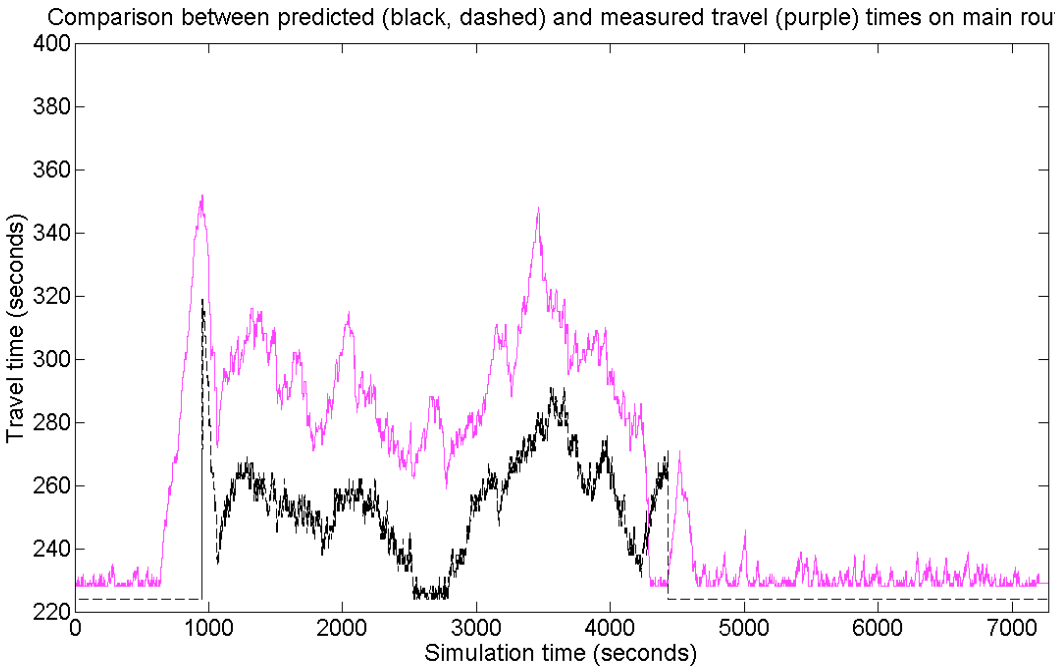

Figure 21: Measured and SB-predicted travel times on the main route when there is an additional inflow.

It is obvious that the standard SB model underestimates the travel times since it does not know about the extra inflow that must pass the accident before the guided car. With the disturbance compensation switched on the predictions improve, as seen in figure 22.

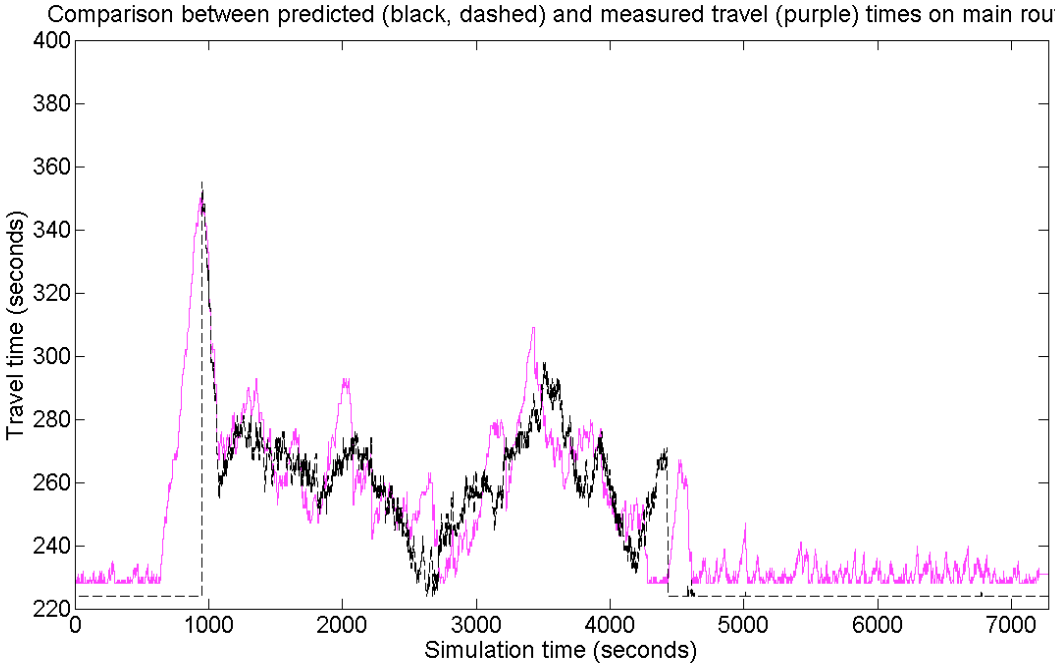

Figure 22: Measured and Disturbance-predicted travel times on the main route when there is an additional inflow.

Table 1. Evaluation table for disturbance model. Population with extra inflow from link 31 on the test network.

	Normal Case	Accident, No Control	Single Bottleneck Model	Disturbance Model
Main route:				
TT	231	981	253	244
Agents	4000	4000	3002	2968
Fit	-	-	91.3	96.1
Alternative route:				
TT	227	228	242	245
Agents	3999	3999	4997	5031
Fit	-	-	98.9	99.1
Total				
Std. Nash deviation	5.7	839.2	33.6	20.7
Average TT	229	605	246	245

The model compensates for the inflow and improves the fit value from 91.3 to 96.1 on the main route (see table 3). The average travel time is not improved significantly. The reason is probably that the network is used in a close to optimal way, meaning that the capacities of the routes are used fully. Nash deviation, however, is reduced by 40 percent, which shows that we are much closer to the control goal than when the standard SB model supplies the input. This is also reflected by the travel time averages, which are almost identical when using the Disturbance model. The improvement in Nash times is clearly displayed in figure 23, where the Nash times for the Distribution Model and the Single Bottleneck Model are plotted simultaneously.

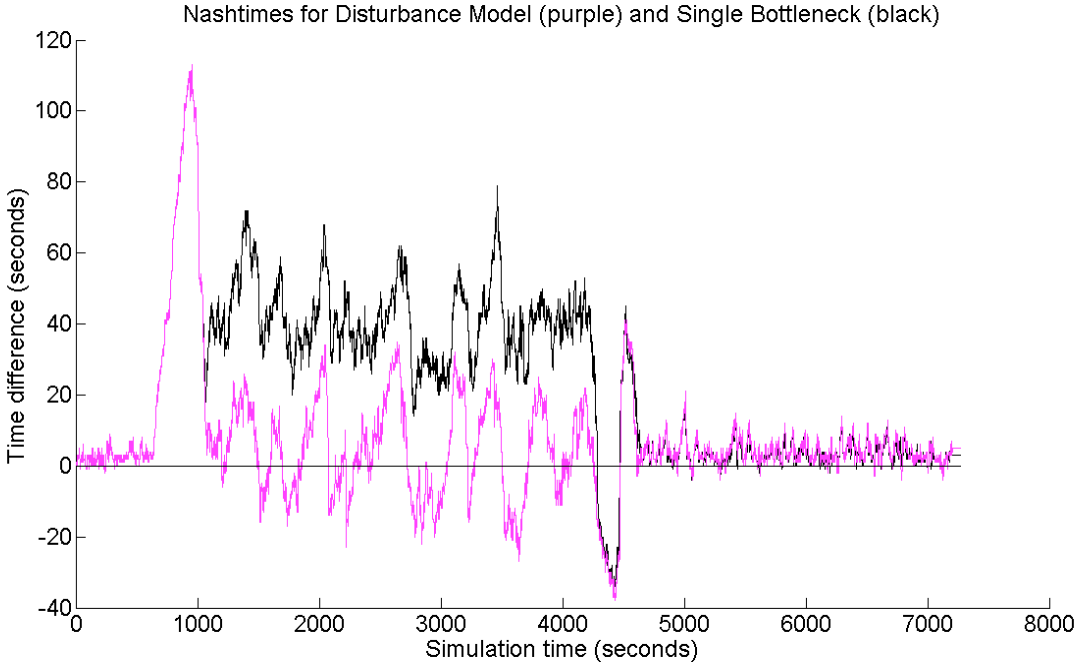

Figure 23: Nash times for the disturbance and the regular single bottleneck model, respectively.

8. Incident detection

The prediction approaches described use information about the accident; its location, its amplitude, and the time when it occurs. A traffic manager, who identifies the accident and reports its location and capacity reduction, can supply this information. This manual way of incident identification that is used today (for example in Berlin) is dependent of a fast and effective infrastructure that uses both human and technical resources. We have implemented a functionality in our prediction models that eliminates the need of manual detection of accidents and estimates capacity reductions. In turn it needs to know measured travel times to detect an accident on a link – a kind of information easily available with COOPERS-technology.

8.1 The detection model

To detect an accident, we must continuously search the route for congestions. When there is a queue the travel time on a link is unusually high. The algorithm starts in the end of the route, checking if the last agent leaving a link had a measured travel time longer than the free speed travel time. An unusually slow link means that a bottleneck is found. The measured outflow on the detected bottleneck link is then used for the travel time prediction, instead of the normal capacity of the link. Just like the incident detection, the measuring of flows uses information about how often agents leave a link. The procedure is described by the following pseudo code.

Mathematical description

The notations used in the expressions are described below. The sums' index refers to the sequence of links on the route, where $i = 0$ refers the first link on the route.

y_i	observed travel time on link i
t_{IQT}	ignored queuing time on a link
tt_i	free speed travel time for link i
x_i	number of agents on link i
b	index of the bottleneck link (the link where the accident has occurred)
n	index of the last link on the route
lt_j	registered leave time for the j last agent
f_b	observed flow through bottleneck (vehicles/second)

To identify the bottleneck we check the following inequality for all links on the route, starting in the end of the route and going upstream.

$$y_i > tt_i + t_{IQT}$$

If the measured travel time on a link is longer than the free speed travel time + t_{IQT} seconds, the model has found a capacity reduction. There can be congested links upstream of the detected bottleneck link – such queues are assumed to be due to spill back from the queue at the bottleneck.

It is normal that agents take a little longer time than the free speed travel time to travel a link, also when there is full capacity. When the traffic density is high, but still under congestion level, the agents travel closely and sometime have to wait for each other or lower their speed a little. The constant t_{IQT} makes the detection less sensitive to this kind

of noise. In the test network we ignore additional travel time up to 20 seconds.[5] The flow out from the link is then calculated based on the rate, with which the *r* last agents left the link:

$$f_b = \frac{\sum_{j=1}^{r} lt_j}{r}$$

By default the last 20 measurements are used. A higher number means a higher historical resolution with a more exact value, but at the same time a greater momentum and slower reflection of changes in the flow. After the bottleneck is found, the travel time is predicted as usual:

$$TT_{queue} = \frac{\sum_{i=0}^{b} x_i}{f_b} + \sum_{i=b+1}^{n} tt_i$$

$$TT = \max\left(TT_{fs}; TT_{queue}\right)$$

$$\hat{y} = TT_{main} - TT_{alt}$$

In a real world implementation the supply of travel time data for all links on the routes would mean either a quite extensive infrastructure of sensors or COOPERS-devices sending this data. However, for a sufficiently reliable measurement of links flows, it would be enough with a not too small proportion of the agents having a COOPERS-device installed.

8.2 Evaluation

In contrast to when evaluating the prediction models described previously, one could expect neither the prediction ability nor the travel times to improve in this case. After all it is a handicap not having immediate and totally reliable information about the accident. But on the other hand it is not reasonable to assume perfect information about accidents, as the other models do.

[5] We started out having a factor $y_i > tt_i \cdot x_{IQT}$ where $x_{IQT} = 1.20$ would imply that travel times up to 20 percent longer than the free speed time would be ignored. On very short links (e.g. link 14 where the accident occurs) micro-queues of less than a second was considered a sign of congestion, and for this reason the factor was replaced by a constant.

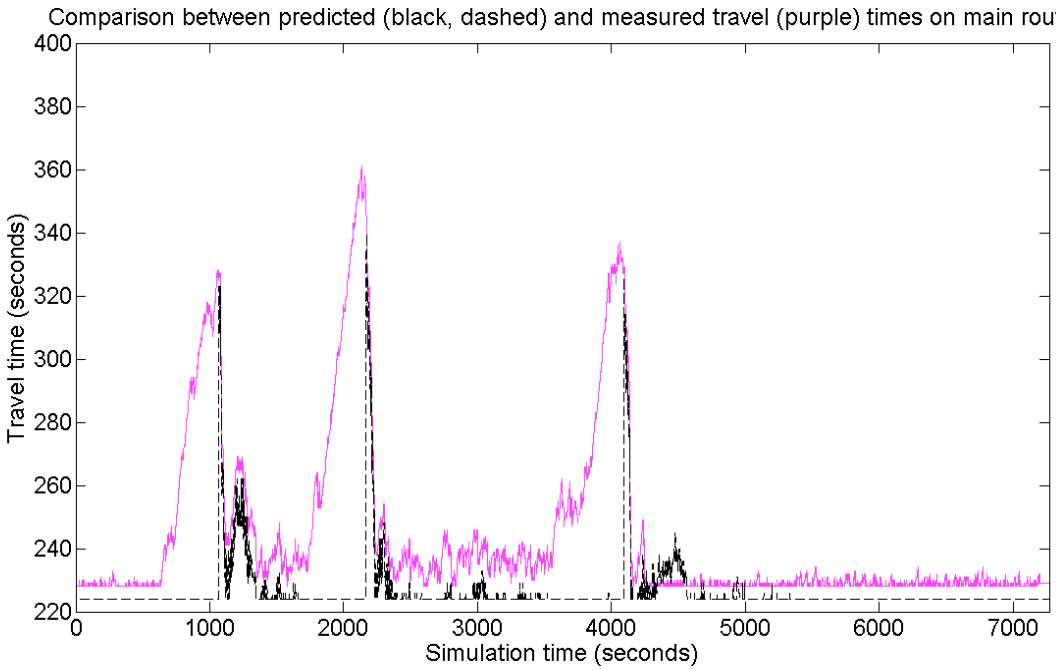

Figure 24: Travel times on main route with automatic incident detection.

There are three obvious places with prediction errors in the travel time plot in figure 24. When an accident occurs it takes some time before it will be detected. The constant t_{IQT} = 20 seconds corresponds to a small part of the total time delay. It can take up to ten minutes before the guidance react to the accident in the test network, which corresponds to the point where the observed travel times start to rise, to when the queue is reflected in the predicted travel times. These delays can be seen on three places in figure 24. When the bottleneck is detected, agents on the sign link will be getting a control signal that reflects the capacity reduction and vehicles will immediately be guided over to the other route. The long travel times in the plot represents agents who entered the main route before the accident occurred, and hence could not be helped by the route control. They will have to continue travelling the route that they started, even though an incident was detected by the traffic management system. This error is also to be seen in the main route travel time plots of the other models, but only once. In figure 24 the pattern appears three times. The reason is that the model "forgets" the accident. When an incident is detected the model uses this link as bottleneck for the travel time predictions for a certain period defined by a user parameter. The accident must be forgotten – or else the model will use the outflow as capacity even after the road is restored to normal, and it would not be able to detect other incidents. In the simulation that generated the plot above the model was set to remember the accident for 30 minutes and then again use the normal capacity. If the bottleneck had to be detected more often, there would be more large peaks than three in the graph. In a real traffic network, the choice of the reset time would not have such a big impact. When there is an incident on the main route – which is normally the fastest route – this route will always have a queue when the route guidance is active to keep the system at Nash equilibrium. This means that there is no risk that the incident is forgotten unintentionally due to absence of travel time observations. Therefore, in the Berlin scenario evaluated later in this report, the model is set to forget the accident every second – there will still be measured travel times of queuing agents.

Another and more effective way of handling the forgetting problem is to have a traffic manager that reset the detection function when the road is cleared and its capacity is restored to normal. In contrast to when incidents occur, the restoration does not happen unexpectedly; hence resetting the model is trivial to do manually.

Apart from that the Incident detection model is a little slower than the regular SB when the accident starts and is forgotten, the predicted graph is also less smooth. This is due to inaccuracies in the flow measurements. If the average measured flow for the last *n* cars is used, every new car will have an influence on the flow measurements of *1/n* parts. Thus, one has to weigh the disadvantages of noisy behaviour with the advantages of measuring only the most recent cars.

	Single Bottleneck Model	SB with Incident Detection
Main route:		
TT	232	243
Agents	3405	3321
Fit	96.8	93.6
Alternative route:		
TT	229	230
Agents	4594	4678
Fit	98.3	98.0
Total		
Std. Nash deviation	8.6	31.2
Average TT	230	236

Table 3: Evaluation table for the Incident detection model. [6]

Table 4 informs that the fit value is 3 percent lower for the main route with the incident detection. This prediction error is a result of the extended time delay discussed above, and it also makes us lead a few more agents into the accident route before the bottleneck is detected, which is reflected by in the increase of average travel time on the main route.

[6] The normal test population is used: 2000 agents/hour on each route. Accident: 50 percent capacity on link 14 from 7:15 to 8:15.

9. Multi Bottleneck Model

When running the Berlin scenarios, we realized that the travel times were predicted well when there was only one queue. When more than one queue appeared, due to inflows and natural bottlenecks in the network, our models predicted too low, at least if the first queue is the more severe one. When the last queue on a route is the more severe, then its queuing time will swallow the other queuing time fully or to some extent. For developing a model that could handle this, the situation below was created on the test network. An accident is set on the small link 11, causing a queue there. Another accident is set on link 14, and due to the inflow from link 31, a queue builds there as well.

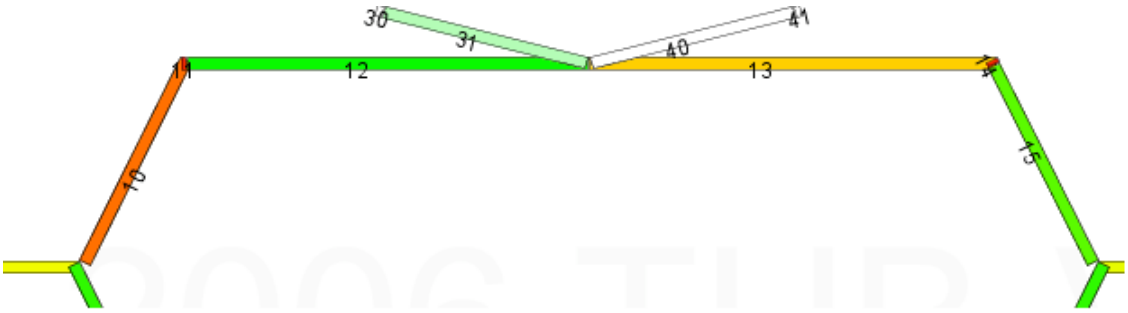

Figure 25: Scenario for developing the Multi Bottleneck Model. Accidents are reducing the capacity on link 14 to 77 percent and on link 11 to 63 percent. Test population: PlansInflow (as PlansNormal but with additional inflow on link 31 of 500 agents per hour).

It can be seen from figure 25 that the queue on link number 11 and number 10 is denser than the one on link number 14 and 13. Green colour represents free flow conditions and then there is a scale from yellow to red representing worsened conditions. Thus should all previous models underestimate the travel times on this route since only one of the bottlenecks is considered (link 14).

9.1 The Multi Bottleneck Model

The Multi Bottleneck Model divides the routes into segments, where every segment has its bottleneck. First, the incident detection procedure is done for the whole route and all bottleneck capacities where a queue has built up are stored. Then all flows are stored; flows on and off the route, but also flows on the route, i.e. from one segment to the next. The travel time for each segment is done as in the previous models; the only function not implemented is the distribution check (the problem with non-homogeneous traffic distribution is reduced partly by the route segmentation performed by Multi Bottleneck model). For the first segment it is done as before, but for the following segments, the number of agents ahead of the guiding object when reaching that segment must be simulated by assuming the flows to be constant during the time it takes to get there. Simulating/predicting the number of agents on every segment by the time the guided agent gets there allows the model to treat all segments with the same queue check approach. If there is no bottleneck on the last link of a route, all the links after the last bottleneck will be treated as free speed parts as in the previous models.

Mathematical description

The main calculation is to determine the total number of agents, X_{tot}^{k} going through the bottleneck on segment *k* before the agent currently being guided. This can be expressed mathematically as:

$$X_{tot}^{k} = \sum_{i=s}^{b} x_i + \left(F_s - F_b\right) \cdot \sum_{j=1}^{k-1} STT_j + \sum_{i=s}^{b} \left(f_i \cdot \left(\sum_{j=1}^{k-1} STT_j + \sum_{m=s}^{i} tt_m \right) \right)$$

where the following notation is used:

x_i	number of agents on link i at t_0 (start of simulation)
F_s	Intra flow onto link s at t_0
F_b	Intra flow out from link b at t_0
STT_j	travel time for segment j
f_i	in or outgoing additional flows to/from link i
tt_m	free speed travel time for link m
b	bottleneck link index of the segment
s	first link of the segment
k	index of the calculated segment
f_{bk}	observed flow through bottleneck on segment k

The first term expresses the agents on the link at time t_0. The second term represents the agents coming onto the segment from the previous segment subtracted with the agents leaving the segment during the time it takes for the guided agent to get to the beginning of the segment. The last term is the disturbance compensation for agents coming onto the route from additional links. Those flows are assumed to be constant during the time it takes to get to the intersections, counted as the predicted time to the start of the segment plus the free speed time to that certain intersection (node).

After that calculation, the regular queuing travel time, TT_{queue}, is calculated, i.e. the simulated number of agents on the segment is divided by the segment's bottleneck capacity. At last, this is compared with the free speed travel time; $TT_{f.s.}$ for the route and the biggest value is taken as the prediction.

$$TT^{k}_{queue} = \frac{X^{k}_{tot}}{f_{bk}}$$

$$STT_k = \max\left(TT^{k}_{f.s.}; TT^{k}_{queue}\right)$$

These simulations and calculation are done for all *n* route segments, which add up to the predicted route travel time for the route. The predicted output is as usual the predicted time difference between the routes, and this value is sent to the controller that directs the agents.

$$TT_{route} = \sum_{k=0}^{n} STT_k$$

$$\hat{y} = TT_{main} - TT_{alt}$$

9.2 Evaluation

To show the predictive abilities of the Multi Bottleneck Model, we chose a scenario where the model predicts the travel times when no control is applied. This is because the model uses incident detection to discover the bottlenecks and that the incidents easily are forgotten as soon as the control is applied on the test network. This is not a problem on the Berlin scenario, as mentioned in the previous chapter. Figure 26 displays the main route predictions of the Single Bottleneck Model on this scenario. Incident detection is active, to make is more comparable with the multi bottleneck approach.

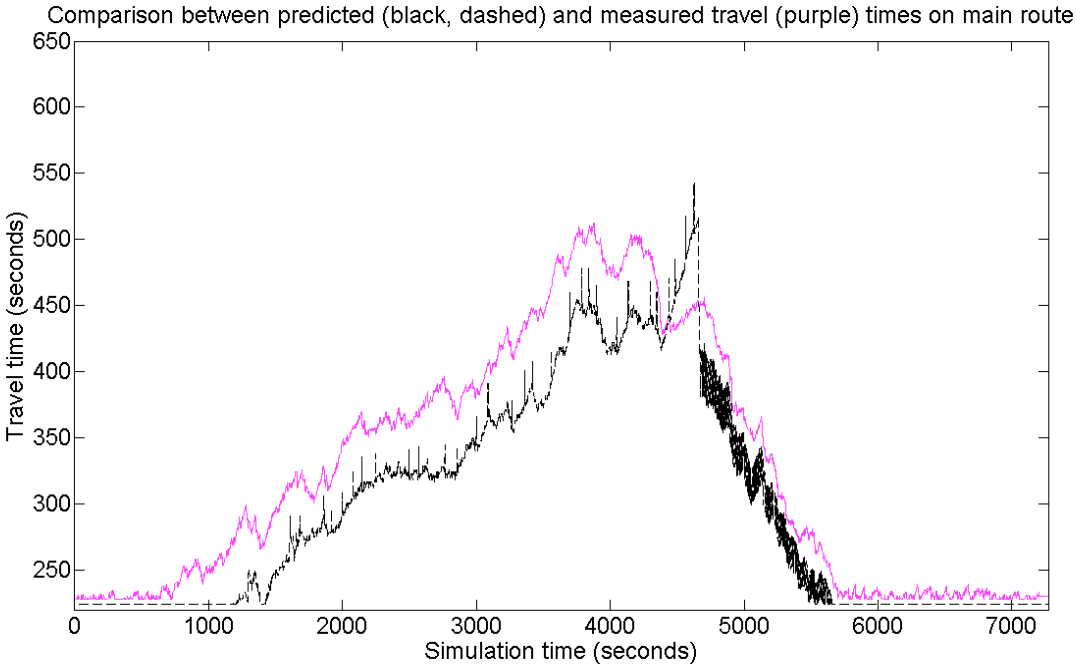

Figure 26: The regular SB with incident detection on a scenario with two queues.

The SB predicts to low in figure 26, which is expected since only one the queuing time from one of the two congestions is included in the predictions. Since the basic SB model does not handle inflow traffic at all, figure 27 shows the predictions when the disturbance compensation was activated:

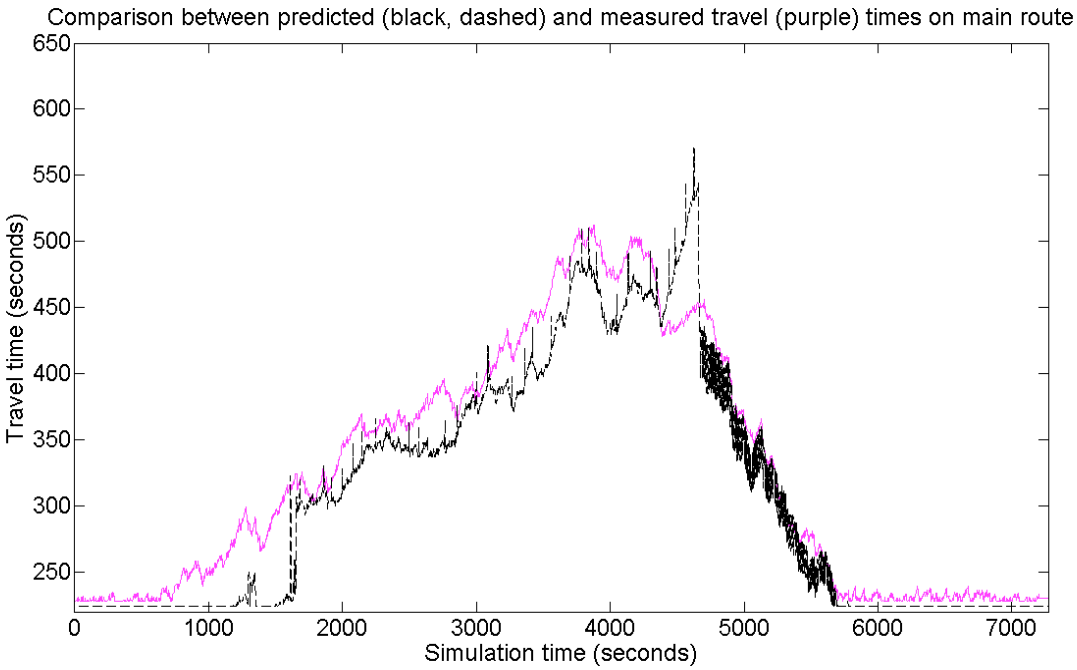

Figure 27: SB, incident detection and Disturbance compensation on a scenario with two queues.

The result is visibly better, but there is still an error, especially in the queue build-up phase, i.e. from 1000 seconds to 4500 seconds. Presumably, this is due to the limitations to one queue. Applying the Multi Bottleneck Model to the scenario, the results get as in figure 28.

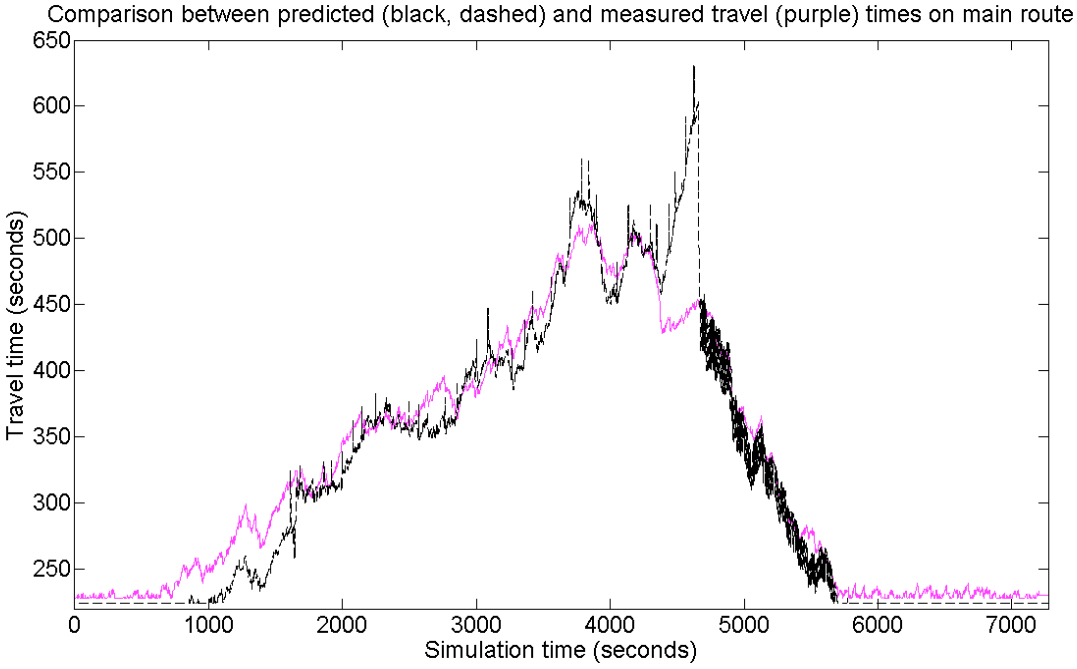

Figure 28: Multi Bottleneck model on a scenario with two queues.

Table 2. Evaluation table for multi-bottleneck prediction on the no control scenario. Prediction improvements for the main route.

	Single Bottleneck Model with Incident detection	Single Bottleneck Model with Disturbance compensation	Multi Bottleneck Model
Fit	91.2	93.6	95.4

The change in fit values between the models is significant. The result indicates that the Multi Bottleneck Model reduces prediction problems due to multiple congestions. Better prediction accuracy implies generally improved travel times, which will be evaluated in the part III of this report.

Part III - Evaluation of the Berlin scenario

10. The Berlin simulations

Choosing a location in the Berlin network in MATSim and applying the predictive models and the guidance there was the next step in evaluating the models. A Berlin location constitutes a more complex system and the scenarios presented here are more similar to real world traffic than the scenarios in the test network (which were designed for implementing the models).

10.1 Merging the models

At this point the individual versions of the single-bottleneck model have been evaluated in the test network. To make it possible to run the model using several of the extra features at the same time, the java-classes for the individual models we merged into one class. This class has final class variables specifying what functions should be active. Another important motive for merging the classes were to get rid of redundant code that were common for all classes (Much redundancy had already been eliminated earlier when the super class `AbstractControlInputImpl` had been created.) Hence we are permitted to try the Single Bottleneck model in its most complete version. There are now only two model classes, the *Single Bottleneck Model* and the *Multi Bottleneck Model*. The different features of the Single Bottleneck Model can be tested separately though, by setting user parameters that switch the different functionalities on or off.

10.2 Scoring

MATSim has a scoring function that produces a score value in Euros for every agent according to how well the activities of the day were fulfilled. The scoring is calculated as a fitness function which is the sum of the utilities of the *activities* and the (dis)utilities of the *travel*. The utility of an *activity* depends on the duration of the activity, waiting time, late arrivals and early departures. The (dis)utility of the travel is only dependent on the travel time (Charypar, 2004), There is of course, a certain mathematical complexity behind this, but to simplify, one can conclude that longer travel times, resulting in for example coming to work too late, produces a lower score. For example, one hour of travel time lowers the score by 12 Euro (Charypar, page 19).

Subpopulation score

The score that MATSim produces is for the entire population. Since only a very small part of the agents in the Berlin simulations are actually affected by the guidance, the changes in the score due to the guiding are relatively small. Therefore, a scoring function that filters out a subpopulation was developed.[7] The filter used in our simulations produces scores for all agents that passed both the sign link and the direction link during the whole day.

To find out how many agents have to be equipped with a COOPERS' device to get a benefit for all agents receiving the guidance we set up simulation runs with different compliance rates. E.g. a compliance rate of 50 % is equivalent with a scenario in which

[7] This was done by Grether, D., at Technische Universität Berlin.

half of the travelers possess a device. The agents equipped with a device are randomly selected in each simulation run.

10.3 Travel time evaluation

The Tegel highway connects the north-western suburbs of Berlin to the city centre and also to the ring road around Berlin (the "Berliner Ring"). An accident that reduces the highway capacity to 50 percent is simulated between 16.20 and 17.50, when many vehicles are leaving the centre and travelling north.[8] Figure 29 shows the traffic situation at 17:20 in the normal case and the accident case. In the accident case almost the whole main route is congested due to the accident, but the alternative route has unused capacity.

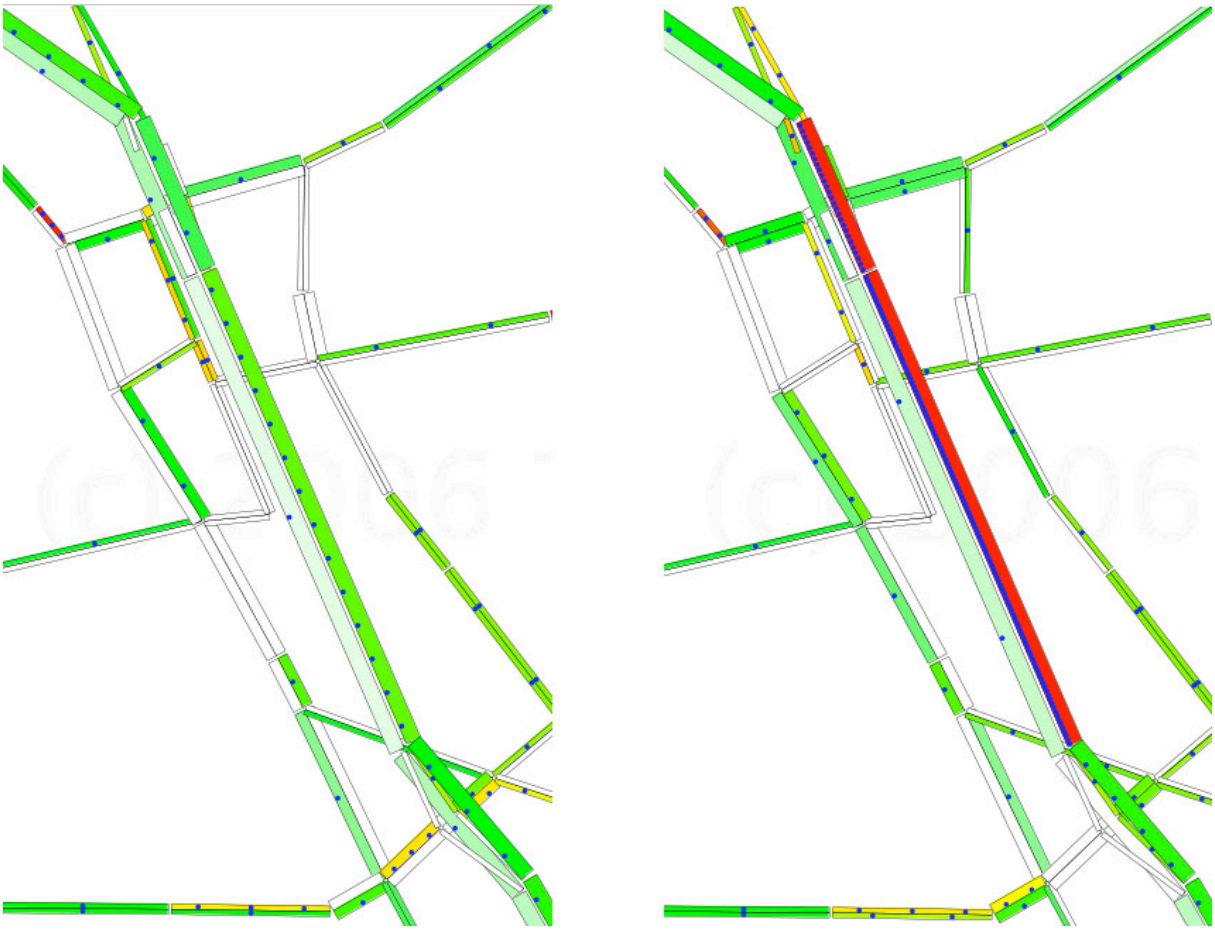

Figure 29. The no accident case (left) and the accident case (right) at 17:20. The accident causes a long queue on the highway, but the on the alternative route the demand is low.

The scenarios were simulated for a whole day. The evaluation values, except the score, are extracted only between 16.10 and 18.10, in order to examine the effect of the route guidance only around the time of the accident. The reason for the longer time after the accident is that there must be time for a queue to dissolve after the accident is gone, so that all agents that were queuing will be measured. Figure 30 shows the rising travel time caused by the accident. The average travel time for agents on both routes is

[8] A 50 percent capacity reduction can be interpreted as one lane is shut down on a two-lane highway, the other having full capacity. Another, perhaps more realistic interpretation is that one out of three lanes is shut down, leaving two passable lanes but not with free speed.

1004 seconds, compared to 367 seconds without accident. All the simulation results are presented later in this chapter in tables 5A and 5B.

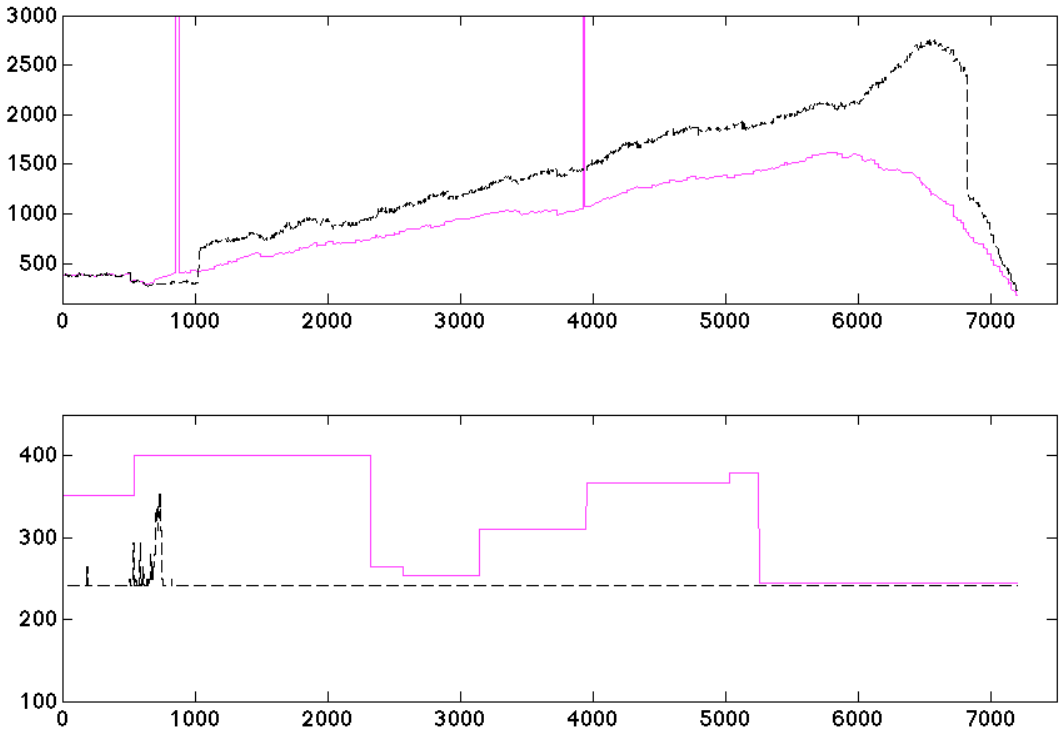

Figure 30. Travel times in the accident – no control scenario. Purple graphs are observations; black are values predicted by the Single Bottleneck Model. There are two outliers in the upper diagram.

Reactive Control

We start out applying route guidance based on observed travel times. The result is displayed in figure 31. Thanks to the feedback control, some traffic is redirected into the alternative route, which lower s the overall average travel time from 1004 to 580 seconds. (The average travel time is based on all agents passing both the sign link and the destination link, i.e. all agents that travel one of the two routes in the regulated system. Also the travel times of agents that do not comply with the advice are taken into account.)

The characteristic system behaviour that were displayed in chapter 2 can be seen in figure 31, although natural variations in the demand makes the pattern less clear in this case. For example, at time t = 4000 seconds, the traffic is guided into the main route, based on observed travel times of 400 seconds. In reality, the main route travel time is 800 seconds, which is much slower than the alternative route.

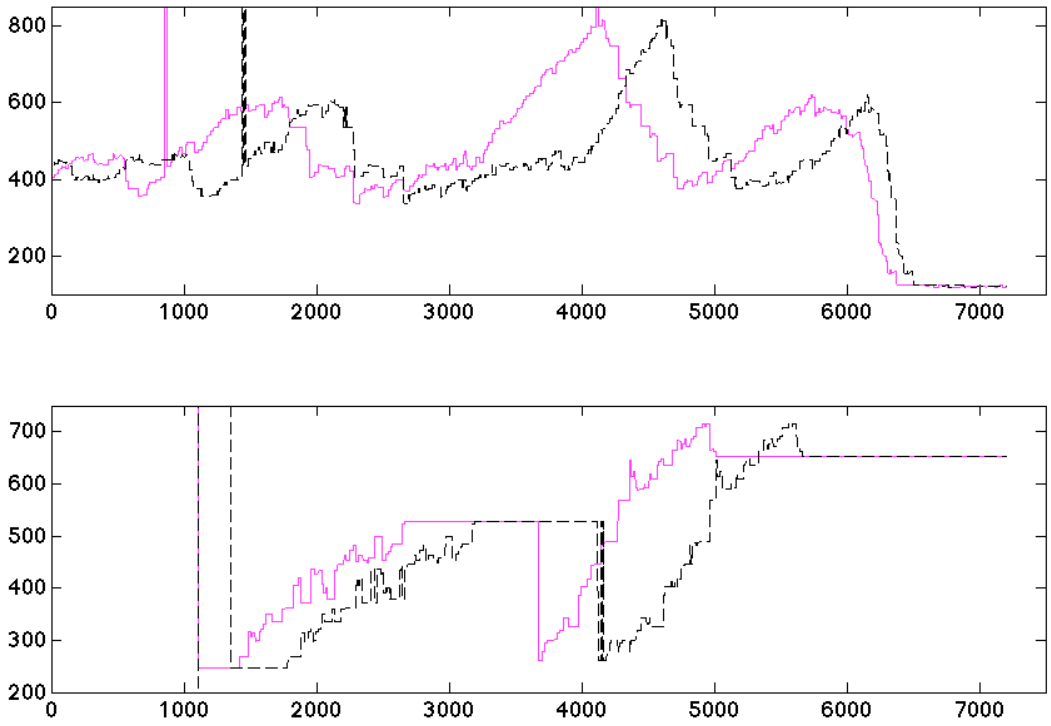

Figure 31. Route guidance based on observed travel times for the main route (above) and the alternative route (below). Purple represents the real value; black represents value used in feedback.

Basic Single Bottleneck Model

The system performance increases significantly when the guidance is based on predictive travel times. The destination link is now reached in averagely 349 seconds – even faster than in the no-accident-scenario without control (367 seconds). The traffic is distributed into the two routes, keeping the predicted travel times around 250 seconds on both routes and accordingly the predicted Nash time around zero. Figure 32 shows that the model generally predicts a little too high on the main route, but too low on the alternative route. A possible cause is the presence of several inflows on the alternative route, and one outflow on the main route. If this is the case one can expect an error reduction from disturbance compensation (evaluated below).

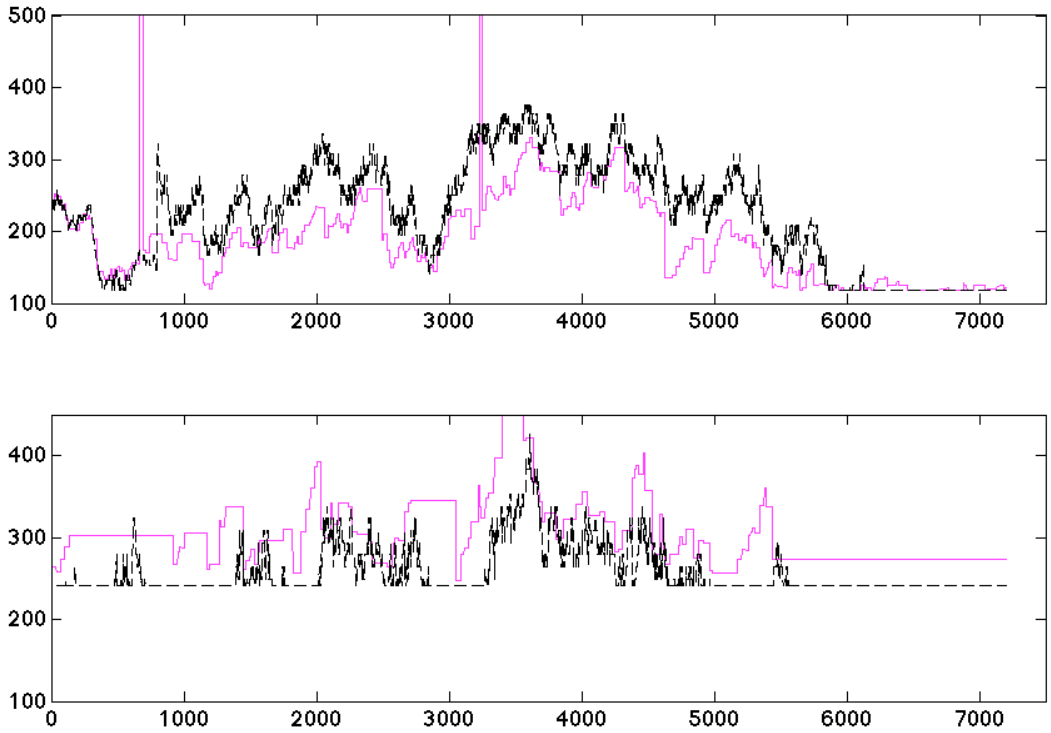

Figure 32. The Basic Single Bottleneck model. Main route (above) and the alternative route (below). Purple represents the real travel times; black represents predicted value applied in feedback.

Compared to when no control is applied, the number of agents taking the alternative route increased from 7 to 123 (note that this corresponds to a increase from 70 to 1230 agents if the whole Berlin population would be simulated). The NetVis visualization to the right in figure 33 shows mostly green or yellow links in the controlled system, indicating that the demand is on an acceptable level. The situation is similar throughout the time with reduced capacity. Without the modifications made of some capacities, the slip road leading off the highway would have caused queues.

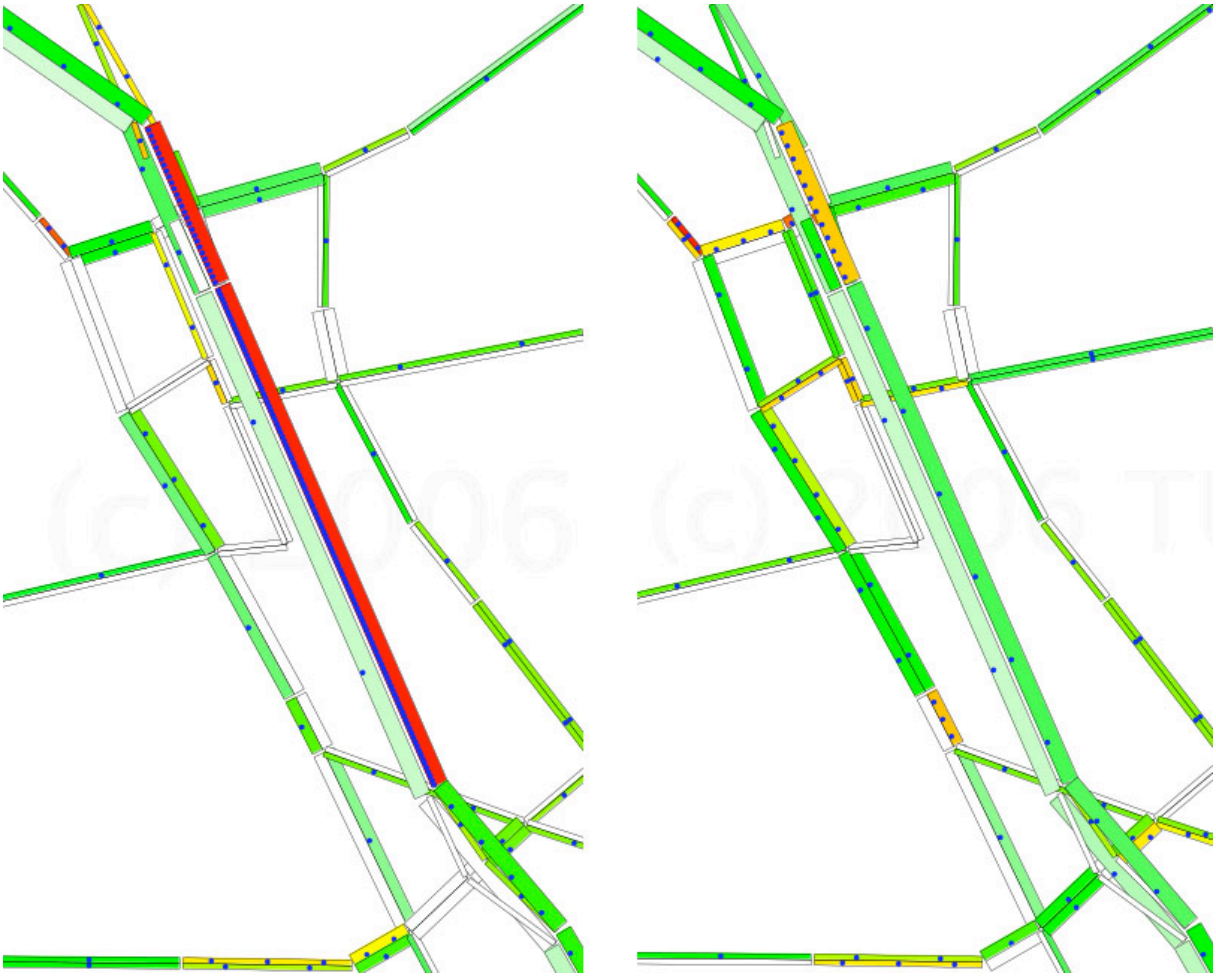

Figure 33. System at 17:20. Left: no control; right: control with Basic SB model. Compared to the open loop system, the queue on the highway is significantly reduced and the demand on the alternative route is considerably higher.

Apart from the benefits described above, the predictive feedback has positive effect also on the travel times before and after the accident. At 16:20, when the accident occurs, there is already heavy traffic on the northbound highway at Tegel. But as the traffic management system is active all day, some of the highway traffic has already been diverted onto the alternative route, which makes the initial conditions for the accident better than in the no-control scenario. Figure 34 visualizes the situation. In order to briefly estimate the benefits from predictive feedback control in ordinary traffic situation, the Basic SB model was evaluated also on a no-accident scenario, and the results are presented in the evaluation table 5A.

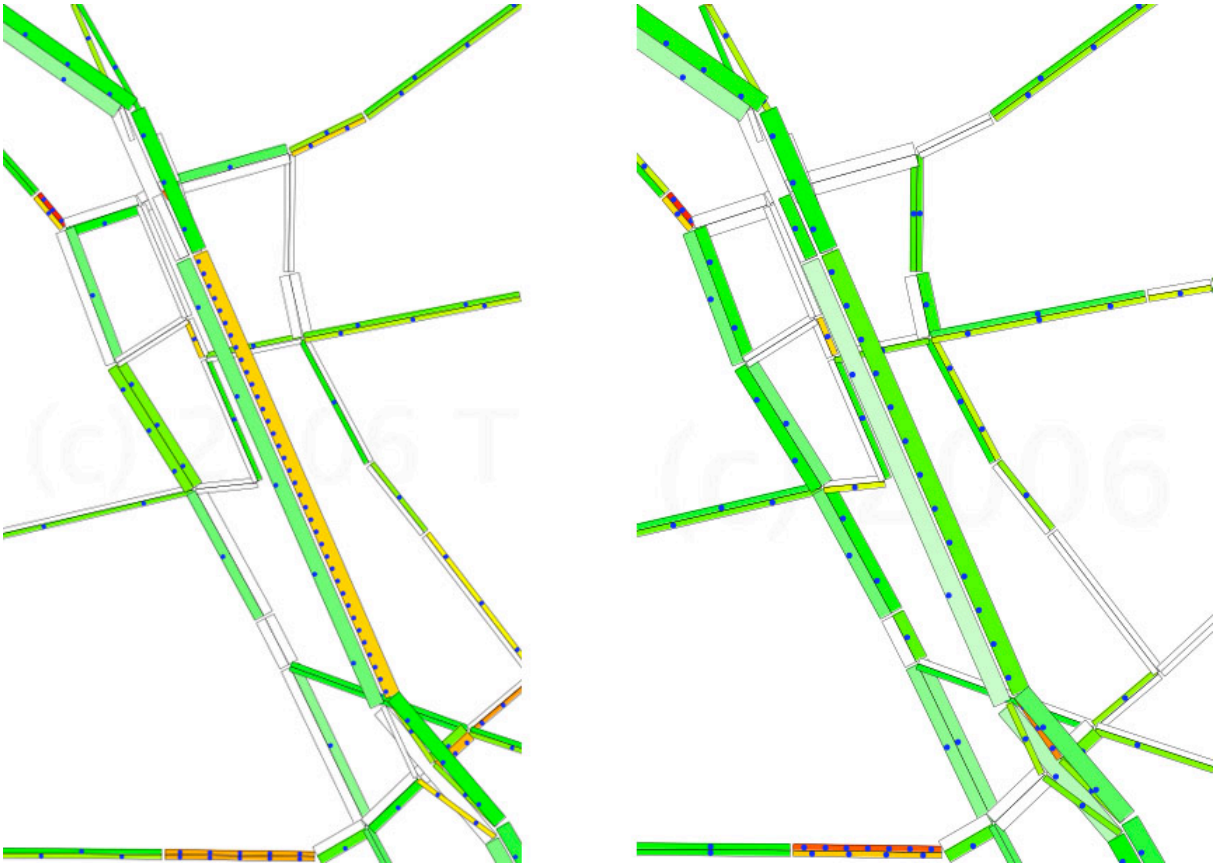

Figure 34. Traffic situation at 16:20 (just before the accident occurs). No control (left) and Basic SB Control (right).

Distribution model

Switching on the distribution functionality does not result in significantly shorter travel times (347 compared to 349 seconds without the distribution feature). Figure 35 presents the travel times. The Distribution model overestimates a little on travel times on the highway, just like the basic SB model. But the prediction errors are reduced on the alternative route, where the model fit increases from 85.0 to 90.5 percent. This contribution to the fit value comes from situations when the distribution model identifies traffic patterns on the alternative route when most of the traffic is located in the beginning of the route. The basic SB model assumption that the traffic will pass through the bottleneck in a steady pace would lead to underestimation of the travel times, but the distribution model recognizes that pattern and predicts a higher value than the free speed travel times. (See chapter 6 for a more detailed explanation of the problem with heterogeneous distribution of the traffic.)

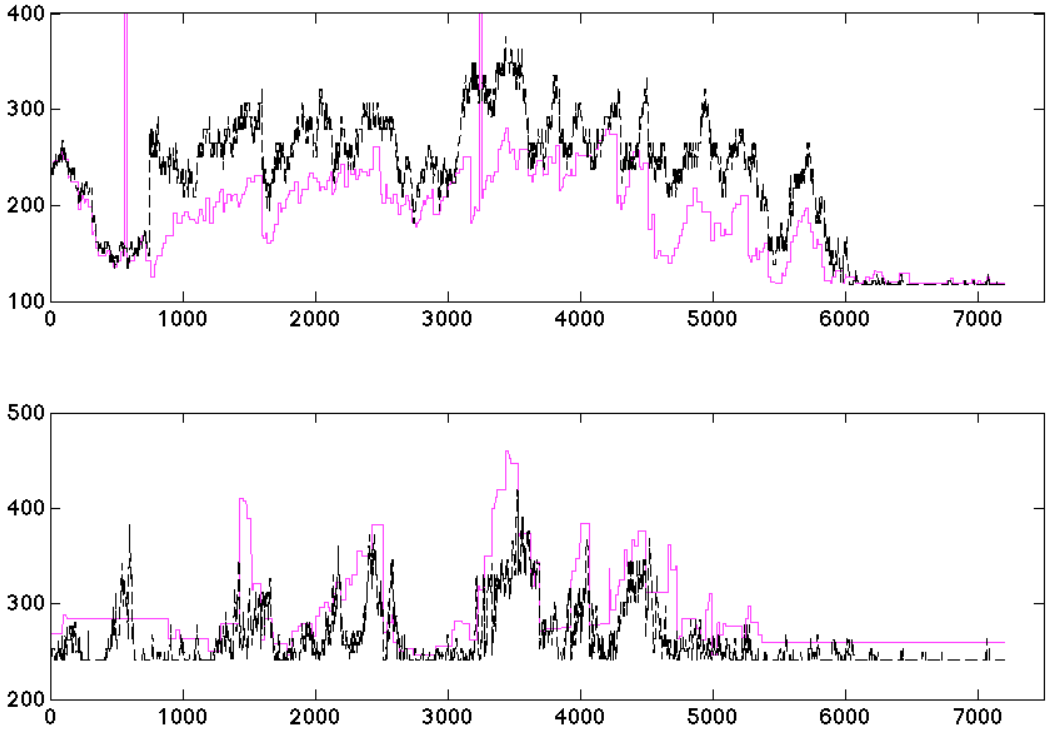

Figure 35. Distribution model. Main route (above) and the alternative route (below). Purple represents the real travel times; black represents predicted value applied in feedback.

Disturbance model

The evaluation of the basic SB model produced the hypothesis that the overestimation of travel times on the highway could possibly be due to an outflow upstream the bottleneck location. The evaluation results of the SB model with compensation of background traffic, does not confirm this hypothesis; see figure 36. The Disturbance model does no predict significantly better than the Basic SB model on neither route. In systems where there is proportionally many agents that travel only part of the route, predictions would certainly benefit from including such background traffic. The more intersections on a route, the higher is the influence on in- and outflows. The disturbance functionality should be evaluated further on another system.

In order to improve the prediction technique, the cause of the overestimation of the main route travel times in the Tegel scenario should be investigated. Useful information can possibly be derived from the fact that the models predict better when the highway capacity is not reduced. Figure 37 presents the travel times in the normal day, no accident case. The prediction errors are considerably smaller on the highway (model fit = 94.3 percent). There is a marked static error in the predictions on the alternative route, due to the insufficient basic data; only seven agents travel the complete alternative route during the time window evaluated.

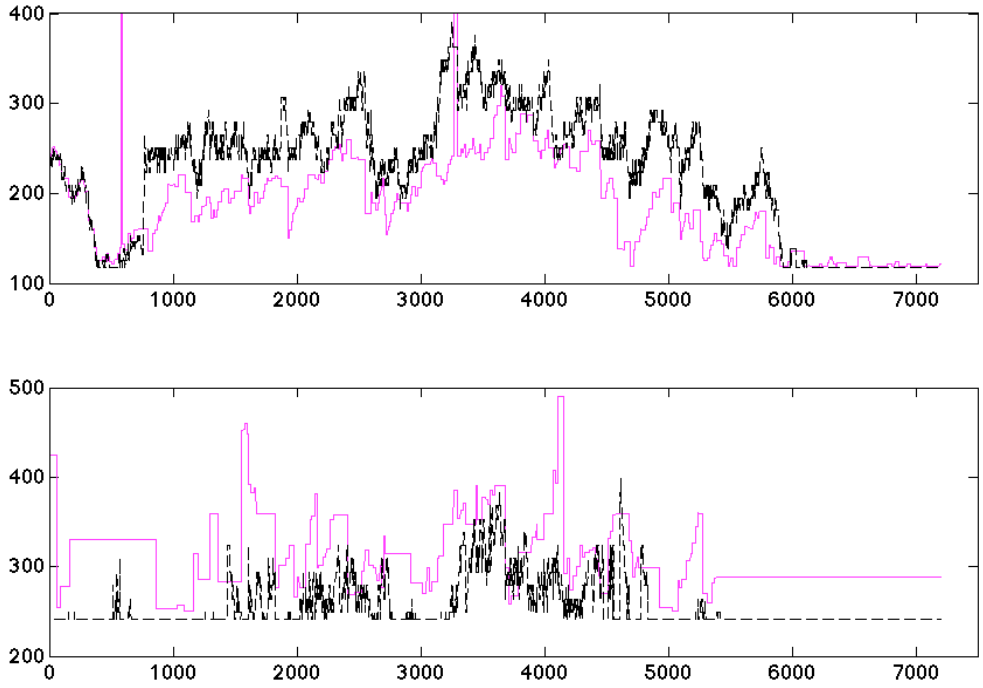

Figure 36. Disturbance model. Main route (above) and the alternative route (below). Purple represents the real travel times; black represents predicted value applied in feedback.

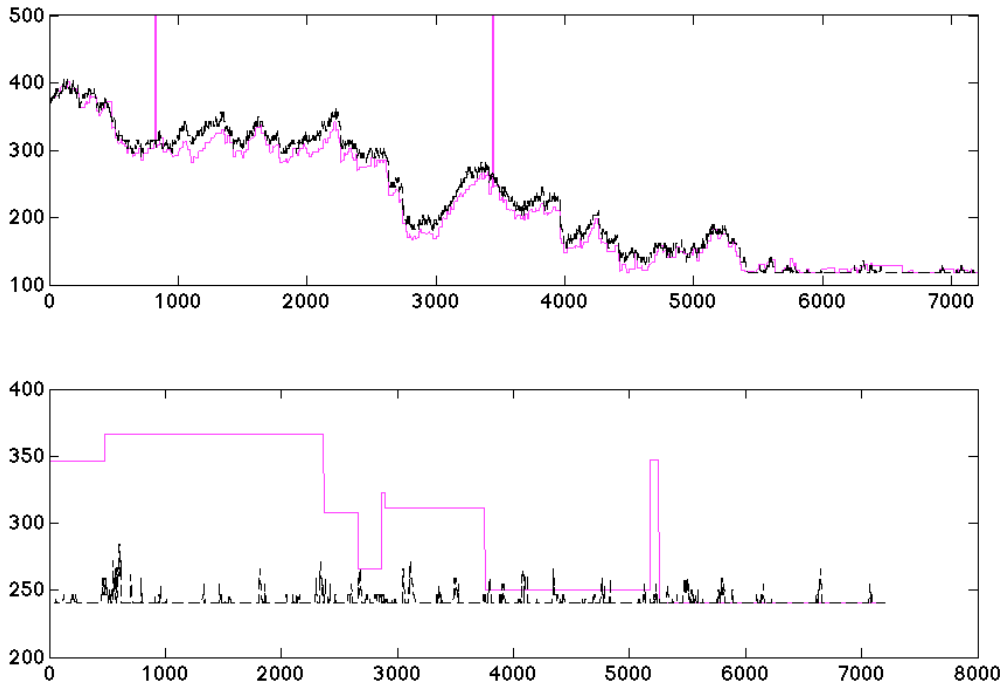

Figure 37. Prediction of the Basic SB model when the highway has normal capacity and is not subject to route guidance. Purple represents the real travel times; black represents predicted values.

Incident Detection

The Single Bottleneck model with automatic incident detection is the configuration that performs best of all models in the accident scenario, measured in terms of model fit on the highway route. This result was unexpected, as this model detect the bottleneck by measuring traffic flows, instead of beforehand information about when the accident occur and its amplitude. The highway travel times are 80 percent correct on average; the predictions are plotted in figure 38.

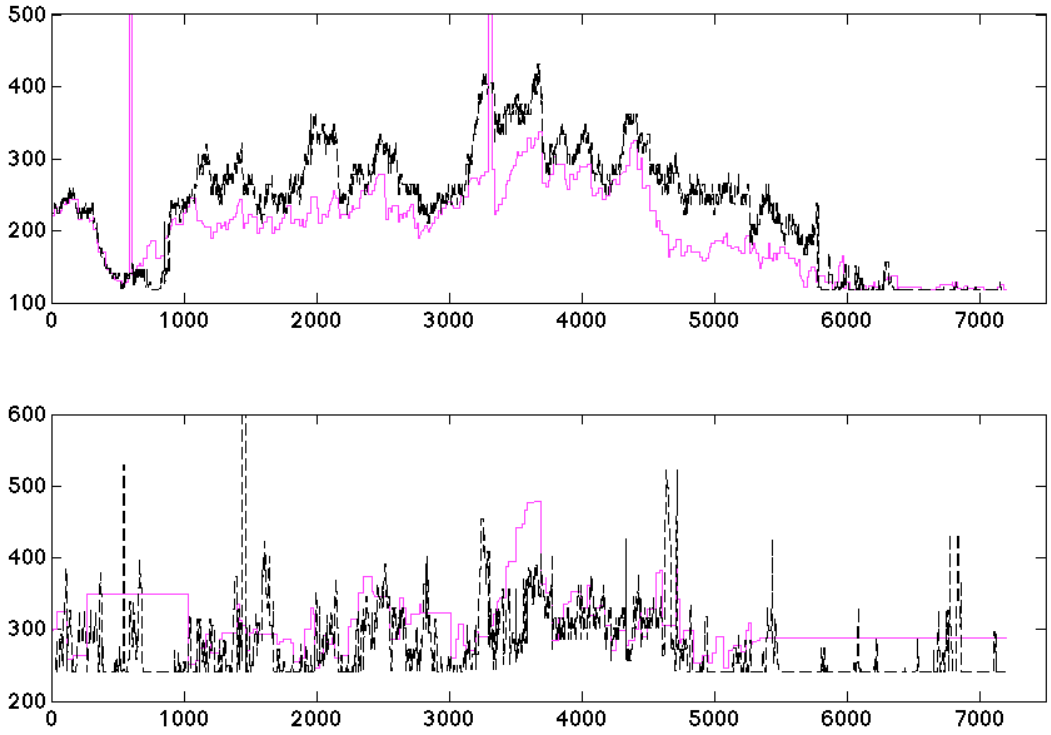

Figure 38. Incident detection. Main route (above) and the alternative route (below). Purple represents the real travel times; black represents predicted value applied in feedback.

Shifting focus to the alternative route, the predictions seem to be considerably influenced by noise. The parameter t_{IQT} (ignored queuing time) discussed in chapter 8 is set to 20 seconds, like in the evaluations in the test network. As soon as the travel time on one link is 20 seconds longer than its free speed travel time, the link is considered to have an incident (or some other kind of temporary capacity reduction). That bottleneck will be the used in the travel time prediction. Choosing the parameter value is a trade-off between sensitivity and robustness. A longer ignored queuing time would result in less noisy predictions, whereas a lower value would slower detection of bottlenecks and more free speed travel time predictions. This parameter, just like the other model parameters in the additional SB features, deserves attention in order to optimize the prediction ability.

Single Bottleneck Model with several additional features

Figure 39 shows the prediction ability of the Single Bottleneck Model when three additional features are active simultaneously; distribution check, disturbance compensation and incident detection. Of all models evaluated in this chapter, this combination is the one that is closest to reaching the control goal. Figure 40 shows the

Nash times during the evaluation period. Prediction errors are reflected by the Nash time, but the main reason for the negative Nash time average is the travel time difference between the routes. The observed Nash time used as evaluation measure here corresponds to the Nash equilibrium only partially. The Nash time approaches zero when both routes are equally fast, and the predictions are correct. But the control goal is also achieved when one route is faster, and all agents take that route. This is the case between time t = 6000 seconds and t = 7000 seconds in figure 39 and 40. To refine the Nash time as an evaluation measure, one could let $y = 0$ represent the both cases of Nash equilibrium when calculation the total measure.

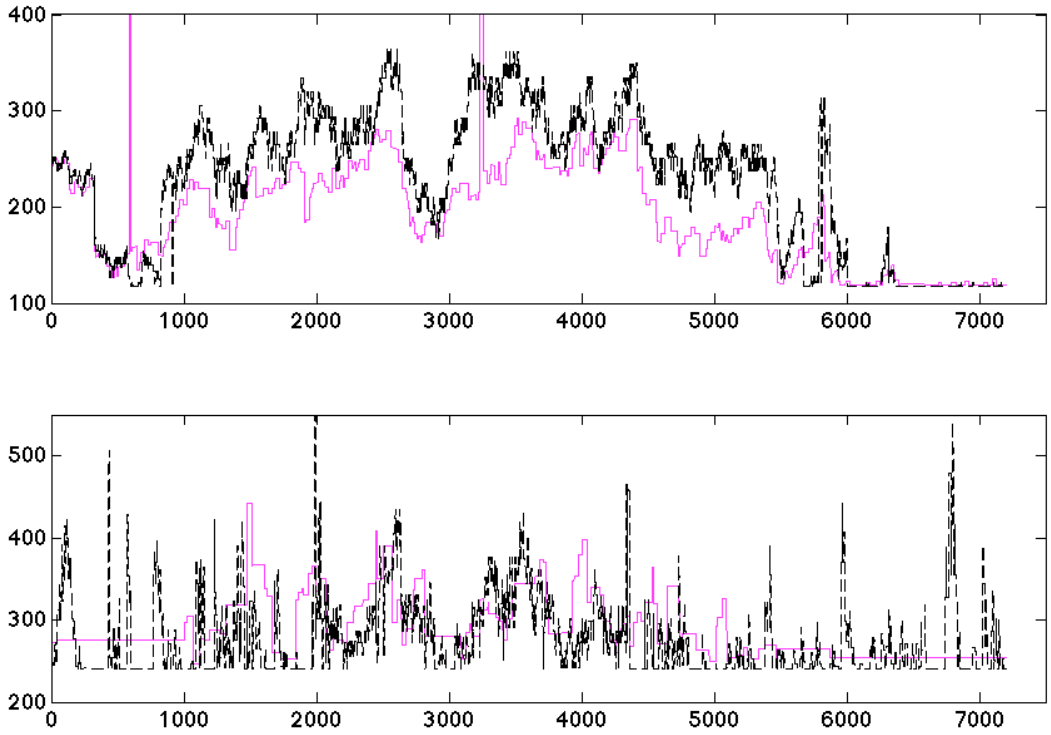

Figure 39. SB model with distribution, disturbance and incident detection functionality active. Purple represents the real travel times; black represents predicted value applied in feedback.

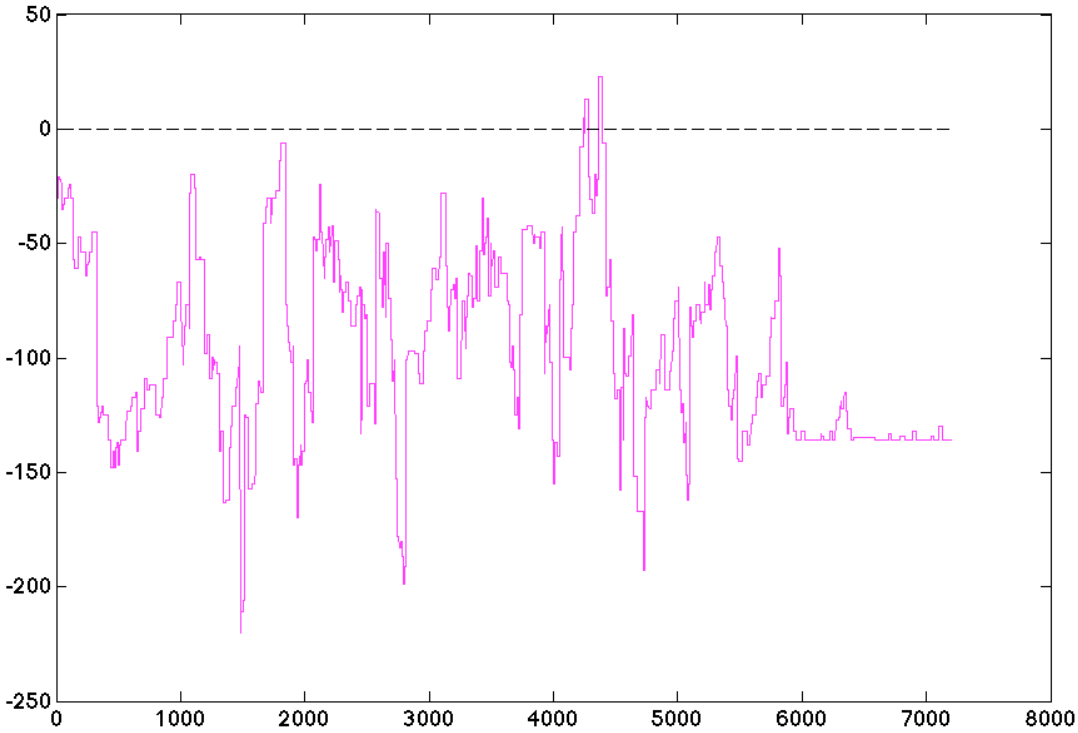

Figure 40. Observed Nash times for the accident scenario with the SB model with all features. The Nash times are centered around –100, which corresponds to the prediction error on the main route, subtracted with the prediction error on the alternative route.

Multi Bottleneck Model

The Multi Bottleneck Model was designed to handle the situation when there is more than one bottleneck influencing the travel times of a route. In the initial studies of the Tegel location, the alternative route was an example of such conditions. When the final simulations were carried out, however, no multi-queue pattern arose on neither route. Therefore, the model cannot be evaluated here in an informative way. For this purpose, another and more suitable scenario should be simulated. Given the current conditions and the prediction algorithms of the models, the performance of the multi-bottleneck model should be compared with the results of the SB model when all the features are active. Figure 41 and table 5B, (see the column "SB all features") shows that the SB gives better predictions, as well as better system performance. Hence the MB Multi Bottleneck model should be object for further evaluation and/or improvement.

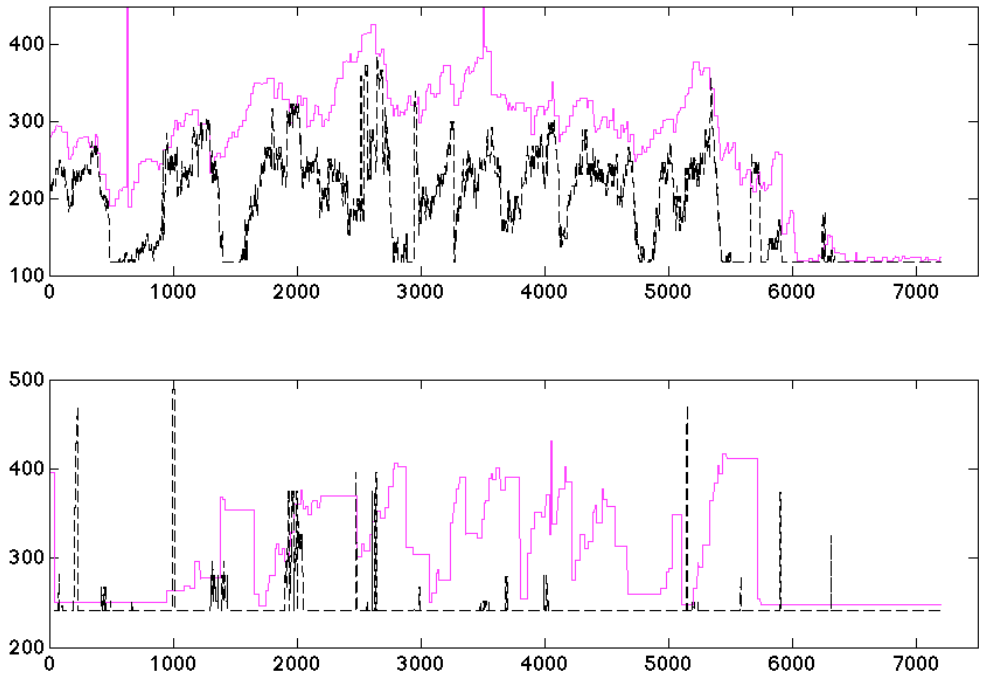

Figure 41. Multi Bottleneck Model. Main route (above) and the alternative route (below). Purple represents the real travel times; black represents predicted value applied in feedback.

Table 5A. Evaluation table for the Berlin network.
"Agents" are all agents that travel the whole main route and the alternative route, respectively.
"Travel time", "fit" and "Std. Nash deviation" is based on the group of agents defined above.
"Total Score" is based on all agents in the population. "Subpopulation Score" is based on agents that pass both the sign link and the destination link.

	Normal case	No accident, Basic SB	Accident, No Control	Reactive (No model)	Basic SB
Main route					
Travel time	368	292	1014	604	358
Agents	515	499	511	427	390
Fit	(94.33)	93.61	(61.12)	(no pred.)	77.85
Alternative route					
Travel time	311	292	317	482	320
Agents	7	133	7	101	123
Fit	(79.26)	80.70	(78.09)	(no pred.)	84.96
Total					
Travel Time	367	292	1004	580	349
Std. Nash deviation	116	160	764	256	127
Score					
Total Score	136,01820	135,89427	132,03027	135,90236	135,81067

Table 5B. Evaluation table for the Berlin network.

	Distribution	Disturbance	Distribution+ Disturbance	Basic SB Incident detection	SB all features	Multi Bottleneck
Main route						
Travel time	357	354	352	369	359	432
Agents	393	391	389	395	396	402
Fit	77.79	77.95	70.21	80.22	79.02	72.53
Alternative route						
Travel time	313	325	308	313	311	333
Agents	119	126	126	123	122	116
Fit	90.46	82.67	87.73	84.89	87.16	81.90
Total						
Travel time	347	347	341	356	348	410
Std. Nash deviation	110	135	118	121	105	78
Score						
Total Score	135,79666	135,80525	135,83151	135,87289	135,85715	135,85100

10.4 Score Evaluation

The subpopulation score reflects the economic benefit (or disbenefit) of the route guidance for the agents traveling either of the two routes. Table 6 displays the score analysis of the different guidance strategies for the Tegel location. The first row of each table contains aggregated scores for the subpopulation of interest. The second row displays the difference in the aggregated score from the accident case when no control is applied. Average score values are found in the third row. The fourth row shows the difference in the average score from the accident case when no control is applied. For all models except the Distribution model the guided agents benefit from the guidance. When using reactive control agents receive the lowest benefit; on average 1.24 euros. The Disturbance model gives the highest gains; agents benefit 2.45 euros on average. The aggregated benefits range from 2281 euros with reactive control to 4495 euros with the disturbance model. Keep in mind that all the values are computed for a 10 percent sample of total Berlin population. Hence, multiplying the aggregated values by ten gives the total simulated benefit.

Table 6. Aggregated and average score values for the subpopulation.

	Accident, no control	Reactive control	Basic SB	Distri-bution	Distur-bance	Detection	SB with all features	MB
Score sum	242011.49	244292.78	244756.12	241850.15	246506.94	245625.84	245611.85	245523.56
Difference		2281.29	2744.63	-161.33	4495.45	3614.35	3600.36	3512.08
Score avg.	132.03	133.27	133.53	131.94	134.48	134.00	133.99	133.95
Difference		1.24	1.50	-0.09	2.45	1.97	1.96	1.92

In total one can conclude that if guidance is applied, drivers that are guided will benefit. Unfortunately the benefit for the traffic system as a whole is not that obvious. The average scores of the whole population are displayed in the last row in table 5A and 5B. When guidance is applied the total score even decreases slightly. Hypothetically this is due to the negative influence of the control on travelers using the alternative route under normal traffic conditions, i.e. when there is no accident. The accident takes place in the evening peek which means that agents that receive the guidance are on their way home from work. In the scoring setup, there is not an economic penalty for arriving late at home, and economic benefit from home activities is low. It is likely that the total score would increase if the accident would occur during the morning rush hour, when delays have a considerably larger negative impact on the score. Just as the model evaluation in general, the score should be evaluated in different situations. The problem is to find alternative routes that have capacity left for additional traffic in the morning hours. Within the realms of this project the time was not enough for further evaluation. A suggestion for further research is to develop a MATSim algorithm that automatically finds potential alternative routes. As suggested in the introduction of this report, with an extension of the control system several alternative routes could be used for diverting the traffic of a highway with reduced capacity.

Another reason for the marginal influence on the system as a whole is that guidance focuses only on the travel times on the two routes. The reason for negative network effects indicated by the total score can that the guidance has a local control goal, i.e. it cares only about the subsystem that is subject to control. Therefore it seems worth to understand and improve the system as whole, taking not only travel time minimization into account.

Equipment ratio

The results presented so far are all based on the assumption that 80 percent of the agents are equipped with a COOPERS' device, and comply with the advice. It can be argued for that this is a relatively high equipment ratio, imposing the question of how many agents must be equipped with a COOPERS' device in order to get a significant benefit from the route guidance? Table 7 shows results of simulation runs using the Single Bottleneck model with all features and different compliance rates. It is clear that even with a compliance rate of only 20 percent, the subpopulation is better off than if no control were applied.

Table 7. Scores of agents traveling on the highway during the day for different equipment rates. Even when only 20 percent of the agents possess a device, guidance results in economic benefits.

	No control	20 %	30 %	40 %	50 %
Score sum	242011.49	242475.36	246567.64	244707.11	244662.06
Score avg.	132.03	132.28	134.52	133.50	133.48

10.5 General comments on the Tegel location

It has been complicated and time consuming to find a location in Berlin for the evaluation on the prediction models. Tegel was the fourth system that was examined. Despite all examination and awareness of problems the Tegel location has not been a completely satisfactory locations for evaluating the route guidance and the predictions models. The fact that the traffic conditions at Tegel have not been exactly the same

throughout our work can be partly due to the evaluation of the traffic simulating software.

The accident scenario has been in focus for the evaluation. It was not expected that the system could benefit from route guidance under normal traffic conditions. The vast majority of the population is experienced drivers that know which route is normally the fastest (in MATSim, the day-to-day replanning algorithms are designed to achieve optimal plans). Therefore it was a surprise that applying route guidance considerably reduced the travel times in the normal day scenario (se table 5A). This benefit also proves that the travel plans of the population used are not optimal.[9] Since these plans were generated, the replanning algorithms have been improved. Consequently, it is possible to produce more realistic input to the simulations. More realistic input implies more reliable evaluation, not least in the Score measure.

The modifications of the capacities constitute another reason for why the agents' plans are not optimized to the network. Since no day-to-day replanning has been carried out, then agents have not had the chance to adapt to the slightly changed conditions.

General comments on the performance of the models

The travel times plots presented in this chapter have in common that the predictions for the main route are mostly too high, and the those on the alternative route are too low. The standard deviation for the Nash time is therefore well above zero (see figure XX). A reason for the prediction errors on the alternative route is the absence of queues. The traffic situation is not that, which the Single Bottleneck model was designed to handle. The part of the travel time that exceeds the free speed time, is the result of more random and temporary dynamics than a queue mainly determined by a bottleneck capacity. For the analysis of such dynamics (including the question whether they are realistic or due to model simplifications/flaws), more detailed information about the traffic patterns is needed. Better knowledge of the route dynamics would also be beneficial for further improvement of the prediction models.

[9] The "run 86", iteration 80 (run86.it80.selected.plans.xml) have been used in the Berlin evaluation scenarios.

11. Conclusions

There is no doubt that the prediction of travel times improves the route guidance, measured in terms of lower travel times. The simulations in the test network very clearly confirm that the Single Bottleneck approach for calculating travel times is well suitable in MATSim. Compared to reactive control, the Nash times are reduced by 95 percent in the test network and 57 percent in the Berlin simulation. The travel times are reduced by 26 and 41 percent, respectively.

Berlin simulations during the autumn showed that there were traffic dynamics that the regular SB-model does not handle. For this reason we continued with developing the additional features described in part II. Unfortunately these features could not be tested as thoroughly as we hoped because of the limited complexity of the test network and the technical problems with the Berlin simulations. According to the test network, however, all of the models have positive effect on the situation that they were designed to handle, although it could not be determined to what extent they bring us closer to the control goal and/or improve the travel times. The Distribution model, the Disturbance model and the Multi Bottleneck model all increase the model fit significantly in the test network.

In order to draw reliable conclusions about the models, more evaluation should be carried out. Indications are not dependent only of what evaluation measure that is used (Score, average travel times of agents subject to route guidance, average travel times for the whole network, etc), but also of which part of the network that is chosen for evaluation. Simulating the route guidance with feedback of the travel times on other locations can give much useful knowledge.

The traffic simulating software caused us some practical problems. Still MATSim is without doubt a powerful tool for the development of traffic management systems. To implement our approach with predictive feedback control in a real world system, further evaluation should be carried out on real scenarios. A first step towards a route guidance system in use, are reliable MATSim simulations on different times and places in Berlin.

One of our aims that were achieved was to eliminate the need of parameter tuning. With good predictions and a minimal message hold-time, there is no need to apply more advanced controllers than the bang-bang controller, which has no control parameters. Neither the prediction models need to be estimated – all the necessary information is in the network file and accessed via MATSim. Once the routes are specified, neither the prediction model nor the feedback controller needs any more estimation. In contrast to any controller that produces a continuous control signal, the bang-bang control signal is already compatible with the discrete traffic system. The binarization process that is otherwise carried out in the class `VDSSign` involve a trade-off between having a freshly updated control signal and the exactness of the control signal when it is translated into a sequence of binary numbers.

12. Ideas for future research

The goal to develop predictive models that can improve traffic guidance control compared to using measured feedback is achieved, but as we can see from the Berlin simulations, there are still modifications that need to be done. One idea that we developed in a late phase of the thesis work and was not feasible to develop due to time constraints, was that although the measured times are too late, they could help us to online-tune the models. If the predictions have had too low estimates of the travel time during some time, the models could be tuned to predict higher. This would be a "feedback tuning" of the model itself that further could improve the control.

Another idea is to use a traffic simulator such as MATSim for online predictions. Using COOPER devices for collecting data, a simulator could place agents in a network at the right locations, with the right speed etc. Then the simulator is run forward guiding an agent either left or right and the travel time is measured. This approach would probably be fruitful when the routes are long and the traffic situation complex.

Some specific changes can be made that should improve the control performance. Outliers, i.e. travel times that do not reflect the current traffic situation, should not be used for feedback. For example, there are agents that stop on the route for some activity, get back on the route much later during the day and are eventually measured by the sensor in the end of the route. A way of handling the problem is to ignore such outliers and instead giving the last realistic Nash time as feedback (in reactive control) or as input to the model (in predictive control). However, the choice of criteria for which measurements that should be regarded as not normal is a non-trivial problem.

The automatic detection of capacity reduction (which is used both in the Multi Bottleneck model and in the incident detection of the Single Bottleneck model) could be improved by an extension of the detection criteria. The calculations of prediction will be less sensitive and more stable if the travel times of at least two agents, instead of only one, must exceed the free speed + ignored queuing time on a link before it is considered to be a bottleneck.

In the theory chapter of this report we discussed the possibility of extending the guidance system to more than two routes. If there is an existing infrastructure supplying the measurements for any route (which is the case in MATSim or with COOPERS-devices in the real world), adding more alternative routes should be fairly easy. Such an extension could lead to a much more efficient usage of the capacity of the traffic network, and is thus in our opinion a good way of further improving the predictive feedback control.

The Multi-Bottleneck model was a result of analysing Berlin simulations. We noticed that the main problem for all the SB-models seemed to be the occurrence of two queues on the alternative route. Several separate congestions are probably a common situation when there are many inflow and outflows along a long route – which may be a common in real traffic systems. With this motivation in mind, we hope that the Multi-Bottleneck approach will be evaluated more thoroughly.

The simulation score is interesting as an evaluation measure. It takes into account the performance over a whole day for every agent in the population, and then translates this performance into an economic entity. As such it facilitates socioeconomic evaluation of measures like new traffic management strategies, which in turn facilitates the decision process in politics. In our evaluation we have used the standard scoring parameters,

meaning that only the *time* spent travelling affects the score of an agent (the score is negatively proportional to the travel time; but there is also extra penalties, for instance for arriving late at work). The route guidance is designed to optimize the travel time – but time is not always correlated with cost. When taking the alternative route is faster than travelling the congested highway, it can nevertheless be more costly for the reason that the agent travels a longer distance. Using a *distance cost* (based on fuel consumption, mechanical wear, etc) in the scoring would result in a realistic cost for the individual agent. In addition to this, the distance factor in the score can be specified to include also an *environmental cost*, with a greater negative contribution from emission (possibly dependent of the fuel type, the number of persons in the vehicle, etc). That way the simulation score would more realistically reflect the cost of travel for the collective.

References

Cetin N., Burri A., Nagel K., (2003). "A Large-Scale Agent-Based Traffic Microsimulation Based On Queue Model", STRC 03 Conference paper, Session Microsimulation, Third Swiss Transport Research Conference

Charypar D., Nagel K., (2004). "Generating complete all-day activity plans with genetic algorithms", Transportation Working paper series, 2004:04, Transport Systems Planning and Transport Telematics Department of Berlin University of Technology.

COOPERS home page, www.coopers-ip.eu/index.php?id=project, 29 Jan, 2008

Diakaki, C., M. Papageorgiou, and T. McLean (1997). Simulation studies of integrated corridor control in Glasgow. Transportation Research C, Vol. 5, No. 3/4, pp. 211-224.

Gawron, C., (1998). "An Iterative Algorithm to Determine the Dynamic User Equilibrium in a Traffic Simulation Model", International Journal of Modern Physics C, Volume 9, Issue 3 (May 1998), 393 – 407.

Helbing D., (2003). "Letter To the Editor; A section-based queueing-theoretical traffic model for congestion and travel time analysis in networks", Journal of Physics A: Mathematical and Theoretical, Gen. 36 No 46 (21 November 2003), 593-598

Illenberger J., Flötteröd G., Nagel K., (2007). "Enhancing MATSim with Capabilities of Within-Day Re-Planning (I)", VSP Working Paper, 07-09

Nemec M., (AustriaTech – Gesellschaft des Bundes für technologiepolitische Maßnahmen), "COOPERS - Cooperative Networks for Intelligent Road Safety", www.sevecom.org/Presentations/2006-02_Lausanne/Sevecom_2006-02-02_C%20COOPERS.pdf, 29 Jan, 2008

Papageorgiou, M., Wang, Y., (2005). "Real-time freeway traffic state estimation based on extended Kalman filter: a general approach", Transportation Research Part B 39 (2005), 141–167, page 141

Scott W. Ambler, "UML 2 Class Diagrams", http://www.agilemodeling.com/artifacts/classDiagram.htm, 29 Jan, 2008

Rommel, C. (2007) Automatic Feedback Control Applied to Microscopically Simulated Traffic – The potential of route guidance in the Berlin traffic network. VSP Working Paper, 07-16.

Vandaele N., Van Woensel T., Verbruggen A., (2000). "A queuing based traffic flow model", Transportation Research Part D: Transport and Environment, Volume 5, Issue 2 (March 2000) 121-135

Wikipedia, article for "cellular automata", http://en.wikipedia.org/wiki/Cellular_automata 30 January, 2008