CHALMERS

Integrated Traffic Control Design

Master's Thesis in Intelligent Systems Design

Michael Nilsson

Department of Signal & Systems
Division of Automatic Control & Automation & and Mechatronics
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2013

Abstract

In this thesis a modular code skeleton for combining Variable Message Signs (VMS) and Adaptive Cruise Control (ACC) as a combined system are being presented. Basic control theory for the two control types are given with corresponding pseudo code. Also simulation with the different controllers are compared and discussed. Finally some tips and tricks regarding the simulation software VISSIM are given. Even tough the simulation software VISSIM is in the focus, this thesis can be used to support implementation of combined controllers in other software as well. Important to note is that the controllers implemented are merely to verify that the code skeleton is working. In other words, the simulation results presented in this thesis are **not** to be used when judging the performance of a controlled network.

Acknowledgements

There are many people I would like to thank for supporting me while doing this master's thesis, so thank you all, you know who you are. But there are some people I would like to mention especially since their help have been very important. First I would like to thank Stefan Pettersson at Viktoria Institute for accepting me as a master's thesis student at the company and providing me with necessary contacts, a working area and support on regulator questions. A big thank also goes to János Polgár for a tremendous help with getting me into the VISSIM software and how to program the different modules. Also a huge thank you goes to Adam Balázs Kulcsár who made this thesis possible both by giving me the idea for it and for, during the whole project time, committed and wholeheartedly helped and guided me in the right direction based on his great knowledge and experience in the field of traffic control. Another thank goes to Koenraad Verduyn and his colleagues at PTV Vision for helping me with the VISSIM license and for assistance in some software and programming issues. Finally, I would like to thank my lovely girlfriend Maria for cheering me up when it sometime felt hopeless and for listening on my chatter about traffic during this project.

Michael Nilsson, Göteborg 2013-06-01

Abbreviations

ACC = Adaptive Cruise Control
VMS = Variable Message Sign
GUI = Graphical User Interface
BPW = Backward Propagating Wave
PTV = Planung Transport Verkehr

Contents

1

Introduction

Since the number of vehicles and the need for transportation permanently grows, traffic congestions have become one of the most important traffic control related challenge nowadays. By being able to improve the control of road networks, the capacity of a road can not only be increased, but also make the roads more safe. By adapting the speed of vehicles based on the current situation on the road the amount of sudden-braking can be reduced. There are different perspectives on how to control cars in a network. Two common views are the microscopic view and the macroscopic view. In the microscopic view each car is adapting it's speed to the current traffic situation. Often the actuator adapting the speed is the person who drives the car. Drivers have different perspectives on how to drive. One perspective is an ego-centric driver who only has focus on improving it's own situation to get as short travel time as possible while another is a social driver interacting with it's surrounding. Surely a social driver will give the network a better over-all performance than the ego-centric driver would. However, even though the social driver is doing it's best to create a good driving situation for all the vehicles on the network, there are still limitations. A driver might be able to see and adapt it's speed based on the situation some hundreds meter ahead in good driving conditions but the driver can't know what is happening kilometres ahead. Here the macroscopic control view is different. When using macroscopic controls it's not a single object that matters, it's a more global view of the flow. In this case the macroscopic view is whole segments of a road where the density, which is based on the numbers of cars per kilometre on the road, is what's being used to control the traffic. If the density peaks on a segment and causes congestion, this information can be used to change the allowed speed using variable message signs several kilometres before the congestion. Doing that will reduce the severity of the congestion both on a safety level and on a flow level. The idea is to combine the social driver (in our case let the social driver be an adaptive cruise control) and feed this driver with information about what happens kilometres ahead and by doing this improve the traffic network performance.

1.1 Motivation

In the area of traffic control much work has been done. It already exists cars on the road equipped with adaptive cruise controls (ACC) [1] and there are different control methods for these Adaptive Cruise Controllers [2]. When it comes to variable message there has been done a lot of research [3] [4] [5] [6]. A VMS can show different kinds of information e.g. warn for slippery road, inform about upcoming work zones, give traffic guidance etc but in this study only different speeds will be shown by the VMS. Variable message signs are used on several roads now days and the possibility to increase the capacity of roads by changing the speed based on the traffic situation is known [7]. However, to be able to apply the idea of a combined controller there needs to be an infrastructure (road signs, sensors and vehicles equipped with necessary systems) available to do this there has to be a system for communication between car and infrastructure along the roads. A project with this kind of infrastructure is the AUTOPIA project [8]. Using a V2I (Vehicle-to-Infrastructure) and basic control methods for Variable Message Signs and Adaptive Cruise Control, a combined control method can already be implemented in real-life and hopefully on larger road networks in the future. Because after all; "The final goal is indeed to integrate the microscopic and macroscopic control strategies to create intelligent traffic systems of high efficiency" [9].

1.2 Method

The method used during this thesis was modelling and simulations. Models for ACC controller 3.2, VMS controller 3.1 and Combined controller (which is a combination of ACC and VMS) has been implemented in a traffic simulation software called VISSIM to make it possible to run simulations on traffic scenarios when these three controllers types are affecting the traffic. This meant that a controller for each model had to be designed. Since the main goal of the thesis was to create a code skeleton to make it easier for others to implement more advanced controllers using the skeleton the result from the simulations using the controllers designed in this thesis were only to validate the code skeleton.

1.2.1 The VISSIM software

The software used to do the simulations in this thesis is VISSIM, which is a software developed by PTV Vision [10]. VISSIM is a microscopic traffic simulator meaning that it simulates each vehicle separately. VISSIM has a simple GUI to create simulations of traffic situations and this GUI is probably enough for many users. However, it is not enough to control each vehicle separately during the simulation, and this is something needed in this thesis. To solve this problem PTV Vision has extensions to their basic configuration of VISSIM making this possible. One of the extensions is the COM-module and by using this module it's possible to affect the vehicles, speed signs, traffic lights, and other aspects details, while the simulation is running. The COM-module is a must to be able to do the different kind of simulations done in this thesis. The COM-module comes with a library of functions and almost everything is possible to affect in the network while the simulation is running which is very useful. The COM-interface supports many programming languages, for example Java and Python. The language used during this thesis was the C++. One thing which is not possible to do from the COM-module is to change the car-following model used by cars on the network. Therefore PTV Vision also has an extension to make it possible to use an external driver model. In this thesis this extension made it possible to implement the simple ACC control algorithm. A good feature is the possibility to decide which vehicle types that shall use the external driver model from inside the GUI (where the simulation is run). This makes it possible to create an own car type with an external driver model while all the other car types uses the driver model provided by PTV Vision.

1.2.2 Visual Studio 2010

The development environment used to write the the code for the simulations was Visual Studio 2010. This is a development environment from Microsoft [11] and it support many different programming languages. The programming language used during this thesis was C++.

1.2.3 Matlab

Matlab [12] is a well-known tool for calculations and computations in the area of engineering. Matlab uses matrices as input to do different kind of operations. During this thesis Matlab was used to create graphs based on data from the simulations. These graphs made it easier to understand the results from the simulations since the data from the simulations came as large files of numbers.

1.3 Contribution

The contribution to the area of traffic control from this thesis is mainly a code module skeleton to use for combing controllers. The purpose is to give thoughts of how to create this type of skeleton, how to implement controllers to it and tips and tricks on how to use VISSIM. Another contribution is to show how a combined controlled system can affect a road network. Based on the results from this thesis the hope is that further research will be initiated. By showing how to build a combined controller and also give some results of how such controllers can affect network the hope also is that it tempts more people to continue the research of combined traffic controllers which hopefully leads to better and more advanced combined controllers which in turn leads to a increased capacity of the existing traffic networks and less accidents in the traffic.

2

Control explanation

2.1 The ACC controller

This thesis uses a simple car-following model for cars driving faster than 35km/h. For cars driving below 35 km/h the standard Vissim car-following model takes control of the cars. The reason for this is that the simple control loop used at higher speeds is not working good at low speeds and cruise controllers in real cars often turns of at low speeds. The car which is being controlled will be refereed to as the *target car* and the first car in-front of and on the same lane as the target car will be refereed to as the *leading car*.

net_distance
: the distance between the target car and the leading car.

lead_vehicle_speed
: the speed of the lead vehicle.

desired_distance
: the desired distance between the target car and the leading car. This distance is based on the leading car speed. It's simply the lead_vehicle_speed times a time constant, in this case 1 second. In other words, if the leading car is driving in 108 km/h, which is 30 m/s, the desired_distance is 30 meter.

max_speed_acc
: is the maximum speed allowed to be reached by the target car. Basically this is

the desired_velocity of the target car which is set on each vehicle when it enters the network or by the VMS system on the network. The max_speed_acc is used when there is no car to follow.

current_speed
is the current speed of the target car.

k
is the gain of the P-controller.

ACC_OnOff
The speed limit for activating/deactivating ACC on ACC-equipped vehicles.
ACC_OnOff < current_speed: **ON**, ACC_OnOff >= current_speed: **OFF**).

This is a part of the ACC code from the *External Driver Model* module. Since the whole precise code is in the appendix, only a pseudo-code describing how this simple ACC operates will be shown in this chapter. As one easily can understand, there are many improvements which can be done to make this ACC better. The ACC-controller is a simple P-controller but it will have an integrating part since the controlling is done by adjusting the acceleration based on the speed. The reason that an integrating part are included in the controller is because the acceleration is the derivative of the speed which means that the integral of the speed is the acceleration.

if (net_distance > desired_distance)
*The distance between the vehicles is **not** smaller than allowed*
if (lead_vehicle_speed > max_speed_acc)
The leading vehicle is going faster than the highest allowed speed
do
desired_acceleration = k * (max_speed_acc - current_speed)
*Target vehicle is **not** adapting it's speed based to the leading vehicle speed*
else
*The leading vehicle is **not** driving faster than the highest allowed speed*
do
desired_acceleration = k * (lead_vehicle_speed - current_speed)
The target vehicle is adapting it's speed based on the leading vehicle speed
else
The distance between the target vehicle and the leading vehicle is smaller than allowed
if (lead_vehicle_speed > max_speed_acc)
The leading vehicle is going faster than the highest allowed speed
do
desired_acceleration = k * (max_speed_acc - current_speed)

*Target vehicle is **not** adapting it's speed based to the leading vehicle speed*

else

*The leading vehicle is **not** driving faster than the highest allowed speed*

do

desired_acceleration = k * (lead_vehicle_speed - current_speed) - ε[1]

The target vehicle will increase the distance to the leading vehicle

2.2 The VMS controller

The Variable Message Sign controller is done in the *COM module* and like the ACC code, the VMS code will be described as an pseudo-code. The VMS controller is also a P-controller and uses some extra variables to further change how the controller works.

k
: is the gain of the P-controller.

rho_cr
: is the critical density threshold, meaning that when a controlled segment of the road has a density equal or higher than rho_cr the VMS shall activate.

updateFreq
: decides how often the VMS should be updated.

FIRSTCONTROLLEDSEGMENT
: The first segment (link) that is affected by the VMS system.

LASTCONTROLLEDSEGMENT
: The last segment (link) that is affected by the VMS system.

The VMS controller is put inside a function and this function, in turn, calls for other functions to create the full VMS controller function. Therefore only the main function will be pseudo-coded and the functions that the main function uses are described afterwards with a text telling what happens inside these functions. The main function, which call all the other functions, has the name *updateSpeedSignLocal* and together with *activateVMSlocal*, *setNewSpeedLocal* and *resetSignLocal* it creates the VMS controller.

[1] Set to a value. The value can be decided by using an empirical approach doing simulations with different values and compare the result.

[**updateSpeedSignLocal(segmentNumber)**], *main function*
  for (all controlled segments)
    if [**activateVMSlocal(current controlled segment)**], *see function description 1*
      See explanation below
      do
        tmpDifferenceDensity = oldSegmentDensity - currentSegmentDensity
          tmpDifferenceDensity stores the difference in density between
          the current and previous sample
        tmpSpeed = [**getSpeedDecision(tmpDesiredSpeedDecision)**], *see function description 2*
          tmpSpeed will after getSpeedDecision have the
          speed currently shown by the selected segment VMS
        tmpSetSpeed = tmpSpeed + (k * tmpDifferenceDensity)
          The current speed shown by the VMS on the selected segment
          plus the difference in density times the factor k will be the basis
          for the new speed to be set on the VMS for the current segment
        [**setNewSpeedLocal(tmpSetSpeed, segmentNumber)**], *see function description 3*
          See explanation below
    else
      *if activateVMSlocal returns **false***
      do
        [**resetSignLocal(selectedSegment)**], *see function description 4*
          See explanation below

activateVMSlocal(segmentNumber), function description 1

Compares the density of the current link with **rho_cr**.
If the link density is equal or higher than **rho_cr** the function returns *true* else it returns *false*.

getSpeedDecision(tmpDesiredSpeedDecision), function description 2

Simply requests the current speed shown by a selected VMS sign. A thing to remember is that all VMS signs have own numbers. In other words, it not possible to directly request what speed that should be shown on a segment without knowing which number the VMS sign has on that segment. One solution for this is to have the same number on the segment as on the sign. As an example segment 5 in the network also has the VMS sign number 5. If there are several lanes with several VMS signs the first sign can have number 50, the following 51 and so forth.

setNewSpeedLocal(tmpSetSpeed, segmentNumber) , function description 3]

This function uses the speed which has been calculated in the main function and the selected segmentNumber. These two parameters are then used in this function to set and update the VMS sign for the selected segment. However, due to the fact that the tmpSetSpeed can be a decimal value, e.g. 87.9, which is not preferred to be set as a speed limit, the function has to take the closest speed that a VMS sign can show and display this instead. During this thesis, the speeds that a VMS could show were *50, 70, 90, 110, 120*. This means that in the example 87.9 the speed that the VMS would show is 90. The setNewSpeed function also call another function namely approxSearch. This function that has the task to take the tmpSetSpeed and find the closes speed that can be set by the VMS and return this to the setNewSpeedFunction. As mentioned before the whole code is in the appendix for those interested in a deeper understanding of the code.

resetSignLocal(selectedSegment), function description 4
This function takes the selectedSegment and resets the VMS sign for this segment to show it's start speeds. In other words, the speed that the VMS sign showed at the start of the simulation. These speeds are those used in the non-controlled simulation.

The mathematical expression for VMS is the following;

$$Speed_{new} = Speed_{current} + (k * Density_{diff})$$

where $Speed_{new}$ is the new speed which will be set by the VMS, $Speed_{current}$ is the current speed set by the VMS, k is a gain factor and $Density_{diff}$ is the difference in density on the controlled segment between the last sample and the current sample.

2.3 Control layout

The way that the VMS and ACC were connected to each-other was that variable message signs (VMS) sent data with the highest allowed speed on the controlled segments to the cars equipped with adaptive cruise controllers. This highest allowed speed was then the highest allowed speed for the ACC cars. If there was a car in-front of the ACC car which drove slower than the highest allowed speed set by the VMS the ACC car adapted it's speed to this slower vehicle. Picture 2.1 describes how the different controllers are connected. First *VISSIM* gathers data about the density of all the segments of the road and constantly makes it possible for the *VMS* to receive this density data from the current and the last sample (ρ_{diff}). This makes it possible for the *VMS* to activate as soon as a critical density (refereed to as rho_cr, described at section 2.2) has been detected on any segment of the road. When a critical density has been reached on a segment the *VMS* takes the difference in density, ρ_{diff}, for that segment of the road and multiplies it with a factor *k*. This product ($k \times \rho_{diff}$) is then added to the current speed shown on the VMS signs. After this addition the *VMS* has a speed that it would like to write on the VMS signs. However, since there are some predefined speed limits already on real road networks, the *VMS* has to compare the speed that it want to set to the speed limit which is the nearest allowed to show. This speed is set and shown on the VMS signs. The speed shown by the VMS signs are then one of three inputs to the ACC. Vehicles not equipped with ACC will from now on follow the new speed set by the *VMS*, refereed to V_{VMS}. Vehicles equipped with ACC will only follow this new VMS speed in two cases. As picture 2.1 shows the $VMS1_{new}$ (which is the speed set on the VMS on the segment related to the $VMS1_{new}$ signal) signal meets with two other signals namely V_{ACC} and $V_{vehicleahead}$. The $V_{vehicleahead}$ is the speed of the vehicle ahead, if there are any, and the V_{ACC} is the speed of the current controlled ACC vehicle. All this speed data along with the distance between the ACC vehicle and the vehicle ahead (and the data to calculate this) are sent to the ACC controller which uses it to decide the ACC vehicles new acceleration. What happens is that every car with ACC is looking on the vehicle ahead. By using the distance to this vehicle and the speed it has together with the data of it's own speed and the speed shown by the traffic signs, the ACC will set a new desired acceleration for the ACC car. How this computation is done is described in 2.1 but what it basically do is calculating the difference between it's own speed and the speed shown by the *VMS*, or the speed of the vehicle ahead. The result from this calculation is then multiplied with k. Based on this data a desired acceleration for the ACC equipped vehicle is calculated. Important to notice is that the layout picture 2.1 is designed to easier show how the control system operates. However, there are no middle gain modules as **k** and points before the actual controllers where difference in speed and density is calculated. All these gains and calculations are done in each controller so basically all the data needed for controlling is directly sent to the corresponding controller.

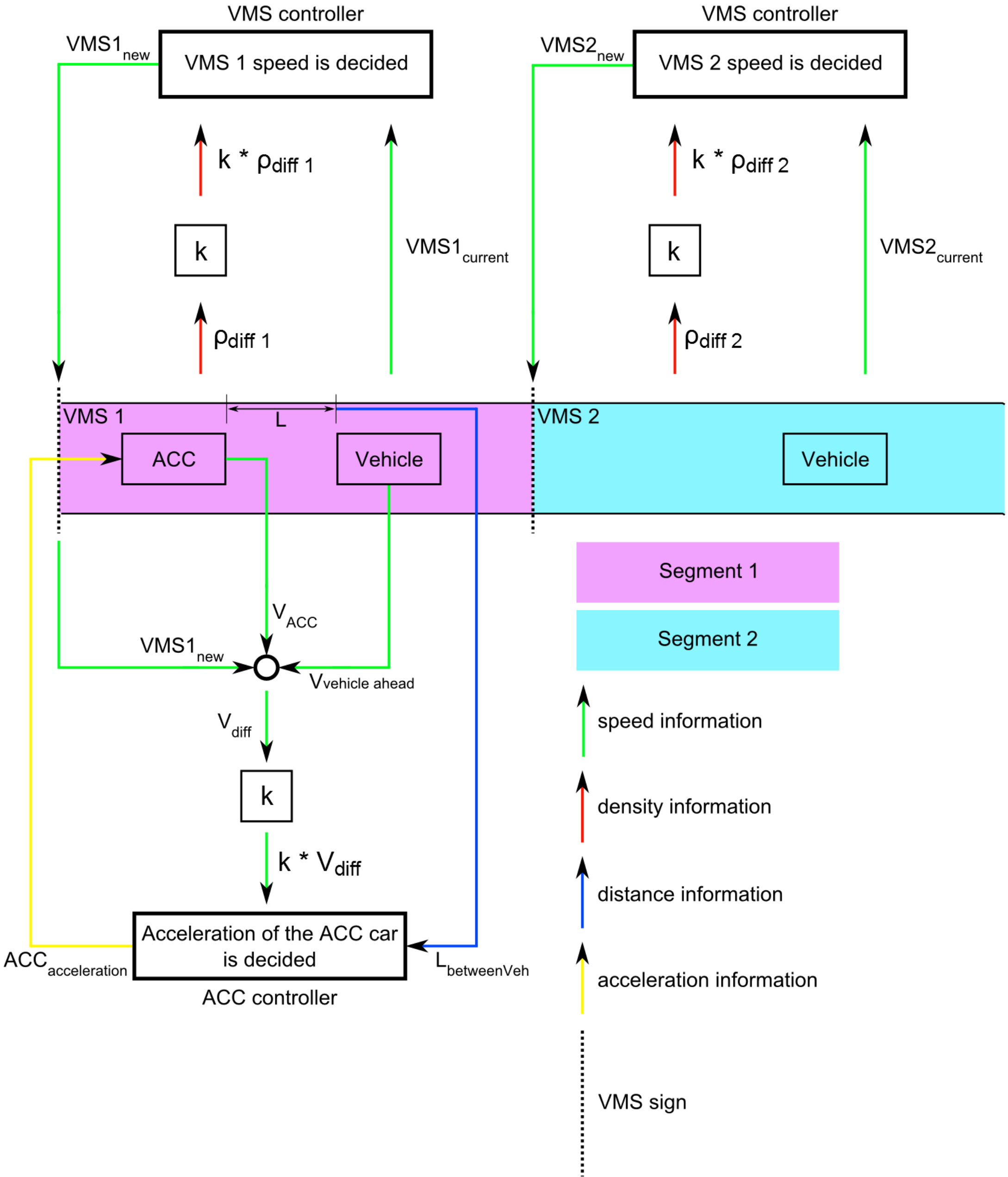


Figure 2.1: A layout of how the system is operating

3

Aspects of car-following models

There are many models for controlling both variable message signs and adaptive cruise controls. Firstly there will be a short description of how a macroscopic view can be interpreted and then there will be a description of a well-known macroscopic model. After this a microscopic view will follow. The models presented are **not** used in the controllers implemented in this thesis. They are only used as examples.

3.1 Macroscopic

A common explanation of the word Macroscopic is *visible to the naked eye*. This means that it's something which can be observed just by looking at it. This way (the sentence) a macroscopic control can be interpreted as something which is possible to look at and control by using what is observed. As an example, in this thesis, a road is divided into several five hundred meter segments where the density (the number of vehicles per kilometre) are observed to be able to control the vehicles on the road by changing the speed limit on the VMS (Variable Message Sign) at the end of each segment. A macroscopic control looks at the big picture, i.e. the flow, the average speed of the vehicles, the density (as already mentioned), etc. A well-known macroscopic model for traffic flows is the Lighthill-Whitham-Richards model (3.1). There are several steps to describe this model. All, but one equation, are taken from a thesis done by Tamás Luspay[13].

$$\frac{\partial \rho(t,x)}{\partial t} + \frac{\partial \rho(t,x) V(\rho(t,x))}{\partial x} = 0 \tag{3.1}$$

The Lighthill-Whitham-Richards (LWR) model is based on the vehicle conservation law [13, p.12-13]. Picture 3.1 shows a segment of a road where x_a and x_b are two arbitrary positions (on it). The goal is to describe how many vehicles that are on the road between the two points (the pink area in the picture) which represents the density of the road.

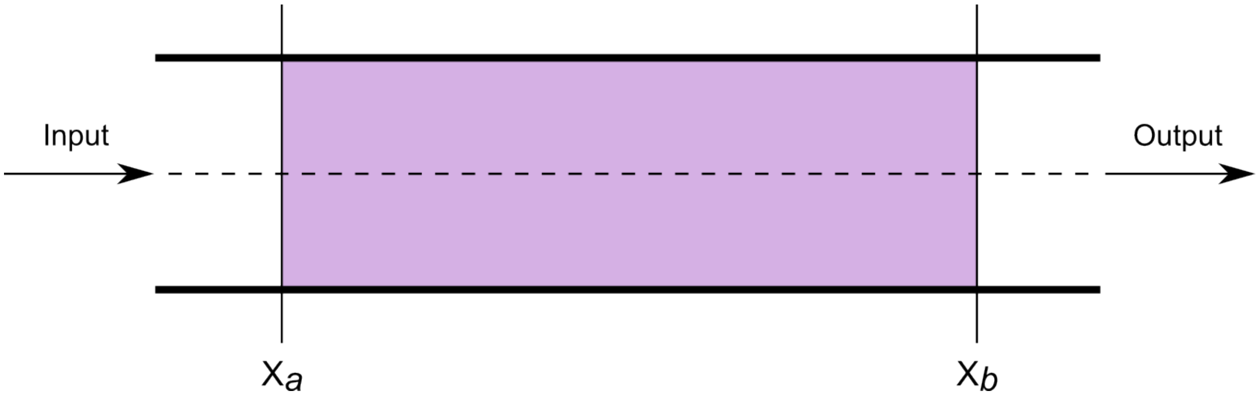


Figure 3.1: A picture representing a part of a road. The pink area represent the area between the two points A and B. The number of cars on the pink area is to be determined.

One way to do this is to slice the road between position x_a and x_b and count the number of vehicles on each slice, or more precisely, look at the density of each slice. This method results in equation 3.2.

$$N_{a,b}(t) = \int_{X_a}^{X_b} \rho(t,x)\,\mathrm{dx} \tag{3.2}$$

Another way to decide the number of vehicles on the pink area 3.1 (the area between x_a and x_b) is to look at the inflow at point x_a and the outflow at x_b. Important to note is that the road between the two measuring points has no off-ramps or on-ramps. In other words, the vehicles does not appear or disappears between the measuring points. The above mentioned information gives the equation 3.3 where the number of cars on a segment can be calculated using flow measuring at two points.

$$\frac{\mathrm{d}N_{a,b}(t)}{\mathrm{dt}} = q(t,x_a) - q(t,x_b) \tag{3.3}$$

Another important thing to note is that a starting condition has to be given. This is necessary because otherwise it wouldn't be possible to decide the amount of vehicles on the road by measuring flows. If there already are an unknown number of vehicles on the segment when the flow measurement begins, there will not be taken into account for. Therefore a starting condition has to be given or else the inflow on point x_a and the outflow at x_b has to been measured from the beginning of the road's existence. The two equations 3.2 and 3.3 can now be merged together by taking the expression for "number of vehicles" from the first equation (3.2) and substitute with the "number of vehicles" variable in the second equation (3.3). This substitution gives the following equation, 3.4.

$$\frac{\mathrm{d}}{\mathrm{dt}} \int_{x_a}^{x_b} \rho(t,x)\,\mathrm{dx} = q(t,x_a) - q(t,x_b), \tag{3.4}$$

Equation 3.4 is the integral form of the vehicles conservation law [13, p.12]. The next step to reach the LWR model is to rewrite the flow part of the newly derived equation 3.4. Luspay describes that "Since the spatial and temporal variables are independent in the Eulerian frame, the order of derivation and integration can be changed. Moreover the right-hand side can be written in an integral form, by using the Newton-Leibniz formula."[13, p.12] The Newton-Leibniz equation [14] which is being used in this thesis looks like,

$$\int_a^b f(x)\,\mathrm{dx} = F(b) - F(a) \tag{3.5}$$

Applying this expression on the integral form of the vehicles conservation law equation 3.4 gives the following result,

$$q(t, x_a) - q(t, x_b) = \int_{x_b}^{x_a} \frac{\partial}{\partial x} q(t, x)\,\mathrm{dx} = -\int_{x_a}^{x_b} \frac{\partial}{\partial x} q(t, x)\,\mathrm{dx} \tag{3.6}$$

By replacing $q(t, x_a) - q(t, x_b)$ from equation 3.4 with the newly derived expression and at the same time simplifying the equation the result will come out as,

$$\int_{x_a}^{x_b} [\frac{\partial \rho(t, x)}{\partial t} + \frac{\partial q(t, x)}{\partial x}]\,\mathrm{dx} = 0 \tag{3.7}$$

Luspay also states that "The definite integral of the expression is always zero independently from the upper and lower limits. The only expression which satisfies this is the zero function, hence we can formulate the differential form of the vehicle conservation law:"[13, p.13]3.8

$$\frac{\partial \rho(t, x)}{\partial t} + \frac{\partial q(t, x)}{\partial x} = 0 \tag{3.8}$$

Now the vehicle conservation law is derived, but there is still a problem. The equation contains two unknown distributions ($\rho(t, x)$ and $q(t, x)$). Here the LWR model enters as a solution to the problem. The LWR model says that the mean speed in a traffic network is directly affected by the density on the road,

$$v(t, x) = V(\rho(t, x)), \tag{3.9}$$

This formula introduces the variable V which is the speed on the network where the speed is related the density on the road. It is then possible to use the basic flow equation, 3.10 by replacing $v(t, x)$ with $V(t, x)$ gives 3.11.

$$q(t, x) = \rho(t, x) * v(t, x) \tag{3.10}$$

$$q(t, x) = \rho(t, x) * V(t, x) \tag{3.11}$$

Now, using equation 3.8 and replace $q(t, x)$ using the flow expression from equation 3.11 gives the equation 3.12.

$$\frac{\partial \rho(t, x)}{\partial t} + \frac{\partial \rho(t, x) * V(t, x)}{\partial x} = 0, \tag{3.12}$$

This is the dynamical equation of the LWR model.

3.2 Microscopic

Controlling vehicles on a microscopic level means that each vehicle are controlled separately. To analyse and control cars at a microscopic level has become possible during the last decades due to the fast development of information technologies [9]. The generic concept of microscopic control is that every vehicle is controlled individually based on driving rules. An example of a controller that makes microscopic control possible is the Adaptive Cruise Control (ACC). Adaptive Cruise Control is a function which makes it possible for a vehicle to keep track of the distance and velocity of a vehicle in front and adapt it's own speed according to a driver rule. Mainly this technology was developed to increase the comfort for drivers [9], but it can also be used to control vehicles at a microscopic level. There are many different driving models that can be applied to an ACC. One well-known model is the Gazis-Herman-Rothery (GHR) model [15] which has the following equation.

$$a_n(t) = cv_n^m(t)\frac{\Delta v(t-T)}{\Delta x^l(t-T)}$$

Luspay describes that "$a_n(t)$ is the acceleration of vehicle n at time t by a driver and is proportional to, v the speed at the nth vehicle, Δx and Δv, the relative spacing and speeds, respectively between the n:th and the n - 1 vehicle (the vehicle immediately in front), assessed at an earlier time t-T, where T is the driver reaction time, and m, l, c are the constants to be determined." [16]. In this case the assumption is that the actuator is a human driver, but this model can also be used on an ACC were an actuator make the decisions during the car-following. This is a simple example of a car-following model which makes it easier to get a basic understanding how an ACC could operate. However, there are many more models for car-following.

3.3 BPW

A backward propagating wave (BPW) occurs at congested areas on a road. On the space-time-density graph 3.2, two waves have been marked with circles. The appearance of these waves are similar and they are seldom hard to locate but they can vary in length, angle and curving. Those waves occurs as a result of continuing patter of vehicles that have to break for vehicles in front of them. In graph 3.2 where the x-axle represents time and the y-axis represents a position on the road, it's possible to see that the first wave begins at segment 25 and continues to about segment 20 and that this occurs between sample 180 to 300 (where a sample is taken every tenth second). In other words, the beginning of the congestion starts earlier and earlier on the road over time. This means that more vehicles are entering the congested area than leaving it, which will produce this kind of wave. The simple reason for this, as mentioned before, is that vehicles have to brake for other vehicles in front of them. One way to reduce these waves is to inform other cars downstream to reduce their speed before the congestion. In that case the congestion has an chance to be resolved before these informed vehicles reach the congested area.

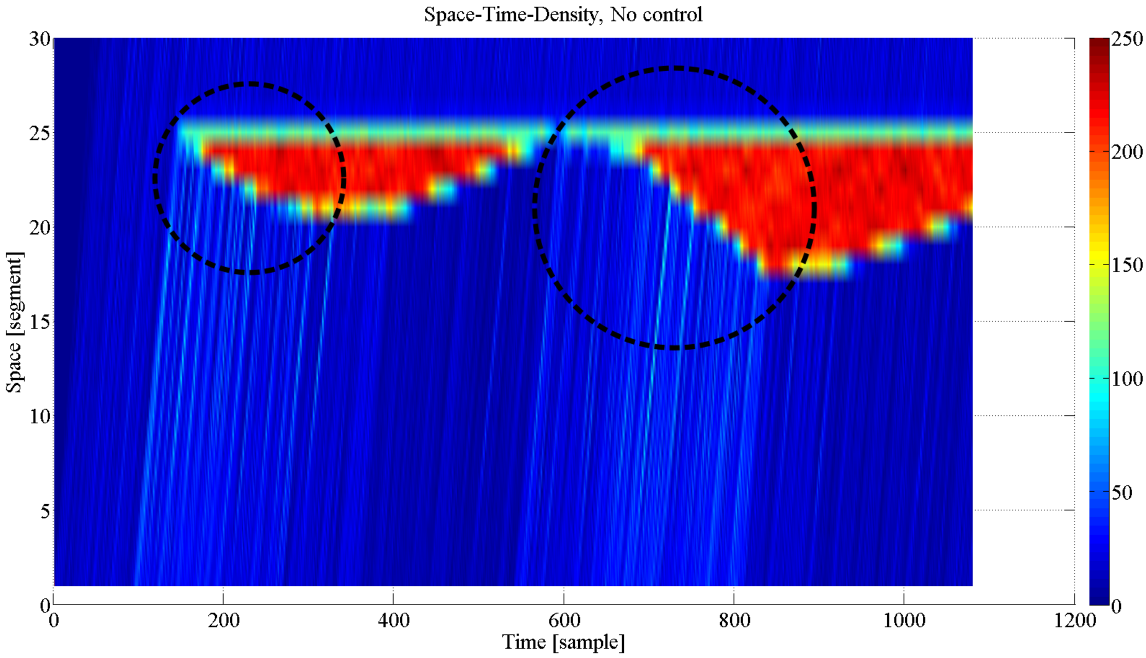


Figure 3.2: A space-time-density graph were the backward propagating waves are marked.

4

Results

The first two sections of this chapter describes the simulation setup both input-wise and network-wise. The following sections presents the result from a simulation from each of the following control methods: uncontrolled, Adaptive Cruise Control (ACC), Variable Message Sign (VMS) control and Combined controller (which as earlier described is the ACC and the VMS working together). All the simulations presented in this chapter has been done with the same input and the same network unless stated otherwise. Important to note is that since the controllers implemented in this thesis are used to verify the code skeleton no conclusions of how controllers can improve a traffic network in general can be taken from these results.

4.1 Simulation setup - Input

In total 22000 vehicles was requested to enter the road during the simulation. These vehicles were requested to enter the road during a total time period of three hours (10800 seconds). These three hours were divided into twelve fifteen minutes (900 seconds) periods. During each of these periods different amount of vehicles were requested to enter the road. The goal was to create a high demand during two parts of the simulation to make sure a congestion would occur which, when analysing the data afterwards, would result in a visible backward propagating wave. An exact description of when and how many vehicles that were requested to enter the network at each fifteen minute period can be seen in figure 4.1. The initial desired speed of vehicles entering the network were as following: cars 120 km/h, trucks 80 km/h and buses 90 km/h. The percentage of vehicles entered the network were cars 55%, trucks 25% and buses 20%. Important to note is that cars equipped with ACC drove very close to 120 km/h while the other vehicles that entered the network didn't because some of them wasn't allowed to drive that fast and also their driver model was trying to behave like a real human driver. PTV describes their traffic flow model like this; "The traffic flow model in VISSIM is

1 HOUR	Time [seconds]	0 - 900 (900 sec)	900 - 1800 (1800 sec)	1800 - 2700 (2700 sec)	2700 - 3600 (3600 sec)
	Input [nr of vehicles]	1500 (1500 vehicles)	3000 (4500 vehicles)	2500 (7000 vehicles)	1500 (8500 vehicles)
2 HOUR	Time [seconds]	0 - 900 (4500 sec)	900 - 1800 (5400 sec)	1800 - 2700 (6300 sec)	2700 - 3600 (7200 sec)
	Input [nr of vehicles]	1000 (9500 vehicles)	1000 (10500 vehicles)	2000 (12500 vehicles)	3000 (15500 vehicles)
3 HOUR	Time [seconds]	0 - 900 (8100 sec)	900 - 1800 (9000 sec)	1800 - 2700 (9900 sec)	2700 - 3600 (10800 sec)
	Input [nr of vehicles]	3000 (18500 vehicles)	1500 (20000 vehicles)	1000 (21000 vehicles)	1000 (22000 vehicles)

Figure 4.1: A table over the amount of vehicles that were requested to enter the network each fifteen minutes period during a total time span of three hours.

a discrete, stochastic, time step based, microscopic model with driver-vehicle-units as single entities. The model contains a psycho-physical car following model for longitudinal vehicle movement and a rule-based algorithm for lateral movements. The model is based on the continued work of Wiedemann" [17, p.126]. The ACC-equipped cars ability to achieve a driving speed close to the desired speed was an important factor when using ACC equipped cars to improve the performance of the network, even tough the ACC controlled cars had more than one parameter which could be modified to be "better" than a human driver. Important to note is that at speeds equal or lower than 10 m/s the VISSIM driver model took over the control of the ACC cars.

4.2 Simulation setup - Network

Each simulation used the same network. The network was divided into several 500 meter segments, referred to as links in VISSIM, where some of them were without variable message signs and some with variable message signs. All segments, except one, had three lanes. The exception was a segment close to the end of the road which only had two lanes. The reason for this was that it is easier to make a congestion this way. Table 4.2 presents a detailed explanation of the network layout.

Segment numbers	Length [meters]	Number of lanes	VMS (Variable Message Sign)
1 - 7	3500	3	NO
8 - 24	8000	3	YES
25	500	1	NO
26 - 30	2000	3	NO

Figure 4.2: A table of the network layout.

4.3 The uncontrolled simulation

The uncontrolled simulation was the reference to compare the controlled simulations with. The key numbers from this simulations can be seen in figure 4.6. Readers may notice that the number of vehicles which have left the network are fewer than the input. The reason for this is that there still were vehicles on the road when the simulation ended. Since the road was considerably congested when the simulation stopped, the discrepancy between the total input and the vehicles that had left the network was substantial. The table also show that there were many stops for each vehicle. A stop is defined as when a vehicle have the speed of zero [17, page 434] hence the conclusion is that there have been congestions on the road during the simulation run. By studying the space-time-density graph 4.3 (created with data from the simulation) and the space-time-speed graph 4.4 (generated with data from the same simulation) it is possible to see that, during two time periods, both a clear increase in density and a heavy decrease in speed occurred at same time period. Those simultaneous changes were an indicator on that a congestion on the road existed during these periods.

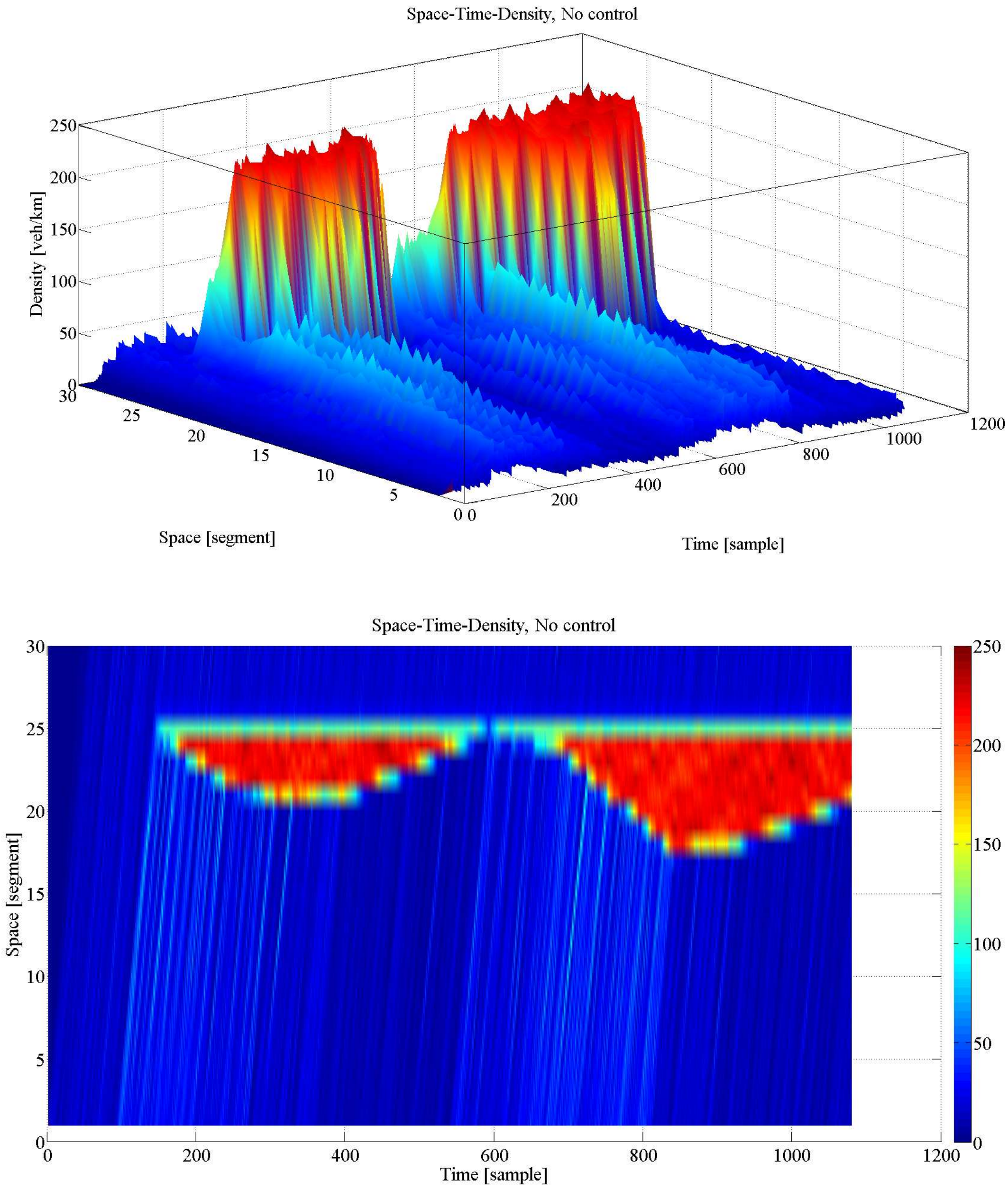


Figure 4.3: The space-time-density graph for the uncontrolled simulation from a 3D and a 2D view.

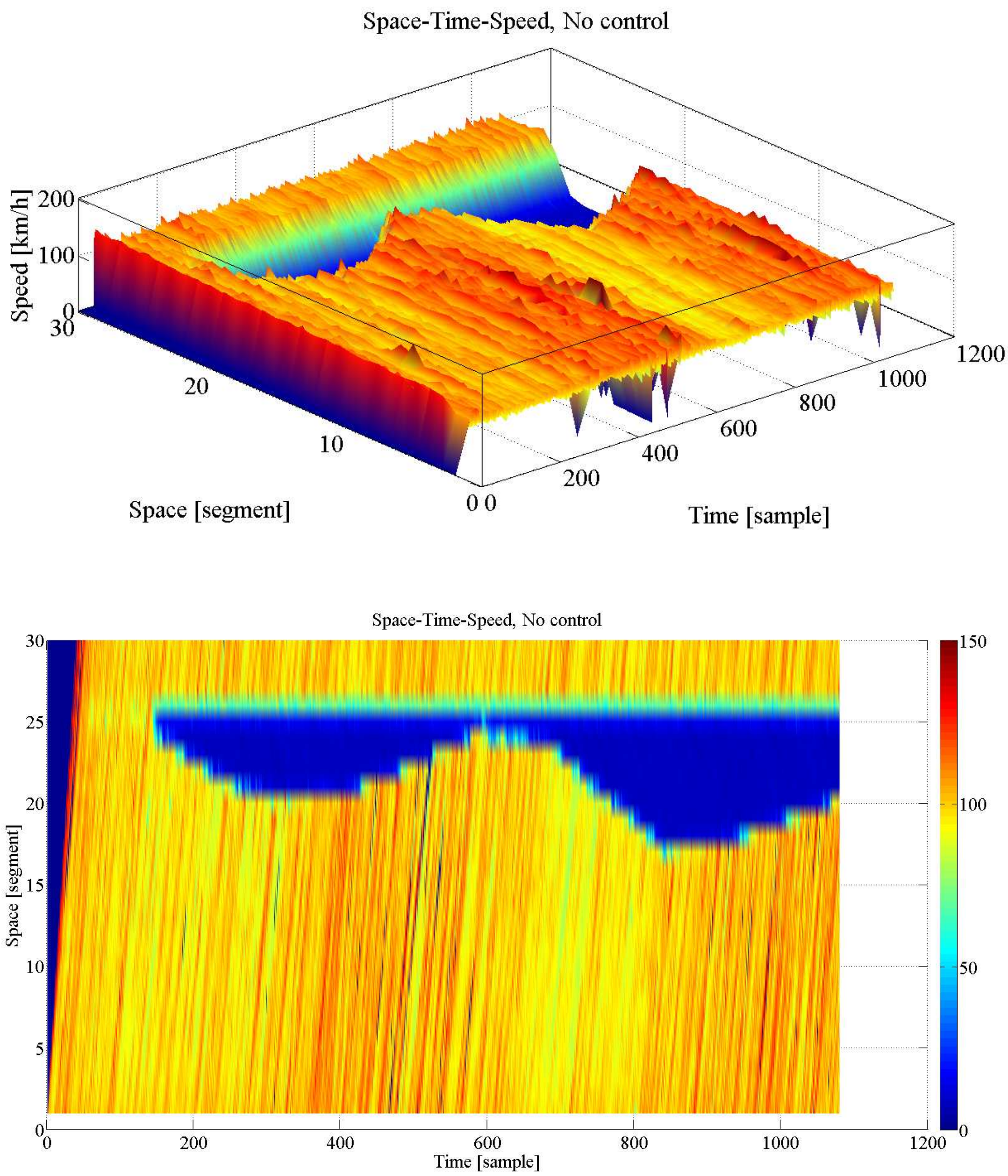


Figure 4.4: The space-time-speed graph for the uncontrolled simulation from a 3D and a 2D view.

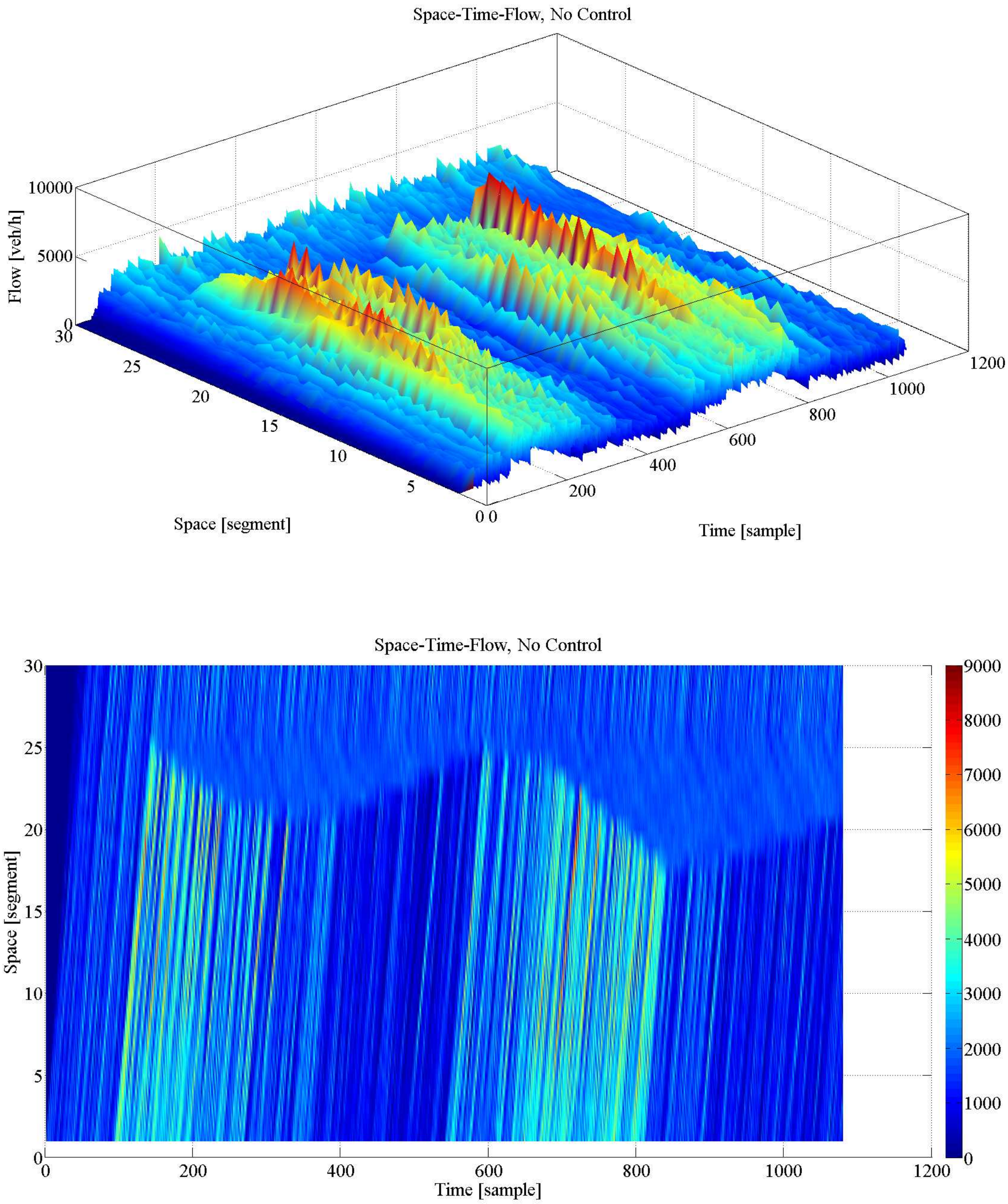


Figure 4.5: The space-time-flow graph for the uncontrolled simulation from a 3D and a 2D view.

Information	Uncontrolled simulation
Average number of stops per vehicle [decimal number]	53,930
Average speed [km/h, decimal number]	44,555
Total travel time [hours, decimal number]	1773,846
Number of vehicles that have left the network [integer number]	4833
Number of stops [hours, integer number]	294349

Figure 4.6: Key numbers from the uncontrolled simulation.

4.4 VMS controller

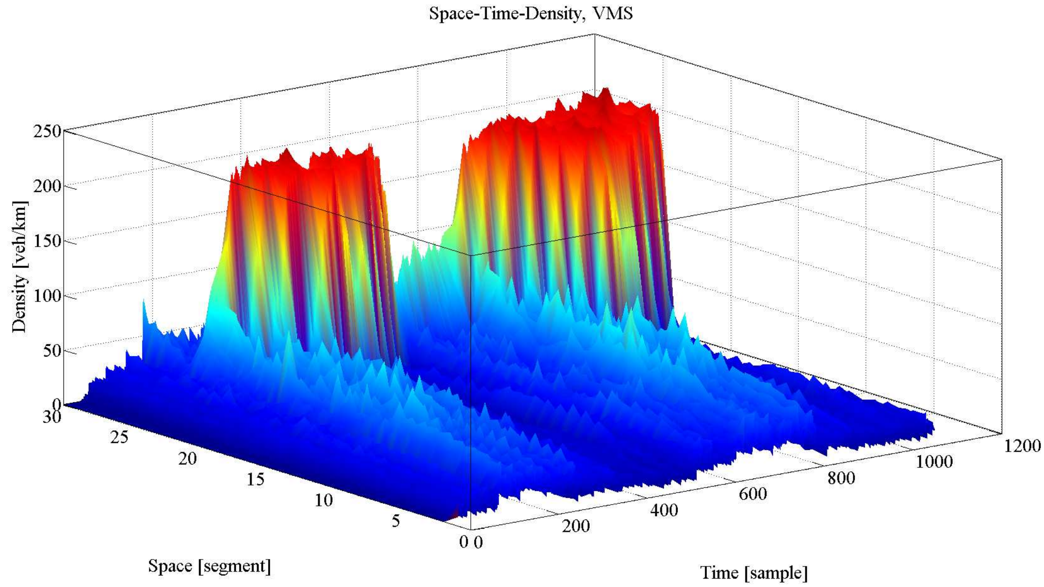
Parameter settings for the VMS controlled simulation,

k = 6.0

rho_cr = 10.0 [veh/km]

updateFreq = 10.0

Space-time-density 4.7, space-time-speed 4.8 and the space-time-flow 4.9 graphs are generated from the VMS simulation data. The result table 4.10, with some parameters from the VMS simulation also contained the corresponding parameters from the uncontrolled simulation. This way it was easier to see the difference in performance caused by the VMS controller.



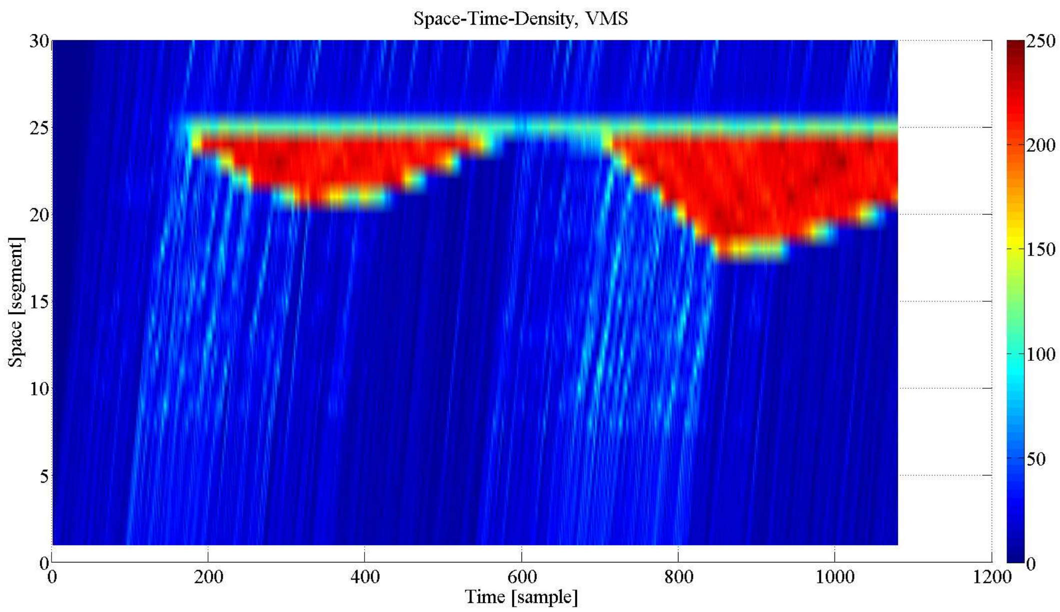


Figure 4.7: The space-time-density graph for the VMS controlled simulation.

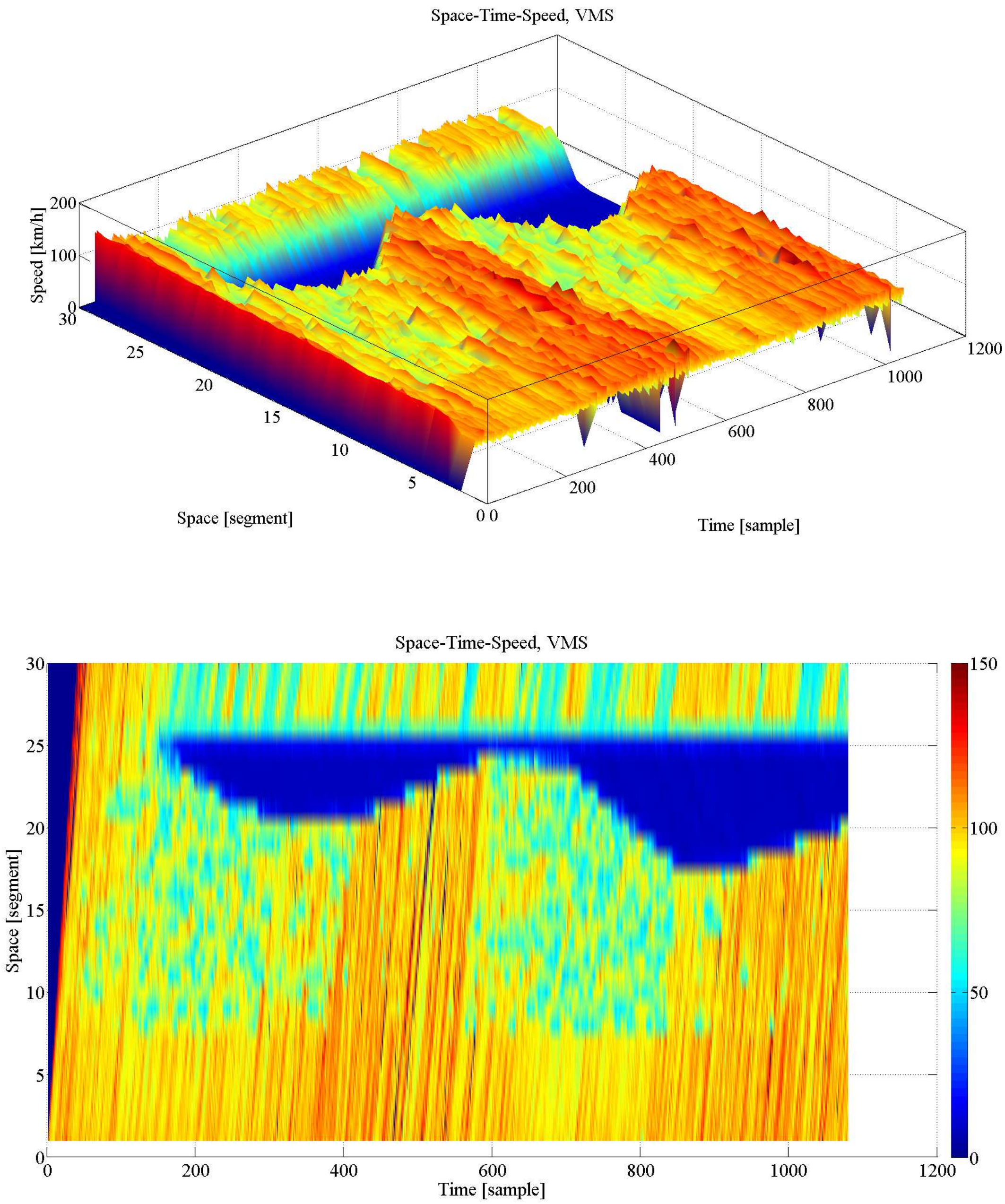
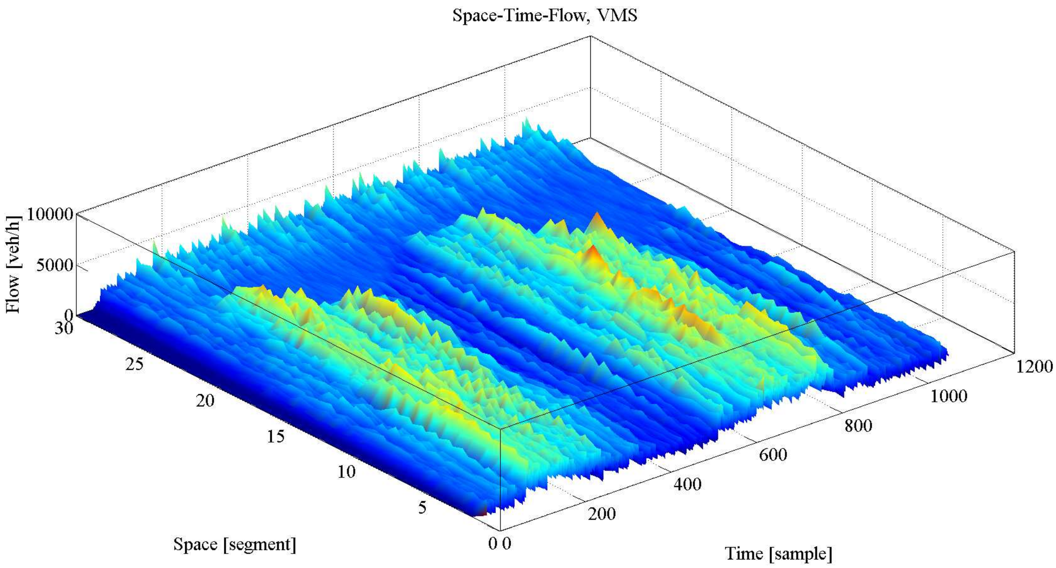


Figure 4.8: The space-time-speed graph for the VMS controlled simulation.



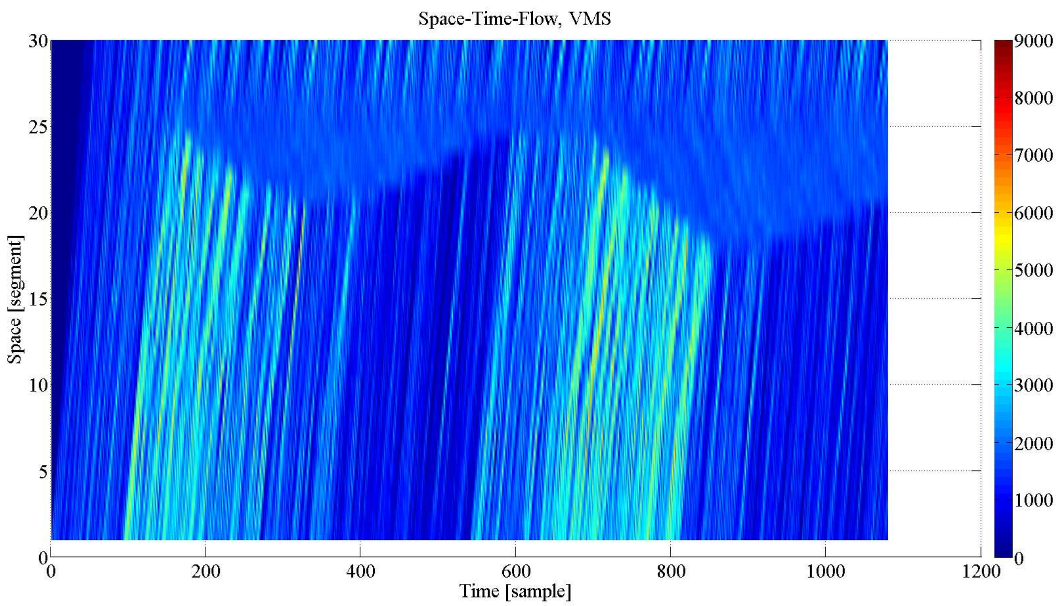


Figure 4.9: The space-time-flow graph for the VMS controlled simulation.

Information	VMS	No Control
Average number of stops per vehicle [decimal number]	51,053	53,930
Average speed [km/h, decimal number]	42,947	44,555
Total travel time [hours, decimal number]	1838,265	1773,846
Number of vehicles that have left the network [integer number]	4810	4833
Number of stops [hours, integer number]	278650	294349

Figure 4.10: A comparison of key numbers between Variable Message Sign (VMS) controlled simulation and uncontrolled simulation. The biggest difference is the change in number of stops.

4.5 ACC

Parameter settings for the ACC controlled simulation,

k = 3.0

ACC_OnOff = 10.0 [m/s]

ACC_percentage = 50% (of the total input was equipped with ACC)

updateFreq = 10.0

The space-time-density 4.11, space-time-speed 4.12 and the space-time-flow 4.13 graphs have been generated from ACC simulation data. The result table 4.14 with some parameters from the ACC simulation also contained the corresponding parameters from the uncontrolled simulation. This made it easier to see the difference in performance caused by the ACC controller.

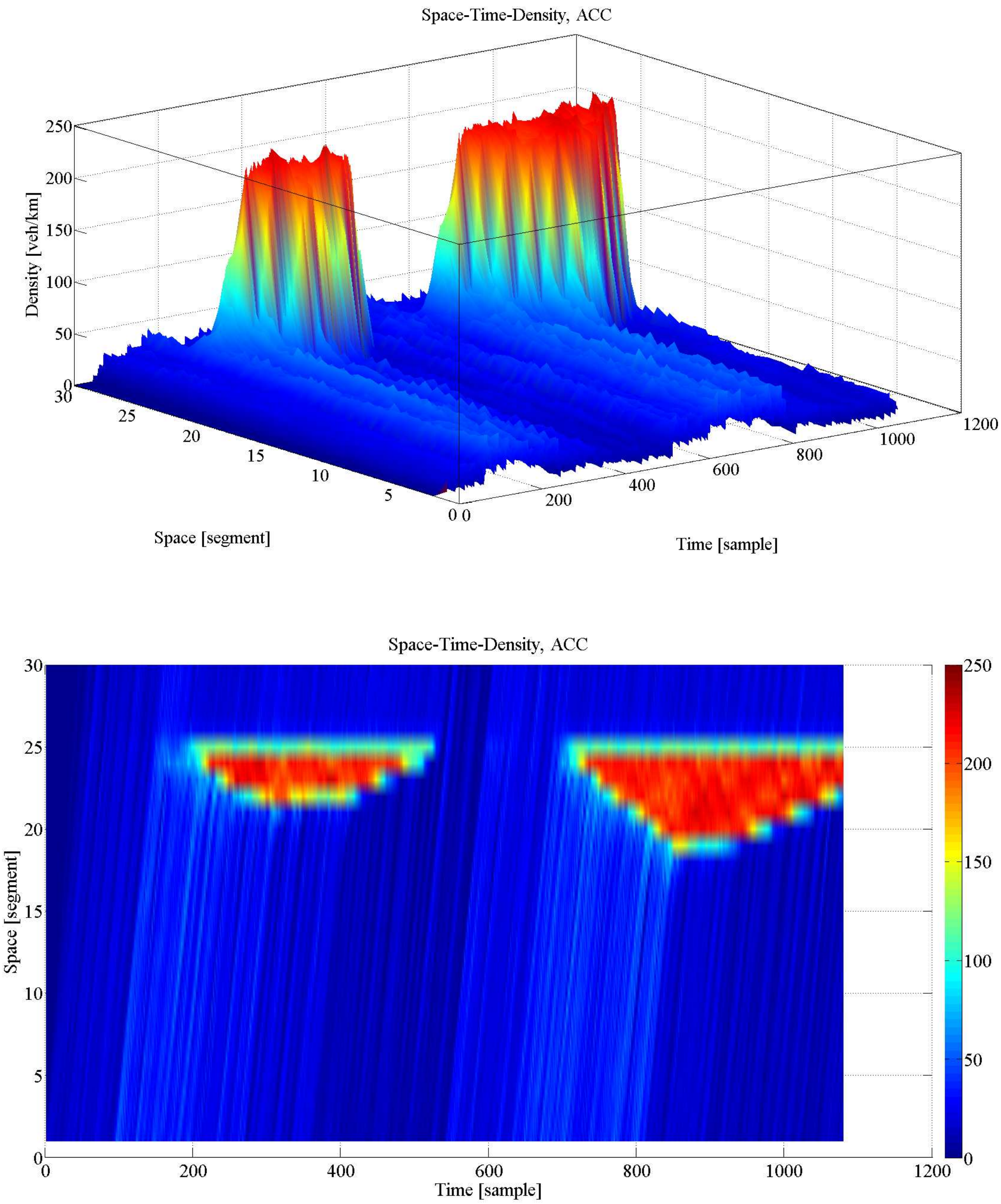


Figure 4.11: The space-time-density graphs for the ACC simulation.

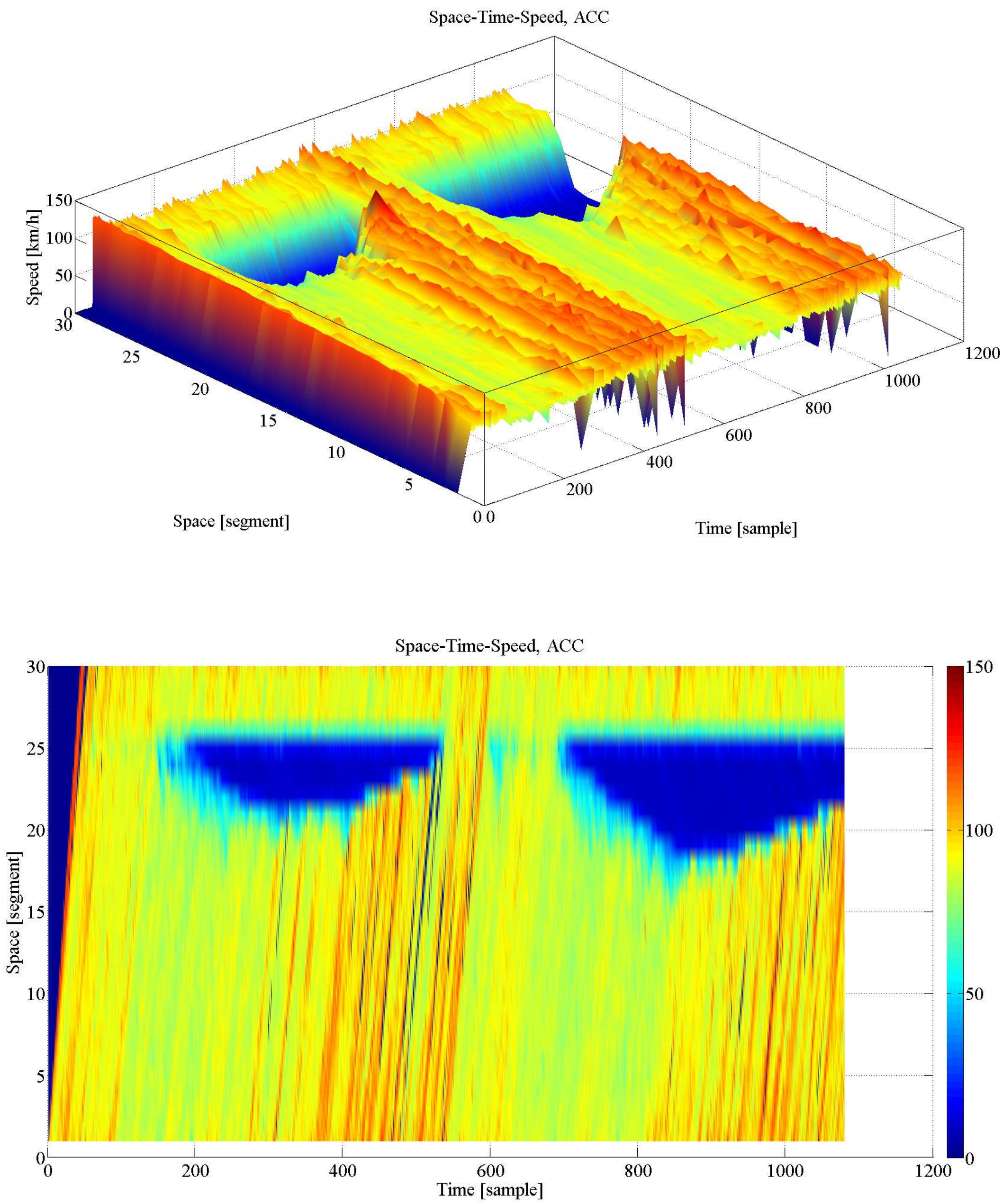


Figure 4.12: The space-time-speed graph for the ACC simulation.

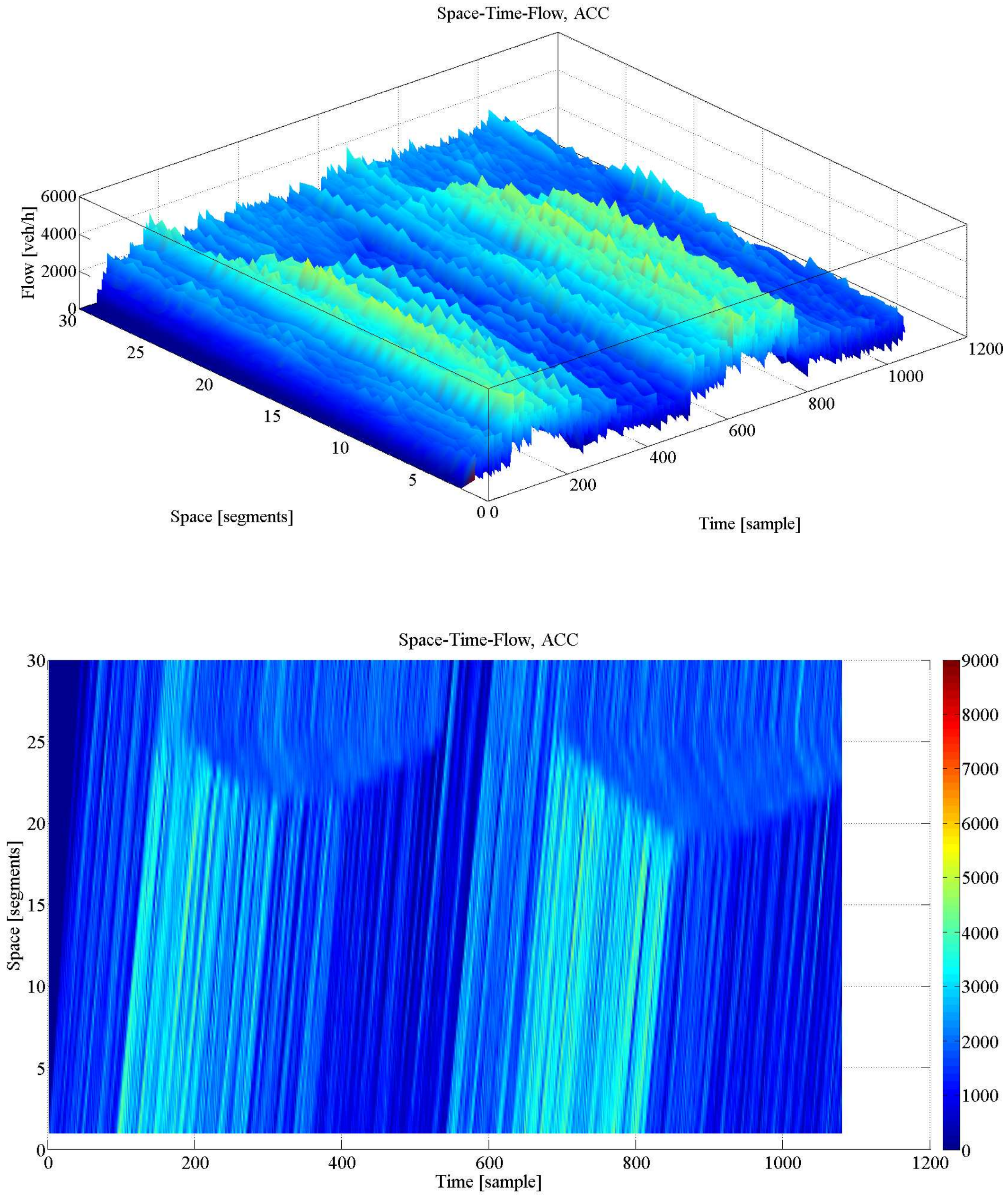


Figure 4.13: The space-time-flow graph for the ACC simulation.

Information	ACC	Uncontrolled
Average number of stops per vehicle [decimal number]	27,326	53,930
Average speed [km/h, decimal number]	50,975	44,555
Total travel time [hours, decimal number]	1563,067	1773,846
Number of vehicles that have left the network [integer number]	4981	4833
Number of stops [hours, integer number]	149147	294349

Figure 4.14: A comparison table between the ACC and the uncontrolled simulation. The improvement in performance using a ACC controller is, as the table shows, significant.

4.6 Combined controller

Parameter settings for the combined controlled simulation,

k(for the VMS) = 0.5

k(for the ACC) = 5.0

rho_cr = 10.0 [veh/km]

ACC_OnOff = 10.0 [m/s]

ACC_percentage = 50% (of the total input was equipped with ACC)

updateFreq = 10.0

The space-time-density 4.15, space-time-speed 5.5 and the space-time-flow 4.17 graphs have been generated from the combined controller simulation data. The result table 4.18 with some parameters from the combined controlled simulation also contained the corresponding parameters from the uncontrolled simulation. This made it easier to see the difference in performance caused by the combined controller.

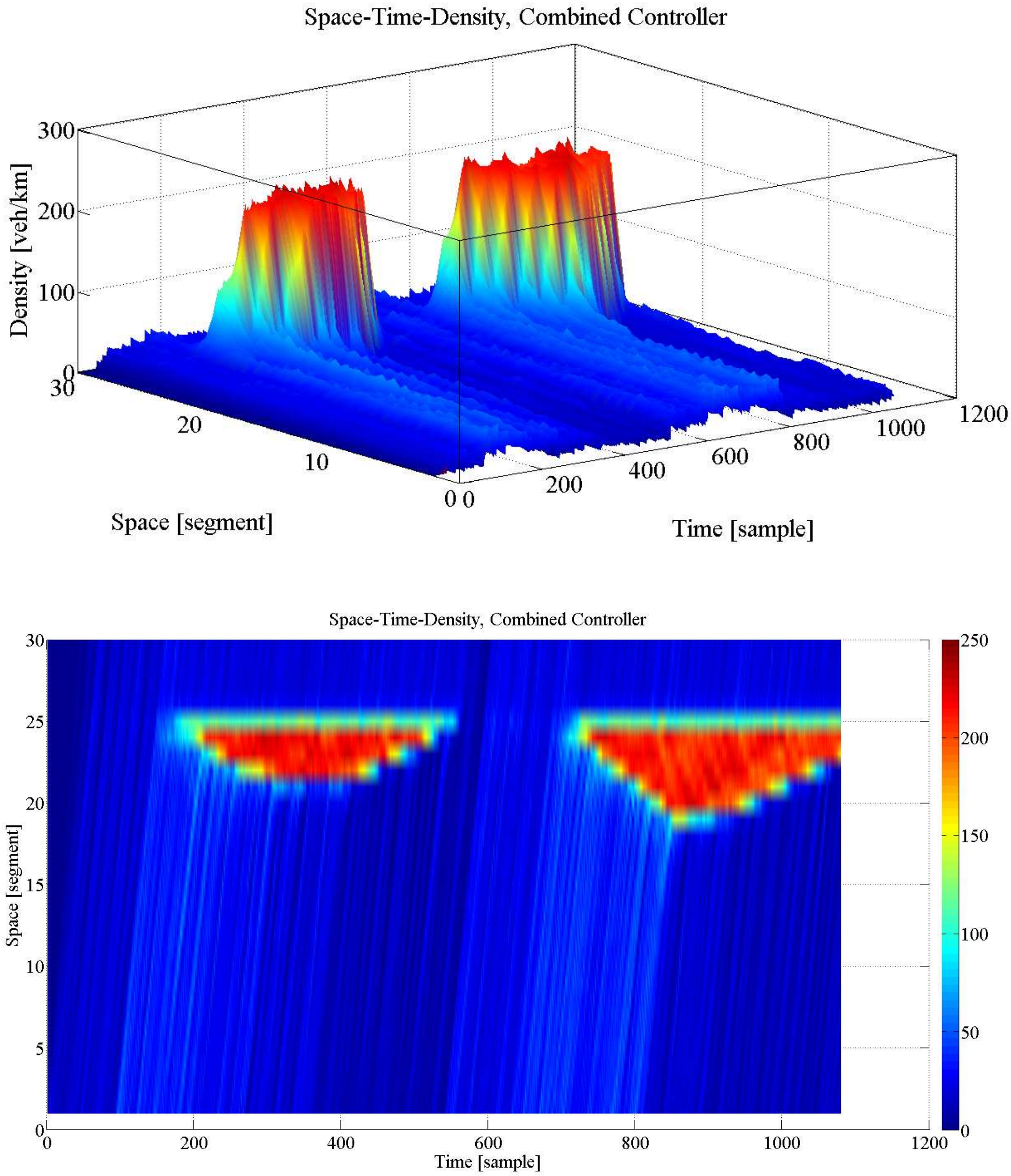


Figure 4.15: The space-time-density graphs for the combined controlled simulation.

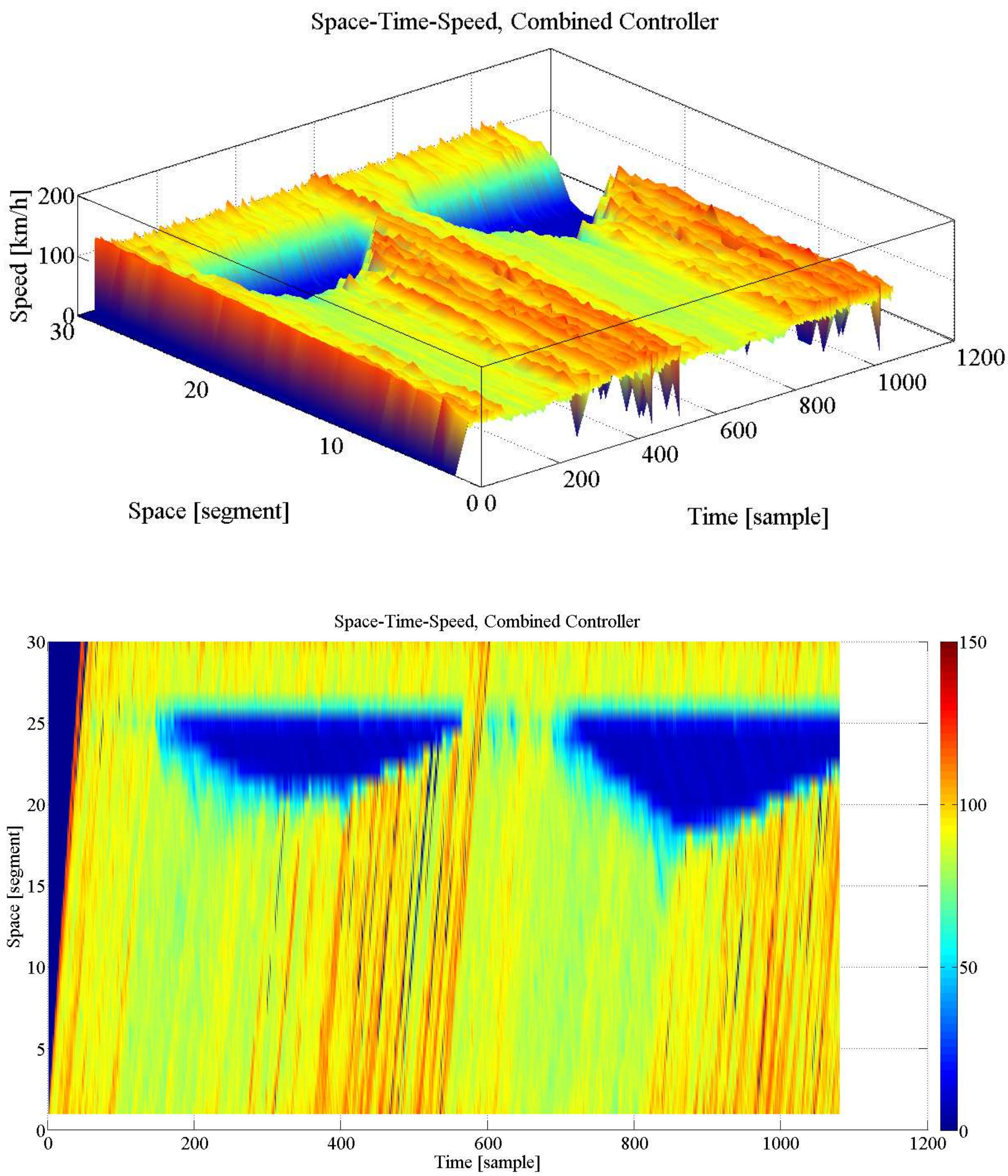


Figure 4.16: The space-time-speed graph for the combined controlled simulation.

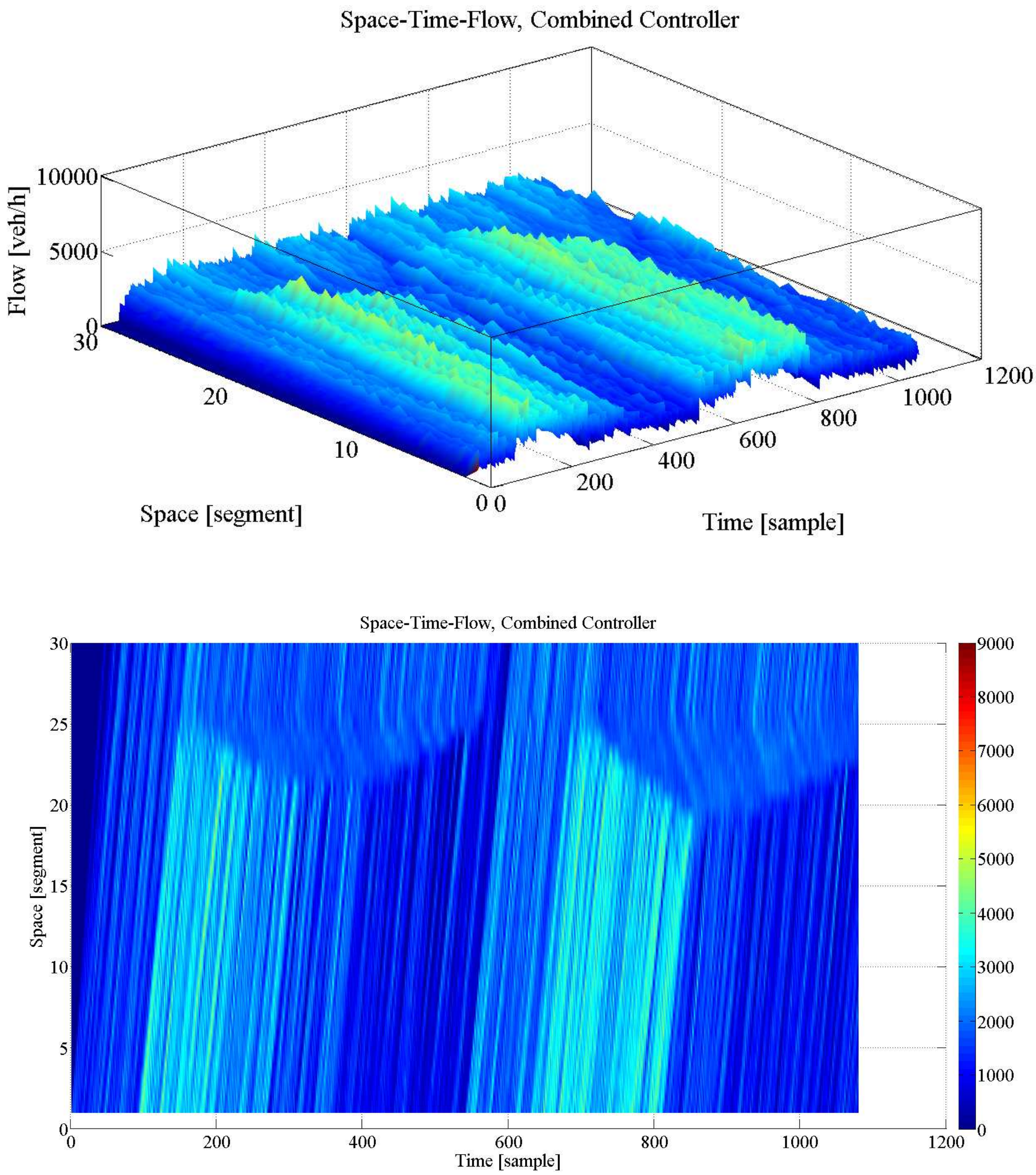


Figure 4.17: The space-time-flow graph for the combined controlled simulation.

Information	Combined controller	No Control
Average number of stops per vehicle [decimal number]	30,856	53,930
Average speed [km/h, decimal number]	50,219	44,555
Total travel time [hours, decimal number]	1590,409	1773,846
Number of vehicles that have left the network [integer number]	5031	4833
Number of stops [hours, integer number]	168411	294349

Figure 4.18: A comparison table between the combined controlled and the uncontrolled simulation.

5

Discussion

Here a discussion and personal aspects of the result are given. The chapter begins with a short review of the result followed by some personal thoughts. After this a section of discussion and personal thoughts about each separate simulation are given. The following sections also contains several comparisons between the simulations. Because of the time limit of this thesis the controllers implemented have been used to analyse if the code skeleton are working. In other words, no conclusions regarding how different controllers can effect the network from a performance point-of-view can be taken. As an example a combined controller (ACC + VMS) should outperform a pure ACC controller which, in the results presented during the last chapter, is not the case except when it comes to increased flow.

5.1 Review of the results

The major performance change between the combined controlled versus the uncontrolled simulation runs was that the number of stops during the combined controlled simulation was reduced with more than 42% compared to the uncontrolled. This reduction in stops did not lower the flow in the combined controlled case. In fact the flow was increased even though the difference was small, about 4% (5031 vehicles in the combined controlled case compared to 4833 vehicles in the uncontrolled case). The change in average speed between the two mentioned simulation runs was more than 12%. Even though the combined controller made a huge improvement of the performance the ACC controller itself performed even better in many aspects meanwhile the pure VMS controller performed clearly worse than both the combined controller and the ACC controller. Although it did reduce the number of stops compared to the uncontrolled simulation. The implemented VMS controller was very simple and has, as the two other controller types, a high improvement potential. For a overview table with all the simulations see table 5.1 can be The important part of the result was to see if the combined controller

Comparsion	No control	Only ACC 50%, k_{acc} = 3, ACC_{off} = 10	Only VMS k_{vms} = 6, cr_{dens} = 10	Combined control 50%, k_{acc} = 5, k_{vms} = 3 cr_{dens} = 10, ACC_{off} = 10
Average number of stops per vehicle	53,930	27,326	51,053	30,856
Average speed [km/h]	44,555	50,975	42,947	50,219
Number of stops	294349	149147	278650	168411
Total travel time [h]	1773,846	1563,067	1838,265	1590,409
Number of vehicles that left the network	4833	4981	4810	5031

Update frequence = 10 seconds

Figure 5.1: A comparison table showing the key values from the uncontrolled simulation and each controlled simulation. Also the controller parameters for each of the simulations can be seen here. Yellow coloured values are highlights the best result in each key value-category. Observe, 50% in the table where the parameters settings are stated means that 50% of the vehicles were equipped with ACC.

worked as expected, not from a simulation and traffic improvement point of view, but from a programming and usability perspective and based on the results it seems that the combined controller incorporated both the ACC controller and VMS controller. This in turn means the contribution of this thesis to the research area of traffic control has been achieved and can help new researcher to try out new, and hopefully better, combined controllers. Further more focus can lay on, changing one of the controllers and still do combined controller simulations and how the the change, of one of the controller types, affects the results. Before looking closer on each separate simulation a personal feeling is that even if the combined controller didn't perform as good as hoped it did improve the controlled contra the uncontrolled simulation result significantly. And, as mentioned earlier, most important is that the results indicates that the written code skeleton managed to combine two controllers and this were, along making the control methods as separate modules, the main goal of this thesis.

5.2 Uncontrolled simulation

All three graphs from the uncontrolled simulation show that there were two congestions during the simulation time. This was important to have since it made it possible to see if controlled simulations could reduce the severity of a congestion. The first congestion also seemed to sync up quite well with the increased input (3000 vehicles) between 900 to 1800 sec which corresponded to sample 90 - 180. A delay was to be excepted due to the reason that the vehicles had to reach the area and build up a to high throughput demand on the reduced capacity link (one lane instead of three) before a congestion appeared at the start of the reduced capacity link. At a 3600 to 5400 seconds period, the input was reduced to 1000 vehicles. This explains the decline of the congestion which started at approximately sample 410 (4100 seconds). This means that the setup was creating congestions as expected and therefore was usable for all the remaining simulations. This was of importance because if there were no congestions during the simulation followed by a resulting backward propagating wave, the controlled simulations probably wouldn't have had the same effect on the network performance. And this applies especially for the VMS controller since it adapting itself based on the density of the road and without any congestion the VMS wouldn't affect the system as wanted which is assist in resolving congestions that occurs on the road[2]. The time-speed-flow graph 4.5 shows how the flow is affected by the congestion. The reason to that the flow increased before the congestions is that the input was highly increased and since the three lanes of the road still had capacity to handle the high input demand, the flow increased on these areas. A dangerous effect of the congestion was that the flow changed drastic from high to low. Since vehicles approached the congested area in a driving situation where the flow was high, they had to brake to adapt to a driving situation were the flow was lower. This in turn leads to an increased risk for collision. As graph 4.4 shows the average speed before the congested area, on a short time period, goes from approximately 110 - 130 km/h to 0 - 30 km/h which means that the vehicles had to brake hard which increased the risk of accidents. Compared to the combined controlled simulation it was possible to see a less drastic change in flow before the congested areas. The conclusion that can be drawn from this is that the uncontrolled simulation resulted in many stops and at least one serious congestion with a backward propagating wave. This means that the network setup and input was a good system to use when testing control methods since there was a bad driving situations on the road.

5.3 VMS

The result from the Variable Message Sign (VMS) controller simulation was, in the big perspective, better than without control. Even though the flow in the VMS simulation was a little lower compared to the uncontrolled simulation the difference in number of stops was clearly higher in the uncontrolled simulation. As table 4.10 shows the amount of vehicles which left the network were 4810 in the VMS simulation while it were 4833 in the uncontrolled simulation. However, when looking at the number of stops the VMS simulations had 278650 stops and the non controlled had 294349 instead. By applying

[2]It's possible to make the VMS affect the traffic in uncongested conditions. E.g. by reducing the critical density limit. And it was probably many situations were my critical density limit triggered the VMS even though it wasn't necessary or even good at the time.

simple math the big picture is easier to see.

$$\frac{NoControl_{carsLeftNetwork} - VMS_{carsLeftNetwork}}{NoControl_{carsLeftNetwork}} = \frac{4833 - 4810}{4833} \approx 0.5\%$$

$$\frac{NoControl_{nrOfStops} - VMS_{nrOfStops}}{NoControl_{nrOfStops}} = \frac{294349 - 278650}{294349} \approx 5.3\%$$

According to the equations above the flow during the VMS simulation was reduced by 0.5% compared to the uncontrolled simulations by the same time the number of stops was lowered by more than 5%. Since the table shows that the average speed has been reduced by 1,608 km/h in the VMS simulation which represent a change of approximately 3.6% in average speed between the two simulations, a logical conclusion is that the density has been increased. Otherwise the flow would have suffered a higher reduction in the VMS simulations. Another important thing to notice while studying the space-time-speed graph 4.8 for the VMS controller is all the light-greens (indicating lower speed) notches in the area before each congestion. The frequent changes from high speed (yellow) to low speed (light-green) indicates that the VMS signs were "jumping" causing these sudden changes in speed. This is not a desired behaviour since it causes unwanted driving conditions and an increased risk for collisions. By comparing the 2D graph of the uncontrolled simulation with the respective graph of the VMS controller these notches are easy to notice, see figure 5.3, which gives a clear picture of that something is different from the uncontrolled simulation. Even though these speed notches were not good it at least shows that the VMS worked. An explanation to the speed notches phenomena created by the VMS controller is the fact that the P-regulated VMS has a limited number of speed steps. As an example the speed can't be set to 86 km/h, it can only be set to predefined speeds which during the simulations in the thesis were 50, 70, 90, 110 and 120 (all in km/h), so when the input increases the density of the segments follows which in turn can result in a change of the VMS signs and their respective set speed. This means that if a cluster of vehicles enters a previously empty segment the density changes drastically making the VMS set a low speed and forcing the new incoming vehicles to drive very slowly. After a period of time all the previously mentioned incoming vehicles have passed the area in a slow pace and once again the speed is set to high. This means that vehicles that have entered the segment can drive fast even if they drove slow on the segment before making them getting closer to the cluster ahead. As the space-time-density 4.7 for the VMS simulation indicates, the VMS lowered the average speed with the side effect to clump up vehicles. One theory that could explain why the VMS simulation graphs looks like they do is that the average speed on segments temporarily was lowered and the vehicles slowly got closer together which increased the density on the segments. Then the vehicle cluster at the front of this new mini-congestion could start to accelerate to high speeds which in turn allows the next cluster to do the same when they reach the next speed sign. In this way the mini-congestion disappears or, more likely, moves until a new mini-congestion forms. The average speed change may then be explained by the fact that one cluster of vehicles can drive away from the other clusters of vehicles giving segments this fast moving cluster passes a high average speed.

This means that some segments will experience a temporarily large decrease in average speed while other segments will experience the opposite. Figure 5.6 shows this theory. In this figure, three speed levels exists. The vehicle clusters do not blur into each other and they do not change their compositions, meaning that each cluster remains constant all the time. In reality the vehicles in a cluster drive with a high difference in speed due to the fact that they act like human drivers which seldom drive in the exact same speed. Trucks and buses have a lower top speed so after a while two vehicle clusters will meet up and become one, or more, vehicle clusters forming new vehicle composition. Finally, it might been that a higher resolution, in other words, more predefined speed steps could have made the VMS controller better.

5.4 ACC

The best result during the simulations in all aspects except for flow was achieved when using the ACC controller. The reason for this may be many, but one benefit that the ACC controller has is that it basically is a PI-controller. The VMS controller is a P-controller with limited speed steps (as an example the speed 85 km/h is not possible to set on the VMS sign). However, it is not any problem to set a speed like 85 km/h by using an ACC controller. As figure 4.11 shows, the density on the segments are at the same level during the whole simulation without any serious spikes except for the segments where the congestions are created. Unlike the uncontrolled simulation (see figure 4.14) the two serious congestions during the ACC simulations were both resolved faster in time and in space. In other words the length of the backward propagating wave was shorter and the number of vehicles involved in the most serious congestions (red area) were fewer. This is probably a part of what is being reflected in the comparison table (see table 4.14) between the two different simulations. Another interesting comparison is between the flow of the uncontrolled contra the ACC controlled simulation. As the comparison table 4.14 shows, the difference in flow is not as big as the difference in number of stops between the two simulations, even though the flow was notably higher in the ACC controlled simulation. However, the difference in the space-time-flow graphs for the two simulations were quite immense. At this comparison between the two simulations (see figure 4.13) one can see that the flow is much more stable and smooth in the ACC controlled simulation. There are no clearly deviant spikes in the ACC graph, but the uncontrolled case have some major spikes in flow. This means that there are cluster of cars on the road with either a high density, a high speed or both. These sudden changes in flow also leads to a more unstable driving condition with fast decrease in speed time to time which is shown in graph 4.4. Figure 4.12 shows a time-space-speed graph that can provide a deeper understanding of the relation and driving experience between an ACC controlled and an uncontrolled simulation. As the time-space-speed graph shows the average speed was lower in the ACC case which largely depends on the fact that the ACC cars were limited to 120 km/h. This means that the flow was increased even tough the average speed was lower. This in turn implies that the density has to be higher. In other words, by keeping a high density, low speed and adapt the speed to the car in-front regularly at an appropriate distance instead of fast when close, a higher flow and a better driving condition can be achieved. As mentioned in the beginning of the ACC controller result section, the flow of the combined controller was better than for the ACC controller. A reason for this may be the VMS:s capability to change the speed of all the vehicles on the road before an upcoming congestion which in turn results in a reduced average speed of all the vehicles.

5.5 Combined controller

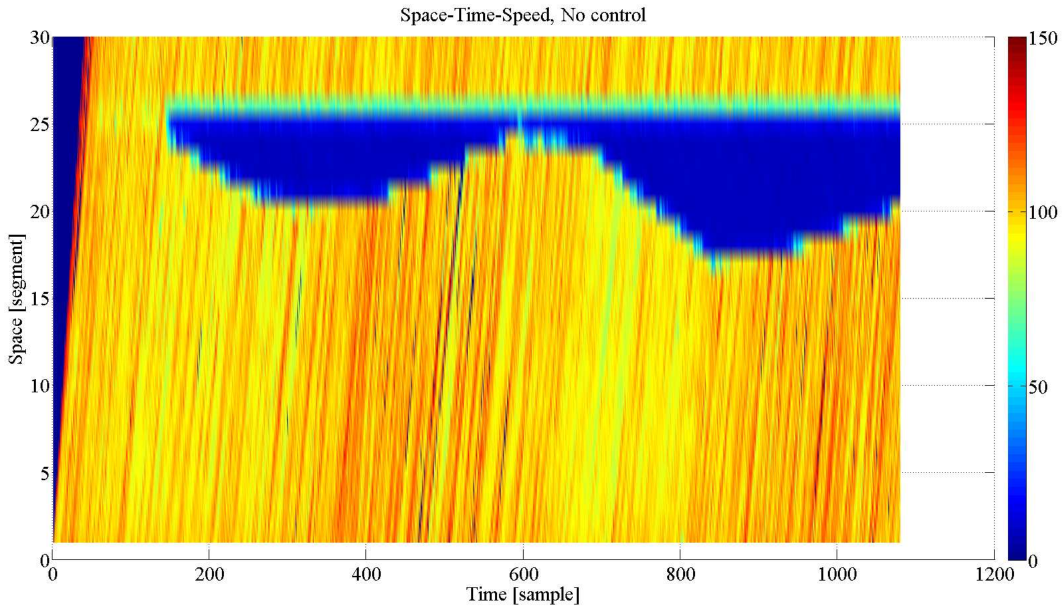
Compared to the ACC controlled case, the only improvement for the combined controller is the flow. In all other aspects the ACC controlled simulations out-performed the combined controller. The reason for this is probably that the VMS controller are performing bad making speed notches in the traffic and on this way reducing the ACC controller in a negative way. From a logical point of view a controller provided with more information and possibilities to affect a system should perform better but since the VMS are bad the result is still that the only ACC controller is better. As mentioned in 5.3, the VMS controller did improve the network performance compared to the uncontrolled case but as mentioned in 5.4 the ACC controller did improve it significantly more and since the VMS affects all the vehicles on the road and also the ACC equipped vehicles it is understandable that the combined controller will be perform somewhere between (easy to see in table 5.1) these controllers. This is because of the VMS controller implementation and in a situation with a well designed combined controller incorporating a good ACC controller and good VMS controller the combined controller should perform better than a only ACC controller. So, as stated before, the simulation results from the different controllers can not be used to make any conclusions regarding the benefits of different controller types the results can merely be used to see it the code skeleton seems to work as expected. However, the case which will be used in the comparison between the combined controller and the ACC controller is the one with the highest positive difference in flow compared. Table 5.2 shows the comparison. Interesting to notice is that the average speed is lower in the combined controller simulation than in the ACC controller simulation, but still the flow is higher in the combined controller case even while the number of stops increases. A lower speed and higher flow indicates that the density needs to be higher in the combined controller case. This is a sign that the combined controller works. Since both the ACC cars and the vehicles without ACC are affected by what the signs show the density increases when there is a congestion upstream. When vehicles drive slower the space between each vehicle decreases. However, if the system can reduce the overall speed of vehicles before an upcoming congestions and therefore reduce congestions upstream, why is the combined controller performing worse than the ACC controller and especially, why is the number of stops higher? This could be explained by a comparison graph 5.4 between the of the VMS speed graph and the ACC speed graph. On the ACC simulation speed graph there are fluctuations in speed at the areas before the congestion but the intensity of the speed changes and the the resulting speed notches are much smaller and more smooth compared to the VMS speed graph at the same areas. This implies that the speed notches created by the VMS, according to theory given in the VMS discussion 5.3 described by figure 5.6, is reduced by the ACC controller. Now lets take a look at the speed graph for the combined controller 5.5. However, one can draw the conclusion that the speed notches created by the VMS part of the combined controller reduces the improvements done by the ACC part but at the same time the combined controller inherits the good part of the VMS which is the increased density. The speed notches created by the VMS could also explain why the number of stops in the combined controller is higher than in the ACC controller.

5.6 Discussion resume

The uncontrolled simulation was outperformed by all three different controllers presented in this thesis. The VMS controller caused notches in the speed while the controller was active. These notches probably depends on the simple control algorithm which basically was a P-controller and because there were a limited number of speeds possible to present on the signs. The ACC controller was the best of all the controllers except when it came to flow where the combined controller was better. The difference in flow between the combined controller and the ACC controller were almost imperceptible compared to the difference in number of stops between the two controllers. The combined controller clearly improved the network compared to an uncontrolled network but the ACC controller was better when looking at the safety aspects since the number of stops was lower in the ACC controlled simulation compared to the combined controlled. However, the important part is that the combined controller seems to incorporate both the VMS controller and the ACC controller, meaning that a successful integration between the two traffic control systems was achieved.

Information	ACC	Combined controller
Average number of stops per vehicle [decimal number]	27,326	30,856
Average speed [km/h, decimal number]	50,975	50,219
Total travel time [hours, decimal number]	1563,067	1590,409
Number of vehicles that have left the network [integer number]	4981	5031
Number of stops [hours, integer number]	149147	168411

Figure 5.2: Comparison between a combined controller and the ACC controller. In this case the flow is the only aspect where the combined controller performs better than the ACC controller.



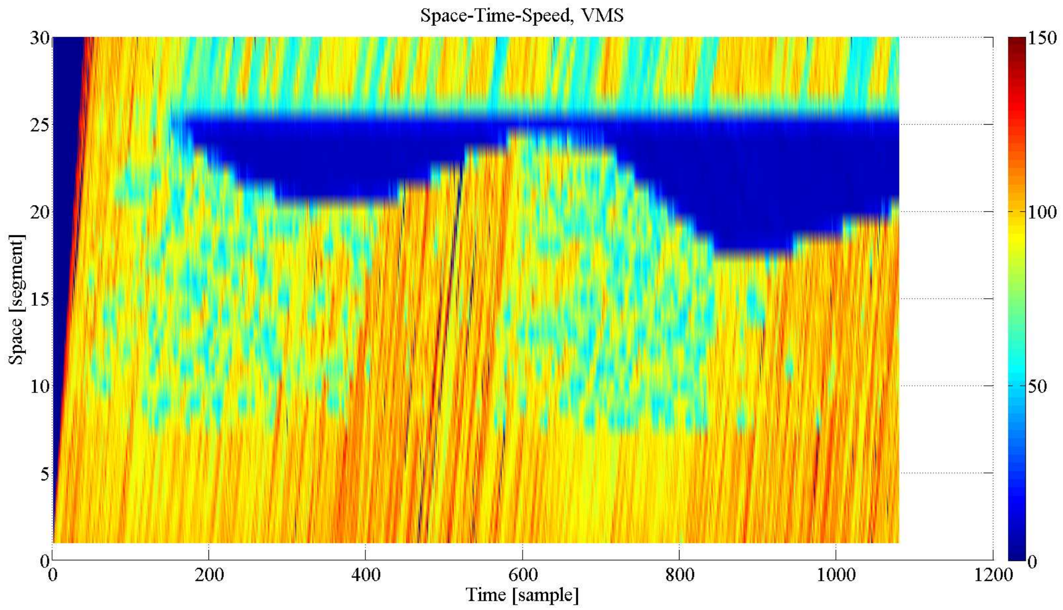


Figure 5.3: The graph at the top is generated from the uncontrolled speed simulation and the one below is from the VMS controlled simulation. The light-green notches, which don't exist on the uncontrolled case, indicates that the VMS affects the system even-though the result indicates that the VMS controller can be improved by not making the set speed cause the vehicles on the road to constantly make large speed changes.

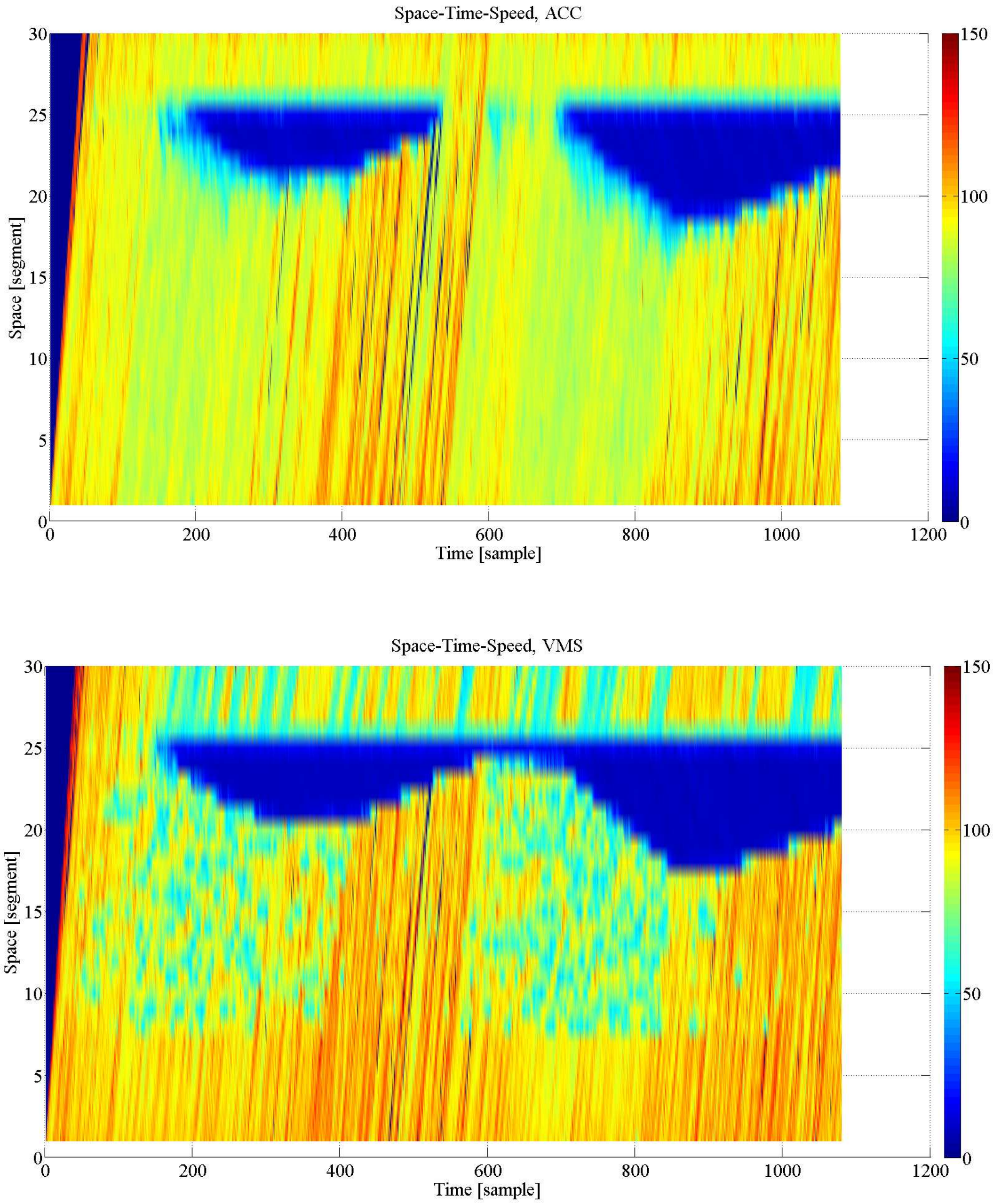


Figure 5.4: The top graph is generated from the ACC simulation and the lower graph is generated from the VMS simulation. As the graphs show there are no notches (quick speed changes) on the ACC simulation graph until the vehicles reaches the congested areas and even though there are fluctuation in speed before these areas they are much more smooth and less intense compared to the ACC simulation graphs at the same areas.

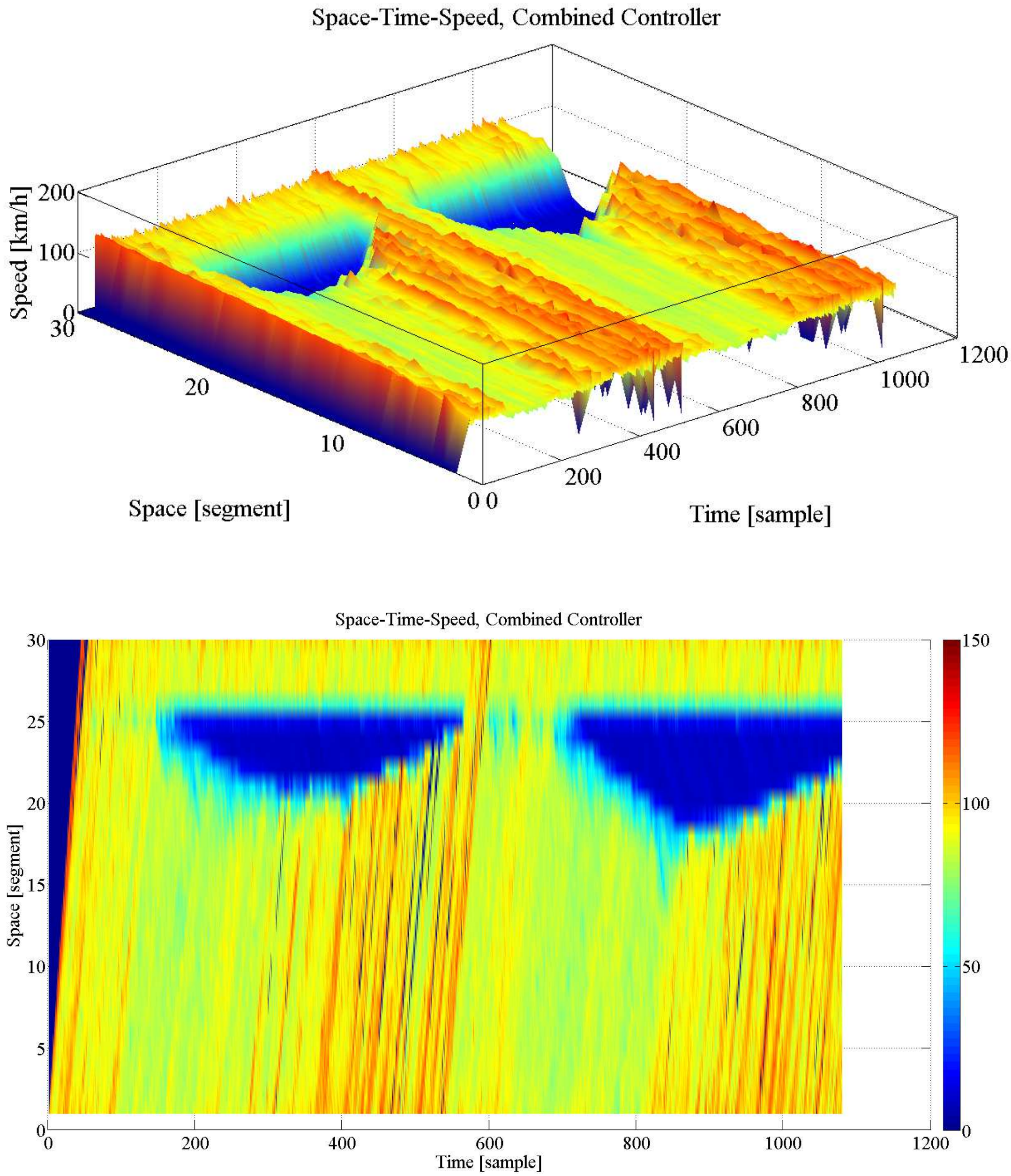


Figure 5.5: A speed graph of the combined controller simulation from a 3D- and a 2D-view.

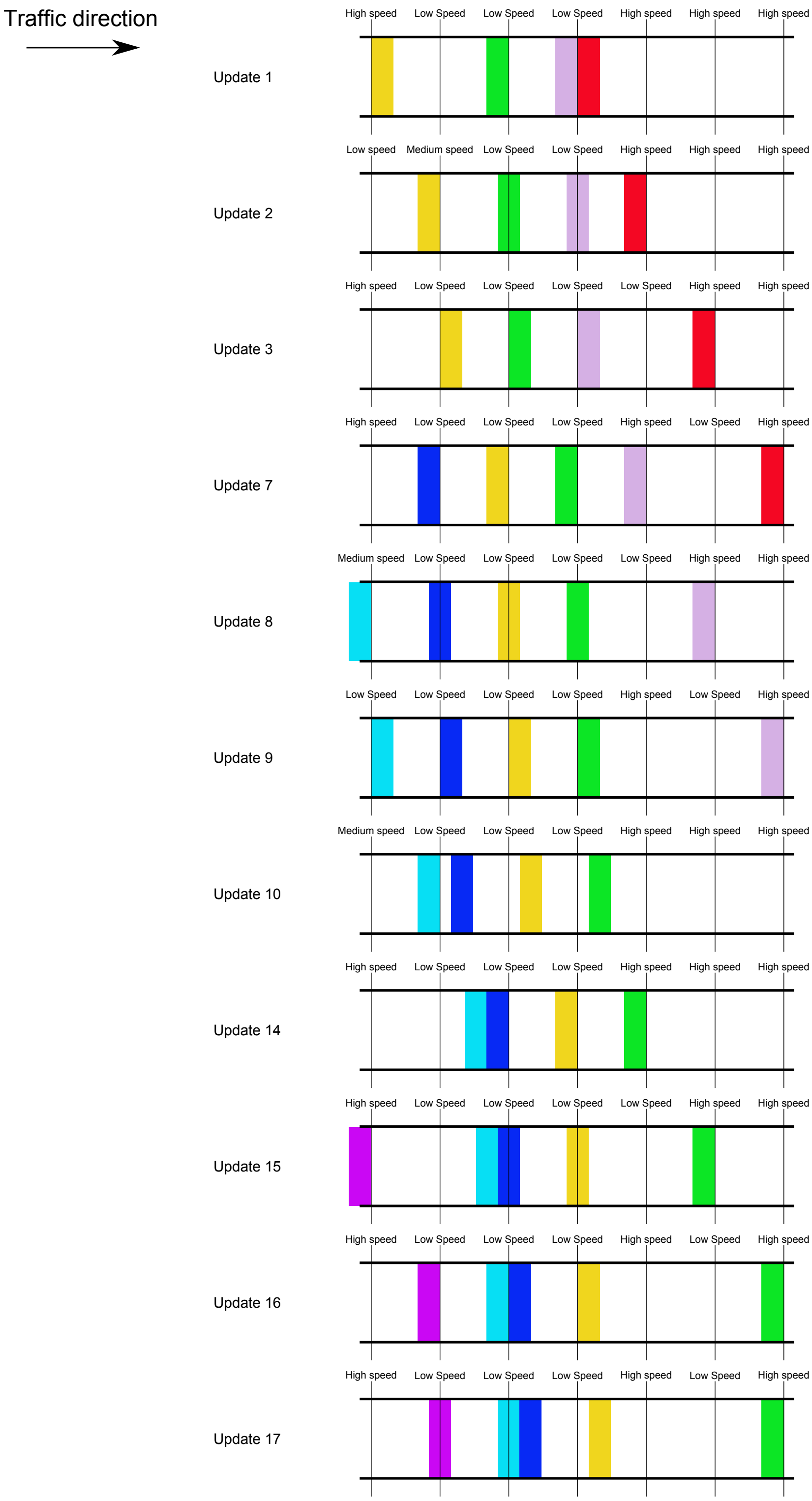


Figure 5.6: A picture explaining the theory why the VMS speed notches appears.

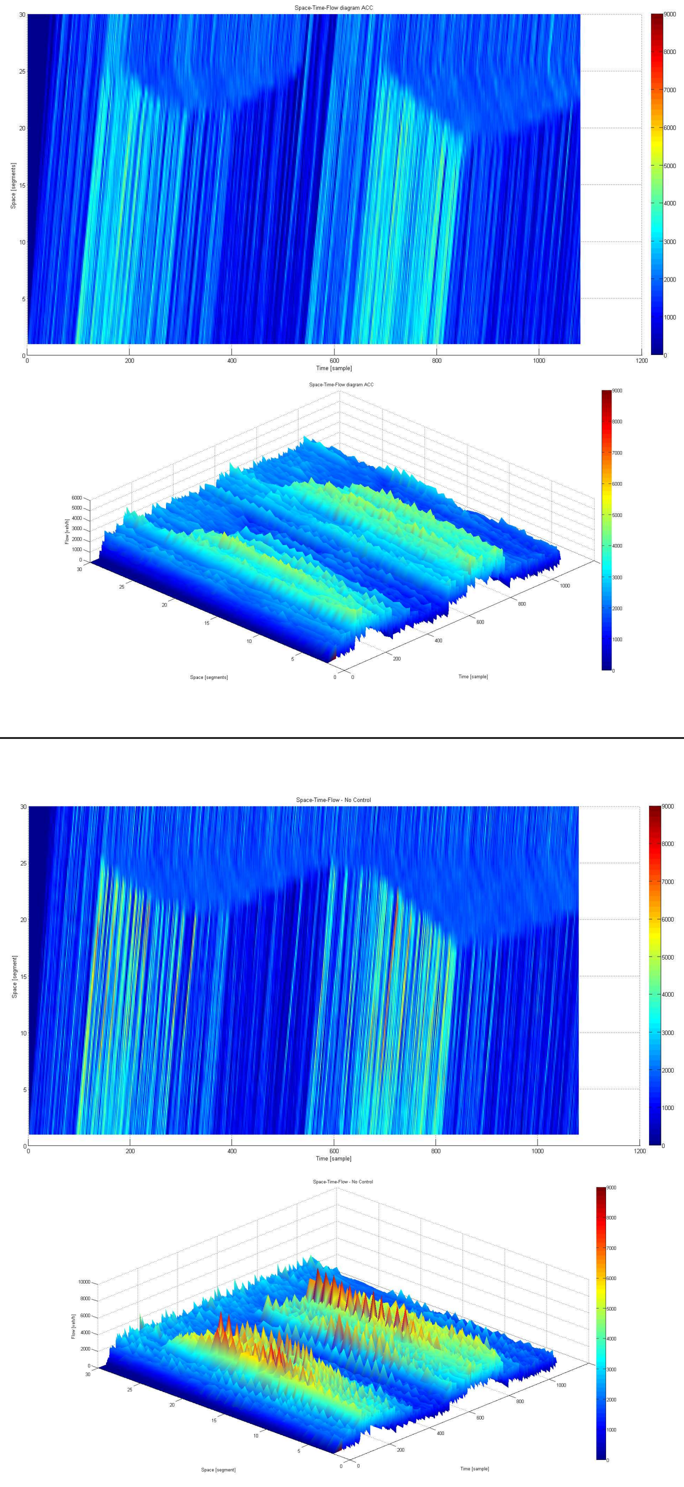


Figure 5.7: The two top graphs are the the space-time-flow graphs for the ACC controller simulation from a 2D- and 3D-view. The two bottom graphs are the space-time-flow graph for the uncontrolled simulation from a 2D- and 3D-view.

6

Further research

The main results from this thesis is a framework which can be used by other researcher when implementing sophisticated VMS, ACC and combined controllers. Even tough the results from the combined controlled simulations in this thesis shows a network improvement compared to the uncontrolled simulations, the controllers are simple and cannot be used to judge how a controlled network performs compared to a uncontrolled network, it merely can indicate how different controllers can affect a network. Even if all the controllers implemented in this thesis requires an improvement to be able to be used when judging the benefits of controlled network, and especially a combined controlled network, the VMS controller are the most simple and ineffective one and are likely the reason that the combined controller are performing worse than the pure ACC controller. The VMS controller had many parameters which could be tuned and adjusted, e.g. the speed steps, the update frequency and the critical density limit, and here probably the discrete speed steps in combination with simple logic were a clear part of the bad VMS result. So, better and more advanced controllers would be interesting to see implemented using this code skeleton and by getting this done be able to see the real improvement of a combined controlled network. But a conclusion that can be drawn from all simulations done during this thesis is that tuning each variable in the controller loop is very important for the outcome of the simulations. Furthermore, in the controller implemented in this thesis, each variable has to be tuned based on the input[3] in order to achieve an improved outcome from the simulations. Implying that not only the network itself is decisive for the settings of the variables in a controller, also tuning the variables based on the input is essential for a good result. Perhaps this problem can be solved, or at least have a smaller impact, if using more advanced control methods. This is something that could be worth to investigating further. Moreover the combined controller, in this thesis, required an infrastructure to vehicle (I2V) communication to send the VMS data to the ACC-equipped vehicles but what if it would be possible for vehicles to communicate with each other, in other words using a vehicle to vehicle (V2V) also? Another interesting topic is to apply an emission model to the simulation. There is a module for this in VISSIM, but due to the time frame for this thesis it was not possible to implement it. Nonetheless, the emissions decrease would be interesting to investigate and it is a good way to make it easier to grasp the improvement of the network. Another interesting investigation would be to find out how much energy an electric car could conserve by using a combined control system. This in turn can for example be translated to how much further an electric car can go on one charge. Today much time and money are spent on reducing drag coefficient, improving battery technologies, building vehicles in light-weight material, etcetera to make electrical vehicles go longer distances. Maybe a combined controller is a relatively cheap way to extend the distance an electric car can travel.

[3] Input is the amount of vehicles requested to enter the road during a time period or a combination of different time-periods with their respective requests of vehicles to enter. Even though input can include much more e.g. different percentage of vehicle types requested to enter the road, this is nothing that has been experiment with during this thesis. The input for the simulation done in this thesis are explained in this section 4.1.

7

Miscellaneous

When using the software VISSIM there are some things that are good to know during the building of networks, working with simulation codes and working with driver model codes. During this master thesis there were some problems which caused much distress but ones discovered they were easy to fix. In this chapter a few remarks will be given about each module that was used during this master thesis and I hope that it will help others to not get stuck.

7.1 GUI (main programme)

When building the network on which the simulation should occur in, there were two things which caused problems for me. I want to make clear that the solution to problems I've encountered during this master's thesis definitely **not** have to be the best or even good solutions. Same goes for the tricks I've learned. It probably exist other ways to manage what I've done. Simply said is that this chapter gives ideas and suggestions on how to solve a problem or working with the software easier.

Link length
I found it difficult to, by drag and drop with the mouse, give a link a precise length. A solution to this problem is to drag a long link and use the "Split Link" function to do give the exact length of links. By splitting also a connector between the two links will appear. To split a link, simply select it and press F8 (the fast command for this operation) and precise the length of the two new links which will be the result of the split (along with a connector between them). Then you can continue by splitting one of the new links and doing this until all the links (segments) you want are done.

Connectors
Between two links there is an connector. This connector, as the name suggests,

connects links. This connector is often very short and are easy to forget and they can cause a problem which is not easy to find unless someone tells you it. The problem occurs when putting, as an example, a traffic light on the connector part. Because if you do that the traffic light won't work as you want. So if you like to put a traffic light at the end of a link of 500 meters, do **not** put the traffic light on position 500 meter on the selected link because that will mean you putting it on a connector which makes the traffic light not changing the speed of vehicles passing it even though it should. You can change the length of the connector, but often it's enough to put traffic light, and alike, some meter from the end of the link. Meaning, in the example above, if you put the traffic light on position 495 meter it will behave correct. Remember, as mentioned before, you can always read out the length of a connector and adjust the of it length to adapt it so it fits you. A connector is selected and editable the same way as link selected and editable.

7.2 COM module

Predefined speeds
It's not possible to try to set a speed of a vehicle unless it's already defined in the GUI. If trying to set a speed which is not defined in the GUI from inside the COM-module the simulation will crash. A solution to this problem is, since it's infeasible to predefine all speeds, to let a function inside the COM-module find the most close speed which is defined in the GUI and set this speed instead. I guess a reason for this problem is that a defined speed you try to set is not a speed, it's more a speed number which has an speed distribution assigned to it. As an example, if you set the speed number 100 to a vehicles it means that the vehicles can be driving between maybe 80 - 130 km/h. The distribution interval and the distribution curve can be be changed by a user. So, it's possible to create a new speed number, let's say 99 and assign it a distribution between the speed 99 km/h to 100 km/h.

Log file
This maybe isn't a chock to anybody but using a log file is an easy way to find out where problems occurs. It's also possible to use the command prompt window to print out when different instances of the code has been reached so it's easy to see that the systems seems to do what you intend it to do.

_variant_t
This is a data type which are used by VISSM. So if you try to fetch data from a function and later on store it to, for example, a log file it will not work. So, don't forget to parse the data to a format you can use. This is also true the other way around, when you want to set data using a function. More information of what kind of data different functions requires is to be found in the COM Interface Manual [18].

GetVehicleByNumber
The function GetVehicleByNumber [18, p.88] seems nice and dandy and it probably is in many cases but in my simulation vehicles constantly left the network and every vehicle has a unique number. So if you want to access a vehicle to be able to change it's parameters you need to keep track of all vehicles on the road which has left and which has entered and this has to be done at every iteration. And if you try to access a vehicle which already left the network, the simulation will crash. So instead I'm used the following method to access all the vehicles; **spVeh = spVehicles→GetItem(i)**. It's the GetItem(i) that simply replacing the GetVehicleByNumber and it's a great function for iterate through all vehicles on the road. I believe that this function is described in the manual but as "**Item ([in] VARIANT Index, [out, retval] IVehicle **ppVehicle)**" [18, p.87].

7.3 Driver Model

Driver Model module does not support multi-core
For some reason the Driver Model module does **not** support multi-core simulations. This is not mentioned in the VISSIM manual (as far as I know) and it's a scary problem because it's possible to run the simulation even-though multi-core are used in a simulation using own made driver models. The way I detected this was when running more than one simulation with the exact same parameters and network setup. When I did this the result was different from each-other even-though the simulator is deterministic it still took a while before I asked the PTV Vision about the apparently stochastic results since I've tried to run more simulations without ACC, with ACC and no VMS, only with VMS and no control at all to try to understand why it didn't work as I thought it would. The gist of this is that sometimes the only way to solve a problem is to ask the people developing the software and ask before too much time are spent unnecessarily. As a student, using a free license, you're not eligible support from PTV Vision but during my thesis they were still very supportive. I was never denied help or support. Anyhow, to turn multi-core simulation on and off go to *Simulation Parameters* and under *Number of cores* you can choose the number of cores you want to use. By choosing *1* here the multi-core is off.

Speed is in m/s
In the driver model speed is in m/s, so if you thinking in km/h which, for me at least, is the most common to do when working with vehicle speeds, don't forget to convert your thoughts to m/s when writing your code.

Cover all cases
A reminder. When creating a driver model make sure to cover all possible cases. There are many possibilities and it's not good if there is a case which is not covered.

Simplify using VISSIM driver model in low speeds

At low speeds it's harder to make a driver model not making vehicles drive through each-other or behave on a way creating a bad driving situation. A way to simplify this problem is to use VISSIMs driver model, or more correctly said, not change the value VISSIM recommends. I did my speed regulation by changing the acceleration of the ACC vehicles but at speeds below 10 m/s I didn't change the acceleration that VISSIM want to use. This limit 10 m/s can of course be changed to something else, it's up to the programmer.

Bibliography

[1] P. Clarke, Adaptive cruise control takes to the highway, Web page (October 1998). URL http://www.eetimes.com/electronics-news/4036396/Adaptive-cruise-control-takes-to-the-highway

[2] D. Corona, B. De Schutter, Adaptive cruise control for a smart car: A comparison benchmark for mpc-pwa control methods, Control Systems Technology, IEEE Transactions on 16 (2) (2008) 365 –372.

[3] L. N. Boyle, F. Mannering, Impact of traveler advisory systems on driving speed: some new evidence, Transportation Research Part C: Emerging Technologies 12 (1) (2004) 57 – 72.
URL http://www.sciencedirect.com/science/article/pii/S0968090X03000779

[4] A. Hegyi, B. D. Schutter, J. Hellendoorn, Optimal coordination of variable speed limits to suppress shock waves, Intelligent Transportation Systems, IEEE Transactions on 6 (1) (2005) 102–112.

[5] S. Mammar, A. Messmer, P. Jensen, M. Papageorgiou, H. Haj-Salem, L. Jensen, Automatic control of variable message signs in aalborg, Transportation Research Part C: Emerging Technologies 4 (3) (1996) 131 – 150.
URL http://www.sciencedirect.com/science/article/pii/S0968090X96000058

[6] H. Kulovits, C. Stogerer, W. Kastner, System architecture for variable message signs, in: Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on, Vol. 2, 2005, pp. 7 pp. –909.

[7] S. Smulders, Control of freeway traffic flow by variable speed signs, Transportation Research Part B: Methodological 24 (2) (1990) 111 – 132.
URL http://www.sciencedirect.com/science/article/pii/019126159090023R

[8] V. Milanes, J. Villagra, J. Godoy, J. Simo, J. Perez, E. Onieva, An intelligent v2i-based traffic management system, Intelligent Transportation Systems, IEEE Transactions on 13 (1) (2012) 49 –58.

[9] G. Orosz, R. E. Wilson, G. Stépán, Traffic jams: dynamics and control, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 368 (1928) (2010) 4455–4479.
URL `http://rsta.royalsocietypublishing.org/content/368/1928/4455.abstract`

[10] P. Vision, http://www.ptv-vision.com/en-uk/, Home page, this is english version of the PTV Vision homepage (October 2012).

[11] Microsoft, Home page (October 2012).
URL `http://www.microsoft.com/en-us/default.aspx`

[12] M. T. L. of Technical Computing, http://www.mathworks.se/products/matlab/, Home page (October 2012).

[13] T. Luspay, Advanced freeway traffic modeling and control, Ph.D. thesis, Budapest University of Technology and Economics (October 2011).

[14] Encyclopedia of mathematics, Home page, newton-Leibniz formula (December 2012).
URL `http://www.encyclopediaofmath.org/index.php?title=Newton-Leibniz_formula&oldid=28964`

[15] D. C. Gazis, R. Herman, R. W. Rothery, Nonlinear follow-the-leader models of traffic flow, Operations Research 9 (4) (July/August 1961) 545–567.
URL `http://or.journal.informs.org/content/9/4/545.abstract`

[16] M. Brackstone, M. McDonald, Car-following: a historical review, Transportation Research Part F: Traffic Psychology and Behaviour 2 (4) (1999) 181 – 196.
URL `http://www.sciencedirect.com/science/article/pii/S136984780000005X`

[17] Planung Transport Verkehr AG (PTV), Stumpfstraße, 1 D-76131 Karlsruhe, Germany, PTV Vision VISSIM User Manual, 5th Edition (July 2009).

[18] Planung Transport Verkehr AG (PTV), Stumpfstraße, 1 D-76131 Karlsruhe, Germany, PTV Vision COM Interface Manual, 5th Edition (November 2011).