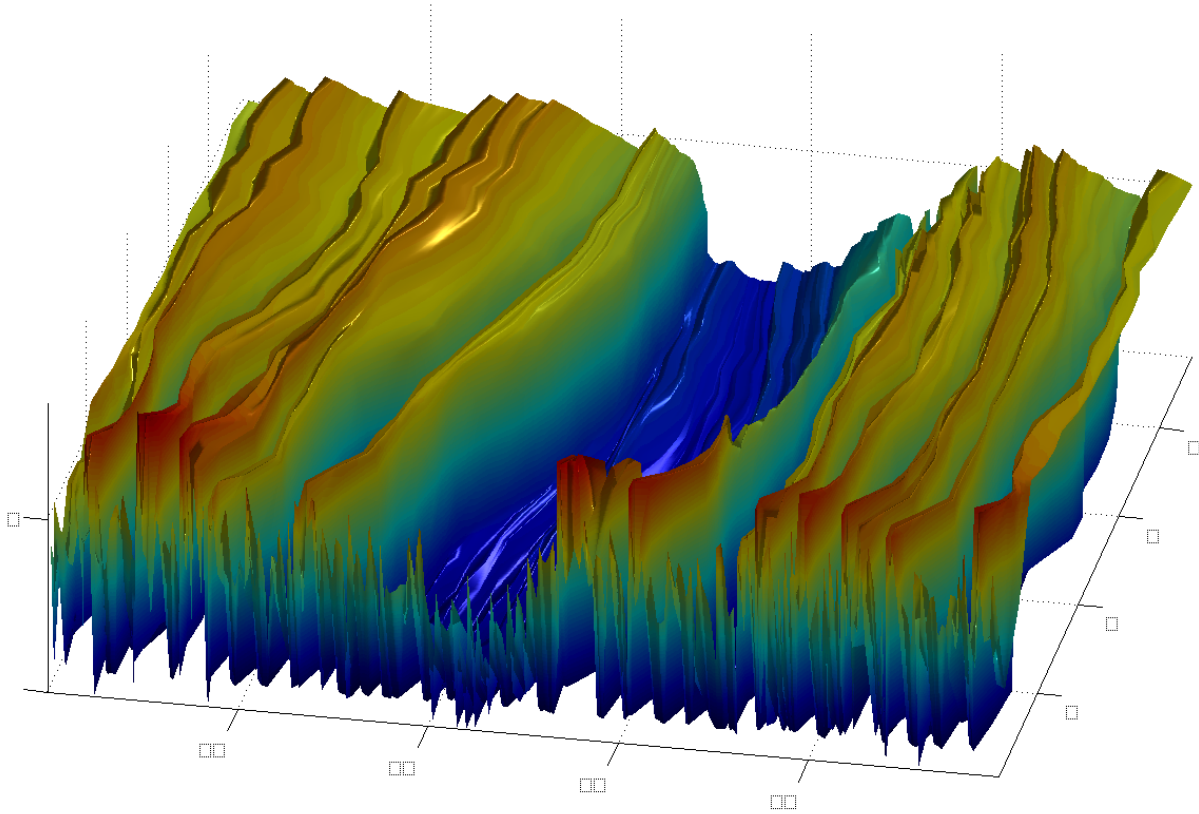
University of Twente
Enschede - The Netherlands

Providing Over-the-horizon Awareness to Driver Support Systems by means of multi-hop ad hoc Vehicle-to-Vehicle Communication



Master's Thesis

E.M. van Eenennaam

Chair for Design and Analysis of Communication Systems (DACS)
Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)
University of Twente, The Netherlands

Supervisors:
Dr. ir. Geert Heijenk
Dr. ir. Georgios Karagiannis
Prof. dr. ir. Bart van Arem

December 9, 2008

Abstract

The application of Driver Support Systems (DSS) is an emerging trend in the automotive industry. Advances in technology enable faster, smaller and more versatile hard- and software systems while consumers demand safer and more efficient cars. A DSS aims to meet the consumer's demand with technologically feasible solutions. The Ph.D. Thesis "Driver Support in Congestion" by C.J.G. van Driel proposes a suite of Driver Support Systems called the *Congestion Assistant*. The Congestion Assistant functions by means of over-the-horizon awareness to aid the driver in traversing traffic congestion on highways. Subsystems are proposed to aid the driver and supply the driver with information. The Congestion Assistant has been tested at a functional level, but an information dissemination system was not presented in the study. In the design information was assumed to be instantaneously and reliably available.

A DSS, which relies on over-the-horizon awareness in a mobile environment, poses some interesting demands on the communication system designed to distribute this information. For instance, latency must be minimal in order to guarantee freshness of information, even if it has travelled several kilometers along the road.

This thesis covers the design of a communication system aimed to provide the over-the-horizon awareness to the Congestion Assistant. Several options are considered and an approach based on multi-hop Vehicle-to-Vehicle communication is proposed. The over-the-horizon awareness can be represented in a structure called a TrafficMap. This is a speed profile which expresses the traffic flow speed at a certain location on the road in a highly compressed form. The TrafficMap is built by means of a distributed system called the TrafficFilter, which is proposed in this thesis. This system is present in all intelligent vehicles on the road; together they build the over-the-horizon view.

A protocol is designed to disseminate the required information in an efficient manner. A directional flooding approach is reasoned to be the most appropriate way to distribute the TrafficMap to all relevant vehicles on the road. A modification to the slotted 1-Persistence Flooding strategy is proposed and evaluated. This modification enables the use of an IEEE 802.11p MAC and physical layer and perform network-layer flooding in an efficient manner by exploiting MAC layer scheduling properties.

Simulation studies are performed to evaluate the performance of the system. The influence of mobility on the communication and the quality of the communicated information with respect to error rate and latency is evaluated. The TrafficFilter is found to be a viable system to build over-the-horizon awareness for future DSS like the Congestion Assistant.

Samenvatting

De toepassing van bestuurderondersteunende systemen (Driver Support Systems, DSS) is een recente trend in de automobielindustrie. Door voortschrijdende ontwikkelingen zijn snellere, kleinere en veelzijdiger hard- en softwaresystemen mogelijk terwijl de consument eist dat voertuigen veiliger en efficiënter worden. Een DSS poogt de eisen van de consument te koppelen aan oplossingen die vandaag de dag mogelijk zijn. In het proefschrift “Driver Support Systems in Congestion” door C.J.G. van Driel wordt een systeem voor bestuurder-ondersteuning voorgesteld dat de naam Congestion Assistant (“file assistent”) draagt. Dit systeem is gebaseerd op kennis van de situatie op de weg waarop een voertuig rijdt en ondersteunt de bestuurder bij het rijden in files op snelwegen. Subsystemen zijn ontworpen welke de bestuurder helpen en van informatie voorzien. De Congestion Assistant is op functioneel niveau getest, maar een systeem om de informatie te verspreiden is niet voorgesteld. In het ontwerp was aangenomen dat deze informatie instantaan en betrouwbaar beschikbaar is.

Een DSS, dat afhankelijk is van een zogenaamd over-de-horizon bewustzijn in een mobiele omgeving, stelt interessante eisen aan het communicatie systeem dat ontworpen is om deze informatie te distribueren. Bijvoorbeeld, de eind-tot-eind-vertraging moet minimaal zijn om te kunnen garanderen dat de informatie de situatie op de weg nog betrouwbaar weerspiegelt, zelfs als deze informatie al verscheidene kilometers doorgegeven is.

Dit afstudeerverslag beschrijft het ontwerp van een communicatiesysteem met als doel het leveren van over-de-horizon bewustzijn aan de Congestion Assistant. Verscheidene opties zijn overwogen en een benadering gebaseerd op multi-hop voertuig-voertuig-communicatie is voorgesteld. Het over-de-horizon-bewustzijn kan worden gerepresenteerd in een datastructuur welke de TrafficMap genoemd wordt. Dit is een snelheidsprofiel dat de verkeersdoorstroom op een bepaald punt op de weg uitdrukt in een uiterst beknopte vorm. De TrafficMap komt voort uit een gedistribueerd systeem dat het TrafficFilter wordt genoemd. Dit systeem bevint zich in alle intelligente voertuigen op de weg; samen bouwen ze het over-de-horizon-bewustzijn.

Een protocol is ontworpen om de benodigde informatie op efficiënte wijze te verspreiden. Een directionele flood benadering is geacht de meest toepasselijke manier te zijn om de TrafficMap te verspreiden over alle voertuigen op de weg. Een aanpassing op de slotted 1-Persistence Flooding strategie is voorgesteld en onderzocht. Deze modificatie maakt het mogelijk een IEEE 802.1p MAC en fysieke laag te gebruiken en voert flooding uit op de netwerklaag. Dit kan efficiënt door het uitbuiten van kennis van de onderliggende MAC-laag.

Simulatieexperimenten zijn uitgevoerd om het gedrag van het ontworpen system inzichtelijk te maken. De invloed van mobiliteit op de communicatie en de kwaliteit van de gecommuniceerde informatie met betrekking tot foutmarge en vertraging is onderzocht. Het TrafficFilter systeem lijkt een goede kandidaat voor het bouwen van over-de-horizon bewustzijn in toekomstige systemen voor bestuurderondersteuning zoals de Congestion Assistant.

Preface

The application areas of Telematics are very diverse. Over the past few years, I have grown an interest in the direction of mobile wireless networking. My Bachelor research was centered around a Nomadic Service Provider [110]. Later a couple of optional Master courses reinforced this interest. When looking for a subject for my Master research, the choice was quite easy—the challenges of the dynamic nature of mobile networks lured again.

Nowadays, it seems almost everybody in the Netherlands has a car. The right to own and drive a vehicle is one of our constitutional rights, or so it appears. But as with many systems, there is a bounded capacity. The challenges of using our road system efficiently and effectively has been on the political agenda for a long time. It is an interesting mix of demands made by drivers, rules set by the government, solutions provided by the industry and theories and pleas from environmentalist organisations and those living right next to areas where the adverse effects manifest.

This thesis presents the literature study, the design of a multi-hop Vehicle-to-Vehicle communication system which provides over-the-horizon awareness to vehicles, and the results of my Master's project. The first part is primarily concerned with the context and problem definition. A solution to the problem has been provided on a conceptual level and the remainder of this document is dedicated to further development and refinement of this solution.

This research relied heavily on the open source OMNeT++ simulator. Because it is open source, it is very flexible: if a required function is not available, you can simply implement it. For sure, this research would not have been possible without this flexibility.

Like John Donne wrote, "No man is an island" [25] and ironically this becomes even more apparent when striving for a great personal accomplishment. I would like to thank the following people for their help during this project:

- My supervisors: Geert Heijenk, Georgios Karagiannis and Bart van Arem
- Mark van der Vusse, Jacco den Hollander, Rijkswaterstaat - Verkeerscentrum Nederland (VCNL), Arie Penning (DVS) for the Location Database (VILD 4.3a)
- Dorette Alink-Olthof (CTW) for the reader Traffic Flow Fundamentals
- The OMNeT++ community, especially the mailinglist.
- Andras Varga for his work and support on the OMNeT++ simulator
- Daniel Willkomm, Marc Löbbers, Andreas Köpke, and other developers of the Mobility Framework
- Sander Veenstra for a discussion on highway, freeway, motorway and expressway terminology

- Jochem Rutgers for a very meticulous proofread of the thesis
- Colleagues at the DACS lab (Daan, Gert, Rámon, Stephan, Wouter) for the many inspiring conversations, some of which were on-topic and some actually provided useful input
- And last—but certainly not least—my friends and family for their continued support

Although entertainment is not the primary objective of this work, I do hope it is a pleasant read.

Martijn van Eenennaam
Enschede, December 2008

Wer später bremst, fährt länger schnell!

–German proverb

Now, if we had this sort of thing: `yield -a` *for yield to all traffic,* `yield -t` *for yield to trucks,* `yield -f` *for yield to people walking (yield foot),* `yield -d t*` *for yield on days starting with t ...you'd have a lot of dead people at intersections, and traffic jams you wouldn't believe...*

–Discussion in comp.os.linux.misc on the intuitiveness of commands

If all the cars in the United States were placed end to end, it would probably be Labor Day Weekend.

–Doug Larson

Each year it seems to take less time to fly across the ocean and longer to drive to work.

–Author unknown

Another way to solve the traffic problems of this country is to pass a law that only paid-for cars be allowed to use the highways.

–Will Rogers

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the advent of the horse carriage and later the motor carriage, nowadays simply known as *car*, people experienced a huge increase in mobility. It became possible to travel great distances whilst being comfortably seated and sheltered against the elements. Since the industrial revolution standards of living have increased. Industrialised countries have witnessed a great expanse on the use of their road systems, as cars transitioned from luxury products for the rich, to goods available to the average person. An ever-increasing number of people made use of the road networks, as a result, numerous crossings and roads became bottlenecks. This results in a form of queuing also referred to as *gridlock*, *traffic jam*, or *congestion*.

As early as 1926, busy road traffic has resulted in a great demise in commuter efficiency, ultimately leading to events where "*Millions Spend (the) Whole Day Going to Work and Then Returning*" [40]. Since then, congestions have grown into daily returning events when people go to work and return home from their work. In 2006, this resulted in €630 million of *economical damage* in The Netherlands alone [30]. Furthermore, traffic jams also result in heavy pollution and inconveniences for people neighbouring busy roads. In order to reach the goals set by the Kyoto Protocol in 1997 [61], it is recognised that reducing the amount of traffic jams is of great importance. This is because in the European Union some 25% of the total CO_2 emissions can be contributed to the transport sector [84]. Traffic congestion is also found to increase aggression in drivers [62]. Many dangerous traffic situations, potentially with deadly results, can be attributed—directly or indirectly—to traffic congestion.

Governments have been adding more roads to the network and at particular bottleneck locations extra lanes have been added—such as *carpool* lanes available only to vehicles with many occupants—and toll roads and taxes on fuel function as incentives for people to use their vehicle less. In order to clean the air in Beijing, one of the world's most polluted cities, the Chinese government passed a law that restricted about half of Beijing's 3.3 million cars from using the roads in the period around the Olympic Games [99].

Besides expanding the capacity—building more roads—and decreasing the demand—getting more people to use a bicycle or public transit—there exists a third alternative. This alternative is to make more efficient use of the system presently in place. It is identified that humans exhibit some traits making them less capable of controlling vehicles when compared to automated systems. A relatively long reaction time with respect to the speeds at which the vehicles travel, fatigue, distraction and (faulty) opinions are only a few factors. Intelligent Transportation Systems (ITS) or Road Transport and Traffic Telematics (RTTT) are technologies expected to make traffic safer and more efficient over the next few decades by supplementing the human driver's shortcomings [51, 109, 12, 108].

1.1 Background

This research is based on findings presented in *Driver Support In Congestion - An assessment of user needs and impacts on driver and traffic flow* [109] by C.J.G. van Driel. In this Ph.D. thesis, the problem of traffic congestion is approached from the user point of view. By means of a questionnaire, drivers were asked with which tasks they would like to be assisted by (a system in) their vehicles. It was found that drivers particularly were in favour of assistance with the driving task in congestions. Based on these findings, a system was designed to support drivers in coping with traffic jams on highways. A Driving Simulation study carried out at TNO (Netherlands Organisation for Applied Scientific Research)[1] showed that drivers were eager to accept such a system. In a Microscopic Traffic Simulation study, it was shown that—with a large enough degree of market penetration—a reduction of the number and effects of traffic jams is possible.

The system designed by van Driel in her dissertation is called the *Congestion Assistant*. The Congestion Assistant is a system onboard a vehicle which is based on knowledge of the situation on the road ahead. It is assumed that this knowledge is available and dependable, but no system for distributing this knowledge is proposed. This research sets out to devise such a system and explore possibilities.

1.2 Research Objectives and Scope

In this research, we have designed a system that provides over-the-horizon awareness to the Congestion Assistant. This system will communicate information concerning over-the-horizon situations required by the Congestion Assistant. The communication system required by the Congestion Assistant—and ITS systems in general—poses several challenges when compared to stationary automation systems:

- The communication network (i.e. the vehicles) has very high mobility. Ordinary Wireless LAN technologies are designed for low mobility. As a result, they cannot be used in a vehicular environment or modifications are required.
- Because of the high mobility and potentially short moments of contact, there is no time to construct multicast trees or maintain other state-dependent structures or perform handshakes and RTS/CTS sequences.
- A potentially large amount of information needs to be collected and distributed. Due to the dynamic nature of the communicating nodes, this information is constantly changing.
- Nodes in the network are geographically dispersed and their number can be large. It is hard, if not impossible, to obtain a complete and accurate overview (e.g. as in a centralised or fixed system).

The focus of this research lies with providing basic functionality to the Congestion Assistant. It may very well be possible to aggregate other interesting information from the data distributed and collected, this writing will not go into much detail but will briefly point out opportunities. On the same note we will also refrain from expanding the Congestion Assistant. It should be noted that integration of this functionality with other systems and retrieval of only a little more information from the surrounding vehicles might clear the way for future fully integrated systems.

As such the objective of this research is three-fold:

[1] http://www.tno.nl

Objective 1 – Gain more insight in the dynamic world of traffic and Intelligent Transportation Systems and the Congestion Assistant in general

Objective 2 – Based on this insight, propose a communication service provider for the Congestion Assistant

Objective 3 – Research the feasibility of this solution

1.3 Research Approach

This research uses three methodologies: literature study, system design and simulation. Study of the Congestion Assistant is required in order to get a notion of the information needs and the communication requirements. Study of the state of the art serves two purposes: set the context and provide partial solutions and applicable techniques. The design part is concerned with defining a system based on best-practices observed in literature to meet the Congestion Assistant's information needs. The conceptual design was presented at the V2VCOM2008 work-in-progress workshop [111]. Finally, simulation studies are performed to evaluate the viability of the proposed solution.

This research addresses the following questions:

1.3.1 What are the Information Requirements of the Congestion Assistant?

What information is needed by the Congestion Assistant, how can we classify this information? From what sources does or can this information originate?

1.3.2 How can the Congestion Assistant's information needs best be fullfilled?

What is the best way to meet the Congestion Assistant's information needs, and how can such a system best be designed?

1.3.3 What is the performance of this method and what are the trade-offs?

What limitations can be identified in the approach chosen? What are the implications for the Congestion Assistant?

1.3.4 Is it possible to meet the Congestion Assistant's information needs?

Can the Congestion Assistant operate properly with the information delivered by the proposed system?

1.4 Outline

Figure 1.1 shows the outline of this thesis. As indicated, the research starts with a thorough study of the Congestion Assistant as described in [109]. The goal is to derive the communication requirements of the Congestion Assistant. An overview of this study is provided in Chapter 2.

Of particular interest is what information is needed and how detailed this information must be for the Congestion Assistant to operate properly.

The outcome of the literature study presents the current state of the art. Focus is on what advances have been made, which technologies have been engineered, and how they can be used in the design of a vehicle-to-vehicle communication service provider for the Congestion Assistant. The results are given in Chapter 3.

A method to build the required over-the-horizon view is provided in Chapter 4, where a solution is presented on a conceptual level: the TrafficFilter system. The TrafficFilter system described in this chapter has been submitted and presented at V2VCOM2008, the paper titled *Providing Over-the-horizon Awareness to Driver Support Systems* [111] is provided in Appendix E.

Chapter 5 delves deeper into the technical aspects of distributing the required information among the possibly thousands of vehicles on the road. A complete system design will be proposed.

Next, the system is tested and evaluated. This is done by means of a simulation study. Chapter 6 reports on the simulation results. Several alternative designs are evaluated. Focus is on the flooding mechanism used, the influence of mobility on message dissemination and the quality of the information which is transferred.

Chapter 7 provides a condensed retrospect of the results together with conclusions, recommendations and future work.

In Appendix A, an evaluation of means to denote positions of vehicles upstream of the vehicle in casu to provide an over-the-horizon view is included. An overview of the simulation environment and the implementation of the system into the OMNeT++ discrete event simulator is covered in Appendix B.

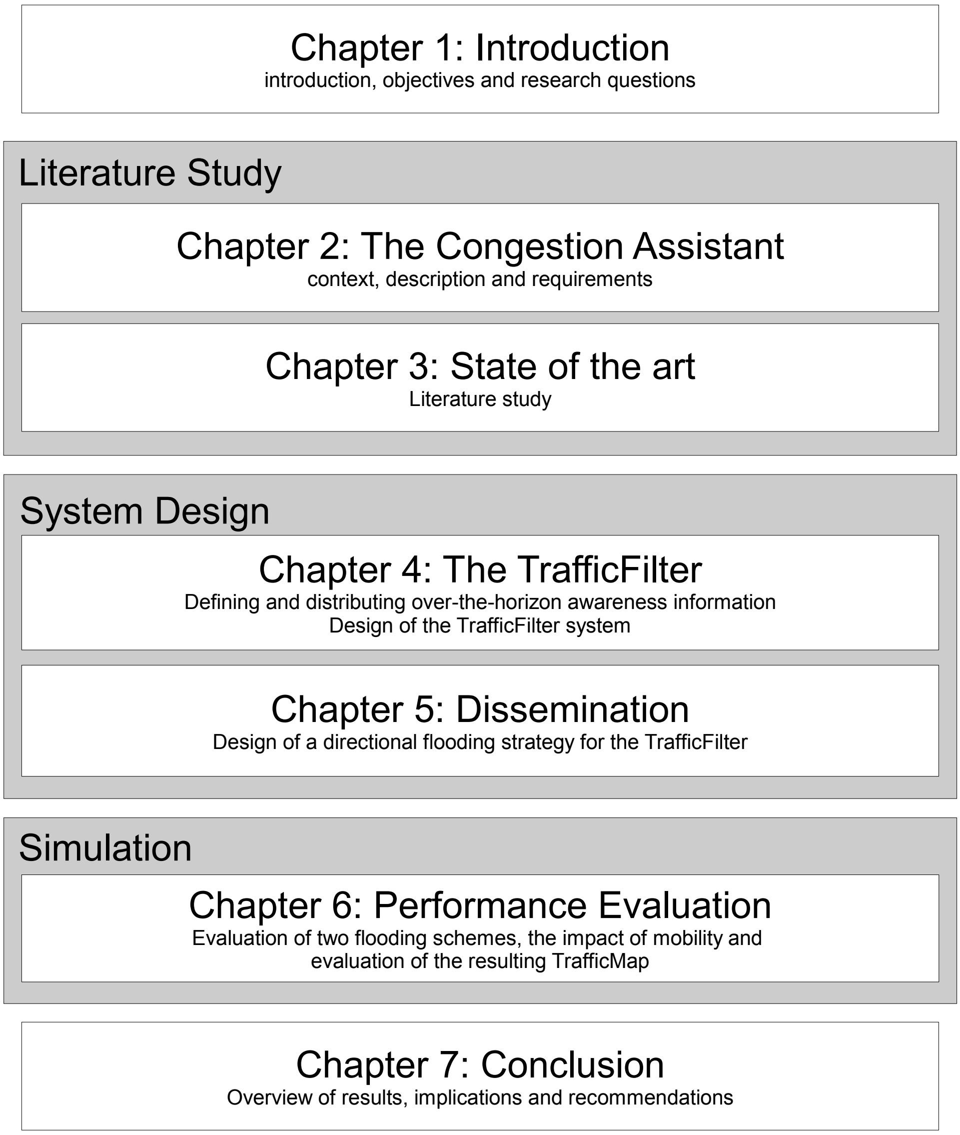


Figure 1.1: Outline of this thesis

Chapter 2

The Congestion Assistant

The Congestion Assistant is the result of research performed by C.J.G. van Driel in three phases: a user needs analysis, driver simulation studies and traffic analysis studies. We will briefly recap the research, the design of the Congestion Assistant and the relevant results presented in [109].

User Needs Analysis

The research was carried out using an internet questionnaire. Focus was on the user's perception of what intelligent vehicles could do to assist the driver in the driving task. One of the conclusions is that intelligent vehicles are regarded by the majority of the participants as favourable, supplying helpfull assistance during tiresomely repetitive driving tasks such as stop-and-go traffic. Furthermore, users indicated the system should assist during potentially dangerous situations. It was found that drivers like to be well-informed about upcoming traffic conditions.

Based on the outcome of the survey the *Congestion Assistant* was designed. This is a system that helps drivers cope with traffic congestion.

Driver Simulation Study

Tests were conducted in TNO's driving simulator. Participants were asked to drive a stretch of highway under different visibility conditions. Several factors were monitored, such as the Time To Collision (TTC), following distance, speed and the mental workload on the user. The Congestion Assistant showed promising improvements in safety and efficiency. It is able to mitigate some of the unfavourable human behaviour that is part of the cause of traffic congestions. Especially the Stop & Go feature was highly appreciated by the participants and provided good gains in efficiency.

Traffic Analysis Study

The Congestion Assistant is designed to improve a driver's efficiency and performance in traversing a traffic jam. A microscopic traffic simulation was performed to evaluate the effects of improvements to individual driver behaviour on the overall traffic performance. It is concluded that traffic as a whole benefits from the improvement of only a small number of drivers (10% equipment rate) and further improves when the penetration goes up to 50%. The results are:

- less congestion
- higher queue discharge flows (more cars leave the jam per time unit)

- reduced congestion inflow (the inflow is spread out over time, cars gradually assume closer following distances; as a result the road accommodates more vehicles per distance unit)
- efficient car following behaviour in congested stop-and-go traffic

The general result was a more stable and homogenous traffic flow with a smaller standard deviation in speed. Small standard deviations in speed are favourable because drivers sometimes tend to overreact or react too late, resulting in shockwaves of braking vehicles or—in the worst case—head-tail collisions. It will be clear that, with smaller deviations in speed, there is more time to react and less compensation is required. The close following distances enabled by the automated system did have one shortcoming in the Stop & Go phase in conjunction with lane changes. The smaller gaps between vehicles resulted in more hard braking. It is argued in [109] that this can be mitigated with lane-change support measures.

2.1 System Architecture

The Congestion Assistant is a system designed to aid drivers in traversing traffic congestions on highways. It performs three tasks; it informs, supports and controls. These tasks are present in the following three functions: Warning & Information (W&I), Active Pedal (AP) and Stop & Go (S&G). These three functions are executed consecutively based on the distance to the jam as presented in Table 2.1 and Figure 2.1.

Distance	**Event**
5km	W&I: First congestion warning
1.5km	Active Pedal on
0km - tail of jam	Stop & Go on, Active Pedal off
head of jam	Stop & Go off, driver resumes normal driving

Table 2.1: Events based on distance to the traffic jam

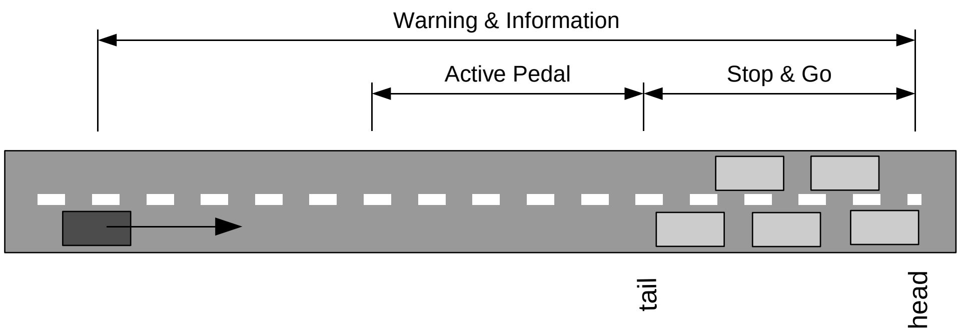


Figure 2.1: A driver approaches a traffic jam on a two-lane highway

Each of these three tasks requires a certain amount of knowledge of the traffic conditions downstream. The remainder of this chapter covers the information required for each of these three systems to operate.

2.1.1 Warning & Information

The W&I informs the driver of upcoming traffic conditions. If there is a traffic congestion ahead the driver will be informed. This enables the driver to prepare for driving in the jam, or choose an alternate route. Furthermore, once in the jam W&I will keep the driver updated on the situation and the progress. The exact nature of this warning (e.g. the exact information displayed) is not known. Although not explicitly stated in [109] we assume the following information to be required for basic W&I operation:

- Own position, speed (to be obtained from internal navigation system, (d)GPS)
- The position of the tail of the jam
- Position of the head of the jam
- Average speed of the jam, movement within the jam

Based on the position of the tail and head of the jam we can calculate the total length of the jam. Using the average speed we can calculate expected incurred delay. Using the current position of the vehicle we can calculate the distance to go until the tail of the jam is reached, the progress within the jam, and the time remaining before normal traffic flow recommences.

It seems sensible to expect that, besides helping the user anticipate the congestion, a congestion warning might also help a user in deciding to choose a different route to his destination. In order to accomodate this, information should be available well in advance. We assume the W&I to stay on and up-to-date during the period the vehicle is traversing the congestion; this will satisfy a user's desire to be well-informed.

In [109] there is no mention of a maximum acceptable delay or spatial-temporal accuracy of the information. This research will provide some insight into what is achievable. Further research will then have to determine if this is adequate for the Congestion Assistant to function properly in practice.

2.1.2 Active Pedal

The AP gives counterpressure starting from a certain distance prior to entering the tail of the congestion. In the experiments this distance was set to 1500m or 500m respectively. The goal is to prevent the unfavourable behaviour of maintaining cruise speed until the tail of the jam is in sight, because then suddenly very hard braking is required. This can result in accidents, but is also found to result in a high inflow of traffic, which causes the jam to grow at the tail. The AP gives counterpressure on the accelerator pedal which results in the driver gradually reducing the vehicle's speed. In microscopic simulations this behaviour is found to reduce the inflow of traffic in the congestion because at a lower speed a closer following distance can be maintained, resulting in more vehicles per distance unit of road. The AP needs the following information:

- Position and speed of the vehicle (onboard (d)GPS)
- Position of the tail of the jam
- Speed of the tail of the jam

Based on this data the system can calculate at which point the AP needs to be engaged (e.g. 1500m or 500m before the tail) and when it should be disengaged. Furthermore, when the system decelerates it must ultimately match the speed of the last car in the jam.

In the Ph.D. thesis [109] it is also shortly discussed that some drivers did not react as intended, and put more pressure on the pedal themselves as well to counter the pressure. It is considered that, perhaps, an active braking pedal would be better. For this research it is assumed that some kind of speed inhibitor device is present in the vehicle which is able to adapt the vehicle's cruise speed to that of the tail of the traffic jam. This will occur gradually over a certain distance, either with or without the driver's consent. Our research is oblivious of the exact implementation of such a behaviour modification device or any legal or commercial issues concerning its deployment.

2.1.3 Stop & Go

As soon as the vehicle enters the congestion the Stop & Go (S&G) subsystem will be enabled. In the Congestion Assistant S&G is defined to function at speeds up to 50km/h, but this is a calibration issue. It has also been suggested to use an Integrated full-Range Speed Assistant (IRSA) [118] which performs ACC (Adaptive Cruise Control) tasks. As soon as the vehicle leaves the congestion the S&G system disengages and manual driving will recommence. The system will need to have the following data available:

- Position of the begin of the congestion (tail), when to engage

- Position of the end of the congestion (head), when to disengage
- Speed to assume / maintain
- Current headway (distance to vehicle in front)

In the Congestion Assistant headways of 1 second and 0.8 second are researched. Other options would be to dynamically calculate the headway based on the current speed and vehicle specifications (maximum braking power), current state of the vehicle (type of tires, wear on brakepads and discs etc.) and road conditions (wet, slippery, dry). This is out of the scope of this research, we just assume a Stop & Go facility is present.

It is not clear if the Congestion Assistant requires the information to be specified per lane, or the combined average of all lanes. For simplicity, the highway could be seen as a pipe through which traffic flows. This approach is taken in this research and is applicable to a stretch of highway between two junctions. It is left as future work to expand the proposed system to a more complex road network, in which case it might be necessary to record traffic flow speed on each specific lane .

2.2 Information Requirements

From the previous section we can summarise the Congestion Assistant needs the following information:

- Own position
- Own speed
- The position of the tail of the jam
- Speed of the tail of the jam
- Position of the head of the jam
- Speed of the head of the jam
- Current headway

We can regard the vehicle as an autonomous unit which obtains information from its environment by means of on-board sensors (odometer, forward-radar, positioning / navigation system etc.) and collaboration with other vehicles or road-side (fixed) units. In order to make decisions all collected data will need to be aggregated and interpreted. The vehicle will be continually interpreting the signals—being sensor readings or vehicle-to-vehicle messages or driver input—to *build a model* of the driving environment. Some of the information required by the Congestion Assistant subsystems can be derived from other data. Some data can be used by several subsystems. The data can come from two sources, being internal (on-board systems) and external (via vehicle-to-vehicle communication, cellular technology, radio broadcasts, etc.), as listed in Table 2.2. The same information might be extracted from data from different sources, resulting in a means to judge the accuracy of the information.

The vehicle gathers information from its driving environment, as shown in Figure 2.2. The resulting data is interpreted and based on this—possibly incomplete or conflicting—information a decision is made. This then results in the activation of actuators, displays or the transmission of messages. In a sense, the vehicle has to become aware of its environmental context. A part of this context, the over-the-horizon view, is treated in detail in chapter A. This over-the-horizon awareness is central to this research.

2.2.1 Internal Information

Internal information is provided by internal sensors onboard the vehicle. This can be a position fix from the navigation system, a speed indication from the odometer, directional information

[1] This information can be extracted from both onboard and external sources

Internal Information	External Information
own speed	speed of tail of jam
own position	speed of head of jam
headway to vehicle in front[1]	headway to vehicle in front[1]
	position of tail of jam
	position of head of jam

Table 2.2: Internal and External Information Requirements.

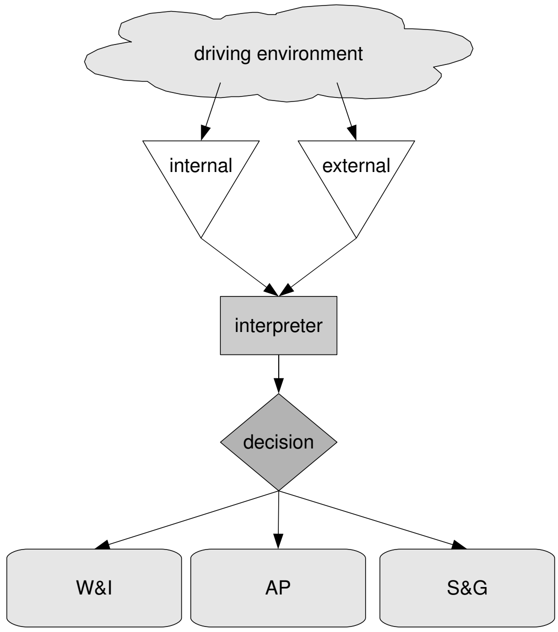


Figure 2.2: Information is interpreted, a decision is made and systems are controlled.

from a gyrometer or acceleration information from accelerometers. The presence of and distance to vehicles in front and behind can be obtained from radar, infrared, ultrasound or video [54]. This information can be tapped straight from the sensors using dedicated wires or using a bus infrastructure, for instance, the CAN-bus (Controller-Area Network) [11]. In this research we will focus on the information obtained from external sources. We will assume a method of extracting the required information from the vehicle itself is present.

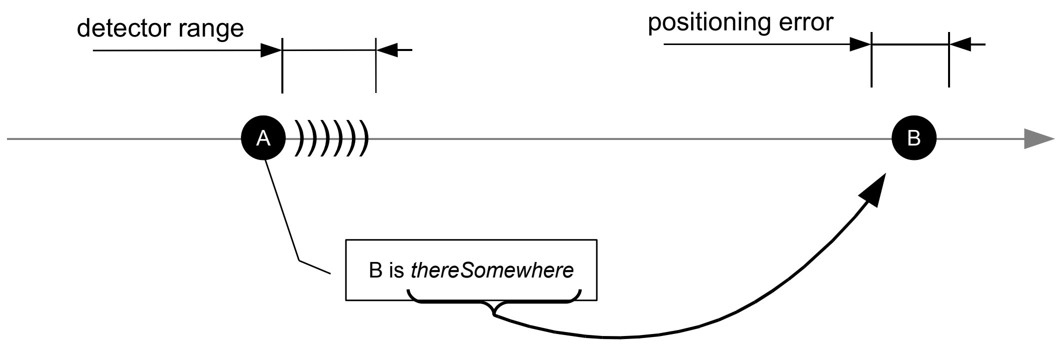


Figure 2.3: Compensating errors in external information with internal information

The information obtained from internal sensors can be used to judge or augment the information supplied by external sources. For instance, when a vehicle approaches a position captured with an accuracy of 50m, forward-looking radar might inform the vehicle that the reported obstruction is 30m closer than reported. We reason that, as long as on-board sensors are able to compensate for the position error in the external information, the information supplied by external sources is accurate enough. BMW's forward-looking radar has a range of up to 120m

[8]. A camera-based system by MobilEye [72] is able to detect objects up to 200m away. This idea is depicted in Figure 2.3. Node A has knowledge of node B's whereabouts. A knows B is *thereSomewhere*, but the positioning is not entirely accurate and includes a certain error margin. As long as A's ability to detect objects in front exceeds the error margin in A's notion of the position of B, then Node A will not be surprised when node B shows up sooner than expected.

2.2.2 External Information

The Congestion Assistant relies on knowledge of the traffic conditions on the road ahead. The positions of the tail and head of the jam (if one exists) need to become known to every equipped vehicle upstream from the jam. A way to determine these positions and how to distribute this information will be proposed in the rest of this writing. Several methods to obtain this information will be highlighted. The remainder of this thesis wil cover the development and evaluation of a means to obtain this external information, with as ultimate goal to construct an over-the-horizon view.

2.3 System Requirements

In the previous section we derived the information required for the Congestion Assistant to operate. In order to design a system to collect this information several requirements are needed to specify a solution.

The Congestion Assistant is envisioned to alleviate traffic congestion on highways. Likewise, modern cars are often equipped with a navigation tool. This results in the following requirements:

- This research focusses on highway traffic.
- For simplicity only a single-direction, single lane highway without intersections is considered.
- Traffic behaves without accidents.
- No vehicles enter or leave the road.
- Vehicles are expected to have a GPS unit on board and know their own position.
- The Congestion Assistant does not perform Safety-of-Life operations. As such Collision Avoidance functionality is not considered and the focus is on communicating the relatively slow-evolving dynamics in traffic flow, with the focus on distances more than one hop away.

Chapter 3

State of the Art

The previous chapter provided an overview of the Congestion Assistant, a system designed to increase the efficiency of road traffic in congestion. The Congestion Assistant requires knowledge on the situation on the road ahead and acts accordingly.

This chapter introduces the general context and provides a literature overview of the field of traffic and mobile communication, which may provide partial solutions. First, the phenomenon *Traffic Jam* is discussed. The notion of determining the position of a vehicle on the road is central to this research. An overview of the accuracy of GPS-based positioning is provided in Section 3.2. It is key to know the position of the tail of the traffic jam, and the own position, and do so within a reasonable margin of error.

Next, weIntelligent Transportation Systems (ITS) and their purposes will be introduced in Section 3.3. Several platforms have been proposed in literature: IEEE 1609, CarTel and Trafficopter. We will evaluate if any of these could be used in the context of the Congestion Assistant.

We will briefly touch the subject of Wireless Communication in Section 3.4, an enabling technology for many ITS applications. Finally the formal Modeling of Traffic will be treated. The reason for this is two-fold; to get an idea of the context, and to obtain a practical model to use when evaluating solutions.

We will conclude this chapter with a brief conclusion in Section 3.6

3.1 Traffic Jams

In order to talk about traffic jams a notion of the meaning of the words *traffic jam*, *traffic congestion* and *gridlock* is required. Looking up the terms in a dictionary one finds:

Jam a: to become blocked or wedged
b: to become unworkable through the jamming of a movable part or component [69]

Congestion a: to concentrate in a small or narrow space
b: an excessive accumulation [68]

Traffic Jam - Both a and b of the definition of jam can be applied to vehicular traffic, as movable parts (the vehicles) can wedge the system. Hence the term *traffic jam* - a number of vehicles blocking one another until they can scarcely move [32].

Traffic Congestion - because *jam* and *congestion* are almost similar, traffic congestion is a synonym for traffic jam.

Gridlock - a traffic jam so bad that no movement is possible [32]. In this light *gridlock* can be seen as an evolved or aggravated state of jammed traffic with the property that there is no movement.

These definitions do not readily translate to a formal model which can be used in a system. For instance, the number of vehicles involved in a jam or the length of the jam may be decisive in determining whether vehicles on a road form a traffic jam, or just dense moving traffic. The Dutch VerkeersInformatieDienst[1] (Traffic Information Service) uses a definition composed of three parts [114]:

slow-moving traffic - traffic that, over a length of at least 2 kilometers, drives at speeds below 50 km/h but generally above 25 km/h.

stopped traffic - traffic that, over a length of at least 2 kilometers, drives at speeds below 25 km/h.

slow-moving and stopped traffic - slow-moving traffic over a long stretch of road with some lumps or clusters of stopped traffic.

An often-used approach to determine the performance of a stretch of road is to calculate the weight of the traffic congestion on it. This allows to classify jams based on their impact. Policy makers can then decide that, for instance, an extra lane needs to be added to the road. It can be used in statistical analysis and to make images such as Figure 3.1 The congestion weight (ZF - Zwaarte van Files) is calculated as follows [30]:

$$ZF = \Sigma_i l_i d_i \tag{3.1}$$

Where d_i is the duration of a jam on a certain section of road i with length l_i. Summing these factors results in a congestion weight for the road, expressed in kilometerminutes. Although this provides a great tool for statistical breakdowns, calculating the severity of a traffic jam is not practical in the operational detection, as required by the Congestion Assistant. As an aside, vehicle navigation systems could be supplied with a heuristic system that calculates the probability weighted with the impact of a traffic congestion—and the incurred delay—related to time and date. This might be useful in route planning. However interesting, this is out of the scope of this research.

[1] http://www.verkeersinformatiedienst.nl

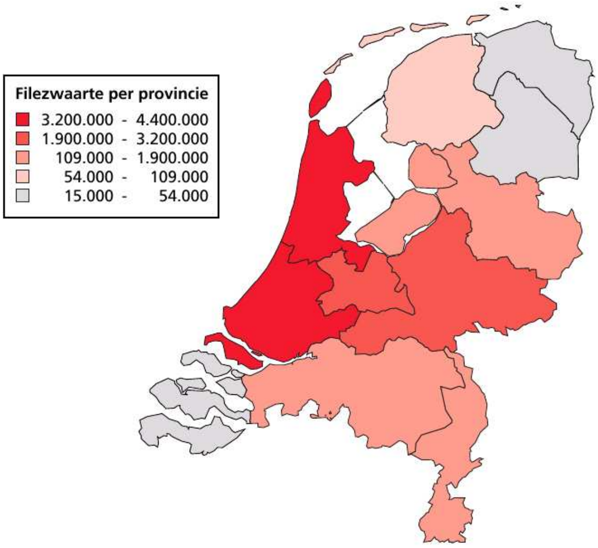


Figure 3.1: Congestion weight in the Netherlands in 2006 [30]

3.1.1 Causes of Traffic Jams

The most basic and obvious cause of traffic jams is simply that the supplied volume of traffic is too large, and the road system cannot cope, i.e. there is not enough capacity to satisfy the demand. Several factors influence the forming of traffic jams by reducing the capacity at a certain point or on a certain stretch of road, or by increasing the demand:

Increase of demand:

- Rush Hour
- Large events (concerts, begin of holidays, large scale evacuations)
- Occupancy rate (number of persons per vehicle)
- Amount of road space required per vehicle (physical size, safety margins)

Decrease of capacity:

- Road construction works
- Road conditions due to weather such as snow, rain or wind
- Visibility conditions such as fog or driving at night
- Merging of lanes
- Overtaking trucks
- Merging traffic
- Parked vehicles or obstacles alongside a road

Because a road network consists of roads and junctions a decrease in flow at one point can easily spread to another location. An example could be a traffic jam on a highway because the roundabout several kilometers removed from the exit is congested.

There can be a complex interplay between factors, influenced by human behaviour. Effects well-known to frequent road users include phenomena such as shockwaves that ripple upstream when sudden braking occurs, a congestion on the lane adjacent to an accident because of on-lookers etc. Many of these phenomena have been formalised by various scientific disciplines ranging from mathematical fluid-dynamics [45, 123] and economical theories such as *the tragedy of the commons* [26] to psychological concepts such as self-control [41]. One of the mathematical models (the Intelligent Driver Model [102]) has been used in this work to model a road with realistic driver response. The modeling of traffic is treated in Section 3.5.

3.1.2 Traffic Jam Detection Systems

Loop detectors embedded in the road can be used to measure the number of vehicles per time unit on a certain road or a certain lane on a road. Loop detectors are usually built as coils of wire of which the inductance changes due to the vehicle moving over the coil's magnetic field. The inductance is measured and a vehicle moving over the coil results in a certain pattern of fluctuations. Depending on the inter-vehicle time and the average length of vehicles, traffic jam conditions can be detected [14]. In contrast to loops, which are fully automated detection systems, traffic helicopters and cameras mounted alongside the highway can also be used to detect traffic jams. The information is collected at Traffic Information Centers.

In a pilot project carried out by Vodafone, several GSM base stations along a German highway were instrumented to enable the monitoring of mobile phones in the area [119]. Using (anonymous) handover information a mobile phone can be tracked and average velocity can be calculated. Correlating the average velocity to a (course) indication of position can be used to deduct the average traffic conditions on the highway, because it is most likely the phones are located in vehicles moving on the road.

One of the downsides of infrastructure-based detection mechanisms, is the huge investment needed to realise them. To install loop detectors, the road needs to be cut open and new asphalt needs to be applied, resulting in construction works impacting traffic flow. Mounting cameras alongside roads and hardware at GSM antennas obviously has its costs. Moreover, all methods rely on communication networks to supply the data to central operations centers. Furthermore, only a limited part of the road is covered by these systems. Although they are generally installed at locations where traffic jams frequently occur, they are not installed at locations where traffic jams occur less frequent. As a result, the system can miss an entire traffic jam. Traffic Helicopters can somewhat mitigate this problem by patrolling along the highway network and reporting any sighted traffic jams, but operating a traffic helicopter service is expensive business and it might take a while before a jam is noticed.

3.1.3 Traffic Jam Notification Systems

Traditional News Broadcasts - For several tens of years, radio news broadcasts have supplied traffic congestion information. Generally on an hourly basis, listeners are updated on where the traffic jams are located on the highways. Because in the Netherlands this list can already become quite long—and airtime is an expensive and limited resource—short jams are often omitted. The Dutch VerkeersInformatieDienst supplies the information on their own website, to radio news broadcasts and even to a hotline[2] which drivers can call to obtain timely and accurate information. The information is often formatted as follows:

```
A4  Delft - Amsterdam
    tussen Schiphol en knp. Badhoevedorp
```

[2] http://www.0900-8855.nl/

```
4 km langzaam rijdend tot stilstaand verkeer
(vertraging: minder dan 5 min)
```

Which means that on highway *A4* there is 4km slow-moving and stopped traffic (notice the use of one of the terms given in section 3.1). Because the *A4* connects multiple cities and to indicate direction the statement “Delft - Amsterdam” is added. To further narrow it down, junctions are given: “Schiphol” and “Badhoevedorp”. Notice, from Figure 3.2, that the section indicated is about 4km long. We know there is a congestion on that section of road, somewhere. This information is detailed enough for a person driving on the *A4* who is merely interested in the presence of a jam, and the incurred delay (less than 5 minutes in this case but this obviously is a rough estimate). The position of the head and tail or the jam are not known, as a result the information is not detailed enough to be of use for the Congestion Assistant.

Figure 3.2: The A4 between Schiphol and Badhoevedorp [37]

Traffic Message Channel - (ISO 14819-1) is a service available via RDS (Radio Data System) on conventional radio broadcasts throughout Europe and North America. The information is typically digitally coded and can easily be integrated with navigation systems. A TMC message consists of an event code and a location code in addition to a time stamp. The sources of the information can be loop detectors, traffic helicopters, road-mounted camera’s etc. which deliver information to a Traffic Information Center. The information is coded in Alert C [31], which is a plain-text standard for the exchange of traffic information. Besides still relying on “traditional” means of traffic jam detection, this system can instantly inform drivers of traffic jams, closed roads and many more traffic-related events. Many present-day Satellite Navigation systems can use the TMC events and react to them. For instance, if a road is closed the navigation system automatically calculates a new route. The fact that TMC relies on traditional means of (among others) traffic jam detection means that it can be slow to react to sudden changes in traffic. As a result the information can already be old and inaccurate.

Added to that, it has been proven not to be too hard to falsify TMC messages [6] using commercially available off-the-shelf hardware. This is particularly disturbing because most people blindly trust their navigation system, even if it is saying the road home is closed due to an aircrash, as in Figure 3.3.

Figure 3.3: Spoofing an aircrash using RDS-TMC [6]

TMC uses a Location Database on a national level in order to deduct the position of an event. The encoding is standardised as ISO 14819-3 and uses a 16-bit position code [6]. Such a position code can denote a section of road (e.g. "M6 Junction 10" [10]) or even a parkinglot. In the Netherlands the Traffic Information Location Database (VILD) contains all the highways and main roads and is maintained by the Ministry of Transportation [9]. The method of denoting a position might be of interest during further research. An evaluation of means to express positional information is provided in Appendix A.

3.2 Accuracy of GPS-based Positioning

A Global Positioning System is a very important ingredient of Intelligent Transportation Systems. Because one of the core activities of transportation is displacement, i.e. changing one's position, it is key to get a notion of this position. A Global Positioning System provides a cost-effective and easy-to-use alternative to methods that have been used in marine navigation for centuries. Such techniques as Dead Reckoning, Piloting and Celestial Navigation are not very practical for road vehicles where the driver and navigator are often the same person.

In Dead Reckoning one uses a known location combined with speed and heading information to estimate the current location. Piloting relies on the use of landmarks and their relative bearing. Celestial Navigation relies on clear visibility of objects on the firmament. Furthermore, it requires the use of angle measurement equipment such as a sextant and the consulting of almanacs. These techniques are all far too cumbersome to be of any use in a vehicle because speed and heading change often, the sky generally is not visible through the roof or clouds and a driver is generally not skilled in using a sextant and almanac. Besides, a driver has to keep his eyes on the road and cannot go through pages of tabular data to determine which course to follow, a task which—in marine navigation—often requires a special operator.

Electronic Navigation eases the problems involved with manual navigation. Techniques used are Radio Navigation, Radar Navigation and Satelite Navigation. When using Radio Navigation a directional antenna is pointed in certain directions. When the signal of a beacon is strongest the bearing of the beacon with respect to the observer can be measured. Using several of such readings can provide a triangulated position relative to the beacons. When the location of the beacons is known the own global position can be calculated. This method is the electronic equivalent of Piloting, and relies on active beacons, whereas Radar Navigation relies on the reflections of the radar signals transmitted by the vessel itself. Using these reflections known landmarks can be identified. Based on the landmarks and their position the own position can be calculated.

Satellite positioning relies on a set of geostationary satellites in orbit around the Earth. These systems use a relatively small receiver unit which receives timing information from several satellites. The satellites are equipped with very accurate clocks. The distances to the satellites can be calculated and because the positions of the satellites are known—they are geostationary—the position of the receiver on the Earth's surface can be triangulated, as illustrated in Figure 3.4. This can easily be done in small circuitry in a small device.

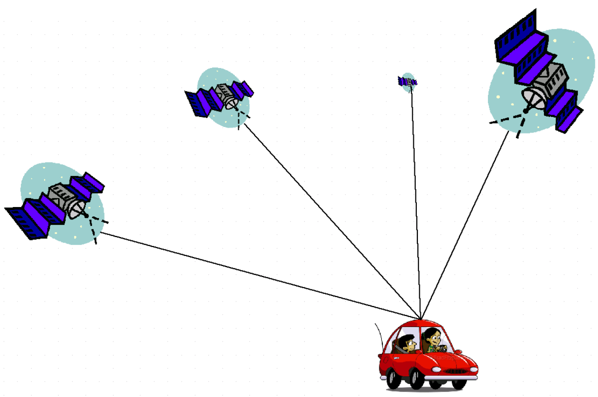


Figure 3.4: Three or more reference points allow triangulation

The most widely used GPS implementation is that put into place by the U.S. military. Europe and Russia are also working on their own systems (called Galileo and GLONASS) but only the U.S. system will be described here.

The U.S. GPS, Popularly known as “GPS system” and formally known as NAVSTAR GPS - NAVigation Satellite Timing And Ranging Global Positioning System, was deployed by the U.S. military and usage was restricted to military applications. The first satellite was launched in 1978, currently at least 24 satellites are in service with 5 in-orbit spares. In 1983 Soviet interceptors shot down a Korean commercial airliner (flight KAL 007) that had mistakenly wanderred into Soviet airspace. U.S. President Ronald Reagan issued a directive that GPS signals should be available to everybody, because more accurate navigation might have prevented this incident [82].

The GPS was, just like the Internet, conceived by the military. And just like the Internet, GPS presently enjoys an enormous commercial use, with annual equipment sales of over $20 billion, of which 95% is used by civilians [82].

In a paper published in 1995 by Zito *et al.* [88] the usefulness of GPS data for traffic monitoring purposes is discussed. At that time Selective Availability (an error induced in the system by the U.S. military for ‘national security reasons’) was still active. Field tests were performed and a GPS position measurement was found to drift within a 43 by 66 meter area with at least four satellites within line of sight during measurements. It was concluded GPS position measurements are accurate within 50-100 meters [88, 77]. Furthermore it was found that, due to the drift in position, it is more accurate to obtain the speed from a direct measurement than from multiple measurements of position over time and dividing the distance between measurements by the difference in time. Based on one observation per second using uncorrected GPS position readings the speed error could theoretically be as large as 180 km/h [23].

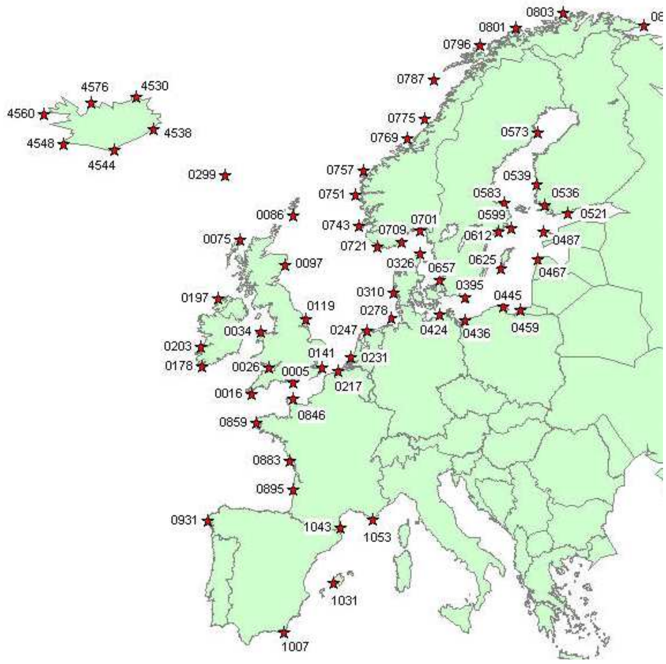
A direct measurement is independent of the position calculations. The method described by May in [71] makes use of the Doppler effect to measure the rate of change in the GPS signal and can derive the speed directly. It works on the level of the electro-magnetic signal and not on the level of the information contained therein. This is reported to correspond closely with the on-board vehicle instrumentation [88].

The work presented by Zito *et al.* in, among others, [88] and [23] was carried out in a period when Selective Availability was active. On May 1^{st} 2000 U.S. President Bill Clinton declared that Selective Availability would be discontinued in an “effort to make GPS more responsive to civil and commercial users worldwide” [17]. The removal of SA resulted in a far more accurate position fix with location errors dropping to around 10 meters for Non-differential GPS and 4 meters for differential GPS [2]. Other research, carried out in London, showed the same improvements for non-differential GPS but very little improvement for differential GPS [77]. This can be attributed to the urban environment and multipath effects due to buildings. It is however identified that GPS without SA still does not satisfy the requirements for Advanced Transport Telematics Systems (ATTS) or Intelligent Transportation Systems as defined by Chadwick in 1994 [13], where the required accuracy for routing and emergency location positioning is postulated to be 5m with an availability of 99.7%. The conclusion of [77] is that such accuracy is possible but the availability is not high enough when using stand-alone GPS. It is proposed that the GPS signal is augmented with a signal in the 100KHz range because of better signal propagation in urban environments.

An example of such a system is the *Eurofix* system developed at Delft University. This system uses LORAN-C signals at 100KHz for the transmission of differential corrections [78], whereas “normal” DGPS uses a frequency around 300KHz.

Discussion

GPS has gone through tremendous increases in accuracy since its introduction in 1978, especially the removal of Selective Availability opened up a host of new application opportunities.



ALRS	STATION NAME	F. Khz
0005	St Catherines Point	293.5
0016	Lizard	284.0
0026	Nash Point	299.0
0034	Point Lynas	305.0
0075	Butt of Lewis	294.0
0086	Sumborough Head	304.0
0097	Girdle Ness	311.0
0119	Flamborough Head	302.5
0141	North Foreland	310.5
0178	Mizen Head	300.5
0197	Tory Island	313.5
0203	Loop Head	312.0
0217	Oostende	311.5
0231	Hoek van Holland	287.5
0247	Ameland	299.5
0278	Düne	313.0
0299	Myggens	303.5
0310	Blåvandshuk	296.5
0326	Skagen	298.5
0395	Hammerodde	289.0

Table 3.1: Some DGPS beacons in Europe [27]

Nowadays position fixes with an accuracy of a few meters are possible. Table 3.2 shows the percentage of position fixes which fall within the accuracy levels as shown in the first column. A position fix with 20-meter accuracy is possible 99% of the time now that Selective Availability is disabled but used to be 44% when SA was still active. These values are based on experiments performed by Ochieng and Sauer in [77] in which there was good satellite coverage so the error which is measured is the error inherent in the system.

In aviation and the maritime sector there usually exists a clear line of sight with GPS satellites. Land vehicles suffer the problem that sometimes the view of the sky can be greatly reduced due to obstructions such as buildings, terrain and trees. Sometimes no GPS coverage is available at all (e.g. inside a tunnel). In [74] the use of Kalman filtering algorithms and gyro/odometer dead reckoning is proposed to still provide a position indication when no GPS fix is available, a practice often called *sensor fusion* [89]

Accuracy level (m)	Fix density with SA (%)	Fix density without SA (%)
5	12	42
10	24	74
20	44	99

Table 3.2: Accuracy of GPS with and without Selective Availability [77]

3.3 Intelligent Transportation Systems

ITS, or Intelligent Transportation Systems, evolves around vehicles which are equipped with some intelligent components. Often this is visualised as a unit dubbed On-Board Unit (OBU) which is connected to the vehicle's systems and equipped with means to exchange information with the environment.

A literature search is performed to find out more about communication platforms proposed for Intelligent Transportation Systems. Our focus will not be on the applications such as traffic management, safety and efficiency driver support systems and the like but more on the level of communication. IEEE 1609 will be highlighted. This is a family of standards for Wireless Access in Vehicular Environments (WAVE). This provides a framework of which IEEE 802.11p is also part. Next, two different systems proposed to gather information and propagate this through the network, CarTel and Trafficopter, will be discussed.

An Intelligent Transportation System (ITS) has been the dream of transportation and automotive engineers for a long time [115]. The move towards more intelligence in the transportation system is also referred to as Road Transport and Traffic Telematics (RTTT) [19]. Nissan in Japan was one of the first to perform ITS-related research in the 1980's, followed by the California PATH (U.S., 1996) and CarTALK (Europe, 2000) projects, and many others. With the advent of wireless radio communication the exchange of information between vehicles and smart units alongside the road is enabled, opening up possibilities well beyond those of ordinary traffic signs and fixed-point observation systems.

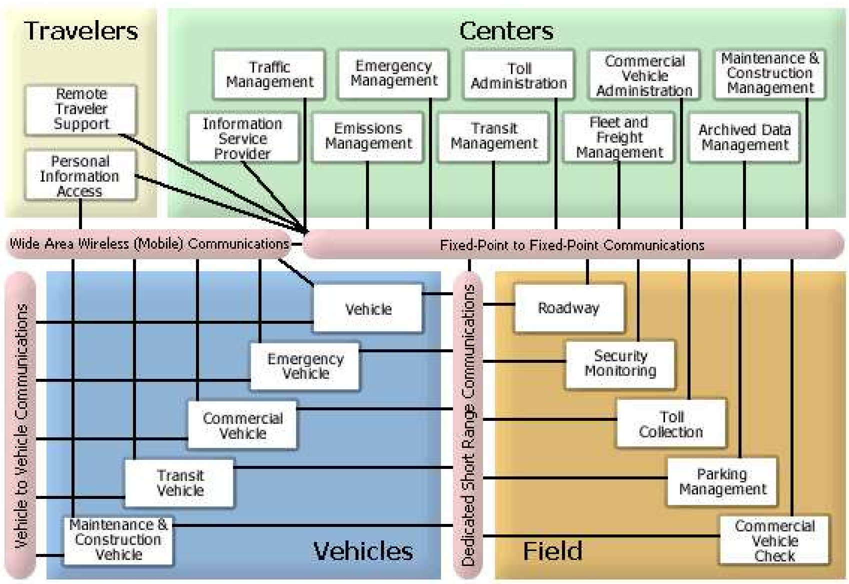


Figure 3.5: Entities within the U.S. DOT ITS architecture [107]

The U.S. Department of Transportation (DOT) has defined the "National ITS Architecture" [47, 107]. This architecture provides a common framework for planning, defining, and integrating intelligent transportation systems [107]. It defines functions, physical entities and data flows to enable the development of an intelligent transportation system. Such a system is based on communication between all involved entities. This is envisioned to pave the way for synchronisation and coordination. One of the ultimate envisioned goals is to have a Traffic Control feature [52], designed to improve the flow of traffic by giving preference to transit and other high occupancy vehicles, and minimise congestion while maximising the movement of people and goods. The required data is gathered from the system, fused and then used to determine an optimum assignment of road resources.

What the U.S. DOT's ITS architecture is really about is integration of just about anything that has anything to do with transportation, and then manage it all. It probably does not come as a surprise that there are some concerns with respect to such a Big-Brotheresque system [3] where law enforcement can use ITS applications for surveillance and drivers can be punished (by means of fines or limitations to license) for 'bad' behaviour on the road, almost the instant they cross the speed limit. The fear is primarily because it might become possible to track every vehicle, which would be a threat to privacy [122]. This, of course, is fuel for conspiracy theorists sketching a dystopian view of the future but there also are more moderate calls for caution [33], where the conclusion is that ambient intelligence technology (such as ITS) goes beyond most of currently existing privacy-protecting borders.

Abstracting from large architectures and ethical deliberations, ITS is about coordination. This coordination can be by means of on-board sensors or by means of sharing information with the environment. This communication can be roughly divided into four levels:

Communication between vehicles and road-side units is called **Vehicle-to-Infrastructure (V2I)** communication, using Dedicated Short Range Communication (DSRC). Examples of this could be Electronic Toll Collecting (ETC) [24] and road side beacons signaling, for instance, dangerous traffic situations such as a train approaching a railway crossing. The goal is to make vehicles more aware of the road, and vice-versa.

Vehicles on the road can also communicate among each other by **Vehicle-to-Vehicle (V2V)** communication. This way vehicles that meet by chance can cooperatively exchange information on a scale that exceeds the use of signaling lights on vehicles to, for instance, indicate braking or a lane change. The goal is to make vehicles more aware of each other and share knowledge which might be of interest with regard to safety and efficiency.

A third level of communication is defined, where the vehicle connects to the fixed network, often called **Vehicle Infrastructure Integration (VII)** [39]. This connection can then be used for sensor-data gathering, where the vehicle functions as information provider pushing information to Traffic Operations Centers. The connection can also be used as traffic information distribution channel. The vehicle then consumes information produced by, for instance, a Traffic Operations Center.

A fourth level is envisioned to connect a vehicle moving on the road to the Internet, giving passengers connectivity resembling that of traditional internet connections [115].

In order for such a system to be feasible it has been identified that communication of private and commercial nature may also use the communication facilities. This is to encourage development and adoption by means of commercial interest [53, 132]. In order to cope with different types of messages, to guarantee fairness and meet requirements of safety-relevant messages several Quality of Service schemes have been proposed [132, 97, 49]

The multi-hop capability of inter-networked vehicles (known as a VANET - Vehicle Ad hoc NETwork) enables several interesting supplements for the cellular wide area communication systems. It is identified in [127] that the vehicles may "complement the cellular infrastructure in hot spot areas where the system gets overloaded and it may be favorable for vehicles to assist one another in reaching the base station (BS) rather than continuously competing to access the uplink". Secondly, it is projected that the vehicles may also extend the coverage of cellular networks, because out-of-range vehicles can still reach the BS through a multi-hop link.

3.3.1 IEEE 1609

On January 9^{th}, 2006 the U.S. Department of Transportation and the IEEE standardised a Family of Standards for Wireless Access in Vehicular Environments (WAVE) known as IEEE

1609 [47]. IEEE 1609 defines an architecture and a set of services and interfaces to support secure vehicle-to-vehicle and vehicle-to-infrastructure communication. The envisioned uses are vehicular safety applications, enhanced navigation, automated tolling, traffic management and many more. IEEE 1609 has two important entities: the On Board Unit (OBU) and the Road Side Unit (RSU). Communication between OBU and RSU (V2I) and OBU to OBU (V2V) is part of the standard.

IEEE 1609 has the following properties: [115]:

- Based on IEEE 802.11p
- Range up to 1000m
- Data rates 6-27 Mbps
- 7 licensed channels in 5.9GHz range
- latency ∼50ms
- Security can be enforced using PKI
- long term stability (because it is controlled by FCC and standards)
- designed to accommodate IPv6

The system is designed so as to accommodate both safety and commercial services. It will also support drivers in keeping their vehicles from leaving the road and provide assistance at intersections by means of driver support functionality. The network can also be used for surveillance, resulting in faster detection of damage to roads (potholes, black ice, snow etc.). The system also provides traffic signaling, incident response and impact mitigation.

IEEE 1609 consists of four standards [47]:

1609.1 – Resource Manager. Manages access to resources, defines message formats and data storage formats and communication interfaces between components. Furthermore, it specifies the types of devices that may be supported by an OBU.

1609.2 – Security Services. Defines message formats and processing.

1609.3 – Networking Service. Defines network and transport layer services such as routing and addressing. IPv6 is defined for communication in addition to Wave Short Messages (WSM), an efficient WAVE-specific protocol which can be directly supported by applications. The WAVE protocol stack is defined by this standard.

1609.4 – Multi-Channel Operations. An anhancement to the IEEE 802.11 Media Access Control is proposed to accomodate WAVE operations

Security and Privacy

The correct functioning of some (if not most) features in ITS depends for a great deal on the security of the applications, the integrity of the messages exchanged and the authenticity of the sources. It is very undesirable if a rogue transponder were to cause phantom traffic jams or falsely reports a road is closed, causing the rerouting of all other traffic. Furthermore, in order to gain public acceptance a certain degree of privacy has to be guaranteed. Among the challenges here are establishing trust between vehicles and infrastructure and vehicles among another, ensuring message authenticity and establishing secure sessions [130]. And all of this has to be done fast enough to be of use in a highly mobile environment. It is identified that communication must be anonymous and untraceable. Because of the enormous size of the network (on a nation-wide or even continental scale) scalability is a great issue. To enable trust a Vehicle Public Key Infrastructure (V-PKI) is presented in [130], which is a central issue in IEEE 1609.2. Security will not be a central theme in this research.

3.3.2 Other ITS approaches

IEEE 1609 appears to be designed for several tasks which can be summarised as follows:

- Distributing information from infrastructure to the vehicle. This can be road information (closed roads, traffic jam information) and connection to the public Internet.
- Data collection by the vehicles. Vehicles function as the eyes for government bodies and companies that manage the road networks.

The collecting of information by vehicles has also been described in a system called CarTel [46]. Vehicles are used as sensor-information providers. Because vehicles are relatively free-roaming a large area can be covered with a single sensor over time. An analogy is the tracking of wildlife. The information is (locally) stored in a database and transferred to a centralised portal whenever connectivity occurs (to cope with intermittent connectivity). The idea is to design a modular mobile sensor platform that can be used for various research, maintenance and other tasks. The design is in the same line as IEEE's 1609 but extends to (sensor)information outside the scope of traffic. The possible tasks of CarTel include environmental monitoring (pollution sensors), automotive diagnostics (monitoring of vehicle, bad driving tendencies) and geoimaging (capturing of images for landmark-based routing). Another interesting envisioned feature is the use of the system for *data muling*. A vehicle can function as a carrier for information captured by remote sensor networks. For instance, when there exists no direct link between two communication peers to carry the data electronically a physical carrier can be used—a vehicle in this case.

Both CarTel and IEEE 1609 focus on obtaining data from the vehicles or distributing it to the vehicles. CarTel uses a centralised portal [46] where, for instance, a speed overlay for a certain route can be requested, see Figure 3.6. Here the speed information is an overlay to a road map (Google Maps [38]) and travel time can be calculated for a certain route. From the paper it is not clear if this information is also available to drivers in their vehicles and if the measurements are up-to-date but it surely fits in the future visions of ITS to have a route planning application in the vehicle which takes current traffic loads into account.

The CarTel system is a sort of datamining using a mobile sensor network's readings as input. Data is stored in a centralised location (hence the term *portal*). This works fine for observing long-term trends and for the strategic driving tasks—the planning of a route—but might not work for tactical or operational driving tasks, which is what the Congestion Assistant is envisioned to do. A request for information about the road ahead from a mobile node has to traverse the V2I interface and fixed Internet in order to obtain information about the next few kilometers from the portal. For delay-critical information an Ad hoc Distributed approach (Vehicle-to-Vehicle) seems more sensible.

A distributed approach based on pure V2V communication is presented in [73], where Moukas *et al.* introduce a decentralised, distributed and self-organising system called Trafficopter that collects and distributes traffic information. Trafficopter has no central authority and relies on multi-hop short-range communication. It functions by exchanging two types of messages:

Beacon-like messages contain position (X,Y), speed and direction of the transmitting vehicle and are transmitted at regular intervals.

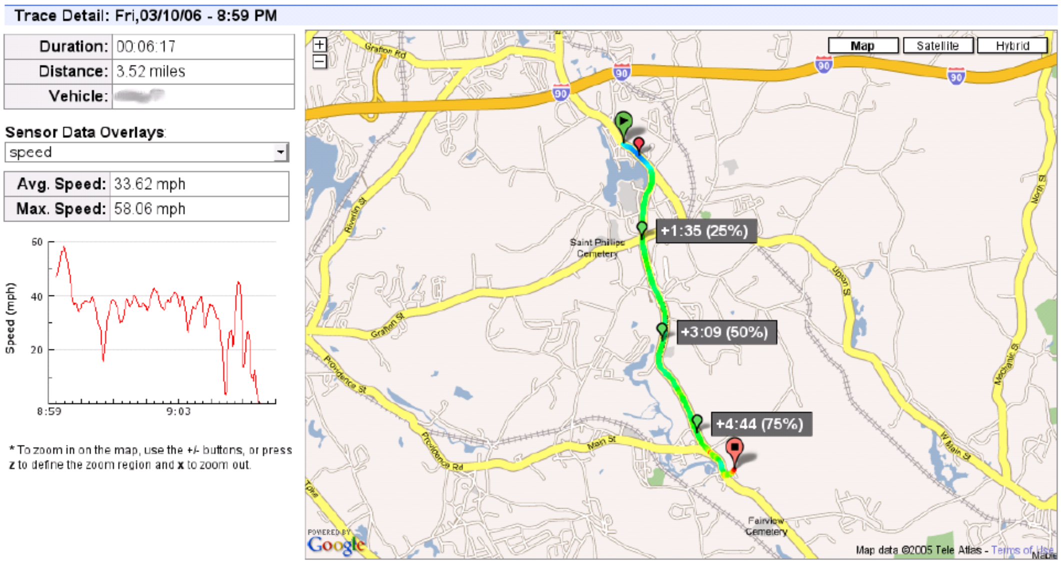


Figure 3.6: CarTel's portal with speed overlay [46]

Query messages are sent by a node to acquire information regarding the route to a certain location. Using position based routing and caching information on the state of traffic on the route to the destination is obtained.

The exact nature of the disseminated information is not further clarified in the article. How the "state of traffic" is expressed in the information exchanged and how this is then interpreted by the node that issued the query remains unsolved. Focus is on the influence of caching and a border-aware or distance-based MAC routing protocol. Simulation studies have been performed.

Trafficopter extends the driver's view beyond the horizon and it is fully distributed, information is generated close to where it is used and does not traverse fixed infrastructure through central servers. The focus is on urban roads with lots of traffic, where explicit querying of the network using end-to-end position-based routing with caching (to potentially reach the required information with fewer hops) is used to find a physical route. Information becomes available after explicitly querying, so there must exist a need at the consumer-side to obtain this information. More general, a node must know what information to query for.

3.3.3 Discussion

A lot of research has been carried out in the area of Intelligent Transportation Systems. Our focus is primarily on what communication frameworks have been defined. There are roughly two approaches to supplying services needed by intelligent vehicles; centralised and distributed. The U.S. DOT ITS architecture is primarily based on the assumption that a lot is taken care of by a centralised 'Traffic Operations Center', although the model as shown on page 26 also allows for vehicle-to-vehicle communication. With respect to our goals it did not provide a usable framework. The Trafficopter system proposed in [73] is the closest any research in literature comes to what is required for the Congestion Assistant. The system that is to supply over-the-horizon awareness cannot be based on the Trafficopter system though, because of some fundamental differences. Trafficopter is query-based and as such retro-active, while the Congestion Assistant demands pro-active information. Trafficopter is geared towards finding routes in urban environments, while the Congestion Assistant will operate on highways and is not used for finding routes but for the average speeds on a (predefined) route, especially with respect to traffic congestion.

As a result, despite the great amount of literature on ITS and driver assistance systems, there is little material to use from previous research to fullfill the Congestion Assistant's requirements.

3.4 Wireless Communication

Most people nowadays are familiar with Wireless LAN technology. IEEE 802.11b is a Wireless Local Area Network technology that was introduced in 1999 as a network cable replacement technology, enabling easy mobility in between sessions. It is ideally suited for situations where the population of computers fluctuates constantly, such as in airport terminals or for local networking when cables are either too cumbersome or undesirable—either for aesthetic or structural reasons. 802.11b can operate under small mobility such as carrying a laptop computer from one table to the other but has never been meant for high speeds.

There have been, however, some experiments with 802.11b at very high speeds [21], where an 802.11b node was built into the payload compartment of a supersonic Hydra 70 MK 4 rocket which was then launched over an area covered by 802.11b base stations. The rocket travelled over 650 m/s while still providing data from a small camera also installed in the payload compartment, until it got out of range. No in-flight association with different base stations occurred. It was concluded 802.11b had some serious shortcomings with respect to mobility. Furthermore, these tests were performed in a desert away from sources of interference; urban environments will be less hospitable to radio communications.

Several wireless communication technologies have been introduced in the past years. Most are in the centimeter-wavelength range (2.4GHz, 5.8GHz, 5.9GHz) but it is also suggested to use milimeter-wavelength (60-78GHz) waves because these frequencies are less sensitive to Rayleigh Fading [79]. A proposed scheme for V2V and V2I communication is that of Dedicated Short Range Communication (DSRC). It is designed to provide a reliable communication link between vehicles and roadside beacons [19, 24] and other vehicles [79, 87].

DSRC is designed to support different media types. Microwave transmission at 5.9GHz is the most prevalent choice, but an infrared (850nm) draft is also defined [106]. Initially the DSRC specification included a simple passive physical layer [24] from the RFID-family but later on the PHY was replaced with IEEE 802.11a, which uses Orthogonal Frequency Division Multiplex (OFDM). Several modifications to 802.11a are proposed, such as 802.11a /RA, where the RA stands for Roadside Access [91] and priority-based service differentiation [97]. This lead to the draft standard of 802.11p. It is designed for DSRC and looks to be a promising wireless technology for enhancing transportation safety and traffic efficiency [4, 127].

3.4.1 IEEE 802.11 family

IEEE 802.11 is a family of standards for Local and Metropolitan Area Networks (LAN/MAN). 802.11 generally specifies a means to access the medium referred to as Media Access Control (MAC) and the Physical Layer (PHY) [48].

802.11 is often referred to as Wireless Local Area Network (W-LAN) or WiFi (Wireless Fidelity). Several varieties exist (a, b, g and the new n and p) which are designed to be conformant to the 802.11 basics. Especially in the area of MANET / VANET research 802.11a and 802.11p are popular [121, 132, 127, 129, 91, 28]. The major differences between the variants is the spectrum in which they operate (2.4GHz ISM for 802.11b, g and n, 5GHz for 802.11a, 5.8/5.9GHz for 802.11p) and the modulation technique used. 802.11a , g and p use Orthogonal Frequency Division Multiplexing (OFDM) whereas the b variant uses Direct Sequence Spread Spectrum (DSSS) but may also use Frequency Hopping Spread Spectrum (FHSS).

IEEE 802.11 was initially designed for portable devices such as laptops and notebooks, with low mobility in mind. To suit the needs of high mobility involved in Intelligent Transportation Systems the 802.11p amendmend is under development. One of the goals is to use a frequency band in which licensing is required because the bands presently used for domestic and commercial use are crowded, resulting in interference and performance degradation. By licensing a band strictly for automotive use the reliability of the communication will increase. This band will be between 5.86 and 5.92GHz. 802.11p will function as a carrier for Dedicated Short Range Communications (DSRC) and as such the spectrum will be divided into seven channels.

Because this mobile type of W-LAN will still be based on 802.11 the basics will be introduced next, largely citing the standard IEEE 802.11-2007 [48]. The 802.11 MAC defines two access methods: Point Coordination Function (PCF) and Distributed Coordination Function (DCF), as shown in Figure 3.7.

The Point Coordination Function

Under Point Coordination an Access Point is responsible for managing resources within the area covered, referred to as the Basic Service Set (BSS). An Access Point may be connected to a Distribution System which connects to the wired Internet or other Access Points. The centralised coordination by the PCF has several benefits with respect to efficiency, reliability and fairness. Because of the need of a point coordinator in the network (and a point coordinator to every station conform a one-to-many relation) the PCF cannot be used in VANETS, or MANETS in general.

(a) The Point Coordination Function (PCF) is often used when infrastructure is available

(b) The Distributed Coordination Function (DCF) is often used when no infrastructure is available

Figure 3.7: IEEE 802.11 Modes of Operation

The Distributed Coordination Function

The Distributed Coordination Function (DCF) is also called 'basic access' because it lacks many features present in PCF. DCF is the foundation of PCF but can also be used by itself, primarily when there is no central authority (Access Point) present. A group of stations operating under the DCF are called an Independent Basic Service Set (IBSS).

DCF uses a CSMA/CA scheme: Carrier Sense Multiple Access with Collision Avoidance. This scheme is similar to the one used in Ethernet (CSMA/CD) with the exception that it is not possible to detect a collision so the system is altered to avoid collisions to the best of its ability. The reasons why collisions cannot be detected are the following:

- A transmitting node sends so much power into the medium that it would not hear other broadcasts. Due to the propagation of the radio signal the power decays exponentially as the distance increases. In order to cover more area a lot more power is needed. Contrast this to the isolated copper wire of Ethernet in which propagation loss is hardly of a significant magnitude.

- Collisions often occur at the receiver, with the transmitter oblivious of the collision. This is also due to the propagation properties of the medium.

When a station wants to transmit a frame it senses the medium. If the medium is sensed to be busy (i.e. another station is transmitting) the station waits, until the medium becomes idle. When the medium is sensed to be idle, the transmission can commence. Collisions can still occur and will result in corruption of the received frame. A special field at the end of the frame contains the Frame Check Sequence (FCS), a Cyclic Redundancy Check (CRC) calculated over all the other fields. If this CRC is not correct, the transmission is not received intact and will be discarded.

Request to Send / Clear to Send (RTS/CTS) functions as Virtual Carrier Sense and is designed primarily to combat the hidden terminal problem. RTS/CTS effectively reserves the medium in both collision domains for the planned duration of the transmission.

The RTS/CTS scheme introduces additional overhead into the system, both with respect to link load/utilisation and delay before transmission of the actual data can commence. In an environment where the packets that are to be transmitted are small it is recommended not to use the RTS/CTS sequence, this is called the RTS Threshold and is a value between 0 and 3000 octets [48]. Generally a station is configured to use RTS/CTS never, allways or based on a threshold. In the latter case RTS/CTS is used when the frame is longer than a specified length.

Carrier Sense and Media Access

Carrier Sense determines if the medium is busy or not, an activity called Clear Channel Assessment (CCA). When an ongoing transmission is detected the station refrains from accessing the medium. If transmission were to proceed both the ongoing and the newly initiated transmission would be thwarted, wasting precious resources because both stations will have to retry transmission at a later time.

The PHY provides a physical CS mechanism to the MAC layer, and the MAC layer provides a virtual CS mechanism by means of the Network Allocation Vector (NAV). The NAV is based on information overheard in RTS/CTS frames and indicates the predicted time the medium will be busy. A Request-To-Send is transmitted by the sender, upon which a Clear-To-Send will be transmitted by the receiver. All surrounding nodes now are aware of the sender and receiver's plans to use the medium and can set their NAVs accordingly. Obviously, the medium is determined to be busy when the station itself is transmitting.

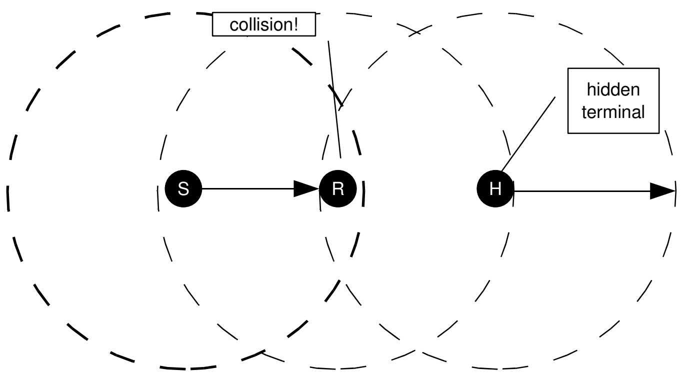


Figure 3.8: A hidden terminal

The RTS/CTS scheme is designed to combat the so-called *hidden terminal problem*, as illustrated in Figure 3.8. Here station S wants to send a packet to station R. S senses the medium, and finds it idle so it transmits. H, in the meantime, also senses the medium and also finds it idle. If H were to transmit a packet (either to R or any other node) H's transmission will collide with S's transmission and R will receive garbled bits. By means of the RTS/CTS mechanism

H can be informed that R intends to receive a message from S, H is kindly requested not to interfere.

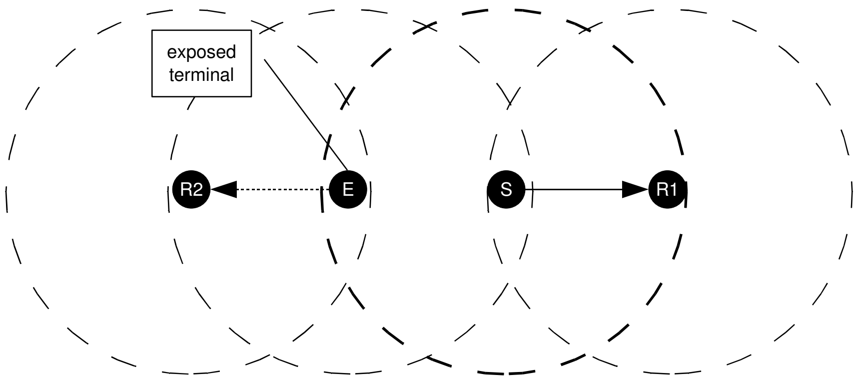


Figure 3.9: An exposed terminal

Unlike the hidden terminal problem, the *exposed terminal problem* does not cause collisions but is caused by RTS/CTS. As illustrated in Figure 3.9 station S wants to send a packet to R1. Node E overhears the RTS sent by S in preparation of the transmission of the packet and sets its NAV, even though E too has a packet it wants to transmit to R2. These two transmissions could have been executed in parallel perfectly well because E and S will not interfere each other's destinations. The result is a loss of capacity, as E will now wait for S to finish. There is no scheme to deal with this problem defined in 802.11 [126].

In ad hoc networks the NAV (and hence RTS/CTS) can be used, but a lot of the communication is of a broadcast nature. For broadcasts RTS/CTS is not used, as a result communication relies solely on physical CS.

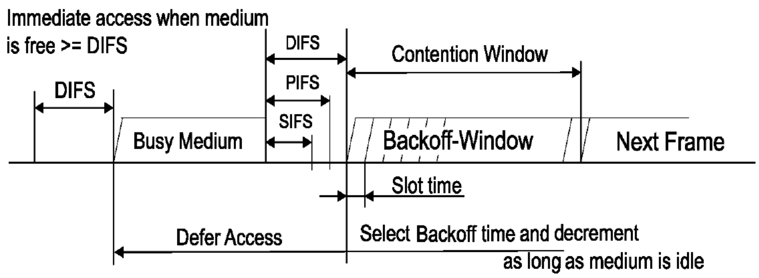


Figure 3.10: IEEE 802.11's Basic Access Method [48]

When the medium is determined to be idle by CCA for at least a DIFS period (or an EIFS when the previously received frame was found to be corrupt by means of the FCS) the station may access the medium, as illustrated in Figure 3.10. A backoff period is randomly chosen from the contention window $(CW_{min}, \ldots, CW_{max})$. If the backoff timer expires and the medium is sensed idle at that moment transmission may commence. If the medium is sensed to be busy another station had a shorter backoff period and the backoff will be suspended until the other station finishes transmission. The host will then resume backoff with the remaining backoff time. If a node's backoff timer reaches zero and it has still not won contention (e.g. has not been successful in claiming the medium and transmitting its message) a new backoff period will be started, up to a maximum of RetryLimit, by default 7 in the standard.

The backoff time is a multiple of aSlotTime, a value defined by the hardware and PHY as:

```
aSlotTime = aCCATime + aRxTxTurnaroundTime + aAirPropagationTime + aMACProcessingDelay
```

Characteristic	Value in 802.11b (DS)	Value in 802.11a	Value for 802.11p[1]
Frequency Band	2.4GHz	5GHz	5.9GHz
Bandwidth	20MHz	10MHz	10MHz
aSlotTime	20μs	13μs	13μs
aSIFSTime	10μs	32μs	32μs
aCWMin	31	15	15[2]
aCWMax	1023	1023	1023[2]

Table 3.3: PHY Characteristics for 802.11 a, b and p

Or in words: the time it takes for Clear Channel Assessment, then switching the radio to *transmit* and transmitting the frame, the time the frame is in the air and the time required for processing at the receiver. Such timing information can be found in the standard [48] Section 9.2.10.

In order to ensure rapid transmission of data spanning multiple frames a More Fragments bit is defined in the MPDU header, a station may then claim the medium almost immediately (after a Short Interframe Space (SIFS)) after the previous transmission has been acknowledged. With respect to the nature of communication required for the TrafficFilter—this nature being *messaging*—communication will be restricted to single frames. The mechanisms for handling fragmentation over multiple consecutive frames will not be discussed.

Acknowledgments

When a station correctly receives a message it sends back an acknowledgment to signal correct reception. If such an acknowledgment is not received by the sender within a certain period, it will retransmit the frame, assuming it collided.

When using the DCF's broadcast functionality it is not possible to use acknowledgment of reception (either at MAC-level or at a higher level), because then multiple stations would acknowledge successful reception of one transmission, causing a collision. As a result all transmissions are unacknowledged and a station can not get definite guarantee that the transmission has been received. The only way a station can learn its transmission is successfully received by another station is when a rebroadcast is overheard, which functions as an acknowledgment by inference.

The absence of acknowledgments is normally used to signal an error condition—either in the initial transmission or in the transmission of the Acknowledgment. In case no acknowledgment has been received a retransmission can be executed. With flooding in a VANET it is not even certain there is a vehicle within range because transmission is unsolicited, so the non-existence of the acknowledgment that can be inferred from the rebroadcast can also mean there is no station to reply, hence retransmissions will be a waste of resources.

No retransmissions will be needed, partly because it is not possible for a source to determine correct reception at the receiver but also partly because of the nature of the broadcast in this specific case; after a short while another message will carry new and up-to-date information. The absence of an error detection and retransmission mechanism makes broadcast less reliable than transmissions which are individually addressed.

Reasons for a low reliability of the wireless transmission include interference, collisions and

[1] At the time of writing, IEEE 802.11p still has Draft status. As such some characteristics are uncertain. These values are derived from 802.11a on which 802.11p is based and the Draft of 802.11p (2006) [49]

[2] IEEE 802.11p uses the Enhanced Distributed Channel Access (EDCA) Quality of Service (QoS) extension provided by IEEE 802.11e. The values given relate to the class with the lowest priority (AC0).

dynamic properties of the medium such as shadowing and multi-path propagation. In VANETs the dynamics of node mobility are added to this equation, as nodes are constantly moving in and out of each others' transmission range and the propagation environment continually changes.

Duplicate Detection

The IEEE 802.11 MAC is equipped with a means to detect duplicate frames. This event can arise when, by means of a retransmission, one station receives the same frame twice because the initial transmission collided with a transmission outside this node's range but within that of the source or when the acknowledgment was lost.

Duplicates are detected by means of cached address, sequence and fragment numbers and a Retry bit in the header. In the standard it is defined that non-QoS stations should reject any duplicate frames. In the broadcast environment of a VANET there will not be any retransmissions—not in the sense that the exact same frame is retransmitted to cope with packet loss. It might however be that the payloads of the frames are similar. Determining this will be up to higher layers.

3.4.2 802.11 in VANETS

As noted above, VANETS will primarily use 802.11's DCF. Traffic information and safety related communications will predominantly use the broadcast functionality because the information is intended to be received by all neighbours.

It is argued in some literature [126, 124] that 802.11 or parts (such as the RTS/CTS sequence) does not perform well in a multi-hop ad hoc setting. Nonetheless most research in the MANET / VANET community uses 802.11-based MAC and PHY. There are several reasons:

- 802.11 is a mature standard, this means it has been thoroughly verified and a lot of software (e.g. simulators) is available
- A lot of 802.11-based hardware is available off-the-shelf, resulting in both cheap and easy prototyping and cheap production, especially when the aim is to have a great number of nodes
- Some research uses an *altered* 802.11 MAC tuned especially for broadcast
- Many MANET/VANET communication schemes (including the one proposed in this thesis) only use 802.11's broadcast functionality because CSMA/CA functionality is required, and this is offered by the DCF.
- IEEE 1609 defines use of an enhanced 802.11 PHY and MAC in ITS applications, known as IEEE 802.11p.

IEEE 802.11p

In de U.S. the FCC (Federal Communications Commission[3]) has licensed a dedicated wireless spectrum of 75MHz for ITS-related communication. In Europe ETSI (European Telecommunications Standards Institute[4]) allocated spectrum for ITS applications in the 5.9 GHz band. The spectrum is divided into seven channels of ten megahertz each. One channel is a **control channel**, the other six are **service channels**. Although the name control channel might suggest otherwise, it is often interpreted as a dedicated safety-message channel. Non-safety messages

[3] http://www.fcc.gov
[4] http://www.etsi.org

may use the other channels. Applications on the service channels could be private applications or advertisements [53].

Spectrum

The U.S. Department of Transportation and the E.U. CAR 2 CAR Communication Consortium (C2C-CC) are converging to use IEEE 802.11p Wireless Access in Vehicular Environments (WAVE) and Dedicated Short Range Communications (DSRC). Although, initially, most U.S. ITS work was focused on the 915MHz band [133], in the fall of 1999 the FCC allocated 75MHz in the 5.9GHz band exclusively for ITS, aligning the frequency bands in the two continents. The main difference is in the way the channels are planned to be used, as in some countries regulations do not allow the use of certain frequency bands. As shown in Figure 3.11, the control channels coincide. It is the C2C-CC's aim to achieve spectrum harmonisation and further standardisation of IEEE 802.11p [18].

Frequency (GHz)	Use in Europe	Use in U.S.
5.925		
	Road Safety & Traffic Efficiency	Public safety, intersections
5.915		
	Road Safety & Traffic Efficiency	Public safety, Private
5.905		
	Critical Safety	Public safety, Private
5.895		
	Control, Critical Safety	Control
5.885		
	Road Safety & Traffic Efficiency	Public safety, Private
5.875		
	Non-safety	Public safety, Private
5.865		
	Non-safety	Public safety, Veh-Veh
5.855		

Figure 3.11: 802.11p PHY frequency allocation in Europe [18] and the U.S. [132]

There are several reasons why standardisation is an important issue. The automobile industry is operating on a global level, achieving great benefits (primarily cost reduction) through economies of scale. If the required electronics would be standardised it suffices to manufacture, fit and support one system in stead of several specifically tailored for target markets. Another reason is that vehicles, by their very nature, are not restricted to geographically defined governmental jurisdictions. As such they are apt to move from one regulatory area to another. In order for a system to operate in a multitude of regulatory domains (specifically with respect to radio spectrum and transmission power) a device would have to be able to switch from one mode to the other and support reception and transmission on multiple frequency bands. Norway, for instance, does not allow the use of the 5.855-5.925 GHz band [105]. Needless to say, a device able to switch between multiple frequency bands would be more expensive and complex as would a device standardised to one band.

Not only on the physical level do OBUs in vehicles need to be able to cooperate, this also pertains to the protocols in the MAC and higher levels. For instance one manufacture could use logic which is the reverse of that used by a second manufacturer: vehicle A says “I am braking” while vehicle B says “Slow down, vehicles behind me”. They both have the same aim (prevent head-tail collisions) but cannot inter-operate. It is important there is harmony throughout the protocol stack, because many future systems will rely on communication between vehicles.

Standardisation also speeds up market penetration. Some systems (such as detecting a vehicle behind a patch of trees) rely on both vehicles being able to communicate. The Congestion Assistant works at penetration rates of 10% and over. It would not be beneficial if, indeed there is a 10% share of V2V-enabled vehicles but they are comprised of non-interoperating systems A and B in a fifty-fifty distribution.

3.5 Modeling of Traffic

Part of this research requires nodes in a simulator to behave as vehicles on a highway. The focus of this mobility is longitudinal, the speed which a driver maintains under various circumstances. Several models are available in literature.

3.5.1 Traffic Models

These models try to capture the dynamics of human behaviour in traffic in a formal representation. They models can be macroscopic (i.e. they regard a road or road network as a large system with properties such as in- and outflow and density) and microscopic (i.e. every vehicle is modelled separately). There also exist methods for deriving macroscopic (fluid-dynamic) models from microscopic car-following models [63, 43].

Examples of macroscopic models include Gas-Kinetic-Based models [45, 83, 123] and fluid-dynamic models [42]. Macroscopic models are primarily used for optimising traffic flow [58] and are not limited to vehicular traffic but also pertain to pedestrian dynamics [44]. Macroscopic models excel at mirroring real traffic on a flow-level, but in order to simulate multi-hop behaviour a model is needed on the vehicle level. Another reason why a model on the vehicle level is needed, is the importance of deriving the speed of a specific vehicle which is at a certain location at a distinct point in time.

Microscopic traffic models model each vehicle and its behaviour separately. These models are generally more computationally complex because every vehicle is treated individually. Microscopic models can be cellular automata models [75] or so-called Car-following or Follow-the-leader models [102]. The Optimal Velocity Model [5] and Intelligent Driver Model [102] are two models which fall into the latter class. In this research the Intelligent Driver Model is used as presented by Treiber *et al.* in [102] for several reasons:

- The IDM behaves accident-free because it depends on relative velocity
- It shows self-organized characteristic traffic constants, hysteresis effects and complex states
- All model parameters have a reasonable interpretation and are empirically measurable
- The model can be easily calibrated to empirical data
- It allows for fast numerical simulation
- An equivalent macroscopic counterpart is known.

Intelligent Driver Model. - the IDM is a continuous car-following model which is essentially defined by an acceleration function [102]. This acceleration function $\frac{dv}{dt}$ for vehicle α is defined by the velocity v_α, the gap s_α and the velocity difference (the approaching rate) Δv_α to the vehicle in front:

$$\frac{dv}{dt} = a^{(\alpha)} \left[1 - \left(\frac{v_\alpha}{v_0^{(\alpha)}} \right)^\delta - \left(\frac{s^\star(v_\alpha, \Delta v_\alpha)}{s_\alpha} \right)^2 \right] \tag{3.2}$$

The parameters are summarised in Table 3.4. Expression (3.2) is a combination of two tendencies:

- Accelerate on a free road with $a_f(v_\alpha) = a^\alpha \left[1 - \left(\frac{v_\alpha}{v_0^{(\alpha)}} \right)^\delta \right]$ – a driver seeks to reach his desired speed v_0.

Parameter	Typical value
Desired velocity v_0	120 km/h
Current velocity v_α	$\cdots$
Safe time headway T	1.6 s
Maximum acceleration a	0.73 m/s^2
Desired deceleration b	1.67 m/s^2
Acceleration exponent δ	4
Jam distance s_0	2 m
Jam Distance s_1	0 m

Table 3.4: Parameters of the IDM [102]

- Decelerate when vehicle α comes too close to the one leading with $-b_{int}(s_\alpha, v_\alpha, \Delta v_\alpha) = -a^{(\alpha)} \left(\frac{s^\star}{s_\alpha}\right)^2$

The deceleration depends on the ratio between the “desired minimum gap” $s^\star$ and the actual gap s_α, where $s^\star$ is defined as follows:

$$s^\star (v, \Delta v) = s_0^{(\alpha)} + s_1^{(\alpha)} \sqrt[2]{\frac{v}{v_0^{(\alpha)}}} + T^\alpha v + \frac{v \Delta v}{2 \sqrt[2]{a^{(\alpha)} b^{(\alpha)}}} \tag{3.3}$$

As a result, the desired gap is dynamically varying depending on the velocity and the approaching rate.

□

The IDM is a continuous model, but can also be used in a discrete manner. A discrete event simulator takes timesteps and in every step every vehicle will evaluate its $\frac{dv}{dt}$ and alter its speed v_α accordingly.

A Java applet visually demonstrating the Intelligent Driver Model is available [101]. The applet functions in discrete time and after every step a simple check is executed:

```
if (V<0.0){V=0.0;}
```

This removes any negative speeds. This check is required when using the IDM in a discrete application, or the timesteps should be so small as to approximate a continuous-time model. Because of the timesteps a speed can easily overshoot the 0 into the negative domain. If timesteps were to approach 0 (to resemble a continuous system) this would not occur. In [65] Linesch and Perez present a non-linear traffic model given in Equations 3.4 and 3.5. The exact details are not important in this context, but notice that, in contrast to the IDM, there are a couple of Heaviside (step) functions defined as $Z(x) = H(x) * x$. They ensure that terms meant for deceleration do not decelerate below a speed of zero.

$$\dot{x}_n = v_n \tag{3.4}$$

$$\dot{v}_n = A \left(\frac{1 - v_n T + D}{x_{n+1} - x_n} \right) - \frac{Z^2 (v_n - v_{n+1})}{2 (x_{n+1} - x_n - D)} - kZ (v_n - v_{per}) \tag{3.5}$$

Applying the Heaviside step function to use the IDM in discrete time would look something like this:

$$v_t^{(\alpha)} = Z \left(v_{t-dt}^{(\alpha)} \right) \frac{dv}{dt} \tag{3.6}$$

Here the speed of vehicle α at time t depends on its former speed (one timestep dt ago) multiplied by the acceleration or deceleration $\frac{dv}{dt}$. By means of the Z speeds do not drop below zero.

Table 3.4 shows that s_1 is set to 0 in agreement with [102, 20] and the applet [101]. In [102] the reason for this is given, because a nonzero s_1 would be necessary for features requiring an inflection point in the equilibrium flow-density relation. This is required for certain types of multi-scale expansions, but left at 0 for this research.

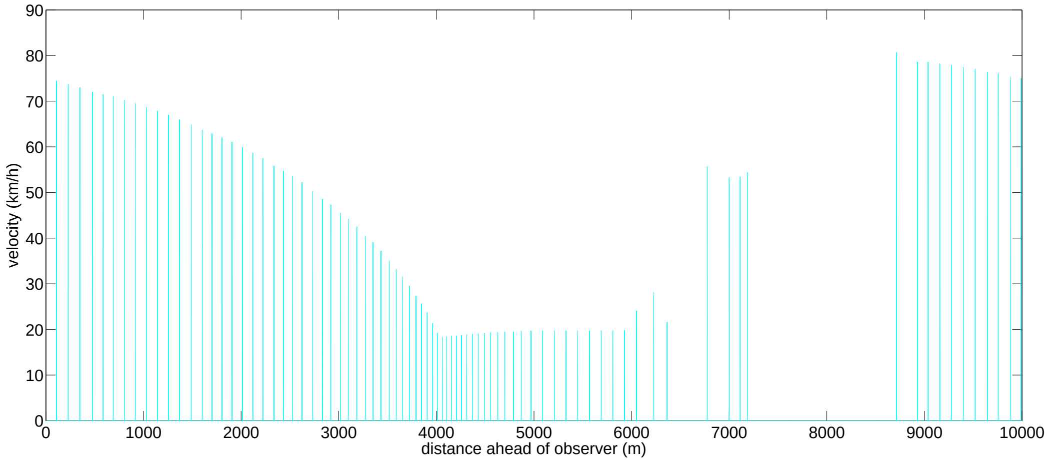


Figure 3.12: Road generated with the Intelligent Driver Model

The resulting road contains fluent density and speed transitions, as can be seen in Figure 3.12. A very important factor in the IDM-generated road is how long it evolves. The model starts with a road containing randomly placed vehicles (uniform distribution) and then apply the IDM to it for a number of timesteps. A jam is introduced on a certain interval by deliberately setting the maximum speed low on that interval. Vehicles will try to assume this speed, and queueing will occur.

The dynamic behaviour over time of the IDM-generated road is shown in Figure 3.13. Without congestion the vehicles show the typical 'stop-and-go' fluctuations often found in IDM-generated roads as shown in Figure 3.14. In this model the road is circular; when a vehicle leaves the end it is inserted at the beginning.

3.5.2 Discussion

The Intelligent Driver Model is used in a discrete event simulator (OMNeT++) to evaluate the system designed to provide an over-the-horizon view to the Congestion Assistant in a dynamic environment. Although the Intelligent Driver Model produces more realistic behaviour than a randomly generated set of speeds for vehicles on a road it still is an abstract representation of reality. By the very nature of a model it abstracts from many details found in real traffic. Furthermore, the IDM will be used without calibration—matching it to measured real data—and will be used as a qualitative reference.

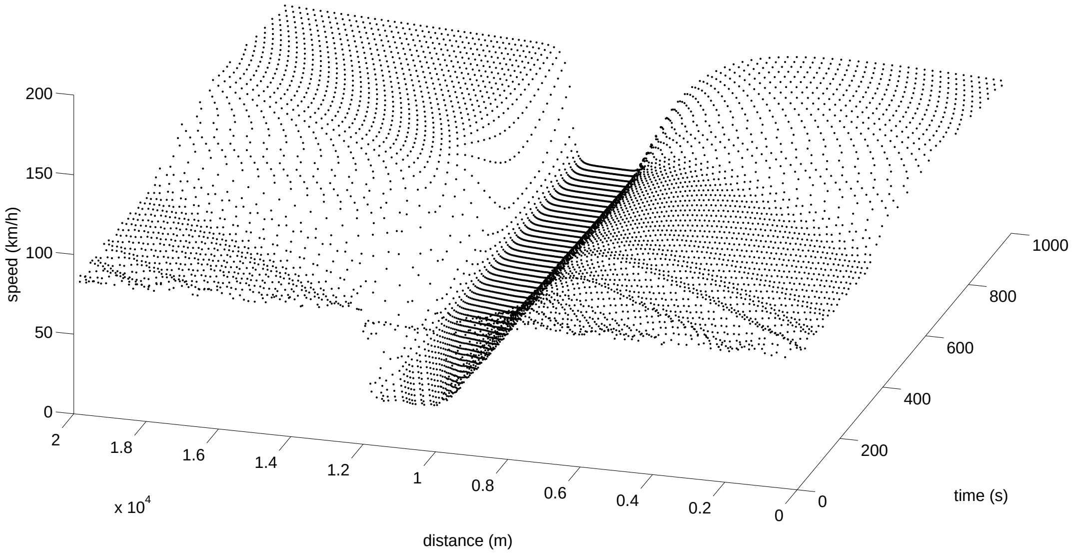


Figure 3.13: Road with congestion

3.6 Conclusion

This chapter considered the context of this research. A traffic jam has been defined as slow moving or stopped traffic over a length of road of at least two kilometers. Several means to detect traffic jams have been described such as loop detectors in the road, cameras alongside the road, traffic helicopters and pilot projects which measure traffic density and flow speed by means of observations of the mobile phones in an area. Means to notify drivers of traffic jams have been highlighted. Besides traditional radio news broadcasts hotlines, websites and RDS broadcasts have been defined.

It is observed that present traffic jam detection systems have a level of accuracy which is unsatisfactory for applications such as the Congestion Assistant. Likewise, present traffic jam notification systems are not accurate or timely enough, or not at a level of granularity demanded by the Congestion Assistant.

A survey of the accuracy of GPS-based positioning leads to the conclusion that modern-day GPS systems can generate position fixes which are accurate enough for the Congestion Assistant. The next section set out to find a platform which can be used to distribute the information as part of an overview of Intelligent Transportation Systems. This section is by no means complete but focusses on how the technology can be used in the context of this research. The U.S. ITS architecture and IEEE 1609 are reviewed. They provide an integrated platform aimed at accomodating any kind of intelligent behaviour in road traffic. Two specific platforms are highlighted: CarTel and Trafficopter. It is concluded that, in the context of this research, no usable platform exists.

Wireless Communication is very important in ITS. An overview of the IEEE 802.11 family is provided because many proposed systems rely on the mature Media Access Control and

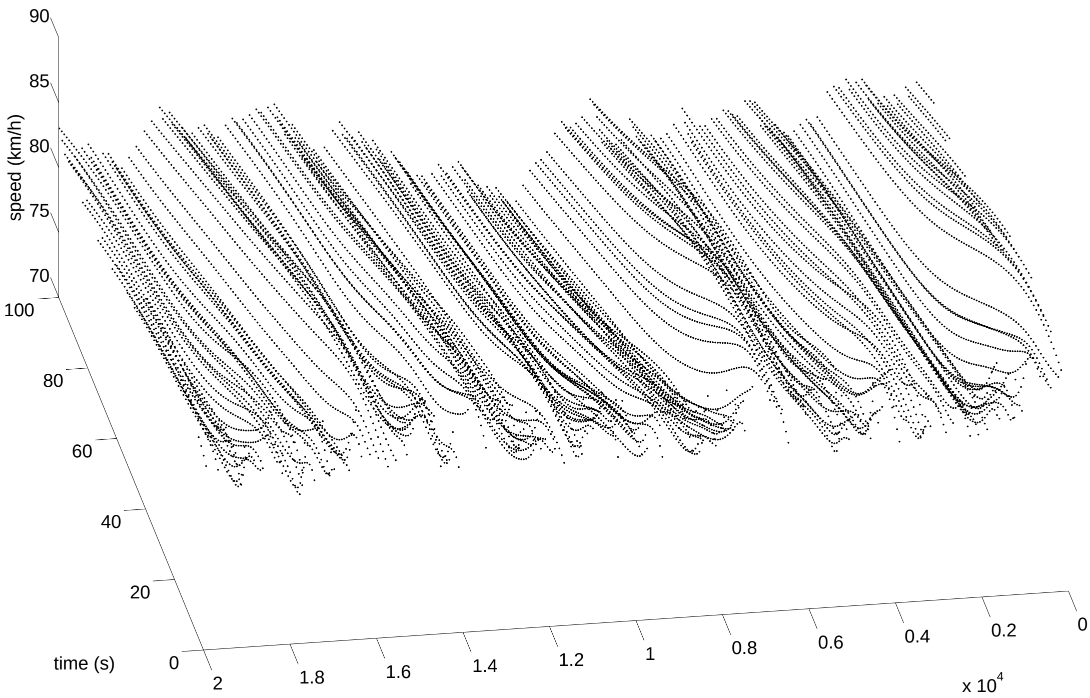


Figure 3.14: Road without congestion

availability of hardware. IEEE 802.11p is a derivative of the 802.11 family defined for use in a vehicular environment.

In the next chapter the findings on ITS, wireless communication and modeling of traffic will be used to design a system geared towards providing over-the-horizon awareness to the Congestion Assistant.

Chapter 4

The TrafficFilter

This chapter explains the idea of the TrafficFilter, a means to generate a TrafficMap in every On-board Unit (OBU) on the road. The TrafficFilter is a distributed system in which many vehicles work together to build an over-the-horizon view of the road ahead. This information is contained in a datastructure called the TrafficMap which is introduced in Section 4.1. This view can then function as input to ITS applications such as the Congestion Assistant, which was described in Chapter 2.

In order to build an over-the-horizon view we need information from vehicles that *are* currently over the horizon. They capture information—function as eyes, so to say. The information will be carried upstream through multi-hop V2V communication. In effect, the information travels against the flow of traffic. This can be on the same lane, or on the opposite lane.

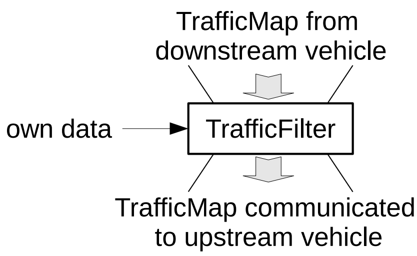


Figure 4.1: Information passes through the TrafficFilter to generate a TrafficMap. Multiple vehicles collaborate in a distributed fashion.

TrafficMap is introduced in Section 4.1. Then we will focus on several methods to distribute the information. The goal is to make sure every vehicle on the road has enough information to construct its own TrafficMap. In Section 4.3 a way to implement the TrafficFilter by means of a sampling approach is introduced.

The conceptual design proposed in this chapter results in a system design and protocol specification presented in Chapter 5.

4.1 The TrafficMap

Figure 4.2: Conceptual idea of an over-the-horizon-view

The Congestion Assistant requires an over-the-horizon view of the traffic on the road ahead. Complete knowledge of the road would without doubt be the most ideal situation because it contains every detail of the traffic situation. Since it is generally not possible to obtain a complete and accurate view—with respect to measurement-location relations and time—the information will be an abstraction of reality. In fact, a thorough abstraction consisting of only two measurements might even suffice, embodied in the following information (see also Section 2.2):

- Position and speed of the tail of the upcoming traffic jam
- Position and speed of the head of the upcoming traffic jam

The position and speed of vehicles is passed upstream by means of vehicle-to-vehicle communication. As shown in [91, 125] it is important these messages are small, short and leave enough bandwidth for other, more delay-critical applications [87]. Most Vehicle Safety Communication (VSC) systems require very low delays (in the order of 100ms [12]). Another important thing to notice is that, by the very nature of an over-the-horizon view, the captured information is passed on by multi-hop communication for great distances. In this sense, every byte transferred is a costly byte with respect to aggregate costs. We better make sure this byte is worth the effort!

Hence it is important to choose a good packet format and information encoding. The question now is, how do we represent the data in such a way that we preserve the level of detail we need while reducing the amount of data needed to represent this information?

For a proof of concept we will aim for a one-dimensional impression of the traffic on the road ahead. We consider a one-lane stretch of highway with traffic driving in one direction only. The cars are represented as abstract nodes with a speed and a position.

In a more realistic situation we need to cope with intersections, junctions and multiple lanes, resulting in a two-dimensional system. Vehicles are represented as tuples, composed of a means to denote a location plus the velocity (and possibly heading) of the traffic at that location at the time of measurement. The rationale behind this is that, when supplied with the position, speed and heading of vehicles in front a vehicle can form a representation of the road ahead, as sketched in Figure 4.3. In this figure every vehicle has knowledge of its predecessors. This knowledge is a mapping of speeds to a location on the road. We will refer to this collection of tuples as the TrafficMap.

TrafficMap. – a collection of tuples representing speed and position information along the current route of travel up to the Virtual Horizon. □

Virtual Horizon. – defines up to which distance awareness is extended. When using multi-hop V2V communication this implies information will travel up to the distance defined by the Virtual Horizon. □

For example, referring to Figure 4.3, if the Virtual Horizon is defined to be h and $[v+w+x] \leq h < [v+w+x+y+z]$ then entry 5 ("$4+z$98") would be removed from the TrafficMap in order to reduce the amount of data. A Virtual Horizon is required to ensure data does not travel beyond the distance where it looses its value (e.g. uncertainty incurred due to elapsed time since capturing increases beyond acceptable levels) and to prevent less-important data from flooding the network.

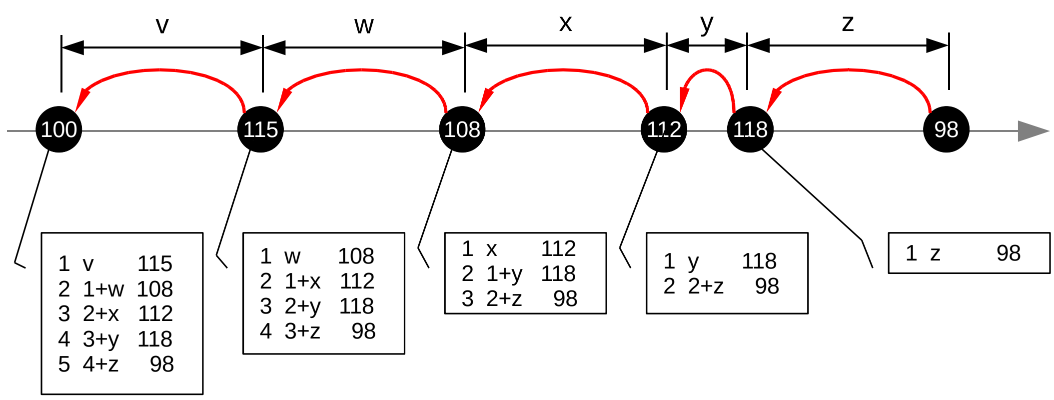


Figure 4.3: Nodes on a line with a value denoting speed in km/h. The distances between vehicles are expressed by v through z. For example, the distance between the left-most vehicle and the fourth is $v + w + x$. The values in the boxes denote the TrafficMap. For instance, the third entry in the TrafficMap present in the left-most node reads "$2+x$ 112", which implies a recursive relation because 2 is defined as $1+w$. As a result the third entry reads "$v + w + x$ 112" when expanded.

A road with vehicles can be seen as a set of moving nodes on a line. For simplicity, assume every node has the properties *position* and *speed*. We abstract from heading information and map all vehicles to a one-dimensional line where a vehicle can be plotted as a point on a position / speed plane. Such a plot can be created with a traffic model as described in Section 3.5 on page 40. A vehicle moving on the road will try to construct a similar plot about the road ahead based on information it receives through vehicle-to-vehicle communication. It is reasoned in [53] that the information produced by vehicles should be indicative; they cannot dictate how another vehicle must process and interpret a message. In line with this reasoning we assume it is the task of a vehicle to publish 'information', and not so much directives or commands for other vehicles. Based on interpretation of this information the other vehicles can decide to act.

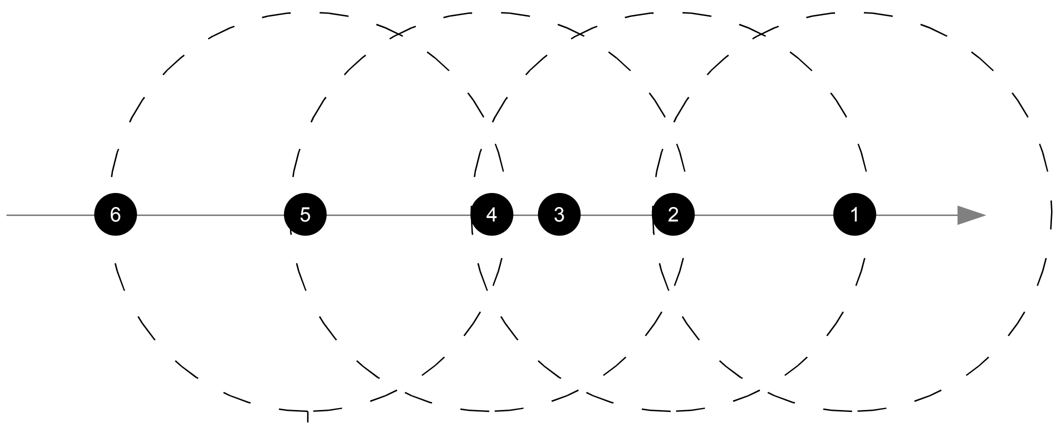


Figure 4.4: From the perspective of node 6 (the observer) node 1 is a source (assuming 1 added an entry in the TrafficMap) and nodes 2,4 and 5 are relay nodes. Node 3 is a latent node.

In the remainder of this work we will refer to source, relay and observer vehicles. A source vehicle is a mobile node that broadcasts its own information (e.g. add an entry to the TrafficMap). An observer node is a (potentially) mobile node that receives this information. Relay nodes do not add information but merely pass it on. A latent node does not publish or relay information, but will receive it. The information functions as a means to observe traffic, hence the receiver of such information is called an observer. The moment an observer passes the information on it becomes a relay node itself if no new information is added, or a source node when it injects its own information.

4.1.1 TrafficMap Contents

Appendix A provides an analysis of three ways to express the information contained in the TrafficMap. The kind of encoding used influences the accuracy and size of the information contained in the TrafficMap which can, in turn, affect the performance of the TrafficFilter system. The remainder of this report uses the Absolute Positioning scheme described in Appendix A.2.1. As such the information contained in the TrafficMap contains the following fields:

Parameter	**Description**	**Size**
Speed	denoted in km/h	1 byte (0-255)
Position	Latitude, Longitude	8 bytes ($DDD°mm.mmm'$,$DDD°mm.mmm'$)
Heading	direction of travel	1 byte (255 quanta of 1.4° each)

Table 4.1: The information contained in the TrafficMap

4.2 Information Distribution

In the previous section the TrafficMap concept was introduced. This TrafficMap is the information construct which functions as the over-the-horizon view and comprises position, speed and heading information. If an OBU were to receive information from which it can build a TrafficMap it can draw various conclusions and systems such as the Active Pedal and Warning & Information can be fed with data.

The challenge now, is to distribute this information to all equipped nodes (i.e. all vehicles with an OBU) on the road. Several methods come to mind:

- Radio Broadcast (RDS)
- Cellular technology (UMTS, GPRS)
- Vehicle-to-Infrastructure communication (V2I)
- Vehicle-to-Vehicle communication (V2V)

Radio broadcast and cellular technology could be combined as proposed in Figure 4.5, where cellular technology (2) provides information to a Traffic Operations Center (TOC) supplementing the measurements from loopdetectors (1). The TOC broadcasts on an RDS channel. Cellular technology suffers from a scalability problem: if all vehicles were to use the cellular infrastructure it would no longer be possible for people in vehicles to make phonecalls. Furthermore, the granularity for transmitting RDS signals is not very fine: an RDS transmission covers a great area. Transmitting specific information for all vehicles in that area may not be possible. Another problem with this approach is the need for a central TOC: here a vast amount of data will have to be managed, possibly incurring a great delay.

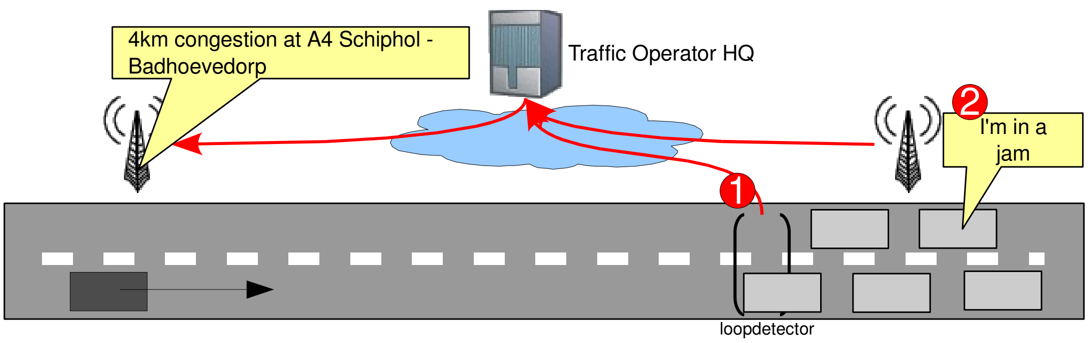


Figure 4.5: Using a centralised approach to distribute information

Looking at where the information is generated and consumed, which is on the road, it seems logical not to transfer the data over too great a distance. This is where V2I and V2V come in. Especially V2V seems interesting because of its ad hoc nature. Later on we will point out that V2I may also play a role to overcome gaps in the network (stretches of road with a low node density). Furthermore, it is reasoned that V2I stations may "sniff" TrafficMaps which could be sent to a TOC to gain insight in the current state of traffic with greater resolution than loop detectors.

Based on vehicle-to-vehicle communication, several approaches can be used:

Simple Flooding Every vehicle broadcasts its own position, speed and heading. This is then flooded upstream using a very simple flooding approach: every node rebroadcasts.

Intelligent Source When a vehicle notices it is in the head or tail of a traffic jam it explicitly floods its own position, speed and heading upstream.

Summarisation Information is flooded upstream. Some more processing takes place along the way to reduce overhead.

From the information received the OBU can construct an over-the-horizon view of the traffic ahead. The TrafficMap designed for the Congestion Assistant but the information contained may also be of use for many other applications such as Cooperative Collision Warning [29, 109], Cooperative Driving at Intersections [64], Cooperative Adaptive Cruise Control (CACC) and many other Cooperative Active Safety (CASS) and efficiency applications [108].

4.2.1 Simple Flooding

Broadcasting messages to all nodes is an important activity in computer networks [16]. A very simple approach is when a node, upon receiving a message, simply forwards it or broadcasts its own (position, speed, heading) information based on a timer. From the information received the OBU constructs and maintains a map of the surroundings, tracking vehicles in front (and possibly close behind). When a message is received it is rebroadcast in order to cover the entire network.

Because on a highway the *entire network* can be quite large and because the information ages and becomes stale there must be a limit to the distance or number of hops a message may travel. Furthermore, as every message travels further the aggregate network load increases. A *time-to-live* or maximum number of hops or kilometers can be used to limit the propagation. This limit has been defined as the *virtual horizon* in Section 4.1. Because the information is only needed upstream a node will decide to rebroadcast based on its own location and that of the source. If the source is upstream the message will be discarded. The result is a flood that propagates in one direction.

A direct benefit of this method is that the OBU can track *all* vehicles, because (assuming no packets are dropped[1]) the system is lossless with respect to the information transferred. The resulting TrafficMap can be used to deduct the information needed for the Congestion Assistant.

Simple Flooding, however, is a very inefficient approach. Consider we have a hundred vehicles per kilometer. If we aim to achieve a virtual horizon of 10km and every vehicle were to flood its own information, this would amount to 1000 floods. Although the packets will be small (containing only information on one vehicle) they still require resources with respect to medium utilisation and contention delay.

A simple flood with only one source is known to result in a phenomenon known as a Broadcast Storm [103], resulting in redundancy, contention and collisions. Broadcast Storms have three causes:

- Because radio propagation is omni-directional and one physical location may be covered by multiple hosts many broadcasts will be redundant. One message will be repeated by all hosts.
- The hosts within transmission range of one another are likely to be close to each other; heavy contention for the medium is the result.
- All these nodes are synchronised by one incoming message. When a node rebroadcasts the probability of collision is very high because many other nodes may be rebroadcasting the same message at the same time.

[1] Note that this assumption is unrealistically optimistic

The result is a flood of collisions that propagates through the network. If only one source can cause a Broadcast Storm, it will be clear that the problem will surely exacerbate if every host in the network were to flood its own information. This effectively dismisses Simple Flooding as a technique to carry an over-the-horizon view.

4.2.2 Intelligent Source

In this scheme a vehicle *senses* that it is in the head or the tail of a traffic jam. This could be deduced from front and rear proximity sensors, own speed and the speed over the past few minutes. The vehicle might also correlate its state to the type of road it is on. The latter information can be gathered from a navigation system which is instrumented with maximum speeds and descriptions of roads. An OBU might also make use of *beacon messages*, which are short status messages broadcast periodically as proposed by many systems [87, 15, 4, 53, 29, 73]. These beacon messages enable local awareness and can be used to, for instance, detect vehicles in blind spots. Upon noticing the transition from free-flowing to jammed the vehicle will broadcast a "tail of jam" notification which will travel upstream. Any vehicle receiving this notification will take notice of it—execute actions such as informing the driver or updating internal information—and rebroadcasts. The same goes for the "head of jam" notifications. It will be clear that here a lot of responsibility lies with the vehicles, because a congestion is explicitly notified to all other vehicles.

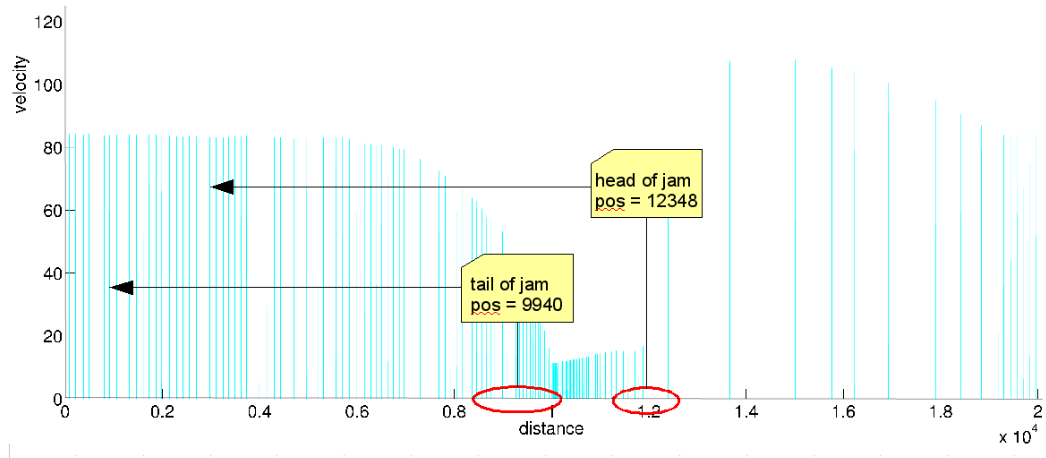


Figure 4.6: A section of road with a jam. Congestion Notifications are passed upstream

The head and tail notifications are passed upstream, and every vehicle entering the head or tail section of the jam (e.g. an *area* denoted by the notifications) evaluates if its own position differs much, based on a defined error margin, from the position passed through the external information flow. A vehicle can remove, invalidate or supersede the old message and inject its own information into the network. This way we can ensure up-to-date information because every equipped vehicle entering the tail and head zone re-evaluates the validity of this information before (re)broadcasting.

The messages used will have to be compact and related to a geographically identifiable location on the road. They will function as a warning beacon in the area well before the actual congestion starts. After generation the messages are passed upstream to fullfill their duty—inform approaching vehicles. As a head message traverses the congestion it might pick up relevant information such as the average speed in the jam. Messages must be identifiable in order to supersede or invalidate active messages planted by others. Multiple notifications must be able to coexist. When a jam is just beginning several vehicles can separately plant a notification (possibly close to each other). Such conflicts would need to be resolved.

This method is quite similar to Abiding Geocast, a method described by Yu and Heijenk in [128]. Here notification of a safety-related event is disseminated to all vehicles moving towards the location of the event. A safety line and effect line have been defined, notifications are passed on in the area between both lines and the goal is that every vehicle has received the notification before crossing the safety line.

Because we want to provide a full overview of the road ahead (as opposed to only report the head and tail of a jam) the Intelligent Source approach does not seem a good option to build an over-the-horizon view. Furthermore, a tail of jam notification indicates the position of the tail of a jam, but the decision that it actually is there has already been made. As such it is more like a directive to anticipate than information from which this conclusion can be drawn. It is reasoned that the identification of the tail or head of a jam is something which is better derived from an over-the-horizon view than from the partial view the vehicles presently in the jam have.

4.2.3 Summarisation

From the review of traffic models in Section 3.5 we learn that a vehicle is not completely autonomous, its behaviour is largely dependent on factors such as speed limits and other traffic. As a result, many vehicles on a road will exhibit roughly the same speed as those around them, especially in congested situations. This means that there can be a lot of redundancy in a TrafficMap representation if we were to represent every vehicle, because several vehicles in the same area will have roughly the same speed value.

Summarise with Integer Sequences

Because of the relation between the speeds of consecutive vehicles we can average the speed over intervals and pass a value denoting the interval and an average speed for that interval. The AT&T Online Encyclopedia of Integer Sequences (OEIS) [95] contains 136,959 sequences[2]. From these we choose two because of their different properties.

The idea behind a sequence-based over-the-horizon view is that the sequence is used to construct bins into which speed measurements are averaged for the interval covered by the bins. When the structure is handed to the next vehicle it performs a mapping adjust the bins to its own position.

A000045 - Fibonacci numbers are a sequence of positive integers named after Leonardo Fibonacci, an 11^{th} century Italian mathematician. He did not discover the sequence [94] but first wrote about it in European literature in his work *Liber Abaci* in 1202 [92]. The first 24 numbers of the sequence are:

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657$$

The recurrent relation is as follows:

$$F(n) = \begin{cases} 0 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ F(n-1) + F(n-2) & \text{if } n > 1. \end{cases} \tag{4.1}$$

Fibonacci numbers [100] approximate the so-called *golden spiral* or *golden ratio*, a behaviour often found in nature. It was a hunch that, perhaps, the Fibonacci sequence might be useful to

[2] As of February 2008

Number $F(n)$	End-position	Size
5	3	3
6	5	2
7	8	3
8	13	5
9	21	8
10	34	13
11	55	21
12	89	34
13	144	55
14	233	89
15	377	144
16	610	233
17	987	377
18	1597	610
19	2584	987
20	4181	1597
21	6765	2584
22	10946	4181
23	17711	6765
24	28657	10946
25	46368	17711

Table 4.2: Fibonacci bins

average speeds in the over-the-horizon view. If we were to use this sequence to map positions to intervals which increase in size as the distance from the observer increases and average over these intervals this would give a very accurate approximation on intervals close-by (because the averaging is only executed over a small number of values) and a less accurate approximation as distance increases.

If we take the first 25 Fibonacci numbers we obtain the bins as in Table 4.2. Note that we can skip the first few bins since their size is small and within the first 3 meters there generally is no vehicle. Also note that the size of bin n equals $F(n-2)$ because of the recurrent relation expressed in Equation (4.1). This is another interesting property which gives rise to the idea of *shifting* as the map is passed upstream. An example of a mapping of vehicles on a road to a Fibonacci TrafficMap is given in Figure 4.8(a) on page 55.

When shifting, to make the new n^{th} bin we take the old $n-1^{th}$ and $n-2^{th}$ bins, average them and store them in bin n. Of course this only holds in the ideal situation. An interval that was of length 3 now becomes of length 5, and so forth. To maintain scale consistency we need to merge bins with weighted splitting of the average values. The example shown in Figure 4.7 uses a speed in meters per second, but might as well be in kilometers per hour. Note that the results are rounded to the nearest integer for simplicity.

What happens in Figure 4.7 is the following. The set of 7 bins containing information is mapped to a similar structure with a shift of 10 to simulate a node handing over its information to the next node. This node maps th ereceived information to its own position by means of the shifting.

The value '12' is obtained by merging parts of the 14, 12 and 4 respectively: $12 \sim 12,008 = \frac{(\frac{3}{5}\times 14+12+\frac{2}{13}\times 4)}{1,75}$. Because of the shift of 10 the 4^{th} bin of size 5 containing 14 overlaps 3 with the

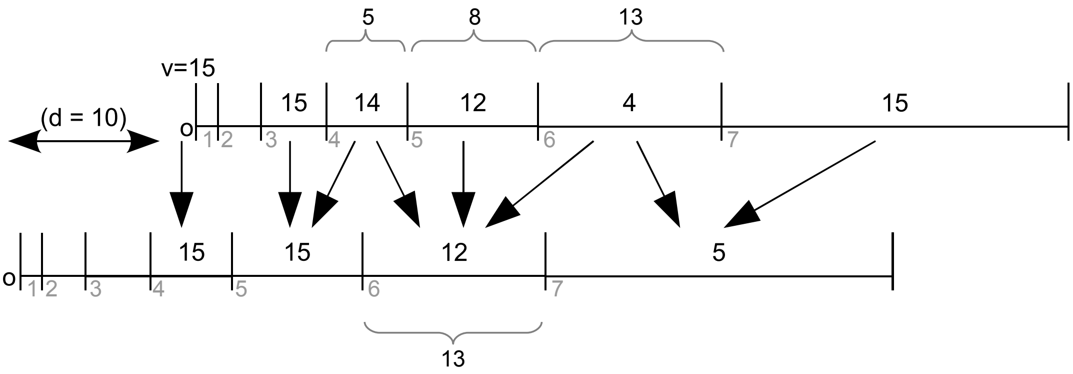


Figure 4.7: Shifting of a Fibonacci sequence-based TrafficMap

6^{th} bin of size 13 in the new structure. The bin containing 12 is transferred fully to the 6^{th} bin while the 6^{th} bin of size 13 in the old structure containing 4 overlaps only for $\frac{2}{13}$ with the 6^{th} bin in the new set.

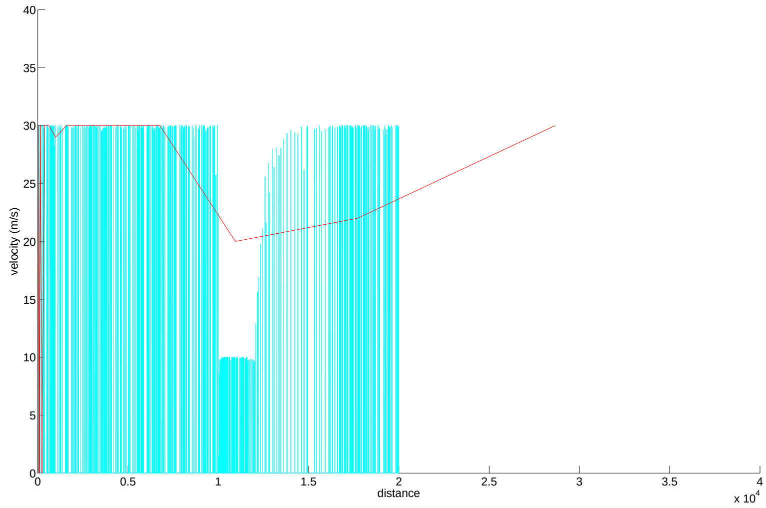
In the above example we have compressed 5 bins to 3 plus added a new value containing the speed of the last observer, because of a shift of 10. The graphical output of a recursive run of the Fibonacci TrafficFilter is presented in Figure 4.8(b). Note that the recursion is only one hop—and already the original information is completely gone. It is concluded that the shifting introduces an unacceptable distortion because the intervals are of different sizes, hence some skewing occurs.

A001477 - The nonnegative integers have the property of being equally spaced. As observed earlier, the Fibonacci sequence-based TrafficMap suffers the problem of skewing. This will not occur in a structure with bins of equal size.

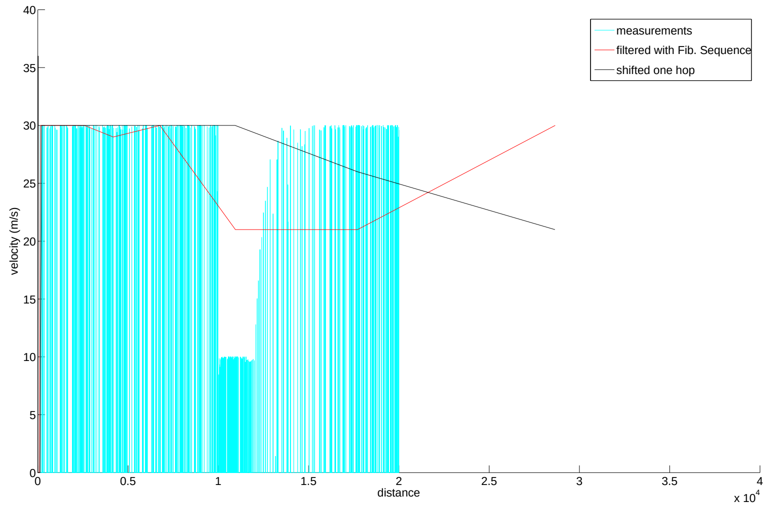
If we are to use $0, n, 2n, 3n, 4n, \ldots$ where n is a certain interval, we obtain a *raster* to which we can map positions on the road. A simplified example is presented in Figure 4.9 for $n = 5$.

For the first n meters ahead the *average* speed is 14. On the next section this is 13, and so on. The new value for $[n, 2n]$ is calculated as: $\frac{3}{5}14 + \frac{2}{5}13 = 13.6$. A shift of 3 is shown in Figure 4.9. In Figure 4.10 a graphical representation of calculations is presented. The figure depicts 5 shifts of 3,4,5,3 and 2 respectively, representing five hops. What becomes clear from the calculations—but more so from the graphical representation—is that the 'jam' initially located between 10 and 20 is spread out, and can no longer be detected accurately. One can reason there is a jam and that it is located around the center point of the dent in the graph, but length and speed can no longer be derived.

Evaluation of sequence-based summarisation We can say that a Fibonacci TrafficFilter, as well as any other Integer Sequence-based mapping, will not work for this application. Both work relatively well at mapping details of the original input to the bins (depending on the sequence used and—of course—the size of the bins) but the shifting cannot be performed without loss of precision due to the smear of the values caused by sloppy averaging and the skewing of intervals (when intervals are not of homogenous size). As a result there will be overflow into neighbouring bins. Choosing the bin size very small will approximate something not unlike a Riemann sum. This, in turn, gives rise to the idea of *sampling*.



(a) Fibonacci representation



(b) A one-hop shift

Figure 4.8: A Fibonacci-based TrafficFilter

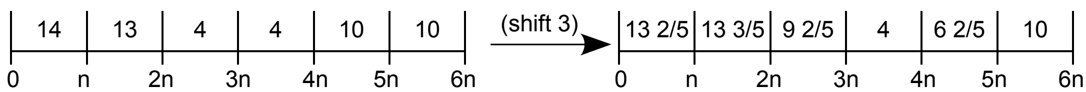


Figure 4.9: Shifting of a Nonnegative Integer sequence-based TrafficMap

Summarise with Sampling

If we look at the speeds of vehicles on a road and the lines connecting them as a continuous signal (e.g. a space / velocity plot with a curve fitted) we might be able to use the Nyquist-Shannon Sampling Theorem [90] to convert the signal to a numeric sequence or Pulse Code Modulated (PCM) system. This implies, however, that—by definition of the Nyquist rate of critical frequency—the sample rate must be twice that of the highest frequency we want to capture. The question that now arises is; can we find such a frequency in highway traffic? This would preserve the speed "signal" present on highways and sample it to a collection of numbers. When the sample rate is high enough an almost exact copy (lossless transformation) can be

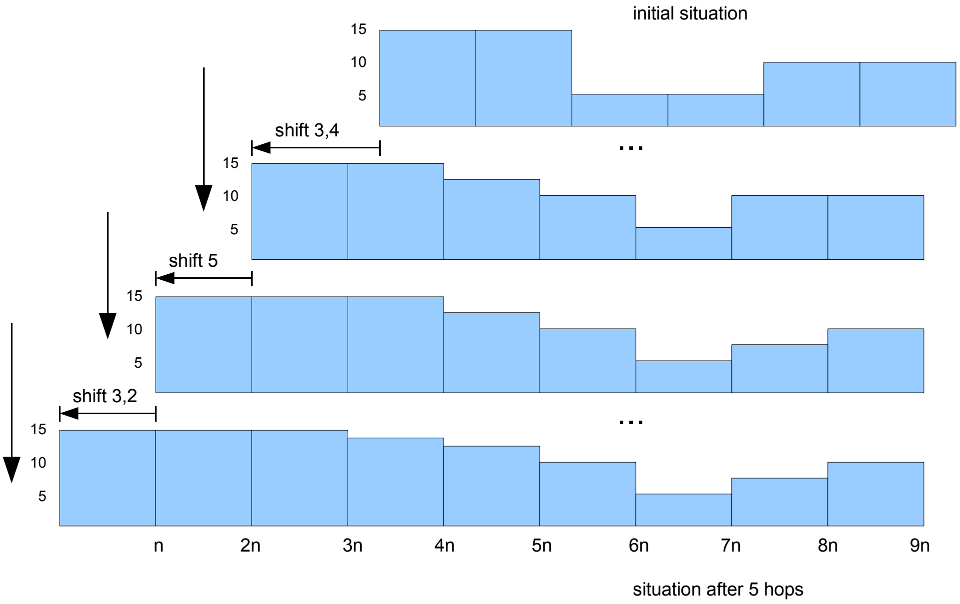


Figure 4.10: Graphical representation of Nonnegative Integer sequence TrafficMap. The shifts of 3,4 and 3,2 are merged. The values on the y-axis denote a speed in m/s.

obtained. This will, however, result in a lot of samples and hence a large amount of data. Unless we find a way to convert this large set of samples to a reduced set. It might be an idea to pass the resulting curves through an MPEG-1 Layer 3 encoder or something alike. This would result in a curve matching that of the speeds of vehicles on a road. We will not consider this approach in this research, but will look into a different, and slightly simpler, approach using sampling which we will call threshold-based sampling.

Threshold-based Sampling Integer Sequence-based approaches as described in Section 4.2.3 still result in maintaining information, even if the first several kilometers are without any significant congestion (e.g. speeds are close to a free-flowing or maximum speed). A sampling approach such as described above will do the same. This problem can be overcome by using a smart way to choose which sample to include in the TrafficMap. We will next have a look at a method which addresses this problem, based on methods used in statistical research: a sample is obtained from a population. The (speed,position) tuples of al vehicles on the road are the population. We want to—as is usually desirable in statistics—find a sample that is representative to the population. We want to capture the smallest set of samples that still forms a reasonable representation. We can now formally define the context for the TrafficFilter for a one-dimensional road:

Definition: Let:

X = { all (position,speed) tuples for unique position }

S = { the sampled tuples }

$\tilde{v}_i(\delta, v) \in X$, measurement i

$\overline{v}_i(\delta, v) \in S$, sample i with position δ and speed v.

$$\forall_{i,j} | i > j \Rightarrow \tilde{\delta}_i > \tilde{\delta}_j, \text{ vehicles cannot pass each other or be at the same position.}$$

$$0 < \delta < \text{ road length}$$

$$0 \leq v \leq \text{ maximum speed}$$

□

Where S satisfies the demand that it approximates the minimal representative set of X. Because of the distributed nature of the system and because information flows upstream (w.r.t. the trafficflow) we use a method quite similar to Run-Length Encoding (RLE) or Sample-and-hold with a variable hold time. We will illustrate the idea with the following example. Imagine a one-dimensional road with vehicles labeled $[A, T]$ in ascending order.

1. node A generates a sample (location / velocity tuple), and broadcasts this to neighbours within transmission range
2. node B receives the sample and evaluates this. It injects its own sample at the front and transmits.
3. node C receives and evaluates. Samples A,B and C are alike so B is removed and C injected.
4. ...
5. node K receives and sees samples A,J. Apparently from A to J the velocities are somewhat the same. But node K's velocity differs a lot, so it injects its sample *without* removing J.
6. node P receives A,J,K,O, and deducts its own sample differs from O's and injects its sample without removing O.
7. ...
8. node T receives the map, and evaluates.

The resulting TrafficMap in node T looks like this:

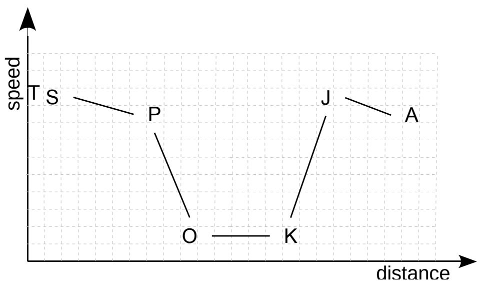


Figure 4.11: The TrafficMap in node T

The TrafficMap is constructed as it is handed from node to node. A node adds its own measurement value if it deviates enough from the previous sample. A sample is bound to a certain location, denoting the measured flow speed of traffic at that position. The sample and hold nature is reflected in the fact that, as in the example, a sample (say A) is valid until a new sample is triggered. It is obvious that in the above TrafficMap we might as well remove J and O without loosing too much information.

A run of this system over the IDM generated road is shown in Figure 4.12. As can be seen the road is represented in the values in the TrafficMap, plotted here as black bars. A plot with the IDM-generated road shows there are more gradual transitions in road traffic speed, as a result such a slope triggers more samples, depending on the way the thresholds are defined.

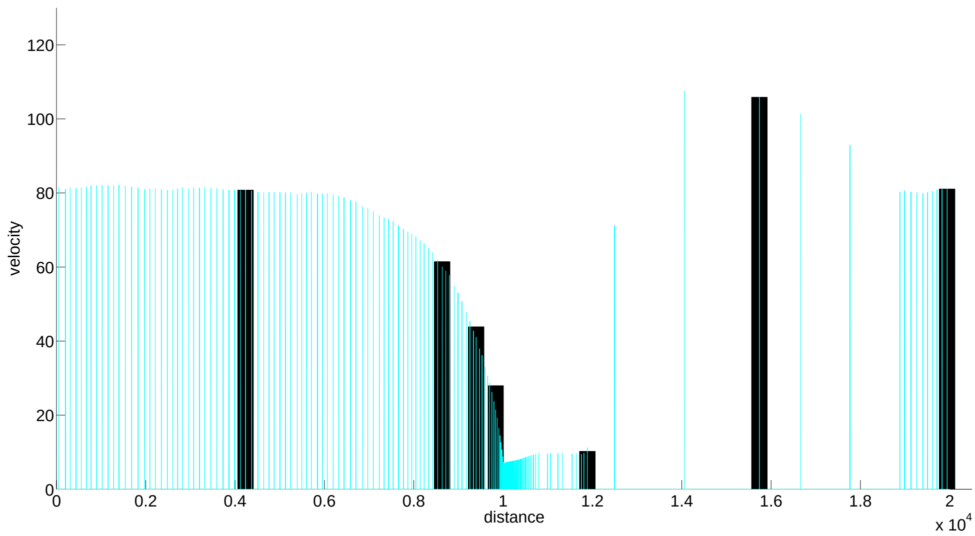


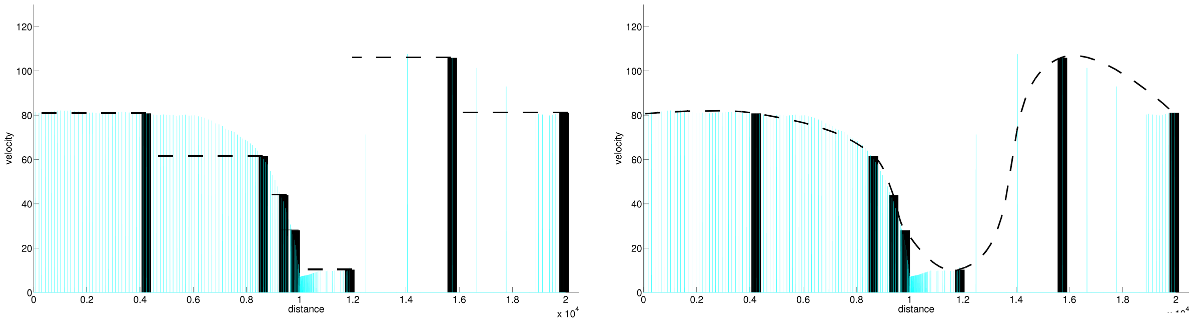
Figure 4.12: TrafficMap created with Threshold Sampling approach

4.2.4 Conclusion

Simple Flooding of information does not seem to be a viable approach, though it might provide a good reference when evaluating performance. Intelligent Source seems very appealing because it consists of only two messages being broadcast by a limited set of sources, namely the vehicles in the tail or head section of a jam. This provides very basic knowledge of the road ahead, but might also be very limited, as a vehicle gains only knowledge of two locations on the road. What happens in between is not known, and the brake curve of a traffic jam might differ from jam to jam so the Active Pedal has little indication of when to engage. Although this is an interesting concept it is expected to involve complicated decision-making schemes and a lot of guessing since a vehicle requires full local knowledge. Furthermore, a lot of responsibility rests with the vehicles; what if a malicious vehicle publishes false head-of-jam or tail-of-jam messages?

Summarisation by means of integer sequences was proven to be both ineffective at capturing details and suffer from inability to deal with the shifting, required if we are to use a relative positioning scheme anchored in the observer.

Summarisation by means of hard position-bound samples on the other hand seems the most promising approach. It functions by capturing sudden deviations based on pre-defined or dynamically set thresholds. As a result the details of the road traffic can be captured with good accuracy. Because a sample is bound to a location both relative and absolute positioning systems can be used. This scheme is strictly indicative as argued to be desirable by Jiang *et al.* in [53]. It can provide the over-the-horizon view that is required and every vehicle is free to draw its own conclusions as illustrated in Figure 4.13.



(a) Horizontal lines resembling a step-function (b) Smoothed Curve

Figure 4.13: Different ways of interpreting the TrafficMap

4.3 The Threshold-based Sampling TrafficFilter

As concluded in the previous section Threshold-based Sampling is an interesting method to provide an over-the-horizon view to intelligent vehicles. In this approach a vehicle decides whether or not to add a sample to the TrafficMap based on a threshold, denoted as ε in Equation (4.2).

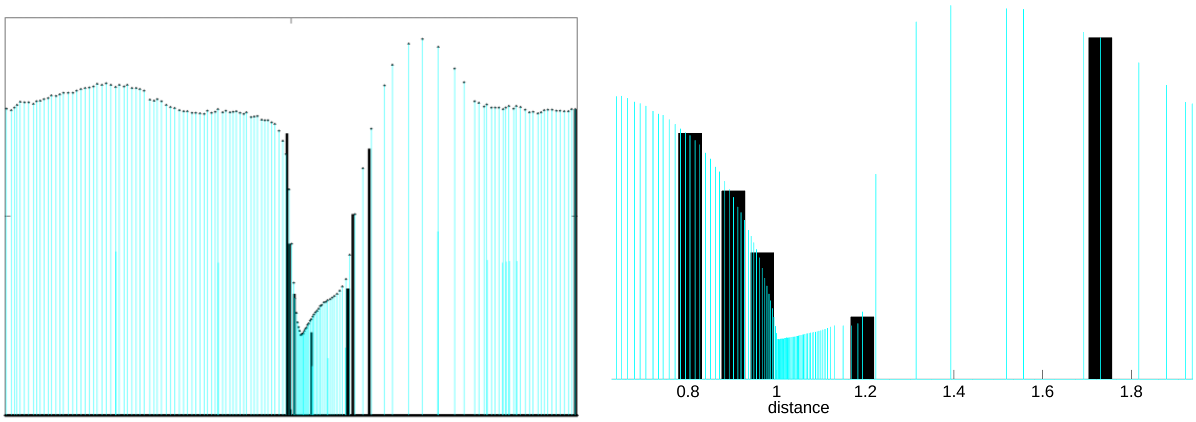
$$V_{new} = V_{own} \text{ iff } |V_{previous} - V_{own}| > \varepsilon \tag{4.2}$$

This threshold is based on the difference between the present (*own*) speed and the last-recorded speed in the TrafficMap (*previous*). It goes without saying that choosing a good threshold is key to obtaining a good representation of the road; if the threshold is too large we get only a few samples and might miss important details. If the threshold is too small the TrafficMap might grow explosively and contain a lot of redundancy. We can come up with several possibilities, which we will list now.

Static threshold With a static threshold, defined as $\varepsilon = x$ a new sample is triggered whenever the vehicle's own speed deviates from the last sample with more than x. The relation is very simple, if we choose a small ε we get a lot of samples, resulting in a lot of redundancy. A large value for ε may result in a grave loss of precision when, for example, a drop of 29 km/h occurs and the threshold is set at 30 km/h.

Relative threshold A sample is triggered whenever a deviation of more than a certain percentage of the old value is detected. This results in a lot of samples at lower speeds (because a threshold of 20% is easily triggered) and less detail in higher speeds. This seems desirable, because it is especially the dynamics of the slow-moving traffic we are interested in. As can be seen in figure 4.14(a) the curve of the jam is clearly visible from the samples (the black bars). However, the dramatic increase in speed after the congestion has not been captured, because it did not rise the required amount.

Edge-accentuation So far the threshold was based on a difference from the last sample. This difference can be positive (accelerating) or negative (braking). It is reasoned that the Congestion Assistant primarily requires knowledge of the braking slope of the TrafficMap, so the *sensitivity* to braking slopes is increased by dynamically decreasing ε when $V_{own} > V_{previous}$. The effects of this are displayed in figure 4.14(b).



(a) A relative threshold is more detailed for lower speeds (b) Edge-accentuation: different sensitivity per slope

Figure 4.14: Different threshold schemes

A speed-difference based threshold

Because none of the three methods discussed above seems satisfactory by itself, the function for ε will include a combination of the properties mentioned previously. ε will be a sensitivity function that operates on the own and the previous speed. It is reasoned that this provides enough information to decide whether or not to generate a new sample, although it might be expanded by also including prior samples in the decision making.

Projecting V_{own} and $V_{previous}$ to a two-dimensional plane we can generate two areas as shown in figure 4.15. The thresholds are shown by means of the two slopes left and right to the equilibrium-line. This simple model is tuned by means of the *offset* and *slope* for each area. To decide whether or not a node should add a new sample to the TrafficMap it is checked whether the actual situation falls within the defined areas. The IDM-generated road exhibits small consecutive deviations and hence most $V_{own}, V_{previous}$ tuples will be close to the equilibrium line, it is the deviations from this line we are interested in.

By increasing the areas (e.g. drawing them closer to the equilibrium) the sensitivity is increased and more samples are captured; the threshold is reduced. The two areas are called “Accelerating” and “Braking” edge because a large own speed and a low previous speed correspond to the tail of a jam (recall the TrafficMap is passed against the flow of traffic). Likewise, a low own speed and a high previous speed means the traffic downstream is accelerating. It is exactly this dynamic behaviour that justifies adding another sample to the TrafficMap.

Note that the model proposed here is rather simple. Instead of a single slope a curve could be used. The reason why the offsets are used is that otherwise a traffic jam would constantly generate new samples because of the stop-and-go traffic. A more advanced version of the ε-function might include dynamic adaptation of settings such as the offset and the nature of the curve, possibly based on the density of samples. If there are few samples there might be need to increase sensitivity, just as an abundance of samples might indicate the sensitivity is too great and the ε-function needs to be adjusted accordingly.

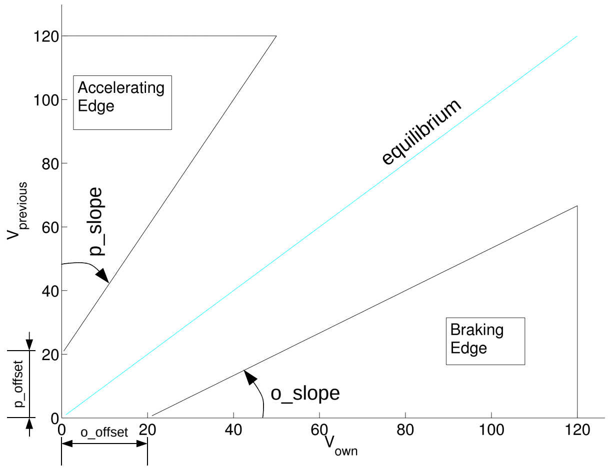


Figure 4.15: The threshold function ε mapped to a 2-dimensional plane

A distance-based threshold

So far the threshold is based on a speed difference. It might also be a good idea to increase the probability of adding a sample as the distance to the last sample increases. There could be a distance-related factor in the ε-function which reduces the threshold as the distance grows. Such an additive could guarantee, for instance, at least one sample per kilometer. It is expected that forcing a sample per kilometer can lead to more accuracy when changes develop slowly and over a large distance, such as the left-most sample in figure 4.12 on page 58. This sample would then be placed more to the right. A direct result is the increase in number of samples, for instance the TrafficMap in figure 4.12 would require in excess of 20 samples.

4.3.1 Adding a Sample to the TrafficMap

Based on the threshold ε a decision is made to add a sample to the TrafficMap, as illustrated in Figure 4.16. The first entry in the TrafficMap is a special field which is set by the node that (re)broadcasts it and will be overwritten on next rebroadcast, as a result the entry at index 1 is a moving variable and not part of the TrafficMap's over-the-horizon-view. This information can be used by OBU to, for instance, determine whether a node should rebroadcast and timers can be set accordingly [131] to enable efficient dissemination [103].

The encoding of the first entry, which denotes the position of the rebroadcaster, can be the same as the other entries in the TrafficMap if we are using a global positioning scheme (such as Absolute Coordinates or Road Information discussed in Appendix A.2). However, if the TrafficMap were to use a Relative Position encoding the position of the rebroadcaster needs to be expressed in a global manner, because this position functions as an anchor for the relative expressions.

When the own measurements differ significantly from the sample at index 2 (determined by the ε-function) the own measurement is *pushed* onto the stack. If it does not differ the entry at index 1 is replaced with the own measurement. The resulting TrafficMap is then rebroadcast. Note the resulting redundancy in the case a sample is added. This only occurs after adding a sample but does introduce some extra overhead.

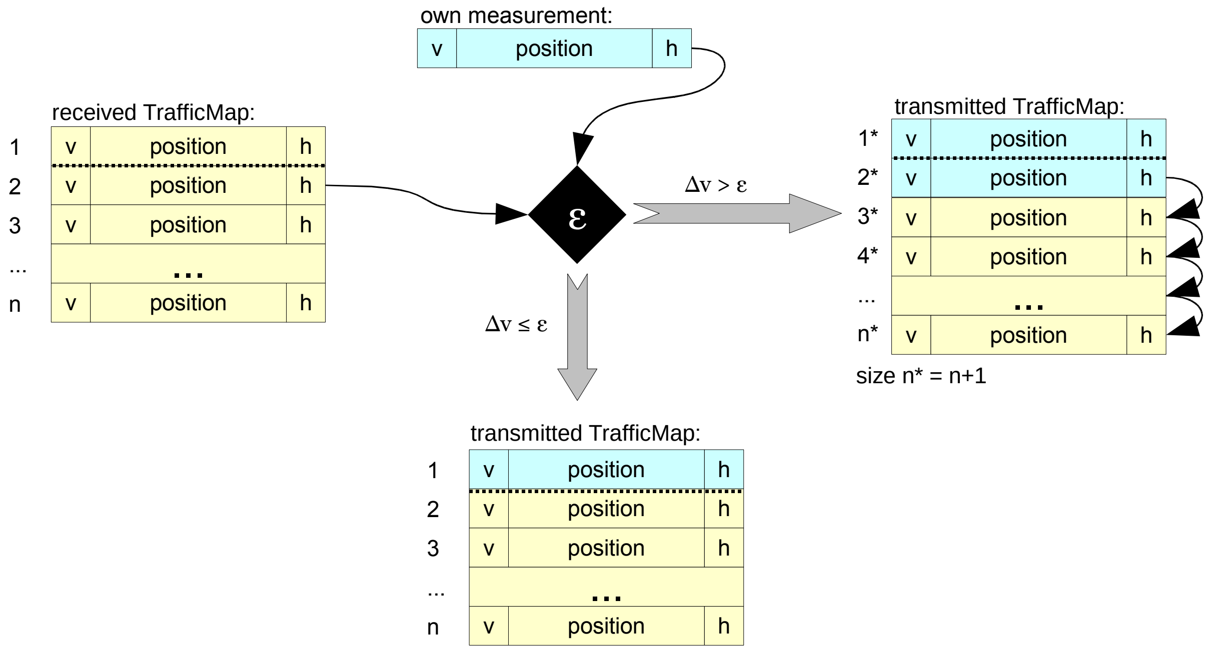


Figure 4.16: Adding a sample to the TrafficMap

4.3.2 Averaging

A single vehicle is responsible for adding an entry to the TrafficMap. Although vehicles are influenced by factors such as speed limits and other traffic it is clear that there can be deviations, even in free-flowing traffic. The difference between a car overtaking a heavy truck, for instance. In order to make a sample representative for the general area around the vehicle we could introduce elaborate majority-voting schemes, but a simple averaging would probably also suffice. The idea is as follows:

1. A node decides to add its measurement to the TrafficMap because it is allowed to do so by the ε-function.
2. The TrafficMap is rebroadcast.
3. A vehicle 100m upstream receives it. Its ε-function does not allow it to add a new sample. It might, however, slightly alter the last entry (at index 2) based on its own information if it is within the averaging distance Δ.
4. The TrafficMap is rebroadcast.

The result is that a sample is like a drop of paint, it gradually hardens and does not accept adjustment after a certain amount of time, or distance in this case, expressed as Δ. The averaging is expressed by the following equation:

$$\overline{v}^{\star} = \frac{\overline{v} + (\tilde{v}_{\delta} \times \theta_{\delta})}{1 + \theta_{\delta}} \tag{4.3}$$

Or in words: the resulting value $\overline{v}^{\star}$ is composed of the previous value of $\overline{v}$ (the v-component of the previous entry in the TrafficMap) plus a weighted amount of $\tilde{v}$ at position δ from the location where $\overline{v}$ was captured. The weighing is handled by θ_{δ} which is defined as follows:

$$\theta_\delta = \left(\frac{\Delta - \delta}{\Delta}\right)^\alpha \tag{4.4}$$

This gives a value between 0 and 1 for any δ between 0 and Δ, the averaging interval. By means of α we can tune the nature of the curve, as illustrated in Figure 4.17. Depending on α and Δ and the vehicle density a sample $\overline{v}$ is made by one or multiple vehicles. The value of α and Δ could be directly based on the density of traffic, the effects can be researched.

Equation (4.3) ensures $\frac{1}{1+\theta}^{th}$ of the original sample $\overline{v}$ is carried on in $\overline{v}^\star$. The result is an average calculated over an a priori unknown number of values.

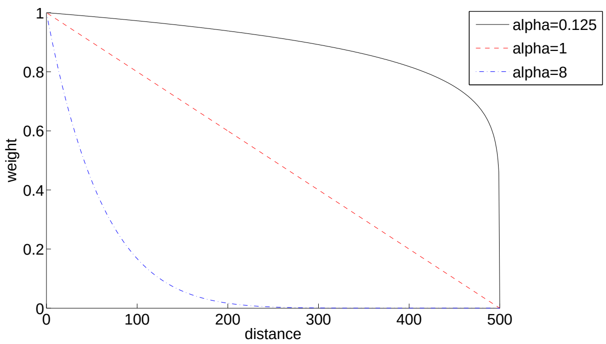


Figure 4.17: θ as a function of α and Δ

4.3.3 Reducing Redundancy

At the moment of sampling a node only has information on the previous samples, it does not know what the vehicles behind it are going to add to the TrafficMap. In fact, it will never learn this information as the information flow is against the flow of traffic. As a result there could be redundant samples in the TrafficMap like those shown in Figure 4.18. This is not a bad thing, it is better to capture a little more and then remove redundancy afterwards than have little information to begin with. Every node that rebroadcasts can perform such reduction operations under a couple of assumptions:

- By grace of the sample-and-hold concept the speed value of a sample is extended to the next sample in the TrafficMap (against the traffic flow)
- Two consecutive samples which are somewhat the same—especially in free-flowing traffic—can be reduced to one, the most remote one.
- Remote information has a high degree of uncertainty because the information is old and the situation might have changed. A small fluctuation can be removed.

Every node executes a processing step to see if redundancy can be filtered out. Redundant samples generated because of a generous ε-function can be removed or merged based on a complete overview of the redundant sample's up- and downstream conditions. This step is implemented in the reduceMap function. The reduceMap uses a simple means to remove tuples from the TrafficMap. Configuration variables are a distance beyond which reduction will be applied (`reduce_at`) and a window on which remote averaging will be applied (`reduce_interval`). Whether to keep or remove a sample is also threshold-dependent, defined by `proc_sens`. This is a simple sensitivity function (similar to the ε-function). In fact they could be implemented in

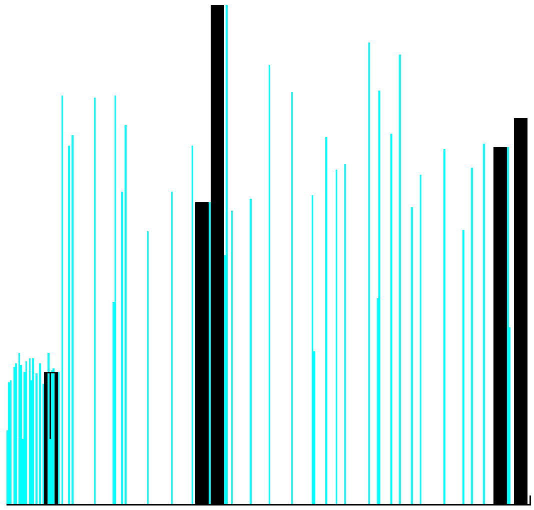


Figure 4.18: Redundancy in the TrafficMap

the same way but we will not go into further detail where the proc_sens is concerned because it is a matter of defining a threshold largely related to the situation on the road.

The reduction function does the following:

- remove or merge nearly equal consecutive values.
- summarise rapid in- and decreases. Here a set of stairs is reduced to less samples.

Just like the capturing threshold this is a part of the system which is heavily dependent on the nature of traffic on the road. The goal is to remove only redundant samples and reduce the size of the TrafficMap to keep the number of transmitted bytes low. This will be beneficial when the aim is to reach a large virtual horizon. This is illustrated in Figure 4.19. Here the original stretch of road is presented together with its representation twice as far away on the same road. The top figure is alligned to indicate the overlap. As can be seen, the observer of the TrafficMap on the bottom sees two traffic jams up ahead. The observer of the top TrafficMap only sees one (and has probably just passed the other one). Note that the top TrafficMap's 6 samples have been reduced to 4 in the bottom TrafficMap, without too much loss of detail.

The reduction step will also remove samples which are simply too far away (i.e. beyond the virtual horizon). Samples beyond this distance are discarded to ensure information only flows as far as defined by the virtual horizon. At this point no limitation to the size of a TrafficMap has been defined but it seems reasonable that, in order to meet demands of a maximum message size, remote samples may also be removed when there simply are too many samples to fit in one message. This could be the result of erratic dynamics in traffic. This will preserve the information close-by but will drop samples on the far end, effectively drawing the virtual horizon closer.

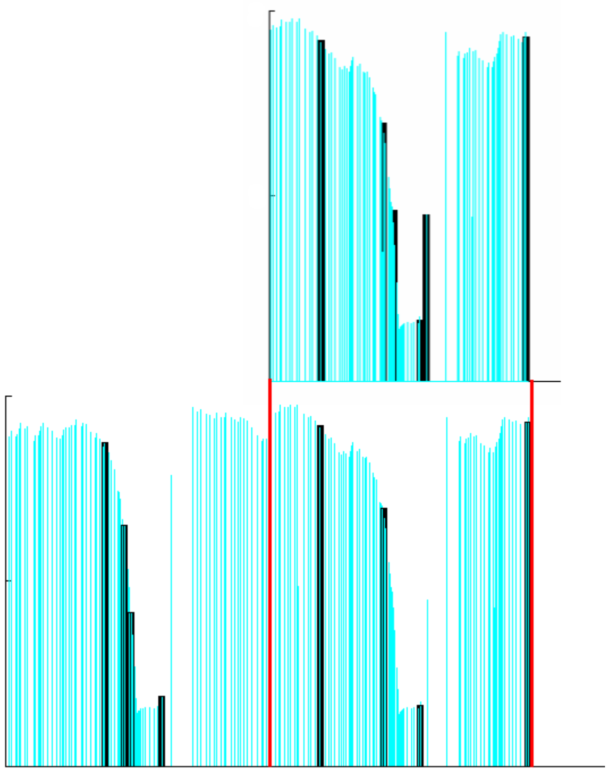


Figure 4.19: A TrafficMap is passed upstream. Some redundancy is removed along the way

4.4 Putting It All Together

We can now define the TrafficFilter:

TrafficFilter. – A distributed system present in the OBU of every (equipped) vehicle. The TrafficFilter protocol entity ensures efficient dissemination of messages which contain an over-the-horizon view called the TrafficMap. The TrafficMap is constructed as it travels through the VANET against the flow of traffic. Based on thresholds vehicles add, average and remove samples to ensure the TrafficMap contains an accurate over-the-horizon view up to the Virtual Horizon.

The TrafficFilter decides whether a vehicle functions as source, relay or latent node. For the system to work it is important all equipped vehicles are observers, i.e. reachability must be close to 100%. □

The TrafficFilter performs three functions: add, average and remove samples in the TrafficMap. The TrafficFilter uses the ε-function in order to determine if another sample should be added to the TrafficMap. Every node evaluates the received information and will only add a new sample if it can contribute new information as defined in Section 4.3.1. Nodes can still contribute 'a bit' if they are within the averaging interval Δ, as mentioned in Section 4.3.2. This way all nodes work together to construct the TrafficMap as a representation of reality.

To cope with the fact that no node has complete knowledge of its up- and downstream at the same time a hind-sight function removes or merges samples as described in Section 4.3.3. Here redundancy is removed from distant samples while keeping the important remote information intact.

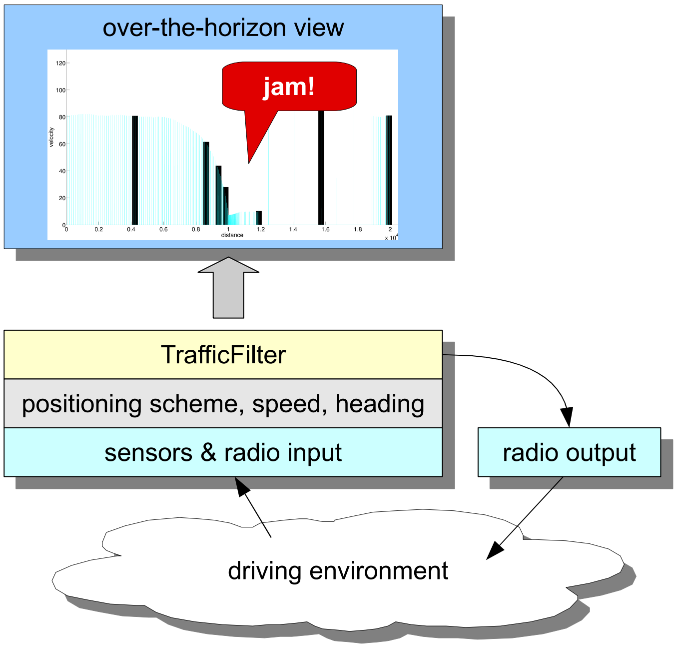


Figure 4.20: The TrafficFilter

The result is a distributed system that will generate an over-the-horizon view in an efficient manner. Efficient with respect to the number of bytes that need to be communicated, because every node will still have to perform some computations. It is reasoned that, if a node has nothing to contribute to the TrafficMap *and* it overhears a node further upstream rebroadcast the TrafficMap it may refrain from broadcasting. Such a node will then temporarily function as a latent node, because it is neither source nor relay. This is an important foundation for the dissemination strategy defined in Chapter 5.

4.4.1 A Note on Parameters

The TrafficFilter generates a TrafficMap. The quality of the TrafficMap (e.g. how well it represents the actual situation) depends on the present situation of traffic on the road and how the various configuration variables of the TrafficFilter are set. It is reasoned that these variables might need to be dynamically set based on the nature of traffic—speed, density etc.—in order to result in the best TrafficMap over a wide variety of traffic situations. The variables of the system, their meaning and default value are summarised in Table 4.3. Values are chosen based on generating results that 'to the eye' represented the actual situation (as generated by the IDM road model) *reasonably well*. Note that in this stadium these values cannot yet be defined as it is hard to relate them to empirical data.

At this stage the TrafficFilter is defined at a conceptual level. A more complete overview spanning across layers in the protocol stack will be provided in Chapter 5.

Variable	**Meaning**	**Default value**
θ-function	averaging function based on distance	α, Δ
α	averaging slope	3
Δ	averaging interval	500m
ε-function	capturing threshold	`o`- and `p`-components
`o_offset`	trigger-free zone	5
`o_slope`	rise of sensitivity w.r.t. braking	8/9
`p_offset`	trigger-free zone	7
`p_slope`	rise of sensitivity w.r.t. acceleration	5/6
Virtual Horizon	when to drop remote samples	10000m
`proc_sens`	reduction sensitivity	40km/h
`reduce_at`	reduce beyond this distance	$\frac{\text{Virtual Horizon}}{2}$
`reduce_interval`	averaging window for reduction	1000m

Table 4.3: Configuration parameters of the TrafficFilter

Chapter 5

Dissemination of TrafficMap information

In the previous chapter we introduced the TrafficFilter, a distributed system that generates an over-the-horizon view in every equipped node on the road. The TrafficFilter's task is to select the samples that are added to the TrafficMap by means of the add, average and remove functions described in Section 4.3. This TrafficMap is then disseminated against the flow of traffic by means of a directional flooding scheme. This chapter will focus on the details of such a flooding scheme.

Vehicles moving on a road form a Vehicular Ad hoc NETwork (VANET). VANETs have some properties which differentiate them from MANETs (as described in Section 3.4.2). Mobility is generally constrained because the nodes follow roads. This results in predictable mobility patterns (within certain bounds). Speed is generally high in VANETs, but can also differ greatly; from communication between stopped vehicles to communication between vehicles moving in opposite directions.

In contrast to MANETS, nodes in a VANET generally do not have strict power, weight and size limits. This allows for more computations to be performed—there is enough power available and more powerful hardware can be used. As such a node can safely be assumed to be aware of its position, and this information can be used for efficient dissemination.

5.1 TrafficFilter Broadcast

Every equipped vehicle on the road needs to receive TrafficMap information in order for its Congestion Assistant to work. To ensure this, every equipped node also has to take part in the distribution of the TrafficMap information. As a result, every OBU is equipped with a TrafficFilter protocol entity.

The best way for distributing information targeted at the entire population in an ad hoc network is a flood. As mentioned before, the transmission range of nodes is limited and only covers a few tens of other nodes at best. The solution here is to have every node repeat the information to its neighbours. There are roughly three approaches to flooding: blind flooding (every node repeats the message), sender-based decision [67, 85] (the sender tells a node to pass it on or not) and receiver-based decision [121, 117, 120] (a node decides for itself whether to pass information on or not).

Quite often blind flooding is also referred to as simple flooding [117]. A sender-based decision flooding scheme can be used if the sender has some neighbour knowledge [67, 85]. This can be, for instance, when a tree has been setup on which the information is distributed. In this case the sender can tell nodes with a high number of neighbours to broadcast (they are the branches of the tree) while the nodes with few neighbours (the leafs) refrain from broadcasting.

A receiver-based decision flooding scheme places the decision to rebroadcast with the receiver. Upon reception of a message, a node evaluates whether it should rebroadcast. A receiver-based flooding scheme is the most appropriate for an ad hoc network with a transient population and topology because little state has to be maintained.

These receiver-based decisions can be Probability Based [1], Counter Based, Area / Distance / Location Based or Neighbour Knowledge Based [117, 131]. The simplest approach, blind flooding, simply rebroadcasts with probability 1. A node can count how many rebroadcasts in hears in a certain time frame and if a threshold is not exceeded it can rebroadcast. Nodes can also perform some calculations based on positions; for instance a node can calculate from received information which area has already been covered by a broadcast. It can then judge whether its own broadcast adds a significant contribution to the coverage.

Quite often a proposed scheme will use several elements, such as the Border Aware Floodingscheme proposed by Zhu *et al.* in [131]. This scheme combines distance estimation with a counter in order to reach high coverage with a minimal number of broadcasts.

It is identified by Wisitpongphan *et al.* in [121] that some of the research in the field of MANETs can be applied to VANETs, but several differences result in the demand for specialised flooding techniques. The differences identified in [121] are as follows:

- VANETs consist of highly mobile nodes moving at high speeds, generally moving in the same or opposite directions.
- Other than most MANETs (which are often modelled as square or torus topologies) a VANET is usually shaped as a one-dimensional line or a strip (a line with a small width due to several lanes) or a grid composed of multiple lines.
- Most VANET communication relies on broadcast transmission to disseminate information to all reachable nodes in a certain geographical region, as opposed to queries for routes and unicast routing in MANETs which usually aim to deliver the message only to certain nodes. This difference lies in the fact that the message contained in route requests are of no use to many of the nodes, while the information contained in a message that contains warnings or traffic information is of concern to many nodes.

Two other major differences between MANETs and VANETs are node capacity with respect to memory, computational power and battery life and the access to peripheral devices. A

VANET generally consists of nodes the size of a car; hence more bulky equipment can be installed. Because vehicles are powered by engines and generally contain large batteries that are constantly being recharged by alternators it is possible to install high-power hardware. This is opposed to the generally ultra-lightweight hardware of MANET nodes. A VANET node also has enough power to transmit at high power rates without the need for power-save modes while some MANETS are configured to sleep every now and then, to consume as little energy as possible in order to expand battery lifetime. This class of MANET is often referred to as a Wireless Sensor Network (WSN) [112].

Because most modern cars are equipped with audio systems, navigation equipment and a plethora of sensors in the vehicle a node in a VANET can potentially have access to all these systems, acting as peripherals to supply information. Whereas a MANET node generally does not come equipped with a GPS unit—it takes up space and weight and requires a lot of power from the battery—a VANET node can safely be assumed to have positional information readily available.

These factors, of course, have some effects on the flooding techniques we can employ.

5.1.1 Broadcast Suppression Techniques

An efficient flooding strategy makes use of a broadcast suppression technique to combat the Broadcst Storm problem. The ultimate goal is to have a minimum set of nodes rebroadcast while still reaching the entire population. Simple flooding can be described as 1-Persistence flooding, it is a brute-force scheme where every node rebroadcasts. Ensuring some of these nodes refrain from broadcast helps prevent the Broadcast Storm problem. Wisitpongphan *et al.* evaluate three schemes [121]:

Weighted p-Persistence Broadcasting – When node j receives a packet from node i it checks the Flood ID. If a packet with the same ID has been received before j does nothing, else it rebroadcasts with probability p_{ij}. This probability is derived as follows:

$$p_{ij} = \frac{D_{ij}}{R}$$

Slotted 1-Persistence Broadcasting – When node j receives a packet from node i it checks the Flood ID. If the packet has not been received before, it broadcasts with probability 1 in the assigned time slot $T_{s_{ij}}$.

Slotted p-Persistence Broadcasting – The same as the Slotted 1-Persistence scheme, but a node rebroadcasts with probability p in the assigned slot. A values of $p = 0.5$ is considered.

It was concluded in [121] that the slotted 1-Persistence and slotted p-Persistence schemes provided the greatest reduction in broadcast redundancy while still offering acceptable end-to-end delay and reachability. It was found that slotted 1-Persistence provided the fastest dissemination. As such this will be the flooding strategy used in this research. Section 5.3.4 provides more details on the flooding strategy.

5.1.2 Temporal Aspects of the Flooding

1. Every reachable node in the network must receive a TrafficMap at least once per time unit. This implies a maximum period between two consecutive TrafficMap floods: the Maximum Inter-TrafficMap Time (MIT)

2. Every reachable node in the network must receive a maximum number of TrafficMaps per time unit. This implies a minimal period between two consecutive TrafficMap floods or a maximum number of TrafficMaps per time unit: the TrafficMap Flood Limit (TFL)

The Maximum Inter-TrafficMap Period

Dynamics on the road occur in space and time. Besides positions and speeds the rate at which they change plays an important role. The exact value of the MIT is subject to the traffic situation and needs to guarantee important changes in traffic dynamics do not go unreported by the system—a case in which the driver would be able to see a change before the Driver Support System does, rendering the Support System useless. At this stage a worst case scenario analysis will have to suffice.

The CA's Active Pedal is defined to operate at a distance of 500-1500m from the tail of the traffic jam. It is reasoned that the system must at least be able to communicate a change at 500m faster than the change starts to affect the vehicle. Traffic moves forward, as such oncoming vehicles move towards the location of the unexpected change—for instance, a stopped vehicle. Such changes are not always stationary like a stopped vehicle but may also move with or against the flow of traffic. Work performed by Kerner [59, 60] describes so-called *wide moving jams* [57], congestions moving against the flow of traffic. These are found to move against the flow of traffic at speeds between 14 and 16 km/h (4.44m/s).

It is reasoned that a vehicle must receive TrafficMap information at a safe distance from such an anomaly in traffic flow in order for automated systems or the driver to perceive—preferably the automated systems before the driver. A simple worst case scenario is given next. This scenario considers a *jam* moving *against* the flow of traffic. The case that the vehicle in front suddenly breaks down (accident) is not considered here. Designing the TrafficFilter conform constraints required by safety-of-life applications such as Collision Avoidance Systems automatically results in higher demands. Such systems are out of the scope of this research.

> A convoy of 8 vehicles is driving on the highway at 120 km/h (33.33m/s) spaced at 66m (the recommended 2s headway). Suddenly a stopped vehicle shows up 500m ahead. It takes about 15s for the first vehicle to reach the obstruction. Assuming 5m per vehicle the 'obstacle' of crashed cars will move against the flow of traffic at 2.66m/s.

A vehicle moving at 33m/s will encounter the tail of this crash at $33 + 2.66 = 35.66m/s$. At this speed the 500m distance is covered in 14.02s. To come to a full halt when traveling 120km/h requires about 150 meters [98] including 1s reaction time. It is reasoned that communication times are small compared to human reaction time so communication times can be ignored in this estimate. At 120km/h and a traffic anomaly moving against traffic and a safe braking distance of 150m results in a lower-bound of $150/(33.33 + 4.44) = 3.97$ seconds. This means that, if the system is to react to such dynamics in traffic, it needs to be informed in time (i.e. within the 3.97 seconds derived above).

It seems reasonable to require that every vehicle receives a TrafficMap every 3.0 seconds because otherwise the system would not be able to react to sudden changes in traffic dynamics. This means we set the MIT to 3 seconds. Not receiving a TrafficMap for one MIT can be the result of bad connectivity in the network: there are few vehicles or few equipped vehicles within range, or there are other reasons for bad propagation. In the first case the system cannot operate but that is not a problem—there is no traffic. In the second case it is important to know the system is not able to operate and as a result the Congestion Assistant cannot operate reliably. We reason it is important to inform the driver of the operational status of his Driver Support Systems because of the trust placed in them.

As reported in [121] and confirmed by our own findings (presented in Chapter 6) a distance of 10km can be covered in the order of several tens of milliseconds, a time small compared to human reaction time ($\sim 1s$).

'Traditional' flooding schemes such as discussed in [121] assume there is one source which wants to spread its information across the entire network. In the case of the TrafficFilter the flood has multiple sources—any node can be a source, and any node can initiate a new flood if too much time has passed since last reception of TrafficMap information. This means that multiple independent sources can inject floods into the network and these cannot be blindly flooded because the traffic might aggregate to unacceptable levels. It becomes important for a node to judge if propagation of a received TrafficMap can be justified, should be postponed or cancelled all together.

Just like stretches of road where traffic dynamics trigger addition of samples to TrafficMaps the same reason may also trigger generation of multiple TrafficMap floods. The system should be able to reduce such peaks to 'normal' waves of TrafficMaps by summarising and merging information from several TrafficMaps to one when the observed local traffic is stable.

Of influence are the following factors:

- Density of nodes - expressed as ρ. The probability of rebroadcast could be inversely proportional to ρ, when there are more potential rebroadcasters they can each have a lower probability of rebroadcast while still maintaining a high probability that *at least one* of the nodes rebroadcasts. When ρ is low probability of rebroadcast must be high because there are few other vehicles that can do it.

- The flow speed of the traffic - when the flow is stable there is not much to report. It might be a waste of resources to propagate unchanged information through the network. Two exceptions are:

  - Absence of TrafficMap messages may indicate propagation failure, rendering the system *blind*. Periodic broadcast based on the MIT period will function as a 'heartbeat'. There are TrafficMap 'sessions' active and the road is being observed. This can be an indicator of the reliability of the information used by applications or presented to the user.
  - Nodes which were previously out of range (e.g. just enter the highway) may detect that TrafficMap messages are being flooded and can participate.

- Direction from which a TrafficMap message is received:

  upstream a TrafficMap message received from upstream (behind the vehicle) indicates that propagation is taking place upstream. A node does not have to do anything, this is a confirmation that the flood has passed the current node or it concerns information on a location which the vehicle has already passed.

  downstream a TrafficMap message received from downstream (in front of the vehicle) can contain new information. A node uses this information in the external estimator and passes this information on to vehicles upstream.

5.2 The TrafficMap Message

The TrafficFilter protocol entities in every node exchange messages referred to as TrafficMap Protocol Data Units (TM-PDUs). The information contained in the messages has been defined in Chapter A where the TrafficMap was defined to express an over-the-horizon view. The absolute positioning scheme discussed in Section A.2.1 is chosen for a proof-of-concept for simplicity, resulting in 10 bytes per TrafficMap entry. The structure of the TM PDU is presented in Figure 5.1. The information can be divided into two parts: the information required for efficient flooding and the information which constitutes the TrafficMap. The first part is the Network-layer PDU header, which encapsulates the Application layer information, the second part.

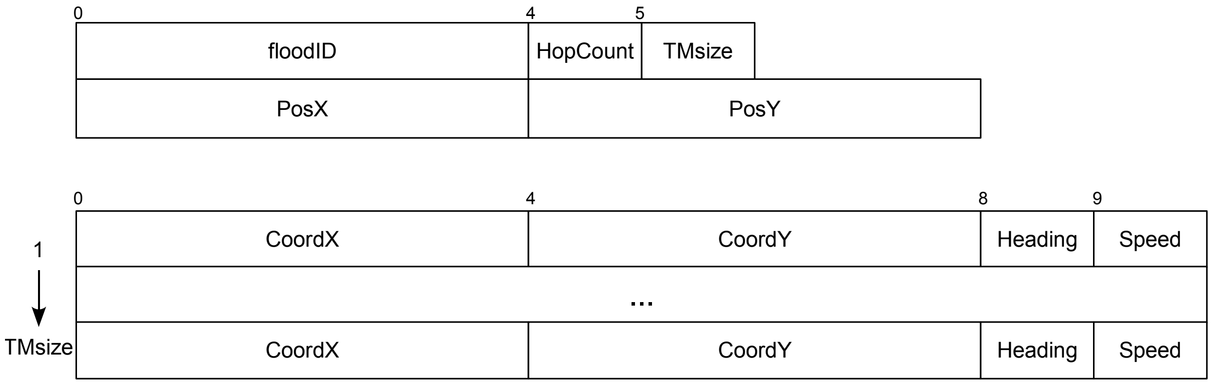


Figure 5.1: Structure of the TrafficMap Message

5.2.1 The Network-layer PDU

The Network-layer PDU contains a floodID, a hopCount, a TMsize and an expression for the position of the sender, this being the vehicle which transmitted the message (i.e. the vehicle that rebroadcast a flood or initiated a new one). The floodID is a 32-bit (positive) number which uniquely identifies a flood. This aids the flooding strategy in determining reception of duplicate messages (e.g. rebroadcasts by other nodes). The hopCount could serve a purpose in estimating the condition of the network up ahead: if a high hopcount is observed this means the network is not partitioned down the road, and conclusions can be drawn on the quality of the over-the-horizon view. When a node schedules to rebroadcast a received message the hopCount is incremented by one. The hopCount is also used in estimating the efficiency of the system in the simulator in Chapter B.

The TMsize byte contains the number of samples in the payload, i.e. how many samples there are in the TrafficMap. As a result, the size of the payload is derived as `TMsize * sizeOf(TMEntry)` and will be multiples of 10 in this case. TMsize can be a value between 0 and 255, but in this research the TM is generally envisioned to contain around 30 samples, resulting in a payload of 300 bytes.

The expression for the position of the sender is used by the flooding strategy. Because this is expressed in the Absolute Coordinates the position takes up 8 bytes. If an expression for the lane on which a measurement was taken were to be included an additional field would be added. This is not considered further in this research.

5.2.2 The Application-layer PDU

The application layer contains the TrafficFilter logic, as such the Application-layer PDU is made up of the tuples of data which together form a TrafficMap. Every sample in the TrafficMap is made up of a way to denote this position, eight bytes in this case for an X and Y coordinate, and a speed and heading indicator. As a result a complete TrafficMap message at MAC level will count $14 + TMsize \times 10$ bytes.

How this translates to an implementation in the simulator is covered in Section B.4

5.3 Disseminating TrafficMap Messages

This Section covers the design of the TrafficFilter protocol entity. It will perform the TrafficFilter operations *add, average* and *remove* as defined in Section 4.3 that operate on the TrafficMap and ensure timely and efficient dissemination among all relevant vehicles.

The dissemination strategy relies on a modified version of the slotted 1-Persistence Flooding scheme and will be both timer- and eventdriven. The design given here is focused on actions executed and states in which the system can be. How this can be mapped to different layers and functional blocks is covered in Chapter B.

5.3.1 Timers

A timer τ can be used to guarantee a Maximum Inter-TM period is honoured: if $(\tau = MIT) \wedge (noTMhasbeenreceived) \Longrightarrow$ Broadcast a TrafficMap. When a broadcast is overheard τ will be reset.

A timer can also be used to periodically (with timer $\mu\tau_{max} \ll MIT$) check the internal estimator. The ε-function will be evaluated based on the current speed v_{own} and the stored $v_{previous}$ (which may be several seconds old). The ε-function concludes whether a sample should be added. When a sample has been added, immediate broadcast must follow to convey this information to vehicles upstream.

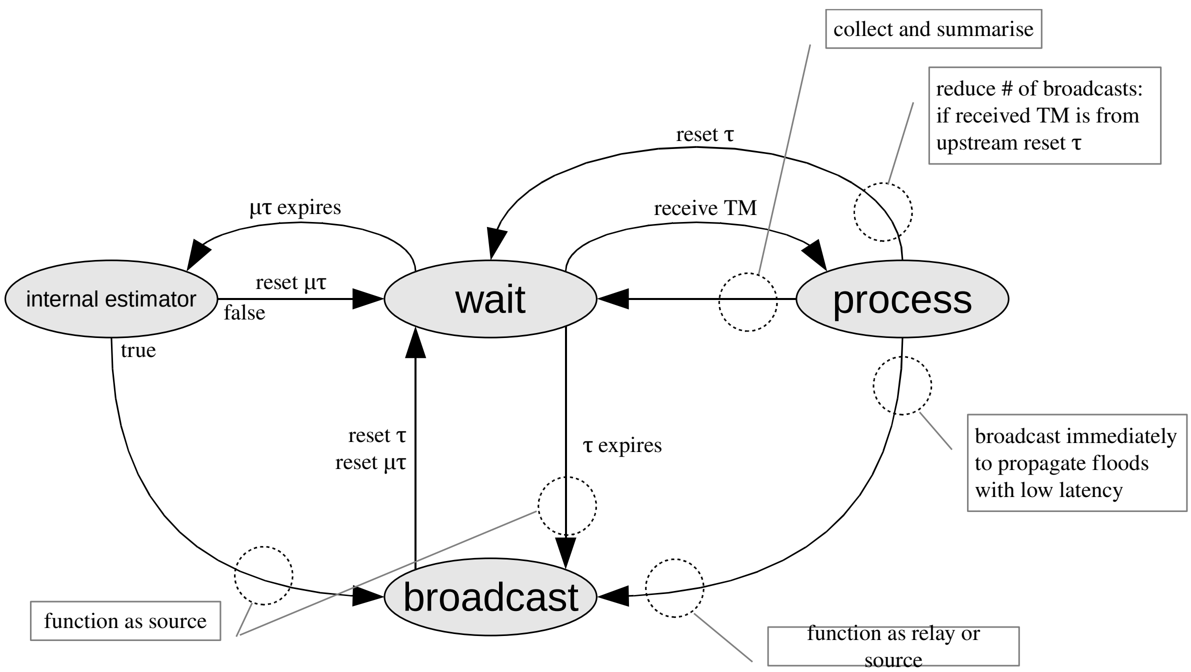


Figure 5.2: A Broadcast Strategy with timers and events

5.3.2 Events

An event can trigger a reaction. Receiving a TM triggers the processing of it (use the information contained in the message, add a new sample, average the data present in the TM or remove some of the data in the TM) and potentially, rebroadcast. In this case the event has an external origin.

An event can also be generated by the internal estimator when the ε-function finds a deviation that justifies adding a new sample to the TrafficMap. This can trigger a broadcast because the information that has just been added is important (otherwise the ε-function would not allow a new sample to be added) and needs to be disseminated at once.

Figure 5.2 shows a state transition diagram of the dissemination strategy. The system is initiated in the 'wait' state and will leave this state based on events or timers (such as the expiration of τ). If a TM is received control moves to the state 'process', where the received TrafficMap is processed for use by the Congestion Assistant. Based on the relation between p_{own} and p_{sender} a direct broadcast will be executed after which τ will be reset and the system will return to the 'wait' state, or the system will reset τ and immediately return to 'wait'.

The state 'internal estimator' decides whether a sample should be added based on internal data. If this is the case a broadcast will be performed as soon as possible in order to publish this information with low latency; an exceptional situation has just occurred and other nodes on the road must be notified.

The 'broadcast' state in Figure 5.2 consists of two components: a CSMA/CA broadcast provided by the 802.11 MAC that will transmit as soon as the medium is idle and a Flooding Scheme for disseminating flooded TrafficMap information. This latter event happens in reaction to reception of a TrafficMap message, the former happens upon expiration of τ or when triggered by the 'internal estimator', a new flood will be initiated.

As a result, the system is timer/event-driven. The degree to which one of the two methods is used determines the latency, because with a pure timer-based system a node may have to wait a whole MIT period before rebroadcast can occur. In this case, using $MIT = 3s$ and a transmission range of 250m it can take $\frac{10000}{250} \times 3 = 120s$, or two minutes, before 10km has been traversed (worst case). Clearly this is not desirable.

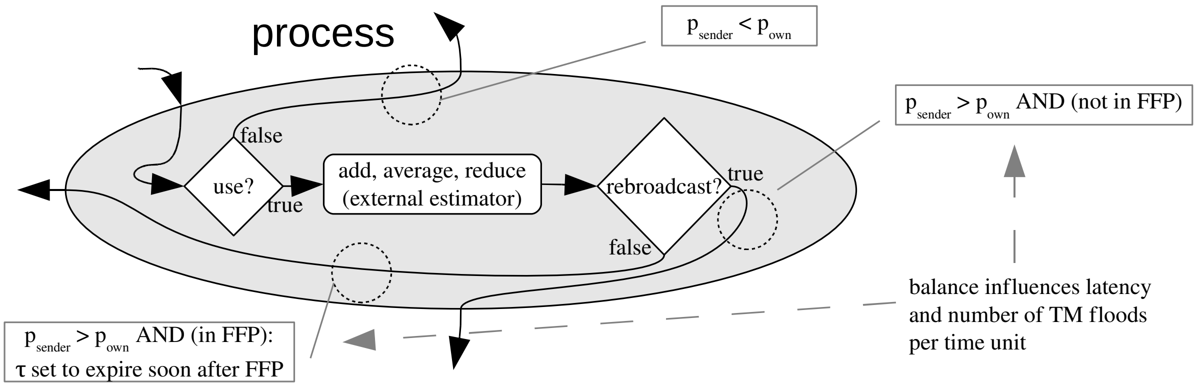


Figure 5.3: The Process state in more detail

Figure 5.3 shows the 'process' state in more detail. If the received message originates from downstream (e.g. ahead of the current vehicle) it is used for further processing. Next the message goes through the external estimator which compares the received TrafficMap to the local TrafficMap. It decides to add, average or remove samples. This process is the core of the TrafficFilter as described in Chapter 4 on page 45 and is summarised in Figure 5.4.

After the external estimator completes the decision is made whether to broadcast directly or enter the Flooding Scheme. In order to prevent creation of too many floods several floods can be combined using a 'collect and summarise' approach. The 'collect and summarise' function ensures there is a certain maximum number of TrafficMaps per MIT-period.

The system uses two broadcast methods: synchronous and asynchronous, see Figure 5.5. The synchronous broadcast is the Flooding mechanism, which is synchronised by reception of the same message at a group of nodes in a certain area. The rebroadcast will take place in the node furthest removed from the sender, in order to rapidly cover many kilometers.

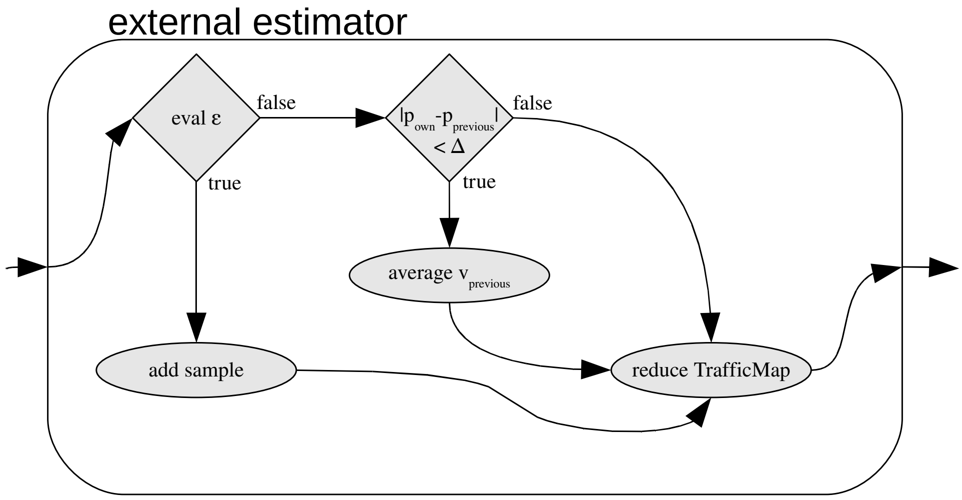


Figure 5.4: The External Estimator in more detail

The asynchronous broadcast is used in two cases:

- An event happened in the internal estimator that triggers transmission of a TrafficMap, initiating a new flood. A broadcast is scheduled as soon as the medium is perceived to be empty (by CSMA/CA).
- The τ-timer has expired and to satisfy MIT we need to initiate a new flood. This means a MIT-period has passed in which no TrafficMap flood has passed the current vehicle. This can be caused by gaps in the network; existing floods have died out because the chain of propagation is broken. A node will inject a new flood.

It is envisioned that the majority of transmissions will take place as synchronous transmissions (i.e. mediated by the flooding scheme) for maximum efficiency. A broadcast triggered by an event does not imply *immediate* broadcast, as more safety-critical application may get priority in the MAC layer.

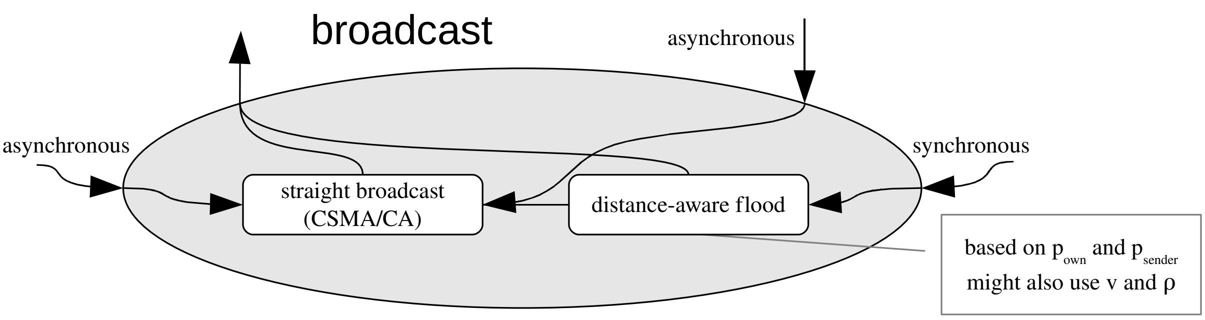


Figure 5.5: The Broadcast state in more detail

5.3.3 Flooding

The system keeps floods that are moving against the flow of traffic going. If an event happens of which others should be notified or when for a MIT period no flood has been observed a new flood is launched. The flooding prioritises rebroadcast by distant nodes by using the slotted p-Persistence strategy which has been proposed by Wisitpongphan *et al.* in [121] and [120].

Because priority depends on distance the situation can occur that a node that has just added a sample does not 'win' the rebroadcast (it does not propagate), as a result his update is not included in the flood. To cope with this, these source nodes may receive priority over relay nodes.

5.3.4 Slotted p-Persistence Flooding

When a broadcast is received by a node transmission of the rebroadcast will be scheduled in timeslot S_{ij} with probability p. In [121] the probabilities $p = 1$ and $p = 0.5$ are evaluated. p is found to be of influence on the link load and the packet penetration rate, the rate at which a message spreads through the network. The findings in [121] were that $p = 1$ results in the lowest delay but a higher link load. Because the aim is to cover 10km by the flood low end-to-end delay is of importance. For this reason we set $p = 1$.

Scheduling a rebroadcast occurs only when the packet has been received for the first time, this can be recognised by the Flood ID. If a duplicate is received before the rebroadcast is executed the packet will be discarded and rebroadcast will be cancelled; this node will not rebroadcast because an other node already did.

The timeslot chosen by a node to rebroadcast a packet follows from Equation (5.2) and depends on D_{ij} (i.e. the distance between the nodes i en j where j is the node which transmitted the message), the (estimated) transmission range R and an a priori determined number of slots N_s. The time a node has to wait $T_{S_{ij}}$ can be calculated as follows:

$$T_{S_{ij}} = S_{ij} \times ts \tag{5.1}$$

where ts (the one-hop delay or slot time) is the sum of medium access delay and propagation delay. S_{ij} is the allocated slot that is determined as follows:

$$S_{ij} = N_s \left(1 - \left[\frac{min(D_{ij}, R)}{R}\right]\right). \tag{5.2}$$

The result is that a node for which D_{ij} is larger will pick an earlier timeslot and will sooner be able to rebroadcast, but only if no other node rebroadcasts in the mean time, as depicted in Figure 5.6. Here node S is the sender and nodes 1–5 are the possible retransmitters. Because node 5 has the largest distance to S ($\forall_i | i \in (1, \ldots, 4) \Rightarrow D_{5S} > D_{iS}$) node 5 will pick an early timeslot and hence a short wait time. Node 5 will transmit first, and nodes 1–4 will refrain from rebroadcasting.

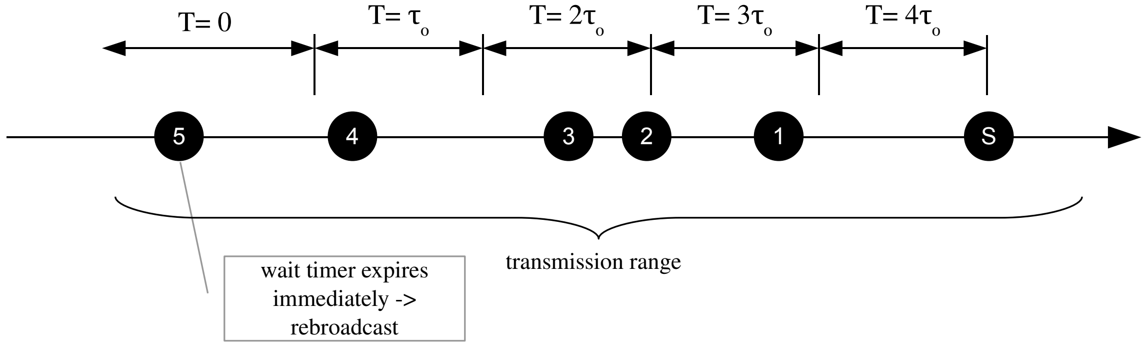


Figure 5.6: Slotted 1-Persistence: node S broadcasts a message, node 5 rebroadcasts and nodes 1–4 refrain from rebroadcasting.

The distance-aware flood component in Figure 5.5 is presented in detail, based on the slotted 1-Persistence Flooding, in Figure 5.7. First off, a check is executed to see if this packet has been observed before by means of a floodID as defined in Section 5.2.1. In case the packet has been seen before, it is dropped at once because it is a rebroadcast. If it has not been seen before it is important the message is rebroadcast by one of the nodes in the area, so a timeslot is determined based on the distance to the sender as defined in Equation (5.2). The node then waits for the time $T_{s_{ij}}$ to pass. If this timer expires and no duplicates are received the packet is transmitted.

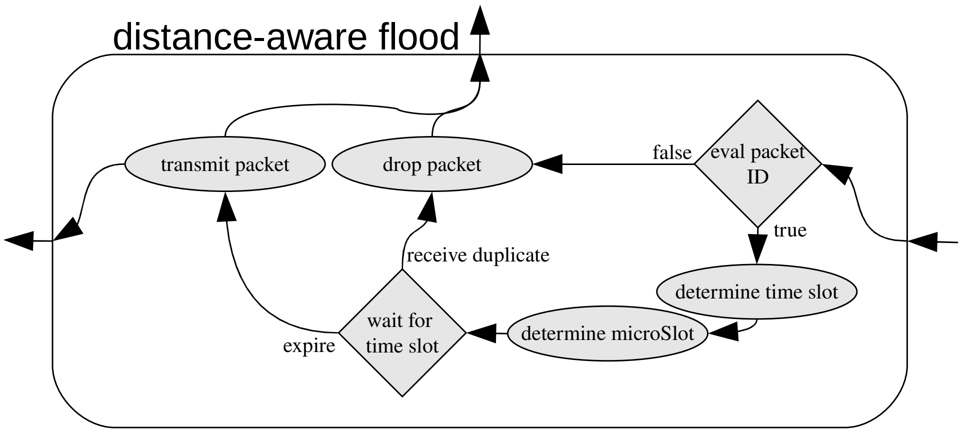


Figure 5.7: Distance-aware flooding by means of Slotted 1-Persistence Flooding

Note that the slotted 1-Persistence Flooding relies on an estimated transmission range, expressed as R in Equation (5.2). Using an estimate which corresponds to the actual *real* transmission range is a determining factor in the performance of the flooding scheme. An evaluation of the effects is provided in Appendix B.5.

Breaking Synchronisation

The slotted 1-Persistance flooding scheme breaks the synchronisation of 1-Persistence flooding, which would otherwise result in all nodes rebroadcasting simultaneously. It is identified, however, that a similar synchronisation—albeit on a smaller scale—can occur within one slot when vehicle densities are higher. A means to solve this problem is using a probability less then 1, as in the p-Persistence or choosing a larger number of slots N_s. The first scheme was shown not to perform as well as the slotted 1-Persistence and the second option will also deteriorate performance: the slot time ts will remain constant because propagation and MAC processing delay are constant, but with more slots the cumulative delay can be much higher. We propose a solution which relies on the CSMA/CA in the IEEE 802.11 MAC layer, and alter the slotted 1-Persistence flooding scheme accordingly.

The proposed scheme uses the fact that the MAC layer performs Clear Channel Assessment (CCA) before a transmission. If multiple nodes want to broadcast contention occurs, unless they are synchronised within a time slot as in the slotted 1-Persistence scheme. If the nodes wait a small additional time (chosen from a certain window) they will not all start to transmit at the same time. If the medium is observed to be busy, a node will back off and retry later.

The minimum time between transmissions is defined in the standard [48] as SIFS (Short Inter Frame Space). If we are to use DIFS (Distributed Inter Frame Space) the MAC in every node involved in the contention can determine a transmission is going on, and later retry. Because this scheme is a slotting scheme at a fine granularity it is referred to as the microSlotted 1-Persistence

Flooding scheme. This scheme defines a number of microSlots to wait additional to the wait time defined by the slotted 1-Persistence Flooding scheme:

microSlotted 1-Persistence Flooding. – Functions the same as the Slotted 1-Persistence Flooding scheme but adds a small additional delay of $T_{ms_{ij}}$ (i.e. [0-9] DIFS) to every $T_{s_{ij}}$ defined by Equation (5.1). $T_{ms_{ij}}$ is defined as:

$$T_{ms_{ij}} = N_{ms} \times \left(1 - \left[\frac{min\left((D_{ij} \text{ mod } S), S\right)}{S}\right]\right) \times DIFS \tag{5.3}$$

where the modulo operator scales the domain D_{ij} to the size S of the timeslot within the nodes are located. This gives priority to the most remote node within a slot. S is defined as the geographical size of a slot:

$$S = \frac{R}{N_s}$$

and N_{ms} is the number of microSlots per slot based on the estimated transmission range R, number of slots N_s and average vehicle length:

$$N_{ms} = \frac{\left(\frac{R}{N_s}\right)}{\text{avg. vehicle length}} \tag{5.4}$$

As a result, the total wait time for node i after receiving a packet from node j is derived as the sum of Equations (5.1) and (5.3):

$$T_{wait} = T_{s_{ij}} + T_{ms_{ij}} \tag{5.5}$$

Node i will schedule to hand the message over to the MAC layer after T_{wait}. Behaviour is just like the slotted 1-Persistence Flooding: If, in the mean time, a packet is received with the same Flood ID the pending transmission is cancelled; another node performed a successful broadcast and node i does not need to contribute. □

This flooding strategy assumes an 802.11-compatible MAC and PHY because it relies heavily on its CSMA/CA system. This strategy is designed to use a standard MAC, assuming it is not altered and we have no control over its operations. This means that, once the Network layer sends a message down to the MAC for broadcast, it cannot be recalled. The 802.11 MAC ensures that the scheduled transmissions are executed serially in stead of parallel (which would result in collisions). A benefit is that the probability that at least one broadcast in a slot is successful increases.

A drawback is that all messages scheduled in the MAC layer still have to be executed, resulting in a larger medium busy time. An evaluation of the performance of the microSlotted and the original slotted 1-Persistence Flooding scheme is provided in Section 6.2.

5.3.5 Source Node Priority

There are two conflicting interests in the system:

- Rapidly cover a great distance
- Capture the best possible TrafficMap

We want to traverse the network in as few hops as possible. Hence we want a rebroadcast to be carried out by the most remote node. In the mean time, we want a good match between the TrafficMap and the actual situation. This suggests a rebroadcast by the node which has just added a sample to the TrafficMap.

Now, when for instance node 3 in Figure 5.6 has observed a deviation in traffic (e.g. hard braking) node 3 may add a new sample to the TrafficMap. However, node 3 does not get a chance to transmit this TrafficMap, because node 5 won the race for the rebroadcast. It is probable that the deviation that causes node 3 to add a new sample will also influence other nodes but it could very well be that important information is lost or at least delayed from being disseminated. This could be solved by shifting all timeslots one slotTime and reserving $T = 0$ for source nodes, a method we will call Source Node Priority (SNP). Node 5 still is the most remote node and will rebroadcast after one slotTime. Node 3, however, has important news and claims slot 0: it rebroadcasts immediately. This is illustrated in Figure 5.8. Because it does not happen often a node adds a sample to the TrafficMap the first slot will now not be used primarily to rebroadcast but the second slot will be used primarily. The result will be an additional slotTime delay per hop.

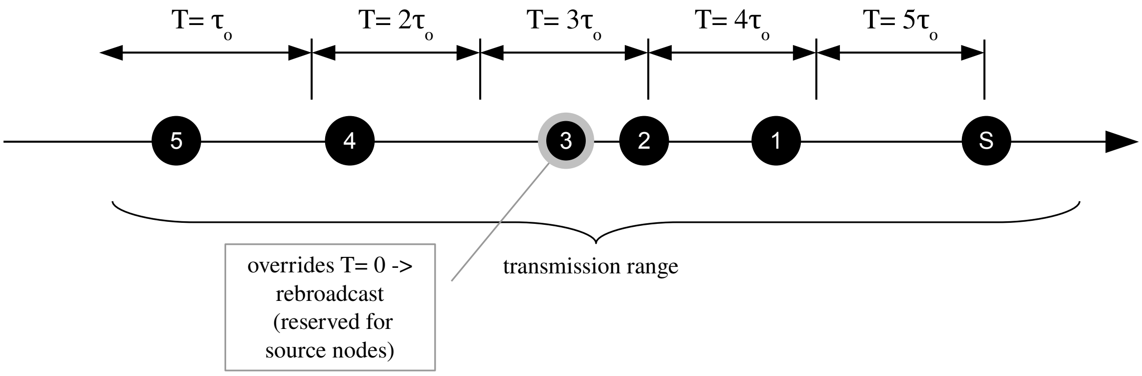


Figure 5.8: Distance-aware Flooding by means of Slotted 1-Persistence with Source Node Priority.

Of course, a conflict can arise when at the same moment multiple nodes are source nodes and all transmit in slot 0. The microSlotting scheme ensures the most remote source node gets priority and will broadcast first. Multiple Source Nodes can each transmit their packets after one-another. Some information will be lost, as the receiving nodes will only consider the first packet they receive because the flood IDs are the same but their last sample added may differ.

Source Node Priority. – the allocation of slots in the microSlotted 1-Persistence Flooding is slightly altered. The first slot is reserved for Source Nodes, nodes which have added a sample to the TrafficMap. All other nodes are scheduled in subsequent slots as defined by Equation (5.6). □

Equation (5.2) can now be altered to incorporate the Source Node Priority:

$$S_{ij} = \begin{cases} 0 & \text{if a sample has been added;} \\ 1 + N_s\left(1 - \left\lceil \frac{min(D_{ij},R)}{R} \right\rceil\right) & \text{otherwise.} \end{cases} \tag{5.6}$$

The distance-aware flooding mechanism presented in Figure 5.7 can now be adapted to use SNP, as shown in Figure 5.9. The flooding scheme functions the same, with the exeption of a check if a sample has been added. If the external estimator has added a sample to the TrafficMap broadcast will follow immediately. In case no sample has been added a timeslot is chosen and the node waits for its time to rebroadcast, default behaviour similar to the scheme without SNP. In effect, SNP allows a rebroadcast to bypass the time it has to wait based on the flooding scheme when a sample has been added. This behaviour is expressed by Equation (5.6).

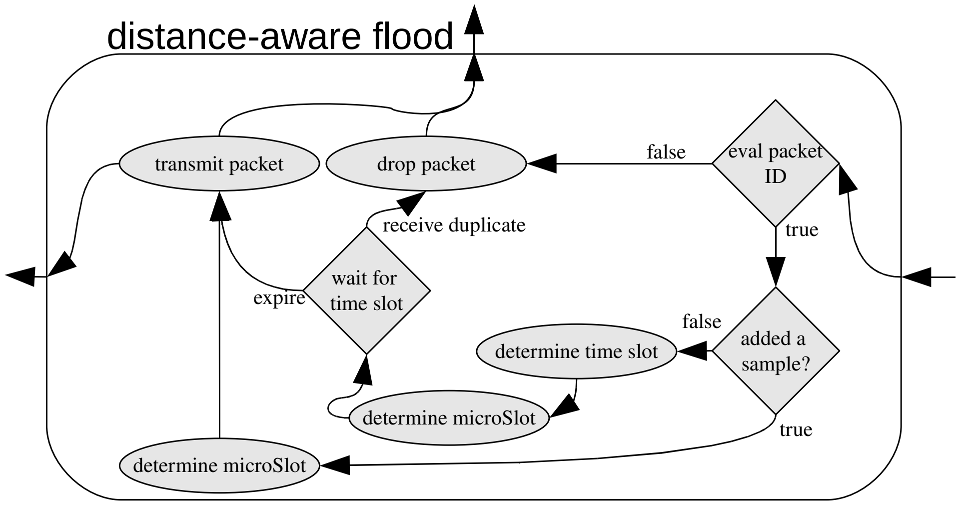


Figure 5.9: Flooding based on distance, prioritises source nodes by means of transmission in slot 0.

The effects of the addition of Source Node Priority to the Flooding scheme is evaluated in Section 6.4.

5.3.6 Timing of TrafficMap Floods

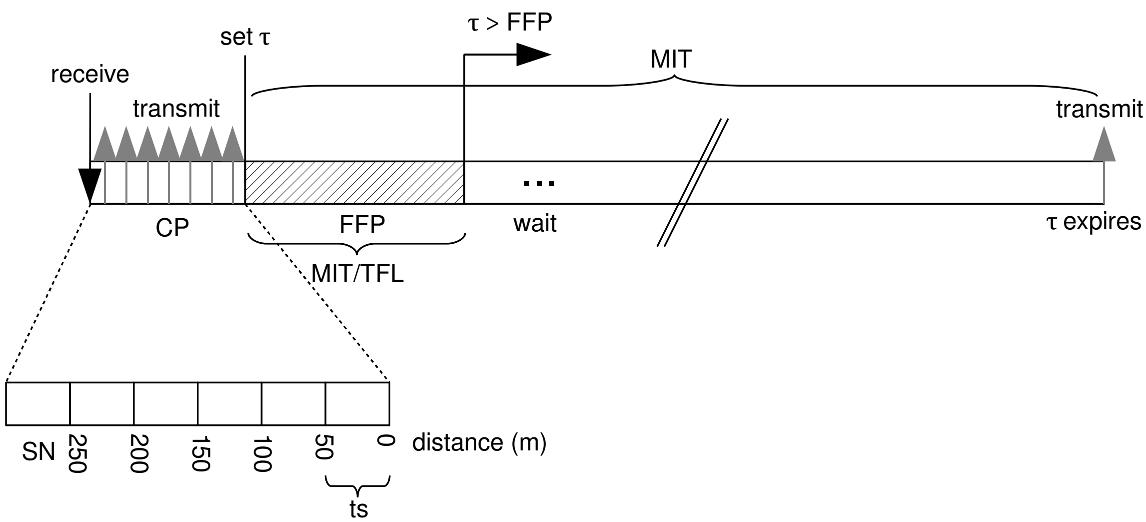


Figure 5.10: Timing of a rebroadcast. After rebroadcast a Flood Free Period (FFP) follows. During the Contention Period (CP) nodes wait for the first rebroadcast to take place. Source Nodes (SN) have priority over Relay Nodes, and can claim timeslot 0. After τ has expired a transmission *must* be executed. τ is reset after a transmission.

To prevent that too many TrafficMap floods exist simultaneously some nodes will have to refrain from rebroadcast. Incoming floods will be "buffered" (for a maximum of MIT seconds). In figure 5.2 this is referred to as "collect and summarise". This can be implemented in two ways:

- Allow a maximum of TFL transmissions per MIT, the TrafficMap Flood Limit. A counter will keep track of the number of broadcasts. This can result in many broadcasts in rapid succession until the quotum has been met. Updates in de remaining time will have to wait for the counter to be reset. This can potentially result in long delays of vital information.
- Introduce an FFP: Flood Free Period (e.g. $FFP = \frac{MIT}{10}$) after every transmission. During FFP a node may not pass on or inject floods, information will remain buffered. After every transmission the τ timer will be reset; transmission will only take place if $\tau > FFP$.

A relation follows from these two approaches:

$$FFP \times TFL = MIT. \tag{5.7}$$

MIT is proposed to be set to $3s$ in Section 5.1.2 based on some estimations. The exact value for MIT will have to be evaluated based on more thorough (field) studies. From the relation presented above it follows that a longer FFP increases the potential delay of a flood if it rapidly follows a previous flood but reduces the number of transmissions per MIT.

5.3.7 Configuration

The configuration arguments of the dissemination strategy (with the modified version of the slotted p-Persistence Flooding scheme) are presented in Table 5.1. As explained in Section 5.3.6 MIT, TFL and FFP are important to ensure lower and upperbounds to the frequency with which a TrafficMap flood is generated. This should occur often enough to swiftly disseminate the required information but not too often so as to leave the medium free for other applications.

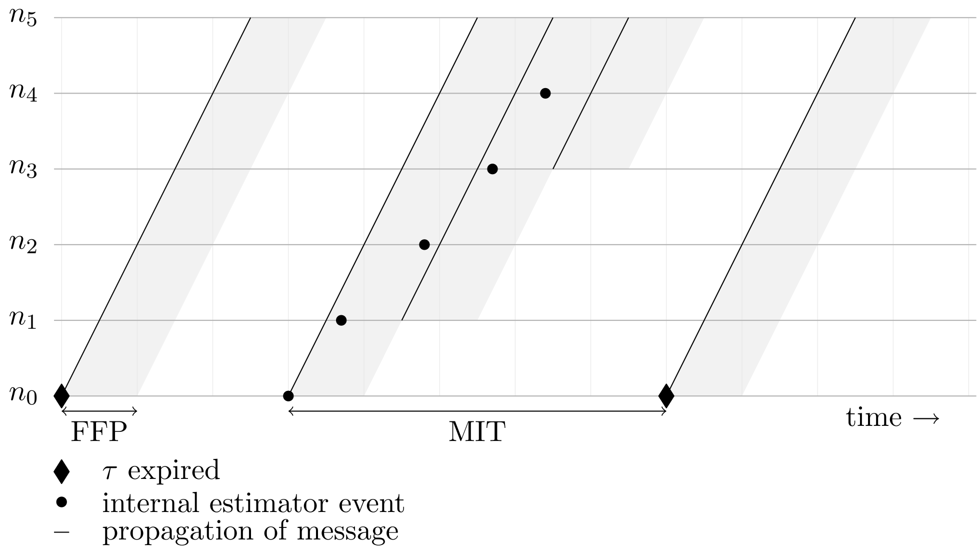


Figure 5.11: Timing of transmissions. A MIT is $5 \times FFP$, $TFL = 5$. The internal estimator in nodes $[0 \ldots 5]$ triggers an event, broadcasting the modified TrafficMap message will be delayed until after the FFP.

Parameter	Description
MIT	Maximum Inter-TM Time, ensures periodic messaging.
TFL	TrafficMap Flood Limit: maximum number of TrafficMap floods per MIT. 'Collect and summarise' will occur when the number of received TrafficMap floods exceeds the TFL.
FFP	Flood Free Period, period after transmission in which nodes in the transmission range are not allowed to broadcast. $\frac{MIT}{FFP} = TFL$, See Figure 5.11.
R	The estimated transmission range.
N_s	The number of slots. In [121] it is said that N_s theoretically depends on ρ_{local}, the (local) traffic density. A more dense network will require more slots. A problem is determining ρ for the entire network under all circumstances. An alternative presented in [121] is to make N_s depend on the time of day (more slots during rush hour).
N_{ms}	The number of microSlots, based on the assumption that one microSlot should contain in general only one vehicle
slotSize	The geographical projection of nodes that fall within the same timeslot. To be obtained from: $\frac{R}{N_s}$.
slotTime ts	The duration of one slot. Defined as MAC processing + propagation delay.
ρ	Number of instrumented vehicles per km of road.
ρ_{local}	The local density of instrumented vehicles, the number of peers within transmission range. If ρ increases more vehicles will occur in a slot slot and more slots will be required. $\rho_{local} = \rho \times R$.

Table 5.1: Configuration Parameters of the TrafficFilter

lane	ρ sparse	ρ dense
left	20	140
center	19	126
right	14	87

Table 5.2: Traffic densities measured in vehicles per kilometer of highway measured on the A5 in 1992, split in lanes. The fact that more vehicles are measured in the left lane can be explained by a larger percentage of long vehicles in the middle and right lane (1%, 10% en 40% respectively) [57])

The efficient flooding is central to the dissemination strategy and relies on estimates and assumptions which are based on the state of traffic. R, N_s and N_{ms} determine the performance of the microSlotted 1-Persistence Flooding. These values need to match the average vehicle density per kilometer ρ but may also be derived with finer granularity based the local density.

In this research ρ is used as a simple average to derive the number of vehicles on the road (as expressed by Equation (B.1) on page 137. It is argued that deriving a good estimate of the number of interferers will be very important in practice if the parameters are to be set dynamically.

An absolute upper limit to the number of vehicles per lane per kilometer is in the order of $1000/5 = 200$, vehicles will be bumper-to-bumper. Research performed by Kerner [57] reports traffic densities on a German highway as presented in Table 5.2. The values for the sparse column are somewhat arbitrary as sparse density can be as low as 0 vehicles per km but the values in the dense column can be used as an indication for a maximum value of the density. In [70] May classifies traffic densities in the U.S. with letters A through F. Traffic is of Type

F when densities are between 67 and 250 vehicles per mile per lane. This corresponds to less vehicles per kilometer (156) than the maximum derived above (200), but is in the same order as the value found by Kerner (140).

5.4 Trustworthiness of the TrafficMap Information

Possibly hundreds of vehicles could be involved in the construction and dissemination of TrafficMap information. These vehicles could introduce errors (either deliberately or by accident) into the TrafficMap. The Averaging function (Section 4.3.2) was designed to make a sample representative to more than one vehicle. Nonetheless, a system like the TrafficFilter needs to be thoroughly analysed for security holes.

In [121] it is mentioned that the distance to other vehicles can possibly be derived from the Received Signal Strength Indication (RSSI) when there is no or poor GPS coverage. Because a position is entered into the TrafficMap it can only be broadcast when the own position is known.

Furthermore, it is reasoned by Friederici and Gerlach in [34] that RSSI could be used for plausibility checks of a reported position. This is in line with the idea of sensor fusion; if data from one sensor cannot be trusted its quality can be judged by means of a second—preferably independent—measure. Correlating RSSI to a position reported by a vehicle may help in judging whether the reported position is correct and the received information can be trusted.

An incorrect position could result from poor GPS coverage in the transmitter or a deliberate error introduced by an attacker. It was concluded in [34] from field experiments that due to multipath and fading effects a direct RSSI is a poor indicator for the distance to the sender, but it could be part of a more sophisticated security system to estimate the trustworthiness of received information.

Chapter 6

Evaluation of the System

In the previous Chapter the design of the TrafficFilter was discussed. Based on this description a simulator has been implemented in OMNeT++, which is described in detail in Appendix B. The simulator was instrumented to measure several performance metrics and evaluate the performance of the system under various circumstances. This is done by means of the Performance Metrics as described in Section 6.1.

The system will be evaluated in three steps. First, in Section 6.2, we will evaluate the performance of the proposed changes to the Slotted 1-persistence flooding scheme as designed by Wisitpongphan *et al.* in [121] to see if the addition of microSlots has the desired effect. This scenario will not feature mobility, because we want to study the performance of the flooding schemes in isolation.

Then, in Section 6.3 we will discuss the effects of mobility on the performance of the dissemination of TrafficMap messages. This experiment will consider dissemination using the microSlotted 1-persistence flooding scheme on a static network and on a network which is mobile according to the Intelligent Driver Model.

In Section 6.4 the contents of the messages will be inspected. These messages contain information which constitutes the over-the-horizon view. The correct operation of the TrafficFilter system can ultimately be evaluated by the relation between the TrafficMap contents and the actual situation on the road, this relation should be strong in both the spatial and the temporal domain. This scenario will test the addition of SNP to the microSlotted 1-persistence flooding scheme.

Finally this chapter closes with some remarks on the analysis performed in Section 6.5.

6.1 Performance Metrics

We will now introduce the metrics which are measured during simulation and explain how this is done. These metrics give insight in the actual operation and can be used to draw conclusions with respect to the performance of the system.

6.1.1 Simulation Scenarios

The simulation study consists of 50 runs for every traffic density value. The simulations compare two alternatives for every scenario presented in this chapter. In Section 6.2 two Flooding Strategies will be compared in the absence of mobility. In Section 6.3 the effects of Mobility will be researched and finally, in Section 6.4 SNP, a feature to possibly improve the quality of the TrafficMap information will be researched. Every simulator run consists of 100 floods. The resulting mean values and their 95% confidence intervals are given in the plots, although the confidence intervals do not show up in every plot because the intervals are too small.

The simulation model reflects the situation on a one-lane road of 10 kilometers in length where the density is varied between 10 vehicles per kilometer and the theoretical maximum vehicle density of 200 for a bumper-to-bumper scenario [70], or 125 vehicles per kilometer in the last two scenarios (Sections 6.3 and 6.4). When nodes are not mobile they are distributed with a uniform inter-vehicle distance:

$$\left[\frac{1}{2}average\_spacing, 1\frac{1}{2}average\_spacing\right] \tag{6.1}$$

where the average_spacing is derived from

$$average\_spacing = \frac{roadLength}{numVehicles} - vehicleLength \tag{6.2}$$

and

$$numVehicles = \rho \times roadLength. \tag{6.3}$$

for a given road of length *roadLength* and traffic density ρ. This node placement strategy guarantees a fully connected network, i.e. no gaps exist between clusters of vehicles larger than $1\frac{1}{2}$ times the inter-vehicle distance defined by Equation (6.2). In order to guarantee such a fully connected network the inter-vehicle distance can never be larger than 250m (i.e. the transmission range used in this research). This results in a minimum ρ of 4 vehicles per kilometer.

When mobility is involved in the experiments nodes move according to the Intelligent Driver Model as discussed in Section 3.5.1. The model reflects a circular road of 10km in length with a reduced speed zone between 4000 and 6000 meters. For more information on the implementation of the IDM in OMNeT++ have a look at Appendix B.3.

6.1.2 Reachability

Reachability is of prime concern for the TrafficFilter: the information is targeted at all vehicles on the designated section of road. If a dissemination scheme is not able to reach a high number of nodes it has failed its primary task: disseminate the information over a long distance.

Often reachability is expressed as the number or percentage of nodes in the network which receive a message. In this research we express reachability as the degree to which floods fully propagate from one end of the road to the other, from which we can conclude the former definition because vehicles are arranged on a line. If end-to-end propagation is successful, we can conclude the entire road has been covered and all nodes in the network have received the message.

Reachability is measured by counting the number of floods launched on the far side of the road and deriving which percentage of these floods makes it to the other end. Partial propagations are not counted.

6.1.3 Delay

It is important the information that is available in a vehicle is up-to-date. One of the factors that influences how up-to-date this information is, is the time spent propagating this information. After information is produced we want to pass it on via multi-hop communication to vehicles several kilometers away. The aim is to do this as fast as possible (i.e. the delay is low) to preserve the freshness of the information and enable rapid response to sudden changes.

The delay is measured in the simulator as the time between the initiation of a flood (by the vehicle closest to the 10km point) and reception by the vehicle closest to the 0km point. In this sense, the delay expresses the time it takes a message to travel 10 kilometers.

6.1.4 Hop Count

Multi-hop communication is based on nodes passing information to other nodes. However, every hop incurs costs in resources such as time and medium utilisation. If processing and transmission delay are constant factors we need to limit the number of hops required to traverse the network. As such, the number of hops is a very important indicator.

The number of hops is measured by a counter in the TrafficMap message which is flooded. If a node rebroadcasts a previously received message, it does so with the hop count increased by one. When the flood reaches the vehicle closest to the 0km point the hop counter is extracted from the message. In a sense, the message keeps track of the number of hops it takes to traverse 10 kilometers.

6.1.5 Transmissions and Receptions

Every node in the network holds a transmission counter and a reception counter. Whenever a node transmits (i.e. a message is handed over to the MAC layer for transmission) the transmission counter is increased by one. Note that a transmission attempt is not necessarily successful. As such it counts the number of transmissions performed by the Broadcast state in Figure 5.2 on page 76. Whenever a node's Network layer receives a correctly received message from the MAC layer its reception counter is increased by one. At the end of the simulation run these counters are obtained from all nodes and averaged for the 100 executed floods. The resulting values are then averaged for the number of nodes as expressed in Equation (6.4).

$$\#transmissions = \frac{\Sigma_{i=1}^{n_j} \#\mathrm{TX}_i}{n_j} \tag{6.4}$$

Here n_j is the number of nodes in run j. The number of transmissions and receptions in node i in run j are $\#TX_i$ and $\#RX_i$ respectively. The number of receptions is calculated likewise:

$$\#receptions = \frac{\Sigma_{i=1}^{n_j} \#\mathrm{RX}_i}{n_j} \tag{6.5}$$

This results in an average number of transmissions and receptions per run. The mean of 50 runs and the 95% confidence intervals are shown in the results.

6.1.6 Overhead

Overhead is a metric derived from the number of transmissions and receptions. It gives a general insight in the efficiency of the system under test. The overhead is calculated as $\frac{\#transmissions}{\#receptions}$ per node which is later used to calculate the mean for all nodes per run as shown in Equation (6.6). If one transmission yields many receptions, overhead is low.

$$\frac{\Sigma_{j=1}^{r}\left(\frac{\#transmissions}{\#receptions}\right)}{r} \tag{6.6}$$

The number of runs is expressed as r. This results in an average overhead per simulation run. The mean of 50 runs and the 95% confidence intervals are shown in the results.

If one transmission yields one reception overhead is 1. If one transmission yields multiple receptions overhead is lower (e.g. 1/12). Note that transmissions which result in collisions are counted in numTrans, but are not counted in numRecv if no successful reception follows. Thus the overhead forms a measure dependent on the number of nodes within range and the number of collisions.

6.1.7 Medium Utilisation

Next to the transmission and reception counters nodes also count their total time spent receiving information. To this end the physical layer module SnrEval in the Mobility Framework (illustrated in Figure 6.1) has been instrumented to sum the durations of all received noise during a simulation run. This is prior to deciding whether the received noise is a correct frame, and also includes own transmissions. The average of the medium utilisation taken over all nodes is used as input to calculate a mean value and derive confidence intervals.

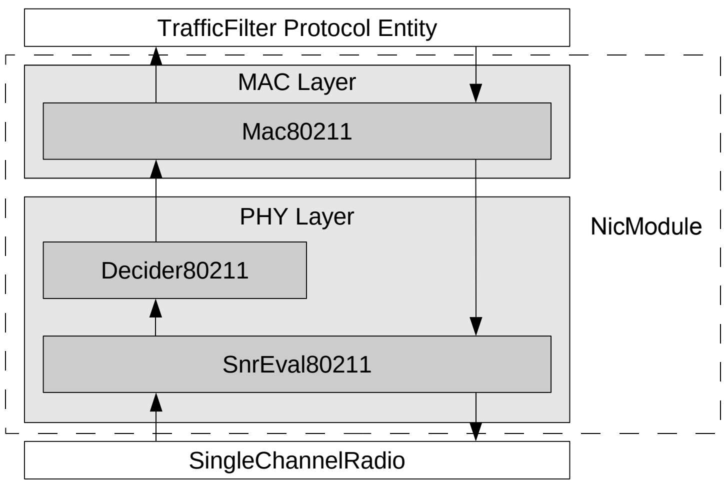


Figure 6.1: The NIC module

The Medium Utilisation gives insight into how many resources of available radio spectrum are used. It does not say anything about whether these resources are spent to produce a successful transmission or a collision.

6.1.8 Slot Utilisation

The metric slot utilisation is specific for the slotted flooding strategy used in this research. The slotted 1-persistence flooding works by assigning nodes a time slot in which they can perform their rebroadcast. The way the timeslots are allocated determines to a great extent the efficiency of the system. Every node keeps track of the number of transmissions executed in each slot. These numbers can be averaged per run, and then for the 50 runs performed. From the resulting values a slot allocation distribution can be derived. This distribution sheds some light on the internal functioning of the flooding strategy and has effect on all other metrics.

6.2 Comparing the two Flooding Strategies

The slotted 1-persistence flooding scheme was found to be very effective and efficient by Wisitpongphan *et al.* [121]. In order to investigate if the proposed modification in Section 5.3.3 (i.e. the addition of microSlots) is beneficial simulation experiments were executed for both schemes. We will now have a look at the results of the simulation. The parameter vehicle density (ρ) is varied and the two schemes—slotted 1-persistence and the modified version—are both simulated under the same circumstances. As explained in Section 6.1.1, these experiments are executed with fixed nodes.

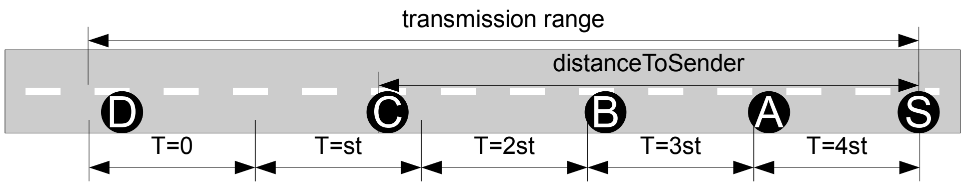


Figure 6.2: Slotted 1-Persistence Flooding with 5 slots. Node D rebroadcasts, A-C refrain from rebroadcasting.

In order to arrive at a realistic transmission range the transmission power was set to 168.98 mW (derived using Friis Free Space propagation formula, see Appendix C.1 for the derivation). This resulted in an interference range of 500m and an effective transmission range of about 250m, which coincides with the rule-of-thumb that transmission range generally is half of the interference range.

We will briefly recap the flooding strategy, referring to Figure 6.2. The slotted 1-persistence flooding uses five slots each of one slotTime period defined as 5ms. The slots also partition the transmission range into five areas, each 50 meters in size. Remember the $min(D_{i,j}, R)$-component in Equation (5.2) on page 79, in effect the slot which contains D (i.e. has the largest distanceToSender), the first broadcast slot, is open to the left.

6.2.1 Reachability

This metric, the percentage of the floods launched by the lead vehicle that eventually make it to the trailing vehicle, is an indication of the percentage of stopped floods. Intuitively it is clear that at low densities there exists a probability that a flood is not propagated due to a gap in the network, the vehicles are simply too far dispersed for radio transmission to bridge the gap. To exclude incomplete propagation due to gaps we only use vehicle densities to guarantee a fully connected network. This enables the study of detrimental effects on propagation brought on by collisions.

Not fully propagated floods fell victim to collision. What can be observed from the results is that the reachability of the slotted scheme rapidly drops and seems to converge to a value around 13%. This can be explained by that some floods will still get through. The microSlotted scheme maintains a delivery ratio of around 100%, only at $\rho = 200$ does it drop to 97%. The reason why the microSlotted scheme does not maintain 100% delivery is because the probability a flood will not fully propagate—however minute—is still present (the dips at 30 and 50 are 98.81% and 98.855% respectively).

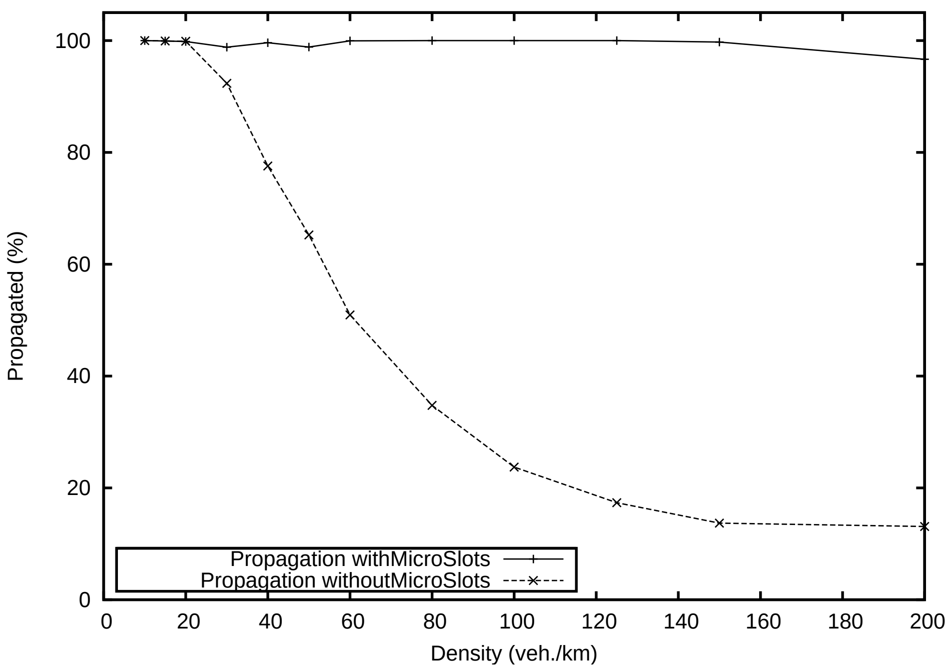


Figure 6.3: Flood propagation percentage

6.2.2 Delay

The microSlotted scheme relies on adding a small extra delay to the wait time defined by the slotting scheme. This delay is in the order of [0-9] DIFS periods (0-0.000522s)[1] per hop. This results in a tiny—though not insignificant—extra delay for the modified scheme, visible in Figure 6.4 for densities of 10 and 15 vehicles per kilometer. As the vehicle density increases beyond 20 vehicles per kilometer the two schemes clearly behave differently; the original scheme shows an increasing delay as the number of nodes increases, while the modified scheme exhibits a diminishing delay. This behaviour of the slotted scheme can be attributed to an increasing number of collisions in the first slot for the original scheme, which means a rebroadcast in the next slot is executed one slotTime later. If that broadcast collides a slotTime later the nodes in the next slot get a chance to broadcast, until a successfull broadcast occurs.

At low densities a high delay is observed. This is due to the fact that, with fewer nodes, the nodes are not always at optimum distance from each other. For instance, it could very well be there are no vehicles in the first three slots but there is one in the fourth. This would incur three slotTimes of delay just for this hop, resulting in a 15ms penalty to end-to-end delay. With increasing density it becomes increasingly more probable there is a vehicle in the extremes of the estimated transmission range, resulting in covering a large distance per hop and thus covering the full 10km in fewer hops. Both contribute to a lower delay. The first because rebroadcast takes place immediately in the first timeslot, the second because fewer hops are needed altogether. Beyond 80 vehicles per kilometer the delay of the microSlotted scheme also starts to grow. This is because even with the microSlots probability of collision increases with the density and rebroadcast by nodes in later slots are required.

The extra delay added by the microSlots is negligible compared to the overall delay. With an average of 50 hops (see the next Section on Hop Count) the delay incurred by the microslots amounts to at most $50 \times 9 \times DIFS = 2.61ms$ whereas the overall delay is in the order of 100ms.

[1] An IEEE 802.11p DIFS is 5.8μs [49]

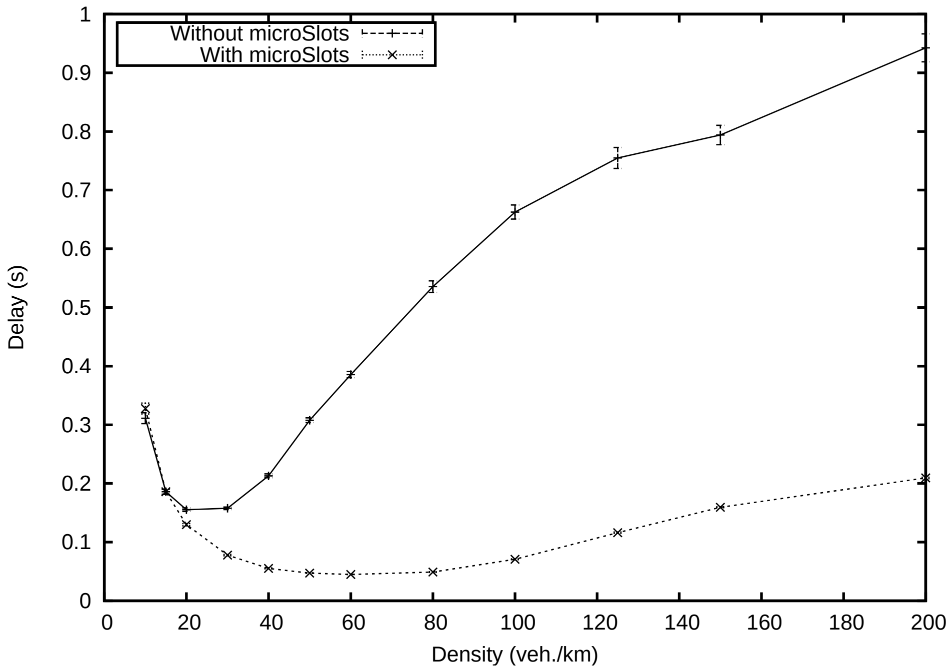


Figure 6.4: Delay mean and 95% confidence intervals

6.2.3 Hop Count

After the density exceeds 15 vehicles per kilometer the probability of slots with more than one vehicle increases[2]. In the slotted scheme this was expected to result in collisions because the vehicles in one slot will be synchronised when performing their rebroadcast, as observed in Section 5.3.4 and described in more detail in Appendix B.5. If a collision occurs in the first slot, the vehicles in the second slot will not discard their rebroadcasts (they have not successfully recognised a rebroadcast by a more distant node) and will rebroadcast when it is their time, as shown in Figure 5.7 on page 80. If, in this slot too, there are more than one node a collision will occur. In the extreme, collisions will occur in all successive slots and the flood will die out.

If a collision occurs in the first slot, a successfull rebroadcast can occur in the second slot, or in the third and so on. This implies that, although a rebroadcast does take place, the geographically covered distance is smaller than the theoretical optimum. As a result more hops are needed to traverse the full 10 kilometers.

For the original scheme, the number of hops increases as vehicle density increases. This can be attributed to collisions. The modified scheme appears to suffer to a lesser degree from an increase in hopcount due to collisions. This has two reasons:

- Because of the microSlots, collisions in the first slot are unlikely because there is less synchronisation, so a rebroadcast in the next slot is hardly needed.
- With increasing density, the probability of a collision within one microslot increases. However if a collision is to occur in the first microSlot the CSMA/CA mechanism of the MAC layer will ensure the transmission scheduled in a later microSlot can still go through, albeit with a slight delay due to backoff.

It is expected that, with the modified scheme, a vehicle density has to become so high as to guarantee multiple vehicles per microSlot before the effects of collisions start to have serious

[2] $\frac{1km}{20} = 50m$, equal to the slot size

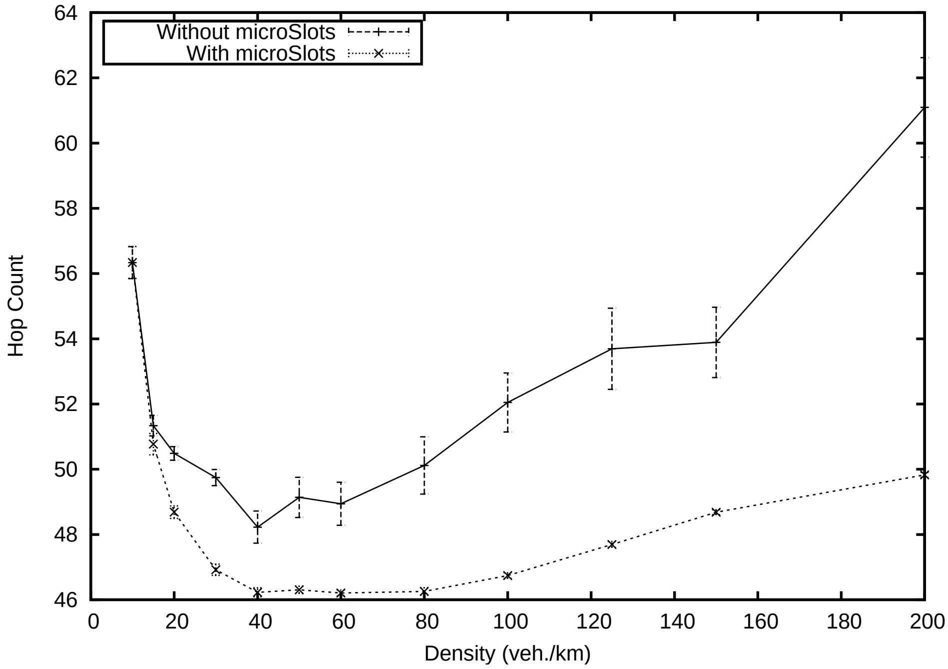


Figure 6.5: Average number of hops required and 95% confidence intervals

effects. With a microSlot size of $\frac{50}{10} = 5m$ this becomes a possibility when multiple lanes have high vehicle densities, such as during rush hour. This effect becomes visible when the density approaches 200. The average spacing here is 5m and the probability of two vehicles being in the same microSlot increases.

6.2.4 Transmissions and Receptions

As the node density increases a single transmission will result in a growing number of receptions, simply because more nodes are within transmission range. When multiple nodes close together transmit simultaneously they cause a collision, this will be counted as a transmission by every node involved (e.g. it consumes "resources" from the medium) but no successful receptions take place.

Figure 6.6 shows that with a growing node density the average number of transmissions per node seems to converge to a value around 30. Interesting to note is that the slotted scheme approaches this value from above, while the microSlotted scheme approaches it from below.

The average number of receptions per node rapidly increases with node density up to $\rho = 50$, because one transmission covers multiple nodes, resulting in multiple receptions.

After $\rho = 50$ the two schemes diverge. This can be explained as follows: a reception is only counted if it's CRC concludes the frame has been successfully received. As noted before, when the density increases, so does the probability of collision, especially with the slotted scheme because more nodes will be synchronised in a single slot. This becomes very clear in the drop after 50. The microSlotted scheme, too, suffers from collisions but to a far smaller degree.

From these results, one can reason that the modified scheme reaches more nodes while using fewer transmissions. It was expected that the microSlots would create more transmissions because several transmissions will be serialised by the MAC layer in stead of performed in parallel (resulting in collisions). For both schemes, the number of transmissions to be carried out in the first slot is still the same but with the microSlotted scheme there will be a reduced need for

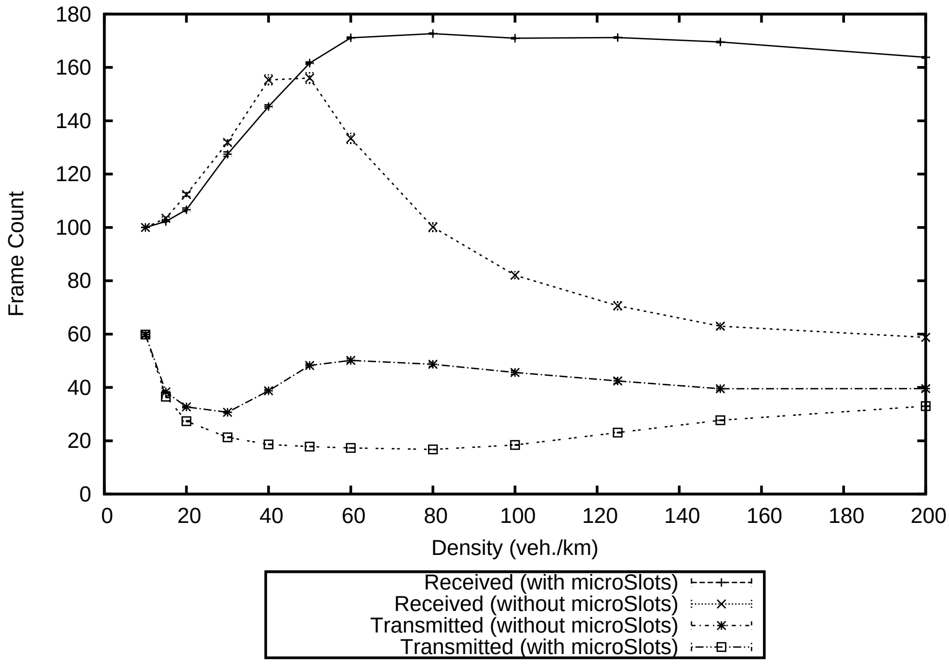


Figure 6.6: Transmissions and receptions

transmissions in later slots. This effectively compensates for the serialised broadcasts which are possible with the microSlotted approach.

6.2.5 Overhead

We have defined overhead as the number of transmissions per reception, or the fraction $\frac{\#transmissions}{\#receptions}$ based on the average number of executed transmissions and receptions per node for a given ρ.

The simulation results shown in Figure 6.7 indicate that the modified scheme seems to use the medium more efficiently, because it achieves more receptions with fewer transmissions. In effect, a higher reachability is realised with a lower medium utilisation. The difference between the two schemes can be attributed to the higher number of collisions (and hence a higher number of transmissions and lower number of receptions) of the slotted scheme.

6.2.6 Medium Utilisation

The collisions caused by the slotted scheme result in a high medium utilisation, for every flood many rebroadcasts will not be successful. Between 100 and 125 vehicles per kilometer the two lines cross as the slotted scheme assumes a more or less horizontal trend and the microSlotted scheme keeps rising. The reason why the slotted scheme levels can be explained by the fact that many floods no longer fully propagate. As a result the average medium utilisation per node is lower because many nodes (those furthest away from the initiator of the floods) observe no transmissions at all.

The microSlotted scheme uses the available resources more efficiently but still obtains a higher average medium utilisation. This can be attributed to the nearly 100% end-to-end propagation and the fact that collisions occur. Added to this is the redundancy incurred by messages scheduled in the MAC layer and serialised by the CSMA/CA backoffs.

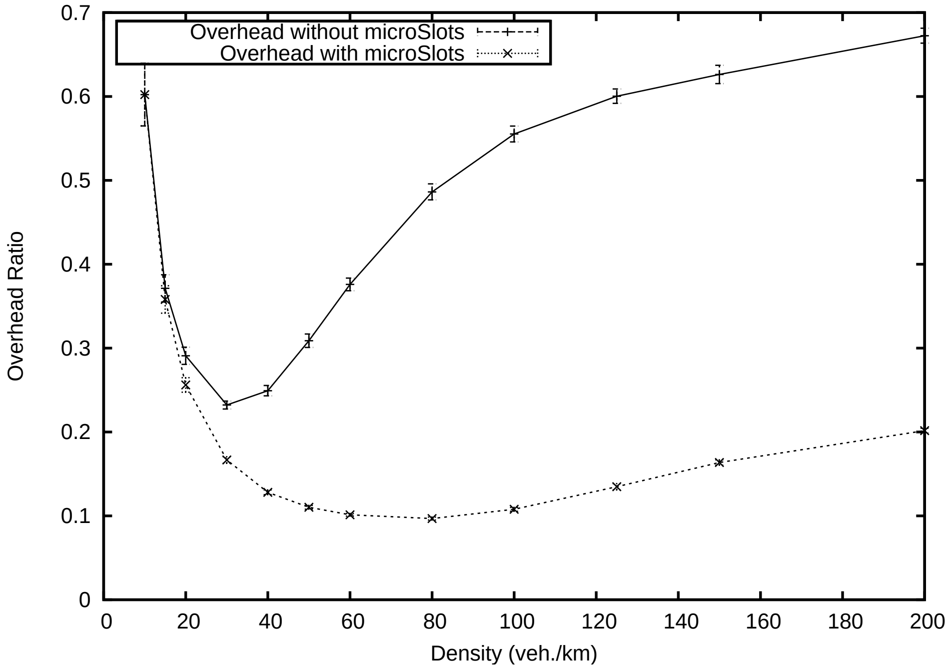


Figure 6.7: Overhead

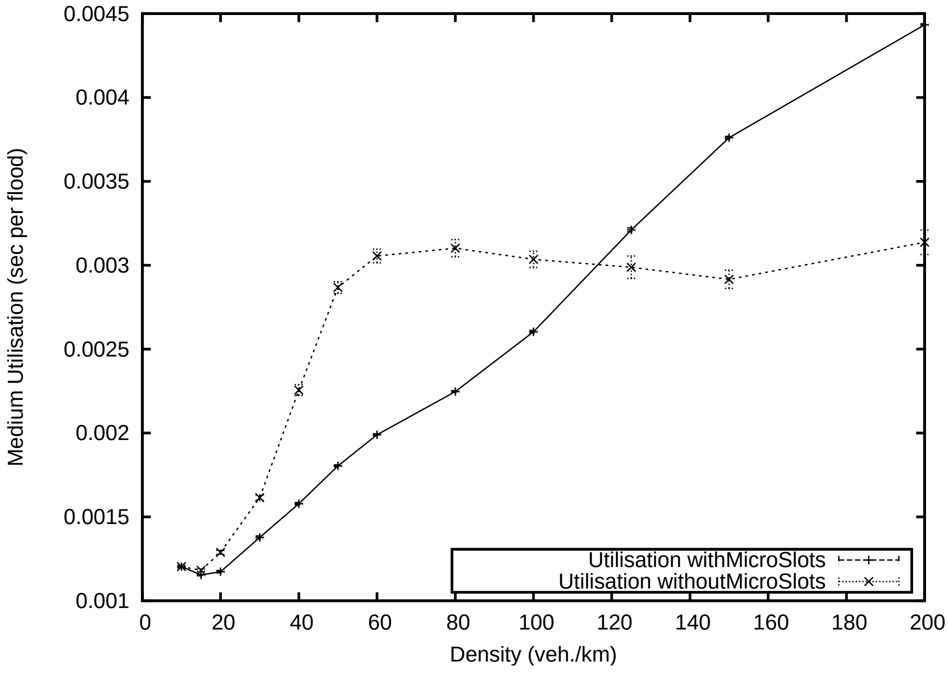


Figure 6.8: Medium Utilisation

6.2.7 Slot Utilisation

As noted in Section B.5, choosing the correct estimate for the transmission range is of great impact on the efficacy of the flooding scheme. Intuitively it is clear that, in order to achieve a low end-to-end delay, it is paramount the geographical footprint of the first slot is correctly aligned with the most remote receivers of a transmission.

The flooding scheme, then, must see to it that most of the successful transmissions are carried out in the first timeslot whenever a capable node is present. This can become a challenge when more nodes exist within this single slot. The behaviour discussed in the previous sections on performance metrics is reflected in the distribution of the slot utilisation, as depicted in Figure 6.9.

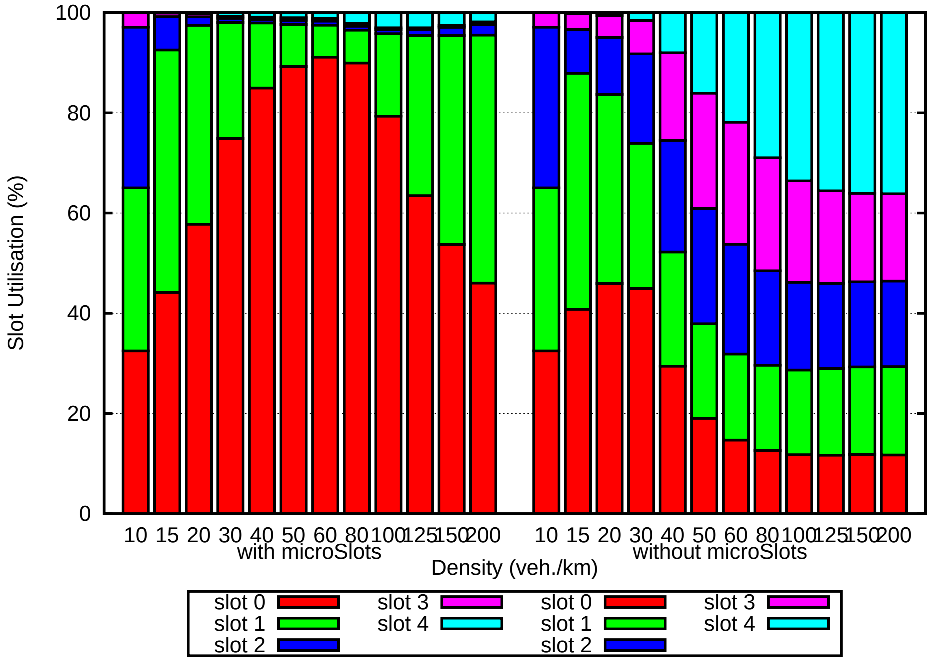


Figure 6.9: Slot Utilisation - shares of slots [0. . . 4] are stacked.

At 10 nodes per kilometer roughly 33% of the transmissions is carried out in slot 0 while another 32% occurs in slot 1 and 2. Because the probability for a slot to contain multiple vehicles is low, this results in the same behaviour for both schemes. The reason why transmissions occur in slot 1 or 2 and not in slot 0 is quite simple, there exists a probability there simply is no vehicle present in the geographic area covered by slot 0. With more vehicles per kilometer it becomes more likely a vehicle occupies the area covered by slot 0. Because of the uniform distribution of the nodes, every slot is equally likely to have a vehicle in it.

The interplay of the probability of correct reception (w.r.t. bit errors in the packet) and cumulative probability a node is present in slot n but none is present in previous slots results in an approximately even share for the first three slots.

As vehicle density increases it becomes clear more transmissions are scheduled in later slots when using the slotted scheme. This is because a growing share of the transmissions per slot collides so the rebroadcast task is handed over to vehicles in later slots. This explains the increase in delay, the increase in hops required to cover the full 10km and why some floods do not make it across the entire road at all. These latter floods suffer consecutive collisions in all slots.

The microSlotted scheme, however, shows an increasing utilisation of slot 0 up to around 60 vehicles per kilometer. The microSlotted scheme has a low probability of collision within one slot because nodes are less synchronised. As a result when a transmission does not succeed in slot 0 it is likely to succeed in slot 1. This explains the low utilisation of slots 2-4.

In the case of the slotted scheme it was expected to get a uniform distribution of slot utilisation: nodes in slot 0 collide, slot 1 collide, . . . , slot 4 collide. This results in a chain of collisions which ultimately terminate the propagation of a flood. However, this is not the case. Once a node schedules a transmission in a certain slot this is logged. Nodes are uniformly distributed, so it can be assumed that in every slot there is an equal number of nodes. Reasoning backward, the expected uniform distribution of slot utilisation suggests that every slot also has an equal number of rebroadcasters.

From the results, this obviously is not the case. Note that, in Figure 6.9, the share of slot 4 for the original scheme rises to around 38% as ρ increases. It is reasoned this can be attributed to the probability of occurrence of the event that triggers the rebroadcast, namely the succesfull reception of an earlier broadcast. As distance to the transmitter increases, probability of correct reception diminishes. It could very well be that all nodes in the closest slot receive the message (slot 4) while only a small percentage of the nodes in the most remote slot (slot 0) receive the message.

This would in turn limit the number of interferers in slot 0, which could also be beneficial. This latter effect may be responsible for the fact that a certain percentage of floods still tends to come through, even at high node densities. It should be noted that this effect is highly dependent on the propagation model used in this research (the standard Free Space Propagation model which is standard in the Mobility Framework). And on the uniform distribution of nodes.

6.2.8 Discussion

Overall, the addition of microSlots to the slotted 1-persistence flooding scheme is a great improvement to efficiency. Delay and the number of hops are lower and the probability a flood will make it from end to end is significantly larger than with the original slotted scheme. It can be argued that serialising the transmissions in the MAC layer because of the microSlots results in a larger medium-busy-time than performing transmissions in parallel, but this is offset by a reduced need for retransmission attempts in later slots.

The slotted 1-persistence flooding scheme seemed like a good candidate to flood TrafficMap messages. With the addition of the microSlots it is expected to be even more efficient. In the remainder of this research the microSlotted 1-persistence scheme will be used as part of the TrafficFilter system.

It should be noted that we did not search for the optimal number of slots or varied the transmission range. It could very well be that a more optimal configuration of slotTime, numSlots (and hence geographic slotSize) and number of microSlots exists. This largely depends on ρ, the density of traffic, and the realistic transmission range achievable when using IEEE 802.11p in a real highway environment. While 802.11p still has Draft status no standard-compliant hardware is available and no real fieldstudies can be performed to derive a propagation model based on empirical evidence. To cope with this we assumed a transmission range of 250 meters, even though the Draft specifies a range up to 1km [49, 115], as such these results can, especially with respect to delay and number of hops, be expected to be on the conservative side.

The microSlotted scheme may duplicate messages because they are scheduled within the same slot and then serialised by the CSMA/CA in the MAC layer. Although this seems to be a waste it also contributes to robustness; if for some reason the first transmission collides the second can still be heard. This duplication, however, does not duplicate the flood: receivers will notice both transmissions are part of the same flood (by means of the Flood ID) and will only propagate the message once, as depicted in Figure 5.7 on page 80.

6.3 The Influence of Mobility on Flooding

In the previous section we have chosen to use the microSlotted 1-persistence flooding scheme because it clearly outperformed the original slotted 1-persistence scheme. Now the simulator is modified to perform two sets of runs:

With Mobility — these runs will use the microSlotted 1-persistence flooding scheme while hosts will be mobile according to the IDM implementation

Without Mobility — this run uses the microSlotted 1-persistence flooding scheme with no mobility

The results will be compared to find out if the flooding scheme also works with mobility. The circumstances will be similar to the previous simulation runs. The mobility is controlled by an implementation of the IDM in each node with the parameters as described in Table 3.4 on page 41. In the scenario with mobility the simulator runs for 300s to allow mobility to stabilise and then 100 floods are executed over a period of 300s. The scenario without mobility is the same as for the microSlotted scheme in Section 6.2.

The maximum density is limited to 125 vehicles per kilometer because beyond that Segmentation Faults start to occur deep in the insides of the Mobility Framework. Although it is open source it was decided not to spend time fixing this problem but in stead settle for a maximum of 125 vehicles because simulation runs with $\rho = 125$ already take a long time. This choice can be justified by the fact that the results from the set with mobility converge to those without mobility, which have already been researched for greater densities in the previous section.

6.3.1 Effects of Mobility on Flooding

On a road with a reduced speed zone halfway it was observed in Section 3.5.1 and particularly in Figure 3.12 on page 42 that after the reduced speed zone traffic disperses and the local density will be lower. It is to be expected that the inter-vehicle distances can very well exceed a node's transmission range, breaking the chain of propagation.

When comparing the two flooding strategies in Section 6.2 mobility was disabled and nodes were placed such that reachability was guaranteed (e.g. the network is fully connected). When a mobility model is introduced that allows nodes to (autonomously) throttle their speeds partitioning of the network can occur, just like in real-life situations.

6.3.2 Reachability

In order to get a dependable over-the-horizon view the underlying communication system must be able to deliver information in the first place. If we aim to construct a 10km over-the-horizon view communication must be able to traverse 10km of road. As can be seen in Figure 6.10 this is not always the case. Here the percentage of floods that traverses the entire road is plotted for every density. It becomes apparent that with a density below 15 vehicles per kilometer gaps occur in front of the head of the jam which cannot be bridged by the transmission range.

As density increases this gap at the head of the jam becomes smaller, and floods propagate the full 10km. This behaviour can also be observed in Figure 6.11(a) through 6.11(f). The reason why at higher densities floods *do* get through is that the density becomes so high that the IDM deems it necessary not to accelerate too much after the reduced speed zone. This is a direct result from the fact that the roads outflow determines its inflow and vice-versa.

The presence of gaps in the network in the scenario with mobility introduces deviations in the results for those densities.

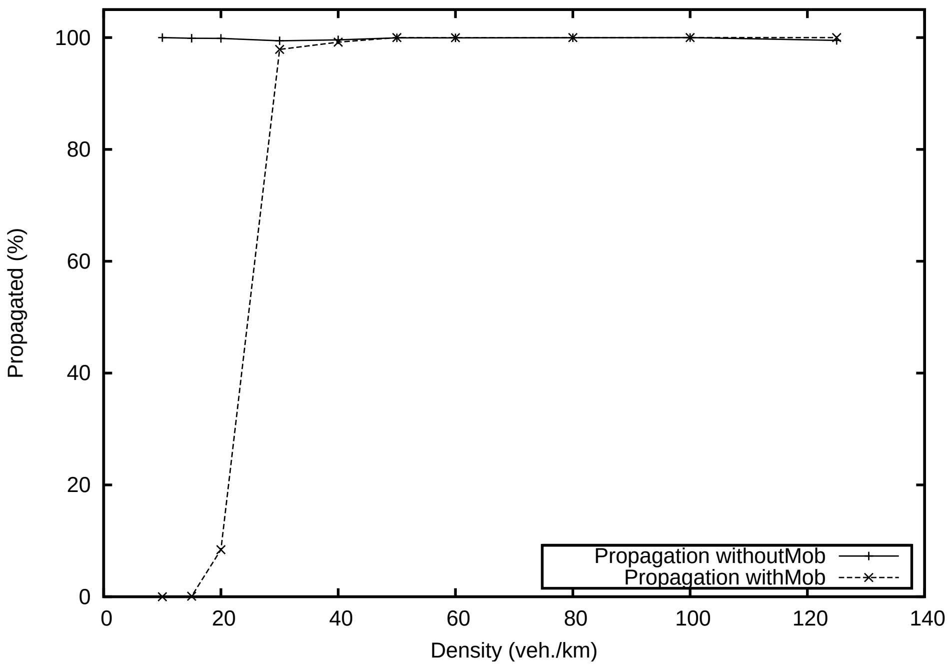


Figure 6.10: Propagation of Floods under mobility.

6.3.3 Delay

The delay measurements of the runs for 10, 15 and 20 vehicles per kilometer are tainted by not fully propagated floods. After the density exceeds 30 behaviour resembles that of the situation without mobility, only shifted down a bit.

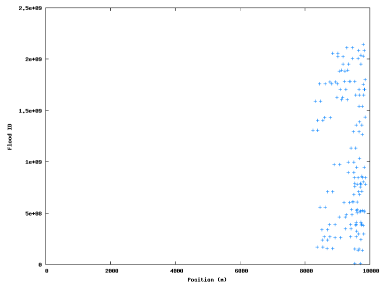
It should be noted that with an increasing density the results for the case with mobility converge to those for the case without mobility for the simple fact that node mobility reduces dramatically as the density increases; this is behaviour of the IDM: shorter headways result in lower speeds. At 100 nodes per kilometer the speed averages around 1.8km/h. At these speeds the network can almost be said to be stationary over the span of one flood propagation:

$$\text{One end-to-end propagation time: } \sim 50ms.$$

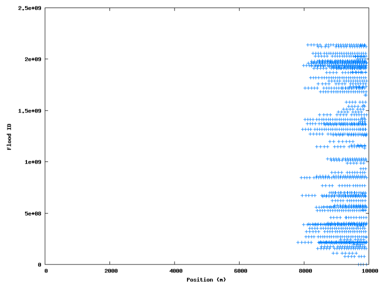
$$\text{Distance travelled at 1.8km/h: } \frac{1.8}{3.6} * 0.05 = 0.025m$$

During one flood the nodes move on average 2.5 centimeters. The increase in delay at 125 nodes per kilometer in case of mobility can be attributed to an increase of the number of hops which result from more collisions: traffic is very dense and hence the microSlotted 1-persistence scheme will synchronise some nodes. After the resulting collision rebroadcast will be executed by a different node, but some time later.

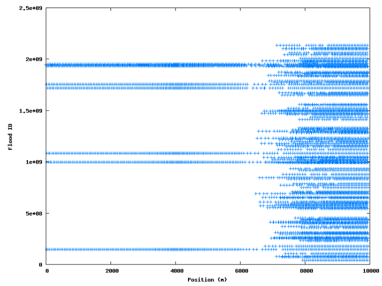
As the density increases speeds decrease in case of mobility. However, the distribution of nodes is different than the uniform distribution used in the case without mobility. Interesting to see is that, between 30 and 100, the delay with mobility is significantly lower than without mobility. This can be attributed to the distribution of the nodes, which in turn results in a different slot allocation - see Section 6.3.7.



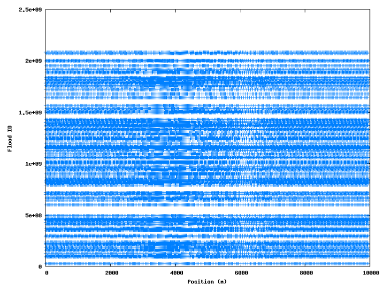
(a) 10 veh./km



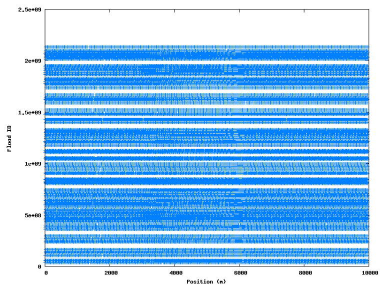
(b) 15 veh./km



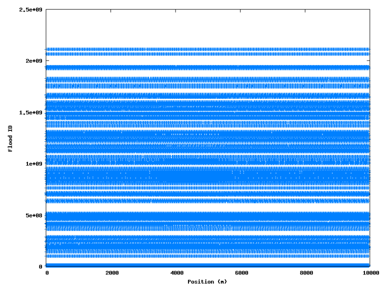
(c) 20 veh./km



(d) 30 veh./km



(e) 40 veh./km



(f) 50 veh./km

Figure 6.11: Propagation of floods per density with distance on the x-axis and Flood ID on the y-axis. A point indicates a node at that location received the flood with that Flood ID (random 32-bit number). The gap created in front of the head of the jam remains clearly visible up to 30 vehicles per kilometer but has no effect on propagation after 20 vehicles per kilometer.

6.3.4 Hops

As noted above, after 30 vehicles per kilometer the behaviour of both situations is somewhat similar. At a density of 125 the mobility seems to slightly increase the number of hops. This can be attributed to a more even (uniform) distribution in the case without mobility and a more clustered distribution in the case with mobility. The very high densities in the jam have an adverse effect on propagation.

Mobility has little effect on the number of hops, as long as the network is fully connected. The differences (sometimes above, sometimes below the reference without mobility) can be attributed to a difference in vehicle distribution.

As noted before, densities below 30 do not guarantee a connected network and measurements are flawed.

6.3.5 Overhead

The mobility has little effect on the overhead of the microSlotted 1-persistence flooding scheme. Ignoring the measurements for 10. . . 20 the overhead shows no significant difference. The number of transmissions is the same in both cases, except when the density exceeds 100 vehicles per kilometer, then more broadcasts are required to cope with mobility. This is reflected in all other metrics, of which the slot utilisation in Figure 6.16 shows clearest what is happening: rebroadcasts increasingly occur in the second slot.

The number of received messages in the case with mobility wanders around that of the static network, confirming the observation that mobility has little effect on the flooding.

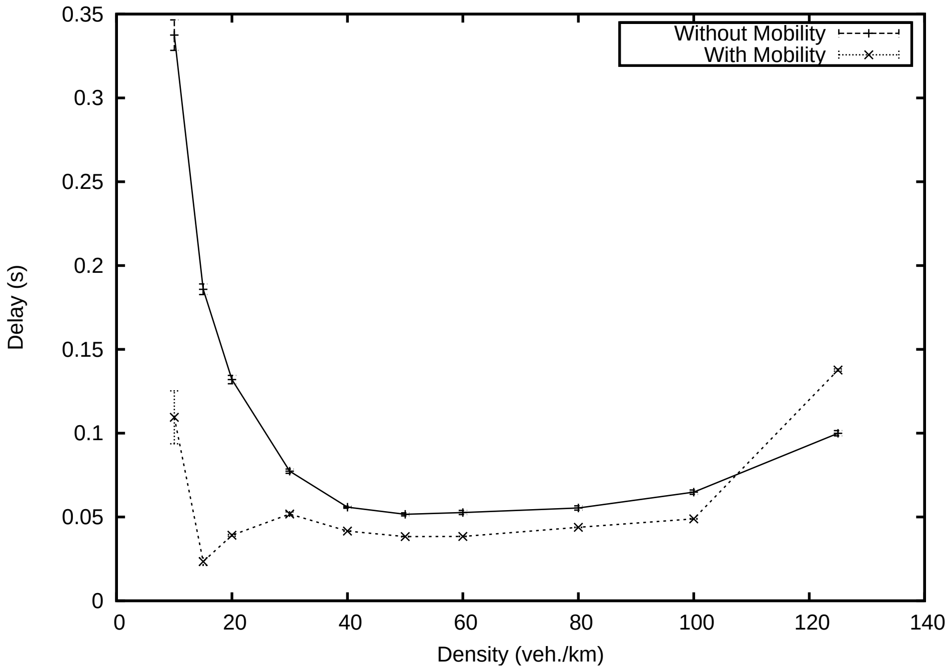


Figure 6.12: Delay

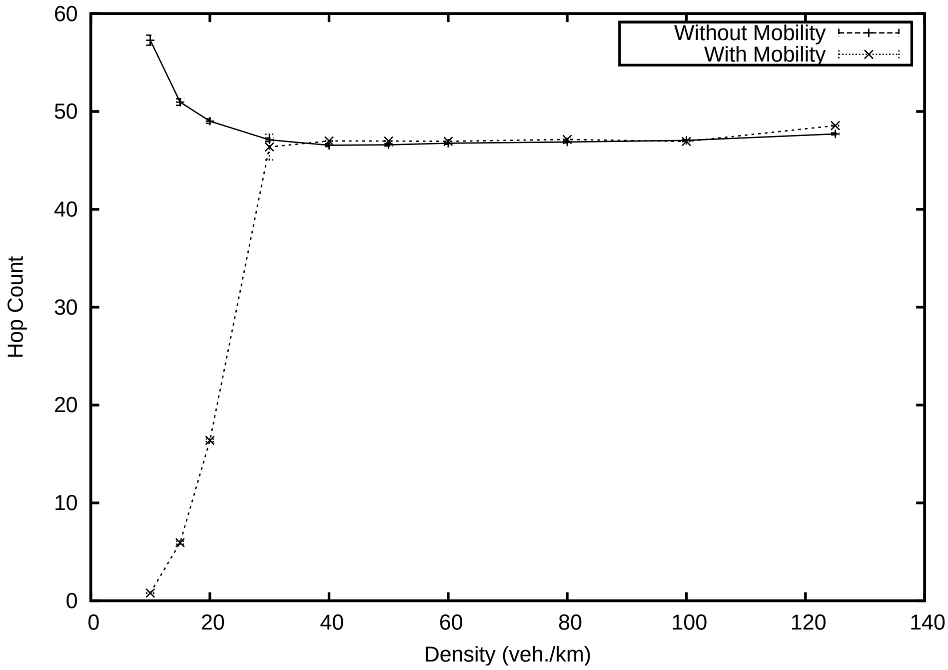
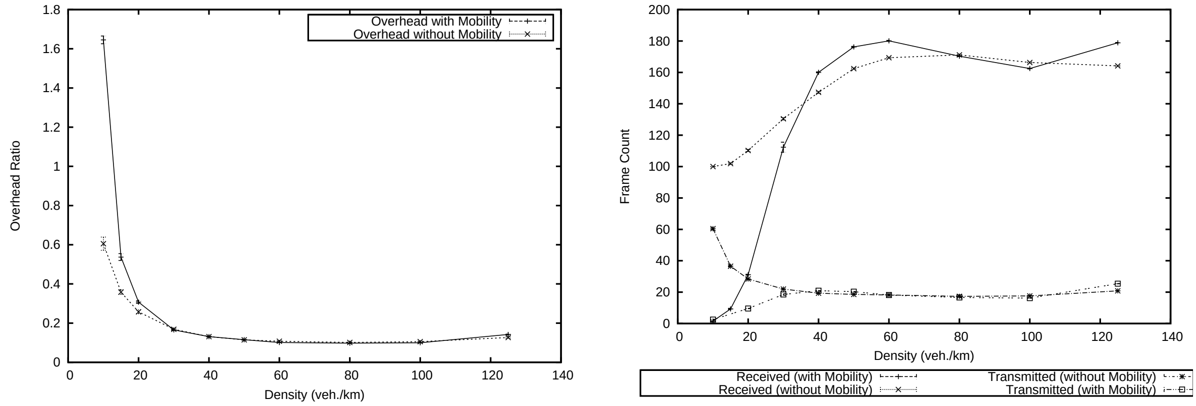


Figure 6.13: Number of hops

6.3.6 Medium Utilisation

Because for the densities [10..20] the network is not fully connected the medium utilisation measurements for these densities in case of mobility are to be ignored. After the density exceeds 30 vehicles per kilometer medium utilisation stays about the same as in the case without mobility, except for the density of 125 vehicles per kilometer: the increase in number of hops results in



(a) Overhead

(b) Transmissions and Receptions with mobility

Figure 6.14: Overhead and Transmissions and Receptions

an increase in transmissions, which in turn results in a larger medium utilisation.

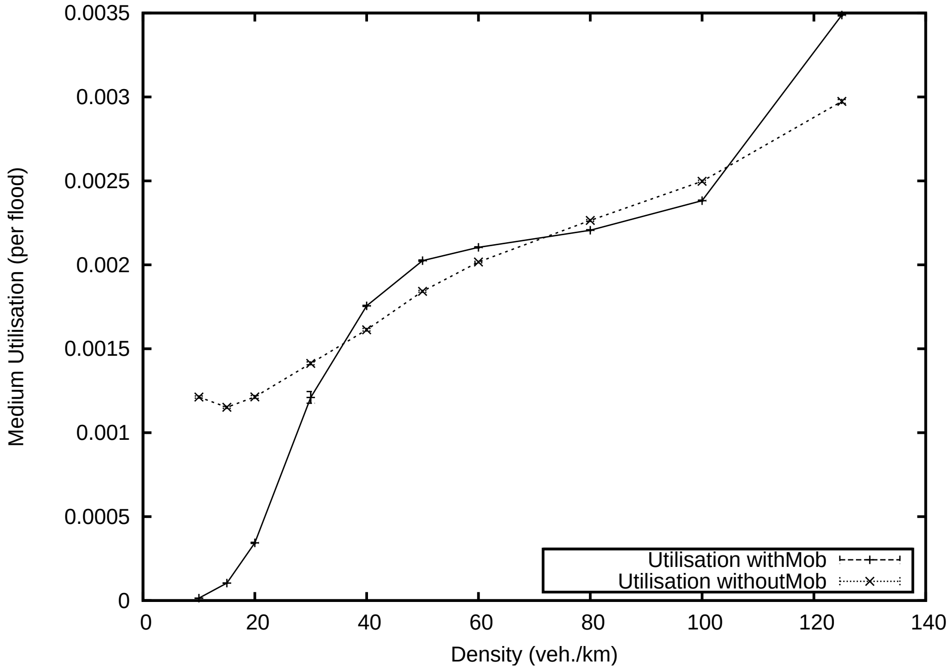


Figure 6.15: Medium Utilisation with mobility

6.3.7 Slot Utilisation

With mobility slot utilisation is different from the case without mobility. Figure 6.16 suffers the same problem as the others: for densities [10. . . 20] floods do not fully propagate. Ignoring the first three bars in the histograms it is remarkable to see that, with mobility enabled, more use is being made of slot 0, up to a density of 100 vehicles per kilometer. Because more transmissions take place in slot 0, it is to be expected delay will be lower. As observed in Figure 6.12, delay indeed is lower up to $\rho = 100$. An explanation for this is that the nodes are distributed differently by the IDM than the uniform distribution used in the static case; resulting in a more optimal arrangement.

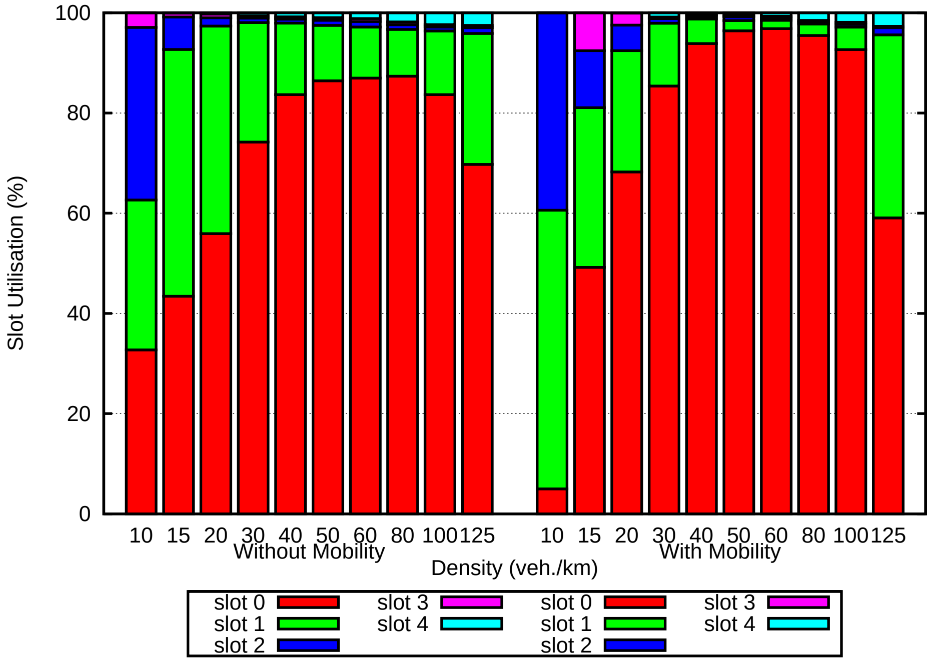


Figure 6.16: Slot Utilisation

6.3.8 Discussion

Node mobility seems to have little effect on the flooding scheme's ability to propagate information. The greatest threat to propagation is a partitioning of the network brought on by the acceleration of vehicles after leaving the reduced speed zone, or gaps introduced by sparse traffic. This is a realistic problem that can be overcome in two ways:

- Increase the transmission range. In this simulation the maximum achievable transmission range is approximately 250 meter. Increasing this range only works down to a certain density. This is—in the light of the application of the TrafficFilter—not such a bad thing because a low traffic density inherently means there is no traffic congestion. As a result, there is nothing to report. However, if several kilometers ahead another traffic jam occurs this information will not be propagated to vehicles beyond the head of the second jam, as depicted in Figure 6.17. This approach is also highly dependent on the properties of traffic. In this research the gaps were observed in the model derived from the IDM. It is to be expected this also occurs in real traffic, although our findings cannot directly be projected to real-life situations. Field studies will be required to derive the nature of the gaps at the head of traffic jams, and derive an adequate transmission range to bridge this gap.

- Augment the vehicle's communication abilities. This can be done by using vehicles in the opposite lane to carry messages, or using fixed infrastructure to convey this information to vehicles upstream. Study will be required to research the feasibility of this approach.

Option one comes with a drawback: the larger the transmission range, the greater the numbers of interferers. A dynamic solution could be to throttle the transmission power (and hence the achieved range) based on the local density of traffic. With respect to the collision domains, this need only be the number of equipped vehicles nearby. The reasoning behind this, is that when a high density is observed the network is less likely to be partitioned by large gaps, and a smaller transmission range can be used. The delay and number of hops required to travel 10km would surely increase, but the impact on the medium (utilisation and collisions) is reduced.

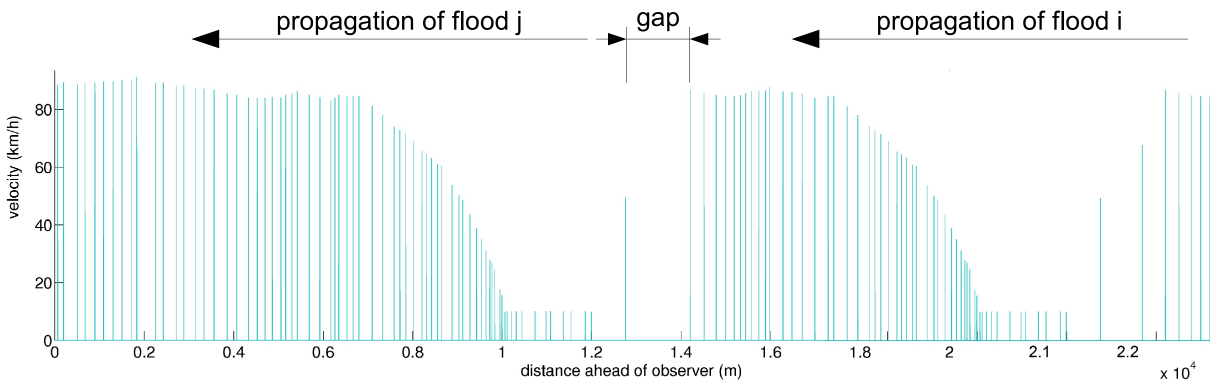


Figure 6.17: Notification of a second jam ahead does not propagate due to network partitioning

The influence of mobility on the physical layer and signal propagation level has not been considered in this research. On higher levels mobility does not seem to be of great influence. What is of influence is the way the nodes are distributed on the road, because this influences local densities and the number of interferers or the presence of gaps.

Some of the observed differences between the case with and without mobility can be explained by that the nodes have a different distribution at the time the floods are propagated. Figure 6.18 shows a plot of the mean and standard deviation of the node distributions for the cases with and without mobility. These values are calculated over the average inter-vehicle distance during a simulation run. As expected, the mean values (averaged over 50 runs per density) are the same for the two situations (they overlap so much the plot only shows one line). The standard deviation (plotted in logarithmic scale on the right y-axis) shows that, indeed, the distributions are different. The standard deviation is much larger without mobility. This can be explained as follows. The Intelligent Driver Model aims to find an optimal speed based on the distance to the vehicle in front, and the difference in speed. The result is that every node is influenced by its predecessor and uses the same deterministic algorithm to determine its own behaviour. This in contrast to the case without mobility, where the inter-vehicle distance is derived from a uniform distribution presented in Equation (6.1) on page 90.

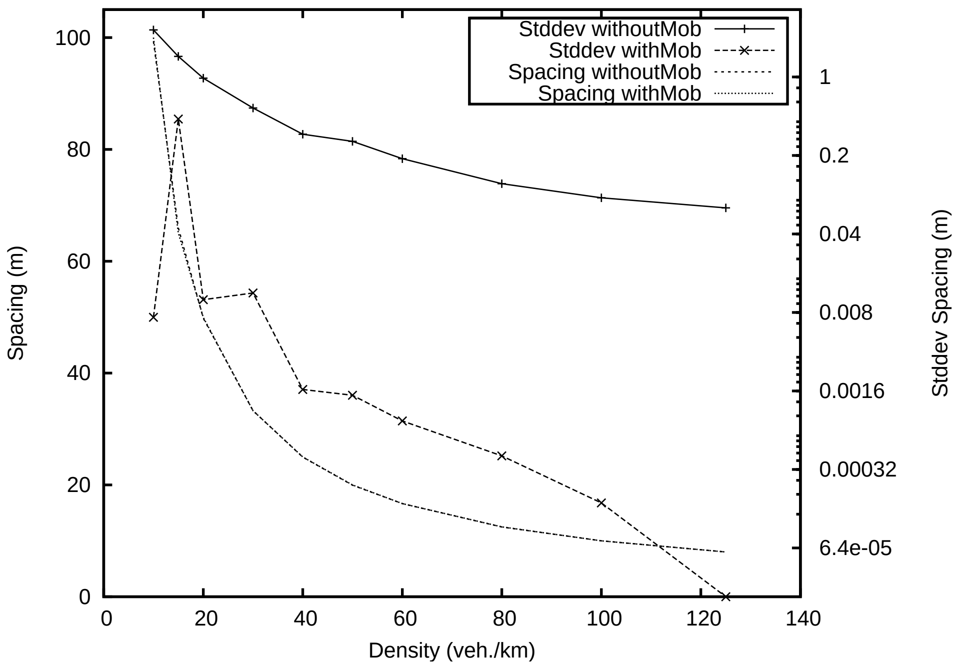


Figure 6.18: Comparison of the node distributions. Note that, although the average spacing is equal, standard deviation differs significantly.

6.4 Evaluation of the TrafficMap Contents

The flooding scheme's performance has been tested and it has been found to suffer little from node mobility. The task now, is to determine if the information distributed by the TrafficFilter and collected in the TrafficMap is a good representation of the actual situation on the road.

In Section 5.3.5 it was identified that there is a trade-off between rapid dissemination and selecting the best samples. A solution to safeguard the quality of the information contained in the TrafficMap, the Source Node Priority scheme (SNP), was proposed. Under SNP every node that has added a sample to the TrafficMap schedules a broadcast in the first slot. The remaining five slots are allocated based on the distance to the sender as described in Section 5.3.4 on page 79. Furthermore, the threshold function is modified to also use a distance-based threshold as described in Section 4.3 on page 61. This guarantees a sample every kilometer.

A series of simulation runs was performed to evaluate the quality of the contents of the TrafficMap both with and without SNP. SNP is expected to result in a better match between the actual situation and the captured information, but at the cost of a rapid end-to-end propagation. In both cases the simulation runs for 300s to stabilise, then 100 floods will be executed. Vehicle densities below 20 are ignored, because no fully connected network exists at those densities. At $\rho = 20$ approximately 8 out of the 100 floods completely propagates as found in Section 6.3.2.

When a node receives a TrafficMap, information contained therein is approximately 50ms old (based on previous experiments). However, it takes at most three seconds (the MIT period defined in Section 5.3.6 on page 83) for new information to come in and even more if the network becomes partitioned. A means to cope with this is not defined here but is left as future work.

6.4.1 Reachability

Earlier it was observed that end-to-end connectivity only becomes guaranteed from 30 vehicles per kilometer and up. This has two distinct causes: the transmission range of the nodes and the local density of traffic. If the transmission range were to be increased, fewer nodes would be needed to guarantee a fully connected network and tolerance to larger inter-vehicle distances.

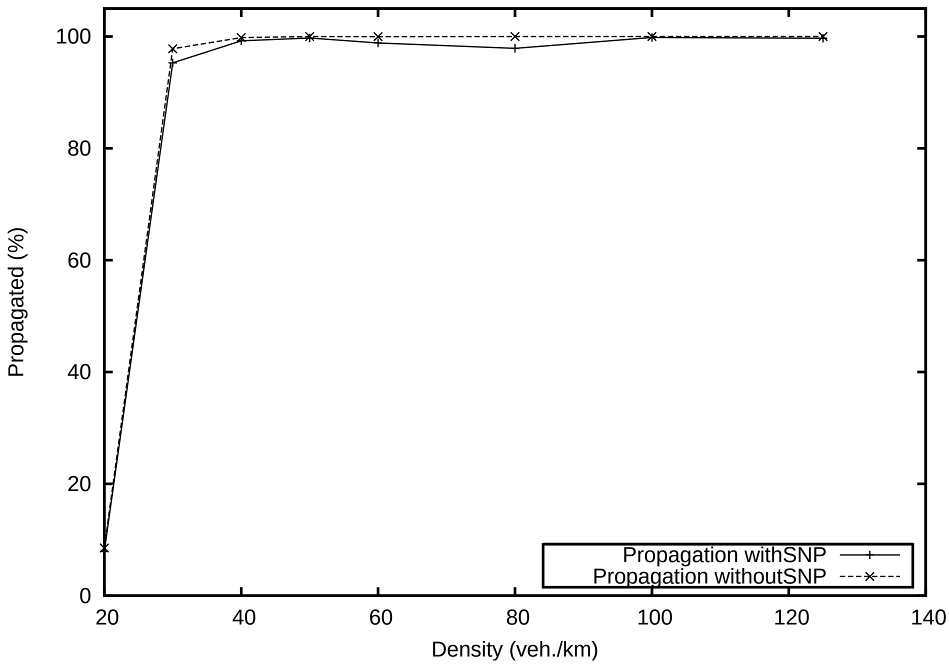


Figure 6.19: Propagation of Floods under SNP

An interesting thing to note is that Source Node Priority seems to have an adverse effect on reachability: 94.5% of floods traverses the full 10km when using SNP while without SNP this is 97.8%. This seems to be related to the node density; it could be some gaps or clusters of consecutive collisions exist under SNP around $\rho = 30$. As can be seen from Figure 6.19, the difference is very small and we cannot draw a solid conclusion.

6.4.2 Delay

The SNP scheme was expected to incur more delay; if no sample has been added to the TrafficMap rebroadcast must wait *at least* one slotTime. This means a delay of 5ms for every hop in which no sample has been added. Because the distance-based component of the ε-function triggers addition of a sample every kilometer every TrafficMap contains at least 10 samples. If a propagation takes 50 hops, 10 of those will be immediate rebroadcasts while the other rebroadcasts are delayed at least 5ms, resulting in an approximate penalty of 200ms. The simulation runs confirm this suspicion: without SNP the end-to-end delay is approximately 50ms while with SNP delay rapidly goes up to 250ms.

The increase in delay at $\rho = 125$ has the same reason as found in Section 6.3.3 in the previous series of simulation runs: due to mobility the local node density becomes very dense, resulting in collisions which in turn result in broadcasts in later slots.

6.4.3 Hops

The SNP scheme works by giving priority to a node that has added a sample. This is not necessarily the node which is located in the geographically most desirable location (i.e. farthest removed from the sender). As a result, the geographical distance covered per hop will be less and more hops will be required to cover the full 10km. This becomes clear in Figure 6.21.

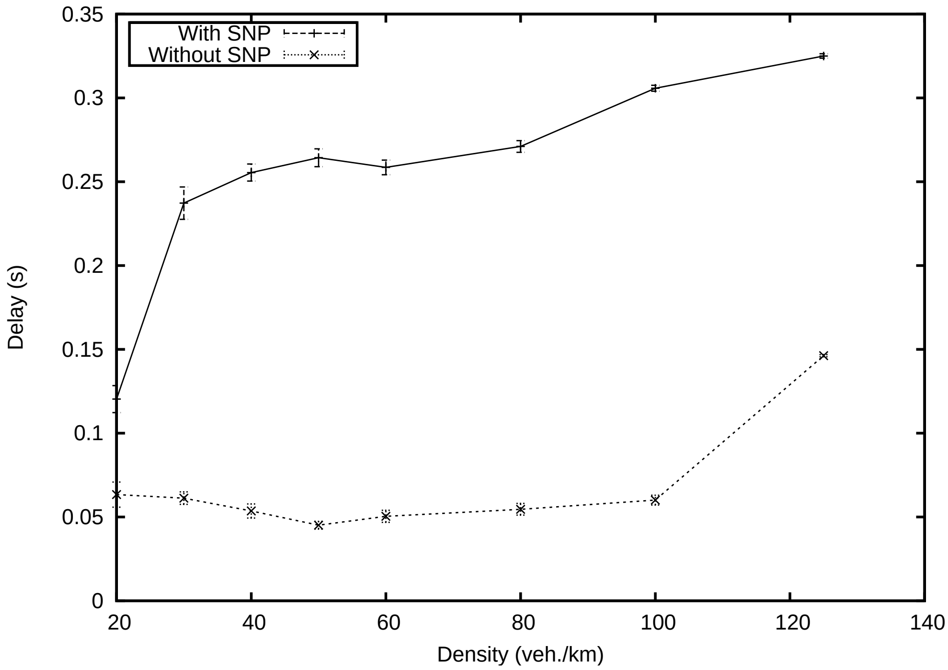


Figure 6.20: Delay

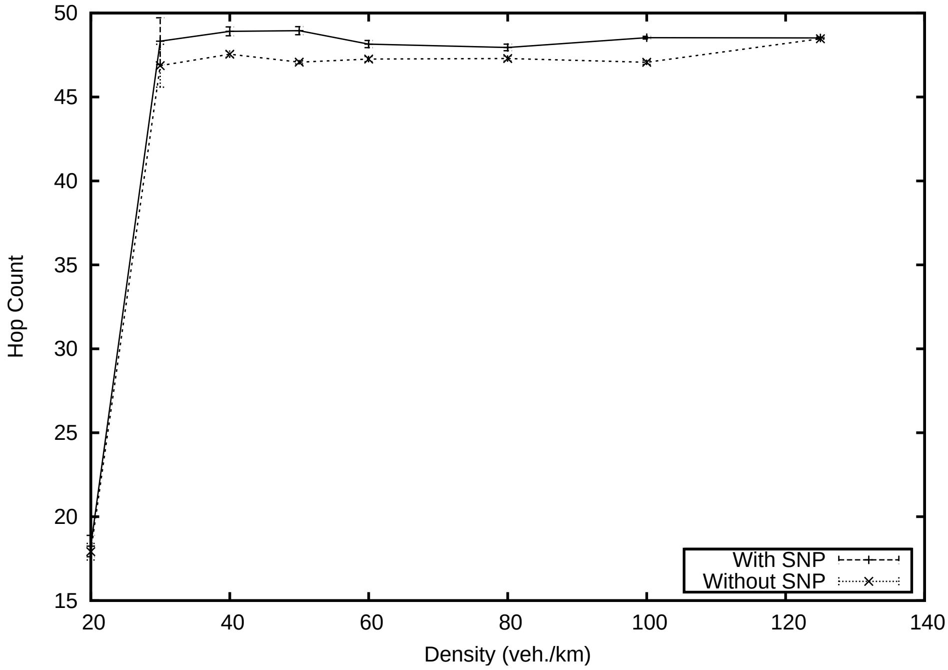
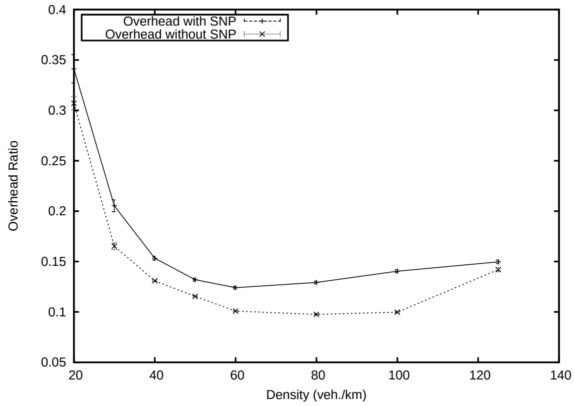


Figure 6.21: Number of hops

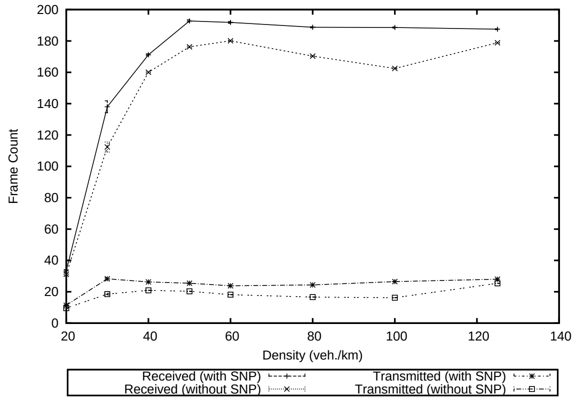
6.4.4 Overhead

Because more hops are needed, more transmissions are required under SNP. The result is a larger overhead per flood, as shown in Figure 6.22(a).

The Transmissions an Receptions plot in Figure 6.22(b) shows that the SNP uses more transmissions but also results in more receptions. More transmissions are required because more



(a) Overhead



(b) Transmissions and Receptions with SNP

Figure 6.22: Overhead and Transmissions and Receptions

hops are needed because the distance covered in one hop is smaller. Because of the microSlots the increased number of transmissions can still be performed quite efficiently, i.e. broadcasts have a great probability of succeeding and broadcast in later slots is seldom required.

6.4.5 Medium Utilisation

SNP requires more transmissions because the effective geographically covered distance per hop is smaller. This raises overhead and medium utilisation. There is, however, another factor: mobility creates a relation between a vehicle and its predecessor. If the vehicle in front brakes, so does the vehicle following it. When a flood arrives, both vehicles want to add a sample to the TrafficMap, synchronising their rebroadcasts. Both SNP and non-SNP cases use microSlots which are allocated based on the distance to the sender of the frame. This helps break the synchronisation a bit but may still allow collisions, which becomes evident from the higher medium utilisation, especially after 60 vehicles per kilometer, when the probability of multiple vehicles per microSlot increases.

6.4.6 Slot Utilisation

The SNP scheme works by allocating slots differently. The microSlotted 1-persistence scheme allocates five slots based on distance to the sender. The SNP scheme reserves slot 0 for the nodes which just added a sample. Remember from Section 5.3.4 that the microSlotting scheme allocates microSlots based on distance to the sender. The remaining five slots are allocated according to the original slotted scheme based on the distance to the sender. The difference of slot distribution has its effects on delay, number of hops and medium utilisation as shown above.

Figure 6.24 shows the slot utilisation without SNP and with SNP. The left half, without SNP, shows the typical high utilisation of the first slot (slot 0). When SNP is used, we see that between 20 and 40% of the nodes broadcasts in the first slot (the Source Node slot). We see a large number of transmissions in the second slot, and also in the third slot. This explains the large delay: if successful rebroadcast occurs in the third slot, the delay for this hop alone is 10ms.

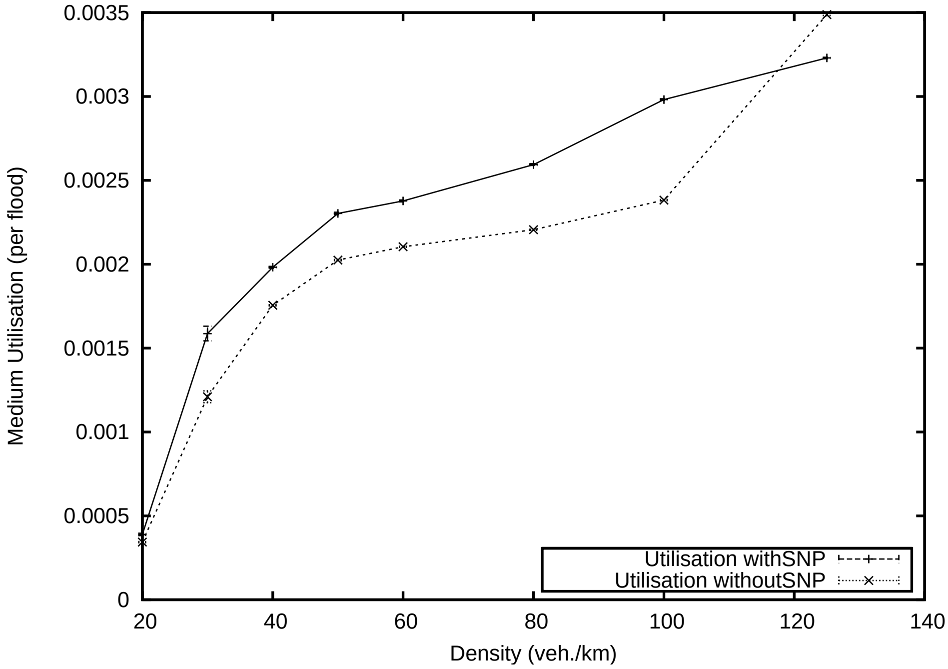


Figure 6.23: Medium Utilisation with and without SNP

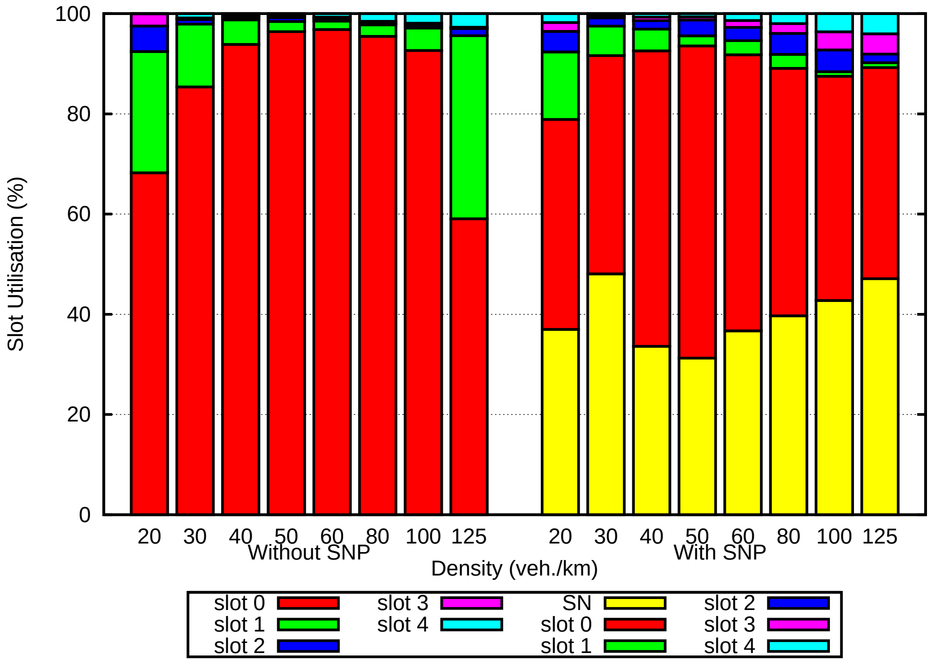


Figure 6.24: Slot Utilisation

6.4.7 Match of Communicated Information with Actual Situation

The ultimate goal of the TrafficFilter is to provide an accurate view of the speeds of vehicles ahead of the self-vehicle. One nice feature of using a simulator is that we can stop time, take a snapshot of the state of all vehicles and compare this to the information communicated in the TrafficMap messages. As discussed in Appendix B.6 on page 155 the simulator is instrumented to output a complete overview (ACT) of the position and speed of all nodes upon reception of a TrafficMap message (TM) by the node closest to the beginning of the road. Both TM and ACT are written to the output file and later processed.

The processing is explained in great detail in Appendix B.6, but in summary it does the following. The sets TM and ACT are extracted for every flood in the simulator, they contain (position,speed) information. Next, the distance between the tuples in ACT to the interpolated TM is calculated (measuring only deviations in speed). This results in a value expressed in km/h which denotes a deviation for every point in ACT. This value is averaged for all tuples in ACT (all vehicles on the road) and expresses an average deviation for this flood, an example of which is presented in Figure B.10 on page 157). These values are averaged for all floods in a simulation run (100). In order to arive at the statistics presented in Figure 6.25 the mean and 95% confidence intervals are calculated over 50 runs.

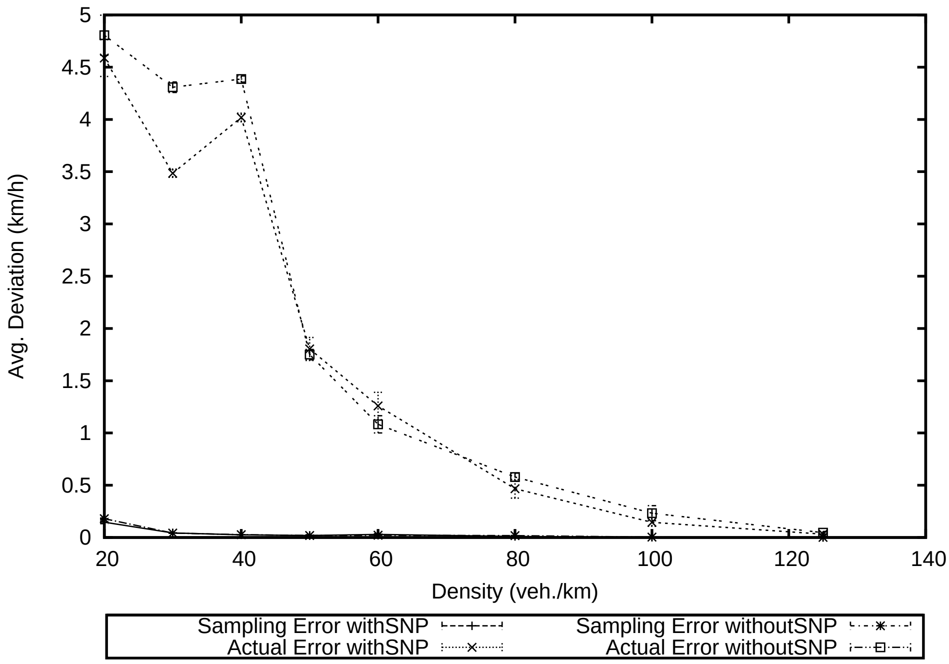


Figure 6.25: Match of TrafficMap information and actual situation

Referring to Figure 6.25, the two lines near the x-axis show the average absolute Sampling Error, i.e. how well the source nodes are able to capture the information. Both lines are so close to the x-axis and barely indistinguishable that the Sampling Error can be said to be almost negligible. This means that a sample in the TrafficMap is a good representation of the actual situation at that position at the time the message is received.

What is of more interest is the Actual Error, which is the result of the test described above. This shows how well the interpolated TM (a representation of reality) matches with the actual situation (reality itself). Both lines (with and without SNP) are of a significantly greater magnitude than the sampling error and, perhaps more important, both are of somewhat equal values.

The conclusion which can be drawn from this metric is that the information contained in the TrafficMap has a good relation with the actual situation and, more important, this relation is barely influenced by the way we select rebroadcasters. SNP provides no significant benefits, but the price is high: almost five times more delay, and the number of hops and medium utilisation are higher.

Still, this final phase of the research was not for naught: it provides the insight that the contents of the TrafficMap are good representations of the actual situation. We have shown that it does not matter if we choose the geographically ideal node for rebroadcast or the node which has just added a sample, the data delivered contains roughly the same information.

6.5 Discussion of the Evaluation

The simulation method used here deserves several remarks. First off, mobility is derived from a model. This model qualitatively shows driver-behaviour but clearly has some limitations: every driver is modelled equally. This will be different in reality because of driver abilities, state (alert or fatigued, thrillseeking or not) and vehicle. Furthermore, the IDM is used without being calibrated. As a result we can only draw conclusions which are indicative.

We did not explore the sensitivity to parameters such as MIT and the thresholds of the TrafficFilter, so there may exist more optimal settings for the TrafficFilter system. It is expected these settings depend on traffic density, flow speed and possibly even the transmission environment (e.g. its influence on the estimated transmission range which in turn influences slot allocation).

The radio wave propagation model is a simplified version of reality. Real 802.11p OFDM propagation at 5.87GHz can be expected to be much less predictable. Furthermore, this simulation study is limited only to interferers, other nodes. This study did not consider multipath and reflection effects incurred by the environment, such as buildings and large metal objects such as signs and other vehicles, or even the ground.

The transmission range used in this research (250m) is a “realistic” assumption; in IEEE 1609 it is expressed that transmission ranges up to 1km are desirable. As a result this research can be said to use a worst-case scenario when it comes to transmission range. When transmission range increases, so does the number of interferers. This research showed that the microSlotted 1-Persistence Flooding strategy performs well under large node densities, but if the transmission range were to be doubled the possible number of interferers could increase exponentially.

It is expected that power control can be used to dynamically reduce or increase the transmission range. When the density of (equipped) vehicles is large it might be necessary to reduce the collision domain to reduce the number of interferers (i.e. reduce the transmission power). When the equipped vehicles are sparse the transmission range may be increased in order to cross gaps between vehicles to still provide end-to-end connectivity.

6.6 Conclusion

This chapter described simulation studies carried out to obtain insight in the performance of the proposed system, the TrafficFilter. The TrafficFilter relies on efficient flooding of information. To this end the microSlotted 1-Persistence Flooding has been proposed in Section 5.3.4. It is based on the Slotted p-Persistence Flooding proposed in [121] and employs the addition of a small delay of multiple DIFS periods in order to break the synchronisation between all nodes within the geographically designated area of a slot.

The microSlotted scheme has been found to be efficient; it clearly outperformed the Slotted scheme with respect to delay and the number of hops required to travel the 10km. More important; the microSlotted scheme is able to achieve a high delivery ratio (high reachability) even when the vehicle density approaches 200, while the Slotted scheme rapidly drops.

The microSlotted 1-Persistence Flooding seems to suffer little from mobility. Gaps in the network and the distribution of nodes (for a given ρ) have more impact.

The analysis of the relation between the over-the-horizon view conveyed by the TrafficMap and the actual situation showed that the TrafficMap contains a good abstraction and does not necessarily need a priority scheme such as the Source Node Priority as proposed in Section 5.3.4.

Chapter 7

Conclusion

This chapter concludes this thesis. First, an overview of the results is provided in Section 7.1 to succinctly summarise the findings and the system which has been designed in this research. Next, Section 7.2 provides conclusions with respect to the findings of this research. The research questions presented in Section 1.3 are answered in Section 7.3. Section 7.4 points out opportunities for future work. Finally, Section 7.5 finishes with general recommendations applicable to the field of ITS and VANET research.

7.1 Overview of Results

This thesis started with a review of the Congestion Assistant proposed by van Driel [109]. The Congestion Assistant is an ITS application which relies on over-the-horizon awareness in order for the three proposed subsystems to work:

The **Active Pedal** smoothens the transition from free-flowing traffic at high speeds to slow or stopped traffic in a congestion. The goals are to increase safety and efficiency. In order to engage the Active Pedal, knowledge on the nature of traffic ahead is required.

The **Warning & Information** function keeps the driver updated on the situation ahead. Based on information from several kilometers ahead a driver can decide to take an alternate route. Furthermore, people were found to like being well-informed [109]. In order to provide this information to the driver a means to construct an over-the-horizon view is required.

The **Stop & Go** performs automated longitudinal control of the vehicle while in the congestion. This system needs to automatically be engaged at the start of a congestion and disengaged when normal driving recommences, it does not necessarily depend on over-the-horizon awareness.

In the Congestion Assistant [109] no means to acquire this knowledge has been proposed. The research described in this thesis set out to define a system geared towards the acquisition of such awareness.

A distributed ad hoc system called the TrafficFilter has been defined to provide over-the-horizon awareness to the Congestion Assistant. The TrafficFilter has two core tasks:

- Selection of information (by means of sampling) in a datastructure called the TrafficMap
- The efficient dissemination of this information by means of TrafficMap messages

For efficient dissemination, it is important that a TrafficMap fits in a single frame so it can be flooded through multi-hop Vehicle-to-Vehicle communication. Hence, it is paramount to select just enough samples to be complete, and not so much as to be redundant.

Based on the information contained in the TrafficMap messages, an over-the-horizon view up to the virtual horizon can be constructed in every equipped vehicle. On-board systems like the Congestion Assistant can then use this information.

The TrafficFilter is defined to use a standard IEEE 802.11p MAC and PHY of which only the DCF Broadcast functionality is used. A Network-layer Flooding strategy based on an improved version of the Slotted 1-Persistence Flooding is used to propagate messages against the flow of traffic.

The proposed improvement is the addition of microSlots to the Slotted 1-Persistence Flooding scheme. The slotted scheme was found to synchronise nodes which fall within the geographical area covered by one slot. This synchronisation is broken by including a small delay in the order of several DIFS periods to the wait time defined by the slotted scheme.

A possible modification to the allocation of broadcast slots called Source Node Priority (SNP) was proposed, but found not to contribute much.

7.2 General Conclusions

This section points out conclusions from the research presented in this thesis.

7.2.1 TrafficFilter

The information contained in the TrafficMaps has a good relation with the actual situation on the road. The results show that, for low densities, the errors are larger than for for high densities. This can be attributed to the higher speeds and greater dynamics in highway traffic at lower densities.

The system designed in this project, the TrafficFilter, relies on the presence of an ad hoc multi-hop network to carry the information. This research considers all vehicles to be equipped with an OBU capable of—at least—forwarding the message according to the microSlotted 1-Persistence flooding scheme. Findings indicate that the TrafficFilter can only be effective at high market penetration rates when the vehicle density is low. If the vehicle density increases, the minimum required equipment rate decreases. This relation has not been researched in this work, but is important future work.

The TrafficFilter is designed to measure an accumulation of vehicles. As that which the system tries to measure increases (the presence of traffic), so does the probability a connected network exists that can carry this information upstream. This combination gives a good outlook on a practical application: when the need to communicate is there, so is the ad hoc network.

7.2.2 Flooding in a VANET environment

The modification to the Slotted 1-Persistence Flooding scheme is found to be beneficial. The time and the number of hops required to traverse $\sim 10km$ of road are significantly lower than with the original scheme. It is possible to cover the approximately 10km in the order of 50–100ms whereas the Slotted scheme needs in the order of 200–950ms . The modification also greatly reduced the ratio of stopped floods due to collisions; it guarantees a very high delivery ratio from low to high traffic densities (at 200 vehicles/km, 97% of the floods propagates the full 10km using microSlots versus 13% for the slotted scheme).

The microSlotted scheme does have a little overhead in the form of some redundant broadcasts because they are already scheduled by the MAC layer and cannot be recalled. This, however, also adds a bit of robustness to the system: when the first broadcast is lost because of collision with another broadcast, the second transmission might still make it through. Because these two transmissions are in rapid succession and come from geographically almost equal locations low end-to-end latency can still be maintained.

The performance of the microSlotted 1-Persistence Flooding scheme does not deteriorate under influence of node mobility. It even showed improvements with respect to end-to-end latency (on average 20ms less time is needed if mobility is enabled). This can be attributed to the distribution of nodes. Because the distribution depends to a great degree on the actual situation on the road and the IDM serves as an abstraction, it cannot be concluded that delay in general benefits from mobility.

Gaps in the network are a fundamental problem if the aim is to achieve a large virtual horizon. It was found that, for a transmission range of approx. 250m a traffic density of at least 30 vehicles per kilometer was required in order to provide a connected network in case of node mobility with a traffic jam halfway. This problem could be solved by using a larger transmission range, V2I systems which relay the information upstream, vehicles on the opposite lane carrying messages or maybe even cellular technology.

7.3 Answers to Research Questions

In the introduction to this thesis, four research questions were presented. Answers can now be provided:

What are the Information Requirements of the Congestion Assistant?

It is concluded the Congestion Assistant requires knowledge about the speed on the road ahead. Although it might suffice to know the position of the head and tail of the jam, it is better to provide every vehicle with a speed profile, called a TrafficMap. ITS applications like the Congestion Assistant can then operate on this information.

Given the large costs involved in modifications to the infrastructure in order to achieve coverage on a national level, it is reasoned best to have the vehicles themselves generate the information.

How can the Congestion Assistant's information needs best be fullfilled?

When vehicles are used as information providers in order to provide information to other vehicles, a vehicle functions both as producer and consumer of information. Since cellular and radio broadcast approaches have scalability issues for the envisioned number of nodes, and potentially long delays, an ad hoc multi-hop Vehicle-to-Vehicle (VANET) communication approach is the most viable.

A multi-hop V2V communication-based system has been proposed in this thesis: the TrafficFilter. This is a distributed system in which vehicles on the road cooperatively build an over-the-horizon awareness contained in TrafficMap messages which are efficiently disseminated against the flow of traffic.

What is the performance of this method and what are the trade-offs?

The TrafficFilter uses microSlotted 1-Persistence Flooding which was found to be more efficient than Slotted 1-Persistence Flooding. It is possible for a TrafficMap message to traverse approximately 10km of road well under 100ms while picking up enough information to contain a good representation of that same 10km of road.

Given the ad hoc nature of a VANET and the fact that nodes also function as repeaters to provide coverage over a great area the system is inherently sensitive to segmentation of the network. This implies that gaps between clusters of equipped vehicles can exceed the transmission range, and cause the virtual horizon to become very close. This is a fundamental problem which needs to be solved when the Congestion Assistant is to supply information about the next several kilometers, and should be operational and accurate at all time.

Is it possible to meet the Congestion Assistant's information needs?

On a conceptual level: definitely. There are no fundamental problems: as that which the system tries to measure increases (the presence of traffic) so does the probability a connected network exists that can carry this information upstream. This combination gives a good outlook on a practical application: when the need to communicate is there, so is the ad hoc network. This assumes a large degree of market penetration, the exact value of which is at this moment unknown but depends largely on the transmission range of future radiocommunication hardware.

On a practical level: the TrafficFilter has been tested on a one-dimensional one-way highway in a simulator. More study is required to discover the interaction between the TrafficFilter and the Congestion Assistant across more complex road and traffic situations. Ultimately, to answer this question, field studies are required.

7.4 Future Work

Because the area of VANETs is still a young area of research, a lot of opportunities exist. As IEEE 802.11p is standardised equipment for field studies will become available, opening up a multitude of new possibilities for research.

7.4.1 The TrafficFilter

This research was limited to a one-lane stretch of highway. It remains unclear how well the TrafficFilter will perform in a more complex topology with multiple lanes, intersections, junctions, bridges etc.

The information contained in the TrafficMap messages can be interpreted in various ways; for instance by matching the samples with blocks or with a best-fit curve as presented in Figure 4.13 on page 59. The information in the TrafficMap could also be used to make predictions, derive heuristics etc. In this research we refrained from doing so but smart analysis of the data contained in a TrafficMap may yield a lot of information.

The TrafficFilter system uses thresholds. These thresholds result in optimal operation under certain conditions (w.r.t. traffic flow speed and density) but might not be correct for other conditions. In order to operate under a wide variety of conditions the thresholds will need to be dynamic. It is expected that the present target virtual horizon, the number of samples in the TrafficMap and the observed local traffic density and flow speed can be used to derive a dynamic set of thresholds.

In this research some analysis has been performed concerning the dynamics of traffic and the rate at which received information ages; the result of this analysis was a Maximum Inter-TrafficMap Time (MIT) and a TrafficMap Flood Limit (TFL). These are two configuration settings of the TrafficFilter system and have a close relation with how well the system will perform in practice. Because of the nature of the traffic model used in this research, the values chosen for MIT and TFL are placeholders. They work within the scope of this implementation with this model. In realistic situations MIT and TFL might need to be adjusted.

In the TrafficFilter a function performing averaging of sample data is proposed. This has not been implemented and tested in the simulator. As such, researching the effects of this addition are left as future work. The reduction step, which is proposed to merge several samples and thus reduce redundancy to make the data to be propagated smaller, has not been evaluated either.

The interaction between the TrafficFilter and the Congestion Assistant has not been researched. It could be that the TrafficFilter, by design or because of more fundamental issues, influences the effectiveness of the Congestion Assistant. Properly evaluating this will require an integrated mobility / network communication simulation study on a scale larger than the ones performed in this research.

7.4.2 Flooding in a VANET Environment

The microSlotted 1-Persistence Flooding, used in this research, uses 5 slots and an effective transmission range of 250 meters. When the IEEE 802.11p draft reaches standard status there will be more certainty about issues which are, at this moment, still open within the draft. It could very well be that the transmission range will differ in practice; as a consequence 5 may no longer be the optimal number of slots.

It is identified in this thesis that gaps in the network are a threat to TrafficMap propagation, thereby negatively impacting the extent of the over-the-horizon awareness. It is reasoned that these events are likely to occur because the TrafficFilter is designed to operate under congested traffic conditions, and it is under these traffic conditions that a gap may occur at the head of the jam. How to bridge these gaps—or low vehicle densities in general—remains a challenge. Possible techniques are using the lane on the other side or resorting to communication with fixed infrastructure, either through roadside transponders or cellular technology.

There exists a certain critical vehicle density at which, for a given transmission range, propagation over great distances becomes possible (i.e. probability of success is larger than zero). In this research this density was found to be between 20 and 30 vehicles per kilometer for a transmission range of 250m. These results pertain to the situation in the simulator used. More research is required to get a better notion of the exact relation. From this relation a target penetration rate at which the system will start to function could be derived.

The propagation model used in this research was the Free Space Propagation model. In order to make better judgment of VANET performance using IEEE 802.11p a propagation model, validated by field studies, is required.

Flooding schemes like the ones described in this thesis rely on the ability to estimate the transmission range of the OBU. It has been shown that a good estimate which coincides with the actual transmission range is of critical importance to the performance of the information dissemination. In order for these Flooding schemes to be of any use in practice an IEEE 802.11p propagation model, validated by field studies, is required.

7.5 Recommendations

- It is important that ITS technologies are standardised, in order to provide a safe and efficient system. From a safety point of view it is important OBUs from different vendors can interoperate without error. From an efficiency point of view a standardised interface and set of protocols is easier to maintain, test and implement than vendor-specific systems with adaptation interfaces for other vendors' systems.

- It is reasoned to be important to inform the driver of the operational status of his Driver Support Systems because of the trust placed in them.

- A system like the one proposed in this thesis relies on a high penetration rate to be beneficial. If the penetration rate is low, gaps occur in the chain of rebroadcasts and the information will not travel further. It is advised to, by some means, stimulate the adoption of such systems when they become available. This is even more so important for applications which rely on high penetration ratios, like the one proposed in this thesis. The individual may not directly benefit from having a TrafficFilter / Congestion Assistant, but the improvements for society as a whole may be reason for governments to consider legislation to stimulate adoption.

- In the light of complexity, passing the information on and actually using it in an on-board system can be seen as two separate things. Passing the information on requires only a very simple OBU consisting only of simple circuitry and an antenna, while integration with other on-board systems may not (readily) be an option in new or existing car models. As such retrofitting could be an option even in old vehicles. This could help in rapidly reaching a large market penetration of multi-hop communication-able vehicles which will function as the carrier network for future fully integrated vehicles.

- The system proposed in this thesis, the TrafficFilter, is but one facet of the integrated possibilities of future ITS applications. An integrated ITS platform would require a broader scope on the applications and their requirements such as safety-of-life features like Collision Avoidance Systems. Once all the seperate applications have been defined an attempt can be made to derive an efficient integrated solution. It is expected that a lot of ITS applications require the same information as input, so it does not make sense to have separate systems gather the same information. In this situation each system will suffer from the increased medium utilisation, contention and delay. As such in order for the vehicles to be able to cooperate, the ITS applications must be able to cooperate. This could be thought of as a symbiotic relation.

Appendices

Appendix A

An Analysis of Position and Speed Encodings

A TrafficMap expresses position and speed information. In this appendix three possible expressions of such information are highlighted. These schemes are Absolute Positioning (direct use of global position information, Relative Positioning (performing a mapping to the own local area) and a novel approach referred to as Road Information Based Positioning. The latter approach relies on knowledge which is distributed a priori and context-awareness of an OBU.

A.1 The Horizon

Assuming there are no natural or man-made obstructions, the distance D to the horizon can be calculated with the following formula [96]:

$$D = 112.88km \times \sqrt[2]{h} \tag{A.1}$$

Where h is the height of the observer above sealevel in kilometers. An observer standing atop a mountain 1 km high (h = 1 km) sees the horizon 112.88 km away. Equation (A.1) neglects the refraction of light in the atmosphere and a small simplification has been made with respect to the relation of the height of the observer and the radius of the earth (approx 6371 km) but the error is negligible.

When applying Equation (A.1) to a driver in a vehicle D will be approximately between 3.5 and 5km, assuming a completely smooth surface of the earth. Generally terrain features and buildings dramatically reduce the distance to the horizon to the range of tens to hundreds of meters. In traffic the distance can even be smaller when, for instance, a large truck blocks the view on the road ahead.

In order to augment the vision of drivers or automated systems an over-the-horizon view can be realised. This view is not limited by line of sight when the view is constructed from data disseminated through V2V messaging and is only limited by the *Virtual Horizon* as defined in Section 4.1. The virtual horizon is the maximum to which the over-the-horizon view extends. In some cases it suffices to extend the view to just behind the truck in front—to see if there is traffic on the opposing lane for safe overtaking—but for the Congestion Assistant the Virtual Horizon extends to—possibly—several tens of kilometers.

A.2 Position Encoding

We will now have a look at the encoding of positions. A position needs to be known in order to pin-point the exact location of a measurement. It is of importance that a position is accurate and can be efficiently encoded.

We identify three methods of position encoding:

- absolute coordinates
- coordinates relative to the observer's location
- high-level information obtained from road maps in conjunction with Location Database information

We stress the difference between the position of a vehicle and the position of a measurement which is added to the TrafficMap. A certain vehicle is located at a certain position, it derives this information from its on-board equipment. When a vehicle adds information to the TrafficMap, it adds its own position. This position then denotes the position at which the measurement was taken, and has no longer a relation with the vehicle that performed the measurement. As such, a measurement shows the flow of traffic at that location.

A.2.1 Positioning with Absolute Coordinates

The use of absolute positioning seems the most straight-forward option because a node is able to obtain its own absolute position using GPS. A source vehicle can then simply broadcast this value and propagate this upstream through multi-hop communication. When an observer a few hundred meters upstream knows its own absolute position and that of vehicles in the tail and head of the jam the distance between itself and the tail and head can be calculated.

The position information can be obtained from a vehicle navigation system (GPS position). It is assumed the coordinates are mapped according to WGS84 (World Geodetic System '84, last revised in 2004) [50]. WGS84 is a so-called *datum*, a geodetic reference system that specifies the size and shape of the earth, and the base point from which the latitude and longitude of all other points on the earth's surface are referenced. Without going into too much detail this results in a three-dimensional position fix consisting of a Latitude, Longitude and Altitude. Latitude denotes the North-South position with +90° at the North Pole and -90° at the South Pole. Longitude denotes the angle along the equator between a point and the prime meridian through Greenwich. The Earth is divided in 180 degrees eastward and 180 degrees westward. As a result 0° is the meridian through Greenwich, 180°W = 180°E.

A degree (°) is divided into 60 minutes (') or in decimals. A minute is divided into 60 seconds (") or decimals. A second generally is divided into decimals. For example, the city of Enschede is located at $52°13'N$ $6°53'E$'. This notation is accurate to approximately 1.86km. To achieve greater accuracy, fractions of minutes or seconds are added. For example, $DDD°mm.mmm$. These coordinate values need to be mapped to a string of bytes of minimum size while still maintaining enough accuracy for the Congestion Assistant.

RFC 1876, proposed for positional information in the Domain Name System in 1996 [22], proposes a mapping of WGS84 to 12 bytes, as presented in Figure A.1. Latitude, Longitude and Altitude each comprise four bytes. Note that there are some roads which occupy the same latitude and longitude coordinates: bridges and the like. How to denote a position on these more complex roads is left as future work.

Using the encoding scheme proposed in RFC 1876 the position is accurate to $\frac{1}{1000}^{th}$ of arc, which means accurate to 1,84 meter Latitude and 1,86 meter Longitude [93]. The Altitude

```
   MSB                                           LSB
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
   0|                   LATITUDE                    | DDD mm.mmm'
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
   2|                   LATITUDE                    |
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
   4|                   LONGITUDE                   | DDD mm.mmm'
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
   6|                   LONGITUDE                   |
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
   8|                   ALTITUDE                    | cm
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
  10|                   ALTITUDE                    |
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 (octet)
```

Figure A.1: RFC 1876's position encoding [22]

is expressed in centimeters from a base of 100,000 meters below the GPS reference spheroid [22]. As a result of the tendency of the road network to be generally flat—with the exception of the occasional tunnel or bridge—the use of Latitude and Longitude suffices, resulting in an expression of 8 bytes.

Conclusion The exact datum used (WGS66, 80, 84 or variations such as Hartbeesthoek 94)[116] is of no importance. What is of importance is that a Latitude and Longitude is available, and the same datum is used throughout the system. When using different datums it must be clear which one is used, so conversions can be applied. These values each fit within a 32-bit integer, making a total set of 8 bytes denoting a position as exact as the coordinate system allows.

A.2.2 Positioning with Relative Coordinates

Relative Coordinates are relative with respect to the observer. A benefit is that a position can be encoded in a smaller number of bytes because there is no need to address locations on the entire Earth (as absolute positioning does), but the coordinate system can be limited to the area within the *virtual horizon*. 16 bits are enough to uniquely address 65536 locations. Using one-meter granularity this is already more accurate than the absolute coordinate system and even more accurate than GPS positioning generally is as noted in section 3.2 on page 23.

A Relative Coordinate sample is adjusted by every receiver to be mapped to its own absolute position. In a sense, relative coordinates are *anchored* in the observer. When they are transferred to an observer at a different location a mapping is required in order to anchor the relative coordinates to the new observer. This process is illustrated in Figure A.2.

One-dimensional Relative Positioning

The one-dimensional relative coordinate system is strictly for proof of concept. The distance is used to denote the position along the same road. With 16 bits and one-meter accuracy positions up to 65km ahead can be designated. This is well beyond our target virtual horizon of 10km.

Two-dimensional Relative Positioning

We can map Cartesian coordinates along our own axis of travel. This has several benefits over one-dimensional coordinates. First off, our transportation systems are generally two-

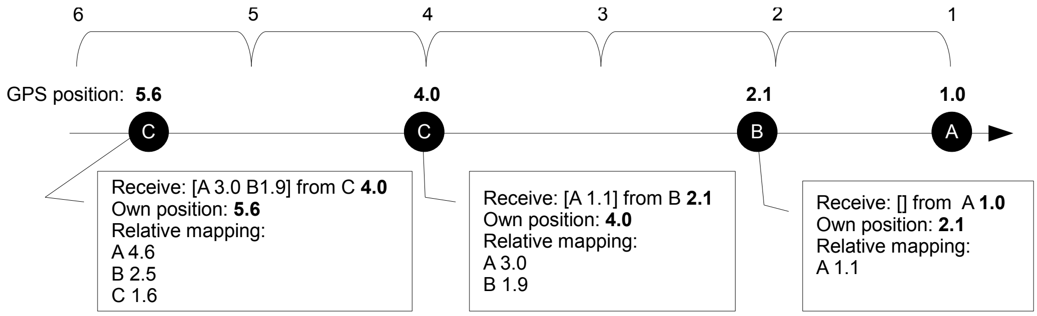


Figure A.2: The mapping process maps absolute positioning (in bold) to (condensed) relative positioning.

dimensional. Secondly, a one-dimensional coordinate system does not allow crossings and junctions while a two-dimensional system does.

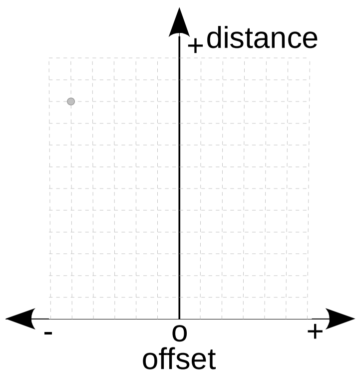


Figure A.3: Denoting a relative position using Cartesian coordinates.

Using a 16-bit unsigned integer to denote y (distance), and a 16-bit (signed) integer to denote position in the x plane, results in a 65 by 65 km area ahead within which a vehicle can be positioned with 1m accuracy. This results in 4,294,967,296 addressable locations or 4225 km^2, this is approximately one tenth of the Netherlands.

An alternative is to use Polar coordinates (or actually, Circular coordinates because no Z-axis is involved) along our own axis of travel in conjunction with the 16-bit distance.

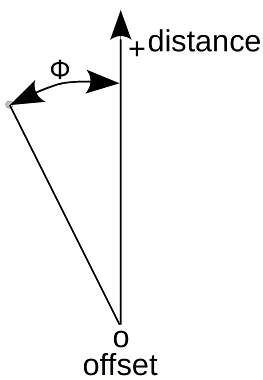


Figure A.4: Denoting a relative position using Polar coordinates.

The system only requires information from downstream, henceforth it suffices to express angle ϕ in π radians. This would probably amount to some signed value which is between $-\frac{1}{2}\pi$ and $\frac{1}{2}\pi$.

An interesting feature of this method is the reduction in data and also the 'loss of precision' as the distance increases, since our ϕ is a discrete value. A greater width can be covered (up to 65km both left and right).

If one byte is used for ϕ (which yields 265 'forward directions'), the result would be $265 \times 65536 = 16,777,216$ different locations, covering an area of $6746.5km^2$.

- On the arc at radius 1m, 256 points are located. They are equally spaced at a distance of 12.27mm because $\frac{1 \times \pi}{256} = 0.01227$.
- On the arc at radius 65536m, 256 points are located. They are equally spaced at a distance of 804.25m, since $\frac{65536 \times \pi}{256} = 804.25$.
- At radius 326m discriminatory distance is 4m. Beyond this distance it is not possible to (accurately) depict a lane or discern two adjacent vehicles.

The accuracy in ϕ is dependent on the distance r in the relation $\phi = \frac{r \times \pi}{256}$.

Conclusion Cartesian coordinates provide a straight-forward means to denote a position relative to an observer. Circular coordinates provide a means to denote a position with less bytes of overhead. Notice the great density at close range and the low density at long range, this induces large error margins at long range. Consider, for instance, a position 65536m away, located on the arc indicated by ϕ. At this distance resolution is 804.25m; a position fix will be mapped to the nearest coordinate and hence induces an error of at most 402.125m.

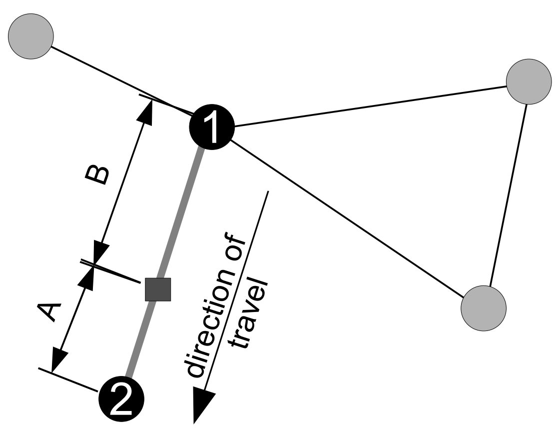
For both Cartesian and Polar coordinates a mapping of data to the own position is performed in every vehicle by means of the own position and heading. This mapping plus the effects of propagation and processing delays introduces a certain error. It is interesting to find out how large this cumulative deviation is in simulation and ultimately in practice, but this is left as future work.

A.2.3 Positioning with Road Information

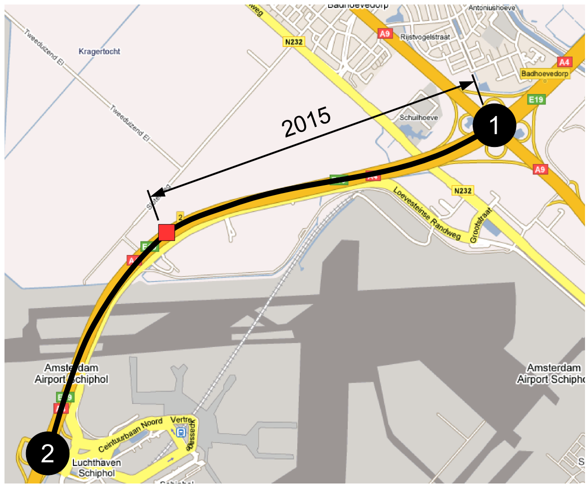
A vehicle knows its global location through on-board GPS, possibly supplemented by odometer measurements and corrected by means of a road map to 'snap' to the road in case the obtained position is a few meters off. Using the onboard mapping equipment already present in navigation systems an OBU knows on which road it is driving, in which direction, in what lane and even at what position along that road.

In the Netherlands the roads are standardised and the network is extensively mapped. A location along a road is expressed by means of hectometer signs next to the road. For this solution to be viable universally it is important the road network is standardised, mapped and measured. Locations have been mapped for use by, among others, the RDS/TMC system in national Location Databases. These databases have been discussed in section 3.1.3 on page 21. From [6] it becomes clear the 16-bit identifier in the RDS/TMC message refers to a location mapped in decimal degrees. An example is 41.98449°N 12.49321°E, a junction near Rome. The *ddd.ddddd°*-format is accurate to ${\frac{1}{100000}}^{th}$ of arc, which means an accuracy of 1.12m Latitude and 1.11m Longitude [93].

This database gives a set of standardised points on the road map, uniquely identifiable with a 16-bit code and are absolute to anyone using the same location database. The locations in the database can function as reference points. Since these points are often geographically separated



(a) The road network is a graph, locations are the vertices connected by edges (roads).



(b) The example of Table A.1 mapped to Figure 3.2

Figure A.5: Road Information Based Positioning

from each other by several kilometers one point is a poor indicator. A vehicle driving on a stretch of road can define that stretch by means of two points, connected by the road.

The locations form a graph as depicted in Figure A.5(a); locations are vertices and road sections are the edges connecting them. A position on such an edge can be identified by means of a simple distance:

A – to the nearest vertex

B – to the first vertex in the pair (“distance travelled from 1 to 2”)

The ordering of vertices in the pair indicates direction of travel. Using a road map and the two points the OBU can infer the heading of the vehicle, which is the tangent to the road at that location—assuming a vehicle generally moves with the direction of the road. If this is not the case there probably is an emergency situation. Table A.1 gives an example of the Road Information Positioning applied to the message treated in section 3.1.3. Here two points are denoted, Junction Badhoevedorp and Exit Schiphol. The ‘Distance’ is a measure of how far from Point 1 to Point 2 the vehicle has progressed, 2015m in this case. This information is represented graphically in Figure A.5(b).

Conclusion Road Information provides a positioning scheme based on the context in which vehicles operate. Because it is absolute no mappings are needed, the locations in the database function as local reference points. A position is always mapped to a road and the position can be encoded in a compact way using only 6 bytes. This method will have to perform frequent lookups in the Location Database in order to map the received information to the own situation but this will not be a performance bottleneck, the complete location database for the Netherlands[1] contains 8215 entries as of March 2008. This translates to about half a Megabyte of data.

[1]The author obtained a copy of the Location Database (VILD version 4.3a) from VCNL/DVS.

Data	Contents	Interpreted as	Size
Point 1	location code: 9231	Junction Badhoevedorp-A4	2 bytes
Point 2	location code: 9227	Exit Schiphol-A4	2 bytes
Distance	2015	meters from 1 towards 2	2 bytes
			total: 6 bytes

Table A.1: A position expressed in Road Information Based Positioning format

A.3 Speed Encoding

Vital to the over-the-horizon view is the speed of a distant node or the average speed at a distant stretch of road. The main goal of the TrafficMap is to elucidate spots of slow-moving traffic. A very simplistic approach would be to define two ranges; congested and free-flowing. Any speed below 50 km/h could be marked as congested, anything above as free-flowing. It will be clear that this method does not provide a high resolution representation of the speed and as such might not provide sufficiently detailed information to calculate, for instance, the expected delay as part of the Warning & Information function.

A speed can be expressed in meters per second, kilometers per hour, miles per hour or even knots. Speed will be expressed in kilometers per hour, and will be mapped to a byte. As a result speed can vary from 0 to 255 kilometers per hour.

A.4 Heading Encoding

When using absolute or relative positioning a means to denote a heading is needed. Reasoning that a road has only two directions one bit could be used to denote this. The problem here, is that it still is not clear which direction is indicated because there is no reference. In traffic jam warnings direction is often expressed as [road number] in the direction of [city / offramp / junction], the encoding of which would require many bytes. Better would be to use a heading in degrees or in Cardinal points like Figure A.6. Using Cardinal points such as North, North East, North-northEast and Northeast-by-north, a mapping of 16 or 32 directions to 4 or 5 bits can be derived. Or the full 360 degrees could be divided into 256 quanta to map to one byte for an accuracy of 1.4 degrees.

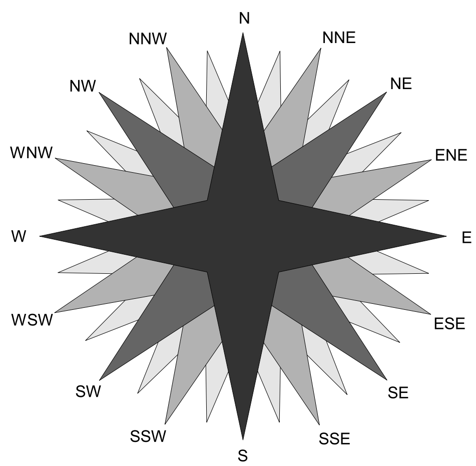


Figure A.6: Cardinal points on a compass

A.5 Lane Encoding

Dutch Highways generally have two lanes per direction. although there are some highways with more lanes. Generally they are numbered from inside to outside, making the 'fast lane' numbered 1 and the others incrementally 2, 3 etc. This could be mapped to 3 bits, assuming no highway has more than 8 lanes per direction. The Route Périphérique in Paris, one of the busiest highways in Europe, has 8 lanes in total, 4 per direction. Using 3 bits seems a relatively safe assumption, although there may exist parts of highways which have more lanes.

It is, however, not yet clear if there is a need to have knowledge of the lane on which a vehicle drives. The highway could be seen as a pipe, through which traffic flows. Thereby the flow speed of the independent lanes could be averaged. If this is a realistic assumption is left as future work. Henceforth explicit lane encoding will not be concidered at this moment.

Scheme	Absolute Position	Road Information	Relative Postion
Coverage	Global	National	Local
Information	Latitude, Longitude, (Altitude)	L1, L2, distance to L1	distance,offset or (distance, ϕ)
Size	4+4=8 bytes	$2 + 2 + 2 = 6$ bytes	4 or 3 bytes
Size (complete)	10 bytes	7 bytes	6 or 5 bytes
Max. Accuracy	1.8m	approx. 1m on a road, L1 and L2 approx. 1.1m	1m, ϕ: $\frac{r\times\pi}{256}$
Computational Complexity	low	higher	higher
Relation to road	none	strict	none

Table A.2: Comparison between three position approaches

A.6 Comparison

The methods to build an over-the-horizon view by means of the TrafficMap primarily differ in the way they denote a mobile node's position. Table A.2 holds the properties of the different schemes. All data is mapped to a byte, the complete size includes a byte for velocity and one for heading (which is, obviously, not needed in the Road Information scheme because this information can be derived from the position encoding as proposed in Section A.2.3). With respect to the size in bytes the relative positioning excels both at accuracy and size, especially when using Cartesian coordinates. The only inaccuracy in the relative system is introduced through the per-hop mapping. The effects of this are not yet researched but could be minimised by using appropriate corrections depending on node movement and processing delay.

Because of the fixed nature and the small size to denote a position with the Road Information scheme is also very compact. It is a context-dependent positioning scheme, as every node needs to have the Location Database. This should, however, not be a problem since this information generally is already present in modern satellite navigation systems with RDS/TMC functionality. The fact that this information is distributed 'a priori'—either during production or as part of the operating system image—means that during operation a small pointer (the Location Code) can be used to denote an exact location expressed with great accuracy in the Location Database. The downside is that such a database needs to be present and up-to-date for any country the vehicle travels through.

The Absolute Positioning scheme provides a global scheme which is completely infrastructure-independent. This is a large benefit because it might open the door to very diverse applications but the benefit is offset by the larger size of 8 bytes for a position fix. Note that, no matter which positioning scheme is chosen, the operational obtaining of a position fix from the GPS in every vehicle may introduce additional errors.

A special note on altitude is in order. In this research it is assumed the road network is two-dimensional and exists only in latitude and longitude dimensions. There are short sections where some roads are stacked (bridges, for example, have the same latitude and longitude but two differing altitude components). The Location Database described in Section 3.1.3 and Appendix A.2.3 contains only the latitude and longitude of a location. This reinforces the decision to only latitude and longitude. It is reasoned that, with a latitude, longitude and a heading it can be reasoned whether a vehicle is on or under the bridge by consulting a map under the assumption vehicles always travel along the tangent of a road, and not perpendicular to it.

A.7 Conclusion

For the simulator studies performed during this research the absolute positioning scheme is used because it provides a direct mapping from the simulator's internal positioning scheme. Future work will have to evaluate exactly which scheme to use in practice. It is clear that, for the system to work, every node should be capable to correctly interpret the information. In case the Road Information Based Positioning (or an equivalent scheme) is used every node needs to be equipped with a Location Database. As such a standard will be required for use across hardware from different manufacturers. This standardisation does not only pertain to the system described in this thesis, but to ITS systems in general.

Added to the ability to interpret is the required ability to produce information. As such it is assumed every equipped node is able to measure its speed, position, heading etc. with adequate accuracy.

Appendix B

The Simulator

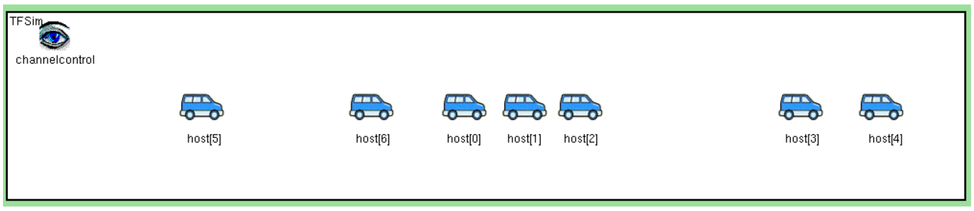


Figure B.1: Vehicles in the Simulator

This Appendix covers the implementation of the TrafficFilter system as designed in Chapter 5 into the OMNeT++ discrete event simulator. We will first introduce some background on the building of a simulator. Next the OMNeT++ discrete event simulator and the Mobility Framework are introduced in Appendix B.2. Third we will cover the implementation of the Intelligent Driver Model into OMNeT++ using the Mobility Framework, followed by the implementation of the TrafficMap message and the flooding strategy in Section B.5. Section B.6 covers the implementation of a means to judge TrafficMap quality. We will finish by briefly describing how the simulator is instrumented to extract the required measurements and a note on node placement.

B.1 Building the Simulation

Many VANET communication studies use a rather simple mobility model [104, 126, 29]: a certain density or distribution is defined, as is the length of the road. For instance, 500 vehicles are equidistantly placed on a road with a length of 10km. These vehicles show no realistic traffic pattern other than behaviour that approximates forced constant flowing traffic. Often node mobility is not considered [79], because communication happens at a timescale so much smaller that the—relatively slow—node that mobility can often be abstracted from. We took this approach to generate the Matlab figures in Chapter 4.

The opposite of these simplistic mobility models is using real world data, collected from road-side detectors, as used by Kato and Tsugawa in [56]. Although this is realistic traffic, it will be clear that this method cannot be applied to all research, because it is hard to trace individual vehicles (without their consent) over several kilometers with good accuracy.

An alternative that is both flexible and manageable is using a vehicular traffic simulator to supply the mobility input to the network simulator, as performed by Xu *et al.* in [125] where the SHIFT [7] traffic simulator provides input to NS-2 [76] or the work performed by Yin *et al.* in [127] where the output from CORSIM (CORridor SIMulation) traffic simulator [81] is fed into the Qualnet simulator [86].

Another approach is to integrate the node mobility into the network simulator like used by Yu and Heijenk in [128]. Here the OMNeT++ simulator is used together with the Mobility Framework. OMNeT++ and the Mobility Framework are also used in this research. Because of its Open Source nature OMNeT++ provides an easily extendible simulation platform. The TrafficFilter system comprises a complete protocol stack from physical layer to application layer and some cross-layer communication is required. This could easily be facilitated in OMNeT++, as described in Section B.2.

Since we want to test the TrafficFilter it should be tested in both congested and freeflowing conditions. We will have to set a few boundaries. Simulation of the TrafficFilter will require a mobility model that mimics the behaviour of real road traffic. Using the IDM on a one-dimensional one-lane highway we can make a road with traffic on it, as previously discussed in Section 3.5 on page 40.

Nodes will be equipped with an 802.11p PHY and MAC, the microSlotted 1-persistence flooding strategy as proposed in Section 5.3.4 and the TrafficFilter as defined in Section 4.4.

B.1.1 Requirements for Mobility

As presented in [121] traversing 10km of road in a well-connected VANET takes in the order of tens of milliseconds if we consider the most optimal situation, but this might dramatically increase due to contention delay. A vehicle approaching a jam moves at approximately 120km/h = 33.33m/s under free-flow conditions. In 100ms it moves 3.33 meters as indicated in Table B.1.

km/h	**m/s**	**m/100ms**
120	33.33	3.33
15	4.4	0.44
10	2.77	0.27

Table B.1: Distance traversed in a certain time at a certain speed

A wide moving jam, in the mean time, moves 0.44m or 44cm at 4.4m/s as presented in Section 5.1.2. With respect to the short times in which communication can occur (i.e. several tens of milliseconds) the dynamics of vehicular traffic are slow. Even at high speeds a node does not move more than one vehicle-length. In this light it seems reasonable to abstract from mobility when evaluating the performance of the Flooding scheme. The TrafficMap is not used to convey time-critical safety related information and as such we can assume the traffic conditions not to change during the dissemination of one TrafficMap message (i.e. one TrafficMap end-to-end traversal period).

The TrafficMap operates in the order of tens of meters (even more so w.r.t. GPS accuracy). A deviation of 3.33m introduced by not catering for the mobility during simulation seems negligible.

Road Topology

The road is implemented in a circular fashion in the simulator. Vehicles that reach the far end of the simulation area 'wrap' to the other side while their speed and headway stay the same—as it would be if the road were truly round. A certain average density of vehicles (ρ) will be placed based on the following simple formulas:

$$numVehicles = \rho \times roadLength \tag{B.1}$$

and

$$interVehicleDistance = \frac{roadLength}{numVehicles} - vehicleLength. \tag{B.2}$$

We assume the length of a vehicle to be 5m. For instance, for $\rho = 30$ we would find $numVehicles = 300$ and the inter-vehicle distance (e.g. the headway for each vehicle) would be $33.33 - 5 = 28.3m$

At simulation start the nodes will be distributed according to this average inter-vehicle distance with a speed of 0. When simulation starts the vehicles will accelerate according to the IDM. To simulate a traffic jam we introduce a reduced speed zone between 4 and 6km. Vehicles obey the speed limit and will decelerate to meet the new maximum speed. As a result, queuing will occur.

Having a wrap-around road model with respect to mobility does have the downside that the inflow is defined by the outflow. This may become a problem if we are to manipulate the drivers simulated by the IDM but in our simulation we do not study the effects on the behaviour of the driver, we merely require a collection of vehicles moving in a somewhat realistic manner on a stretch of road.

A simulation will run for a bounded amount of time. Because the mobility requires some time to settle to the typical congested pattern we let the model run 300s before starting TrafficMap dissemination.

When communication is involved in the simulation scenarios the most remote vehicle initiates the flood, as a result every flood travels 10 kilometers to simulate the dissimination of a TrafficMap with a virtual horizon of 10km.

B.2 OMNeT++ Discrete Event Simulator

The Objective Modular Network Testbed in C++ — OMNeT++ [55, 80] is an open source discrete event simulator. It is highly modular and uses a high-level language (NED) [113] in which modules can be declared and combined. Using C++ the behaviour of the modules can be provided. OMNeT++ provides simulation models for TCP/IP, Peer-to-peer Networks, Ad hoc, wireless and sensor networks [113].

The OMNeT++ source code can be downloaded from the community website[1]. Because it requires the system to be able to compile C++ files and support a host of advanced functions OMNeT++ may require a lot of packages to be installed prior to compilation. Installation is well documented on omnetpp.org so the reader is referred to instructions there. For this research we used OMNeT++ 3.4b2 installed on Linux 2.6.22 (OpenSuse 10.3).

One thing worth noting is that OMNeT++ uses 'lex', a Lexical Analyser Generator. On Ubuntu (Debian) and OpenSuse this package is not installed by default and the ./configure script does not check for it. During compilation a couple of errors regarding 'lex' show up. Installing 'flex' (Fast Lexical Analyser Generator) present in the repositories enables compilation without problems. A check for 'lex' is included in the configure script but the script finishes successfully even when lex is not found. A hint towards the developers of OMNeT++ is to include a check for a 'lex' compatible utility in the configure script that reports lex is not present and configuration was not successfull. The reason for this is that not having lex installed is a show-stopper about halfway compilation. The developers have been notified of this problem.

After installation it is recommended to run the TicToc Tutorial provided in the /doc directory. This is a simple how-to that shows how to build a simple "network" of two hosts that are continuously bouncing a message back and forth. The manual and API provided in /doc are also very helpful.

Using an editor of choice the user can create NED files providing an architectural structure of the system under test and provide implementation of functions—such as what to do in response to receipt of a message or expiration of a timer. The NED/C++ files of the simulation are then compiled against OMNeT++'s simulation core and GUI. The GUI ('tkenv') provides a straightforward way to run the simulation step-by-step, normally or in fast-forward. A nice graphical representation of the network animates the exchange of messages. For batchmode operation a commanline user interface is also supplied which can simply be enabled by compiling against the 'cmdenv' libraries.

OMNeT++ functions as the core on which the Mobility Framework runs. When implementing a simulation most interaction will be with classes provided by the Mobility Framework. OMNeT++ is not an application in which you build a simulation; it is a set of tools and libraries against which a simulation written in C++ can be compiled. As a result, debugging with utilities such as valgrind[2] or gdb[3] becomes very convenient.

B.2.1 The Mobility Framework

Several frameworks exist for use with OMNeT++ in order to make the generic simulator suited for specific areas of research (WSNs, MANETs, fixed networks). The Mobility Framework is created by the Telecommunication Networks Group at the Technical University of Berlin and is the preferred platform for simulating mobile and wireless networks using OMNeT++. It includes an 802.11 implementation (ad hoc mode only)[66].

[1] OMNeT++ Community website, http://www.omnetpp.org
[2] The Valgrind Tool suite, http://www.valgrind.org/
[3] The GNU Project Debugger, http://www.sourceware.org/gdb/

We used Mobility Framework 2.0p3. Installation was a matter of setting the right environment variables, configure (by means of a mkmk script) and make the components. Most configuration parameters will be read from the OMNeT++ installation.

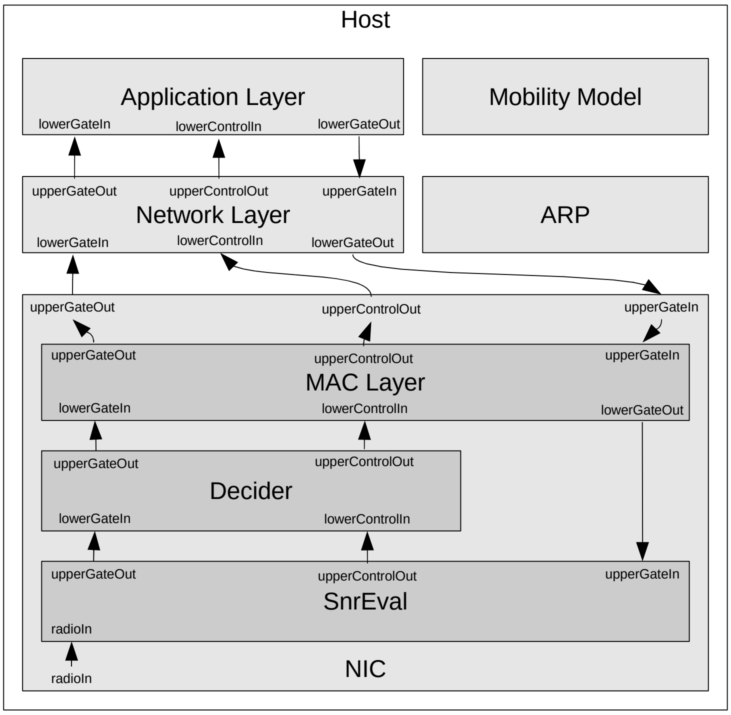


Figure B.2: Modules in the Mobility Framework template

The Mobility Framework is accompanied by extensive documentation in a manual and API reference [66], plus many networks and modules contributed by users which can be useful as examples.

The Mobility Framework runs "on top" of OMNeT++ and provides functionality for node mobility and wireless signal propagation. The geographical simulation area called "playground" in OMNeT++ becomes a bounded area in which hosts can wander around according to a mobility model. Several mobility models are included with the distribution (i.e. constant speed mobility, mass mobility, circle mobility, rectangle mobility etc.). The playground can be configured to be a box (i.e hosts will "bounce" when they hit the wall) or wrap around (i.e. when exiting right they immediately enter left).

The mobility has several effects on the connections; because of the nature of mobility there can not be static connections (although one could reason there could be when the transmission range is larger than the playground). Generally, if the transmission range is smaller than the playground nodes will be moving in and out of range of each other. The ChannelControl module takes care of the connections between nodes. It maintains the position of all NIC modules (the Network Interface Card as defined in the *nic.ned file). Based on these positions ChannelControl can decide which hosts are within range of each other. This does not imply communication is destined to happen, it is merely possible. Two separate modules, Decider and SnrEval, evaluate the success of the actual communication.

Mobility is managed by a separate module derived from the BasicMobility class. This module maintains information such as the position, speed and angle of movement. Furthermore, it enables the host to move by means of a makeMove() method. Using self-messages the host can be instructed to move. The ChannelControl is notified of the new position of the host and it updates its table of the positions of all NICs and, subsequently, the position of the nodes as displayed on the screen when using the Graphical User Interface.

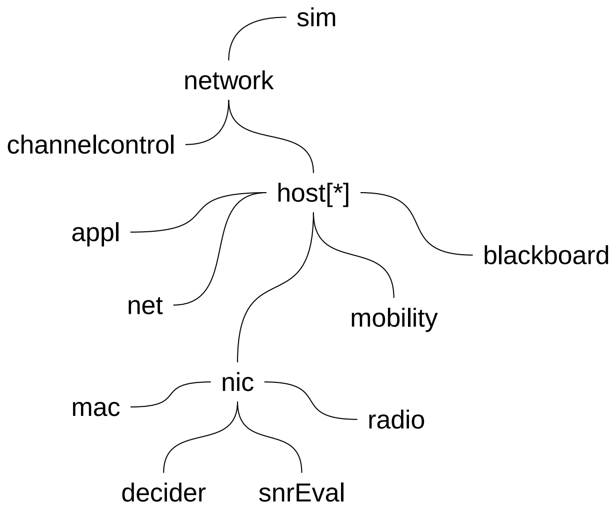


Figure B.3: Logical structure of the Mobility Framework

The Mobility Framework comes with a template, the structure of which is depicted in Figure B.3. This structure is constructed out of modules (see Figure B.2) which can be subclassed. These modules resemble the OSI stack. Implementations of modules contributed by other users are available, such as various Mobility Models and

A Simulation is defined by a network of hosts and a ChannelControl module. A host is defined by application layer, network layer and a NIC. The NIC module groups together the MAC and physical layer. The MAC layer is implemented inside the NIC and decider, snrEval and radio function as the physical layer. The mobility module is responsible for tracking host mobility. The blackboard module is specifically designed for exchanging information "outside of the simulation". These are all one-to-one relations except for the link between the network module and the host module, because a network can contain multiple hosts. The number of hosts is a configuration variable and as such we can easily add more hosts to the simulation.

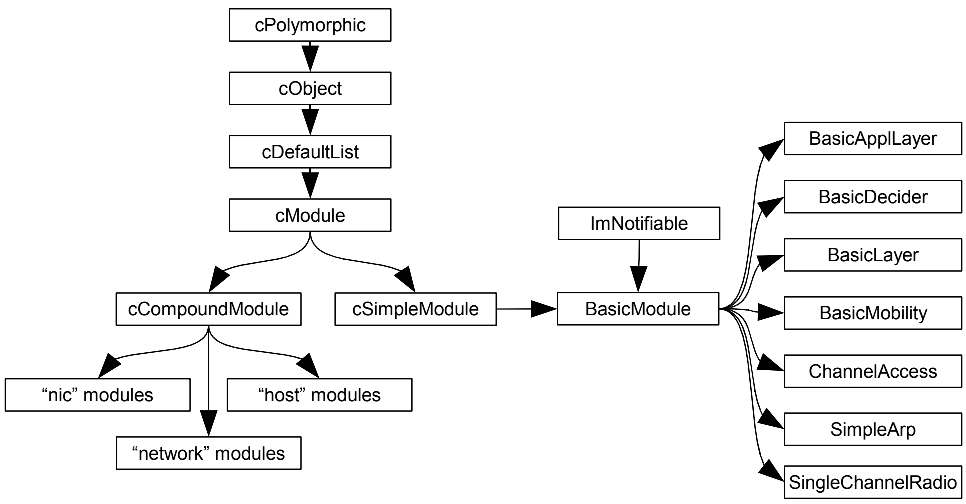


Figure B.4: Inheritance structure of OMNeT++ [113] with the MFw [66].

The class hierarchy within the Mobility Framework is presented in Figure B.4. Most user-classes are derived from the classes that are subclassed from BasicModule. These classes, and the ones subclassed from the cCompoundModule have their counterparts in the logical structure defined in the NED files. Note the multiple inheritance on BasicModule and the use of the Composite Pattern [36] for the cSimpleModule and the cCompoundModule. This structures objects into a part-whole hierarchy. For example, the NIC module is a cCompoundModule while the Mobility Module is a cSimpleModule, but both are cModules. It is this that allows the structuring as shown in Figure B.3. As such every logical module is a BasicModule and also a cModule. We will use this inheritance later on to obtain the speed of the vehicle in front. Class names prefixed by a 'c' are part of the OMNeT++ core, the other classes are part of the Mobility Framework.

B.3 Implementing the Intelligent Driver Model in OMNeT++

The Mobility Framework (MFw) provides a good starting point for building custom mobility models. It is designed for host mobility and a basic 802.11 implementation is provided, which comes in handy. Many mobility models are provided with the MFw, unfortunately these are random and only based on local knowledge; a host is not aware of its environment. The IDM demands a node to be aware of headway and speed difference relative to the vehicle in front. Obtaining these two values for a host is not trivial due to the decentralised mobility implementation in the MFw. The first challenge, thus, is to enable sentient hosts in order to simulate a (human) driver moving the hosts around.

B.3.1 Workaround MFw's decentralised mobility

In the MFw mobility is managed in a decentralised manner: every host decides where to go and at what speed. This is excellent for implementing a mobility model derived from a microscopic traffic model such as the Intelligent Driver Model. A host has knowledge of its own position and speed. But a host is not aware of its environment, it does not matter if there are many or few hosts on the playground, the mobility model works on a per-host basis. It does not matter to the simulator if two hosts occupy the exact same space; clearly this does matter in a vehicular mobility model.

In order to allow the ChannelControl module to calculate which hosts have a probability of communication it keeps track of the geographical position of the host (or the NIC, actually, with the likely assumption that the NIC and the host are colocated). After a move the ChannelControl is updated with the new location. As a result, the ChannelControl is the only module which has full awareness of the positions of all nodes.

In order to allow a host to find out the distance to the vehicle in front and the relative speed we have three options:

- Use a separate message channel to distribute speed/position information
- The Mobility Framework features a so-called Blackboard. This can be used for inter-layer and inter-host communication that does not use the communication channel. In effect, this communication occurs "outside" the simulation.
- Alter the implementation of the Mobility Framework.

Let us now have a look at all three options.

Use of a separate message channel An option is to connect all hosts with a simple and perfect channel so they can exchange the required information. Every node then keeps track of all vehicles or only the vehicle in front.
Pro:

- no changes to the simulator core or Mobility Framework are required
- information exchange will not interfere with the simulation because exchange takes place when "time is stopped"

Con:

- The introduction of an additional communication channel clutters the model
- A Mobility Framework exists for a reason: to facilitate mobility. It makes sense (with respect to design principles) to leave mobility-related information within the framework.

The information is already present within the Mobility Framework, all that is needed is a method to extract this information. Building a separate system to exchange this information results in duplication of a lot of data (especially with a large number of hosts) and a lot more computational overhead. This does not seem very efficient.

Using the Blackboard For this solution we would use the Blackboard facility:

- Every host would subscribe to a mapping of <id,position,speed> for all other hosts
- Every host would publish its own speed information in this mapping whenever a move occurs
- Whenever a change occurs, EVERY subscribed observer is notified.

What this means: 10 times per second a node updates its own position/speed info. The goal is to have in the order of 1000 nodes. 10000 position updates per second per node to provide a complete overview of which we use only part (that would be only the information relating to the lead vehicle: 1 out of 1000 vehicles) seems overkill (besides being very inefficient). It is decided not to use the Blackboard facilities but to settle for a simpler solution.

Alter the MFw The information needed by the Intelligent Driver Model is already present within the Mobility Framework: the ChannelControl keeps track of all NIC positions and every host's mobility module is aware of its own speed. From an object-oriented design point of view it makes sense to obtain required information from the object that manages it. Furthermore, it seems more efficient to obtain only the needed information—as opposed to keeping a global overview—and store the same information only once, as opposed to keeping the same information both in a central location and scattered across all nodes. Synchronisation between the two would incur extra overhead.

With only a few changes to the Mobility Framework we can enable a host to be aware of the distance to the vehicle in front (headway) and the speed of this vehicle. The following steps relate to Figure B.5 and explain the modifications made to the MFw.

1. `IDMMobilityModel` is a `BasicMobility`. As such it has a `ChannelControl* cc`.
2. In `ChannelControl` we add the method `double getHeadway(int id, const Coord* ownPos, double* peerSpeed, bool* leadVehicle)` and call this from `IDMMobility: cc->getHeadway()` `peerSpeed` is a pointer to a location where after the method returns we expect the speed of the vehicle in front and `leadVehicle` is a pointer to the boolean which will be set to true if this vehicle is the next vehicle to wrap[4].
3. `ChannelControl` holds a mapping of NICs[5]. First the ownPos is mapped to the internal mapping of the NIC registry:

```
cellX = static_cast<unsigned>(ownPos->x/findDistance);
cellY = static_cast<unsigned>(ownPos->y/findDistance);
```

[4] This information is used to initiate the flooding, see Section B.5.1 which explains how the flooding is initiated.

[5] This mapping includes possible interferers but excludes hosts certainly out of interference range. This is an optimisation in the MFw; only the nodes that have a probability of being influenced by a transmission will be evaluated.

4. We iterate over NICs in this cell, resulting in a pointer to a `NicEntry` related to the host ahead of the current host. This `NicEntry` holds the exact position of the host in front. The difference of ownPos and the position of the peer is a double value we will return later on. `NicEntry` also holds another interesting attribute: the `nicPtr`. This points towards the cModule that this `NicEntry` belongs to.

5. We are now going to navigate cModules by their pointers: `parentModule()` returns the Host module to which the NIC module belongs as defined in the logical structure in the NED files (see Figure B.3).

6. Now that we have the Host module, we can ask for its mobility module: `moduleByRelativePath("mobility");`

7. We are now at the peer's mobility module. This module holds a `HostMove move` object. We do, however, run into a problem here: `move` is protected and thus not accessible! At this level we are dealing with a cModule. Casting this to `BasicMobility` allows us to alter `move`'s status to public. An alternative would be to cast to `IDMMobilityModel` and write a `public double getSpeed()` which has access to the protected move object. It was considered better not to introduce own classes into the Mobility Framework (as would be required to perform the cast to `IDMMobilityModel`), so `move` was changed from protected to public.

8. From `move` we can request the speed attribute, and store this in `&peerSpeed`. The `getHeadway` function can now return.

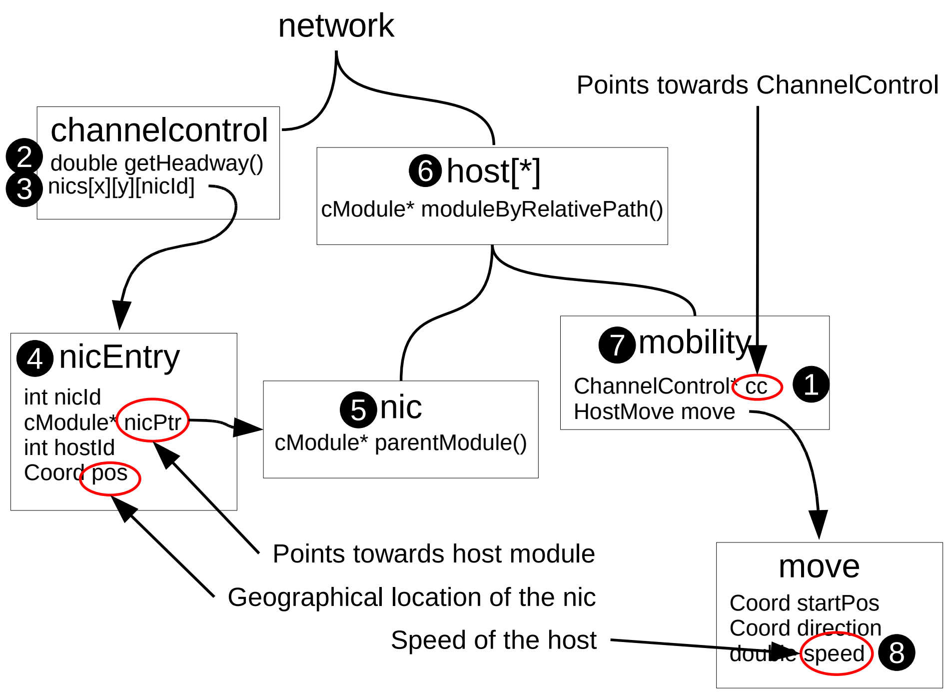


Figure B.5: Traversing the template

Basically, we alter the Framework in two places: add a getHeadway function to ChannelControl and in BasicMobility we switch move from protected to public. No other modifications are required. We then recompile the Mobility Framework.

A vehicle can now obtain the distance to the vehicle in front, and the speed of this vehicle. As a result, we have all ingredients for the IDM.

B.3.2 The IDMMobilityModel

The class IDMMobilityModel is responsible for making each host move in a way prescribed by the Intelligent Driver Model. Although the IDM is a continuous model it can also be used in a discrete application, if the timesteps taken are small enough and we ensure speed does not cross 0 into the negative domain, as expressed by Equation (3.6) on page 41. We use timesteps dt of 0.1s, which means that ten times per second every node evaluates its current speed.

Nodes start with speed 0.0 and a self-message is scheduled for simtime()+dt. When the simulation time starts a node 'accelerates': the `virtual void handleSelfMsg(cMessage *msg)`-method receives the self-message. When this self-message is received the makeMove() method is invoked.

The makeMove() method first obtains the headway to the vehicle in front and the speed of this vehicle by calling `cc->getHeadway()` which was modified as described in Section B.3.1. When the headway and the speed of the vehicle in front are known they can be applied to the IDM formulas as presented in Section 3.5.1. This gives a new speed which, multiplied by the time dt results in the distance covered in dt, stored in stepTarget.x. Because nodes are only moving in the x-plane (y remains constant) we only need to alter the x-component of the position.

The speed limit is enforced by means of `v0`, and based on the position this is set to the free-flow maximum speed or the reduced speed specified for the jam. Next some bookkeeping ensures a node does not wander off the playground but wraps around. Finally the ChannelControl (and possibly the GUI) are updated on the new position of this host.

After the makeMove() has been executed, handleSelfMessage schedules a new self-message at simtime()+dt.

```
void IDMMobilityModel::makeMove(){
    double peerSpeed;
    double headway = cc->getHeadway(hostId,&stepTarget, &peerSpeed, &leadVehicle)-veh_len;
    double deltaV = move.speed - peerSpeed;

    //we may now decide to decelerate or accelerate:
    move.speed=move.speed + dt*(accelerate*( 1 -
                              pow( ( move.speed/v0 ),delta ) -
                              pow( ( (s0 + T*move.speed +
                                   ( move.speed*deltaV )/sqrtab ) /headway ) ,2 )
                                       );

    //move a bit, assuming exactly dt-seconds have passed:
    stepTarget.x = move.startPos.x + dt*( move.speed / 3.6 ) ;

    // update position
    move.startPos = stepTarget;

    //check and obey speed limit:
    if(move.speed<0){move.speed=0.0;} //no negative speeds!
    if(move.startPos.x > reducedSpeedZoneBegin && !speedChanged){
        v0 = reducedSpeedZoneSpeed;
        speedChanged = true;
    }
    if(move.startPos.x > reducedSpeedZoneEnd && speedChanged){ //exit reduced speed zone
        v0 = par("v0");
        speedChanged = false;
    }
```

```
    //do some bookkeeping:
    fixIfHostGetsOutside();
    updatePosition();
}
```

B.3.3 Testing of the IDM implementation

We test the IDM implementation by comparing the behaviour of the vehicles to that of the vehicles in the Matlab model. We use the same settings in both models and compare the resulting speed/position relations. We use equal road length (10km), number of nodes and equal starting positions. Nodes are placed with equal inter-node distance. The start speeds are 0 km/hour and a reduced speed zone exists between 4 and 6km. Ten times per second the IDM evaluates the present headway and speed and adjusts the speed for the next move if necessary. The simulation is run for 200s before evaluating the state of the vehicles on the road in order to stabilise. The initial acceleration of the nodes is over and the position and speed of a node are determined by the IDM as configured according to values used by Treiber *et al.* in [102].

The starting positions are deterministic, as is the Intelligent Driver Model because no random variables are introduced. It is to be expected that repeatedly running the model results in the same results over and over. This is the case for the OMNeT++ implementation of the IDM, but not so for the Matlab implementation.

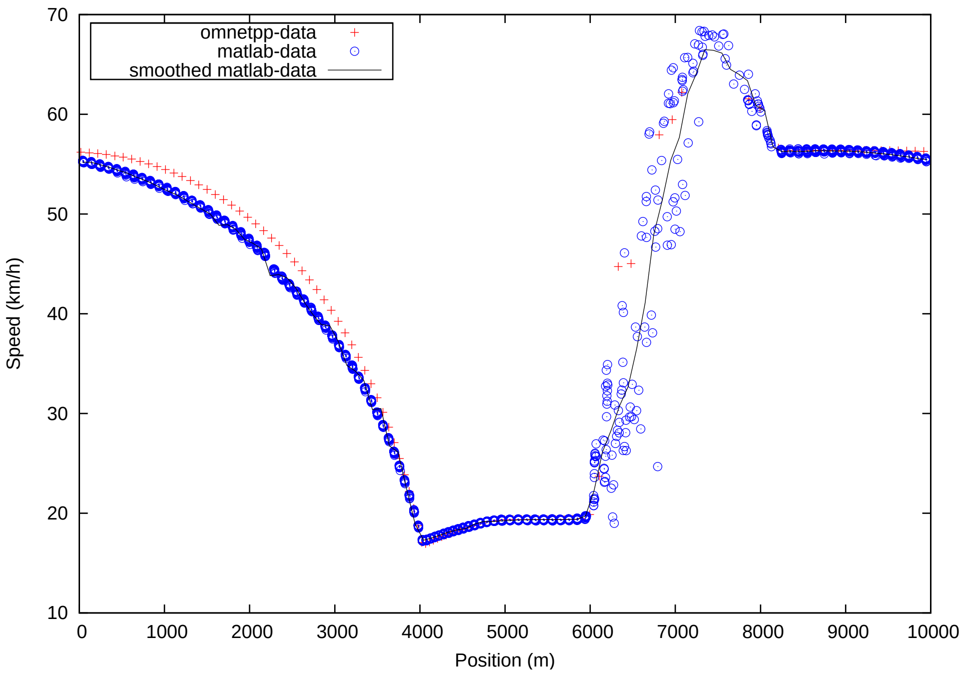


Figure B.6: Comparison of the Matlab and OMNeT++ IDM implementation

In Figure B.6 one run of the OMNeT++ model (fully deterministic) is compared to 20 runs of the Matlab model and the smoothed version of these 20 runs. The deviations in the Matlab model can be attributed to two design decisions which—at first—did not seem very important:

- Speeds of vehicles are kept in an array. The index of the speed is the vehicle's present location. This means: positions are rounded to integers, which can result in gross loss of precision if the mapping is done over and over again (and it is, in fact, a recursive error). By averaging a large number of runs we can circumvent this error by averaging the results.

- The properties of IDM vehicles are based on their predecessor. The current vehicle evaluates the speed and position of the vehicle in front after it has been updated. In the OMNeT++ model vehicles are addressed by HostID so the relative order of evaluation is always the same. In the Matlab model, the road array is simply traversed backwards (from `road[road_len]` to `road[1]`), hence a vehicle is addressed based on its geographical position. At first glance this does not seem to be a problem, but when a vehicle leaves the end of the road (and enters the beginning because we use a circular road) its order of evaluation is altered! It is now not evaluated prior to movement but after movement of its predecessor. This explains the small ”dip” in the Matlab plot around 2400m.

When averaging multiple matlab runs the results converge towards the results of the OMNeT++ IDM implementation. A small deviation is still observed but this is deemed acceptable, as we have not validated either of the IDM implementations.

Conclusion

From the data and observations it follows that the IDM implementation in the IDMMobilityModel in OMNeT++ exhibits the same behaviour as the IDM implementation in Matlab, and both models show the expected behaviour. Because of small errors in the Matlab model, the OMNeT++ model can even be said to be a better implementation. These errors, however, do not grossly alter what the model is supposed to do: generate “realistic” driver behaviour. Because no traffic model can capture true realistic driver behaviour and the IDM (by the very nature of a model) is an abstraction we conclude that the mobility model in the simulator meets the demands. The implementation of the IDM in this research is not calibrated with measurements derived from real-life traffic, as this would require more time than was available. It should also be noted that the implementation of the IDM is simply used to obtain (position,speed) data of vehicles and as such is not guaranteed to result in realistic results when a feedback of information derived from communication is applied to the host mobility. This research also refrains from attempting to alter the vehicle’s mobility pattern based on observations made of the vehicle’s behaviour.

B.4 Implementation of the TrafficMap Message

This section explains how the trafficmap message defined in Section 5.2 is implemented. The Mobility Framework provides a template which resembles the OSI stack: Physical, MAC, Network and Application layer. The protocol entities in these layers communicate by passing messages just like their OSI counterparts. Messages from upper layers are encapsulated and sent down to the lower layer. This encapsulation goes all the way to the physical layer: the Airframe. The Airframe models the electro-magnetic signal propagation.

In OMNeT++ messages extend the cMessage class. This class is used for a number of things in the simulator: events; messages; packets; frames and any kind of other entity traveling in a network. A cMessage has several attributes of which the following are most important:

name – used in graphical representation

kind – a message type used to identify the kind of message and what fields are available

length – used to calculate the duration by means of a bitrate

bit error flag – indicates the message is not correctly received. Simulates bit errors by means of a probability of $1 - (1 - BER)^{length}$.

timestamp – tells the simulator when to schedule the event / reception of message / expiration of timer etc.

Fields can be added by simply subclassing cMessage or one of its derivatives such as ApplPkt, NetwPkt, MacPkt and AirFrame.

Messages can be sent to other modules (such as from an application layer entity to a network layer entity) or can be sent to a module itself. These so-called self-messages can be useful as timers. For instance, the τ-timer introduced in Section 5.3 is modelled as a self message which is scheduled one MIT period into the future. When the scheduled time has arrived the scheduler delivers the message to the designated module (the same as the one that scheduled the message in case of a self-message) and appropriate action can be taken by the module; for instance a time-out could have occurred. Messages can also be cancelled or rescheduled to a different time.

When using messages to model packets, encapsulation can be used. This, for instance, allows encapsulation of an ApplPkt in a NetwPkt. The scheduling of a message (for instance, an Airframe which models propagation through the air) is usually done with a delay which reflects propagation or processing delay. This way certain kinds of delay can be modelled and messages do not arrive instantaneously.

Messages are defined in C++, but a message definition language is provided which tells the simulator core the contents of custom messages. This eases the creation of messages and saves a lot of work. The C++ code is generated from these definitions. A message definition looks as follows [113]:

```
message MyPacket
{
   fields:
      int srcAddress;
      int destAddress;
      int hops = 32;
};
```

This is then translated to C++ code prior to compilation and compiled into a message object of type MyPacket. This automated mapping works for simple types, but when using Standard Template Library (STL) types such as vector the keyword `abstract` can be used in the message definition, and an own implementation (perhaps by a simple vector but maybe even exotic constructs like a semiSortedSet) can be provided. Like the rest of OMNeT++, the message definition provides a lot of flexibility.

B.4.1 Application Layer Packets

The TrafficMap message is declared in the TFPkt.msg file. TFPkt extends the ApplPkt class defined by the MFw and adds a few extra fields:

```
abstract double TMposX[];
abstract double TMposY[];
abstract double TMheading[];
abstract double TMspeed[];
```

Like explained above, the abstract fields tell the simulator core that we are dealing with a type of list of which the implementation is not standard in OMNeT++. The implementation of these user-defined datatypes is given in TFPkt.cc; they are simple vectors the contents of which can be accessed by means of get and set functions. The number of elements in the four arrays is always equal (in this application), and is passed on to the network layer when the message is being sent down.

Notice the use of doubles for the values in the lists. This is because OMNeT++ uses doubles to express these values and they are used without mapping for consistency with the simulator environment.

B.4.2 Network Layer Packets

A message TFNetwPkt is defined which extends the NetwPkt message class. The custom fields defined are:

```
double posX = 0;
double posY = 0;
unsigned long floodID =  0;
unsigned char hopCount = 0;
unsigned char TMsize = 0;
```

The posX and posY denote the position of the source. The floodID is used to identify to which flood this packet belongs. It is simply copied from the received message when propagating a flood, or generated randomly when a new flood is initiated. The hopCount is described in Section 5.2 and the TMsize reflects the number of samples in the encapsulated TFPkt.

B.4.3 MAC and Physical Layer Packets

The packets used in the MAC and PHY layers are those provided by the Mobility Framework: Mac80211.msg and AirFrame80211.msg. These are unaltered and are used to encapsulate higher-level packets. We will not further discuss these here.

B.5 Implementation of the Flooding Strategy

The flooding scheme relies on an estimated transmission range. It is assumed a node is capable of estimating its own transmission range. Due to the nature of radio wave propagation it is hard to make good predictions of the transmission range, the best we can do is derive a general estimate. When looking at the way the timeslots are allocated in the slotted schemes used by Wisitpongphan *et al.* we make a couple of observations, illustrated in Figure B.7:

- A good estimated transmission range is key to obtain a good slot allocation:
  - If we over-estimate we end up with a system that hardly (or not at all) uses the first slot because nodes are less likely to correctly receive a message. This will result in a longer end-to-end delay, because rebroadcast will always have to wait for one (or more) slotTimes.
  - If we under-estimate we end up with many nodes in the first slot (because of the $min(D_{ij}, R)$ in Equation (5.2) on page 79 the domain is open on the far side) which results in a crowded first slot in which collisions have a high probability of occurring. This would mean for every hop we incur extra delay plus occupy the medium without doing anything useful.
- The estimated transmission range influences the slotSize, and hence affects the number of vehicles per slot. An over-estimated R results in more interferers per slot.
- When vehicle density is high there are many vehicles per slot. The slotted nature synchronises vehicles in clusters. This is expected to result in an increased number of collisions between vehicles that are in the same slot under high node densities. On page 81 the microSlotted scheme was proposed to counter this.

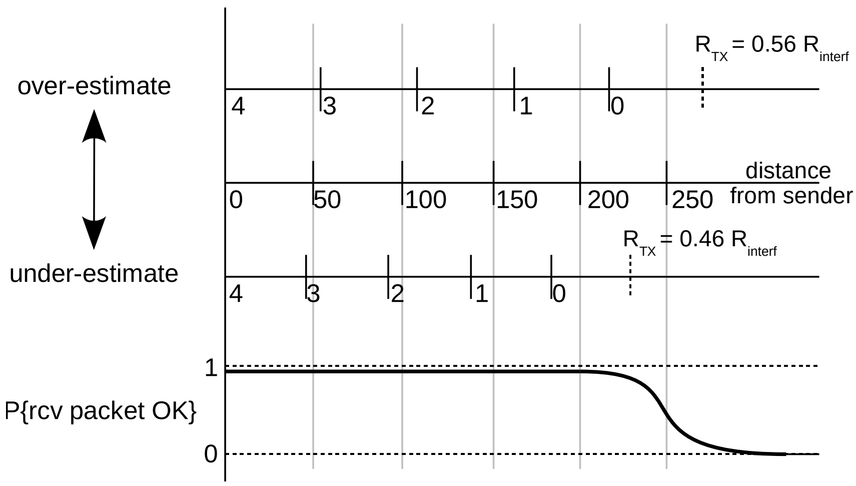


Figure B.7: Estimating the Transmission Range has effect on slotsize and number of nodes per slot.

The estimated interference range is calculated in the ChannelControl module in the initialisation phase of the simulator in order to evaluate radio signal propagation and bit errors. The TFNetwLayer makes use of the IDMMobilityModel via its `myMobility*` pointer. The mobility model has a `ChannelControl cc`. TFNetwLayer can now access the transmission range calculation via `myMobility->cc->calcInterfDist()`. One problem: this function is protected.

ChannelControl has been altered before (for the `getHeadway()` function) so we make another change; set calcInterfDist() to public, and recompile. However, cc is protected within BasicMobility. For now this has been solved by making it public, which is not the neatest option.

The interference range is calculated in the Mobility Framework as follows:

```
waveLength      = (speedOfLight/carrierFrequency);
minReceivePower = pow(10.0, sat/10.0);
interfDistance  = pow(waveLength * waveLength * pMax / ( 16.0*M_PI*M_PI*minReceivePower ), 1.0/alpha);
```

This is, in fact, the Friis Free Space propagation model [35]. For a mathematical derivation refer to Appendix C.1 and C.2. We aim for an effective transmission range of 250 meter which results in an interference range of about 500m judged by the rule of thumb that interference range is generally twice the transmission range. In [124] it is shown that for an open space environment the interference range of a receiver is approximately 1.78 times the transmission range. Hence, if we find the interference range we state:

$$R_{trans} = 0.56 R_{interf} \tag{B.3}$$

because

$$1 : 1.78 \sim 0.56 : 1$$

We find the required transmission power as follows, based on the Friis formula derived in Appendix C.2:

$$R_{interf} = \left(\frac{\lambda^2 \times P_t}{(4\pi)^2 \times P_r}\right)^{\frac{1.0}{\alpha}} = \left(\frac{\left(\frac{3.0\times10^8}{5.87\times10^9}\right)^2 \times P_t}{(4\pi)^2 \times 10^{-12}}\right)^{\frac{1.0}{3.5}} \tag{B.4}$$

Solving for $R_{interf} = 500m$ we find $P_t = 168.97mW$. This is the transmission power that will be used in all simulation runs. In order to achieve a 1km range (as achieved by Wisitpongphan *et al.*) power levels which by far exceed the maximum as defined by the standard are needed (i.e. 21630.099mW = 21.6W according to the Friis formula). In order to fully verify the work of Wisitpongphan *et al.* we would need to use the same propagation model. The extreme deviation can be explained by the fact that Wisitpongphan *et al.* use 802.11a which uses OFMD and the Mobility Framework uses DSSS (802.11b). The Friis Free Space Propagation formula used in the MFw to calculate the interference range works as long as the bandwidth is narrow enough to use a single value for the wavelength [35]. This might result in an error when using a bandwidth of 20MHz (802.11b,a) of 10MHz (p) but may also render the Friis formula worthless when evaluating OFMD propagation.

A solution would be to build a correct OFDM propagation model for 5.9GHz IEEE 802.11p communication using 10MHz bandwidth, but this would take more time than is available. It is decided to use the propagation model as an abstraction, we have a maximum interference range and a means to determine correct reception (the snrEval and Decider modules) and will use these. The result is that we cannot make any hard conclusions where radio signal propagation is concerned.

B.5.1 Testing of the Flooding Scheme

In order to be valid our implementation of the slotted 1-persistence flooding scheme should reproduce the results of Wisitpongphan *et al.*. As mentioned in the previous section, we are using a different propagation model which makes a direct comparison difficult. In order to make a comparison we need:

- equal length of road
- equal number of nodes
- equal distribution of nodes on the road
- equal transmission properties (power, pathloss)
- equal size of broadcast packet: 25kb.

As discussed before, due to the propagation model used by the MFw we settle for a transmission power that gives an effective transmission range around 250 meters, which is a reasonable practically achievable transmission range.

The size of the broadcast packet used by Wisitpongphan *et al.* is rather unrealistic for the TrafficFilter application where packet size is envisioned to be in the order of 1 Kbyte in order to fit into one broadcast packet in line with the messaging nature of the communication. The maximum packet size will be 2312 octets as defined by the IEEE 802.11 standard [48].

Although direct comparison is not possible, we can make some observations:

- End-to-end delay on a (connected) network is in the order of tens of milliseconds, just as reported by Wisitpongphan *et al.*
- End-to-end delay greatly depends on the timeslot in which a rebroadcast is performed and the number of hops.
- Propagation sometimes stops in the head of the traffic jam. Reasons for this are clusters of consecutive collisions and gaps larger than the transmission range. The latter occurs beyond the head of the traffic jam, where vehicles rapidly accelerate and local vehicle-density rapidly declines. A workaround would be to evaluate the flooding scheme without mobility, this way we can control the maximum inter-node distance to be less than the transmission range. This allows us to study the flooding mechanism in isolation from the mobility-induced gaps.

The IDMMobilityModule is instrumented with a parameter ”mobEnabled”, which is a boolean value used to toggle whether the vehicles should move according to the IDM or remain static. We can now study the flooding strategy without mobility, as described in Section 6.2 on page 94.

The mobEnabled switch simply en- or disables certain functionality in the simulator model, like shown below:

```
if(mobEnabled){
    scheduleAt(simTime() + dt, msg);
}
```

These switches can be set in the configuration file. This makes the simulator generic for evaluation of the system with or without Mobility, microSlots and SNP.

Hosts are initially placed with an inter-vehicle spacing drawn from a uniform distribution between $\frac{1}{2} interVehicleDistance$ and $1\frac{1}{2} interVehicleDistance$, to meet the required vehicle densities without introducing gaps.

Without node mobility we can evaluate the flooding strategy in isolation, without any mobility-incurred effects. In order to make a good evaluation between the slotted 1-persistence flooding and our own addition of the microSlots another variable was added to the model in order to toggle the use of microSlots by means of a configuration option. This functions the same as the mobEnabled switch. Setting microSlotsEnabled to true in omnetpp.ini results in the use of microSlots when scheduling the rebroadcast.

A flood is initiated by the lead vehicle (i.e. the vehicle which has the largest x coordinate). This information is obtained by a vehicle when it calls the `getHeadway()`-function (part of the alteration of the MFw described in Section B.3.1). The reasoning behind this is quite straightforward: the most remote vehicle launches a flood, which then propagates upstream. The floods are launched every MIT seconds (MIT was defined as 3.0s in Section 5.1.2). Every host is equipped with a timer which counts down from MIT to 0. Upon expiration a vehicle evaluates whether it is the lead vehicle. If it is, a new flood is launched.

The expected results of the microSlots were the following:

- a reduced number of collisions
- a small increase in the link load, as the transmissions in a slot are now not executed in parallel (resulting in collision) but are serialised by the MAC layer. It is expected that, because hosts in the first slot successfully rebroadcast and timers will be cancelled in subsequent timeSlots, the increase in link load will be of a bursty nature.
- a small extra incurred delay of several microSlot-times (which are in the order of DIFS-periods).

The Source Node Priority (SNP) as introduced in Section 5.3.3 alters the allocation of timeSlots in the flooding strategy. Just like the microSlots, SNP can also be toggled by means of a configuration parameter in omnetpp.ini.

The flooding strategy was implemented in the network layer (module TFNetwLayer). Upon reception of a message on the radio interface this is passed through the Decider and snrEval to the MAC layer. This, in turn, hands it over to TFNetwLayer. The FloodID is stored and the contents are presented to the application layer (TFApplLayer). The application layer, after having processed the TrafficMap information, sends a message down to the TFNetwLayer. If the message received from the application layer is part of a flood (and hence is to be propagated) the TFNetwLayer forwards the message to the smartFlood function, as depicted below.

```
void TFNetwLayer::smartFlood(cMessage* p, bool sampleAdded){
    bufferedPacket = p;
    double timeToSlot;
    if (senderPosX!=NULL){
        distanceToSender = fabs(myMobility->move.startPos.x - senderPosX);
        if(snpEnabled){
            if(sampleAdded){
                slotNo=0;
            }else{
                slotNo=1+(numSlots-1)*(1-( min( distanceToSender , estTransmissionRange)/estTransmissionRange ));
            }
        }else{
            slotNo=numSlots*(1-( min( distanceToSender , estTransmissionRange)/estTransmissionRange ));
        }
        timeToSlot=slotNo*estOneHopDelay;
        if(microSlotsEnabled){
            timeToSlot+=microSlot();
        }
    }
    senderPosX=NULL;
    if(!floodTimer->isScheduled()){
            scheduleAt(simTime() + timeToSlot, floodTimer);
    }
}
```

Smartflood determines when to schedule the rebroadcast of this flood based on whether a sample has been added by the TFApplLayer and configuration variables snpEnabled and microSlotsEnabled. A slot number is determined and in case of microSlotsEnabled some additional delay is added. Then a floodTimer is scheduled. When this timer expires the message is sent down to the MAC layer.

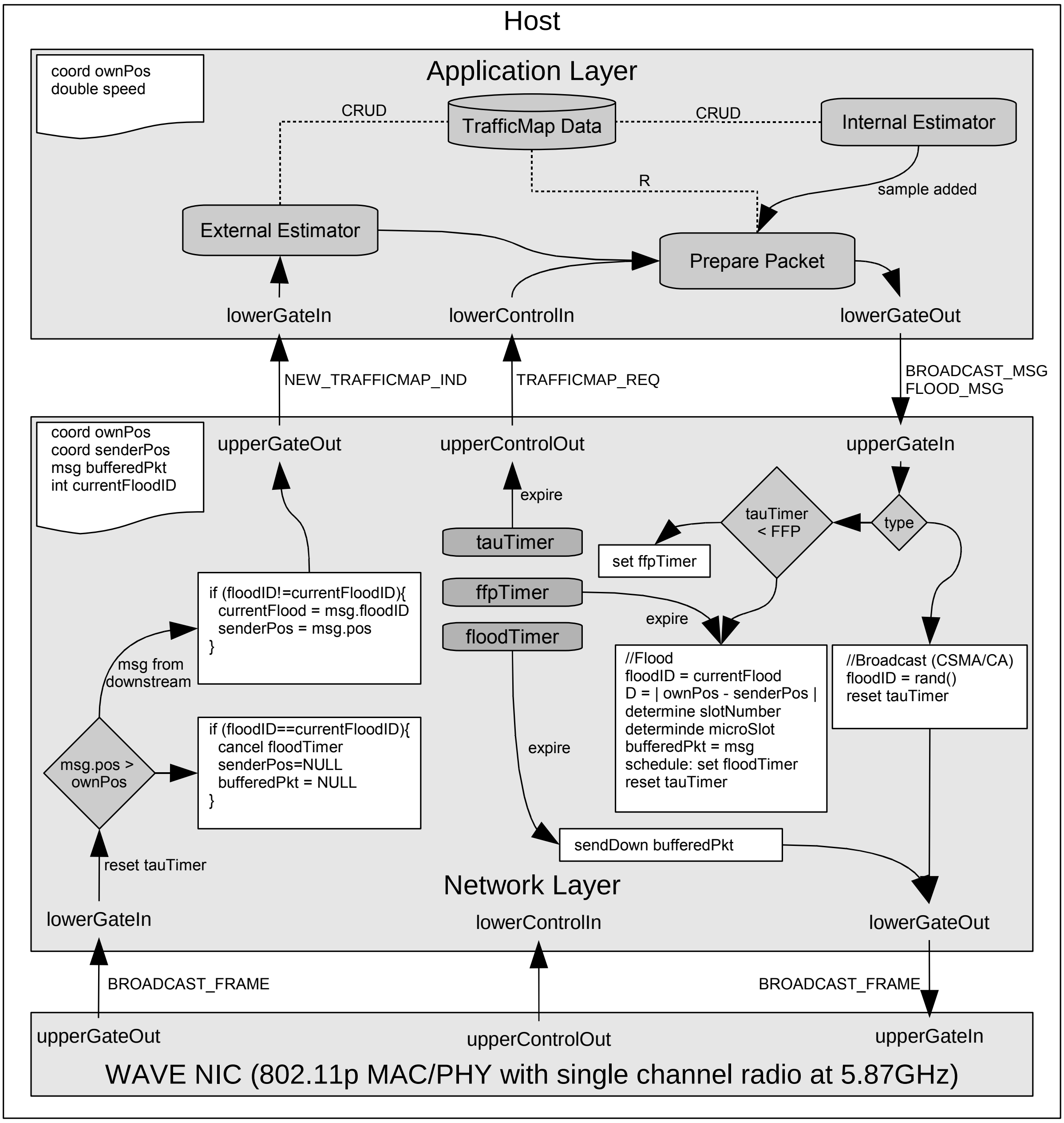


Figure B.8: TrafficFilter: The complete design

B.6 Implementing a means to judge TrafficMap information

In the previous section the design of the complete TrafficFilter system was presented. This section is concerned with determining the quality of the received information. An OBU in a vehicle constructs a TrafficMap based on information which has been in transit for a while. The goal of this research is to supply ITS applications with an over-the-horizon view. In order to evaluate if the collected data is a good representation of reality an abstraction will be made: the TrafficMap and the actual situation are considered as collections of points in a two-dimensional (position,speed) plane. The distance between correlated points is a measure of how well the TrafficMap represents reality.

B.6.1 Evaluating the quality of the information in a TrafficMap

Consider two collections of points:

$TM \ = \{(position, speed)\}$, TrafficMap contents of the node closest to position 0 at time t.

$ACT \ = \{(position, speed)\}$, the actual situation at time t when TM is captured.

A sample in TM is a representation of the traffic flow speed at that point, so the position information needs to remain unchanged. We define a test to determine how well a TM represents the ACT.

Consistency Test: How well does the interpolated version of TM match ACT?

Or in other words, how well can the original signal (ACT) be derived from its compressed representation (TM)?

This test determines how well a sample in TM matches ACT. We parameterise TM around the position of a point in ACT which is currently under evaluation. This interpolates a line between the two enclosing samples in TM as presented in Figure B.9. Note, however, that the position-component of S_i or S_j may also be equal to the position component of X.

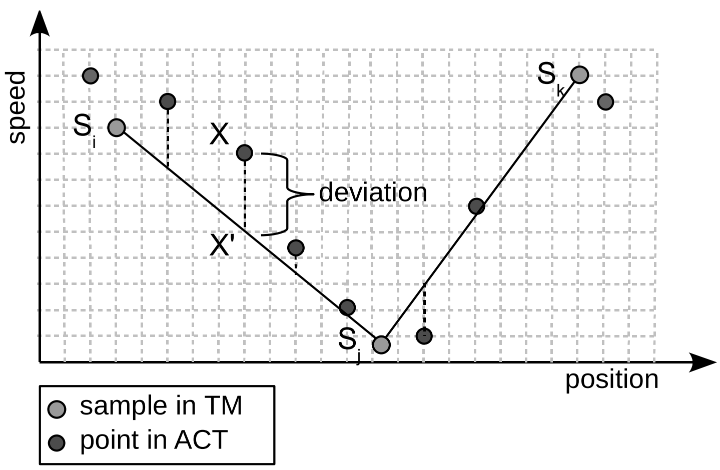


Figure B.9: Consistency Test: mapping ACT to TM

$$S_i, S_j \in TM, X \in ACT \text{ and } S_i^{pos} \leq X^{pos} \leq S_j^{pos}.$$

This last statement may not hold for the first and last items in ACT and TM (the edges of the playground). In this case a wrap-around feature and a correction is required:

$$S_i^{posmax-road\_length} \leq X^{pos} \leq S_j^{posmin} \text{ (at the beginning)}$$

$$S_i^{posmax} \leq X^{pos} \leq S_j^{posmin+road\_length} \text{ (at the end)}$$

Line (S_i,S_j) can be parameterised as:

$$S_i + t\mathbf{v}$$

where $\mathbf{v}$ is the vector from S_i towards S_j and t can be derived as:

$$t = \frac{(X^{pos} - S_i^{pos})}{(S_j^{pos} - S_i^{pos})}$$

t is a measure for where point X lies between samples S_i and S_j. t runs from 0 (equal to S_i) up to 1 (equal to S_j).

$$X' \text{ (the projection of } X \text{ to line } (S_i, S_j)) = (X^{pos}, S_i^{spd} + t(S_j^{spd} - S_i^{spd}))$$

as a result the deviation can be found as $X'^{spd} - X^{spd}$:

$$dev = (S_i^{spd} + t(S_j^{spd} - S_i^{spd})) - X^{spd}.$$

This can be rewritten to read:

$$dev = S_i^{spd} + \left(\frac{X^{pos} - S_i^{pos}}{S_j^{pos} - S_i^{pos}}\right) \times (S_j^{spd} - S_i^{spd}) - X^{spd}.$$

The points in ACT and TM can be plotted as shown in Figure B.10. Of the points in TM two interpretations are given: the RLE-interpretation which is the idea on which the TrafficFilter is based (the step-interpretation resembles the sudden crossing of a threshold) and the simple connecting of points. The latter corresponds to a simple parameterisation of a line between two points, as described above. It is reasoned that, without going into too much sophistication, the shortest connecting line between two points forms a good average of all possible connections between the two.

This reasoning is backed by the fact that, given two points $S_n, S_m \in TM$ it holds that (if the capture thresholds in the TrafficFilter function properly and no *lost updates* occur) all values from ACT which lie between S_n^{pos} and S_m^{pos} are in the rectangle which is defined by S_n and S_m. This line is shown as “TrafficFiltered samples” in Figure B.10. This Figure shows large errors between two samples in TM but small errors around samples in TM, shown as the “actual error”. This is because, in order to add a sample to TM, a threshold must be exceeded. Within the upper and lower thresholds the points in ACT can fluctuate. If a point in ACT crosses the threshold a new sample is added to TM, setting a new “baseline”. A good example of this can be seen in the peak after the jam around 6100m in Figure B.10. This error is positive when the information in TM is an overestimate of ACT and is negative when TM is an underestimate of ACT.

Figure B.10 also shows the “sampling error” which is calculated in a similar fashion as the consistency test above, but then parameterising X_i and X_j and finding the deviation between S and S'. The results of this test are so close to 0 they are negligible.

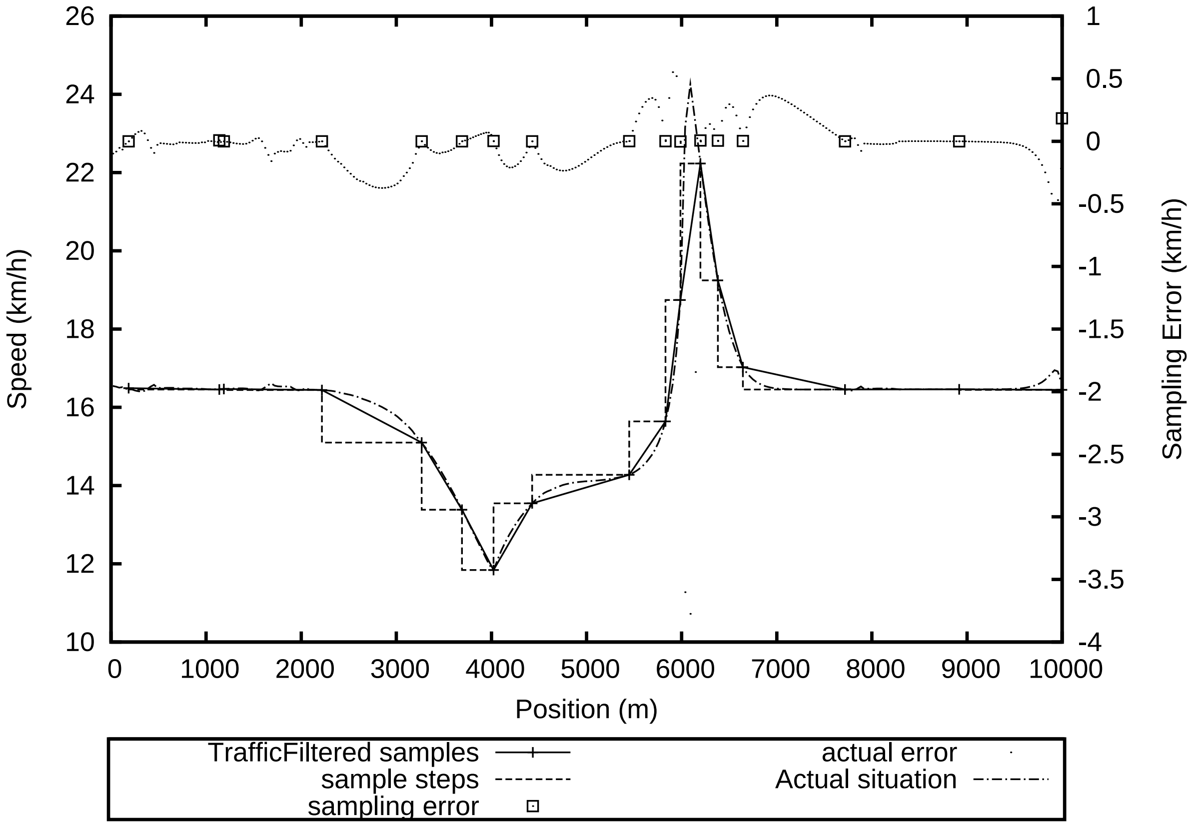


Figure B.10: A flood with $\rho = 30$

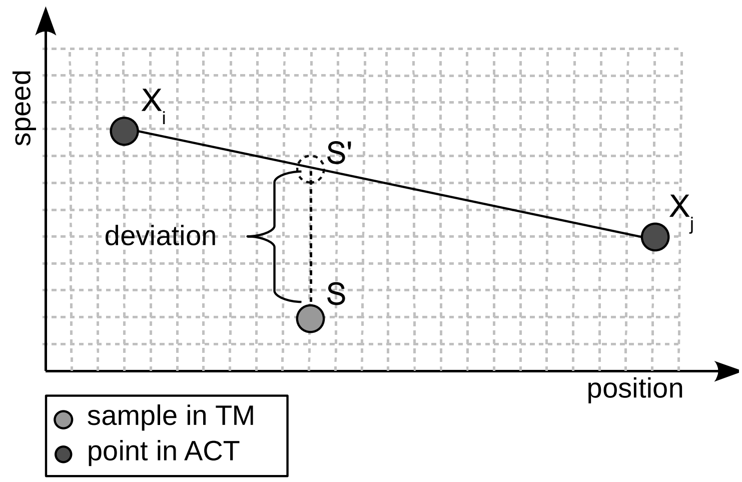


Figure B.11: mapping TM to ACT to derive an indicator of sampling error

B.6.2 Discussion of the evaluation method

In general the Sampling Error is small (most are around 0). These errors are introduced by dynamics in the traffic during the $\sim$ 80ms a message requires to propagate end-to-end at $\rho = 30$. We would like to know the deviation between (position,speed) points in ACT and the interpolation between two samples in TM, so we take the sum of the absolute deviations per sample and divide by the number of samples to derive an average deviation for every flood.

The actual error can become quite large, especially when there are few samples in the TrafficMap or when speed rapidly in- or decreases but stays within the thresholds as is the case around the peak at the head of the traffic jam. If a threshold is exceeded a new sample is added to TM.

The distance-based component in the ε-function as defined in Section 4.3 guarantees a minimum number of samples. If a node receives a TrafficMap message and the distance to the last sample is over a kilometer it is allowed to add a new sample, regardless of the speed-threshold based component. This ensures that, at least every kilometer, the Actual Error visible as the dotted lines connecting the Sampling Error boxes in Figure B.10 is reduced.

These errors are all calculated at the moment of reception by the most remote vehicle. It takes at most 3 seconds before a new TrafficMap message arrives. How the TM-ACT relations behave over those three seconds has not been researched and is left as future work.

B.7 Instrumenting the Simulator

OMNeT++ provides three means to extract information from the simulation: scalars, vectors and snapshots. These are stored in separate files: omnetpp.sca, omnetpp.vec and omnetpp.sna. Here we briefly explain how this can be done.

scalars – values such as the number of packets received at time t

vectors – time-series data such as the buffer occupancy over time

snapshots – an overview of all (or registered) variables of the complete or part of the simulator at time t

The snapshot infrastructure is used to obtain an overview of the positions of vehicles at a certain time. The vector output is not used. This research relied most heavily om OMNeT++'s scalar output system.

The Scalar output facilities output a line according to the following syntax:

```
scalar  [moduleName]  [variableName]  [variableValue]
```

For instance:

```
scalar  "TFSim.host[266].appl" "myPos" 165.311371255
```

This means that host 266's application module has output the scalar variable named myPos which has value 165.311371255. A value is written to the scalarfile by means of the following command:

```
recordScalar("myPos", myPos);
```

OMNeT++ is Open Source, as such it is possible to adapt the simulator anywhere in order to extract information from it. An example is the method getHeadway() which has been added to the ChannelControl class. Another example is an adaptation to the use of the Scalar facilities.

What if we want to output more, related, information? In the next example we print the value of the fourth entry in the internal TrafficMap. A TrafficMap entry has both a position and a speed. We encapsulate the position information (and the fact that this is the 4th sample) in the variableName of the recordScalar method as follows:

```
char buff[50];
n=sprintf(buff,"ITMposX: %d %f",i,ITMposX[i]);
recordScalar(buff,ITMspeed[i]);
```

The result:

```
scalar "TFSim.host[266].appl"   "ITMposX: 4 5803.822242"    15.3723444867
```

Thanks to this little hack we can now readily find all entries in host 266's trafficmap. Using this method we can output related information on one line, which makes it easier to evaluate the information later on.

B.8 Summary

In this Chapter the implementation of a simulator based on OMNeT++ and MFw was described. Traffic can be enabled to move on a road according to the IDM with a configurable reduced speedzone. Every vehicle is instrumented with the TrafficFilter which means that every vehicle can participate in the dissemination of TrafficMap information or even contribute information. Floods carrying TrafficMap information can be generated by the most remote vehicle to model 10km of propagation. Several configuration aspects such as SNP and microSlots can be en- or disabled for separate analysis. The system is instrumented to output all relevant measurements to output files for offline analysis. This analysis is covered in more detail in Appendix D where the toolchain is described. The analysis derives the Performance Metrics which are covered in Chapter 6.

Appendix C

Derivations

C.1 Deriving the Transmission Range

The next Matlab function calculates the interference distance in the same way it is done in the OMNeT++ Mobility Framework as derived in C.2.

```
function range = calcInterfDist(power)
speedoflight=300000000.0;
wavelength=speedoflight/5.87e9;
sat=-120;
alpha=3.5;
minRxPow = 10.0^(sat/10.0);
range = ( (wavelength^2 * power) / ((4*pi)^2 * minRxPow) )^(1.0/alpha);
```

An interference range of 500m is found to result in a transmission power of 168.97mW, the assumption is that transmission range generally is half the interference range.

C.2 Mobility Framework Propagation Model

The standard propagation model used in the Mobility Framework is derived from the Free Space Propagation model, also known as the Friis Free-Space Formula [35]:

$$\frac{P_r}{P_t} = G_t G_r \left(\frac{\lambda}{4\pi R} \right)^2 \tag{C.1}$$

where P_t and P_r are the power transmitted en received respectively. This gives a unitless value which is often expressed in dB. G_t and G_r are the antenna gain at the sender and receiver, λ is the wavelength and R is the distance between the transmitter and the receiver.

In the ChannelControl module the maximum interference range is calculated in order to determine the subset of nodes for which signal-to-noise evaluation might result in interference or successful reception. The maximum R can be found when all other variables are known:

```
//the minimum carrier frequency for this cell
double carrierFrequency = par("carrierFrequency");
//maximum transmission power possible
double pMax             = par("pMax");
//minimum signal attenuation threshold
double sat              = par("sat");
//minimum path loss coefficient
double alpha            = par("alpha");

double waveLength     = (speedOfLight/carrierFrequency);
```

```
//minimum power level to be able to physically receive a signal
double minReceivePower = pow(10.0, sat/10.0);

interfDistance = pow(waveLength * waveLength * pMax / (16.0*M_PI*M_PI*minReceivePower), 1.0/alpha);
```

This does the following:

$$R = \left(\frac{\lambda^2 P_t}{(4\pi)^2 P_r} \right)^{\frac{1}{\alpha}} \tag{C.2}$$

Derivation:

$$\frac{P_r}{P_t} = G_t G_r \left(\frac{\lambda}{4\pi R} \right)^2 \text{(Eq. C.1)}$$

$$\Rightarrow \text{normalise antenna gain}(G_t = G_r = 1) \Rightarrow$$

$$\frac{P_r}{P_t} = \left(\frac{\lambda}{4\pi R} \right)^2 = \lambda^2 (\frac{1}{4\pi})^2 R^2 \Rightarrow \text{divide both sides by } \frac{P_r}{P_t} \Rightarrow$$

$$1 = \frac{P_r}{P_t} \lambda^2 (\frac{1}{4\pi})^2 R^2 \Rightarrow \text{ divide both sides by } R^2 \Rightarrow$$

$$\frac{1}{R^2} = \frac{\lambda^2 P_t}{(4\pi)^2 P_r} \Rightarrow \text{ take square root of both sides } \Rightarrow$$

$$R = \left(\frac{\lambda^2 P_t}{(4\pi)^2 P_r} \right)^{\frac{1}{2}} \text{(Eq. C.2)}$$

□

In urban areas and to compensate for multipath propagation and fading the exponent in (C.1) is often replaced with α such that $3 < \alpha < 5$. The default value for α used in the Mobility Framework is 3.5.

Equation C.2 is used to find the maximal interference distance. Whether the signal is received as data or as noise is determined by the Decider module based on Signal-to-Noise Ratio (SNR).

An estimate of the maximum transmission range is needed to determine the size and allocation of the slots in the slotted 1-Persistence Flooding scheme as described in Section 5.3.4 on page 79.

Appendix D

Toolchain and Workflow

The Linux platform (OpenSUSE), the vast amount of Open Source applications available and the open architecture of the OMNeT++ simulator provide a versatile platform to perform the simulation studies on.

The different generations of the TrafficFilter simulation are called TFn. TF3 provides a simple evaluation of the mobility implemented in IDMMobilityModel. It contains no communications and produces only a speed / position plot as presented in Section B.3.3. TF4 evaluates two flooding strategies: the slotted 1-persistent flooding and the microSlotted 1-persistent flooding. TF5 evaluates the effects of mobility on the microSlotted 1-persistent flooding scheme and TF6 evaluates the quality of the communicated information. TF5 and TF6 use the best flooding strategy found in TF4; TF5 and TF6 use the IDMMobilityModel as evaluated in TF3.

D.1 Setup

Components – A collection of shell scripts and commandline utilities used to perform preparations for the simulation and analysis of the results.

- ulimit (control resources available to this shell)
- cut (remove sections from each line of files)
- rm, cp, cat, tac, head, tail (basic file operations and output options)
- awk (pattern scanning and processing), tr (translate characters)
- sort (sort lines of text files), paste (merge lines of files)
- wc (print line, word or byte count of file), uniq (report or omit repeated lines)
- seedtool (part of OMNeT++ distribution, provides seeds for RNGs)
- bc (commandline calculator used to circumvent bash's integer math)
- gnuplot (plotting suite)
- epstopdf (convert EPS files to PDF)
- python (an interpreted, interactive, object-oriented programming language)
- bash (GNU Bourne-Again SHell)

Programming languages:

- C++, language in which the simulator was built in OMNeT++ and the Mobility Framework
- C, language in which part of the analysis of the results was performed

Auxiliary scripts – The following scripts are called from the toolchain script.

mrng.sh – used to generate predictable random number streams outside the simulator. Creates a table for every density. A table consists of numHost lines and numRuns columns. These are used to give every vehicle a predictable "random" intervehicle distance, exactly equal for the two compared alternatives within a scenario. The random variates in the table are drawn from a uniform distribution $\left[-\frac{average\_spacing}{2}, \frac{average\_spacing}{2}\right]$ obtained from the python random library which uses, just like OMNeT++, the Mersenne Twister as the RNG.

ci.sh – used to calculate the mean and confidence intervals.

countprops.sh – parses the floodtraces to produce an indicator for end-to-end propagation of floods:

1. For every density:
   (a) For every run:
      i. count the number of floods
      ii. For every flood:
         - find maximum propagation of a flood
      iii. average propstats for this run
   (b) find the maximum and minimum propagation
   (c) from all propagations, find which percentage made it end-to-end
2. output propagation plot

plotStatistics.sh – generates plots from the gathered and computed statistics:

- delay
- hops
- transmit/receive
- floods observed
- overhead (TX/RX)
- slot utilisation
- percentage of end-to-end propagation
- channel utilisation

processTrafficMaps.sh – extracts the TrafficMap information from the node closest to position 0 and plots against actual speed/position tuples of the system.

Simulator – the OMNeT++ simulation program compiled from OMNeT++, Mobility Framework and own source based on the MFw template. The contents of the simulator are discussed in-depth in the thesis.

D.2 Toolchain

The bash scripts testFlooding.sh (TF4), testMobCom.sh (TF5), testTrafficFilter.sh (TF6) have roughly this structure:

1.
    - set densities [10,15,20,30,40,50,60,80,100,125,(150,200)]
    - set alternatives [withMicroSlots,withoutMicroSlots] [withMob,withoutMob] [withSNP, withoutSNP]
    - set Transmission Power
    - set number of runs to execute per density
    - set output files: $SCALARFILE, $VECTORFILE and $SNAPSHOTFILE
2. For both alternatives:
    (a) For all densities
        i. Construct omnetpp.ini
            A. Set playgroundSize
            B. Set number of hosts
            C. Add TFSim.host[$currHost].mobility.x = HOST[$currHost]
            D. Add reducedSpeedZone info
            E. Add switches:
                - which alternative to use
                - which transmission power to use
                - which spacing to use
                - how long the simulation should run
        ii. For all runs:
            A. replace all HOST[<integer>] with position from the tables generated by mrng.sh
            B. seed the three RNGs
            C. perform the call to the TrafficFilter simulator: TF4, TF5 or TF6. This runs the actual simulation with the config file omnetpp.ini constructed above.
            D. obtain information on channel utilisation from $SCALARFILE
            E. obtain handles to and count the number of Floods in run from $SCALARFILE
            F. For every flood:
                - find the start and end time of this flood (plus duration)
                - find the start and end position of this flood
                - find number of hops required for this flood
        iii. compute flood statistics for this run:
            - delay
            - distance
            - number of hops
        iv. compute number of transmissions, receptions and floods observerd per node and plot this
        v. output a trace of the positions covered by every unique flood and plot this
        vi. obtain information on slot utilisation from $SCALARFILE
    (b) For all runs
        i. For all densities
            - create delay stats
            - create hop stats
            - create transmission stats
            - create reception stats
            - create observed flood stats
            - (if mobilityEnabled) output a speed / position plot
            - (if trafficFilterEnabled) call processTrafficMaps.sh
    (c) Compute multi-run stats, For all densities:
        - create delay stats (using ci.sh)

- create hop stats (using ci.sh)
- create transmission stats (using ci.sh)
- create reception stats (using ci.sh)
- create observed flood stats (using ci.sh)
- create slot utilisation stats
- create channel utilisation stats (using ci.sh)
- create normalised channel utilisation stats (using ci.sh)

(d) call countprops.sh & (run in parallel after first alternative but sequentially after second)

3. call plotStatistics.sh

Per simulation 2 alternatives are evaluated for 10 different densities. 50 runs are performed for every density and per run 100 floods are executed. In total 1000 simulator-runs are executed to obtain the details of 100000 floods.

D.3 Increasing Efficiency of the Analyser

The tests discussed in Section B.6 to derive a measure for the quality of the TrafficMap contents operate on output generated by the simulator which is part of one flood. Remember that for every density we execute 50 runs, and per run execute 100 floods. The initial implementation of the TrafficMap quality analysis was in AWK just like most of the analysis tool. AWK operates on a line-by-line basis, it traverses an input file linearly. This becomes very time-consuming if we want to crawl through the equivalent of 78TB of data[1], an initial testrun confirmed this. The projected duration was three to four weeks. Up to the evaluation of the effects of mobility (TF5) the simulation and analysis took between one and three days; with the addition of a fine-grained analysis working on the flood-level drastic measures were needed to perform the operations within reasonable time.

AWK is a good and flexible tool for pattern recognition, but these tests are also computationally intensive and operate on a lot of data. It was decided to write this part of the analysis tool in C. Without going into too much implementation details the following parts of the analysis that were most time-consuming were written in C:

- TrafficMap quality checker: checkTM
- End-to-end propagation counter: countprops
- Floodfinder: findfloods

checkTM CheckTM performs Test 1 and Test 2 on every flood found in the outputfile of the current simulation run. It reads from the simulator's scalar output file. For every (fully propagated) flood it extracts the TM and ACT and stores these in memory. Test 1 and 2 are then performed on the information in memory. The speed-up with respect to the AWK implementation was tremendous, instead of 45 minutes the analysis of one run takes 2.8 seconds primarily because the information is read only once and operations are performed on (much faster) memory.

countprops Countprops reads flood traces from the simulator's scalar outputfile and stores these in memory. Every flood can be identified by a FloodID (see Section 5.2). For every unique FloodID the maximum propagation is found; if the flood reaches the area within transmission range of position 0 (i.e. position < 250 meters) we conclude the flood traversed the entire road. From this information we ultimately derive a percentage of floods that fully propagated.

[1] $2 \times 8 \times 50 \times 100 \times \sim 1$ Gigabyte ≈ 8TB

findfloods Findfloods reads the scalar file and for every unique and fully propagated flood it derives the number of hops.

These three programs still perform a very expensive operation (pattern matching on the scalar file) but perform this operation only once to store the information in memory. They show show a significant increase in performance with respect to the AWK implementation. The entire simulation and analysis for the analysis of the TrafficMap quality takes only 25 hours in stead of the projected three to four weeks.

Appendix E

V2VCOM 2008 submission

Providing Over-the-horizon Awareness to Driver Support Systems

Martijn van Eenennaam and Geert Heijenk
e.m.vaneenennaam@student.utwente.nl, geert.heijenk@utwente.nl
Department of Computer Science,
University of Twente, The Netherlands

***Abstract*— Vehicle-to-vehicle communications is a promising technique for driver support systems to increase traffic safety and efficiency. A proposed system is the Congestion Assistant [1], which aims at supporting drivers when approaching and driving in a traffic jam. Studies have shown great potential for the Congestion Assistant to reduce the impact of congestion, even at low penetration. However, these studies assumed complete and instantaneous availability of information regarding position and velocity of vehicles ahead. In this paper, we introduce a system where vehicles collaboratively build a so-called TrafficMap, providing over-the-horizon awareness. The idea is that this TrafficMap provides highly compressed information that is both essential and sufficient for the Congestion Assistant to operate. Moreover, this TrafficMap can be built in a distributed way, where only a limited subset of the vehicles have to alter it and/or forward it in the upstream direction. Initial simulation experiments show that our proposed system provides vehicles with a highly compressed view of the traffic ahead with only limited communication.**

I. Introduction

VEHICLE-TO-VEHICLE (V2V) communication has great potential to make vehicular mobility more efficient and safer. Cooperating vehicles can share information on upcoming traffic conditions or planned trajectories, resulting in the orchestration of more efficient traffic flows. Various vehicle safety applications such as collision avoidance and cooperative driving are expected to result in a drop in collisions. This, in turn, will also benefit the efficiency.

A system proposed by Van Driel [1] looks promising in the effort to reduce traffic congestion on highways. It has been proven that—with a high enough degree of market penetration—a reduction of the number and effects of traffic jams is possible. This system is called the *Congestion Assistant*, a system onboard a vehicle which is based on knowledge of the situation on the road ahead. It is assumed in [1] that this knowledge is available and dependable, but no system for distributing this knowledge is proposed. We set out to devise such a system and explore possibilities.

The Congestion Assistant as described in [1] works by improving a driver's efficiency and performance in traversing a traffic jam, as illustrated in Fig. 1. The *Warning & Information* function informs the driver of upcoming traffic situations. An *Active Pedal* function ensures a reduced inflow of vehicles at the tail of the congestion by gradually reducing the vehicle's speed. Because of the lower speed vehicles can maintain a closer following distance; as a result the stretch of road is used to effectively buffer the inflow of new vehicles into the congestion. The Active Pedal counters the unfavourable human behaviour of maintaining speed untill stopped traffic is observed in front. This behaviour is found to result in a high inflow and high risk of accidents. The gradual reduction of speed implies less hard braking will occur, which benefits safety. Once a vehicle enters a congestion the *Stop & Go* function takes over longitudinal control of the vehicle, functioning as a type of Adaptive Cruise Control. The system maintains a close headway to the predecessor and—because there is no human in the loop—can react swiftly to sudden changes. Driver simulations carried out at TNO and microscopic traffic simulations using the ITS modeller [2] show great opportunities for reducing the impacts of congestion, even at low penetration (e.g. 10%) [1]. The Congestion Assistant acts upon knowledge of the position of the head and tail of a congestion.

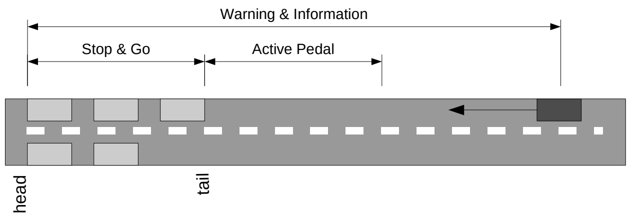


Fig. 1. A driver approaches a traffic jam

Most V2V systems proposed in literature operate on close range, e.g. with only one or a few hops in mind. Examples are a system for cooperative driving at blind intersections [3], cooperative collision warning and avoidance systems [4], [5]. These systems function by creating awareness of each other by means of beacon messages, which contain such information as speed, acceleration, etc., and are sent at regular intervals. An abiding geocast approach is proposed by Yu and Heijenk in [6] to actively notify approaching vehicles of upcoming safety events. Such approaches are either not able to carry the required information far enough upstream or are not detailed enough or impractical because the amount of information aggregated over several kilometers of highway grows rapidly. For instance, signalling a traffic congestion by means of abiding geocast messages could cause many such messages to coexist. If beacon messages were to include information of surrounding vehicles the communication system would easily clog up because of the vast amounts of data due to the number of vehicles present on several kilometers of highway.

It becomes clear we are facing two problems: a) how do we determine there is a traffic congestion and b) how do we distribute this information across several kilometers of highway, possibly several hundreds of vehicles? An approach called CarTel as described by Hull *et al.* in [7] highlights the use of vehicles as a mobile sensor network. Data is captured by many vehicles and collected in a centralised location. This information can then be retrieved from the CarTel portal.

Empirical studies concerning spatiotemporal traffic patterns such as presented by Kerner in [8] are carried out by means of several detector loops embedded in roads. Information can then be represented as a speed profile on a road at a certain time. Such information is primarily captured to study traffic behaviour. Traffic models such as the Intelligent Driver Model by Treiber *et al.* [9] can then be tuned to mimic this behaviour for simulation of the effects of, for instance, a planned additional lane.

We reason that a vehicle equipped with the Congestion Assistant could also benefit from such a speed profile of the road ahead, because all required information can be extracted from such a profile. The difference with existing work is that this information is both captured, processed and used by the vehicles without the need for any centralised authority. To facilitate this we propose a novel approach at building an over-the-horizon view using vehicle-to-vehicle communication.

The main contribution of this paper is the introduction of a system, called the TrafficFilter, in which vehicles collaboratively build a speed profile of the road using V2V communications. The profile, called TrafficMap, represents information needed by the Congestion Assistant efficiently, and can be built with minimal upstream communication.

The remainder of this article is structured as follows. In Section II we introduce the concept of the TrafficMap, the construct that contains the over-the-horizon view. We introduce some terminology and why vehicle-to-vehicle communication is the technology of choice for this application. Next, Section III introduces the process of constructing a TrafficMap by means of a threshold-based filtering technique called the TrafficFilter. In Section IV we discuss ongoing and future research, we conclude with Section V.

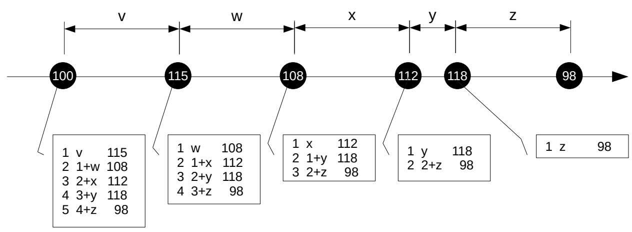


Fig. 2. Nodes on a line with a value denoting speed in km/h

II. Over-the-horizon View by Means of V2V Communications

A vehicle equipped with the Congestion Assistant needs to know the traffic situation several kilometers down the road. This distance is defined in [1] as five kilometers in advance of a traffic congestion, but preferably also includes the head of the jam possibly many kilometers further. Because of these great distances, the limited range of communication equipment, and the increase of interference when communicating over great distances, a V2V multi-hop communication approach is required.

It is reasoned by Jiang *et al.* in [10] that the messages produced by vehicles should be indicative; they cannot dictate how another vehicle must process and interpret a message. In line with this reasoning a vehicle publishes ‘information’, and not so much directives or commands for other vehicles.

To get a notion of the approach, vehicles are represented as tuples, composed of a means to denote a location plus the velocity of the vehicle at that location and the heading of the vehicle. A vehicle can form a representation of the road ahead, as sketched in Fig. 2. In this figure every vehicle moves on a straight line in the same direction so we will abstract from heading information. Every vehicle has knowledge of its predecessors, represented as a set of vehicle locations and velocities. We will refer to this collection of tuples as the TrafficMap. This TrafficMap gives a view on traffic flow speeds at certain locations down the road. The last entry of this list (5 in Fig. 2) defines the *virtual horizon*, the maximum defined distance captured in the TrafficMap. It is important this virtual horizon is close to or beyond the target virtual horizon as defined by the application.

If every vehicle were to add its information to the TrafficMap and (re)broadcast it, several problems would arise. Firstly, the potentially large number of vehicles on the road causes the list to grow rapidly, exceeding the maximum packet size. It is important the TrafficMap fits in one packet because no state will have to be maintained between consecutive packets and loss of one packet will not harm the information transferred in other packets. Secondly, the aggregate amount of data transferred would require a large amount of bandwidth, which needs to be shared with other applications or simply is not available, and thirdly, all these broadcasts would result

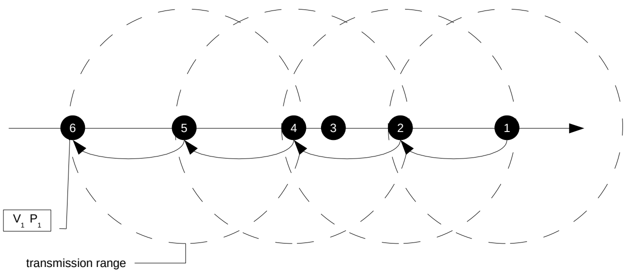


Fig. 3. From the perspective of node 6 (an observer) node 1 is a source (assuming 1 added an entry in the TrafficMap) and nodes 2,4 and 5 are relay nodes. Node 3 is a latent node.

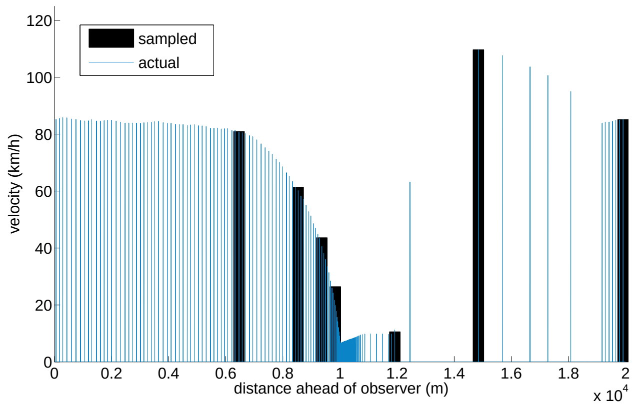


Fig. 4. TrafficMap created with Threshold Sampling approach

in heavy contention and many transmission collisions.

Several smart broadcasting techniques exist to combat the phenomenon known as *Broadcast Storm* [11] in MANETS or VANETS [12]. Such approaches rely on a means to suppress rebroadcasts by means of border awareness [13], location awareness or probability of rebroadcast [12], [11]. We reason that, besides using an efficient broadcasting technique that mitigates Broadcast Storms we should also limit the amount of data to only relevant data. This is what the TrafficFilter introduced in Section III does.

In the remainder of this article we will discuss *source*, *observer*, *relay* and *latent* nodes. The type of a node is based on the relevance of the information a node has and its role in the dissemination process. We define a source node (see Fig. 3) as a mobile node that broadcasts its own information (e.g. add an entry to the TrafficMap). An observer node is a (potentially) mobile node that receives this information. Relay nodes do not add information but merely pass it on. A latent node does not publish or relay information, but will receive it. The information functions as a means to observe traffic, hence any receiver is an observer. The moment an observer passes the information on it becomes a relay node itself if no new information is added, or a source node when it also injects its own information. The TrafficMap is disseminated by means of a geocast-like broadcast scheme that propagates against the flow of traffic, thus carrying information upstream.

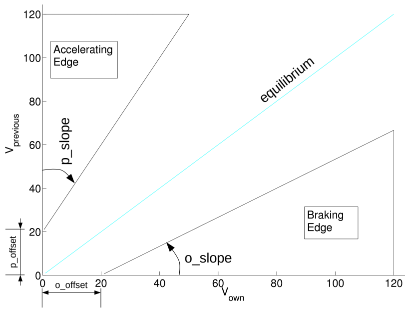
In the next Section we will propose a means to collaboratively build a TrafficMap by all equipped vehicles on the road in an efficient way.

III. The TrafficFilter

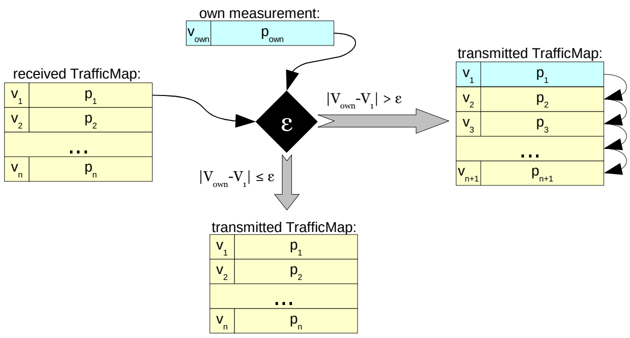
The TrafficFilter ensures only relevant information is added to the TrafficMap. It is a system similar to Run-Lenght Encoding or Pulse Code Modulation with a variable hold time. The aim is to make a set of samples that best represents the actual speed-position relations on a road. Because a vehicle is influenced for a great deal by factors such as speed limit and other traffic many vehicles will show roughly the same speed, a relation often used in both macroscopic [14], [15] and microscopic [9] traffic simulations. This means there will be a lot of similar speeds on our road, mostly clustered. We reduce this redundancy by only adding a sample to the TrafficMap when a certain deviation from the previous sample occurs. Fig. 4 shows a one-dimensional road, 20 kilometers in length, generated with the Intelligent Driver Model proposed by Treiber *et al.* in [9]. The IDM is used to illustrate the working of the TrafficFilter because it provides a quick way to obtain a detailed speed profile. Note that our implementation of the IDM is not calibrated to loopdetector data and is merely used for proof of concept. A vehicle's speed is mapped against its position on the road at a certain point in time, providing a snapshot of a stretch of road. The black bars are the samples in the TrafficMap from a vehicle approaching a traffic congestion, located at position 0.

When interpreting the TrafficMap, a sample remains valid until—in the direction *against* the flow of traffic—a new sample is added. Correlating vehicle position, speed and heading information with road map information allows a vehicle to deduct whether to add a sample and aids in interpreting the samples in the TrafficMap. In our description we will consider a TrafficMap for a single stretch of road, and hence focus on the position and speed information. The TrafficMap contains all the information the Congestion Assistant needs to know about upcoming traffic conditions: in the case of Fig. 4 the tail of the jam is located 10km away and the curve of braking vehicles can be interpolated. The head of the congestion is at 12km. The Warning & Information function can derive information such as the time to the jam, expected incurred delay and—when the vehicle is in the congestion—the progress within the congestion.

The TrafficFilter consists of three functions which operate on the TrafficMap: a capturing function ensures only relevant samples are added. An averaging function ensures a sample represents a small area on the road rather than a single vehicle. A reduction function removes redundancy in remote samples and removes those samples beyond the virtual horizon. These functions



(a) The threshold function ε



(b) Adding a sample to the TrafficMap

Fig. 5. Capturing a sample

work on the TrafficMap which we will now formally define.

Let TM be a set of tuples (v_i, p_i) where p is the position and v the velocity of vehicle i:
$TM = [(v_1, p_1), (v_2, p_2), \ldots, (v_n, p_n)]$ where $p_i \neq p_j$ and $1 \leq i \leq n$ and n denotes the number of tuples currently in the TrafficMap.

A. Sample Capturing

When a vehicle receives a TrafficMap it evaluates whether its speed (v_{own}) deviates *enough* from that of the last source node's speed $(v_{previous} = v_1)$ and if so, add a new sample, i.e.,

$$TM_{new} = \begin{cases} [(v_{own}, p_{own}), TM_{old}] & \text{if } |v_{own} - v_1| > \varepsilon; \\ TM_{old} & \text{otherwise.} \end{cases} \tag{1}$$

Here ε is a function of the two speeds, which decides if the deviation between the two is large enough. It justifies adding a sample to the TrafficMap when the $(v_{own}, v_{previous})$ point falls in either of the two areas (accelerating edge, braking edge) shown in Fig. 5(a). This simple ε-function consists of two straight lines, the offset and slope of which can be configured. The equilibrium-line represents free-flowing traffic, most of the speed differences will be close to this line. It is deviation from this line which is of interest because the flow speed of traffic changes. The exact definition of the ε-function needs to be determined using simulation or field experiments.

Fig. 5(b) illustrates how a sample is added to the TrafficMap. When the own measurements—derived from onboard sensors or GPS—differ significantly from v_1 (as determined by the ε-function and (1)) the own measurement is *pushed* onto the stack and rebroadcast, i.e. the vehicle is a source node. If it does not differ the TrafficMap is either rebroadcast unaltered (relay node) or not broadcast at all (latent node), if upstream vehicles are known to rebroadcast the TrafficMap.

B. Sample Averaging

A single vehicle is responsible for adding an entry to the TrafficMap. Although vehicles are influenced by factors such as speed limits and other traffic it is clear that there can be deviations, even in free-flowing traffic. An example is the difference between a car and a heavy truck it is overtaking. We would like a sample not to be a potentially locally deviating value, but a good representation of the velocity in the immediate vicinity of the node that adds it. In order to make a sample representative for the general area around the vehicle we could introduce elaborate majority-voting schemes, but a simple averaging probably also suffices. The idea is as follows:

1) A node decides to add its measurement to the TrafficMap because it is allowed to do so by the ε-function.
2) The TrafficMap is rebroadcast.
3) A vehicle a short distance upstream receives it. Its ε-function does not allow it to add a new sample. It might, however, slightly alter the last entry (v_1, p_1) if it is within the averaging distance Δ.
4) The TrafficMap is rebroadcast.

The result is that a sample is like a drop of paint, it gradually hardens and does not accept adjustment after a certain amount of time, or distance in this case, expressed as the averaging distance Δ. The averaging is expressed in the following equation:

$$TM_{new} = \begin{cases} [(v_1^\star, p_1), TM_{old}(2 \cdots n)] & \text{if } \delta < \Delta; \\ TM_{old} & \text{otherwise,} \end{cases} \tag{2}$$

where

$$v_1^\star = \frac{v_1 + (v_{own} \times \theta(\delta))}{1 + \theta(\delta)} \tag{3}$$

and

$$\delta = |(p_{own} - p_1)|. \tag{4}$$

In words, the resulting value $v_1^\star$ is composed of the previous value of v_1 (the velocity-component of the entry at index 1 in the TrafficMap) plus a weighted amount of v_{own} at distance δ from the location where v_1 was captured. The weighing is handled by $\theta(\delta)$ which is defined as follows:

$$\theta(\delta) = \left(\frac{\Delta - \delta}{\Delta} \right). \tag{5}$$

Eq. (5) gives a value between 0 and 1 for any δ between 0 and Δ, the averaging interval. Depending on Δ and the vehicle density a sample v_1 is made by one or multiple vehicles. Presently we use a set value for Δ but it could be directly based (i.e. inversely proportional) on the density of traffic, the effects can be researched.

Eq. (3) ensures $\frac{1}{1+\theta}^{th}$ of the original sample $v1$ is carried on in $v_1^\star$. The result is an average calculated over an a priori unknown number of values. The exact definition of (3) and (5) is still subject of research.

Whether or not such averaging is of interest and what the effects are has to result from simulation or field studies. One can imagine that a car overtaking a truck in free-flowing highway traffic must not trigger addition of a sample to the TrafficMap as discussed in Section III-A. In this case the difference of $120 - 80 = 40$km/h should not be allowed to trigger adding a sample and the ε-function should be calibrated accordingly.

However, the averaging of several cars and one truck will result in a lower value due to the truck. Nevertheless it is argued that this gives a value well above speeds that might indicate traffic congestion.

C. Reducing Redundancy

When a sample is captured it represents the actual situation at the current location. As soon as the information travels upstream, its relation to the actual situation diminishes, e.g. the confidence intervals, both spatial and temporal, increase. The traffic situation close ahead needs to be represented in detail because the Congestion Assistant's *Active Pedal* function needs precise information on where the congestion begins. Further away, the traffic situation does not need to be represented in so much detail and a summarisation can be performed to reduce the number of samples. Every node that rebroadcasts (either a relay or a source node) can perform such reduction operations with a couple of assumptions:

- Two consecutive remote samples that are somewhat the same (as defined by a threshold ω) can be reduced to one, the most remote one. The idea is that the confidence intervals overlap and the sample represents an area.
- A distant set of samples indicating a drop in speed (tail of a jam) resembles a set of stairs. They can be reduced to the first and last sample of these stairs. An observer can then interpolate between the samples.

Redundant samples generated because of a generous ε-function can be removed or merged based on a complete overview of the redundant sample's up- and downtream conditions. Whether to keep, remove or merge a sample is also threshold-dependent. This threshold ω depends on traffic dynamics, just like the ε-function used as a capturing threshold, but also on the distance to the current node as confidence intervals increase with the distance.

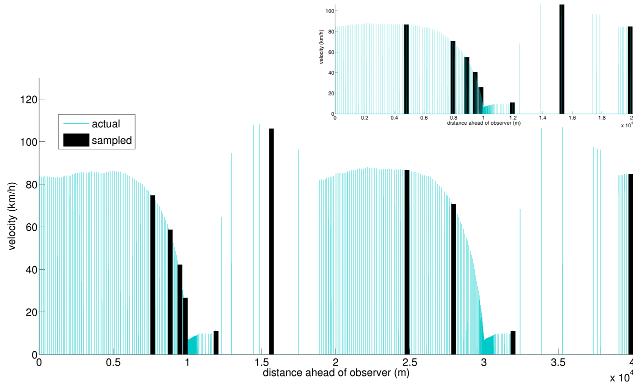


Fig. 6. A TrafficMap is passed upstream. Some redundancy is removed along the way

The goal is to remove only redundant samples and reduce the size of the TrafficMap. This will be beneficial when the aim is to reach a large virtual horizon. This is illustrated in Fig. 6. Here the original stretch of road is presented together with its representation twice as far away on the same road. The former is placed to align with the latter. As can be seen, the observer of the TrafficMap on the bottom sees two traffic jams up ahead. The observer of the top TrafficMap only sees one (and has probably just passed the other one). Note that the top TrafficMap's 8 samples have been reduced to 4 in the bottom TrafficMap, without too much loss of detail.

The reduction step will also remove samples in the following two cases:

- A sample is beyond the target virtual horizon. Samples beyond a certain distance are discarded to ensure information only flows as far as defined by the target virtual horizon. As the information travels it ages, loosing its relation to the actual present situation, rendering it obsolete.
- In order to meet demands of a maximum message size, remote samples might be discarded when there simply are too many samples in the TrafficMap. This could be the result of turbulent dynamics in traffic. An implication might be that the actual virtual horizon draws nearer.

The Congestion Assistant acts upon information from kilometers away. As such it is not a delay-critical application and should leave enough bandwidth for other, more delay-critical applications [16].

IV. Ongoing and Future Work

In Section III we have introduced a system in which vehicles collaboratively build a TrafficMap. This is an efficient representation of the road traffic on a certain stretch of road. The communication protocols for actually distributing a TrafficMap still need to be defined but a hint of the direction is provided here.

Initial simulation experiments, such as those used to generate Fig. 4 and Fig. 6, show that our proposed

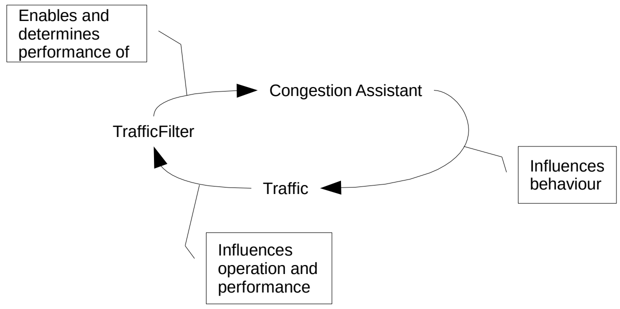


Fig. 7. Interplay between Traffic, TrafficFilter and Congestion Assistant

system provides vehicles with a highly compressed view of the traffic ahead with only limited communication. Future work will include a detailed simulation study to evaluate the performance of the system. We also plan to study the impact of communication system performance (packet loss, latency) and data reduction on the performance of the Congestion Assistant.

It is expected that, to deal with the great variation of dynamics in highway traffic, the thresholds and configuration variables used by the TrafficFilter should be dynamic as well. When the dynamics of traffic are such that a lot of samples are generated, thresholds might be adjusted. When vehicle density decreases, the averaging interval Δ might be increased. A target virtual horizon needs to be defined (as required by the application) and thresholds need to be set to achieve this.

The Congestion Assistant and the TrafficFilter function in a highly dynamic environment; that of highway traffic. The performance of the TrafficFilter depends on the presence of vehicles and hence on traffic, as illustrated in Fig. 7. This traffic, in turn, is influenced by the Congestion Assistant. The Congestion Assistant acts upon information provided by the TrafficFilter. These dependency issues are all for further study.

The TrafficFilter relies on the presence of instrumented vehicles on the road. Using a realisitic communication range of 200–300 meters [17], [6] it is quite possible there are gaps in the network. A means to overcome such a gap is by allowing vehicles on the opposite lane to carry this information.

So far only the spatial side of the TrafficFilter has been described: using a snapshot at a point in time the algorithms of the capture, average and reduction functions have been introduced. Here we will briefly introduce the temporal side of the TrafficFilter, which is still topic of research.

A vehicle enters a highway and it listens for ongoing TrafficMap communications. When TrafficMap communications are observed the vehicle participates, otherwise it initiates TrafficMap communications. There are three factors here: firstly, we need to ensure a certain number of messages per time unit to keep the system alive and allow newcomers to engage, secondly we must ensure a received TrafficMap is passed on swiftly and, lastly, observed deviations in speed must be rapidly communicated upstream.

A timer can be used to ensure a maximum time between TrafficMap communication is satisfied. If no TrafficMap has been received for a certain period the vehicle decides to broadcast. When a TrafficMap is received from downstream (thus ahead in traffic) the information is used by the capture, average and reduction functions and the TrafficMap is sent for transmission using a distance aware (slotted p-persistent) flooding scheme modified from that described by Wisitpongphan *et al.* in [12]. This scheme propagates TrafficMap disseminations upstream, favouring rebroadcast by nodes further away to minimise the number of hops over great distances and achieve low latency. The exact detail of the flooding scheme is subject to further research but the idea is that on a highway several TrafficMap floods are being passed upstream simultaneously (at geographic disjoint locations). The system ensures a minimum of such floods but also a maximum by means of collecting and summarising several TrafficMaps received in rapid succession.

V. Conclusions

The Congestion Assistant described by Van Driel [1] provides a means to alleviate the effects of traffic congestion. It is recognised that a system such as the Congestion Assistant will only work if it is backed by an adequately provisioned communication service provider. We believe that a distributed ad hoc solution using multi-hop V2V communications is the best way to see to the communication needs of the Congestion Assistant system. A possible solution has been presented in this article. It consists of a TrafficMap that only contains essential velocity information, and that is built in a distributed and collaborative way using a so-called TrafficFilter. The follow-up research will evaluate if this approach is a viable one.

References

[1] C. van Driel, “Driver support in congestion - an assessment of user needs and impacts on driver and traffic flow,” Ph.D. dissertation, University of Twente, Nov 2007.

[2] E. Versteegt, G. Klunder, and B. Van Arem, “Its modeller - irvin wp1.1,” TNO Mobility and Logistics, Delft, The Netherlands, Tech. Rep. Report 2005-16, 2005.

[3] L. Li and F.-Y. Wang, “Cooperative driving at adjacent blind intersections,” *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, vol. 1, pp. 847–852 Vol. 1, 10-12 Oct. 2005.

[4] T. ElBatt, S. K. Goel, G. Holland, H. Krishnan, and J. Parikh, “Cooperative collision warning using dedicated short range wireless communications,” in *VANET ’06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks.* New York, NY, USA: ACM, 2006, pp. 1–9.

[5] J. Yin, T. ElBatt, G. Yeung, B. Ryu, S. Habermas, H. Krishnan, and T. Talty, “Performance evaluation of safety applications over DSRC vehicular ad hoc networks,” in *VANET ’04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks.* New York, NY, USA: ACM, 2004, pp. 1–9.

[6] Q. Yu and G. Heijenk, "Abiding geocast for warning message dissemination in vehicular ad hoc networks," *to be published in Proceedings of the IEEE Vehicular Networks and Applications Workshop 2008 (VehiMobi 2008)*, 2008.

[7] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. K. Miu, E. Shih, H. Balakrishnan, and S. Madden, "Cartel: A distributed mobile sensor computing system," in *4th ACM SenSys*, Boulder, CO, November 2006.

[8] B. S. Kerner, "Empirical macroscopic features of spatial-temporal traffic patterns at highway bottlenecks," *Phys. Rev. E*, vol. 65, no. 4, p. 046138, Apr 2002.

[9] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E*, vol. 62, no. 2, pp. 1805–1824, Aug 2000.

[10] D. Jiang, V. Taliwal, A. Meier, W. Holfelder, and R. Herrtwich, "Design of 5.9 GHz DSRC-based vehicular safety communication," *Wireless Communications, IEEE [see also IEEE Personal Communications]*, vol. 13, no. 5, pp. 36–43, October 2006.

[11] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wirel. Netw.*, vol. 8, no. 2/3, pp. 153–167, 2002.

[12] N. Wisitpongphan, O. Tonguz, J. Parikh, P. Mudalige, F. Bai, and V. Sadekar, "Broadcast storm mitigation techniques in vehicular ad hoc networks," *Wireless Communications, IEEE [see also IEEE Personal Communications]*, vol. 14, no. 6, pp. 84–94, December 2007.

[13] C. Zhu, M. Lee, and T. Saadawi, "A border-aware broadcast scheme for wireless ad hoc network," *Consumer Communications and Networking Conference, 2004. CCNC 2004. First IEEE*, pp. 134–139, 5-8 Jan. 2004.

[14] D. Helbing and M. Treiber, "Gas-kinetic-based traffic model explaining observed hysteretic phase transition," *Phys. Rev. Lett.*, vol. 81, no. 14, pp. 3042–3045, Oct 1998.

[15] D. Helbing, "Improved fluid-dynamic model for vehicular traffic," *Phys. Rev. E*, vol. 51, no. 4, pp. 3164–3169, Apr 1995.

[16] S. Rezaei, R. Sengupta, and H. Krishnan, "Reducing the communication required by DSRC-based vehicle safety systems," *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pp. 361–366, Sept. 30 2007-Oct. 3 2007.

[17] F. Bai and H. Krishnan, "Reliability analysis of DSRC wireless communication for vehicle safety applications," *Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE*, pp. 355–362, 2006.

Bibliography

[1] *Probabilistic broadcast for flooding in wireless mobile ad hoc networks*, 2003.

[2] Christophe Adrados, Irène Girard, Jean-Paul Gendner, and Georges Janeau. Global positioning system (GPS) location accuracy improvement due to selective availability removal. *C.R. Biologies 325*, pages 165–170, 2002.

[3] Phil Agre. Privacy in intelligent transportation systems. In *Fifth Conference on Computers, Freedom, and Privacy*, volume 2-4. The Network Observer, March 1995.

[4] Fan Bai and H. Krishnan. Reliability analysis of DSRC wireless communication for vehicle safety applications. *Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE*, pages 355–362, 2006.

[5] Masako Bando, Katsuya Hasebe, Ken Nakanishi, and Akihiro Nakayama. Analysis of optimal velocity model with explicit delay. *Phys. Rev. E*, 58(5):5429–5435, Nov 1998.

[6] Andrea Barisani and Daniele Bianco. Unusual car navigation tricks: Injecting RDS-TMC traffic information signals. April 2007.

[7] California PATH UC Berkeley. Shift - the hybrid system simulation programming language. http://path.berkeley.edu/SHIFT/, 1998.

[8] BMW Active Cruise Control. http://www.worldcarfans.com/2030805.001, August 2003.

[9] W.P.B. Broeders. How to organize a national RDS/TMC service in the netherlands. *Vehicle Navigation and Information Systems Conference, 1996. VNIS*, 7:111–117, 14-18 Oct. 1996.

[10] P. Burton, N. Davies, S. Hoffman, A. Hobbs, D. Woolard, and A. Graham. RDS-TMC in the UK. *Road Transport Information and Control, 1998. 9th International Conference on (Conf. Publ. No. 454)*, pages 130–134, 21-23 April 1998.

[11] Iso 11898-1:2003: Road vehicles – controller area network (can) – part 1: Data link layer and physical signalling. http://www.iso.org, 2003.

[12] A. Carter. The status of vehicle-to-vehicle communication as a means of improving crash prevention performance. Technical report, National Highway Traffic Safety Administation, 2005.

[13] D. Jim Chadwick. Projected navigation system requirements for intelligent vehicle highway systems (IVHS). pages 485–490. Inst of Navigation, Alexandria, VA, United States, 1994.

[14] Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia. Freeway performance measurement system: Mining loop detector data. *Transportation Research Record, Volume 1748-1*, pages 96–102, Jan 2001.

[15] Wai Chen and Shengwei Cai. Ad hoc peer-to-peer network architecture for vehicle safety communications. *Communications Magazine, IEEE*, 43(4):100–107, April 2005.

[16] I. Chlamtac and S. Kutten. On broadcasting in radio networks–problem analysis and protocol design. *Communications, IEEE Transactions on [legacy, pre - 1988]*, 33(12):1240–1246, Dec 1985.

[17] William Jefferson "Bill" Clinton. Improving the civilian global positioning system (GPS). http://clinton4.nara.gov/WH/New/html/20000501_2.html, May 2000.

[18] CAR 2 CAR Communication Consortium. C2c-cc manifesto: Overview of the c2c-cc system version 1.1. Technical report, Car 2 Car Communication Consortium, August 28 2007.

[19] C. Cseh. Architecture of the dedicated short-range communications (DSRC) protocol. *Vehicular Technology Conference, 1998. VTC 98. 48th IEEE*, 3:2095–2099 vol.3, 18-21 May 1998.

[20] Wang Dahui, Wei Ziqiang, and Fan Ying. Hysteresis phenomena of the intelligent driver model for traffic flow. *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, 76(1):016105, 2007.

[21] William P. D'Amico and Mark H. Lauss. Wireless local area network flight demonstration for high doppler conditions. *Johns Hopkins APL digest, volume 24, number 4*, 2004.

[22] C Davis, P. Vixie, T Goodwin, and I Dickinson. RFC 1876 - a means for expressing location information in the domain name system. RFC 1876 (Experimental), Jan 1996.

[23] Glen M. D'Este, Rocco Zito, and Michael A. P. Taylor. Using GPS to measure traffic system performance. *Computer-Aided Civil and Infrastructure Engineering*, 14:255–265(11), Jul 1999.

[24] Wolfgang Detlefsen and Wilhelm Grabow. Interoperable 5.8 GHz DSRC systems as basis for europeanwide ETC implementation. *European Microwave Conference, 1997. 27th*, 1:139–145, Oct. 1997.

[25] John Donne. *Meditation XVII from Devotions upon Emergent Occasions*. 1623.

[26] P.K. Dutta and R. Sundaram. The tragedy of the commons? a characterization of stationary perfect equilibria in dynamic games. Discussion Papers 1988-06, Columbia University, Department of Economics, 1988.

[27] European differential beacon transmitters. http://www.effective-solutions.co.uk/beacons.html.

[28] Stephan Eichler. Performance evaluation of the IEEE 802.11p WAVE communication standard. *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th*, pages 2199–2203, 30 2007-Oct. 3 2007.

[29] Tamer ElBatt, Siddhartha K. Goel, Gavin Holland, Hariharan Krishnan, and Jayendra Parikh. Cooperative collision warning using dedicated short range wireless communications. In *VANET '06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, pages 1–9, New York, NY, USA, 2006. ACM.

[30] Transport en Logistiek Nederland. Transport in cijfers - editie 2007. Technical report, TLN, 2007.

[31] Eurotraffic website. http://www.eurotraffic.org/english/alertc.html.

[32] The free online dictionary, thesaurus and encyclopedia. http://www.thefreedictionary.com/traffic+jam.

[33] Michael Friedewald, Elena Vildjiounaite, Yves Punie, and David Wright. Privacy, identity and security in ambient intelligence: A scenario analysis. *Telematics and Informatics*, 24, Issue 1:15–29, February 2007.

[34] Florian Friedrici and Matthias Gerlach. Towards rssi=based position plausibility checks for vehicular communication. Jun 2008.

[35] Harald T. Friis. A note on a simple transmission formula. *Proceedings of IRE*, 34:254–256, May 1946.

[36] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns - Elements of Reusable Object-Oriented Software*. Addison-Wesley, Dec 2004.

[37] Google maps. http://maps.google.nl, Mar 2008.

[38] Google maps API. http://www.google.com/apis/maps/, Mar 2008.

[39] James Gyarmathy. Vehicle infrastructure integration (VII) enhancing safety, enabling mobility. Nov 2007.

[40] Millions spend whole day going to work and then returning. *The Hartford Courant*, May 1926.

[41] M. Hatakka, E. Keskinen, N. P. Gregersen, A. Glad, and K. Hernetkoski. From control of the vehicle to personal self-control; broadening the perspectives to driver education. *Transportation Research Part F: Traffic Psychology and Behaviour*, 5, Issue 3:201–215, September 2002.

[42] Dirk Helbing. Improved fluid-dynamic model for vehicular traffic. *Phys. Rev. E*, 51(4):3164–3169, Apr 1995.

[43] Dirk Helbing. From microscopic to macroscopic traffic models. Jun 1998.

[44] Dirk Helbing and Péter Molnár. Social force model for pedestrian dynamics. *Phys. Rev. E*, 51(5):4282–4286, May 1995.

[45] Dirk Helbing and Martin Treiber. Gas-kinetic-based traffic model explaining observed hysteretic phase transition. *Phys. Rev. Lett.*, 81(14):3042–3045, Oct 1998.

[46] Bret Hull, Vladimir Bychkovsky, Yang Zhang, Kevin Chen, Michel Goraczko, Allen K. Miu, Eugene Shih, Hari Balakrishnan, and Samuel Madden. Cartel: A distributed mobile sensor computing system. In *4th ACM SenSys*, Boulder, CO, November 2006.

[47] IEEE 1609 - family of standards for wireless access in vehicular environments (WAVE). http://www.standards.its.dot.gov/fact_sheet.asp?f=80, Jan 2006.

[48] IEEE standard for information technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements - part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pages C1–1184, June 12 2007.

[49] IEEE draft amendment to standard for information technology - telecommunications and information exchange between systems - local and metropolitan networks - specific requirements - part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: Amendment: Wireless access in vehicular environments. *IEEE 802.11p Draft 2.0*, November 2006.

[50] National Imagery and Mapping Agency. World geodetic system 1984 - its definition and relationships with local geodetic systems. Technical report, National Imagery and Mapping Agency, 2004.

[51] P.A. Ioannou and C.C. Chien. Autonomous intelligent cruise control. *Vehicular Technology, IEEE Transactions on*, 42(4):657–672, Nov 1993.

[52] Requirement 1.6 - traffic control. *National ITS Architecture version 6.0 - User Service Requirements*, April 2007.

[53] Daniel Jiang, Vikas Taliwal, Andreas Meier, Wieland Holfelder, and Ralf Herrtwich. Design of 5.9 GHz DSRC-based vehicular safety communication. *Wireless Communications, IEEE [see also IEEE Personal Communications]*, 13(5):36–43, October 2006.

[54] Rainer Kallenbach. Die zukunft des autofahrens: Sicherheit - information - assistenz. *58-er Internationalen Motorpressekolloquium*, Jun 2007.

[55] Holger Karl. Leistungsbewertung & simulation chapter 5: Omnet++ - a tool for simulation programming, 2004.

[56] S. Kato and S. Tsugawa. Evaluation of information transmission over inter-vehicle communications with simulation studies. *Intelligent Transportation Systems, 2002. Proceedings. The IEEE 5th International Conference on*, pages 324–329, 2002.

[57] B. S. Kerner and H. Rehborn. Experimental features and characteristics of traffic jams. *Phys. Rev. E*, 53(2):R1297–R1300, Feb 1996.

[58] B. S. Kerner and H. Rehborn. Experimental properties of phase transitions in traffic flow. *Phys. Rev. Lett.*, 79(20):4030–4033, Nov 1997.

[59] Boris S. Kerner. Empirical macroscopic features of spatial-temporal traffic patterns at highway bottlenecks. *Phys. Rev. E*, 65(4):046138, Apr 2002.

[60] Boris S. Kerner and Sergey L. Klenov. Microscopic theory of spatial-temporal congested traffic patterns at highway bottlenecks. *Phys. Rev. E*, 68(3):036130, Sep 2003.

[61] Kyoto Protocol to the United Nations Framework Convention on Climate Change. http://unfccc.int/resource/docs/convkp/kpeng.html, Dec 1997.

[62] Timo Lajunen, Dianne Parker, and Heikki Summala. Does traffic congestion increase driver aggression? *Transportation Research Part F: Traffic Psychology and Behaviour*, 2, Issue 4:225–236, December 1999.

[63] H. K. Lee, H.-W. Lee, and D. Kim. Macroscopic traffic models from microscopic car-following models. *Phys. Rev. E*, 64(5):056126, Oct 2001.

[64] Li Li and Fei-Yue Wang. Cooperative driving at adjacent blind intersections. *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, 1:847–852 Vol. 1, 10-12 Oct. 2005.

[65] Nicholas Joseph Linesch and Michael Perez. A nonlinear traffic model - dynamics on a one dimensional lane. Jun 2007.

[66] Marc Löbbers and Daniel Willkomm. A mobility framework for OMNeT++ user manual, Jan 2007.

[67] Wei Lou and Jie Wu. On reducing broadcast redundancy in ad hoc wireless networks. *Mobile Computing, IEEE Transactions on*, 1(2):111–122, Apr-Jun 2002.

[68] The merriam-webster online dictionary: Congestion. http://www.merriam-webster.com/dictionary/congestion.

[69] The merriam-webster online dictionary: Jam. http://www.merriam-webster.com/dictionary/jam.

[70] Adolf D. May. *Traffic Flow Fundamentals*. Prentice-Hall, 1992.

[71] M.B. May. Measuring velocity using GPS. *GPS World, Vol. 3 No. 8 pp. 58-65*, September 1992.

[72] Vision Range Estimation. http://www.mobileye.com/default.asp?PageID=328, 2008.

[73] Alexandros Moukas, Konstantinos Chandrinos, and Pattie Maes. Trafficopter: A distributed collection system for traffic information. In *CIA '98: Proceedings of the Second International Workshop on Cooperative Information Agents II, Learning, Mobility and Electronic Commerce for Information Discovery on the Internet*, pages 33–43, London, UK, 1998. Springer-Verlag.

[74] Ramjattan A. N. and Cross P. A. A Kalman filter model for an integrated land vehicle navigation system. *Journal of Navigation*, 48(2):293–302, 1995.

[75] Kai Nagel and Michael Schreckenberg. A cellular automaton model for freeway traffic. *J. Phys. I France*, 2:2221–2229, 1992.

[76] The network simulator: Ns-2. http://nsnam.isi.edu/nsnam/index.php/Main_Page.

[77] W.Y. Ochieng and K. Sauer. Urban road transport navigation: performance of the global positioning system after selective availability. *Transportation Research Part C 10*, pages 171–187, 2002.

[78] G.W.A. Offermans, A.W.S. Helwig, and D van Willigen. Eurofix: test results of a cost-effective DGNSS augmentation system. *Journal of the Institute of Navigation 50 (2)*, pages 209–223, 1997.

[79] T. Ohyama, S. Nakabayashi, Y. Shiraki, and K. Tokuda. A study of real-time and autonomous decentralized DSRC system for inter-vehicle communications. *Intelligent Transportation Systems, 2000. Proceedings*, pages 190–195, 2000.

[80] Omnet++ community website. http://www.omnetpp.org/.

[81] L.E. Owen, Yunlong Zhang, Lei Rao, and G. McHale. Traffic flow simulation using corsim. *Simulation Conference Proceedings, 2000. Winter*, 2:1143–1147 vol.2, 2000.

[82] Cheryl Pellerin. United states updates global positioning system technology. http://www.america.gov/st/washfile-english/2006/February/20060203125928lcnirellep0.5061609.html, Feb 2006.

[83] I. Prigogine and R. Herman. Kinetic theory of vehicular traffic. 1971.

[84] Stef Proost and Denise Van Regemorter. How to achieve the Kyoto target in Belgium - modelling methodology and some results. *CLIMNEG - CLIMBEL Network 1996-2001*, 2000.

[85] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying for flooding broadcast messages in mobile wireless networks. *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on*, pages 3866–3875, Jan. 2002.

[86] Qualnet network simulator. http://www.scalable-networks.com/.

[87] Shahram Rezaei, Raja Sengupta, and Hariharan Krishnan. Reducing the communication required by DSRC-based vehicle safety systems. *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pages 361–366, Sept. 30 2007-Oct. 3 2007.

[88] Glen M. D'Este Rocco Zito and Michael A. P. Taylor. Global positioning systems in the time domain: How useful a tool for intelligent vehicle-highway systems? *Transportation Research Part C: Emerging Technologies Volume 3, Issue 4, Pages 193-209*, Aug 1995.

[89] J.Z. Sasiadek and Q. Wang. Sensor fusion based on fuzzy Kalman filtering for autonomous robot vehicle. *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, 4:2970–2975 vol.4, 1999.

[90] Claude E. Shannon. Communication in the presence of noise. *Proc. Institute of Radio Engineers*, 37(1):10–21, Jan 1941.

[91] Salvador Sibecas, Celestino A. Corral, Shahriar Emami, and Glafkos Stratis. On the suitability of 802.11a/RA for high-mobility DSRC. *VTC*, 2002.

[92] Laurence E. Sigler. *Fibonacci's Liber Abaci (translation)*. Springer-Verlag, 2002.

[93] Sigurd. Different degree formats: resolutions and conversions. http://home.online.no/∼sigurdhu/Deg_formats.htm.

[94] Parmanand Singh. The so-called Fibonacci numbers in ancient and medieval India. *Historia Mathematica*, 12(3):229–273, 1985.

[95] N. J. A. Sloane. The AT&T on-line encyclopedia of integer sequences. http://www.research.att.com/~njas/sequences/Seis.html, 2007.

[96] David P. Stern. Distance to horizon. http://www-istp.gsfc.nasa.gov/stargaze/Shorizon.htm, Sept 2004.

[97] C. Suthaputchakun and A. Ganz. Priority based inter-vehicle communication in vehicular ad-hoc networks using IEEE 802.11e. *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*, pages 2595–2599, 22-25 April 2007.

[98] SWOV. Factsheet - volgtijd en verkeersveiligheid. Technical report, Stichting Wetenschappelijk Onderzoek Verkeersveiligheid, september 2007.

[99] The New York Times. Beijing has first workday under car restrictions. http://www.nytimes.com/aponline/world/AP-China-Traffic-Plan.html?partner=rssnyt&emc=rss, July 2008.

[100] Nikolaï Nikolaevich Vorobïev translated by Mircea Martin. *Fibonacci Numbers*. Birkhäuser, 2002.

[101] Martin Treiber. Dynamic traffic simulation. http://www.traffic-simulation.de, Jul 2005.

[102] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Phys. Rev. E*, 62(2):1805–1824, Aug 2000.

[103] Yu-Chee Tseng, Sze-Yao Ni, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. *Wirel. Netw.*, 8(2/3):153–167, 2002.

[104] Sadayuki Tsugawa and Shin Kato. Evaluation of incident information transmission on highways over inter-vehicle communications. *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, pages 12–16, 9-11 June 2003.

[105] The Norwegian research network Uninett. Status of the IEEE 802.11 standards. http://forskningsnett.uninett.no/wlan/ieee80211x.html, 2005.

[106] CEN. DSRC physical layer using infrared at 850 nm. *CEN TC 278 prENV278/9/*, 63, Dec 1995.

[107] National ITS architecture version 6.0. http://www.iteris.com/itsarch/index.htm, April 2007.

[108] B. van Arem. Cooperative vehicle-infrastructure systems: an intelligent way forward? Technical report, TNO, 2007.

[109] C.J.G. van Driel. *Driver Support in Congestion - An assessment of user needs and impacts on driver and traffic flow*. PhD thesis, University of Twente, Nov 2007.

[110] E. M. van Eenennaam. Bipobex: Giving context aware systems a view on the world - providing imagery data to context-aware networked services using BIP over Bluetooth-OBEX. Bachelor's thesis, University of Twente, Jan 2006.

[111] Martijn van Eenennaam and Geert Heijenk. Providing over-the-horizon awareness to driver support systems. *Proceedings of 4th IEEE Workshop on Vehicle to Vehicle Communications (V2VCOM)*, June 2008.

[112] L.F.W. van Hoesel and P.J.M. Havinga. A lightweight medium access protocol (lmac) for wireless sensor networks: Reducing preamble transmissions and transceiver state switches. May 2004.

[113] Andras Varga. Omnet++ - discrete event simulation system - version 3.2 - user manual, March 2005.

[114] De VerkeersInformatieDienst (VID). http://www.vid.nl/fileproblematiek.htm, Mar 2008.

[115] Tim Weil. Wireless access in vehicular environments (WAVE) - the emerging IEEE 1609 standard. Nov 2007.

[116] Geodetic system. http://en.wikipedia.org/wiki/Geodetic_system.

[117] Brad Williams and Tracy Camp. Comparison of broadcasting techniques for mobile ad hoc networks. In *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 194–205, New York, NY, USA, 2002. ACM.

[118] I.R. Wilmink, G.A. Klunder, and B. van Arem. Traffic flow effects of integrated full-range speed assistance (IRSA). *Intelligent Vehicles Symposium, 2007 IEEE*, pages 1204–1210, 13-15 June 2007.

[119] Eddie Wilson. Real-time traffic monitoring using mobile phone data. Jan 2005.

[120] N. Wisitpongphan and O.K. Tonguz. Scalable broadcast strategies for ad hoc routing protocols. *Wireless Pervasive Computing, 2006 1st International Symposium on*, pages 6 pp.–, Jan. 2006.

[121] N. Wisitpongphan, O.K. Tonguz, J.S. Parikh, P. Mudalige, F. Bai, and V. Sadekar. Broadcast storm mitigation techniques in vehicular ad hoc networks. *Wireless Communications, IEEE [see also IEEE Personal Communications]*, 14(6):84–94, December 2007.

[122] Life with big brother - feds plan to track every car - obscure agency working on technology to monitor all vehicles. http://www.worldnetdaily.com/news/article.asp?ARTICLE_ID=40795, Oct 2004.

[123] Sun Xi-Ming and Dong Yu-Jie. Analysis of stability for gas-kinetic non-local traffic model. *Chinese Physics Letters*, 23(9):2494–2497, 2006.

[124] Kaixin Xu, M. Gerla, and Sang Bae. How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks. *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*, 1:72–76 vol.1, 17-21 Nov. 2002.

[125] Qing Xu, Tony Mak, Jeff Ko, and Raja Sengupta. Vehicle-to-vehicle safety messaging in DSRC. In *VANET '04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 19–28, New York, NY, USA, 2004. ACM.

[126] S. Xu and T. Saadawi. Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks? *Communications Magazine, IEEE*, 39(6):130–137, Jun 2001.

[127] Jijun Yin, Tamer ElBatt, Gavin Yeung, Bo Ryu, Stephen Habermas, Hariharan Krishnan, and Timothy Talty. Performance evaluation of safety applications over DSRC vehicular ad hoc networks. In *VANET '04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 1–9, New York, NY, USA, 2004. ACM.

[128] Qiangyuan Yu and Geert Heijenk. Abiding geocast for warning message dissemination in vehicular ad hoc networks. *Communications Workshops, 2008. ICC Workshops '08. IEEE International Conference on*, pages 400–404, May 2008.

[129] Xi Zhang, Hang Su, and HSIAO-HWA Chen. Cluster-based multi-channel communications protocols in vehicle ad hoc networks. *Wireless Communications, IEEE [see also IEEE Personal Communications]*, 13(5):44–51, October 2006.

[130] Yuchen Zhang. Privacy-preserving secure vehicular communications. Nov 2007.

[131] Chunhui Zhu, M.J. Lee, and T. Saadawi. A border-aware broadcast scheme for wireless ad hoc network. *Consumer Communications and Networking Conference, 2004. CCNC 2004. First IEEE*, pages 134–139, 5-8 Jan. 2004.

[132] Jing Zhu and S. Roy. MAC for dedicated short range communications in intelligent transport system. *Communications Magazine, IEEE*, 41(12):60–67, Dec. 2003.

[133] Amy Zuckerman. Can europe and the U.S. live in ITS harmony? - intelligent transportation systems - technology information. http://findarticles.com/p/articles/mi_m0EKF/is_22_46/ai_62601661, May 29 2000.

Glossary

General terms

ACC	Adaptive Cruise Control
BS	Base Station
BSS	Basic Service Set
C2C-CC	Car 2 Car Communication Consortium
CACC	Cooperative Adaptive Cruise Control
CA	Congestion Assistant (ITS application)
CA	Collision Avoidance (IEEE 802.11)
CASS	Cooperative Active Safety System
CCA	Clear Channel Assessment
CRC	Cyclic Redundancy Check
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DCF	Distributed Coordination Function, usually achieved by CSMA
DGPS	Differential GPS - GPS augmented with fixed ground beacons
DIFS	DCF Interframe Space
DOT	(U.S.) Department of Transportation
DSRC	Dedicated Short Range Communication
DSSS	Direct Sequence Spread Spectrum
EIFS	Extended Interframe Space
ETC	Electronic Toll Collection
IEEE	Institute of Electrical and Electronics Engineers
FCC	Federal Communications Commission - U.S. body that governs radio frequency allocation
FCS	Frame Check Sequence (can be a CRC)
FHSS	Frequency Hopping Spread Spectrum
GPS	Global Positioning System
IBSS	Independent Basic Service Set
IRSA	Integrated full-Range Speed Assistant
ITS	Intelligent Transportation System
LAN	Local Area Network
MAC	Media Access Control layer (of OSI reference model and IEEE 802.11)
MAN	Metropolitan Area Network
MANET	Mobile Adhoc NETwork
MFw	Mobility Framework for the OMNeT++ Discrete Event Simulator
NAV	Network Allocation Vector (IEEE 802.11)
NAVSTAR	NAVigation Satellite Timing And Ranging global positioning system
OBE	On-Board Equipment, synonym for OBU
OBU	On-Board Unit, mobile node
OFDM	Orthogonal Frequency Division Multiplex
PCF	Point Coordination Function, used by accesspoints in the form of a RTS/CTS-sequence
PHY	PHYsical layer (of OSI reference model and IEEE 802.11)

RDS	Radio Data Service
RNG	(pseudo) Random Number Generator
RLE	Run-Length Encoding
RSE	Road-Side Equipment, synonym for RSU
RSU	Road-Side Unit, stationary node
RTS/CTS	Request-To-Send / Clear-To-Send, means of claiming the medium
RTTT	Road Transport and Traffic Telematics, similar to ITS
SA	Selective Availability, an error induced in GPS
SIFS	Short Interframe Space
TMC	Traffic Message Channel
TOC	Traffic Operations Center
TTC	Time To Collision, time remaining to collision when no preventive action is taken
VANET	Vehicular Adhoc NETwork
V2V	Vehicle-to-Vehicle (communication)
V2I	Vehicle-to-Infrastructure (communication)
VII	Vehicle Infrastructure Integration
VSC	Vehicle Safety Communication
WAVE	(IEEE 1609) Wireless Access in Vehicular Environments
WSN	Wireless Sensor Network

TrafficFilter-specific terms

FFP	Flood Free Period
MIT	Maximum Inter-TM period
SNP	Source Node Priority
TFL	Traffic Map Flood Limit
TM	TrafficMap
ACT	Actual situation