Simulation of Car-to-Car Messaging: Analyzing the Impact on Road Traffic

Stephan Eichler
Institute of Communication Networks
Munich University of Technology
s.eichler@tum.de

Benedikt Ostermaier, Christoph Schroth
Munich University of Technology
visiting BMW Research and Technology
benedikt.ostermaier@mytum.de
christoph.schroth@cdtm.de

Timo Kosch
BMW Research and Technology
timo.kosch@bmw.de

Abstract

Even though currently there is a lot of research going on in the field of short range wireless data communication between vehicles, inspired by the hope that this new technology can greatly improve traffic flow and prevent accidents, the effects have not yet been clearly analyzed. In this paper, we present one of the first approaches to gain insight into the benefits and drawbacks of deploying a so-called vehicular ad hoc network (VANET). In order to analyze such a complex system, a comprehensive simulation environment is needed. The requirements from the application side on such an environment are described and it is evaluated in how far existing simulation tools meet those requirements. Further, we are considering the traffic effects of a vehicle-to-vehicle traffic obstacle messaging application as one representative for the class of real-time traffic and safety information systems. For these kinds of applications, mainly traffic simulation and network simulation need to be combined. In addition, we discuss the integration of driver, vehicle and radio propagation models into a simulation environment that can be used to Analise the different kinds of VANET applications.

1. Introduction

The availability and applicability of Wireless LAN technologies to dedicated short range communication in automotive traffic along with recent advances in ad hoc network technology have spurred the development towards the establishment of decentral wireless inter-vehicle communication, often called vehicular ad hoc networks (VANETs). While more and more application ideas, algorithms for data exchange and forwarding, concepts for security, and adaptations of the Wireless LAN technology have been developed in this area recently, the benefit of the technology on traffic safety and efficiency has not yet been more than a mere hope. Partly this may be due to the fact that assessing the effects of deploying specific applications is a somewhat complex matter. Deploying a prototype in real world and analyzing the effects is almost impossible, since dozens of prototype vehicles would have to be built, operated and observed. The costs of such an approach are prohibitive. Thus, it is necessary to simulate this very complex system first. Simulation is not a simple task either, since it is necessary to simulate vehicle movement, radio propagation, network protocol behavior, vehicle and driver behavior. Fortunately, microscopic traffic models, driver models, vehicle models, radio propagation models and wireless system models are all available, but need to be combined and integrated with new and specific VANET protocols and respective applications. Combining the different parts is also challenging because of the interdependencies. In addition, existing simulators are not implicitly compatible, neither on the modeling side nor on the implementation side. Whether simulation is time-discrete, time-continuous or event-based or whether a simulator is available only on specific computing platforms and only offers a certain interface has direct implications on the usage and integration with other simulators.

This paper addresses the construction of such a combined simulation environment for the analysis of VANET applications. The main building blocks and their connections are described after a brief overview of related work. The requirements posed by different application classes are explained and their impact on the simulation environment is analyzed. Existing simulation environments for the network and traffic simulation part are presented and their use in a combined simulation environment is assessed. We explain our current choices of simulation modules and present our approach for their integration. We show how we simulate the traffic effects of an obstacle information system based on vehicle-to-vehicle messaging. First results obtained by our simulation are presented.

2. Related Work

So far, most simulations in the field of car-to-car communication rely on rather simple tools for generating network node mobility. For example with the help of BonnMotion [1], simple vehicle movement patterns can be calculated and stored into so-called trace-files. These are utilized by network simulators and used as static representation of node mobility. Thereby, message dissemination can be investigated within the VANET. However, effects of warning messages on traffic behavior cannot be taken into account, since node movement has been completely computed before network simulation.

One of the rare projects dealing with a feedback mechanism between simulators for road traffic and wireless networks is being conducted at the Ohio State University (the OKI project, [18]). The OKI project is focusing on the potential of using VANETs to increase traffic safety, especially in the context of Intersection Collision Warning Systems [11]. In this project, all vehicles approaching an intersection automatically exchange messages containing their positions, velocities and headings. Based on this information, the system in every vehicle computes whether there is a danger of collision with another vehicle. If so, the driver gets a warning and can thus take appropriate action like conducting an emergency brake. In order to evaluate the system, a simulation environment has been implemented for the OKI project. The driving simulator aims at exactly modeling the movement of cars driving across an intersection with a precision of milliseconds. The car-to-car communication is included in a relatively basic manner. A feedback loop between the driving simulator and the communication component ensures that drivers' reactions on message reception are continuously taken into account by the vehicle movement simulator, and that the network simulator, on the other hand, knows about the vehicles' current position during the whole simulation time. Using such an integrated simulation environment enables the evaluation of the intersection collision warning system in the case of the OKI project and is a necessary precondition for the analysis of the impact of VANET applications in general. In this paper, we concentrate on a slightly different application scenario with a local traffic obstacle that is communicated in a multihop fashion. However, we describe the simulation environment in a general manner for simulating and evaluating different kinds of applications.

3. Requirements on Simulation Environment

In general, traffic effects for either safety or efficiency require a changed behavior in longitudinal and lateral control of single vehicles based on additional information obtained by car-to-car communication. Simulating the traffic, calculating vehicle positions, and then feeding a network simulator with this data is fine for simulating network behavior in general but provides no means for the traffic simulator to change the movement of the vehicles based on the information received.

Traffic flow improvement and active safety applications pose different requirements on the simulation environment. For traffic flow analysis, it is important to model traffic behavior and route choices drivers are taking with their vehicles based on information they have. Change of behavior means choosing a different route along the street network based on information on the current traffic situation. Usually, there is some time in the order of seconds or minutes for the single vehicle changing its route after a certain event happened. For active safety, vehicle control needs to be adapted in order to prevent an accident that would otherwise have happened, based on additionally available information in a vehicle. Such collision avoidance can be based on different driver behavior after receiving a specific warning or being supplied with more information on the local traffic context. It can also be based on automatic vehicle reaction. In case of the driver, reaction time is usually around a second. In case of an automated vehicle reaction, an action is usually taking place within milliseconds.

The time duration which is allowed for the vehicle movement to adapt to information received from vehicle-to-vehicle communication poses constraints on the simulation environment. For traffic flow improvement applications it is fine if the vehicle movement simulator starts adapting vehicle behavior one or a few seconds later than a message is received by the network simulator, whereas automatic vehicle control simulation needs to know about incoming data packets almost immediately. When traffic and network simulation are both time-discrete, this poses very different levels of constraints on synchronization between the simulators. Also, if the reaction needs to be done within milliseconds, effects of wireless data communication which can embody varying delay times in the order of milliseconds as well as lost data packets, the wireless channel and the wireless communication system need to be modeled much more precisely. In this paper we concentrate on the case with the lowest timing requirements and traffic flow optimization. However, the simulation environment is intended to be used for the analysis of active safety applications as well. Therefore, the requirements for safety applications based on driver information are also consid-

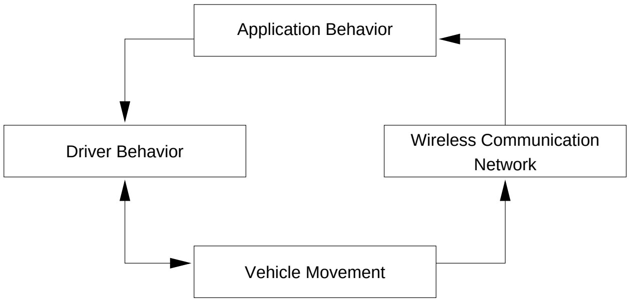

Figure 1. Overall Simulator System Concept

ered. For now, automatic vehicle reaction is out of scope of our work since the requirements not only on simulating these scenarios but also implementation issues concerning system penetration rate and system robustness are so high that it is unlikely any kind of applications in this context will be implemented in a real life scenario any time soon.

4. System Concept

For all application types considered in the previous section, it is necessary to model vehicle movement for every single vehicle, the wireless communication network, the application behavior itself and the behavior of the driver reacting to information presented by the application. Our current simulation environment, reflecting these required models, consists of four logical components which are depicted with their relations in Fig. 1.

The vehicle movement is realized with a microscopic traffic simulator. A network simulator imitates the behavior of the wireless communication network. The network simulator comprises a model of the radio propagation characteristics along with implementations for radio channel access, error correction, data packet routing and data security, to name the most important ones. The application that is used to demonstrate the concept in this paper is a traffic obstacle information application. It consists of a part that decides whether to send such a traffic obstacle information message and a part that receives this message and decides whether to forward the information to the driver. The driver behavior in this example consists of an algorithm to make route choices to reach the destination based on information on the traffic situation. This information is presented to the driver by the application. Therefore, the driver behavior depends on the application behavior but not vice versa since the driver does not directly interact with the application. For example, the drivers decide to move around a dangerous zone in case they have already been warned about it. The driver controls the vehicle and so vehicle movement depends on the driver's control behavior where the driver's behavior also depends on the vehicle's movement.

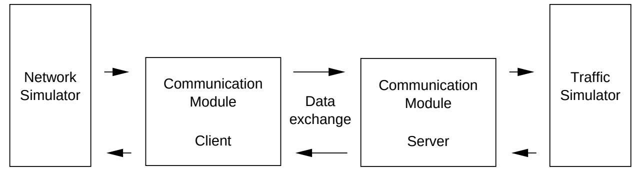

Figure 2. Coupling of Network- and Traffic-Simulator

The communication network depends on the vehicles' positions and therefore on vehicle movements. The network simulator must permanently be notified about the positions of the vehicles that participate in the network in order to have the current connectivity pattern available. The application depends on the communication network to send and receive messages. The application can be implemented as an additional module or as part of the network simulator. In our case, the application is integrated in the network simulator. Also, driver behavior has been integrated with this application implementation, simplifying our simulation environment to two simulators between which synchronization is needed (see Figure 2). If the application logic is supposed to be implemented in a separate user interface or should be implemented in a different programming language in order to be able to transfer it between simulator and prototype without (major) changes, the simulation environment should be split up into three different parts.

In our solution, the network simulator acts as a client requesting for information from the traffic simulator. For instance, it needs to know both the exact number of cars being part of the network and their geographic positions. The traffic simulator, on the other hand, acts as a server answering the requests of the network simulator. Not all vehicles of the traffic simulation need to be part of the network. Therefore, each node of the network simulation is representing a specific vehicle of the traffic simulation, but the traffic simulation can consist of many more cars. The ratio of equipped to the total number of vehicles can be adjusted.

4.1. Network Simulation

A Network Simulator for VANETs needs to support specific functions and contain specific properties of the special automotive application field. Among the most important criteria for an appropriate network simulation tool is a comprehensive model for mobile wireless networks with a representation of the environment to account for effects like shadowing caused by buildings in urban areas or radio wave reflections due to roads and building walls. These features must be readily available and adjustable within a convenient simulator. Proper antenna models plus adjustable transmit powers and reception thresholds are necessary in order to reproduce the properties of the real wireless modules built into prototyping cars.

An appropriate network simulator also needs a model of the widely accepted communication standard IEEE 802.11, in the following called WLAN, which specifies both the physical and the Medium Access Control (MAC) layer of the ISO/OSI reference architecture. WLANs are currently considered as the most promising wireless technology for VANETs. They have shown relatively low sensitivity to high velocities, sufficient transmission ranges and support fast connection setup times, if configured correctly [15]. In addition to the models included and protocols implemented, a network simulator suitable for our simulation purpose needs to be able to be combined with at least a traffic simulator and thus must either contain an appropriate interface or be available as an open source implementation which can easily be extended. The integration of new network protocols and changes to the other protocol layers must be possible as well.

The simulator GloMoSim [20, 7, 5], would be a possible candidate meeting most of the mentioned requirements. It is the freely available version of the Qualnet simulator. However, we do not consider it further since it has not been updated since 2001.

A different choice is the so-called network simulator 2 (NS2) [6] which is among the most widely accepted network simulation tools in the scientific community. Its software architecture is well prepared for extensions and enables attaching software modules for data exchange with other programs. NS2 features a comprehensive model for simulating multihop wireless networks and includes an implementation of the IEEE 802.11 MAC-protocol. As radio wave propagation models, NS2 basically provides the free space model and the so-called Two Ray Ground model, which takes into account both the direct communication path between two vehicles and an additional path due to reflections on the ground. This model is very well applicable to the VANET domain [17]. In contrast to GloMoSim, NS2 also features a simple model for the representation of reflections, refraction and shadowing effects caused by buildings and other obstructions. For this purpose, a simple black and white bitmap containing a picture of the scenario (in our case does black denote streets, white denotes buildings) is imported and interpreted.

Overall, the network simulator NS2 best meets the requirements and is our choice for the network part of the simulation environment.

4.2. Traffic Simulation

In addition to the network simulator, an appropriate simulation of the vehicle movements is needed. Since we are concentrating on traffic flow effects, we only consider microscopic traffic simulators here. Non of the ones we are aware of support the detailed vehicle physics needed for collision avoidance simulation. For these applications, either the traffic simulator needs to be extended or a different simulation tool has to be used. Besides the obvious requirement of simulating traffic in an as realistic manner as possible, it is crucial to be able to influence the behavior and the routes of cars during the simulation. Additionally to this control feature used by the driver behavior component (and possibly by a vehicle control component that we are not considering here), an interface to exchange data with the network simulation tool is mandatory.

METROPOLIS [10, 8] is specially suited for simulating traffic patterns in large urban environments. It is available for both Windows and Linux operating systems. It enables simulations both in real-time and off-line. A scenario builder and a graphical results viewer are provided to adjust simulation parameters and evaluate resulting data. METROPOLIS provides a so-called mesoscopic [9] view of vehicle movement, which tends to model individual vehicles, but describes their behavior in a simplified manner, on an aggregate level. This approach does not satisfy the need for a fairly detailed modeling of single cars and their reaction on receiving a warning message.

SUMO [2], is an open source mobility simulator, developed at the *Deutsche Gesellschaft für Luft- und Raumfahrt (DLR)* (`http://www.dlr.de`). In contrast to METROPOLIS it a microscopic simulator. The implemented mobility model is space continuous and time discrete. What appears to make it an interesting choice for our purpose is that it supports single vehicle routing, the integration of different vehicle types, and multilane streets with lane changing. Street networks can either be generated or imported from diverse map formats. A graphical interface to control a simulation run is included that, since SUMO is open source, could be extended to control an integrated simulation environment. Hopwever, the major drawback of this simulator is that vehicle routes are computed before the beginning of a simulation. This is a characteristic that inhibits SUMO's use for evaluating the effect of car-to-car messaging. Changes in vehicle behavior and the modification of their routes are a necessary precondition for this analysis.

VISSIM [3] is a commercial traffic simulator, as opposed to the free and open source simulator SUMO. It is a very rich tool, featuring a 3D rendering of simulated traffic scenarios, including not only cars but also trains, busses and trucks as well as persons and surrounding buildings. VISSIM is based on the microscopic, time discrete traffic model from Wiedemann [19]. Street networks are represented in a very detailed manner with the possibility to adjust the number of lanes and the position and behavior of traffic lights. In contrast to SUMO, VISSIM also provides configurable driver models, which by representing the characteristics of car-drivers allows to take, for example, inattentiveness or aggressive behavior into account for the simulation. VISSIM is not available as a source code package, so its functionality can only be customized by adjusting certain parameters. Road maps can only be used by translating them with the help of an additional tool, called VISUM [4]. Through programmable interfaces, it is possible to couple VISSIM with other programs. However, the possibilities to recalculate vehicle routes at runtime are limited. The standard version of VISSIM only allows vehicles to move along routes determined before simulation start. Only with a lot of programming effort, the routes of single vehicles can be made modifiable during simulation time. VISSIM is a very promising tool for the traffic effect evaluation purpose. Nevertheless, a few additional features are needed to fully use it for a VANET analysis as targeted here.

Since none of the mentioned simulators fully met our requirements, especially the possibility to influence vehicle movement during simulation time, we have been building on our own traffic simulation solution, called CARISMA Traffic Simulator (CARISMA-TS). CARISMA-TS has been described in [13]. It supports mainly inner-city scenarios with a few hundred to a few thousand vehicles. A number of simplifying assumptions are made: CARISMA only supports two lanes per road and all streets have the same traffic capacity and priority. The simulator works on a time- and position-discrete basis and updates vehicle positions and directions every second. Street networks can be imported from digital map sources in ESRI-Shape-file format [12]. The mobility model applied mainly follows the widely accepted Krauß-model [16]. Different models for destination and route selection are implemented. Intersection conflict management works according to the simple four-way stop principle.

What made CARISMA-TS our choice for the integrated simulation environment, in spite of its comparatively weaker traffic model, was its ability to compute vehicle routes during simulation time. As soon as a car arrives at its destination, both a new destination and the shortest possible path to this new destination are computed. The route decision can be periodically evaluated and altered if necessary at any point in simulation time. Numbers can be freely assigned to street segments. The route search algorithm then evaluates different possible paths to a destination by comparing their cost based on the assigned values. Hence, by assigning a high number to a certain street segment, the algorithm will pick alternative roads as part of a new route for a vehicle. CARISMA-TS features a radio wave obstruction model that it can provide to the network simulator. It assumes that buildings are located along all streets and periodically computes connectivity information for every vehicle.

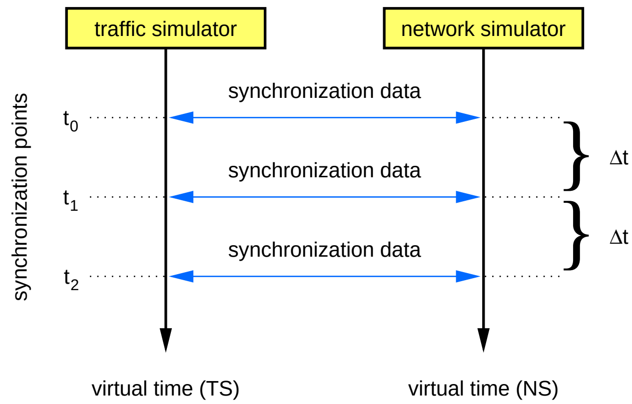

Figure 3. Basic Simulator Coupling Concept

Among the three presented traffic simulators, CARISMA has been chosen due to its open architecture, the compatibility with existing map formats, the availability of a simple line-of-sight model and the possibility to easily realize dynamic routing.

4.3. Coupling Concept

In order to evaluate impacts of disseminating warning messages on traffic patterns, the traffic and network simulator have to run simultaneously and permanently exchange synchronization data among each other. Synchronization data is exchanged at certain synchronization points in virtual time of the two simulators. We denote this concept as coupling of simulators. The coupling starts with an initialization phase, where data concerning the simulation scenario, such as the number of nodes, is exchanged. Afterwards, the actual simulation is started and synchronization data is exchanged periodically between the two simulators.

More precisely, both simulators will simulate a certain amount of virtual time, which we denote as a simulation period Δt. Note that Δt has to be identically for both simulators. Then the simulators will wait for each other, exchange synchronization data and then again simulate Δt units of virtual time (Fig.3). The simulation period Δt should be small to ensure a good alignment. However, if it is too small, computational performance is heavily reduced by adding too much overhead for synchronization.

The parallelism of this concept should provide a gain in performance when running simulations on either multiprocessor machines or on two separate PCs. However, it also introduces some inaccuracies, as one simulator may use data gathered from the last synchronization point, which has already been updated in the other simulator. This is because the alignment of synchronization data is done only periodically, and not when needed by one of the two simulators. Depending on the type of synchronization data, these inaccuracies may be mitigated, as will be shown in Sec. 5.3.

For the actual data transfer between the two simulators, some concept of inter-process communication is needed. Since the coupled simulators may run in different environments, cross-platform interoperability is necessary. Furthermore, the connection should be fast and reliable to ensure good performance. Finally, it should be possible to easily extend the communication mechanism to meet new requirements. Therefore, a standard TCP connection has been chosen to couple the two simulators, running a simple request-response protocol. In terms of implementation, reusability and adaptability of the connection were taken into account. Therefore, the simulators used in our setup can easily be swapped for other ones without major changes in the coupling concept.

5. Implementation

After defining adequate components of the simulation environment, new functionality has been implemented into each of them.

5.1. New Ad Hoc Agent for NS2

Within the framework of this work, a comprehensive model of a new ad hoc agent has been implemented into the NS2 environment, which determines the network nodes' behavior while sending, receiving or forwarding data. It basically implements the message dissemination algorithm for the whole simulation. Messages are flooded to all reachable other nodes, taking care of certain areas of relevance and time limits. The findings of [14] were used as a basis for designing an efficient message dissemination protocol.

5.2. Modifying CARISMA

The CARISMA traffic simulator must also be modified in order to fulfill the requirements of the coupled simulation environment. Its main new task is to periodically receive data concerning changes in vehicle behavior from the network side, realize them and send new node positions and connectivity patterns back to the network simulator. In case a vehicle is warned against a local hazard, for example, it may desire to seek a route around the danger zone. It is CARISMA's job to implement possible route changes in real-time.

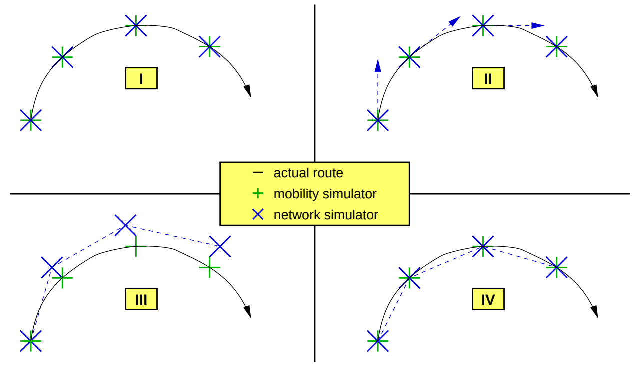

Figure 4. Route Deviation of Different Coupling Approaches

5.3. Realization of Simulator Coupling

Both simulators start with the initialization phase, where CARISMA transmits the geographic scenario size and the number of vehicles to NS2. Afterwards, the simulation phase follows, where synchronization data is exchanged periodically. At each synchronization point, CARISMA informs NS2 about the actual vehicle positions and connectivity data, whereas the traffic simulator relies on being notified about desired changes in the vehicles' routes.

In our case, the minimal length of the simulation period Δt is determined by CARISMA, since this simulator updates the vehicle positions and directions once every second. Hence, the highest possible update frequency between both simulators is one data transfer per second of virtual simulation time. Nevertheless, this interval turns out to be a good trade off between optimum simulator alignment and computational efficiency, since the interaction between the inter-vehicle communication and traffic flows does not have to be investigated on a millisecond basis like in an intersection collision warning system.

Further issues concerning the data exchange are the different coordinate systems, which have to be adapted, and the different level of accuracy when representing vehicle movement. CARISMA simulates acceleration and deceleration of vehicles, where the mobility model of NS2 only supports nodes moving at a constant speed. Four approaches to mitigate the position deviations have been investigated in detail within this work and will be presented consecutively.

Coupling Approach I At each synchronization point t_x, the traffic simulator simply sends the current positions of the vehicles to the network simulator. The network simulator sets the positions of the nodes accordingly. Thus, exactly equal vehicle positions are guaranteed at each synchronization point, which can be seen in Fig. 4 (I).

Nevertheless, this involves several severe problems. For moving a node from one point to the other within one time step, NS2 requires a geographical destination and data regarding the velocity with which the node is supposed to move. Furthermore, the network simulation strongly lacks in accuracy due to the fact that nodes do not move continuously, but remain at a fix position for a whole time step. Thus the mobility model of NS2 lies idle. Connectivity between vehicles would be estimated unrealistically, since the network topology would not change within a whole second. In a real-world scenario however, the structure of VANETs permanently changes due to vehicles connecting or disconnecting when driving through an urban environment.

Coupling Approach II This is an extension of the previous approach. At each synchronization point t_x, the traffic simulator does not only send the vehicle positions at t_x, but also the estimated vehicle positions at t_{x+1} to the network simulator. Estimations are based on current velocity and driving direction. NS2 sets the positions of the nodes and updates the nodes' speed and direction based on the estimated positions at t_{x+1}.

This solution also features consistency of vehicle positions after synchronization. Additionally, the mobility model of NS2 is fully utilized, so the simulation results should get more accurate. However, this concept introduces some inaccuracies which result from the extrapolation of the destination positions for the next synchronization point. The approximation is rather rough, however, since velocity is wrongly assumed to remain constant during the whole interval. Also, bending of roads is not taken into account by this approach. As a consequence, a considerable node positioning error may occur before each synchronization point, which is visualized in Fig. 4 (II). Additionally, there are still leaps in the movement of the nodes in NS2.

Coupling Approach III The third approach differs from the previous approach in that at each synchronization point t_x only the estimated positions at t_{x+1} are transferred to the network simulator. Like in the previous approach, NS2 sets the positions of the nodes and updates the nodes' speed and direction based on the received positions. An exception is the first synchronization point t_0, where the vehicle positions at t_0 are also transferred. The network simulator will set the nodes' positions only at t_0. At every synchronization point it will then only update the nodes' speed and direction. Hence, there are no more leaps in the movement of the nodes in NS2, and this approach also utilizes its mobility model.

On the other side, the inaccuracies which result from the estimation of the destination positions for the next synchronization point are even bigger than that of approach II, which can be seen in Fig. 4 (III). Nodes generally tend to cover a bigger distance in NS2 than in the mobility simulator. Furthermore, at the synchronization points succeeding t_0, there is no consistency of the vehicles' positions between the two simulators.

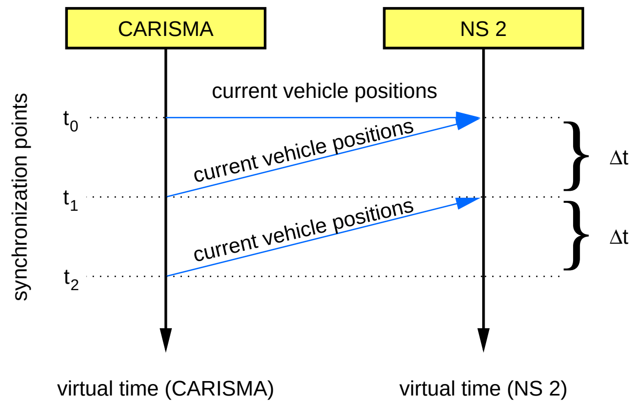

Figure 5. Sequence Chart for Approach IV

Coupling Approach IV The last approach is a refinement of approach III. The idea is to give the traffic simulator a "head start" of one simulation period, so that the vehicle positions for the next synchronization point are no longer estimated, but calculated. At the synchronization point t_0, the traffic simulator sends the current vehicle positions to the network simulator, which will update the nodes' positions accordingly. Then, the traffic simulator proceeds to t_1, while the network simulator does not do any computing and thus its virtual time is still t_0. For all further steps, when the network simulator is at t_x, the traffic simulator is at t_{x+1}, so the traffic simulator will keep a head start in virtual time of Δt. At each succeeding synchronization point, the traffic simulator then sends its current vehicle positions at t_{x+1}, which will be received by the network simulator at t_x and used to update the nodes' speed and directions. This concept is shown in Fig. 5.

With this solution, the consistency of vehicle positions at synchronization points is ensured. The mobility model of NS2 is utilized, and there are no leaps in the movement of the nodes in this simulator. As illustrated in Fig. 5 (IV), there are still deviations between the actual node movement and the movement in NS2. Nodes tend to cover a smaller distance than in the mobility simulator. However, this deviation is smaller than that of the 2nd or 3rd approach, since we are now using interpolation to calculate the node movement in NS2 between synchronization points. The head start of the traffic simulator introduces a new drawback: Since the traffic simulator is at t_{x+1} when the network simulator is at t_x, requested route changes arrive at the traffic simulator Δt units of virtual time too late. However, we ar-

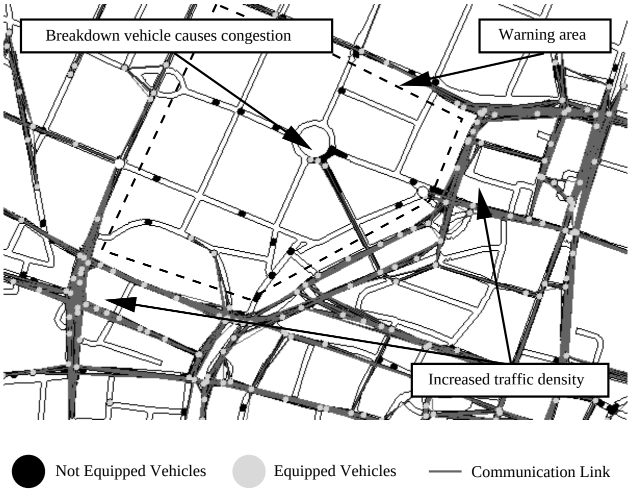

Figure 6. CARISMA Snapshot

gue that this is negligible as long as Δt is not much longer than the average reaction time of a driver on a route change notification by the navigation system.

We have therefore chosen the 4$^{\text{th}}$ approach to align the position of the vehicles between the two coupled simulators.

6. Simulation and Results

The scenario described in Chapter one has been simulated within the environment consisting of NS2, CARISMA and the data exchange mechanism. 900 vehicles are driving through an 8km^2 scenario, of which a small cut-out is depicted in Fig. 6. 400 of these cars are equipped with wireless communication modules (gray dots) and thus can send, receive and forward data. In our scenarios a communication range of 400m has been applied. One specific vehicle breaks down in the beginning of the simulation run, stops moving and starts to periodically broadcast a warning message of defined content. Cars receiving this message seek new routes and drive around the danger zone, which comprises all roads closer than 250m to the breakdown location. Since NS2 and CARISMA are linked, the traffic simulator takes into account these route changes of the warned cars during simulation. However, not warned vehicles (black dots) still drive through the danger zone and possibly get trapped in congestion. A CARISMA snapshot (Fig. 6) shows this scenario after a simulation time of 300s. Average velocities of warned and not warned cars have been logged as a means for quantifying the value proposition of car-to-car messaging.

Several simulation runs were conducted for different vehicle densities and percentages of wireless enabled cars. In most cases, warned cars turned out to be considerably faster than those not warned in this simulation scenario. But in case of a high number (Fig. 7 shows the dependence of velocity development on traffic density in case a warning message is distributed) of vehicles trying to drive around the hazard, traffic density heavily increases and thus average velocities decrease on the alternative routes, as can be seen in Fig. 6. This negative impact of introducing dynamic traffic information must be also taken into account when evaluating the benefits of VANET applications.

Within three further experiments, the average velocities of 1100, 900 and 700 wireless enabled vehicles driving on an 8km^2 scenario were observed. To eliminate statistical aberrations as far as possible, the simulation was conducted 10 times per curve. Fig. 7 shows the changes in vehicle speeds. In the beginning of the simulation, cars drive around according to the so-called Waypoint Uniform mobility model, where average speeds remain at a rather constant level. After a time of 400s (t_1), a vehicle breaks down and begins to periodically send out warning messages containing, among other data, its geographical location. Due to the high vehicle density, the connectivity is excellent in all three cases, so that the messages are disseminated through the whole network within a few seconds. The breakdown car stops, but causes almost no congestion since all vehicles are warned. Hence, these vehicles consider roads closer than 250m to the accident as within a danger zone and seek alternative shortest paths to their destinations.

As a consequence, traffic density becomes higher and thus average vehicle speeds lower on the streets surrounding the location of the accident. Only the velocities of cars being closer than 700m to the breakdown car have been logged, since this area includes the streets becoming more congested due to the dynamic route changes. As can be seen in Fig. 7, average speeds heavily decline after cars are warned in case of high traffic densities, whereas for less than 700 vehicles almost no speed drops were observed. In case of low vehicle densities, the number of cars choosing an alternative route around the hazard is not very high. The few warned vehicles do not considerably affect traffic flow on these roads and thus decreasing average speeds were not observed.

High densities, in contrast, cause congestion-like traffic patterns on the alternatively chosen roads around the breakdown car, which reduces the actual benefits in travel time. A message lifetime of 300s was chosen in this example. Hence, at 700s (t_2), most of the vehicles drop the message because it is not up to date any more. They consider the congestion to be disappeared. Velocities rise again and reach the former levels.

In order to roughly quantify the overall benefits of using IVC for warning against a local danger, the average velocities of warned and not warned cars have been logged during a 500s simulation. The results depicted in Fig. 8

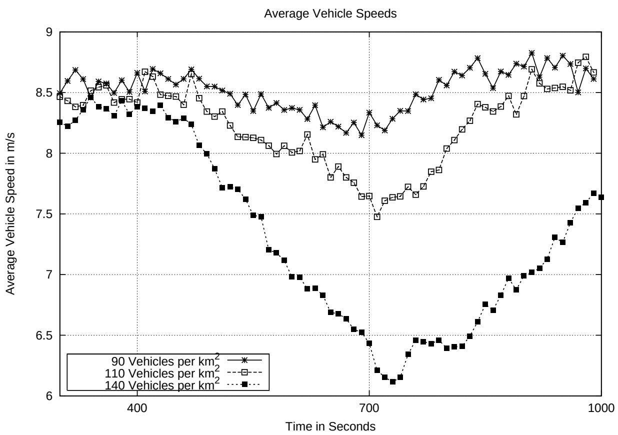

Figure 7. Average Vehicle Speeds

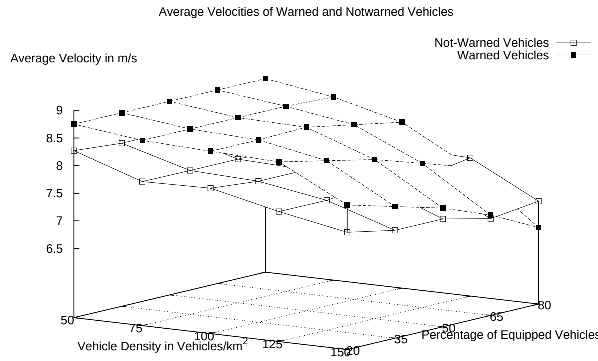

Figure 8. Comparison of Average Speeds between Warned and Not-Warned Cars

prove that wireless enabled cars strongly benefit of receiving warning messages in most cases. Only if traffic density is very high and the percentage of wireless enabled cars, too, warned cars are driving at even lower velocities than those not warned due to the effects described above. This effect can be seen in Fig. 8; the two graphs intersect and the dark graph drops below the white graph. In case numerous vehicles receive warning messages and try to drive around the hazard location, high traffic densities and thus lower average speeds occur on their routes. In the worst case, heavy congestion emerges on these alternative routes. The area within the fix warning radius of 250m, in contrast, is almost free of cars after a while. Only a few mostly not warned cars get trapped in the congestion behind the breakdown car, whereas others can drive through the area at high speeds. Hence, being warned can even be of remarkable disadvantage in some situations. Last, the surfaces decline towards high vehicle densities, since a high volume of traffic generally results in lower average speeds.

Results from real-world experiments would probably slightly vary from those visualized in Fig. 8, depending on the specific drivers' behavior. Within the framework of this work, all drivers, which are warned against using the roads around the accident indeed seek an alternative route. In reality, however, a certain percentage of drivers might simply ignore the warning popping up on their navigation interface or at least would react at a later point in time, than it is assumed within this simulation run. Hence, a more comprehensive driver model could be implemented in order to achieve even more realistic simulation results.

Even more issues should be considered when designing a real-world application. In contrast to the CARISMA model, roads normally do not have equal traffic capacities. In case a vehicle breaks down on a huge highway causing congestion, for example, it is not necessarily recommendable for all following cars to change their route and switch to a small road with only low capacity. In case many wireless enabled cars are warned and recommended to switch to the small road, they would possibly even drive at a lower speed than those remaining on the highway due to the extreme capacity overload of the country road.

7. Conclusion

This paper provides an overview of a concept for coupling a network and a road traffic simulator. With the help of such a simulation environment, the impacts of vehicle-to-vehicle communication on traffic can be investigated in detail, which is crucial for evaluating the benefits for traffic safety and throughput. The simulation environment set up is a big step forward to a realistic representation of different scenarios, where the used simulators still show some weaknesses. Future work is devoted to refine and extend the models applied; CARISMA is expected to be replaced with VISSIM, which provides a more detailed and realistic representation of vehicle mobility. We are also planning to implement the application logic in a separate application environment simulator. This new simulator will provide the environment which an application running in a car depends on. This should enable us to use applications running in the simulation environment in real-world prototypes and vice versa without major modifications. Also, a component for controlling the whole simulation environment shall be added. It will enable a user to design, deploy and after all run simulations with customized behavior. With the help of this tool, the message dissemination algorithm as well as the drivers' reaction on warning message reception and many more parameters can be adjusted via an easily adoptable user interface.

8. Acknowledgments

We would like to thank both Florian Dötzer and Markus Strassberger for their helpful discussions during this work.

References

[1] Bonnmotion - a mobility scenario generation and analysis tool. `http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/`.

[2] Sumo - simulation of urban mobility. `http://sumo.sourceforge.net/`.

[3] Vissim. `http://www.ptv.de/cgi-bin/traffic/traf_vissim.pl`.

[4] Visum. `http://www.english.ptv.de/cgi-bin/traffic/traf_visum.pl`.

[5] GloMoSim - global mobile information systems simulation library. `http://pcl.cs.ucla.edu/projects/glomosim/`, February 2001.

[6] The network simulator - ns-2. `http://www.isi.edu/nsnam/ns/`, 2005.

[7] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla. Glomosim: A scalable network simulation environment. Technical Report 990027, UCLA Computer Science Department, May 1999.

[8] A. de Palma and F. Marchal. Metropolis - a dynamic simulation model designed for atis applications. In *Proceedings of the International Conference on Traffic and Transportation Studies (ICTTS)*, pages 770–781, 1998.

[9] A. de Palma and F. Marchal. Real cases applications of the fully dynamic metropolis tool-box: an advocacy for large-scale mesoscopic transportation systems. Technical Report 2001-18, THEMA, 2001.

[10] A. de Palma and F. Marchal. Metropolis. `http://www.adpc.be/software/metropolis/index.html`, June 2003.

[11] A. Dogan, G. Korkmaz, Y. Liu, F. Ozguner, U. Ozguner, K. Redmill, O. Takeshita, and K. Tokuda. Evaluation on intersection collision warning system using inter-vehicle communication simulator. In *Proceedings of the 7th International IEEE Conference on Intelligent Transportation Systems*, pages 1103–1108, October 2004.

[12] ESRI. ESRI shapefile technical description. `http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf`, July 1998.

[13] T. Kosch. Efficient information dissemination in vehicle adhoc networks. In *Proceedings of the 11th World Congress on Intelligent Transportation Systems (ITS)*, Nagoya, Japan, Oktober 2004.

[14] T. Kosch. Efficient message dissemination in vehicle adhoc networks. In *Proceedings of the 11th World Congress on Intelligent Transportation Systems*, October 2004.

[15] T. Kosch and C. Schwingenschlögl. Mobile ad hoc networking testbed for vehicle-to-vehicle communication. In *Proceedings of the Fifth International Conference on Systemics, Cybernetics and Informatics (SCI)*, Orlando (FL), USA, July 2001.

[16] S. Krauß. *Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics*. PhD thesis, Universität zu Köln, 1998.

[17] T. Tank and J.-P. M. Linnartz. Vehicle-to-vehicle communications for avcs platooning. *IEEE Transactions on Vehicular Technology*, 46(2):528–536, Mai 1997.

[18] D. K. Tokuda, M. S. Nakabayashi, M. M. Hamaguchi, and M. H. Tsutsui. Oki project. `http://www.ece.osu.edu/hpcnl/okipublic/`, 2004.

[19] R. Wiedemann. Simulation des strassenverkehrsflusses. Schriftenreihe des Instituts für Verkehrswesen der Universität Karlsruhe Heft 8, Institut für Verkehrswesen der Universität Karlsruhe, 1974.

[20] X. Zeng, R. Bagrodia, and M. Gerla. Glomosim: A library for parallel simulation of large-scale wireless networks. In *Proceedings of the 12th Workshop on Parallel and Distributed Simulation*, pages 154–161, 1998.