VANET Simulation Environment with Feedback Loop and its Application to Traffic Light Assistance

Axel Wegener
Institute of Telematics
University of Lübeck, Germany
Email: wegener@itm.uni-luebeck.de

Horst Hellbrück
Department of Electrical Engineering
University of Applied Sciences Lübeck
Germany
Email: hellbrueck@fh-luebeck.de

Christian Wewetzer
and Andreas Lübke
Volkswagen AG
Wolfsburg, Germany
Email: {christian.wewetzer, andreas.luebke}@volkswagen.de

***Abstract*—Traffic applications, in which vehicles are equipped with a radio interface and communicate directly with each other and the road traffic infrastructure are a promising field for ad-hoc network technology. Vehicular applications reach from entertainment to traffic information systems, including safety aspects where warning messages can inform drivers about dangerous situations in advance. As performance tests of the real system are very expensive and not comprehensive, today's evaluations are based on analysis and simulation via traffic simulators. In order to investigate the impact of traffic information systems there are two options: First, traffic simulators can be extended by application code and a simplified model for wireless communication. Second, existing network simulators can be coupled with existing traffic simulators. We favor the coupling of existing and well known simulators as we believe that the wireless communication characteristics influence the data transfer significantly and an oversimplified transmission model can lead to flawed results. In this paper we describe the feedback loop between traffic and network simulators named Traffic Control Interface (TraCI) and outline its versatility. We explain its use to determine possible energy consumption reduction when traffic lights send their phase schedules to vehicles.**

I. Introduction

Research efforts in vehicular ad-hoc networks (VANETs) have increased in the past. The main reason for this rising interest is that car to car as well as car to infrastructure communication has a very high potential of extensive use compared to ad-hoc communication between smartphones, PDAs and laptops. Many future challenges like reduction in fuel consumption and improving the safety of traffic can hardly be solved without interaction between vehicles and the infrastructure.

Comparable to many other fields of innovations the development follows four phases. First, analytic studies and calculations or estimations indicate the need for further investigations on an interesting idea. Second, simulations prove the technical feasibility and reduce the risks for further more expensive steps. In the third phase a prototypical implementation is needed especially for vehicular applications to ensure user acceptance and to design a suitable human machine interface. Fourth, product development will engineer devices suited for mass production. As VANET applications need standardization between different manufacturers and further international regulations are needed, the development of VANETs today is still in an early stage.

We will illustrate the above statements with an example that we will refer to for the rest of this paper. The idea is to save energy by informing cars approaching a traffic light about future phases of the lights. An analysis of GPS-traces of cars in a city in combination with traffic light positions indicated a huge potential for fuel reduction due to idle running engines during the red phase and unnecessary hard braking of cars although early smooth slowdowns would have been more efficient. Analytic results and estimations help to find potential fields for improvements but they cannot verify the feasibility of the suggested applications. Therefore, we will perform a series of simulation runs with varying parameters to investigate the feasibility of a VANET approach to implement this idea.

The solution to this task leads to the following contributions of this paper:

- Illustration of the effective simulation setup with feedback loop by using TraCI, ns-2 and Sumo.
- Outline for a development of a distributed "trafficlight-2-car" application.
- Extensive evaluation of the application's impact on energy consumption of vehicles.

The paper is structured as follows: After introducing related work, we will explain the implementation of a feedback loop between network and traffic simulator as a basis for implementing and evaluating our application. Thereon follows a detailed description of the application and an evaluation of its influence on energy consumption. In the last section we conclude the paper and outline future work.

II. Related Work

In recent years, much research has been done on the coupling of vehicular traffic simulators and network simulators. We distinguish between three ways of coupling these simulators: Offline coupling (trace file), online one-way coupling, and online two-way coupling (feedback loop). In offline coupling, traffic simulators like VISSIM[1] or Sumo [1] are used to obtain realistic movement traces of vehicles as files. These files are then converted to movement traces of mobile network nodes for use in network simulators like ns-2 [2]. This kind of

[1]http://www.ptv.de/cgi-bin/traffic/traf_vissim.pl

978-1-4244-3062-8/08/$25.00 ©2008 IEEE

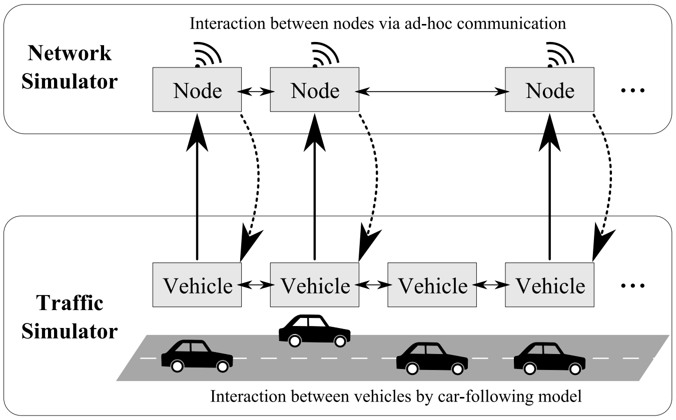


Fig. 1. Feedback loop between simulators provided by TraCI

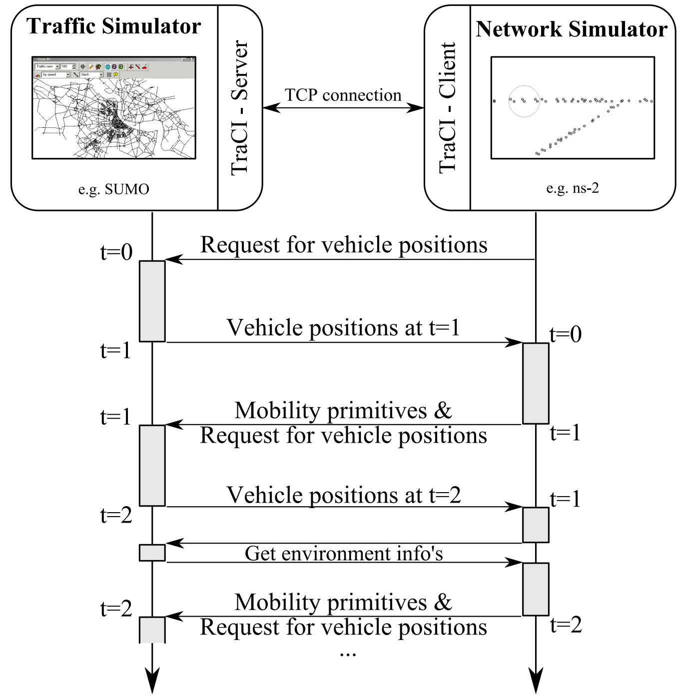


Fig. 2. Sequence diagram of simulators' interaction during simulation start

coupling is used, e.g., by Karnadi et al. [3]. Online one-way coupling [4], [5] eliminates the sequential file-based process by having both simulators run in parallel. The most advanced form of coupling is online two-way coupling [6], [7]. This way, data is not only sent from the vehicular traffic simulator to the network simulator but also vice versa. The application code running in the network simulator is able to influence vehicular traffic, thus altering behavior of individual vehicles during a simulation. In this paper, we use online two-way coupling since only this approach allows for an evaluation of the effects on traffic when vehicles react to traffic light phase schedule information.

In contrast to research on simulators and their coupling, only few research results exist that verify positive effects of inter-vehicular communication. In this paper we investigate positive effects of traffic light schedule information on energy consumption of vehicles, similar to the work of Richter [8]. While Richter solely performs simulation of vehicular traffic and simply assumes a technology capable of distributing this information to vehicles, we simulate both communication technology and vehicular traffic. [9] investigates highway merging improved through inter-vehicular communication in a self-developed traffic simulator. Wireless communication is not taken into account; instead it is assumed that vehicles know location and speed of their neighbors.

III. Simulation Environment

VANET applications aim to improve traffic-safety and -capacity. Therefore, additional information is provided to the driver, who adapts his driving behavior accordingly. In our work we build a feedback loop to our simulation environment as depicted in Fig. 1. In the network simulator each node moves according to a vehicle driving in the traffic simulator. Messages exchanged by the nodes lead to traffic information like "red light in 500 m". The application code will then switch the vehicles' behavior in the traffic simulator to coasting, closing the feedback loop.

In our approach, a network simulator runs interactively with a traffic simulator. We extend the VANET application to model the influence of drivers' behavior by controlling vehicle movements in the traffic simulator. To also integrate effects that will appear during market launch, when only a certain subset of vehicles is equipped with an ad-hoc network interface, we map only a fraction of vehicles to the network simulator as given by the *penetration rate*.

Technically, a road traffic simulator is coupled with a network simulator over a TCP connection by using the Traffic Control Interface (TraCI) [6] as shown in the upper half of Fig. 2. In this paper we use the road traffic simulator Sumo (Simulator for Urban Mobility) [1], which uses a microscopic car following model as proposed by Krauss [10]. The TraCI-Server is implemented as a control module within Sumo. Therefore, Sumo is started with an option, so that the TraCI-module controls the traffic simulator's behavior. The TraCI-Server instructs Sumo to load the road traffic scenario and then listens for a TraCI-Client to connect. On the other side, the correspondent TraCI-Client was implemented in the network simulator ns-2 [2]. In this implementation we designed the client as a C++ class that is accessible from the C++ as well as the TCL part of ns-2. During simulation set-up of ns-2, it connects to the TraCI-Server via TCP.

After successful connection, the network simulator acts as master by sending commands to the road traffic simulator, that are answered – if necessary – with the requested data. The commands can be categorized in three groups:

Simulation control
: These commands are sent periodically to advance simulation time and receive vehicle positions from the traffic simulator.

Mobility primitives
: Every complex mobility pattern, which is a result of

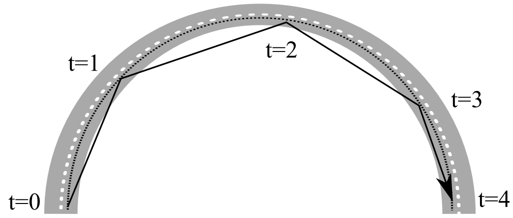


Fig. 3. Handling of vehicles' movements

a decision taken by a driver can be decomposed into a sequence of mobility primitives as commands for a specific vehicle. Those are, e.g., 'change speed' or 'change route'.

Traffic scenario data request
: Since the scenario is kept in the traffic simulator, commands from this group allow the network simulator to access the road map in a comfortable way, e.g., perform driving distance calculations or get the status of traffic lights or arbitrary 'points of interest'.

The simulation is performed in discrete time steps to ensure synchrony in both simulators. Fig. 2 shows a sequence diagram that points out the interaction and resulting synchronization between the simulators. First, the network simulator sends a command to trigger the traffic simulator to start the first timestep of the simulation. After that, the traffic simulator replies with the initial positions of the vehicles. Those vehicles get assigned to network nodes and are placed at their initial position. After simulating the according discrete time step in the network simulator as well, the mobility primitives generated by the VANET application during this time step are sent to the traffic simulator together with the command to perform the next simulation step. The reply provides the vehicle positions at $t = 2$. Those are translated into linear movements that get scheduled in the network simulator. This communication scheme repeats throughout the interactive simulation until the network simulator finishes the simulation. Additionally, traffic scenario data requests can be sent at any time. Hereby, the internal function of the traffic simulator can be used, e.g., to calculate the driving distance to the next traffic light.

To reduce the complexity of vehicle movements, they are interpreted by the network simulator as being linear for a time interval from one time step to the next time step. Fig. 3 depicts the different views of the movement of a vehicle on a curve in the traffic simulator (dotted line) and the network simulator (solid line). The traffic simulator calculates in each discrete time step the position of a vehicle with a given speed on the course precisely. In the network simulator the trajectories during the time steps are assumed to be straight-lined. Therefore, the vehicles' speeds are adapted to ensure all vehicles reach exactly the position as given by the traffic simulator. The resulting error in velocity is below 5 % even during turning at an intersection ($r_{curve} = 10\,\text{m}$) with a moderate speed of 36 km/h and can thus be neglected.

IV. Application

The feedback loop between the network and the traffic simulators as described in the last Section allows quantifying effects of VANET applications on road traffic. We chose a VANET application that offers a huge potential to reduce fuel consumption when vehicles pass traffic lights on the road, especially in urban areas.

This application is based on two ideas for saving fuel. First, vehicles that are headed towards a traffic light should slow down smoothly, if they are unable to reach the traffic light while it is green. Second, after stopping in front of a traffic light, a vehicle should turn off its engine, if the red phase will last for a certain time.

To realize a smooth and energy efficient slow down in front of traffic lights, two driving strategies are conceivable. On the one hand the driver can switch to neutral and coast with an idle fuel consumption of approximately on liter per hour ($C_{idle} \approx 1$ l/h), or he can take his foot from the gas pedal and thus cut-off the fuel feeding of the engine – leading to a fuel consumption of zero during the slow down. An experiment not shown here, has demonstrated that, when decelerating from 100 km/h down to 50 km/h, coasting saves more energy than fuel cut-off. But in the case of decelerating from 50 km/h to zero, the time and space needed for coasting gets very high; the resulting "idle" energy consumption is in that case higher than with fuel cut-off. Thus, in our algorithm we use the "fuel cut-off" strategy and assume a constant deceleration ($a_{fuelCutOff}$) for simplification. However C_idle defines the minimum fuel consumption if the engine is running.

The *Traffic Light Assistance Algorithm* as given in Fig. 4 shows the local variables (lines 1–6) and the implementation of the aforementioned ideas (lines 7–31) as executed on every equipped vehicle.

Lines 14–24 of the algorithm calculate the time needed to get to each traffic light that is located in the next 500 m. Thereby, we assume the vehicle is going on with the actual movement like follows: The velocity remains constant if the vehicle actually drives with constant velocity or decelerates (line 17). When the vehicle accelerates at the moment, we assume an ongoing acceleration (line 19); otherwise in front of a green traffic light that will get red soon, accelerating vehicles would slow down too early. For each traffic light that is red at the calculated time, the needed deceleration to slow down smoothly up to the traffic light is computed and the overall maximum deceleration is stored in dec (line 22).

After calculating the needed deceleration to stop in front of the next red traffic light (dec), we compare it to a threshold $a_{fuelCutOff}$. If dec is higher, fuel feeding would be cut-off resulting in a smooth slow-down. Please note, that the constraints given by the car following mobility model are not affected by the application, i.e., vehicles still brake hard to avoid collisions and stop in front of red lights.

The second idea of turning off the engine while waiting at a red light is implemented in lines 8–13. Thus, the engine is stopped, if the remaining waiting time exceeds a critical time

$t_{minEngineOff}$ // Shortest time for turning engine off
$a_{fuelCutOff}$ // deceleration when fuel is cut-off
v, a // Vehicle's current speed and acceleration
d_{tl} // Distance to traffic light
$tlStatus$ // State (red or green) of traffic light
t_{green}, t_{red} // Remaining time of green or red phase

loop
    if Engine is running and $v = 0$ **then**
        if next TL is red and $t_{red} > t_{minEngineOff}$ **then**
            STOPENGINE
            SLEEP(t_{red} s)
        end if
    end if

    $dec \leftarrow 0$
    for all Traffic Lights within 500 m **do**
        if $a \leq 0$ **then**
            $t_{arrival} \leftarrow d_{tl}/v$
        else
            $t_{arrival} \leftarrow \sqrt{(\frac{v}{a})^2 + 2\frac{d_{tl}}{v}} - \frac{v}{a}$
        end if
        if ($tlStatus$ is red in $t_{arrival}$) **then**
            $dec \leftarrow \text{MAX}(dec, \frac{v^2}{2d_{tl}}))$
        end if
    end for

    if $dec \geq -a_{fuelCutOff}$ **then**
        FUELCUTOFF
    else
        No intervention
    end if
    SLEEP(1 s)
end loop

Fig. 4. Algorithm: Traffic Light Assistance

that is based on the increased fuel consumption of the engine during start-up.

The proposed algorithm assumes that traffic lights get equipped with a network interface that enables them to send regular reports about future phase changes to vehicles that are equipped as well. We define the term *time horizon* as the time the traffic lights can predict their future phase changes. Based on the information that is sent by the traffic lights, each vehicle can decide on its own, if it needs to react accordingly.

The application is based on the data dissemination protocol *AutoCast* [11], [12] as transport layer in ns-2. AutoCast disseminates data units generated by the application code automatically in a specified area for a certain lifetime. Therefore, AutoCast uses a hybrid approach of probabilistic flooding in dense networks and store-and-forward in sparse networks, resulting in an efficient multi hop forwarding of data.

By using AutoCast, the implementation of the application code for a traffic light gets very simple. It checks every second, if its state changes within the time horizon and if necessary it generates a data unit containing the time of phase change, the new phase (green or red) and a dissemination area. The data unit's lifetime ensures, that later arriving vehicles receive the state information as well. The job of data transfer is done by AutoCast – the data units will arrive at the vehicle shortly after they come into communication reach. We refer the reader to [12] for an exhaustive description and evaluation of the AutoCast protocol.

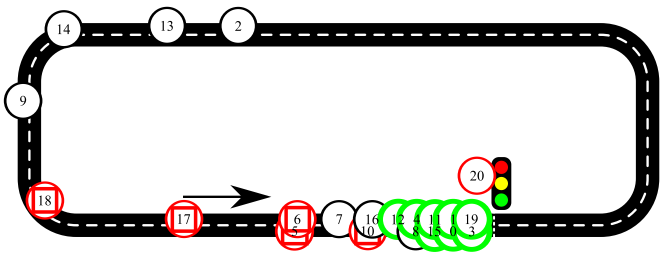


Fig. 5. Evaluated traffic scenario roadmap with vehicle overlay as reported by network simulation.

V. EVALUATION

We set up the simulation environment that uses TraCI for coupling ns-2 and Sumo as described in the preceding section. Additionally to the vehicles, we model traffic lights that are located in the traffic scenario as network nodes. Therefore, the network simulator requests at simulation starting time the count and positions of all traffic lights and insert the corresponding "traffic-light-nodes" into its simulation. During simulation runtime, each traffic-light-node requests periodically phase changes from the traffic simulator.

In the scope of vehicles, a corresponding network node gets instantiated just when a new vehicle starts driving. The vehicles request the traffic simulator for driving distances (d_{tl}) to particular traffic light that are situated on their route. Mobility primitives are used in the case a vehicle wants to slow down by cutting off the fuel feeding. Therefore, periodic *SetMaxSpeed*-commands are sent to the traffic simulator forcing the corresponding vehicle to slow down accordingly.

Hence, the coupling enabled by TraCI is not only used to gain vehicle traces and to close the feedback loop by allowing to influence the drivers' behavior, but also allows us to define the simulated scenario only once in the traffic simulator and let the network simulator request for scenario details.

Before presenting the simulation results we will discuss the chosen scenario and some more details necessary to understand the results presented in the last subsection.

A. Scenario

Simulating vehicular traffic of a large city is complex and the analysis and interpretation of the results is difficult, so we decided to set up a simple scenario where we can vary important parameters to model different traffic situations. We decided to choose a simple circuit as shown in Fig. 5 with one traffic light. The circumference is set to 400 m, based on a rough evaluation of average traffic light distances in German cities. In this setup, vehicles pass the traffic light several times within the simulation time of 20 min. The maximum speed of the vehicles is 50 km/h and the communication range is set to 250 m.

978-1-4244-3062-8/08/$25.00 ©2008 IEEE

For the parameters needed by the algorithm we choose a reasonable minimum time for stopping the engine $t_{minEngineOff} = 20\,\text{s}$ and estimate $a_{fuelCutOff} = -0.38\,\text{m/s}^2$ for the deceleration when cutting-off the fuel at a speed below 50 km/h.

In order to obtain different situations we modify

- the total number of vehicles on the circuit from 5 to 30,
- the phase duration of the traffic lights from 30 s to 70 s,
- the penetration rate from 0 to 100 %.

With a variation of phase duration we can create different situations for cars, so that they all need to wait for a certain period of time each time they approach the traffic lights or they have to stop only each second turn, etc. To minimize statistical uncertainty we repeat each simulation run 20 times with varying random seeds and average the results.

For evaluating the advantage of our VANET application, we use the vehicles' fuel consumption as metric. For the calculation of the fuel consumption we use a simple formula based on the kinematics including effects like friction, airflow and mass inertia as given in (1). We adjusted the formula to fit to the characteristics of a middle class car with a mass of $m = 1400\,\text{kg}$, a front surface of $A = 2\,\text{m}^2$ and a drag coefficient $c_w = 0.4$; additionally we use the physical constants for the density of air ($\rho = 1.29\,\text{kg/m}^3$), the coefficient of friction for car tires on asphalt ($\mu_r = 0.15$) and the gravitational acceleration ($g = 9.81\,\text{m/s}^2$). This set of parameters is used for all vehicles in the simulation.

$$F(t) = \frac{\rho}{2} c_w A v(t)^2 \; + mg\mu_r + ma(t) \tag{1}$$

We calculate the needed energy for moving a vehicle forward by multiplying the needed force ($F(t)$ from (1)) by the distance driven during a timestep ($v(t) * \delta t$). Then, the necessary fuel consumption can be calculated by considering the energy density of fuel $E_{fuel} = 8.9\,\text{kWh/l}$ and the average engine efficiency $\eta = 0.3$. In every timestep the minimum fuel consumption is set to $C_{idle} = 1\,\text{l/h}$ as lower bound, since this reflects the consumption of an idling engine. Finally, we get a vehicle's fuel consumption as sum over all timesteps of driving as can be seen in (2).

$$C = \sum_t \text{MAX}(C_{idle}, \frac{F(t) * v(t) * \delta t}{\eta * E_{fuel}}) \tag{2}$$

When our Traffic Light Assistance is running, we decrease a vehicle's fuel consumption every time the engine is stopped for the duration t_{stop} by the amount of $C_{idle} * (t_{stop} - t_{minEngineOff})$. We subtract $t_{minEngineOff}$ from t_{stop} in regard to the increased fuel consumption during engine start-up, i.e., if an engine is stopped for $t_{minEngineOff}$ s, no energy is saved.

Since the fuel consumption C is calculated in liter per hour, we get the conventional unit of liter per 100 km by dividing a vehicle's fuel consumption by the driven distance. For the evaluation we average the fuel consumptions over all vehicles.

B. Results

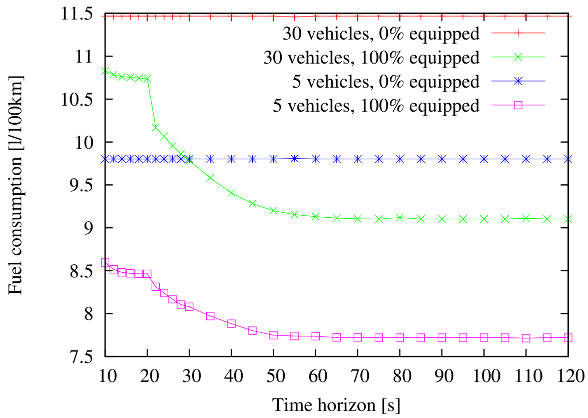
Fig. 5 illustrates the application by showing a snapshot of a running simulation with 20 vehicles (node 0-19) and a traffic light (node 20). The state of the nodes is marked by colors and symbols. The traffic light is actually red and a few vehicles are waiting with switched off engines as marked by green bold circles. Vehicles that are marked with a red square have cut-off their fuel feeding and thus slow down smoothly. The remaining vehicles with black thin circles are not affected from the application in this moment.

The first evaluation in Fig. 6a shows on the y-axis the resulting fuel consumptions when the time horizon of traffic lights varies as given on the x-axis. As already mentioned, the time horizon is defined as the time the traffic lights can predict their future phase changes. To point up the effect of the application, the straight lines show the fuel consumption for 0 % penetration rate for two different traffic densities and the falling lines depict the fuel consumption when all vehicles participate in the application. If the time horizon is below 20 s, engines would never be stopped, since the information how long the current red phase will last arrives to late to exceed the needed $t_{minEngineOff}$ s. As shown, an increasing time horizon of up to 60 s reduces the fuel consumption more and more, whereby a further increase does not reduce fuel consumption, since information about the phase changes after next are just not needed at that time. Therefore, we choose a time horizon of 60 s for the following simulations. Additionally, this evaluation shows that in general a higher traffic density leads to a higher fuel consumption due to more interaction between the vehicles. This finding is confirmed by the evaluation shown in Fig. 6b and experience from the real world.

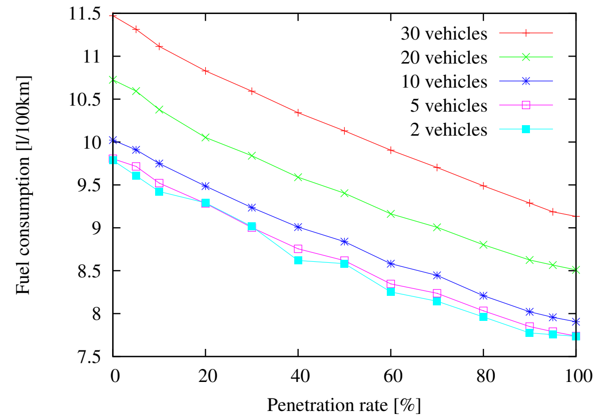
In the following, the parameter *penetration rate* will serve as the x-axis of the curves. Fig. 6b and 6c show on the y-axis the averaged fuel consumption for different traffic densities and green phase durations respectively. Here we can see, that in any case a positive effect results from our application, so it is sufficiently robust for changing scenarios. The variant inclines of the curves in Fig. 6c results from the fact, that during longer phases, the engine could be turned off longer as well. This impacts especially green phases of 30 s and 40 s.

Interestingly, the energy savings result from both strategies (fuel cut-off and stop engine). Their fraction depends on the scenario. So, in case of a low traffic density 70-75 % of energy saving results from slowing down with fuel cut-off and in case of a high traffic density 50-65 % of the total saving results from stopping the engine while waiting. Thus, both strategies contribute substantially to the energy saving.

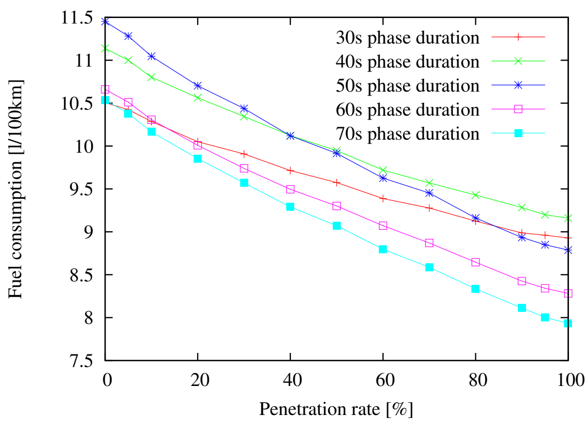
Finally, Fig. 6d shows the evaluation of the economic benefit by averaging all vehicles' fuel consumption in all simulated scenarios and the benefit for individual drivers, by averaging equipped and non-equipped vehicles separately. Here, we see that from an economic point of view, the energy saving increases with increasing penetration rate. From an individual point of view, even the first early adapters achieve nearly the



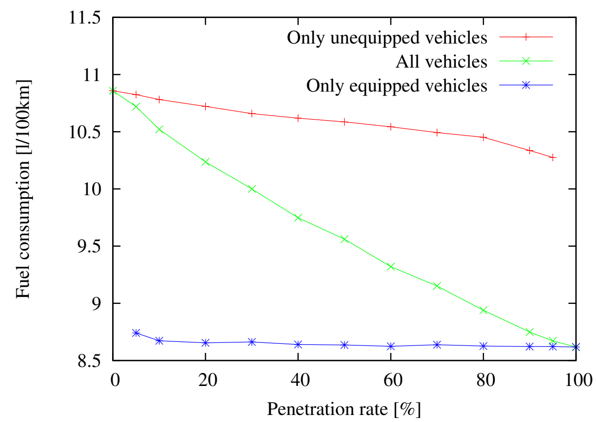
(a) Influence on fuel consumption for changing time horizon.



(b) Influence on fuel consumptions for changing traffic density (60 s time horizon, averaged green phase).



(c) Influence on fuel consumptions for changing green phase duration (60 s time horizon, averaged traffic density).



(d) Effects for equipped and unequipped vehicles (60 s time horizon, averaged traffic density and green phase).

Fig. 6. Evaluation results for the efficiency of the proposed VANET traffic light assistance application

optimum on saving – given that all traffic lights are equipped. Even non-equipped vehicles slightly benefit from the system, since they are influenced by coasting vehicles ahead.

The wireless communication performed by ns-2 turns out to be uncritical. The AutoCast protocol is very reliable and robust against communication failures; it consumes only a maximum of 3 kbit/s throughout all simulations which equals 0.3 % of the available bandwidth. Since traffic light information is only of interest to nearby vehicles, the overall system will scale well.

VI. Conclusion and Outlook

In this paper we have identified the potential of VANET applications and pinpointed the need for realistic simulation for new applications. We have demonstrated the useful coupling of a network and traffic simulator using TraCI. Our approach provides a flexible and efficient solution to build a feedback loop to influence vehicles. With this feedback loop we get the best of both worlds – realistic driving behavior and realistic network simulations – for a thorough evaluation of VANET applications. Our traffic light assistance application reveals the possibilities of TraCI and verifies the positive impact of the proposed traffic light assistance application on the fuel consumption of vehicles. The application would benefit from further communication directly between the vehicles, e.g., to better estimate the end of the line of cars that are already waiting at a traffic light. Furthermore, In the future we will apply feedback loops within AutoNomos – our decentralized traffic information system – to resolve traffic jams and generate warning messages in dangerous situations.

References

[1] D. Krajzewicz, M. Bonert, and P. Wagner, “The open source traffic simulation package sumo,” in *RoboCup 2006 Infrastructure Simulation Competition*, Bremen, Germany, 2006.

[2] K. Fall and K. Varadhan, *The ns Manual*, The VINT Project - a collaboration between researchers, 1989–2001, http://www.isi.edu/nsnam/ns/doc/.

[3] F. K. Karnadi, Z. H. Mo, and K. chan Lan, “Rapid generation of realistic mobility models for vanet,” in *Proceedings of the IEEE Wireless Communications and Networking Conference*, Mar. 2007, pp. 2506–2511. [Online]. Available: http://lens1.csie.ncku.edu.tw/MOVE/index.htm

[4] C. Lochert, M. Caliskan, B. Scheuermann, A. Barthels, A. Cervantes, and M. Mauve, “Multiple simulator interlinking environment for inter vehicle communication,” in *The Second ACM International Workshop on Vehicular Ad Hoc Networks (VANET 2005)*, Sep. 2005, pp. 87–88.

[5] S. Eichler, B. Ostermaier, C. Schroth, and T. Kosch, "Simulation of car-to-car messaging: Analyzing the impact on road traffic." in *13th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*. IEEE Computer Society, 2005, pp. 507–510.

[6] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, "Traci: An interface for coupling road traffic and network simulators," in *Proceedings of the 11th Communications and Networking Simulation Symposium (CNS'08)*, Ottawa, Canada, Apr. 2008.

[7] C. Sommer, Z. Yao, R. German, and F. Dressler, "Simulating the Influence of IVC on Road Traffic using Bidirectionally Coupled Simulators," in *27th IEEE Conference on Computer Communications: Mobile Networking for Vehicular Environments (MOVE 2008)*. Phoenix, AZ, USA: IEEE, April 2008.

[8] A. Richter, "Geschwindigkeitsvorgabe an lichtsignalanlagen," Ph.D. dissertation, Helmut-Schmidt-Universität / Universität der Bundeswehr, 2004. [Online]. Available: http://www.gevor-system.de/

[9] Z. Wang, L. Kulik, and K. Ramamohanarao, "Proactive traffic merging strategies for sensor-enabled cars," in *VANET '07: Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*. New York, NY, USA: ACM, 2007, pp. 39–48.

[10] S. Krauß, "Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics," Ph.D. dissertation, Universität zu Köln, Apr. 1998.

[11] A. Wegener, H. Hellbrück, S. Fischer, C. Schmidt, and S. Fekete, "Autocast: An adaptive data dissemination protocol for traffic information systems," in *Proceedings of the 66th IEEE Vehicular Technology Conference Fall 2007 (VTC2007-Fall)*, Baltimore, USA, Oct. 2007, pp. 1947–1951.

[12] H. Hellbrück, A. Wegener, and S. Fischer, "Autocast: A general-purpose data dissemination protocol and its application in vehicular networks," *Ad Hoc & Sensor Wireless Networks journal (AHWSN)*, to appear in 2008.