We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

13

Connectivity Prediction in Mobile Vehicular Environments Backed By Digital Maps

Robert Nagel and Stefan Morscher
Institute of Communication Networks, Technische Universität München
Germany

1. Introduction

As mobile ad-hoc networks gain momentum and are actively being deployed, providing users and customers with ubiquitous connectivity and novel applications, some challenges implied especially by the mobility of users have not yet been solved. Generally, it can be stated that modern applications impose higher requirements on the underlying communication solutions: more bandwidth, less packet loss, less delay and more reliability of services in terms of availability. These performance metrics are commonly termed *Quality of Service (QoS)*. Due to the variability of node locations in mobile networks, the experienced QoS is highly time-variant. We have discussed in Nagel (2010a) that the level of attained QoS ultimately results from a proper combination of connectivity, i.e., the communication relations in a network, the chosen (and usually invariant) medium access (MAC) protocol and the traffic that is injected into the network at the nodes. If a certain level QoS is desired in a mobile wireless network, at least one of these three properties has to be actively controlled.
We have demonstrated that through controlling the amount of traffic that is injected by the nodes, effective distributed mechanisms can be employed that are, given minimal information about nodes' connectivity, able to provide (and even guarantee) a certain level of QoS. These mechanisms, however, are based on the current connectivity of the network and are effective only at present time. Should an application require a certain amount of QoS over a larger period of time, additional provisions become necessary. Although it is possible to control connectivity in certain boundaries (for instance through power control or adaptive antennas) and at a certain cost, the fundamental physical causes of connectivity themselves (location, mobility, and wireless channel state) cannot be influenced by the application as they are dictated by the user's behavior and the environment. It is, however, possible to anticipate a network's future connectivity – at least for a certain time horizon – and to compute the resulting future QoS. Upon this information, applications, services, and routing protocols could be parameterized accordingly: as an example, if the future QoS of a connection using a certain route is predicted to fall below a necessary level due to a link break, the expected remaining time until the link actually breaks could be used to proactively find and set up a backup route that uses other, potentially more stable links. Also, if a connection was to be set up for a limited time, it may be very helpful to assess if the required QoS can actually be provided by the network for the desired duration before the connection is actually established. While other work mainly uses mobility prediction in cellular scenarios to estimate hand-over times, or to support ad-hoc routing in random-mobility ad hoc scenarios, this chapter

focuses on connectivity prediction in the special case of vehicular networks. Networked vehicular nodes can be assumed to adhere to certain rules that constitute drivers' basic behavior: they move along roads and try to avoid collisions with obstacles, such as buildings and other cars. Founded on the vehicular scenario constraint, we present an algorithm that predicts the location of vehicles based on their current state (position and velocity) and information from digital street maps obtained through the Open Street Map (OSM) project. A filter-based, self-adaptive velocity prediction algorithm is used to model the user-inflicted velocity changes. Using their current positions and the predicted velocities, possible future positions of cars on the street grid and their respective probabilities can be determined. Although the main focus of this chapter is on mobility prediction, we discuss an effective channel parameter estimation technique and propose to predict the network's future connectivity using an adaptive channel model.

It should be noted that the proposed position prediction mechanism does not completely exhaust all opportunities provided by the vehicular scenario. For instance, we assume that vehicles have no information about other vehicles' missions, i.e., the planned route through the road grid. Furthermore, we make no assumptions about other vehicles' capabilities (in terms of maximum acceleration and deceleration, yaw rate, etc.). Also, we do not consider environmental properties, such as weather, street and traffic conditions, etc. We will, however, point out and discuss the potential spots where these additional informations could be exploited to further augment the proposed algorithm.

In the following Section, we present an overview of current work. Subsequently, in Section 3 we formulate the problem mathematically and in Section 4, we describe our algorithm that predicts the future positions of vehicles according to their actual state and a complemental digital road map. In Section 5, we will discuss why the channel model presented in Equation 1 is not sufficient under all conditions and present methods that can adapt to the environment. In Section 6, we discuss some simulation results. Section 7 summarizes this chapter and gives an outlook on further work.

2. Related work

One possible way of predicting a network's future connectivity is to use a model that reflects the individual mobility properties of a node. Given the knowledge of the initial position velocity of a node, a future position could be projected by multiplying the velocity vector with the desired time interval. Obviously, this approach does not account for changes in the length and/or direction of the velocity vector. Several more sophisticated approaches have been suggested and today, mobility prediction has become a common research topic in wireless networks.

Due to the distinct characteristics of vehicular ad-hoc networks, especially the high speeds and restricted degree of freedom in the movement of vehicles, most of the work on prediction for ad hoc networks is too general and thus inappropriate for vehicular networks. Nevertheless, some approaches are discussed here because they give an overview of mobility prediction in general. Material specific to mobility prediction in vehicular networks is very rare and the topic is often neglected in works on vehicular ad hoc networks.

Kaaniche & Kamoun (2010) presents an approach for mobility prediction using neural networks. Although it is not specifically designed for vehicular networks it should perform better than other general approaches as it is independent of the underlying mobility model. A trajectory is calculated for multiple steps in the future using several past positions in quite a similar manner as the adapting FIR filter for velocity prediction presented in this work.

However, the approach does not use any map material and hence the predicted positions may lie far off the road and may thus be unrealistic. The approach using neural networks could be used for velocity prediction in the constellation presented in this works to substitute the FIR filter, however it is expected to perform in a very similar manner and the FIR filter seems less complex to implement.

A similar approach for mobility prediction using spatial contextual maps and Dempster-Shafer's theory for decision making is formulated in Samaan & Karmouch (2005). A framework is presented that allows prediction of the users mobility trajectory based on various bits of contextual information from e.g. user profile and map data. The approach is motivated by the fact that contextual information is becoming more common for adapting services towards the users needs and it uses the additional information in order to predict the users mobility. The concept seems feasible for e.g. cell phone users traveling on foot but does not seem appropriate for vehicular networks as the only contextual information possibly available and relevant to the future mobility is the chosen route to the destination. A complex theory to combine evidence into a prediction is not necessary in this case.

Huang et al. (2008) suggests a prediction algorithm based on fuzzy logic that aims at the prediction of a possible link break or a congested link which then triggers the construction of an alternate route. Similar to our algorithm, the prediction of a link break is based on the prediction of the future vehicle speed, the basis on which the predicted distance to the vehicle can be determined. This requires the generation of a fuzzy rule base that is then dynamically trained using Particle Swarm optimization (which in our approach is done using the adaptive filter for speed prediction). The authors use similar ideas in terms of the speed prediction but implements a fundamentally different concept. Furthermore, it is focussed on route break prediction and hence the performance of the isolated velocity prediction compared to our algorithm cannot be easily evaluated.

In Boukerche et al. (2009), the authors present some general thoughts on mobility prediction in vehicular networks and propose a simple prediction algorithm based on movement vectors in order to reduce the frequency of location beacons without introducing a higher mean error in respect to the positions used for routing packets. In Rezende et al. (2009), the same authors introduce the Network Neighbor Prediction protocol (NNP) that uses the results from their prior works to predict new routes that are going to be available in the near future and to calculate the lifetime of those routes that are currently in use. These works show in their simulation results that mobility prediction is a useful and necessary aspect in vehicular networks and should be researched in greater detail than it currently is.

Another approach, although developed in the context of a different problem, is described by Althoff et al. (2010). The authors compute the set of points that could be reached by vehicles within the prediction times, given the capabilities (minimum and maximum acceleration, yaw rate, etc). of the considered vehicles. The approach is computationally complex and requires a lot of contextual information.

Using the predicted position it should also be possible to predict the future connectivity to a certain extent using an appropriate channel model. In the context of vehicle-to-vehicle (V2V) communications, there is not yet a widely accepted channel model Paier et al. (2009). A common approach for characterizing a channel is to work out a theoretical channel model and then validate it against some appropriate measurements. Channel models are usually classified into stochastic and deterministic channel models, where deterministic channel models use ray tracing and similar techniques based on topological information about the environment in order to solve the the multi-path components (MPC) and derive a precise

channel characterization for a specific realization. Stochastic channel models, on the contrary, try to depict the statistics of the propagation channel in a more general sense that is not so much focussed on a particular situation. An intermittent approach is taken by geometry based channel models (as presented in Cheng et al. (2009)) that do use ray tracing; however, instead of using realistic modeling the calculations are based upon randomly placed objects.
In order to characterize a channel a number of parameters are used:

- The path loss exponent (PLE) α characterizes the average attenuation of the received signal.
- Large scale fading on the one hand refers to slow variations of received power due to shadowing by obstructing objects.
- Small scale fading on the other hand is caused by interference of different MPCs that result in fast fluctuations of the received power. Because these fluctuations are very hard to describe deterministically, they are usually described by means of statistics - most commonly by a Rayleigh distribution.
- In order to determine how much power is carried by the respective MPCs a power delay profile (PDP) is used. The spreading of the received pulse in the time division - often referred to as the channels delay dispersion - is best described in a statistical way by the root mean squared delay spread.
- Because MPCs travel on different paths they experience different Doppler shifts. The root mean square Doppler spread describes the resulting spectrum widening of the received pulse and thus the frequency dispersion.

A large amount of research has been dedicated to the wireless channel in cellular networks. However, looking at the specifics of a vehicular channel, especially in the V2V case it soon becomes clear that its characteristics differ significantly from those of a cellular channel. On the one hand, antennas of both sender and receiver are mounted close to the ground in V2V communications, where with cellular systems usually one of them is mounted high above. This tremendously influences the propagation path of the signals and thus the channel characteristics in terms of diffraction and reflection. On the other hand, communications between vehicles commonly use the 5.9 GHz band which behaves significantly different than the 700-2100 MHz signals used in cellular systems in terms of attenuation and diffraction. Most importantly though, sender and receiver are moving at relatively high speeds in V2V scenarios, which invalidates the assumption of stationarity of the channel characteristics that is commonplace in channel models of cellular systems. That refers not only to a changing impulse response but also to a change of its statistical properties (fading distribution, PDP and Doppler spectrum) Molisch et al. (2009). According to Maurer et al. (2004), Doppler shift and Doppler spread characterize the time-variant behavior of the V2V channel mostly due to movement of the communicating vehicles and the adjacent vehicles.
This section highlights and discusses some of the works into vehicular channel modeling in the context of connectivity prediction - a topic that has not yet received much attention in literature. In Matolak et al. (2006), the authors describe a statistical V2V channel model that is restricted to small scale fading. It uses a tapped delay line model, each tap representing a multi path components received with a certain delay. Each tap has an on/off switching process modeled by a first order Markov chain allowing for persistence parameterization. In general, taps with longer delays have less probability of being on due to their lower energy. Tap amplitudes are modeled using the Weibull distribution where different parameters are proposed for different taps, based on some measurements. The authors differentiate between

different scenarios, in some of which the Weibull parameters are "worse than Rayleigh" ($\beta < 2$), a phenomenon that is often called severe fading.

Maurer et al. present a geometry based IVC channel model in Maurer et al. (2004). They first try to model the dynamic road traffic and the environment adjacent to the road and then try to evaluate multi-path wave propagation through means of ray tracing. The road traffic model is based on the so called Wiedemann model and uses results from the authors previous works. As it seems very difficult to obtain real data with the necessary level of detail and the coverage, a stochastic model is utilized in order to place objects in the surroundings of the road. Different morphographic classes are defined for urban, suburban and highway scenarios that are assigned specific probabilities for different types of objects (trees, buildings, cars, bridges, traffic signs, etc.). Multi-path components are represented by rays, each of which can experience several propagation phenomena like diffraction or reflection. By calculating consecutive snapshots, a time-series of channel impulse responses can be obtained that classifies the channel for the current surrounding. The authors present measurements that validate the channel model with a standard deviation of less than 3 dB in both line-of-sight (LOS) and non-line-of-sight (NLOS) scenarios.

Paier et al. (2009) presents some measurements of V2V propagation in suburban driving conditions using GPS receivers. The authors on the one hand derive both a single slope and a dual slope path loss model from their results where the better dual slope model achieves deviations between 2.6 and 5.6 dB compared to the measured path loss. However, they find that received power is significantly less if no LOS propagation is possible. Fading on the other hand is modeled using a Nakagami distribution with variable parameters as already proposed in other works. While the distribution is Rician $\beta > 2$ as long as a LOS component is present, it turns out that fading can be "worse than Rayleigh" $\beta < 2$ once the LOS connection is lost intermittently at large distances between transmitter and receiver. Furthermore, the authors propose that the Doppler spread is dependent on the effective speed and the distance between transmitter and receiver. The dependance on distance is explained by the increasing number of scatterers at larger distances. Using this dependence, the authors present the speed-separation diagram that can help predict the expected Doppler spread and thus small scale fading characteristics at a certain distance.

In Molisch et al. (2009), the authors provide a survey on V2V channel models and measurements based on a variety of previous works on the subjects, some of which have already been discussed here. We recommend this paper as an introductory reading on the subject as it introduces important factors for channel characterization and includes a table that summarizes important parameters gathered from multiple measurement campaigns. Important aspects like environment characterization and antenna placements are also discussed that we omit here. One important result from the evaluated measurements is that at least path loss coefficients in V2V communication channels are rather similar to well-known cellular systems as long as a LOS connection is given. In terms of small-scale fading and Doppler spread, the results go alongside those presented in Paier et al. (2009). The authors finally conclude that the amount of comparable measurements carried out on V2V channels is too insignificant in order to allow the formulation of a channel model that resembles the real-world V2V channel and important aspects such as antenna placement and shadowing by adjacent vehicles have not yet been sufficiently explored.

Following this conclusion, an adequate prediction of channel quality seems challenging. Analogous to position prediction, an estimation of channel quality can be seen as a trade-off between computational complexity and prediction accuracy. An approach involving

ray-tracing similar to the one presented in Matolak et al. (2006) on the one hand produces rather adequate results if provided with the necessary extent of details concerning the surrounding environment (including moving and parked vehicles), building geometries, plants and road signs. However, it seems unrealistic and infeasible to supply an on-board connectivity prediction engine with this amount of knowledge. Measurements suggest a dual-slope model for the path loss exponent as a very simple approach. Small-scale fading is usually modeled using statistical models with strong dependency upon separation distance which limits the possibilities of a prediction to a qualitative worst case approximation. Paier et al. (2009) also identifies significant differences between LOS and NLOS cases in both path loss and fading statistics.
A sophisticated approach to predict the path loss exponent using a particle filter has been proposed in Rodas & Cascon (2010), based on a log-normal fading channel model in wireless sensor networks. Particles are initialized in a random state with their respective weights being iteratively updated to provide an estimation of the path loss exponent. Weak particles with low weights are periodically replaced to avoid degeneration. The filter is parameterized with the type of the fading distribution and its variance. The authors, too, show that the PLE changes significantly as soon as the LOS is lost.

3. Problem statement

In Nagel (2010b), we have outlined how QoS provisioning based on a network's connectivity can be attained. The basis for the computation is the connectivity matrix $\underset{\sim}{\mathbf{C}}$ that describes the communication relations between n networked nodes. Let $\chi(\underline{\mathbf{x}}_i, \underline{\mathbf{x}}_j)$ denote the channel function, taking as parameters the physical positions $\underline{\mathbf{x}}$ of two vehicles in the environment. A very basic channel function could then read:

$$\chi(\underline{\mathbf{x}}_i, \underline{\mathbf{x}}_j) = \begin{cases} 1 & \text{if } \left\| \underline{\mathbf{x}}_i - \underline{\mathbf{x}}_j \right\| \leq r \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

This means that two vehicles i and j are connected if they are located closer than the radio range r; if they are located further apart, they are not connected. The connectivity matrix $\underset{\sim}{\mathbf{C}}$ is then defined as:

$$\underset{\sim}{\mathbf{C}} = (c_{ij}),\ c_{ij} = \chi(\underline{\mathbf{x}}_i, \underline{\mathbf{x}}_j) \tag{2}$$

Every node i is allowed to inject *(source)* traffic amounting to s_i into the network. Multiplying the source vector $\underline{\mathbf{s}}$ with the connectivity matrix results in the load vector $\underline{\mathbf{l}}$:

$$\underline{\mathbf{l}} = \underset{\sim}{\mathbf{C}}\,(\underline{\mathbf{1}} + \underline{\mathbf{s}}) \tag{3}$$

We have shown that the QoS criterion is fulfilled if the injected traffic is dimensioned so that each entry in the load vector l_i does not exceed a certain pre-defined threshold. For more detail, especially on the distributed algorithm, the reader be referred to the original paper. The problem with this approach, however, is that $\underline{\mathbf{s}}|_{t_0}$ is only valid for the current connectivity matrix $\underset{\sim}{\mathbf{C}}|_{t_0}$. As it is desirable to fulfill the QoS criterion over a certain time Δt, we first need to predict the future physical positions of the vehicles, estimate the channel function and then deduce the prospective future connectivity matrix:

$$\underset{\sim}{\mathbf{C}}|_{t_0+\Delta t} = \Big(\underbrace{\chi|_{t_0+\Delta t}}_{\substack{\text{Channel} \\ \text{Estimation}}} \underbrace{\left(\underline{\mathbf{x}}_i|_{t_0+\Delta t}, \underline{\mathbf{x}}_j|_{t_0+\Delta t} \right)}_{\substack{\text{Mobility} \\ \text{Prediction}}} \Big) \tag{4}$$

After that, the future source vector can be computed (Equation 3) and a decision can be made whether the current demand can be satisfied under the future network conditions and consequently, adequate measures can be taken.

4. Mobility prediction

Generally, the spatial behavior of a vehicle is defined by two factors: On the one hand, speed and direction are controlled by the driver who adapts to the environment and the current situation. On the other hand, movement of a car is restricted to roads so the surrounding road topology is the major limiting factor. This is the key criterion that simplifies location prediction for vehicles compared to regular mobile users. Cars are usually not allowed to travel anywhere, they are bound to a relatively small portion of the world, the lanes. Combined with a small memory of past positions, the current velocity and direction of movement can be calculated. This further limits the amount of available future positions, as cars are usually not expected to u-turn spontaneously and velocity changes are bounded by the maximum deceleration and the maximum acceleration.

4.1 Concept

The prerequisite for the prediction is knowledge about a vehicle's current position, direction of movement and the surrounding road topology. The latter is provided by digital street maps (available, for instance, through the OpenStreetMap Project). All of these factors are very stable in terms of prediction. The destination or rather the mission of the car is assumed to be unknown to the algorithm, so at a crossroads basically all directions seem equally probable. The velocity of a car, however, is far less stable and predictable as it is directly controlled by the user and indirectly influenced by environmental factors such as traffic density, road signs and the weather. Especially abrupt speed changes are almost impossible to predict as they are often unexpected, even to the driver himself. The algorithm is sketched in Figure 1.

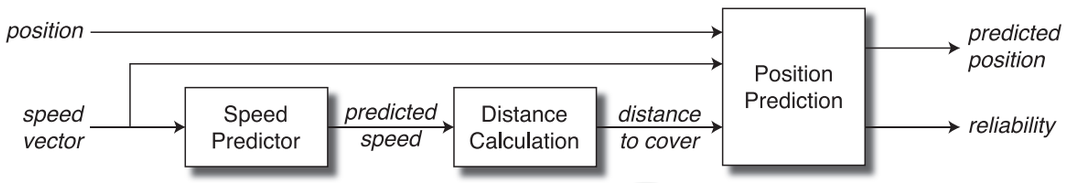


Fig. 1. Algorithm Outline

For speed prediction, we use a filter based approach that employs concepts of adaptive filters initially developed to adapt to varying channel conditions in wireless communications. Like the channel characteristics change depending on the environment, the speed change behavior of a car - or rather its driver - adapts to various environmental factors. This includes urban scenarios with steep velocity slopes and rural roads with fairly constant speeds. The character of the driver and the performance of the car also influence the prediction to a certain extent and are automatically taken into account by the adaptive filter. A self-adapting finite impulse response (FIR) filter approach based on a least-mean-squares (LMS) algorithm with relatively low depth seems ideal to adapt to both the personal behavior of a driver and the current situation. Using past and current velocities, an ideal weight vector for the past situation is calculated. Due to the low depth of the filter, the weight vector is rather unstable and consequently, it is combined with both the mean weight vector over the last iterations and a "boost" vector to improve reactivity at steep slopes. The resulting weight vector is then used

to predict the future velocity, which is in turn used to calculate the distance covered in the desired interval.
The distance to cover, together with the current position and direction of movement, forms the input for the position predictor that outputs the predicted future location of the vehicle. In some cases, multiple positions are possible, for instance due to a crossroads between the current and the future position. In that case, the position that seems most probable to the algorithm is used as an output; however, internally a list of all possible locations is generated. In many situations, predominantly with cars traveling in sparsely populated areas or on highways, the prediction is rather reliable. In urban areas prediction reliability is reduced by intersections where a sudden change of direction can occur and a certain amount of past predictions may be invalidated. To make applications aware of such differences, an additional output variable was added to resemble the estimated reliability of the output.

4.2 Input data

The algorithm requires a number of input data:

Position data: Obviously, the algorithm requires knowledge about the actual position of a vehicle and a timestamp. The position data used in the performance analysis has been downloaded from the "GPS Tracks" section of the OpenStreetMap online portal. Selected tracks were chosen that were provided by users around the globe and thus constitute a rather broad basis of real life data. Additionally, own traces have been used. The temporal resolution of the recorded tracks was or has been resampled to one second. A statement about the spatial resolution is not generally possible as different positioning hardware from various vendors has been used for the sample data. However, we shall assume a positioning accuracy of a few meters.

Map data: Also, the algorithm needs to be provided with map data of the area surrounding the actual position. This data, too, is provided by and downloaded from the open source OpenStreetMap project. It basically consists of an array of so-called nodes that are uniquely identified and reference a GPS position by latitude and longitude. A street is constructed by a list of subsequent nodes, forming a polyline that represents the shape of the street. Actual contiguous roads may be split apart, for instance if the name of a street changes or if two streets merge, on intersections etc.

Number of steps to predict: The major parameter influencing the algorithm. It is common in most parts of the algorithm and hence introduced in the high level diagram. Many parts of the algorithm also refer to it as n. Depending on the input data, the usual assumption is that one timestep equals one second. Most of the evaluations were done using a medium interval of prediction of 8 seconds - however results using different values are discussed in section 6.4.

4.3 Speed predictor

A car typically moves in different classes of environments: urban, suburban, peripheral and highway. Each of those has different characteristics concerning the speed change of a car. On a highway speed changes are rare but usually with rather steep slopes whereas in urban areas, the speed is hardly ever constant for more than a few seconds. This allows for two different approaches in implementing a speed prediction algorithm. On the one hand, specialized algorithms could be engineered for all of the above scenarios and another algorithm that determines the algorithm that is most appropriate in an actual situation. In typical situations one would expect such an approach to give very accurate results, but clearly there are many

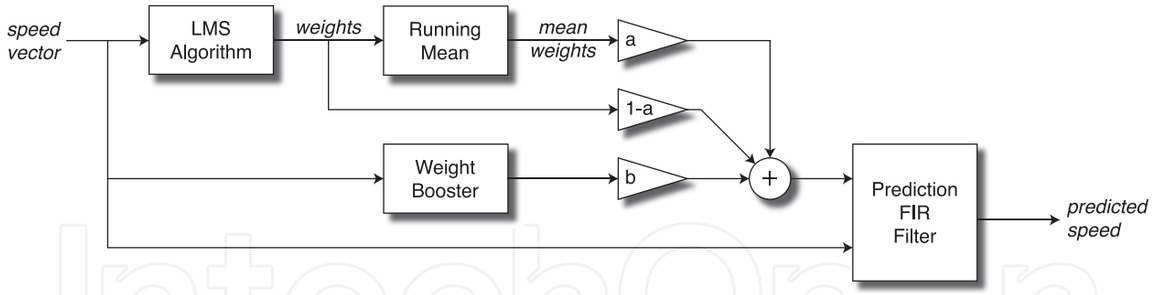


Fig. 2. Speed Predictor Structure

situations where none of the implementations will be adequate. Furthermore, this approach involves increased efforts in development because multiple algorithms need to be designed and there is a high number of factors influencing the situation that are hard to quantize.
On the other hand, it seems more appropriate to design an algorithm that automatically adapts to changing situations and as such can also adapt to factors like driver attitude and others mentioned above. This introduces some delay caused by the responsiveness of the adaption algorithm, but works also in an environment that cannot be properly classified into one of the above scenarios. In some situations, especially with quickly varying conditions, this may result in weaker performance than the approach discussed above, but the overall performance is expected to be better with less development efforts. A solution for this approach is discussed in the next sections.

4.3.1 Structure

The signal flow graph of the speed prediction is shown in Figure 2. The input variables are the current speed of the car and the number of steps n to predict. The only output is the predicted speed for the given time frame.

Prediction FIR Filter: The actual prediction is done in an FIR filter on the right hand side of the signal flow plan. It uses the current speed and a weight vector to predict the velocity from the last speed values. The length of the weight vector is given by the depth of the FIR filter - in the evaluations performed here a depth of 12 was used.

LMS algorithm: The most important building block. It is the adaptive part of the algorithm and calculates an ideal weight vector from its two input values - the current velocity forms the desired signal, a delayed version forms the input for the algorithm. The weight vector is adapted with a fixed step size in the direction of steepest descent in order to achieve the minimum square error and, at the same time, limit the dynamics of the weight vector.The weight vector is recalculated each time step, for more details see Benvenuto & Cherubini (2002); Guillemin et al. (1971).

Mean Weight: Because the weight vector generated by the *LMS algorithm* is very reactive to acceleration and deceleration processes, it is averaged by a running mean block that calculates the mean weight vector over the length of the situation. Both the mean and the LMS weight vector are combined into a slowed weight vector by multiplication with the parameter a or $1 - a$ respectively.

Weight Booster: The LMS algorithm adapts to new situations with a delay that is roughly its depth l plus the length of the prediction interval n, which equals the number of memory elements involved in the adaption. A change in velocity needs to pass through most of the memory elements before its effect becomes visible in the weight vector. For instance, for

parameter	example value	description
n	8	number of steps to predict
l	12	depth of FIR filters and dimension of weights vector
a	0.275	influence of the mean weights vector
b	2	influence of weight booster
c	0.2	boost limit
d	1	boost gain

Table 1. Speed Prediction Parameters with example values used during development

a car traveling in a city, the weight vector produces rather stable results while the car is traveling at a constant speed but it will react slowly to sharp braking or fast acceleration. The algorithm is designed to fix this problem by manipulating the weight vector in order to emphasize the most recent speed history elements to react more quickly to a spontaneous change in behavior: a length l base vector is multiplied with a scalar calculated from the slope of the velocity curve and is bounded above by the boost limit c. In order for the impact of the booster to remain present for a longer period, the generated "impulse" is broadened using a unity-weight FIR filter.

4.3.2 Parameters

The performance and precision of the speed predictor depends on some fundamental parameters that are summarized in Table 1. The given values are the result of some evaluations during the design phase based on few exemplary scenarios and should give a rough idea to start an implementation. However for a proper implementation a more thorough, numerical optimization is recommended but out of scope of this essay.
It is important to note that all of the below parameters influence the prediction in a way that usually makes adoptions to all parameters necessary if one parameter is changed. In many cases more than one possibility exists that can lead to a desired result for one scenario, but looking at multiple scenarios usually only one if any of the possibilities lead to an overall improvement of performance.

Number of steps to predict (n): This key parameter determines the number of steps to be predicted — $n = 8$ means the algorithm predicts the speed in 8 time steps. Obviously a higher value increases the prediction error, whereas lower values gives more precise predictions. The setting of this parameter is very important because its influence on the other parameters is tremendous, for instance a high value for n will on the one hand require a higher a and on the other hand require more influence of the weight booster, b. Also, this parameter is common with all components of the algorithm, so its influence has to be regarded globally. Different settings and their impact, especially on position prediction, are discussed in section 6.4.

Depth of FIR filters (l): The FIR filters' depth used in the speed predictor is a common value because all blocks share the weight vector. Also the parameter l is, unlike all other parameters mentioned here, a design time parameter that cannot be changed easily as it is hard-coded into the FIR filters and the constants. Nevertheless, its influence on the prediction should be discussed here.
For the fact that the depth of a filter resembles its amount of memory elements, higher values for l give more stable and less reactive prediction results. Changes in the situation need more time to propagate through the memory elements, hence it takes a longer time

to adapt to changes. Smaller values for l improve reaction time but also result in less stable and more fluctuating predictions that often overshoot at slight changes.

Influence of the Mean Weights Vector (a): The weight vector in the standard case (disregarding the weight booster) is combined from the current weight vector produced by the LMS algorithm and its running average. Setting a to the maximal appropriate value $a = 1$ produces a very stable weight vector but also removes the direct influence of the LMS algorithm to the weight vector and thus the reactivity. This is caused by the fact that in this model, the mean weight vector is never reset and thus provides an "all time average".

Influence of the Weight Booster (b): This parameter determines the overall influence of the weight booster. Higher values tend to produce overshoots as a trade-off to slow response to a change in situation if lower values are used. Generally all three values influencing the weight booster should be tuned according to the length of the prediction n. With high n, b should be increased because a faster reaction is necessary due to the latency of the LMS algorithm.

Boost Limit (c): The "boost" vector, or more precisely its scalar values, are influenced by the slope of the velocity curve. The "boost limit" defines an upper bound to those scalar values.

Boost Gain (d): This parameter multiplies the influence of the slope difference before the broadening and limiting of the pulse. Thus a higher value generates very quick increase once a steep slope is detected - in other words, it pushes the boost weight vector more quickly to the limit. Lower values produce a smoother response to steep velocity slopes.

4.4 Distance calculation

The speed predictor predicts the vehicle speed some time steps ahead. The position predictor in turn requires as an input the distance to cover in the next time steps to calculate the future position. The most precise approach is to predict a velocity value for each time step in the prediction period and sum up the difference. Because this requires a set of n speed predictors which increases the computational efforts by n, a simpler approach is chosen in this implementation. Our algorithm uses the current speed and the predicted speed and calculates a linear approximation between the two. The distance to cover s is then the area under the speed curve for a duration of t (n time steps):

$$s = t \cdot \left[v_{current} + 0.5 \cdot \left(v_{pred} - v_{current} \right) \right]$$

4.5 Position predictor

The position predictor uses the current position and direction of movement, digital map data as well as the predicted distance to cover as inputs and outputs a predicted position and its reliability. It is invoked once per time step and tries to first find the current road segment of the vehicle, then determines a number of possible prediction paths and finally chooses the most probable path and returns its end point.

4.5.1 Determine current road segment

All known nearby road segments (taken from the digital map) are evaluated for the distance of the current position to the closest point on the respective road segment. Three criteria must be met in order for a road segment to be chosen:

1. The distance to the closest point is smaller than a threshold.
2. The absolute value of the difference between the direction of movement and the road segment's direction is not larger than $\pi/2$ because vehicles usually do not move perpendicular to streets.
3. It is the closest road segment satisfying both criteria 1 and 2.

In case no road segment is found that fulfills all of the above criteria, the algorithm returns the current position as prediction result with an estimated accuracy of 0%. Possible causes range from wrong GPS positions during the initialization phase of the GPS device and inexact map material to driving or parking on streets or private property that is not (yet) included in the map material.

4.5.2 Determine possible paths

First, the remaining distance from the current position to the respective end point of the road segment s_r is calculated. If the road segment's end point is further away than the distance to cover ($s_r \geq s$), the predicted position will be located between the current position and the road segment's end point. Therefore, the predicted position is determined along the road segment's polyline towards the end point, covering the given distance s.
In the case that the remaining distance s_r is smaller than the distance to cover ($s_r < s$), the predicted position is moved to the road segment's end point and that distance is subtracted from the remaining distance. Subsequently, the next road segment of the prediction path is determined. If the mission of the vehicle is known in advance, the next road segment is chosen according to that mission. Otherwise, in order to find the next road segment, the number of possibilities is determined from the digital map: at a junction, all connected street segments are considered possible candidates. The current road segment, however, is not considered as an alternative — in other words, the vehicle is not expected to u-turn. Three cases exist:

(a) No candidates exist, so the current road segment ends in a node that has no other road segments referenced. In this case the relative probability of the current path is decreased in the relation to the amount of distance already covered.

(b) One candidate means there is no choice and the vehicle is moving along a road without an intersection at the current node. Determining the next road segment and updating the path accordingly is trivial.

(c) Multiple candidates are available, so the predicted path hits some kind of intersection. Hence the process of determining the next road segment becomes a bit more complex: Initially, all candidates must be assumed to be equally probable.

The procedure is repeated recursively until all distance s is covered and all possible paths of length s have been determined (effectively yielding a tree of possible road segments, with leaves at all possible future locations).

4.5.3 Pick best path

It may be desirable for an application that the prediction comprises all possible future realizations. However, if the prediction routine should return the future position along only one predicted path, the best of the alternative paths found must be chosen. If the mission of an observed vehicle is unknown to the algorithm, it must more or less issue a guess as to what option the driver of a car will go for. The range of alternatives is narrowed down in three steps:

(a) Estimated probability: In the current implementation of the algorithm, this first step will only remove the paths that end in a *dead end* and hence have a reduced relative probability. All other paths are considered equally probable and hence cannot be classified by their probability. For instance, the car hits an intersection with three alternatives, one of which being a dead end street. The dead end would be removed from the candidates, whilst the other two possibilities are equally probable.

(b) Way Changes: The number of street changes is the primary decision criterion for the algorithm. It is assumed that in case multiple paths exist, the driver stays on the current street. Hence the path with the least number of street changes is favored for the prediction. It is furthermore assumed that if it is necessary to change the road at some stage in all paths, the driver still stays on the current road as long as possible.

(c) Direction Difference: Should the way change criteria be unable to choose one candidate, the total difference of direction along the path is considered. Assuming the driver to be lazy, the path encountering the least change in direction is chosen to be the best path.

Clearly, criteria (b) and (c) do not increase the probability of a certain path. These are merely decision criteria in order to choose one path from multiple options. Choosing a random path statistically produces the same error, but has a severe disadvantage in terms of continuity: as the algorithm is executed each time step, it should return consistent values from one step to the next; when using a random selection, it is most likely that the algorithm will return a completely different position each time it is invoked. From a statistical point of view, this does not change much but for another program or algorithm that is based on the results of the prediction it may very well change things depending on the application. For the very same reason, it is very important for the algorithm to return the estimated probability of a given prediction because another program can then classify the prediction accordingly.

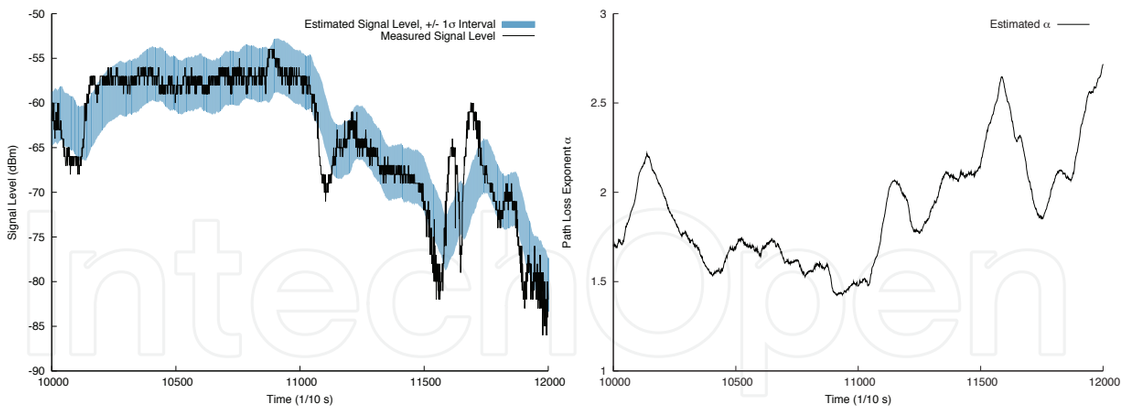
5. Channel prediction

It is clear from Equation 4 that the network's future connectivity does not only depend on the future vehicles' positions. An adequate channel model has to be chosen and constantly updated to reflect the changing radio environment. In Equation 1, we have shown an exemplary simple channel function, the *disc model*. We shall call two nodes i and j connected if the path loss $\beta(d)$, a function of the distance between the two nodes, does not exceed a certain threshold β_t:

$$\beta_t \overset{!}{>} \beta(d) = 20 \log_{10} \frac{4\pi d_0}{\lambda} + 10\alpha \log_{10} \frac{d}{d_0} \tag{5}$$

The path loss consists of two components: a constant addend that reflects the loss related to the wavelength of the signal and a distance-dependent term that represents the propagation of the radio wave through space and the resulting diminishment of power due to the growth of the wave sphere's surface. Due to various propagation effects, the path loss exponent (PLE) α is variable, to reflect various environments' radio properties (and usually ranges between 2 and 3). To account for reflections, scattering and shadowing, an additional stochastic variable β_s is introduced that is log-normally distributed with zero mean and variance σ^2:

$$\underbrace{\beta(d)}_{\text{measured}} = C + \underbrace{10\alpha}_{\text{unknown}} \log_{10} \underbrace{\frac{d}{d_0}}_{\text{known}} + \underbrace{\beta_s}_{\text{unknown}} \tag{6}$$

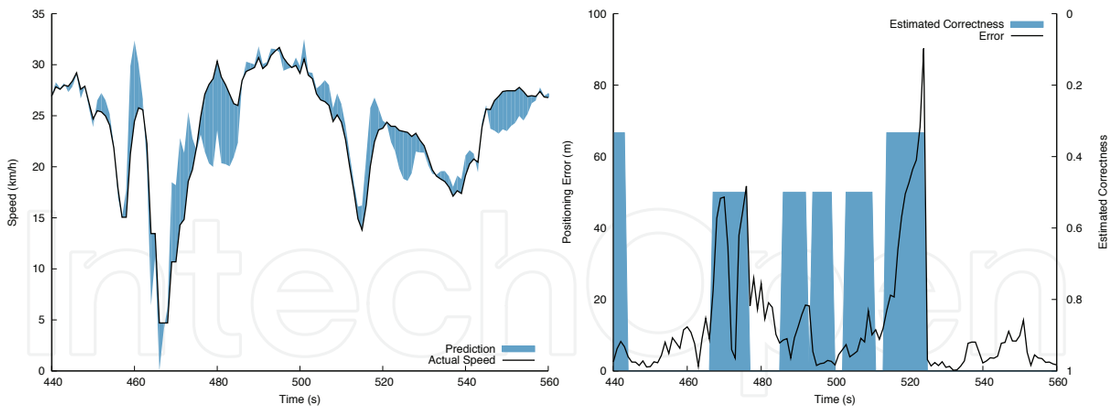


(a) (b)

Fig. 3. Parameter estimation using particle filter: measured and estimated signal level, estimated PLE

Through constant exchange of position messages, two nodes can determine their distance d and at the same time measure the path loss $\beta(d)$ between them (terms marked as "known"). Assuming that β_s is log-normally distributed and knowing the order of magnitude of the variance, we suggest to use a particle filter for online estimation of the PLE α and subsequent prediction, analogous to the work presented in Rodas & Cascon (2010). To study the vehicular channel, we have recorded and evaluated several hours of measurements. Figure 3(a) shows the measured path loss (solid line) and the path loss computed using the estimated PLE plus β_s's 68% (one standard deviation) confidence interval (filled curve). The standard deviation of the measured signal level was estimated around 3 dB, the system constant C was -42 dB and the duration of the displayed dataset is 20 seconds. The estimated PLE is shown in Figure 3(b). For connectivity prediction, we propose to employ the same concepts as used for position prediction to at least estimate the trend of the PLE. Furthermore, as we have discussed in the section on related work, it is very important to distinguish between LOS and NLOS conditions. In Nagel & Eichler (2008), we have introduced a method for V2V channel simulation in environments that include objects that possibly obstruct a direct LOS path and have discussed how a dual-slope channel model can be implemented to account for these objects. Consequently, we propose to complement the path loss formula in Equation 6 accordingly and incorporate the information on buildings and other obstacles from the digital map material (that is already used in the position prediction) to determine if the path between the predicted future vehicle positions is LOS or not. The propagation breakpoint derived from the map should then be accounted for in the PLE estimation. In simulations, we have realized quite accurate channel parameter estimations using a particle filter that accounts for LOS and NLOS conditions, based on information about surrounding radio obstacles.

It is, however, clear that the effects of large-scale fading have a large impact on the future connectivity but are hard to account for. Due to positioning errors, it is virtually impossible to predict small-scale fading effects. When evaluating the connectivity matrix computed from the predicted positions and the predicted channel, additional information about the reliability of the prediction should be provided and accounted for.



(a) Velocity prediction (b) Prediction error

Fig. 4. City Scenario

6. Results

Three representative scenarios were chosen for the performance evaluation of the developed algorithm. These scenarios are based on GPS tracks downloaded from the OSM portal that were selected to provide maximum diversity in the results presented below: the chosen tracks were recorded in a city as well as in suburban and highway surroundings. We have evaluated the three scenarios concerning the accuracy of both the speed prediction and the resulting predicted position under the three different environmental settings.

6.1 City scenario

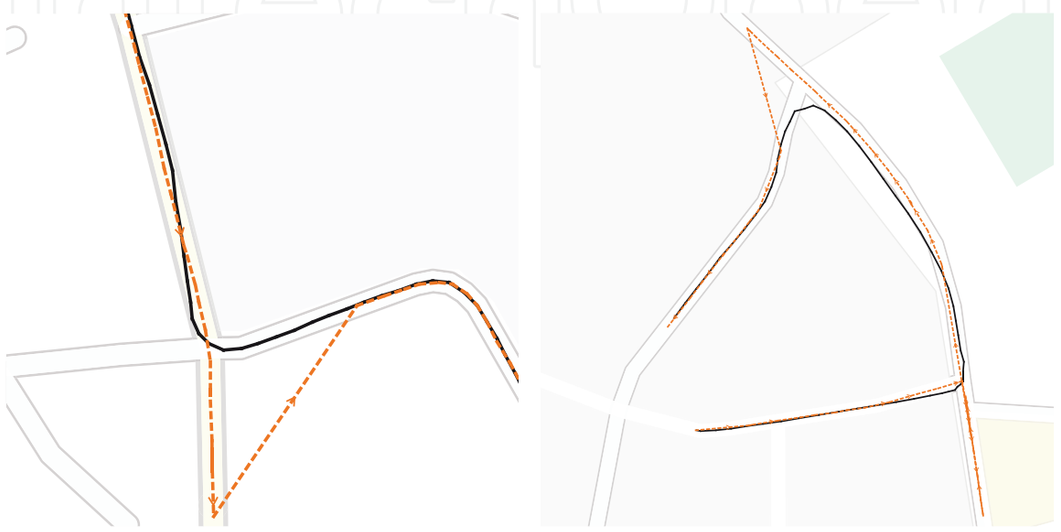
The first data set represents a typical city scenario. It has been recorded in the german city of Herne in the Ruhr area, with speeds of up to 50 kilometers per hour and a total length of about 20 minutes.

Figure 4(a) shows a section of the actual vehicle speed (solid line), the area of the filled curve reflects the velocity prediction error where the upper or lower edge of the area marks the predicted speed. For easy comparison, the predicted values are shifted by 8 seconds, so that the real value and the value that has been predicted for that instant are matched in time. The results show that especially at steep slopes, the algorithm overshoots significantly and predicts too low or too high velocity values. This could be tuned using the parameters of the speed prediction in order to achieve better performance in the particular scenario, but the impact on other scenarios is hard to estimate and thus requires significant research efforts.

Figure 8(a) shows the distribution function of the position prediction error in meters (upper blue line). The mean error is 13.4 meters, the median amounts to 7.5 meters. Figure 4(b) shows a section of the prediction error over time; the filled curve represents the estimated probability (correctness) of the prediction. Note that the second Y axis has been reversed for better readability.

As explained in section 4.5, the estimated probability generally equals 1 if the position predictor identifies only one possible path for the vehicle, based on the map data. In the presented case that the mission or route of the car are unknown to the algorithm, the choice of the path used for prediction is arbitrary once it encounters p multiple possible paths. Hence the estimated probability drops to $1/p$. This, in turn, means that if a large error occurs while the estimated probability is 1, an error in the position predictor or the map material should

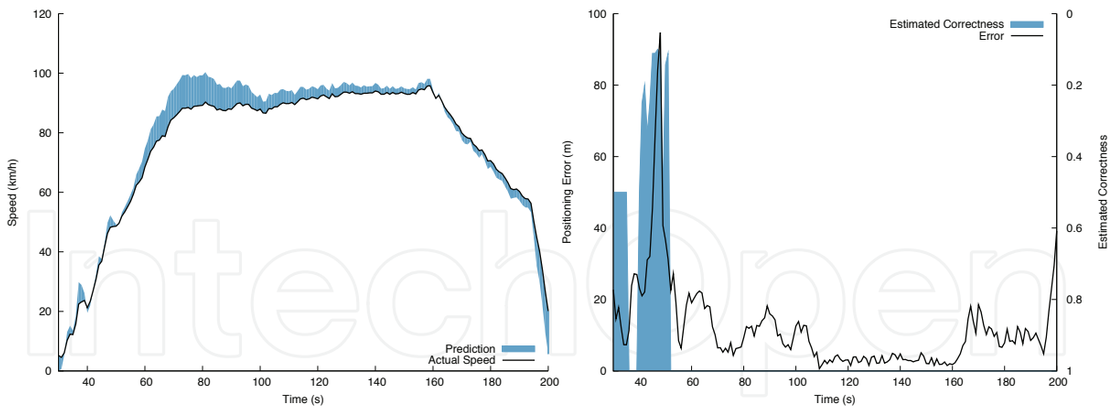
be assumed, while high prediction errors with low estimated probability are most likely to be produced by the fact that the mission of the car is unclear.
Should the position predictor be either unable to find the current segment or to find any path to continue, the estimated prediction probability drops to zero. This is the case at the beginning of the city scenario and is the result of pulling out of a private parking area in a sort of backyard. As there is no map material covering that area, the position predictor cannot give useful predictions based on the fact that it is unclear on which road the vehicle is driving. In such situations, the position predictor simply returns the current position.



(a) Intersection (time step 500) (b) T-Intersection (time step 200)

Fig. 5. Scenario 1 - Prediction Behaviour at Crossroads

To get an idea of the nature of an error, it is helpful to visualize the real and the predicted path as shown in Figure 5. The actual path is shown as a black solid line while the predicted path is shown as a dashed orange line.
Around time step 520, Figure 5(a) shows a typical situation in which the prediction algorithm encounters a crossroads. For the reasons explained in Section 4.5, the algorithm always prefers to choose the path that continues on the current road. This can be seen in the Figure where the predicted green dots continue straight on, while the real, blue trace turns onto the intersecting road. Once the car is on the new road, the prediction adapts to the new situation and continues its prediction along the new road. This can also be seen in Figure 4(b), where the error peaks due to the large discrepancy between the real and the predicted position. At the same time, the estimated probability drops to 0.33, due to the fact that the algorithm recognizes three alternative paths at the intersection. Figure 5(b) shows a similar example around time step 200 as the car moves towards a T-crossing. The algorithm chooses the path with the lowest total change in angle, which in this case is the wrong choice. This, again, results in a drop of the estimated probability and a peaking error.
As can be seen from the examples above, the high error in urban scenarios is widely based on the fact that usually many intersections lie along the path and high prediction errors are introduced if the algorithm chooses the wrong path. We have argued that this fact can be significantly improved if the cars mission is known to the position predictor and, consequently, the correct path can be used for the prediction.



(a) Velocity prediction (b) Prediction error

Fig. 6. Suburban Scenario

This also implies that the mean prediction error is not a very good metric in order to assess the precision of the prediction. The algorithm may predict the cars position with an error of less that five meters in cases where there are no intersections, but at each intersection an error of up to 50 meters is possible that will greatly influence the mean error. A more suitable metric to identify the amount of such situations in the scenario is the discrepancy between the mean error and the weighted mean error that includes the estimated probability. The weighted mean error is simply the mean over the prediction error, multiplied with the estimated probability. Because the estimated probability drops at intersections, a prediction error in this situation is weighted less than an error occurring along a straight, intersection-free road. The weighted mean error in this situation amounts to about 7.4 meters, which is significantly less than the mean error of 13.4 meters and therefore shows that the prediction quality could be greatly improved with knowledge of the cars future route.

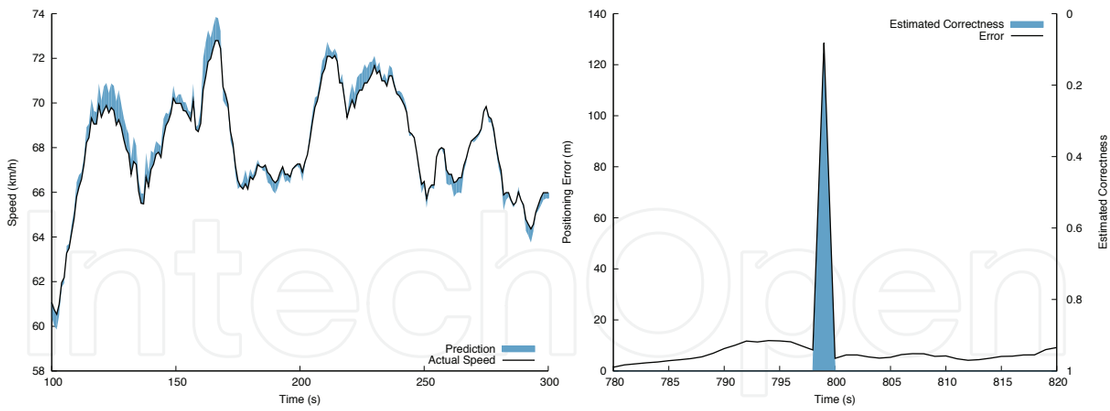
6.2 Suburban scenario

The second scenario covers the suburban area of Wiener Neustadt in Austria. The driver first hits the B17 road and afterwards enters the suburban area where the car is parked at a shopping center.

The section of the velocity graph in Figure 6(a) shows a short drive to the highway like state road. The speed prediction is rather stable whilst traveling at constant speeds between second 50 and second 200. This is accompanied by a rather small prediction error as shown in Figure 6(b). A peak in the prediction error at second 40 occurs due to a wrong choice of the next segment - again based on the fact that the cars mission is unknown. The fact that the algorithm had the choice of multiple directions is visualized by a significant drop of the estimated correctness curve.

Due to a very sharp deceleration around second 200, Figure 6(b) shows another peak without a drop in probability for the fact that no other possible directions are identified. The reason for the braking is unclear, an explanation cannot be found in the map material nor the satellite image of the area. Figure 6(b) also shows a few more peaks based on decisions for the wrong directions and braking actions.

The distribution function shown in Figure 8(b) (upper blue line) is slightly flatter than the one from scenario 1. The median error (about 10 meters) is slightly higher than in the city,



(a) Velocity prediction (b) Prediction error

Fig. 7. Highway Scenario

probably caused by the fact that either map material or the GPS device used to record the track are less precise than in the city scenario. The 90% percentile is significantly smaller (22 meters compared to 34 meters), which supports the before assumption and leads to the conclusion that the overall position prediction is better in the suburban scenario. The mean error of 13.5 meters, however, is almost identical to the city scenario.

6.3 Highway scenario

The third scenario covers a ride on a highway near the English town of Cambridge during rush hour traffic, which explains the unsteady velocity curve shown in Figure 7(a). Again, a representative section has been chosen for presentation.

The prediction error plotted in Figure 7(b) shows constantly low prediction errors. From the distribution function shown in Figure 8(c) (upper blue line) reveals a very steep slope, which means excellent performance of the algorithm as more than 95% of all errors are below 10 meters. The mean error amounts to only 4.6 meters and the median is 3.8 meters.

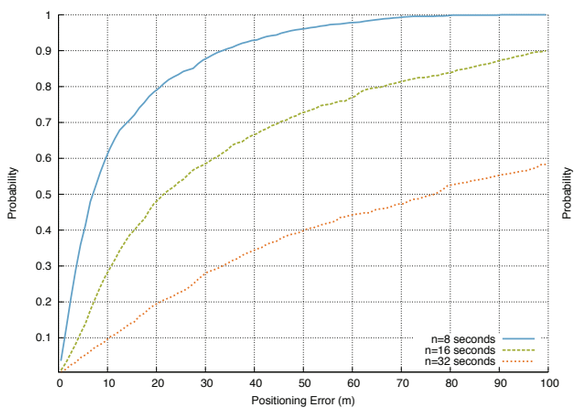
One notable peak of the error, accompanied by a drop in the estimated probability occurs around time step 800 (see Figure 7(b)). At the instant when the car is crossing a highway bridge the algorithm wrongly chooses the current road segment to be part of the road below the bridge that shares a node with the highway. Consequently, the algorithm chooses the wrong road segment to continue prediction. This error is caused by an unfortunate combination of a small measuring error of the GPS device and an imprecision of the map material, where the highway shares a node with the intersecting road (although they are on different levels).

Another error source stems from the fact that in the used map material, a road is represented by a polyline, and all vehicles are assumed to be positioned on this line. In reality, highways consists of a number of parallel lanes that have a certain lateral displacement. Although a suitable representation of road lanes has been suggested, the data available today does not yet represent different lanes. We expect, however, that the error would be decreased if this information could be accounted for in the prediction.

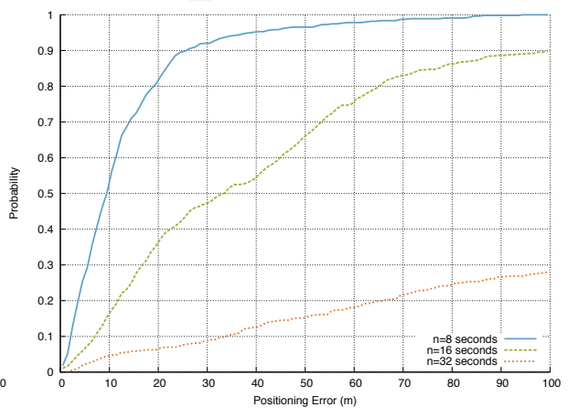
6.4 Influence of prediction length

The results that we have discussed above were obtained through prediction with period lengths of eight seconds. This sections evaluates the same scenarios with longer prediction lengths of 16 and 32 seconds. The resulting position error distribution functions are shown in Figure 8.

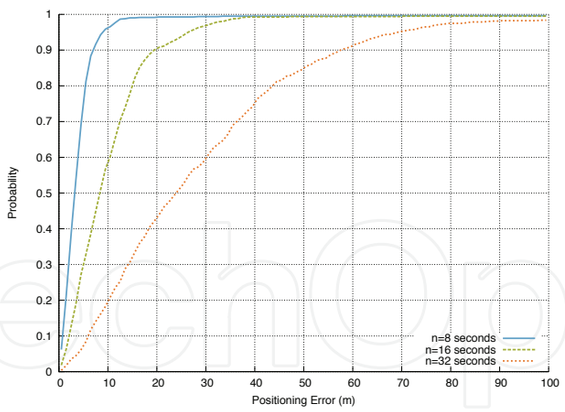
The steepness of the distribution functions decrease as the prediction length increases, due to less prediction accuracy across a longer period of time, i.e., the tendency to produce greater errors. The maximum error also increases dramatically, because in case that a wrong path is chosen, the prediction continues along the wrong path for a much longer time before it is corrected as the car turns the other way. The mean error is also shifted towards higher values with increasing prediction intervals.



(a) Scenario 1 - City



(b) Scenario 2 - Suburban



(c) Scenario 3 - Highway

Fig. 8. Prediction Error Histogram - Different Prediction Lengths

In the city and suburban scenarios, the prediction is already significantly less reliable with $n = 16$ time steps (i.e., seconds) as a prediction interval. It is rendered basically useless with a prediction interval of $n = 32$ and the majority of errors are out of scale of the histogram. The highway scenario, however, behaves much more stable as the prediction interval is increased and still returns useful results using a prediction horizon of 32 seconds. To a large extent, this is based on the relatively stable velocities and absent alternative paths along the way.

6.5 Path loss estimation error

To determine the future connectivity of a network, a node has to predict the positions of all relevant vehicles (usually the one- or two-hop radio neighborhood) and determine the resulting path loss to decide whether it will be connected to this vehicle or not. The first error source of the path loss estimation is, of course, the error induced by inaccurate position prediction. Fortunately, this error term strongly depends on the distance d between the two involved vehicles. Let us assume that the estimating vehicle has perfect ego positioning and position prediction and let Δd denote the maximum positioning error. The resulting absolute estimation error range is then:

$$\begin{aligned}\Delta\beta(d,\Delta d) &= \beta(d+\Delta d) - \beta(d-\Delta d)\\ &= 10\alpha\log_{10}\left(\frac{d+\Delta d}{d-\Delta d}\right)\end{aligned} \tag{7}$$

The second influence on path loss estimation is the accuracy of the predicted path loss exponent. Let $\Delta\alpha$ denote the PLE's maximum estimation error. The resulting absolute path loss estimation error is:

$$\begin{aligned}\Delta\beta(d,\Delta\alpha) &= \beta(d,\alpha+\Delta\alpha) - \beta(d,\alpha-\Delta\alpha)\\ &= 20\Delta\alpha\log_{10}\left(\frac{d}{d_0}\right)\end{aligned} \tag{8}$$

Clearly, there exists a strong negative correlation between the path loss estimation error and the distance to the tracked vehicle, d. With increasing d, the implications of prediction errors become less important with respect to the path loss. The situation is contrary regarding the PLE estimation: because the relation is linear, a large path loss estimation error results if the distance d to the tracked vehicle increases. The problem is that Δd and $\Delta\alpha$ are not known at runtime; therefore, we suggest to keep the prediction results and constantly compare them against the predictions to obtain a statistic of the errors. This information should consequently be used to determine a prediction's reliability.

7. Conclusions and outlook

In this chapter, we have presented an algorithm for the self-adaptive prediction of mobile nodes' future positions. The algorithm is targeted at vehicular applications with nodes that move along the road grid, of which a digital map is available at runtime. We have introduced the necessary building blocks along with their parameterization, discussed some performance studies and pointed out individual strengths and shortcomings. Three exemplary scenarios have been studied: city, suburban, and highway. Given a prediction interval of eight seconds, the algorithm performed well in all of the scenarios, resulting in a mean prediction error of only about 14 meters. On a highway, the mean error is less than 5 meters. As the prediction interval increases, the performance of the algorithm degrades significantly in the city and suburban scenario. On a highway, however, the mean error is around 30 meters for an interval of 32 seconds which may still be acceptable, depending on the application.

We have already argued in the discussion that the position prediction accuracy in the city and suburban scenario is mainly degraded due to an incorrect path selection as the considered vehicle approaches an intersection. In our studies, the future path has been selected randomly from the set of possible paths. The necessary assumptions have been

explained in Section 4.5.2. If the information is available, we strongly propose to consider vehicles' missions for path prediction. Considering the city scenario, if only those prediction errors are evaluated for which the path selection is correct (i.e., the predictor estimates the correctness as 1), the mean error can be decreased to about 8 meters, the median to about 5 meters. Taking the mission into account, these extremely low errors seem feasible.

The computation of the distance to cover is currently calculated using the area under a linear graph between the current velocity and the predicted velocity, thus assuming a linear acceleration. Some thoughts should be given to a substitution of this simple approximation with a more sophisticated implementation. One idea that is rather complex in terms of computational efforts is to use n velocity predictors to predict a velocity for each time instant and thus removing the interpolation. Another approach could be to model the acceleration and deceleration behavior of a typical driver. If the vehicle is autonomous, reproducing the design of the longitudinal controller could increase the prediction performance.

Velocity prediction, too, offers some optimizations opportunities. The key measure necessary here is a numerical optimization of the parameters a, b and c mentioned in Table 1 over a large number of scenarios of adequate length. Appropriate parameter sets could be computed beforehand (and even optimized online) and information about the current driving situation could be used to select the most suitable set. This selection, in turn, could be used to provide other applications with valuable information about the current environment. It is expected that an adoption of these parameters will lead to a somewhat significant improvement of the speed prediction. An urban scenario requires much quicker reactions to speed changes and thus needs more contributions and stronger influence of the weight booster than a highway scenario. The highway scenario, in turn, profits from a more stable prediction based to a large extent on the mean weight vector and requires virtually no influence of the weight booster.

Longer-term prediction of the wireless channel still imposes the largest problem when it comes to evaluate the future connectivity from predicted positions. Further work is necessary to evaluate the dynamics of the radio channel and design an appropriate predictor. We propose to include further information about the environment (see above) in order to distinguish between different surroundings and consequently adjust the appropriate channel parameters (such as the variance of the large-scale fading). An interesting idea in this context is to share and aggregate knowledge of the communication channel obtained from measurements between nearby vehicles. Another situation that requires attention is the channel prediction for vehicles that are actually outside of a node's communication range and channel parameter estimation is obviously not possible. In this case, the channel has to be estimated from measurements conducted with and from nearby vehicles.

8. References

Althoff, M., Stursberg, O. & Buss, M. (2010). Computing reachable sets of hybrid systems using a combination of zonotopes and polytopes, *Nonlinear Analysis: Hybrid Systems* 4(2): 233 – 249.

Benvenuto, N. & Cherubini, G. (2002). *Algorithms for Communications Systems and Their Applications*, Wiley, New York.

Boukerche, A., Rezende, C. & Pazzi, R. (2009). Improving neighbor localization in vehicular ad hoc networks to avoid overhead from periodic messages, pp. 1 –6.

Cheng, L., Bai, F. & Stancil, D. (2009). A new geometrical channel model for vehicle-to-vehicle communications, pp. 1 –4.

Guillemin, E. A., Kalman, R. E., DeClaris, N. & Andersen, J. (1971). *Aspects of network and system theory*, Holt Rinehart and Winston, New York.

Huang, C.-J., Chuang, Y.-T., Yang, D.-X., Chen, I.-F., Chen, Y.-J. & Hu, K.-W. (2008). A mobility-aware link enhancement mechanism for vehicular ad hoc networks, *EURASIP J. Wirel. Commun. Netw.* 2008: 1–10.

Kaaniche, H. & Kamoun, F. (2010). Mobility prediction in wireless ad hoc networks using neural networks, *Journal of Telecommunications* 2(1).

Matolak, D. W., Sen, I. & Xiong, W. (2006). Channel modeling for v2v communications, *Mobile and Ubiquitous Systems - Workshops, 2006. 3rd Annual International Conference on*, pp. 1 –7.

Maurer, J., Fugen, T., Schafer, T. & Wiesbeck, W. (2004). A new inter-vehicle communications (ivc) channel model, Vol. 1, pp. 9 – 13 Vol. 1.

Molisch, A., Tufvesson, F., Karedal, J. & Mecklenbräuker, C. (2009). A survey on vehicle-to-vehicle propagation channels, *Wireless Communications, IEEE* 16(6): 12 –22.

Nagel, R. (2010a). Altruistic traffic limits computation in wireless broadcast networks, *Proceedings of the Third Internation Conference on Advances in Mesh Networks*.

Nagel, R. (2010b). The effect of vehicular distance distributions and mobility on vanet communications, *Proceedings of the IEEE Intelligent Vehicles Symposium*.

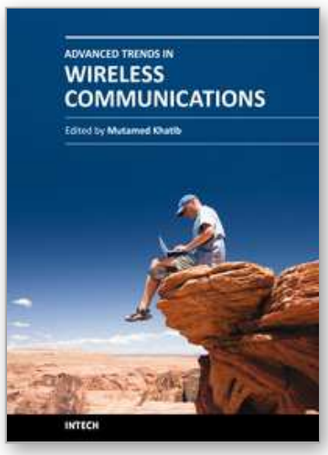
Nagel, R. & Eichler, S. (2008). Efficient and realistic mobility and channel modeling for vanet scenarios using omnet++ and inet-framework, *Simutools '08: Proc. of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, ICST, pp. 1–8.

Paier, A., Karedal, J., Czink, N., Dumard, C., Zemen, T., Tufvesson, F., Molisch, A. F. & Mecklenbräuker, C. F. (2009). Characterization of vehicle-to-vehicle radio channels from measurements at 5.2 ghz, *Wirel. Pers. Commun.* 50(1): 19–32.

Rezende, C. G., Pazzi, R. W. & Boukerche, A. (2009). An efficient neighborhood prediction protocol to estimate link availability in vanets, *MobiWAC '09: Proceedings of the 7th ACM international symposium on Mobility management and wireless access*, ACM, New York, NY, USA, pp. 83–90.

Rodas, J. & Cascon, C. J. E. (2010). Dynamic path-loss estimation using a particle filter, *International Journal of Computer Science Issues* 7(3).

Samaan, N. & Karmouch, A. (2005). A mobility prediction architecture based on contextual knowledge and spatial conceptual maps, *Mobile Computing, IEEE Transactions on* 4(6): 537 – 551.



Advanced Trends in Wireless Communications

Edited by Dr. Mutamed Khatib

ISBN 978-953-307-183-1
Hard cover, 520 pages
Publisher InTech
Published online 17, February, 2011
Published in print edition February, 2011

Physical limitations on wireless communication channels impose huge challenges to reliable communication. Bandwidth limitations, propagation loss, noise and interference make the wireless channel a narrow pipe that does not readily accommodate rapid flow of data. Thus, researches aim to design systems that are suitable to operate in such channels, in order to have high performance quality of service. Also, the mobility of the communication systems requires further investigations to reduce the complexity and the power consumption of the receiver. This book aims to provide highlights of the current research in the field of wireless communications. The subjects discussed are very valuable to communication researchers rather than researchers in the wireless related areas. The book chapters cover a wide range of wireless communication topics.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Robert Nagel and Stefan Morscher (2011). Connectivity Prediction in Mobile Vehicular Environments Backed By Digital Maps, Advanced Trends in Wireless Communications, Dr. Mutamed Khatib (Ed.), ISBN: 978-953-307-183-1, InTech, Available from: http://www.intechopen.com/books/advanced-trends-in-wireless-communications/connectivity-prediction-in-mobile-vehicular-environments-backed-by-digital-maps

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen