PODER EXECUTIVO
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

PREVISÃO DE LOCALIZAÇÃO EM REDES AD HOC VEICULARES

LEANDRO NELINHO BALICO

Manaus

Setembro de 2015

PODER EXECUTIVO
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

LOCALIZATION PREDICTION IN VEHICULAR AD HOC NETWORKS

LEANDRO NELINHO BALICO

Manaus

September 2015

LEANDRO NELINHO BALICO

PREVISÃO DE LOCALIZAÇÃO EM REDES AD HOC VEICULARES

Tese apresentada ao Programa de Pós-Graduação em Informática do Instituto de Computação da Universidade Federal do Amazonas como requisito parcial para a obtenção do grau de Doutor em Informática.

Orientador: Horácio Antonio Braga Fernandes de Oliveira

Manaus

Setembro de 2015

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

B186p

Balico, Leandro Nelinho
Previsão de Localização em Redes Ad Hoc Veiculares / Leandro Nelinho Balico. 2015
76 f.: il. color; 31 cm.

Orientador: Horácio Antonio Braga Fernandes de Oliveira
Coorientador: Eduardo Freire Nakamura
Tese (Doutorado em Informática) - Universidade Federal do Amazonas.

1. Redes Ad Hoc Veiculares. 2. Previsão de Localização. 3. Previsão de Séries Temporais. 4. Rastreamento de Alvos. 5. Roteamento Geográfico. I. Oliveira, Horácio Antonio Braga Fernandes de II. Universidade Federal do Amazonas III. Título

[Folha de Aprovação]

Quando a secretaria do Curso fornecer esta folha,
ela deve ser digitalizada e armazenada no disco em formato gráfico.

Se você estiver usando o `pdflatex`,
armazene o arquivo preferencialmente em formato PNG
(o formato JPEG é pior neste caso).

Se você estiver usando o `latex` (não o `pdflatex`),
terá que converter o arquivo gráfico para o formato EPS.

Em seguida, acrescente a opção `approval={`*nome do arquivo*`}`
ao comando `\ppgccufmg`.

Se a imagem da folha de aprovação precisar ser ajustada, use:
`approval=[`*ajuste*`][`*escala*`]{`*nome do arquivo*`}`
onde *ajuste* é uma distância para deslocar a imagem para baixo
e *escala* é um fator de escala para a imagem. Por exemplo:
`approval=[-2cm][0.9]{`*nome do arquivo*`}`
desloca a imagem 2cm para cima e a escala em 90%.

LEANDRO NELINHO BALICO

LOCALIZATION PREDICTION IN VEHICULAR AD HOC NETWORKS

Thesis presented to the Graduate Program in Computers Science of the Universidade Federal do Amazonas in partial fulfillment of the requirements for the degree of Doctor in Computers Science.

Advisor: Horácio Antonio Braga Fernandes de Oliveira

Manaus

September 2015

A todos que contribuem com a educação.

Agradecimentos

Agradeço a Deus por toda a oportunidade de aprimoramento.

Agradeço ao meu orientador, Professor Horácio, pelos esclarecimentos, ajuda, pelo incentivo que tornou possível a realização deste trabalho e, acima de tudo, exemplo. Ao meu co-orientador, Professor Nakamura pelo apoio e pelas idéias que contribuíram para a melhoria do trabalho realizado. Ao Professor Richard pela oportunidade de realizar parte deste trabalho na UOIT. Gostaria de agradecer o grande incentivo dos Professores Barreto, Loureiro, Pio, Souto, Edjar e Professora Tayana.

Agradeço aos colegas do ICOMP da UFAM, pela hospitalidade, amizade e apoio em minha estadia em Manaus. Ao pessoal do laboratório de Computação Móvel e do ANTs da UOIT pela companhia diária, pelos momentos descontraídos e pelas contribuições prestadas. Aos Professores da UFRR, pelo incentivo e ajuda no durante esse trabalho.

Finalmente, à minha família e amigos, que mesmo distantes me apoiaram e incentivaram.

Resumo

Sistemas de localização desempenham um papel importante em muitas aplicações para Redes Ad Hoc Veiculares (VANets). Embora técnicas de fusão de dados podem prover informações de localização confiáveis para atender a maioria dos requisitos de aplicações em VANets, aperfeiçoamentos nos sistemas de localização são necessários e desejáveis. Características únicas de VANets tais como restrições de mobilidade, o comportamento do condutor e a natureza de alta velocidade de deslocamento dos veículos podem causar rápidas e constantes mudanças na topologia da rede, levando à *disseminação de informações de localização desatualizadas*.

Nesta tese, nós identificamos que para solucionar o problema de *disseminação de informações de localização desatualizadas* em VANets, uma alternativa é o uso de previsão de localização futura de veículos. A principal ideia desta abordagem é utilizar a previsão de localização como uma extensão para o sistema de Fusão de Dados de localização. Em tal abordagem, uma posição futura de um automóvel é predita para um determinado fragmento de tempo futuro e utilizada para tomar vantagem de uma janela de espaço-tempo de uma trajetória vectorial em vez de um ponto de localização estático. Portanto, nesta tese discutimos em detalhes esse assunto, estudando e analisando o uso da previsão de localização como uma forma natural para aprimorar aplicações e serviços em VANets.

Utilizando localização predita como uma métrica para comunicação de dados em VANets, nós propomos uma solução para o problema de *divulgação de informações localização desatualizado* chamada LPRV (*Localization Prediction-based Routing for VANets*). Em nosso algoritmo proposto, o encaminhamento de pacotes é realizado por nós com localizações preditas mais próximas do destino de entrega, sem a necessidade de troca de mensagens de controle adicional. O algoritmo proposto também explora o conhecimento de um mapa digital para limitar o escopo de trocas de mensagens no caminho mais curto para veículos entre a origem e destino.

Palavras-chave: Redes Ad Hoc Veiculares, Previsão de Localização; Previsão de Séries Temporais, Rastreamento de Alvos; Roteamento Geográfico; Geocast;

Abstract

Localization systems play a major role in many applications for Vehicular Ad Hoc Networks (VANets). Although Data Fusion techniques can provide reliable localization information for most of the application requirements in VANets, enhancements on the localization systems are required and desirable. Unique characteristics of VANets such as mobility constraints, driver behavior, and high speed displacement nature of vehicles cause rapid and constant changes in network topology, leading to *dissemination of outdated localization information*.

In this thesis, we identify that to circumvent the problem of *dissemination of outdated localization information* in VANets, an alternative is the use of predicted future locations of vehicles. The main idea of this approach is to use the localization prediction as an extension of a Data Fusion localization system. In such an approach, a future position of a vehicle is predicted for a given future time step and used in order to take advantage of a future time-space window of a vectorial trajectory rather than a static localization point. Thus, in this thesis we further discuss this subject by studying and analyzing the use of localization prediction as natural way to improve VANets applications.

Using vehicles predicted locations as a metric for data communication in VANets, we propose a solution for the problem of *dissemination of outdated localization information* called LPRV (Localization Prediction-based Routing for VANets). In our proposed algorithm, packet forwarding is performed by nodes with predicted future localization closer to the delivery destination, without the need for exchanging additional control message. The proposed algorithm also explores the knowledge of a digital map to limit the scope of message exchanges in the shortest path for vehicles between source and destination.

Keywords: Vehicular Ad Hoc Networks, Localization Prediction, Time Series Prediction, Target Tracking, Position-based Routing, Geocast;

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Motivation

Recent advances in mobile computing, wireless communication and sensing have enabled the development of a number of interesting and desirable applications in Intelligent Transportation Systems (ITS). In this context, Vehicular Ad Hoc Networks (VANets) (Boukerche et al., 2008; Papadimitratos et al., 2009; Yousefi et al., 2006; Hartenstein and Laberteaux, 2008) emerge as new technology to integrate wireless networks capabilities to vehicles, providing ubiquitous connectivity as well as allowing vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication. Thus, interconnected vehicles can collect and share information about themselves and surrounding environments in real time. Therefore, there is an extensive list of potential applications for VANets, where we can highlight categories related to safety, transport efficiency and information/entertainment applications (Hartenstein and Laberteaux, 2008). Among these applications, safety plays a special hole in VANets. The growing number of traffic congestions, fatalities and injuries, due to the increasing number of vehicles in operation worldwide, has been recognized as a social cost and a problem to be solved by modern society (Papadimitratos et al., 2009; Al-Sultan et al., 2014).

Regarding the operation of localization systems in vehicular networks, the estimation of a vehicle's dynamic state is one of the most fundamental Data Fusion tasks for ITS applications (Schubert et al., 2008). Although Data Fusion techniques can provide reliable localization information for most of the application requirements in VANets (Nakamura et al., 2007), enhancements on the localization systems are still required and desirable. Unique characteristics of VANets like mobility constraints, driver behavior, and high speed displacement nature of vehicles cause rapid changes in network topology (Yousefi et al., 2006). Thus, leading to the *dissemination of outdated localization information*, specifically when the network packet delay is high. In this context, some protocols that require accurate position information increase the frequency of periodic messages (beacons) as a naive solution for this problem. However, this approach leads to unnecessary overhead in the number of transmitted packets, which causes a high channel occupancy with an increased number of medium access collisions, actually increasing delay (Boukerche et al., 2009; H. Nguyen, 2012). Therefore, one problem to be solved in VANets' localization systems is how to avoid the *dissemination of outdated localization information.*

To circumvent the problem of the *outdated localization information dissemination* in wireless communications, some pioneer studies (Kaaniche and Kamoun, 2010; Lee and Krumm, 2011; Boukerche et al., 2009; Rezende et al., 2009; Huang et al., 2008; Agarwal and Das, 2003) tackle this problem by predicting the future localization of a mobile node in a small time window. In these studies, well known methods applied in localization, target tracking, and time series prediction such as dead reckoning, Bayesian Filtering, and Machine Learning are proposed as a metric to achieve improvements on a single particular application. However, specifically from the viewpoint of VANets, current proposals do not discuss how the localization prediction thechniques can be used to improve internal tasks and applications, such as an enhancement of the localization system as a whole.

1.2 Objectives

This work aims to provide a general discussion for localization prediction in VANets as enhancement of the Data Fusion localization system, allowing us to identify open issues, understand the requirements and the implications of using localization prediction in vehicular networks. Regarding the problem of the *outdated localization information dissemination* in VANets, the main goals of this work is to demonstrate, design, and evaluate the performance of localization prediction thecniques as natural way to improve applications and services for vehicular networks.

To achieve these goals, several secondary objetives need to be acomplished. In order to demonstrate, design, and evaluate the performance of localization prediction thecniques in VANets, the following goals need to be achieved:

1. *Demonstrate and evaluate proposed approaches for localization, target tracking and time series prediction techniques that can be used to estimate the future position of a vehicle to realistic VANet scenarios;*

2. *propose a routing algorithm using vehicles predicted locations as a metric for data communication in VANets;* and

3. *analyze the performance and demonstrate the efficiency of the proposed routing solutions in VANet scenarios.*

1.3 Main Contributions

The main contributions of this work in the order they appear in this document are:

1. **A survey on localization prediction in vehicular ad hoc networks.** Although this is not the thesis central contribution, this comprehensive survey about localization prediction in VANets is worth to be mentioned. We surveyed proposed approaches for localization, target tracking and time series prediction techniques that can be used to estimate the future position of a vehicle. It discuss how the localization predictions methods can improve most VANet applications, especially critical ones. In this survey we argue that localization prediction for VANets as an extension of a Data Fusion localization system is a feasible approach to circumvent the problem of the *dissemination of outdated localization information* in vehicular networks. We then show how localization prediction techniques can be used to compute an accurate predicted positions based on a number of relatively inaccurate sample position estimations. This survey is presented in Chapter 2.

2. **A prediction-based routing algorithm for vehicular ad hoc networks.** The main thesis contribution consists in a new VANet routing algorithm that uses the knowledge of the vehicles predicted locations to improve the routing performance in several aspects. Our proposed algorithm, called LPRV (Localization Prediction-based Routing for VANets), exploits the knowledge of vehicles predicted future locations and a digital map as metrics to forward data packets, without the need for exchanging any control message. Simulation results demonstrated the efficiency of the proposed solution for different VANet scenarios and the benefits of using vehicles predicted locations as a metric for data communication, especially in terms of delivery rate, number of hops and delay, with a reduced number transmitted packets. This solution is presented in Chapter 3.

1.4 Document Outline

This thesis is divided into 4 Chapters. The first part of this work, composed of Chapter 2 presents an overview and definition of the localization prediction methods in VANets. We highlight potential advantages of using a localization prediction system in several VANet applications scenarios. We describe the localization, target tracking and time series prediction methods and discuss the applicability, advantages, and limitations of the analyzed solutions. We show our performance evaluation when both solutions are used in a realistic VANet scenario.

In Chapter 3, we consider that VANets apply localization prediction techniques. Hence, we propose the LPRV routing algorithm to use the knowledge of the vehicles predicted locations to improve the routing performance. The performance of the proposed solution is evaluated through simulations. Finaly, Chapter 4 summarizes the thesis results by presenting the current contributions and future research directions.

Chapter 2

Localization Prediction in Vehicular Ad Hoc Networks

2.1 Introduction

Many applications for VANets can take advantage of localization techniques. One of the most interesting problems to be solved in vehicular networks is how to provide an anywhere and anytime highly accurate and reliable localization information (Boukerche et al., 2008). Nowadays, most of produced vehicles are delivered with a Global Positioning System (GPS) and third-party in-car navigation systems can be installed on used vehicles at a reasonable cost (Papadimitratos et al., 2009; Skog and Handel, 2009). Also, recent technological developments, notably in mobile computing, wireless communication, and remote sensing allow vehicles to turn into sophisticated computing systems. With several coupled processors and integrated sensors dedicated to the vehicle operation, the development of more sophisticated applications and services for these networks is a reality today.

However, for VANets' critical applications that are dependent on high accurate and available localization systems, GPS shows some undesired problems such as being unavailable or not being accurate enough (Alam and Dempster, 2013). For this reason, a number of other localization techniques such as Map Matching, Dead Reckoning, Cellular Localization, Image/Video Processing, Localization Services, and Relative Distributed Ad Hoc Localization are used combined in VANets to overcome such GPS limitations (Boukerche et al., 2008; Skog and Handel, 2009) (As depicted in Figure 2.1). In this approach, Data Fusion techniques are applied to improve the localization system by combining several localization techniques into a single solution that is more robust and precise than using any individual approach (Nakamura et al., 2007; Alam et al., 2013; Golestan et al., 2012).

Regarding the problem of outdated localization information dissemination in wire-

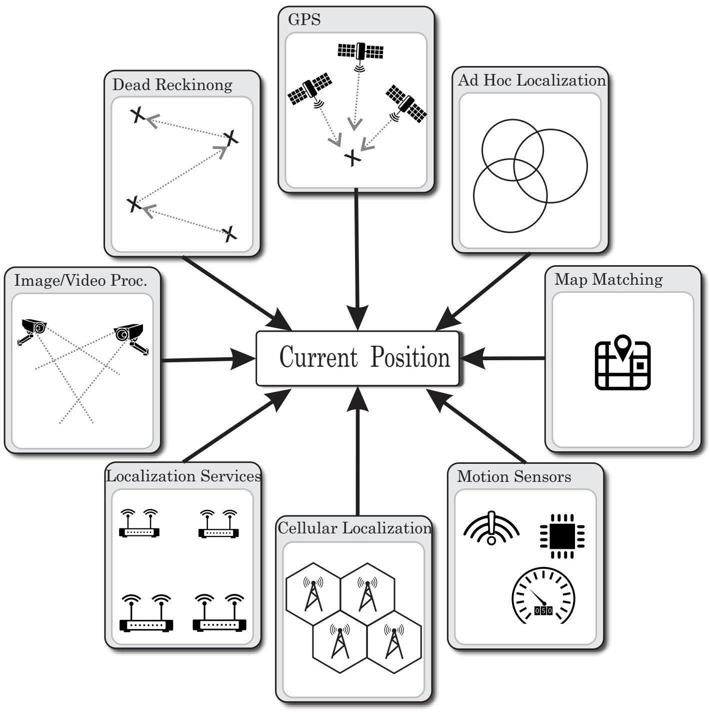


Figure 2.1. Localization techniques to compute vehicles' current localization used combined in VANets to overcome GPS limitations (Boukerche et al., 2008).

less communications, in this work we consider localization prediction as natural way to improve VANets' applications. We study well known methods applied in localization, target tracking and time series prediction, such as Dead Reckoning, Bayesian Filtering and Machine Learning as an enhancement of the VANets' localization system. As depicted in Figure 2.2, the main idea of this approach is to use the localization prediction as an extension of a Data Fusion localization system. In such a method, a future position of a vehicle is predicted for a given future time step and used to improve an application service. The main idea is to take advantage of a future time-space window of a vectorial trajectory rather than an actual static localization point. Thus, as a solution for the dissemination of outdated localization information in VANets, in this Chapter we discuss the use of localization prediction as al way of improving VANets' applications. We then survey proposed localization, target tracking and time series prediction techniques that can be used to estimate the future position of a vehicle. We highlight their advantages and disadvantages through an analytical analysis discussion based on the literature review highlighting its potential application scenarios for VANets. Then, we present a set of experiments that show the results of such techniques when applied to a realistic VANet scenario indicating clearly the applicability, pros, and cons of each one.

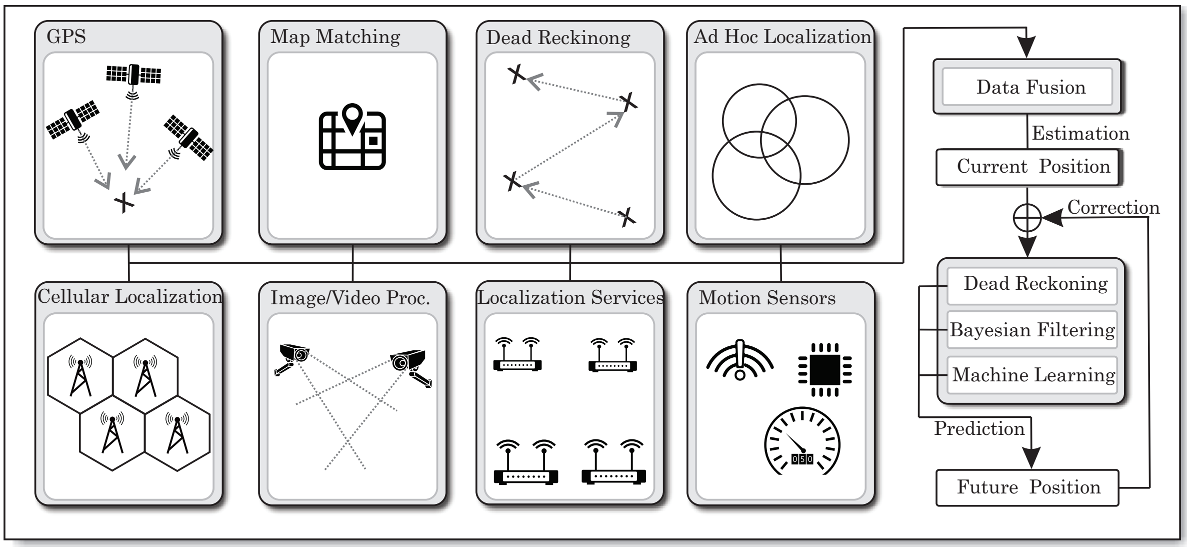


Figure 2.2. Localization techniques, Data Fusion and localization prediction in VANets.

The remainder of this Chapter is organized as follows. In the next Section, we highlight potential advantages of using a localization prediction system in several VANet applications scenarios. In Section 2.3, we state the problem of predicting a vehicle future location whereas in Section 2.4, we describe the localization, target tracking and time series prediction methods in this context. In Section 2.5, we discuss the applicability, advantages, and limitations of the analyzed solutions. Section 3.4 shows our performance evaluation when both solutions are used in a realistic VANet scenario. Finally, Section 4.1 presents our conclusions.

2.2 Applications that can take advantage of Localization Prediction

A key goal of any application for VANets is to provide a time horizon of new information sources relevant to driving safety, comfort and transportation efficiency (Papadimitratos et al., 2009). V2V and V2I communications allow the development of a large number of applications. Each kind of such applications requires or can take advantage of a certain degree of reliability and accuracy in the computed locations of vehicles and/or infrastructure units. Basically, the applications for vehicular networks can be summarized as safety, transport efficiency, and information/entertainment applications (Al-Sultan et al., 2014; Yousefi et al., 2006; Hartenstein and Laberteaux, 2008; Papadimitratos et al., 2009). In all of these categories, expanding the time and space of a localization system by using future predicted locations can improve the

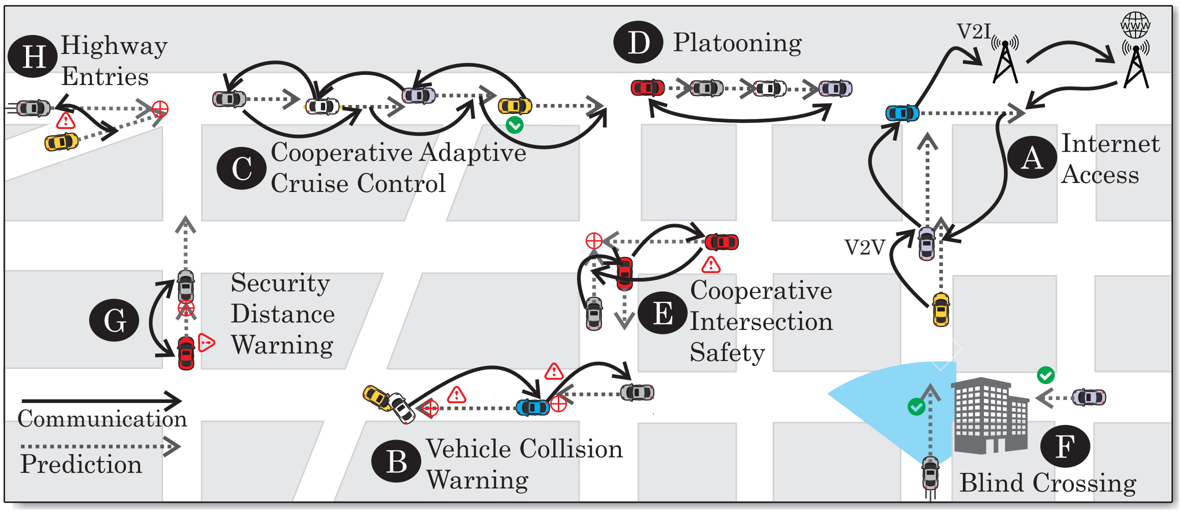

H Highway Entries
C Cooperative Adaptive Cruise Control
D Platooning
V2I
A Internet Access
V2V
E Cooperative Intersection Safety
G Security Distance Warning
Communication
Prediction
B Vehicle Collision Warning
F Blind Crossing

Figure 2.3. Several VANet applications that can take advantage of localization prediction in highways and urban scenarios. (A) Internet access. (B) Vehicle Collision Warning Systems. (C) Cooperative Adaptive Cruise Control. (D) Vehicle Following or Platooning. (E) Cooperative Intersection Safety. (F) Blind Crossing. (G) Security Distance Warning.

performance of the applications. In the following, we will further discuss how these services can be improved by using localization prediction thecniques in highways and urban scenarios.

As depicted in Figure 2.3A, a first example of application for VANet that can take advantage of localization prediction is Internet access. The packet forwarding can use the vehicle's predicted position to guide packets to the more suitable Internet gateways, roadside unity or vehicle in a greedy forwarding fashion, according to the predicted position and the time for the vehicles to reach such locations. Besides the advantage of computing a real shortest path in relation to the vehicle displacement in time and space, this approach can also considerably reduce the packet delay since the shortest path can be computed in terms of time (Balico et al., 2015). The same idea can be applied to the V2V and V2I communication locally, by choosing the next best hop according to its neighbors future predicted location. As these applications also provide services about road and surrounding environmental conditions, besides the benefits to the driver's safety, the use of localization prediction can also improve the driver's experience.

One of the most interesting applications of VANets that can be enhanced using localization prediction is Vehicle Collision Warning Systems (as shown in Figure 2.3B). This type of application is one of the most important for driver's safety since it provides assistance for drivers to avoid hazards. One part of these systems is the Security Distance Warning (Figure 2.3G), in which the driver is notified when a threshold distance to another vehicle is reached. Instead of using the current location of a vehicle, these applications can use the predicted future location of a vehicle to check when the distance between two vehicles, or between a vehicle and an obstacle, reached an unsafe threshold. In this case, the system can check in a few milliseconds or even seconds in advance the potential risk to take further measures by checking if the trajectories described by predictions will collide (Figure 2.3B). Its importance to notice that, such informations provided in advance to detect potential hazards can be crucial to avoid and prevent such dangerous situations. In this case, the use of predicted future locations can improve the speed computation process and also the packet exchange process, while also providing relevant data for guiding the drivers for further reactions.

Furthermore, Cooperative Active Safety applications for VANets (Hrizi et al., 2012) (Figures 2.3B, 2.3E, 2.3F, 2.3G, 2.3H) require an up-to-date knowledge of a vehicle's surrounding entities which is obtained when all vehicles broadcast their status information (position, speed) in a collaborative fashion. The packets containing this information need to be periodically transmitted, leading to wireless congestion and impacting the accuracy and reliability of the safety application. In this case, the use of predicted locations can avoid the need of exchanging periodically localization messages, since the predictions are valid for a time window interval. Thus, the use of a prediction localization system has a potential to increase the reliability of the safety application which is quite a desirable feature.

Another interesting application of VANets is Cooperative Intersection Safety (as depicted in Figure 2.3E). In this application, vehicles arriving at a road/street intersection exchange messages in order to make a safe crossing. Highway Entrance, as show in Figure 2.3H, is also a similar application that can take advantage of localization prediction. Besides ensuring a safe crossing and highway entrance, it is also possible to make a Blind Crossing (as shown in Figure 2.3F), where there is no light control and the vehicles cooperate among themselves to make a crossing, even when the driver's field of view is obstructed by buildings. In these applications, besides avoiding unnecessary packet transmissions overhead, the localization prediction can provide information to detect and prevent potential hazards by checking in advance the potential risk of collisions in the computed trajectories described by predictions.

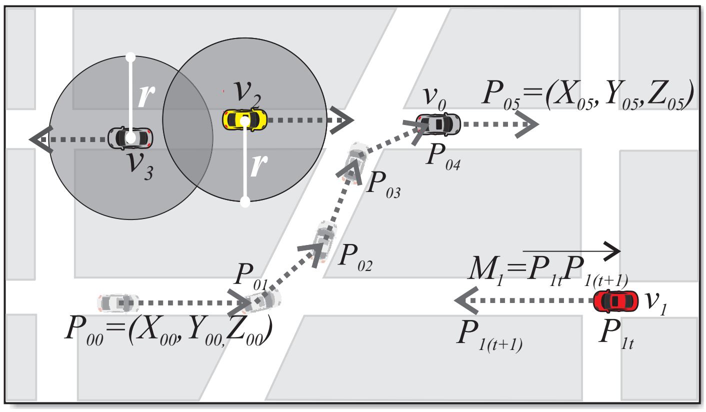
In Cooperative Adaptive Cruise Control (Figure 2.3C), the vehicle speed is adjusted to maintain the same speed of the vehicles ahead and those behind in a group without requiring driver intervention. Usually in this type of application, the speed is set by the driver and the system exchanges messages between the vehicles using V2V communication to coordinate the vehicles' speed adaptively. In this case, the use of predicted future locations can be used to compute the speed of the vehicles in order to keep safe distances among themselves. Vehicle Following or Platooning, as shown in Figure 2.3D, is an application used to make one or more vehicles follow a leader vehicle forming a train-like unit. The use of predicted locations in this kind of application can help to improve the tracking of the leader and members' position based on the computed trajectories described by predictions as well as to help guide its following. Also, it can help keeping a minimum distance between vehicles in advance preventing accidental collision.

As show in this section, several types of applications in VANets that can take advantage of a localization system and also can achieve improvements when using localization prediction. In the next Sections, we formally present the localization prediction problem in the context of VANet some proposed approaches to tackle the target tracking and time series prediction problems.

2.3 Problem Statement

In this section, we formally present the concepts used in this work.

Definition 2.3.1 (Vehicular Ad Hoc Network)**.** We define a VANet as a Euclidean graph $G = (V, E, r)$, where $|V| = N$ is the number of nodes and r is the communication range; $V = \{v_0, v_1, v_2, \dots, v_{N-1}\}$, where $\{v_0, v_1, \dots, v_N\}$ is the set of vehicles; $\langle i, j\rangle \in E$ iff v_i reaches v_j, in other words, v_i is inside the communication range r of a node v_j; and $\forall v_i \in V$, $P_{it} = (X_{it}, Y_{it}, Z_{it}) \in \mathbb{R}^3$ is the computed position of nodes v_i (i.e., using a localization system), $L_{it} = (X_{it}, Y_{it}, Z_{it})$ is the real position of nodes at a discrete time t and S_i its displacement speed.



r
v_2
r
v_3
v_0
$P_{05}=(X_{05}, Y_{05}, Z_{05})$
P_{04}
P_{03}
P_{02}
P_{01}
$M_1=\overrightarrow{P_{1t}P_{1(t+1)}}$
v_1
$P_{00}=(X_{00}, Y_{00}, Z_{00})$
$P_{1(t+1)}$
P_{1t}

Figure 2.4. VANet: network nodes definition, location and location prediction.

Definition 2.3.2 (Vehicle Future Location Prediction - $P_{i(t+1)}$)**.** The prediction of vehicle i future position for a discrete time step $t+1$. It can be defined as a time series regression forecasting problem and also can be formulated as a target tracking problem. Tracking is usually stated as an estimation problem based on a series of measurements. The primary objective of target tracking is to detect and continuously estimate the evolution of the target state with respect to time and update the estimation with measurements (Ramos et al., 2012; Li and Jilkov, 2003). Since almost all maneuvering target tracking methods are model based, we can define the trajectory prediction by the discrete-time state-space model as follows:

$$P_{i(t+1)} = f_t(P_{it}, u_t) + w_t, \tag{2.1}$$

$$o_t = h_t(P_t) + b_t \tag{2.2}$$

where P, u, o are the target state, input control and observation, respectively, w and b are the process and measurement noise, respectively, f and h are function vectors, and $t \geq 1$ is the measurement epoch (Ramos et al., 2012). Thus, based on the knowledge of the current position of a vehicle (P_{it}) at a step time t and the knowledge of the $t-1$ steps, the prediction of vehicle's future position is given by target state estimate $P_{i(t+1)}$ which will estimate the future position $(X_{i(t+1)}, Y_{i(t+1)}, Z_{i(t+1)})$ for the next time step $t+1$. It is important to notice that our approach differs from conventional target tracking methods since each node performs the target tracking only on its own set of localization samples, without any observations from other network nodes. In other words, each network node performs self-target tracking.

Definition 2.3.3 (Vehicle Motion Vector - M_i))**.** This vector represents the movement of a vehicle i from its current position to a future computed position. For the sake of simplification, we consider that a vehicle will maintain the trajectory of a straight line during the time required to it reaches the computed future location. This line is defined as $M_i = \overrightarrow{P_{it}P_{i(t+1)}}$ (as depicted in Figure 2.4), where P_{it} is the current vehicle's position, $P_{i(t+1)}$ is its predicted future position and S_i its displacement speed.

2.4 Location Prediction Methods

Localization-Based and in-car navigation systems have been identified as a key technology to the development and operation of VANets (Papadimitratos et al., 2009; Skog and Handel, 2009; Obradovic et al., 2006). According to Boukerche et al. (Boukerche et al., 2008), an interesting aspect of VANets is that most localization techniques can be applied easily to these network and they can be categorized as Map Matching, Dead Reckoning, Cellular Localization, Image/Video Processing, Localization Services, and Relative Distributed Ad Hoc Localization. Since vehicular networks have no significant power constraints unlike sensor and other types of mobile networks and, also can be equipped with a wide variety of sensors and processor units (Ramos et al., 2012), Data-fusion techniques are a natural solution to improve VANets localization system providing a precision of centimeters (Skog and Handel, 2009) to compute the vehicle's position. However, a common problem in this approach is the dissemination of outdated localization information and also unnecessary overhead of transmitted packets.

To overcome these problems, methods for predicting future locations of vehicles like target tracking and time series regression forecasting are an alternative solution as an extension of the Data Fusion localization system for vehicular networks. The main reason for that relies on the fact that, according with Li et al. (Li et al., 2014), there is a strong regularity in the daily vehicular mobility in both temporal and spatial dimensions, which can allow a high degree in the predictions' accuracy. Also, this study showed that for Shanghai and Beijing vehicular traces, the location predictability can reach levels of 80 % to 99 % of accuracy.

An interesting aspect of target detection, tracking, and recognition is that they are closely interrelated areas, with significant overlaps (Li and Jilkov, 2003). Although the Bayesian Filtering state estimation is the main approach to solve the tracking problem (Ramos et al., 2012; Lee and Krumm, 2011), the problem of predicting a future location of a vehicle can also be seem as a time series prediction. Time series is a set of observations from past until the present. In this case, it is possible to apply Machine Learning techniques to build the models from training data and even to adjust those models dynamically. Learning represents a trade-off between accuracy and generality and for the case of VANets, it represents a compromise in keep the model accurately enough and, at the same time, capable of deal with different trajectories described with a wide variety of mathematical entities. Another method to approach the problem of predicting a future position of a vehicle is the Dead Reckoning. In this approach, the future position of a vehicle can be computed based on its last known position and movement information as direction, speed, acceleration, distance and time (Parker and Valaee, 2006). In the following, these techniques will be discussed further.

2.4.1 Dead Reckoning

Dead Reckoning (DR) is an ancient navigation technique where a current position can be computed using a previous last known location (Krakiwsky et al., 1988; King et al., 2006) or a future position can be computed using a current known location. This technique uses the last known position, also known as a fix, the displacement and the direction information from vehicle's sensors to update the location information. Dead Reckoning as a stand-alone localization technique can be used only for short periods of GPS unavailability in VANets since it can accumulate errors easily. For high speed vehicles, such as vehicles moving at speeds about 100 km/h, dead reckoning can reach localization errors up to 20 m (Boukerche et al., 2008; Parker and Valaee, 2006) when used as a stand-alone localization solution. For this reason, this localization technique in VANets is used to overcome the limitations of GPS/GNSS and it is considered only as a backup system for periods of GPS/GNSS outage.

However, if it is possible to assume that each node in the network is aware of its location, DR can be applied to predict future location of nodes as shown in Argawal et al. (Agarwal and Das, 2003). In the DR model presented, each node constructs a movement model for itself by periodically sampling its location estimates. In the next step, the DR model computes the velocity components S_{Xi}, S_{Yi} and S_{Zi} along the X, Y and Z axes from two successive location measurements $(X_{i(t-1)}, Y_{i(t-1)}, Z_{i(t-1)})$ and (X_{it}, Y_{it}, Z_{it}) taken at times $t-1$ and t as follows:

$$\frac{S_{Xi} = X_{it} - X_{i(t-1)}}{t-(t-1)}, \tag{2.3}$$

$$\frac{S_{Yi} = Y_{it} - Y_{i(t-1)}}{t-(t-1)} \tag{2.4}$$

and

$$\frac{S_{Zi} = Z_{it} - Z_{i(t-1)}}{t-(t-1)}. \tag{2.5}$$

To predict the future location $P_{i(t+1)} = (X_{i(t+1)}, Y_{i(t+1)}, Z_{i(t+1)})$ of the node i at the current time as per the following formula:

$$X_{i(t+1)} = X_{it} + (S_{Xi} \times (t+1) - t), \tag{2.6}$$

$$Y_{i(t+1)} = Y_{it} + (S_{Yi} \times (t+1) - t) \tag{2.7}$$

and

$$Z_{i(t+1)} = Z_{it} + (S_{Zi} \times (t+1) - t) \tag{2.8}$$

where, $t+1$ is the next time step in which the future location will be computed.

This DR approach was used as a basis for a prediction method in a localization system called Dead Reckoning Method (DRM) for Mobile Ad hoc Network (MANet) (Agarwal and Das, 2003). The DRM main idea is that each node is able to track the location of every other node in the network and then able to predict the movement of every other node. Thus, every node is capable of constructing a topology of the network using the knowledge of the predictions. The authors of DRM demonstrated that the DRM-Based technique applied in a geographic routing approach delivered superior routing performance when compared to popular protocols such as DSR and AODV in MANets. In King et al. (King et al., 2006), DR was utilized to improve beacon accuracy in the Position-Based Forwarding (PBF) protocol, a greedy position-based packet forwarding for vehicular highway scenarios. The DR approach tackles the problem of always-outdated perception of neighbor positions for low beaconing rates. In this technique when the GPS signal is temporarily unavailable, a mobile node estimates its current position based on its last measured GPS location and its motion parameters (speed, orientation, and time). In Wahab et al.(Wahab et al., 2013), was proposed a GPS-free localization framework aiming at providing accurate vehicle localization for road safety applications in VANets. The proposed localization framework uses two-way time of arrival with partial use of dead reckoning to locate the vehicles based on communication with a single roadside unity.

2.4.2 Machine Learning

Machine Learning techniques in time series forecasting have been applied in many areas such as financial market prediction, electric utility load forecasting, weather and environmental state prediction, reliability forecasting and wireless communications (Sapankevych and Sankar, 2009). Time series is a set of data samples from past until present and the goal of time series prediction is to estimate some future value based on current and past data samples (Kaaniche and Kamoun, 2010). In this context, the localization prediction problem can be also tackled as a particular case of time series prediction. Mathematically, a time series in VANets can be stated as:

$$P_{i(t+1)} = f(P_{it}, P_{i(t-1)}, P_{i(t-2)}, ...) \tag{2.9}$$

where, for the VANets' context, $P_{i(t+1)}$ is the predicted value of a future position of a vehicle at the discrete time $t+1$ and $P_{it}, P_{i(t-1)}, P_{i(t-2)}, ...$ is a set of computed locations from past until present of a vehicle i and, L_{it} are the respectively target values (real vehicle position). The objective of time series prediction is to find a function $f(P)$ such that, given a set of input/target pairs (P_t, L_t), the predicted value of the time series $P_{i(t+1)}$, at a future point in time $t+1$ is unbiased and consistent. In other words, the predicted point $P_{i(t+1)}$ must be as closest as possible to the real vehicle position $L_{i(t+1)}$.

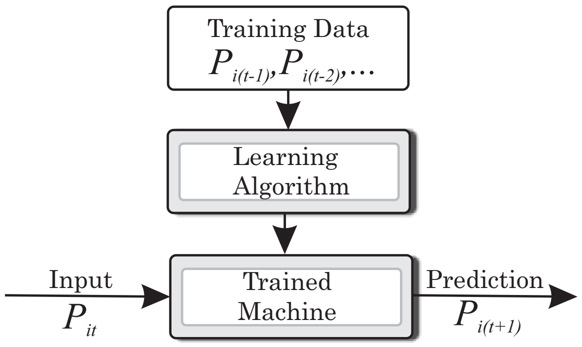


Figure 2.5. Machine Learning phases.

2.4.2.1 Neural Networks

Neural Networks (NNs) were originated in the early 1960s and are parallel distributed information processing systems that implement supervised learning mechanisms that, starting from input/output pairs of examples, are able to generalize and learn in a supervised fashion (Bonissone, 1997; Nakamura et al., 2007). NN s are a well-known option to deal with time series prediction, and for the case of VANets, are suitable by being able to give solutions to complex problems due to their non-linear processing, parallel distributed architecture, self-organization, capacity of learning and generalization, and efficient hardware implementation (Ibnkahla, 2000).

The Multilayer Feed Forward Neural Network (MLNN), also called Multilayer Perceptron (MLP) (often simply called Neural Network), is one of the most popular neural network architectures in use for both classification and regression (Bishop, 1995). The fundamental processing element of a neural network is a neuron (as depicted in Figure 2.7). A neuron can be mathematically described as:

$$P = \phi(\xi) \tag{2.10}$$

$$\xi = \sum_{j=1}^{n} w_j p_j + b. \tag{2.11}$$

A neuron is composed of a linear combiner ξ, an activation function $\phi(\xi)$ and the output signal of the neuron P as depicted in Figure 2.6. The linear combiner output is the weighted sum of the inputs plus a bias term. The activation function gives then the neuron output in terms of the activity level at is inputs: where p_j is the jth input signal, w_j the corresponding synaptic weight, and b the bias term. The activation function may be a linear or non-linear function and there are many activation functions like, e.g. the identity function, the sigmoidal function, the threshold function, etc. The choice of the activation function depends on the nature of the NN application (Haykin, 1998).

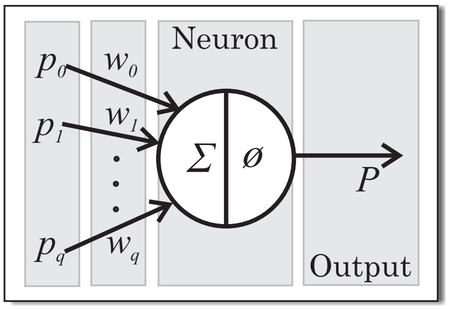


Figure 2.6. Artificial neuron.

A NN is composed of multiple neurons layers connected to each other in a directed graph as shown in Figure 2.6. The input information is processed from the first layer (input layer) to the output layer. Each node in one layer connects with a weight w_{ij} to every node in the following layer. The layer index is denoted by i and P_{ik} is the output of neuron k of layer i given by:

$$P_{ik} = \phi(\xi_{ik}), \tag{2.12}$$

$$\xi_{ik} = \sum_{j=1}^{n(i-1)} w_{ijk} p_{i-1j} + b_{ik} \tag{2.13}$$

were w_{ijk} is the weight that links the output P_{i-1k} to neuron k of layer i. The value $n(i)$ is the number of neurons in layer i.

Applications of NNs do not have a priori knowledge of the correct network weights and a training procedure is required to compute the weights. A method for computing

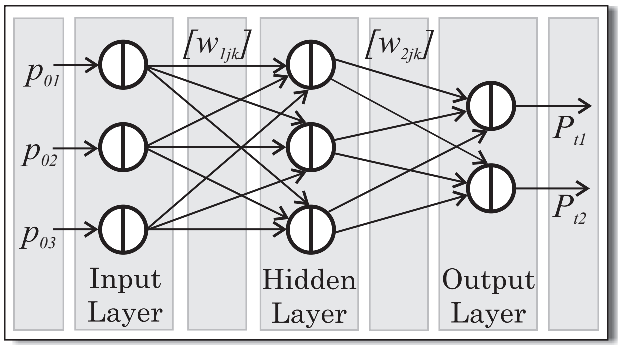


Figure 2.7. A three layer Neural Network.

the gradient of the empirical risk for the activation function of NNs, called the Back Propagation algorithm (BP), was proposed in Rumelhart et al. (Rumelhart et al., 1986) and Lecun (Lecun, 1986). The BP algorithm (Lippmann, 1987) uses a set of input output pairs $(P_t(n), L(n))$ to train the network to achieve the desired mapping. It adjusts the MLP weights aiming at minimizing any differentiable cost function such as the Minimum Squared Error (MSE). The MSE function is the error power between the network output and the desired output, $MSE(n) = ||L(n) - P_t(n)||^2$, where $P_t(n)$ is the NN output vector at time n and $L(n)$ is the desired output (e.g., real vehicle position). The BP algorithm performs a gradient descent on the cost function in order to reach a minimum as follows:

$$w_{ijk}(n+1) = w_{ijk}(n) - \alpha \frac{\partial MSE(n)}{\partial w_{ijk}(n)}, \tag{2.14}$$

were the parameter α is the desired error. This equation can be expressed as:

$$w_{ijk}(n+1) = w_{ijk}(n) - \alpha \delta_k^i P_{(i-1j)}. \tag{2.15}$$

The error term δ_k^t of the output layer is given by:

$$\delta_k^t = \phi'(\xi_{tk}) - (L_k - P_{tk}), \tag{2.16}$$

where ϕ' denotes the derivative of the activation function $\phi'(\xi) = \frac{\partial \phi(\xi)}{\partial \xi}$. The error term δ_k^i of the hidden unit (i,k) can be expressed as a function of the next layer error terms as:

$$\delta_k^i = \phi'(\xi_{ik}) \sum_{j=1}^{n(i+1)} w_{i+1kj} \delta_j^{i+1}. \tag{2.17}$$

Thus, the weight update is performed by propagating errors backwards from the output nodes to the input nodes.

Neural Networks have been applied for node mobility prediction for cellular networks (Liou and Huang, 2005; Capka and Boutaba, 2004). In Kaaniche and Kamoun (Kaaniche and Kamoun, 2010), a Neural Network has been applied to estimate the duration of a communication link based on the time series prediction in MANets. In this approach, a MLP predict the future location of the mobile user based on the time series location observations as the inputs of the NN. The authors also discuss that a variation of the number of neurons of the hidden layer can affect the prediction accuracy. Neural networks also have been applied in VANets for prediction of future lane change trajectory based in Tomar et al. (Tomar et al., 2010), where a NN was proposed to learn and incorporate the human behavior to predict the lane changing trajectory in the near future. The authors discuss that the NN is able to give accurate the prediction for some parts of the path, however, the deviation is significant for certain sections regarding the vehicles' speed.

2.4.2.2 Support Vector Regression

Support Vector Machines (SVM) are supervised learning models based on statistical learning theory, or Vapnik-Chervonenkis theory (VC theory), developed during 1960-1990 by Vladimir Vapnik and Alexey Chervonenkis (Sapankevych and Sankar, 2009; Vapnik, 1995, 1999). The statistical learning theory attempts to explain the learning process through a statistical point of view. Although SVMs were introduced first for binary classification (Vapnik, 1995), they are currently a hot topic in the Machine Learning community and used for many learning fields such as pattern recognition, classification and, in the case of time series prediction, regression analysis (Sapankevych and Sankar, 2009).

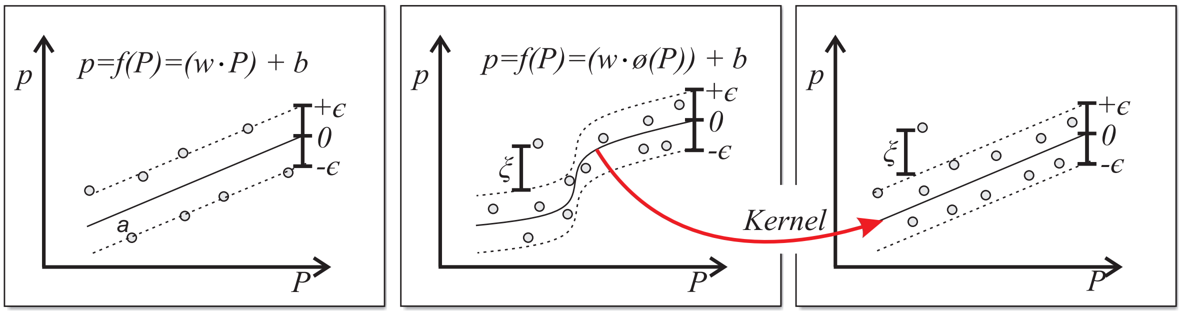


Figure 2.8. SVR prediction function for linear and non-linear regressions.

The SVM concept of a maximum margin hyperplane is mainly applied to classification problems. However, SVM algorithms have been applied for numeric prediction and share many of the classification case properties: they produce a model that can usually be expressed in terms of a few support vectors and can be applied to non-linear problems using kernel functions. Support Vector Regression (SVR) (Müller et al., 1997; Sapankevych and Sankar, 2009; Smola and Schölkopf, 2004) is a method extended from SVMs to solve regression problems. The main idea of SVR is, given a set of input time series data P_k, where k discrete time step of n samples: $k = \{0, 1, 2, ..., n-1\}$, and L_k are the respectively target values (real vehicle position), the goal of SVR (Vapnik, 1995) is to find a function $f(P)$ that approximates the training points aiming at minimizing the prediction error. In other words, the deviant distance between the output predicted values from the training target labels only will be accepted if it is less than ϵ for the same time horizon.

For linear and non-linear regressions the SVR prediction function $f(P)$, given a set of input/target pairs (P_k, L_k), approximates the prediction function by:

$$p = f(P) = (w \cdot P) + b \tag{2.18}$$

$$p = f(P) = (w \cdot \phi(P)) + b. \tag{2.19}$$

As show in Figure 2.8, to deal with the non-linear regression using SVR, it is necessary to map the input space P into a high dimensional feature space $(\phi(P))$. Note that the dot product in Equation 2.19 $(w \cdot \phi(P))$ would have to be computed in this high dimensional space (which is usually intractable) (Smola and Schölkopf, 2004). To overcome this problem, the SVR adopts a strategy in which this dot product is implicitly expressed in a lower dimensional input space (referred to as a kernel function).

The goal of the algorithm is to find the weight vector w and the bias b minimizing the error, as well as, simultaneously maximizing the flatness of the regression function by:

$$C \sum_{i=1}^{n} E(L(i), p(i)) + \frac{1}{2}\|w\|^2, \tag{2.20}$$

where

$$E(L, p) = \begin{cases} L(i) - p| - \epsilon, & \text{if}|L(i) - p| \geq \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{2.21}$$

In the Equation 2.20, the first term $C \sum_{i=1}^{n} E(L(i), p(i))$ is the empirical error (risk). The parameters w and b are measured by the ϵ-insensitive loss function (E, defined in

Equation 2.21). This loss function provides the precision by which the function $f(P)$ is to be approximated, enabling the use of sparse data points to represent the solution. The flatness of the weights in Equation 2.20 means that we seek a small w, which can be achieved by minimizing the Euclidean norm $\|w\|^2$. The regularized constant C determines the trade-off between the empirical risk and the regularization term, which means the trade-off between the flatness of the function f and the amount up to which the deviations larger than ϵ are tolerated. In other words, the errors are ignored by the SVR algorithm as long as they are less than ϵ, but any deviation larger than this is not accepted. It is important to notice that both ϵ and C are both user defined constants and are typically computed empirically. It is implicitly assumed in Equation 2.21 that a function $f(P)$ actually exists and approximates all pairs (P_k, L_k) with ϵ precision (tube size), which means that the optimization problem is feasible. However, to make the optimization problem feasible errors may have to be accepted. Therefore, slack variables ξ_i and ξ_i^* are typically introduced to measure the deviation of training samples outside ϵ-insensitive zone to account for errors. To obtain the estimations of w and b, Equation 2.20 is transformed into a primal function stated in Vapnik (Vapnik, 1995) as follows:
Minimize

$$C\sum_{i=1}^{n}(\xi_i + \xi_i^*) + \frac{1}{2}\|w\|^2, \tag{2.22}$$

subjected to

$$\begin{cases} L(i) - w\phi(P_i) - b_i \leq \epsilon + \xi_i, \\ w\phi(P_i) + b_i - L(i) \leq \epsilon + \xi_i^*, \\ \xi_i, \xi_i^* \geq 0 \end{cases} \tag{2.23}$$

By introducing Lagrange multipliers and forming the dual optimization problem, the decision function given by Equation 2.20 has the following explicit form (Vapnik, 1995):

$$f(P, a_i, a_i^*) = \sum_{i=1}^{n}(a_i - a_i^*) + K(P, P_i) + b. \tag{2.24}$$

The data points on or outside the e ϵ-tube with non-zero Lagrange multipliers are defined as support vectors. The optimal weights w having non-zero Lagrange multipliers are typically less than the entire data set, thus, the entire data set is not need to define $f(P)$. The sparseness of this solution is one of several advantages of using this methodology. $K(P, P_i)$ is defined as the kernel function and it computes the inner product of two vectors P and P_i in the feature space $\phi(P)$ and $\phi(P_i)$ by:

$$K(P, P_i) = \phi(x) \cdot \phi(P_i) \tag{2.25}$$

The kernel function provides a solution to map the input space P into a high dimensional feature space ($\phi(P)$) to perform the non-linear regression using SVR. There are several kernel functions that satisfy Mercer's conditions (Vapnik, 1995) such as Gaussian, polynomial, and hyperbolic tangent. In SVM/SVR applications, the choice of the kernels is a key factor. Although the Gaussian kernels appears to be the most prevalent choice, typically empirical analyses is necessary in the selection of the appropriate kernel function according with the nature of the application. Finally, the resulting SVR architecture is given below in Figure 2.9.

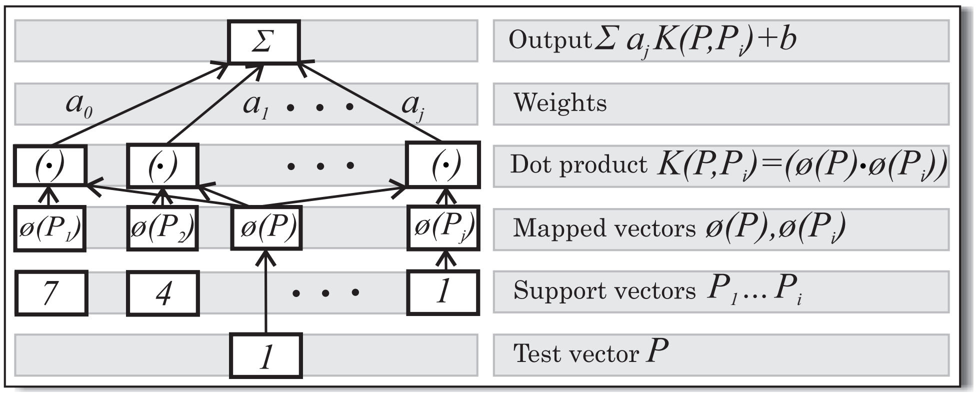


Figure 2.9. Architecture of a regression machine resulting by the SVR algorithm (Sapankevych and Sankar, 2009; Smola and Schölkopf, 2004).

Learning techniques have been applied for target tracking in diverse WSNs scenarios using kernel-based learning (Simic, 2003) and support vector machines (Tran and Nguyen, 2008; Viani et al., 2010). A distributed SVM training was proposed in (Kim et al., 2012) to solve a multi-target tracking problem in WSNs. After training the local SVM at each node, this approach computes the posterior probability of the existence of the targets using Platt's optimization algorithm. By maximum a posterior (MAP), the target trajectories are estimated. In order to overcome challenges such as limited communication and the curse of dimensionality when applying Machine Learning algorithms such as SVR on large-scale WSNs, the authors in Kim et al.(Kim et al., 2013) proposed an ensemble implementation of SVR for the problem of target localization. Experimental results achieved in this work indicate that the performance SVR proposed method provides good prediction accuracy. Also, the performance comparison has shown that the SVR proposed method outperforms the classic SVR predictor in terms of accuracy and robustness for large scale WSNs.

2.4.3 Filtering

The main goal of target tracking systems is to continuously detect and estimate the state of a target or a set of targets. Besides the location information, target tracking can be used to detect and predict future locations of single or multiple targets such as other vehicles, objects and obstacles surrounding a given vehicle (Ramos et al., 2012; Li and Jilkov, 2003; Schubert et al., 2008). It is important to notice that these algorithms are exposed to different sources of noise, introduced by the measurement process and also errors in nodes' location that are used to estimate the target coordinates. Therefore, information fusion (Nakamura et al., 2007) is commonly used for filtering such noise sources.

The targets' state can include, among other information, position, velocity, acceleration, and jerk (derivative of acceleration) and this set of state variables can also vary according to the application requirements and constraints. According to Ramos et al.(Ramos et al., 2012), target tracking systems typically rely on a Bayesian motion estimation framework that require: a motion model that describes the target's dynamic; samples of the target's state; a data association algorithm that takes into account the samples to the correct target; and an initial probability distribution, also known as prior knowledge of the target's state. According to the motion model, the main task performed by the tracking systems is to estimate the parameters of the model, considering the measurements collected about the target.

The filtering component of target a tracking system is responsible for defining how the probability density function (pdf) of the target's state at time step t is computed. Based on these components, the target tracking system has two phases (as depicted in Figure 2.10): prediction, which uses the motion model to propagate the probability function of the target state over the time; correction, which uses the latest collected samples to update the pdf of the target at the current time step. This task is usually performed by a Bayesian filter, such as the Kalman filter and the Particle filter.

2.4.3.1 Kalman Filter

The Kalman Filter (KF) was originally proposed in 1960 by Kalman (Kalman, 1960) and it is a popular Data Fusion method used to fuse low-level redundant data (Nakamura et al., 2007; S. et al., 2009). The KF presents some interesting properties since it can recursively retrieve statistically optimal estimates when the noise is Gaussian and, is the linear optimal estimator even when the noises are not Gaussian

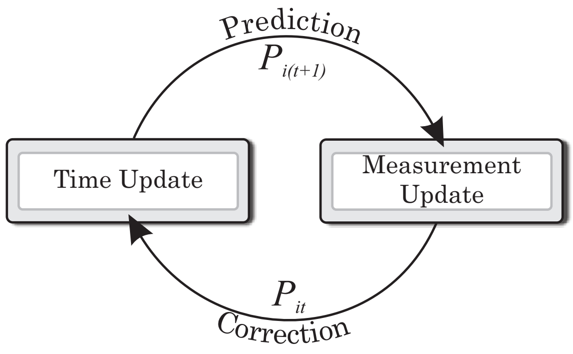


Figure 2.10. Filtering phases.

(Simon, 2006). In other words, the KF is the optimal filter in terms of unbiased minimum variance state estimation when the system can be described as a linear model with Gaussian noise.

The KF applies a linear operator in the current state to generate a new state at each discrete time increment. Besides the measurement noise, the filter can also optionally consider some information about the controls on the system. Then, another linear operator, also subject to noise, generates the observed outputs from the true state (Hossain et al., 2009). The KF estimates the state p of a discrete-time t controlled process that is ruled by the state-space model

$$p_{t+1} = F_t p_t + B_t u_t + w_t \tag{2.26}$$

with measurements (observation) o at time t of a state p_t made according to:

$$o_t = H_t p_t + b_t, \tag{2.27}$$

in which F_t is the state transition matrix applied to the previous state p_t, B_t is the input control matrix model that is applied to control vector u; H_t is the measurement matrix (the observation model), which maps the true state space into the observed space; w is the process noise; and b the measurement noise, where these noise sources are assumed to be drawn by random zero-mean Gaussian variables with covariance matrices Q_t and R_t, respectively.

Based on the measurement o_t and the knowledge of the system parameters, the estimate of p_t, represented by $\hat{p}_t$, and the prediction of the next state p_{t+1}, represented by $\hat{p}_{t+1|t}$ are given by:

$$\hat{p}_t = \hat{p}_{t|t-1} + K_t(o_t - H_t \hat{p}_{t|t-1}), \tag{2.28}$$

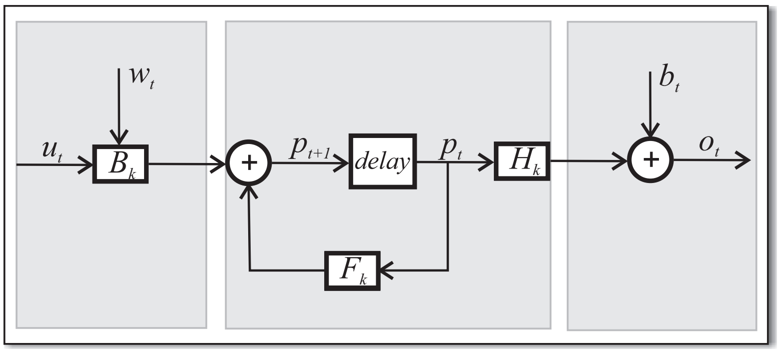


Figure 2.11. Block diagram of the Kalman Filter

$$\hat{p}_{t+1|t} = F_t \hat{p}_t + B_t u_t, \tag{2.29}$$

in which K_t is the Kalman gain determined by

$$K_t = P_{t|t-1} H_t^T (H_t P_{t|t-1} H_t^T + R_t)^{-1}, \tag{2.30}$$

where P_t is the prediction covariance matrix that can be determined by

$$P_{t+1|t} = F_t P_t F_t^T + Q_t, \tag{2.31}$$

with

$$P_t = (I - K_t H_t) P_{t|t-1}, \tag{2.32}$$

where I is the identity matrix.

The Kalman Filter has two phases: time-update (predict) and measurement-update (correct). The time-update phase is responsible for projecting the current state and error covariance estimates forward, obtaining the a priori estimates for the next time step and consists of the Equations (2.26) and (2.27). The measurement-update phase is responsible for the feedback, that is, a new measurement at current time step is incorporated into the a priori estimate to obtain an improved a posteriori estimate. This phase consists of the Equations (2.29), (2.30), and (2.31) (Nakamura et al., 2007). These predict and correct phases form a loop that is performed while the filter is fed by measurements.

The Kalman filter theory applies to linear-Gaussian problems, but many real world problems cannot be represented by linear models, algorithms have emerged based on the original Kalman Filter theory to deal with nonlinear dynamics and non-linear measurement models (Daum, 2005). Variations of the Kalman Filter have also been proposed for relaxing the non-linearity assumption of samples. The Extended Kalman Filter (EKF) (Welch and Bishop, 2001) is a popular technique to deal with non-linear models. The main idea of the EKF is that the state distribution is approximated by a Gaussian law, and this method uses a linearized model of the process using Taylor series, because this is a sub-optimal estimator. Another recent variation of the Kalman filter is the Unscented Kalman Filter (UKF) and represents a great improvement over EKF (Julier and Uhlmann, 1997). The UKF performs estimations on non-linear systems without the need to linearize them, because it uses the principle that a set of discrete sampling points can be used to parameterize the mean and covariance. UKF is known to greatly improve the performance for linear systems when compared to EKF, because it does not have to deal with linearization errors. However, the quality of UKF estimates are close to standard KF for linear systems.

Several tracking solutions are based on Kalman Filters (KF) (Li et al., 2006; Welch and Bishop, 2001; Julier and Uhlmann, 1997; Olfati-Saber, 2005). In VANets, Armaghan et al. (Armaghan et al., 2009) proposed an estimation method based on Kalman filter to decrease the number of transmitted messages. In this method, each vehicle estimates its location for several intervals and sends them out along with actual current position. The estimation is done based on the previous history and record of the vehicle's location. During the time that estimated information is available, there are no transmissions unless some estimation error is detected. In Mo et al. (Mo et al., 2008), the authors presented a location management protocol called Mobility-Assisted Location Management (MALM), to provide location service to vehicles in VANets . In MALM, a vehicle calculates the current location of other vehicles by using Kalman filtering based on the historical location information of other nodes. In Lytrivis et al. (Lytrivis et al., 2011), was proposed the cooperative path prediction algorithm for safety applications in VANets . It considers position, velocity, acceleration, heading and yaw rate measures to create beacons containing dynamic status of a transmitting vehicle. The algorithm uses UKF for predicting both short-distance and short-term for targets within the sensing range of the ego vehicle.

Aiming at improving security on the roads, Ammoun et al. (Ammoun et al., 2007) uses a Kalman filter for trajectory prediction and the estimation of a vehicle's location to evaluate and anticipate the risk of collision at a crossroad. The authors show that despite unavoidable latencies and positioning errors, the application performance is still acceptable when a Kalman filter is used for trajectory prediction and estimation. In Najjar and Bonnifait (E. Najjar and Bonnifait, 2005), Belief Theory and Kalman filters are used to provide accurate position estimations for a vehicle relative to a digital road map. In this method, the Kalman Filter is used to combine the Antilock

Braking Systems (ABS) measurements with a GPS position, which is then used to select the most credible roads. The selection strategy fuses distance, direction, and velocity measurements using Belief Theory. A new observation is then built and the vehicle's approximate location is adjusted by a second Kalman filter

2.4.3.2 Particle Filter

Particle Filter (PF) is a filtering technique that relies on a brute-force approach to estimates the target's state through a recursive implementation of Sequential Monte Carlo method (SMC) (Doucet et al., 2001). The Bootstrap Filter was the first PF method proposed in 1993 by Gordon et al. (Gordon et al., 1993). PF can deal with non-linearity and with non-Gaussian noise when Kalman filter approaches do not perform well. Unlike of the linear/Gaussian problems, the computation of the posterior distribution of non-linear/non-Gaussian problems are extremely complex (Ramos et al., 2012). To overcome this difficulty, the Particle Filter adopts an approach called sampling importance. The key idea is to represent the posterior pdf based on a large number of random samples, called particles, which are sequentially propagated over time (Arulampalam et al., 2002). At each time step, some particles that present low posterior probability are discarded by a process called resampling. To each particle is associated a weight indicating its quality, thus, the estimate is the result of the weighted sum of all particles (Nakamura et al., 2007).

As the Kalman Filter, the Particle Filter algorithm has two phases: prediction and correction. In the prediction phase, each particle is modified according to the existing model, including the addition of random noise in order to simulate the effect of noise. Then, in the correction phase, the weight of each particle is reevaluated based on the latest sensory information available, so that particles with small weights are eliminated (resampling process). The resampling step is the solution adopted to avoid the degeneration problem, where the particles have negligible weights after several iterations. The particles of greater weight are selected and serve as the basis for the creation of the new particles set. Furthermore, the minor particles disappear and do not originate descendants. For illustration purposes, the Particle Filter algorithm presented in Algorithm 1 (Souza et al., 2013) considers only one dimension, in which P is the position, S is the velocity and w is the weight of each n particles in a discrete-time t; the o variable is the input measurement value (observation). However, this algorithm can easily be applied to coordinate systems in $\mathbb{R}^3$. First, the algorithm randomly distributes the particles (line 2). The particle propagation and the calculus of their importance consider the distance from each particle to the measurement position (lines 5-11). The

Algorithm 1 The Particle Filter Algorithm

```
▷ Input:
 1: The measured $o_t$
 Action:
 2: for $i = 1 : N$ do {FOR: Initialize the particles}
 3:     $P_0^i \leftarrow random();$
 4: end for
 5: $totalWeight \leftarrow 0;$
 6: for $i = 1 : n$ do {FOR: Sample particles and compute weights}
 7:     $P_t^i \leftarrow P_{t-1}^i + S_{t-1}^i + gaussian();$
 8:     $S_t^i \leftarrow S_{t-1}^i + gaussian() * 0.05;$
 9:     $w_t^i \leftarrow 1/distance(o_t, P_t^i);$
10:     $totalWeight \leftarrow totalWeight + w_t^i;$
11: end for
12: for $i = 1 : n$ do {FOR: Normalize weights}
13:     $w_t^i \leftarrow w_t^i/totalWeight;$
14: end for
15: $slice_t^0 \leftarrow w_t^0;$
16: for $i = 2 : n$ do
17:     $slice_t^i \leftarrow slice_t^{i-1} + w_t^i$
18: end for
19: for $i = 1 : n$ do {FOR: Ressampling}
20:     $c \leftarrow random();$
21:     $j \leftarrow 0;$
22:     while $j < n - 1 andslice_t^j < c$ do
23:         $j \leftarrow j + 1;$
24:     end while
25:     $resampling_t^i \leftarrow particle_t^j;$
26: end for
27: for $i = 1 : N$ do {FOR: Compute the prediction $x_{k+1}$}
28:     $P_{t+1} \leftarrow P_{t+1} + P_t^i * w_t^i;$
29: end for
30: return $P_{t+1};$
```

normalization process (line 12) prepares the particles weights for the resampling process (lines 15-26). Finally, the prediction of the position is computed (lines 27-29).

Particle Filters are popular for modeling non-linear systems subject to non-Gaussian noise in wireless communication. There are several tracking solutions based on Particle Filters for sensor networks (Vercauteren et al., 2005; Arulampalam et al., 2002; Rosencrantz et al., 2003; Jiang and Ravindran, 2011). In VANets, an interesting study on the suitability of mobility prediction to reduce excessive beaconing to sensitive cooperative safety applications is presented in (Hrizi et al., 2012). The authors discuss the challenges regarding the trade-off in periodically transmitted packets leading to wireless congestion. While adapting the rate of the transmission to some predicted motions impacts the accuracy of this knowledge and the reliability of the cooperative safety application. They extend a Particle Filter to take into consideration VANets peculiarities. The authors showed that the proposed solution can ensure a suitable adaptation of the channel load with a high precision of awareness prediction to traffic safety applications.

In Peker et al. (Peker et al., 2011), was presented an algorithm for vehicle localization and map-matching using PF. The probability of being on a certain area of the digital map according to vehicle speed is used in conjunction with routing information to augment the likelihood function in the weight computation step of the particle filter. The authors performed real life tests and the results achieved show a considerable increase in correctness of Map-Matching and localization accuracy. The proposed algorithm also guides Dead Reckoning when GPS data is unavailable. In Fernandez-Madrigal et al. (Fernandez-Madrigal et al., 2007), the authors use Particle Filters to cope with vehicle localization in combined indoor and outdoor scenarios. In such scenarios, the authors assess the performance of ultra-wide band sensor technology for indoor positioning and GPS for outdoor areas. They also evaluated the use of PF to fuse observations collected from these two types of sensors for vehicle localization. Particle Filters are also used in Chausse et al.(Chausse et al., 2005) to combine GPS localization with data extracted from vision systems to determine a vehicle's location on the road. The combined information is transformed into a global reference using a Map-Matching thechnique.

2.5 Techniques Discussion

It is well known that the moving of vehicle in a city is a dynamic process, including static process (traffic light), which is strong nonlinear. These non-linear characteristics of VANets can severely affect the performance of the predictor algorithm. Table 2.1 briefly compares these techniques in terms of advantages and challenges whereas Table 2.2 briefly compares them in terms of training and accuracy.

The main advantage of DR rely on its good accuracy for predictions when the vehicles have a linear mobility pattern with a fast initial convergence. DR is able to achieve accurate predictions when computing a future position only based on the last know vehicle position. However, its performance is decreased with the randomness of non-linear mobility and it is also subject for cumulative errors, especially when the current position of a vehicle is not provided by a Data Fusion approach. One of the most appealing advantages of DR relies on its simplicity: the algorithm has a low computational cost, it is easy to implement, requires low processing power and memory usage, which sufficiently match the capabilities of vehicles computational devices.

Regarding the time series prediction aspect of localization prediction in VANets, the main advantage of considering NNs and SVRs as approaches is the non-linear

Method	*Advantages*	*Challenges*
DR	Computationally efficient by design Easy to implement No free parameters	Subjected to cumulative errors
NN	Not model dependent Not dependent on linear, stationary processes Can be computationally efficient (feed forward process)	Number of free parameters large Selection of free parameters usually calculated empirically Not guaranteed to converge to optimal solution Can be computationally expensive (training process)
SVR	Not model dependent Not dependent on linear, stationary processes Guaranteed to converge to optimal solution Small number of free parameters Can be computationally efficient	Selection of free parameters usually calculated empirically Can be computationally expensive (training process) Trade-off between accuracy and computational effort
KF	Computationally efficient by design Convergence guaranteed Minimizes mean square error by design Small number of free parameters	Assumes linear, stationary processes Assumes process model is known
PF	Not model dependent Not dependent on linear, stationary processes Small number of free parameters Can deal with non-Gaussian noise	Curse of dimensionality Requires a large number of particles to present accurate results

Table 2.1. Trajectory prediction techniques: practical aspects (Sapankevych and Sankar, 2009; Skog and Handel, 2009; Daum, 2005).

aspect of the prediction problem. In this case, neural network models have the advantage of allowing the approximation of complicated non-linearities which could not be well modeled by other classical models (Ibnkahla, 2000). NNs are self-adaptive, data-driven that do not require any a priori assumptions about the problem space, not even information about the statistical distribution. In fact, NNs are universal function approximators and it has been demonstrated that they can approximate any continuous function to any desired accuracy (Irie and Miyake, 1988; Hornik et al., 1989). Considering the VANets' mobility characteristics, NNs are well suited since they can represent knowledge that is difficult to specify but, in which, there are enough data or observation about the problem. In terms of localization systems, NNs are also attractive since they can generalize and can often correctly infer the unseen part of data even if the data sample contains noise.

The non-linear aspect of localization prediction in VANets is shared with many real-world applications. According to Sapankevych and Sankar (Sapankevych and Sankar, 2009), traditional (and more sophisticated) model-based techniques generally do not perform as well as the SVR in predicting time series gen-

Method	*Training*	*Prediction Accuracy*
DR	Not required Fast initial convergence	Good for short time sample horizon Accurate for linear mobility pattern
NN	Required	Good short time horizon Can detect driver patterns in long time horizon Accurated for non-linear mobility pattern
SVR	Required Can be computationally expensive	Good short time samples' horizon Can detect driver patterns in long time samples' horizon
KF	Not required Fast initial convergence	Good for short time sample horizon Provides the linear MSE solution to the filtering problem
PF	Required Convergence of initial distributions	Good short time horizon Accurated for non-linear mobility pattern

Table 2.2. Trajectory prediction techniques: the computational aspect (Sapankevych and Sankar, 2009; Skog and Handel, 2009; Daum, 2005).

erated from non-linear systems. This is based on the fact that the Machine Learning techniques like SVR and NNs lets the data speak for itself whereas the model-based techniques typically cannot model the non-linear processes well. Consequently, these techniques are less susceptible to the problem of model misspecification as compared to most of the parametric models. The main advantage of the SVRs when compared to the traditional model-based techniques rely on the fact that, by design, the SVR guarantees a global minimum solution and is typically superior in the ability to generalize. For the localization prediction problem in VANets, in theory it means that SVRs by design can be superior in terms of the accuracy in the localization prediction. However, there is a tradeoff in terms of the computational effort required to achieve such accurate results. This tradeoff can also affect the NNs, but it affects more the SVRs due to the computational effort to solve the global minimum solution problem. In theory, these machine learning techniques can lead to very accurate localization prediction. However, due to the real-time characteristic of VANets applications, this accuracy can be reduced by the computational time required to compute the predictions in a feasible time for its use.

Another important issue in the application of the NNs and SVRs in VANets is the free parameter selection and the required training, which can be computationally expensive. This is not a specific VANet issue, but an issue observed in many real world application of such techniques. Regarding the parameter selection, which is more challenging in the case of NNs due to its large number, there are several proposed approaches, however, most of them are usually quite complex and difficult to implement. Also, due to fact that streets in VANets can vary from a large number of geometric shapes, such parameters guidelines can be simply guided by heuristics, simulations or also by experiments on the target area. In this case, the main compromise is to balance between computational complexity, robustness against modeling errors, and accuracy of the algorithm (Skog and Handel, 2009). Regarding the training of the Machine Learning methods, large window size (number of collected location samples) can increase the complexity of the convergence procedure and resulting in long training time, which is not suitable for real-time implementations. In this case, smaller training window sizes are quite preferable. However, since the main advantages of such approaches rely on the ability to generalize, a long term training approach is also feasible when associated with an optimization process. For instance, such algorithm can be trained on data collected over a large time horizon (daily, weekly, or monthly) and the driver's common routes, based on its routine (e.g. routes to work and home), can be easily identified improving the accuracy of the long-term predictions.

A recurrent issue for Machine Learning and Bayesian Filtering approaches is the curse of dimensionality. That is, the computational complexity of the predictor method usually grows exponentially with the dimension of the state vector being estimated (Skog and Handel, 2009; Daum, 2005). Therefore, even vehicles equipped with high computational capacity, non-linear filters and Machine Learning algorithms can be unfeasible for navigation systems with high-dimensional state vectors. In this case, the introduction of sensor's that can provide information about a vehicle's state measurements like wheel odometers, magnetometers, accelerometers, etc., can improve the predictor's method accuracy. However, its computational complexity will also grow exponentially. An efficient approach in this case would be to consider only position information in the state vector of the predictor. Since it is quite common the use of computed position of vehicles in $\mathbb{R}^3$, the problem of the curse of dimensionality can be easily avoided in VANets. Besides, such sensor's data can be more efficiently processed in the vehicle's current position estimation using Data Fusion methods (Nakamura et al., 2007; Boukerche et al., 2008).

In the Bayesian Filtering approach, the model must be complete enough to give an adequate description of the system and, at the same time, be sufficiently simple for the filtering algorithm to become computationally feasible (Griffin and Sage, 1969). Such assumptions are satisfied in VANets since Data Fusion methods can provide a complete mobility model and vehicles can be coupled with reasonable computational units. The Minimum Mean Square Error (MSE) solution to the linear problem is then provided by the Kalman filter, assuming Gaussian distributed noise sources (Skog and Handel, 2009; Kailath, 1998). In terms of VANets, the Kalman filter is the optimal choice when the system is linear with Gaussian noise (Ramos et al., 2012). In this case, the Kalman filter is an attractive approach for VANets scenarios since the linear characteristic of streets, especially in grid models, can be described as a linear vehicle movement model and also the localization error usually is composed by Gaussian noise in the average. KFs are relatively easy to design and code, and they often provide good prediction accuracy. One advantage of KF when compared to Machine Learning methods is the fast initial convergence of the predictions without requiring training. On the other hand, KF accuracy can be surprisingly bad for some practical applications when the physical system is described by non-linear equations or the model is inaccurate or incomplete (Daum, 2005).

Particle Filter can outperform the KF especially for the non-linear case, with the cost of additional computational effort, because it typically requires a large number of particles to provide accurate results (Ramos et al., 2012). Therefore, in systems with a highly non-linear nature and non-Gaussian noise sources PF can keep the non-linear structure of the problem, significantly improving the system performance (van der Merwe et al., 2004; Daum, 2005). However, since the navigation equations in VANets are only partial non-linear, the localization prediction problem can be divided into a linear part and a nonlinear part (Skog and Handel, 2009). In this case, under the assumption of Gaussian-distributed noise sources, the linear case may be solved using a KF, hence, reducing the computational complexity (Schön et al., 2005; Karlsson et al., 2005) and also increasing the accuracy of the localization system. PF are also relatively easy to design and code and works well for a high range of localization problems. The accuracy of PF's approximation is determined by the size of the particle set. In this case, increasing the number of particles also increases the accuracy of the predictor. However, it also increases the computational cost of the localization system. In other words, the number of particles is a trade-off between the accuracy and available computational resources (Golestan et al., 2012). Also, regarding the computational cost, the initial distribution of the particles requires additional time to converge in accurate predictions when compared to the Kalman Filter.

Although there are several promising approaches to tackle the problem of localization prediction in wireless networks, based on a theoretical analysis, Dead Reckoning, Machine Learning, and Bayesian Filtering are efficient and feasible approaches to be applied to the context of vehicular network. In short, all of these techniques present advantage to be considered in the different VANets scenarios. Dead Reckoning has the advantage of providing good accuracy with a lower computational cost for short time predictions subjected to low levels of localization noise sources. Machine Learning can provide accurately prediction estimations for the non-linear case but suffer from the computational complexity required. However, NNs and SVR are able to generalize patterns in the data, thus, they are able to discover the driver's common habits, such as daily routes for long-term predictions. Therefore, such in advance information can be used to improve both localization systems and many other VANets services that can take advantage of long-term localization information. For non-Gaussian noise sources and the linear case, the Kalman Filter provides the linear MSE solution to the filtering problem. For highly non-linear nature mobility models and non-Gaussian noise sources, Particle Filters keep the non-linear structure of the problem significantly improving the localization prediction accuracy.

Finally, even though these analyzed approaches tackles different processes and measurement models, an interesting alternative to improve VANets services through localization prediction can be the combination, in a single solution, of two or more of these solutions to deal with Gaussian/non-Gaussian noise and linear/non-linear models. In the next section, we evaluate the performance of the localization prediction techniques discussed in this work.

2.6 Performance Evaluation

2.6.1 Proposed Approach

In our approach, during the localization process, we assume that each vehicle i periodically observes its current position (P_{it}) at a step time t. Based on the knowledge of the $t-1$ steps, the prediction of a vehicle's future position is given by target state estimate $P_{i(t+1)}$, which will estimate the future position $(X_{i(t+1)}, Y_{i(t+1)}, Z_{i(t+1)})$ for the next time step $t+1$. Regarding the Dead Reckoning approach, the future position prediction of each vehicle is made by computing the coordinates $X_{i(t+1)}, Y_{i(t+1)}$ and $Z_{i(t+1)}$ by using equations 2.6 and 2.7.

The parameters of the machine Learning algorithms have been adjusted through plenty of simulation experiments. We aimed to obtain the best accuracy (less error rate and suitable computational effort). For the NN and SVR, the input vector is composed of $(X_{it}, Y_{it}, Z_{it}, s_t)$, where X_{it}, Y_{it} and Z_{it} are the coordinates of the vehicle's current location and S_t its current displacement speed. Thus, at each time step t an input vector $(X_{it}, Y_{it}, Z_{it}, S_t)$ is added to the set of training data along with the $t-1$ inputs. For each vehicle, the training is performed on the last $t-1$ training inputs. Since during the initial experiments we observed that the SVR required a huge additional time on training, we limited the training of this method to a window of $t-13$ inputs. This decreasing on the training size of the SVR was justified to achieve a good accuracy in a suitable response time. Also, we noticed the trade-off in the free parameter selection and the computational effort for the NN and SVR. During the initial experiments, when applying these methods, the changes in the parameters values decreased the localization error of the predictions to considerable small values. However, when decreasing the error, the computational time required for computing the predictions increased in the inverse proportion. Thus, the heuristic adopted in the parameters' selection and size of training data was the compromise of keeping the error rate in lower levels and, at the same time, keep the time required to compute each prediction feasible to the use of the prediction in a real application.

The NN used in this work is composed of three layers. The input layer is composed of four neurons to map the input vector of coordinates and speed of the vehicle. By the initial experiments, we noticed that, to achieve the lower error rate and suitable training time, the most suitable number of neurons for the hidden layer was 1100. We also noticed the same behavior of NNs discussed in Kaaniche and Kamoun (Kaaniche and Kamoun, 2010): variation in the number of neurons of the hidden layer can affect the prediction accuracy. The output layer has three neurons, corresponding to the coordinates of the predicted future position in $\in \mathbb{R}^3$. We use the tangent hyperbolic activation function for all neurons, since it provide a faster converge to the learning algorithm (Bishop, 1995). The tangent hyperbolic activation function is given by:

$$\phi(\xi) = tanh(\xi) = \frac{exp(\xi) - exp(-\xi)}{exp(\xi) + exp(-\xi)}. \tag{2.33}$$

Since this function outputs values that range between $[-1, 1]$, to perform the regression over the time series localization values, it is necessary to scale the values of the coordinates also between $[-1, 1]$ using a scale factor. The training of the MLP algorithm is performed in 400 epochs before each prediction. We also utilized a momentum value 0.89, and an adaptive learning rate with initial value 0.5 divided by the activation function scale factor. The momentum parameter is used to prevent the system from converging to a local minimum and the learning rate specifies how fast the model adjusts itself to new case.

For the SVR approach, the key parameters that control the complexity of the model are ϵ, C, and the kernel function. The ϵ parameter was set to 0.15 and we noticed in our initial experiments that, decreasing this parameter leads to more accurate results, however, the computational time increases in the inverse proportion. Since the training of the SVR was made with a window of the last 13 past observations, the parameter C was set to 1300 encouraging exact fitting of the predictions. Regarding the kernel function, we used the radial basis kernel defined as:

$$K(P_i, P_j) = expr(\frac{-||P_i - P_j||^2}{2\sigma^2}), \tag{2.34}$$

since the similarity of two examples is simply judged by their Euclidean distance. The parameter σ of the radial basis kernel determines the area of influence in which the computed support vectors have over the data space and it is defined as 1.0 divided by mean squared distance between the sample points training data.

The filtering approach is performed with Kalman or Particle filters. The Kalman Filter has its linear system equations represented by:

$$\begin{cases} P_{t+1} = \begin{bmatrix} X_{i(t+1)} \\ Y_{i(t+1)} \\ S_{iX(t+1)} \\ S_{iY(t+1)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{it} \\ Y_{it} \\ S_{iXt} \\ S_{iYt} \end{bmatrix} + w_k \\ o_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} X_{it} \\ Y_{it} \\ S_{iXt} \\ S_{iYt} \end{bmatrix} + b_t \end{cases} \tag{2.35}$$

in which P represents the state of a discrete-time t, composed by the position (X_i, Y_i) and velocity (S_{iX}, S_{iY}); o is a measurement value; w and b represent the process and measurement noise, respectively. For illustration purposes, the KF linear equations are described $\mathbb{R}^2$. However, they can easily be applied to coordinate systems in $\mathbb{R}^3$. It is important to notice that the KF used in this work does not have parameters to be adjusted and, since this approach can fast converge to a good initial accuracy, there was no need of prior simulation experiments and also training to obtain the best estimation results for this method.

The Particle Filter used in the experiments is represented by the Algorithm 1. The Particle Filter uses 1000 particles. This value was set based on some previous empirical tests that showed that more than 1000 particles do not improve the predictions significantly. It is important to notice that the PF does not require training. However, during the initial experiments we observed that the PF required a huge initial time for convergence of initial distributions to start computing accurate predictions.

Parameter	*Value*
Simulation area	21600 m×23728 m
Number of vehicles	33
Measurements Interval	1.5 s
Number of Training Samples	300
Number of Test Samples	300
Localization error	0.5 m

Table 2.3. Localization prediction simulation parameters.

2.6.2 Methodology

The performance evaluation is performed through simulations using the NS-2 and the default values are shown in Table 2.3. In all of the results, curves represent average values, while error bars represent confidence intervals for 95% of confidence from 33 independent instances (33 different vehicles moving during the simulation). The performance of the localization prediction methods presented in this work is performed in a scenario where the vehicles' mobility is simulated through a set of realistic vehicular traffic data (Uppoor and Fiore, 2011). This data set (depicted in Figure 2.12) is based on information provided by the TAPASCologne project (TAPASCologne, 2014), an initiative by the Institute of Transportation Systems at the German Aerospace Center (ITS-DLR). This project aims at reproducing, with the highest level of realism possible, vehicular traffic in the greater urban area of the city of Köln, in Germany.

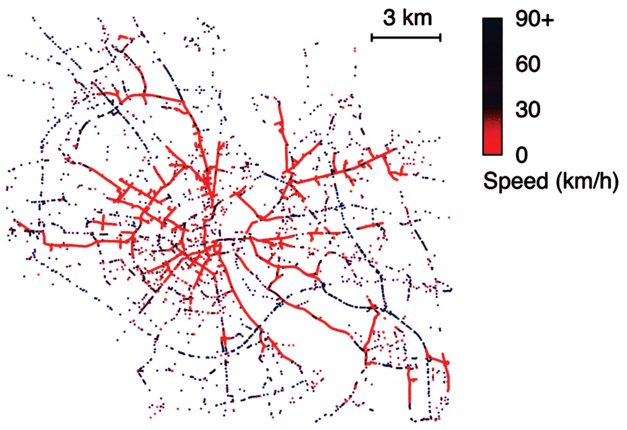


Figure 2.12. Snapshot of the TAPASCologne dataset traffic status at 7:00 am, in a 400 km^2 region of the city Köln (Uppoor and Fiore, 2011).

The evaluation methodology is divided into two phases. In the first phase, according to the values of the data set of vehicular traffic, the simulation is started in time 21600 s. After that time, according to the measurements' interval value (step time t to perform the prediction), each vehicle uses a number of 300 location samples to calibrate the filter algorithms and train the machine learning methods. In the second phase, each algorithm perform the predictions of the future positions during an interval of 300 consecutive localization samples, also according to the measurements time interval. To simulate position computation inaccuracies, we introduced errors on the computed position of the vehicles by using a Gaussian distribution with mean equal to the actual position of the vehicle and a standard deviation of 0.5 m (Langendoen and Reijers, 2003).

2.6.3 Simulation Results

Regarding the benefits of localization prediction in several applications for VANets, we evaluate in this work two scenarios that can severely influence the choice of the prediction algorithm. The first scenario refers to the granularity of the location information in terms of time. In this context, there are several types of VANet application that require localization information in different periodicity. For instance, real-time and non-real-time application. Thus, the choice of the prediction algorithm can also be influenced according with its behavior regarding the level of periodicity, in terms of required time granularity of location information. In the second scenario, we evaluate the impact of localization errors in the vehicle's computed position, since minimizing these errors is the main goal of a localization system. Therefore, we evaluate how these errors can affect the accuracy of the prediction algorithms. Since several applications of VANets differ on the localization accuracy required in order to be able to function properly (Boukerche et al., 2008), the same principle can be extended to localization prediction in terms of the choice of the prediction algorithm.

2.6.4 The Impact of the Measurements Interval

To evaluate the impact of the measurements' interval, we increase this parameter from 0.5 s to 2 s. Thus, when increasing the step time t of the samples, we also increase the total distance traveled by the vehicles from 2465 m to 11174 m. As depicted in Figure 2.13(a), the DR and the KF lead to a small error in the distance between the predicted location and the real future location. While increasing the step time of the localization samples, we can notice that the PF, NN, and the SVR leads to a lower accuracy on the predictions. In this result, the small error on the predicted distance achieved by the DR and the PF is explained by the Gaussian nature of the introduced localization errors and also by the fact that, in the average, the localization samples in our studied scenario has a strong linear characteristic. In Figure 2.13(b), we evaluate the MSE average of the prediction algorithms. The MSE is arguably the most important criterion used to evaluate the performance of a predictor. The MSE assesses the quality of an estimator in terms of its variation and degree of bias. In this result, we can notice that the DR and the PF also lead to a small MSE, thus resulting in more accurate predictions. Also in Figure 2.13(b), we can notice the disadvantage of PF, NN and the SVR with a high increase in the MSE while increasing the measurements' interval. In this case, these algorithms present a high error in the prediction due to the Gaussian nature of the noise and the linear average of the samples, which affects more the accuracy of these algorithms since their best performance is related to the non-linear case.

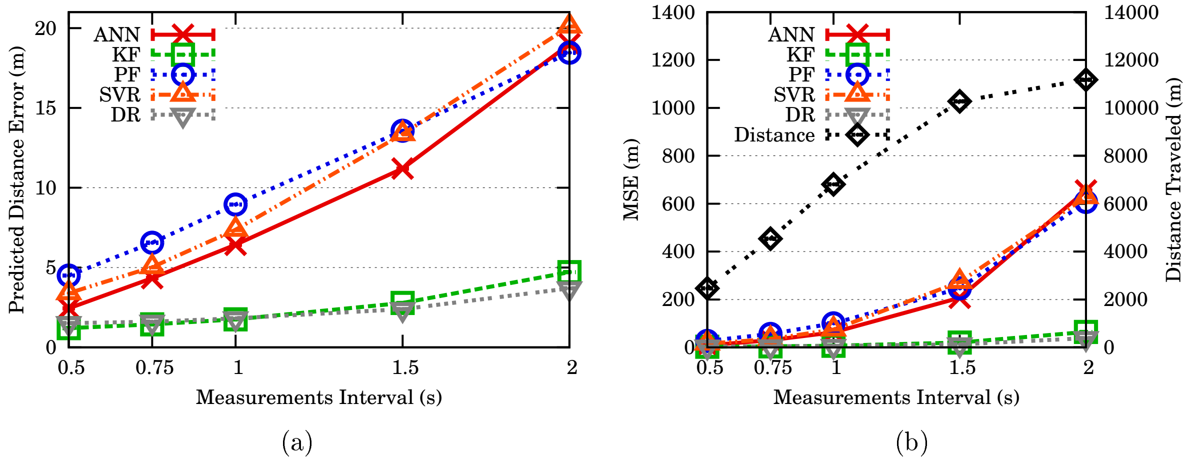


Figure 2.13. Impact of the measurements' interval in terms of distance.

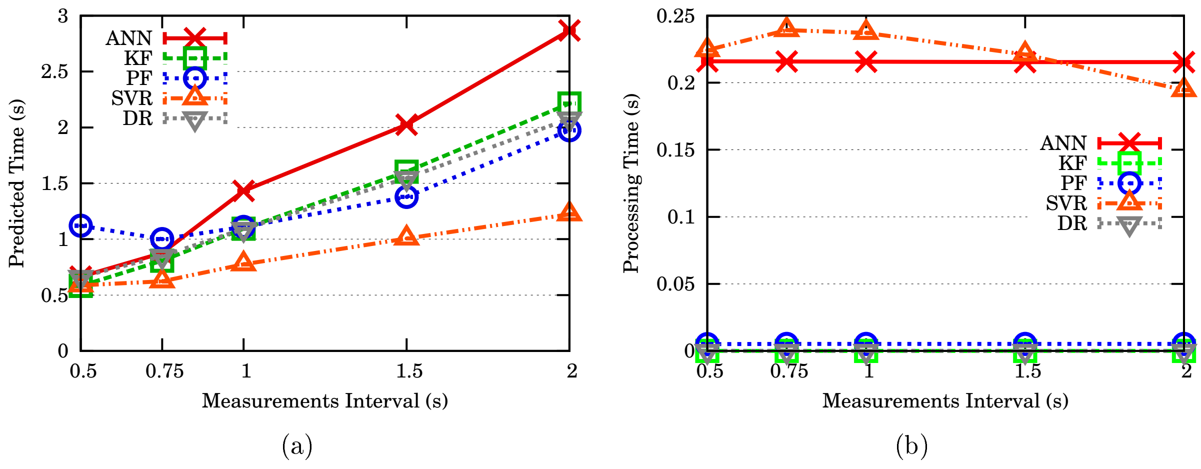


Figure 2.14. Impact of the measurements' interval in terms of time.

We also evaluated the predicted time average, the time average required for the vehicles to reach the predicted locations. In terms of practical applications, the ideal condition is that the predicted time must be as near as possible to the value of the step time t. As shown in Figure 2.14(a), we can notice that the DR, KF and PF give results closer to the ideal expected time value, with a disadvantage for the PF when the measurements' interval are lower than 0.75 s. In terms of the predicted distance errors and the MSE, this result indicates that the distance of the predictions performed by these algorithms are closer to the real location. However, as the PF has high errors, it means that the predicted results point are more often in the wrong direction, while the DR and PF, for this scenario, give more accurate results for both direction and distance. Also in Figure 2.14(a), we can notice that the SVR presents results lower than the expected values, meaning that the predictions in the average are lower than the expected distance to the real location. The NN presents average results of the predicted time greater than the expected values, which in this case represents that the predictions in the average are more often in the wrong direction of the real location.

Another important factor to be considered for the application of localization prediction in VANets is the computational time required to compute the predictions at each time step. Also in terms of VANet applications and regarding the predicted time, the computation of the predicted locations must be performed in a small amount of time such that its use can be feasible. Regarding the computational time required to perform the predictions, we can notice on Figure 2.14(b) that the DR, KF and PF are more efficient. Thus, these algorithms can be applied also for lower capabilities computational devices. Also, these algorithms do not require training and the DR and KF have a fast initial convergence to accurate predictions. However, the PF filter require an initial number of samples to the initial distribution of the particles converge to accurate predictions. Regarding the Machine Learning algorithms, the computational time required for the predictions is higher since these methods need to perform training on the samples set to achieve accurate predictions. However, we can notice that the computational time required for these algorithms while increasing the measurements' interval is almost constant and only correspond to a small fraction of the time step t. This result indicates that the Machine Learning algorithms are also a suitable approach.

2.6.5 Localization Prediction Behavior for the Measurements Interval

To illustrate the behavior of the localization prediction algorithms, we show some snapshots of the resulting Vehicle Motion Vectors (M_i). In these snapshots, the X and Y axis correspond to the vehicle trajectory during the simulation and the predictions performed while the Z axis show the error of each prediction. To illustrate the behavior of the predictions, in figure 2.15(a) we show a snapshot of 10 predictions performed by all the algorithms illustrated side by side to compare them. In this result, we can notice that booth DR and KF have a better accuracy in the prediction in terms of the direction and the distance to the target location. The NN and the SVR also have a good accuracy on the predictions, however, despite predicting the correct direction, these algorithms have higher errors in the distance between the predicted and the target location of the vehicle. The PF filter also has good accuracy on the predictions but is more affected by errors in the distance and the direction of the predictions due to the linear characteristic of this scenario.

With a snapshot of 300 predictions, in Figures 2.15(b) and 2.15(c) we can notice the best accuracy achieved by the DR and the KF for linear and non-linear trajectories. In terms of linear samples, this algorithm have the best estimations in terms of distance and direction of the predictions. The accuracy of these algorithms is only affected by changes in the vehicles speed and when the vehicle trajectory changes to non-linear very fast. However, due to the nature of the Gaussian localization errors, these algorithms have the best accuracy also in terms of non-linear trajectories and they have the advantage of a fast recovery when the trajectory turns back to linear.

In Figure 2.15(d), we can notice that the NN has a good accuracy for linear trajectories and the errors correspond in more cases to distance to the correct location. However, the NN is more affected by sudden and abrupt changes in the vehicle's trajectories which leads to high errors in the direction of the predictions. Also, we noticed that in all of the simulations, these changes constantly affect the performance of the NN since this algorithm require more samples to converge again to accurate predictions. The SVR also achieved good accuracy as shown in Figure 2.15(e). Also, we noticed in the simulations that the SVR is more accurate when the trajectories are more non-linear and this algorithm is more efficient for abrupt changes in the vehicle's trajectories, achieving the best results in this case. However, in the average, the errors in the distance to the correct location affects strongly the performance of the SVR. As depicted in Figure 2.15(f), the PF has a good accuracy for linear and non-linear trajectories. Most of the errors in the predictions done by this algorithm are related to errors on the directions of the predictions. The reason is that the non-Gaussian nature of the Particle Filter results in reducing a small fraction of the introduced Gaussian errors. Also, regarding the NN, SVR and the PF, the non-linearity nature of these algorithms results in a lower accuracy when compared to the DR and KF since, in the average, the trajectories in the analyzed VANets scenarios are mostly linear in the average.

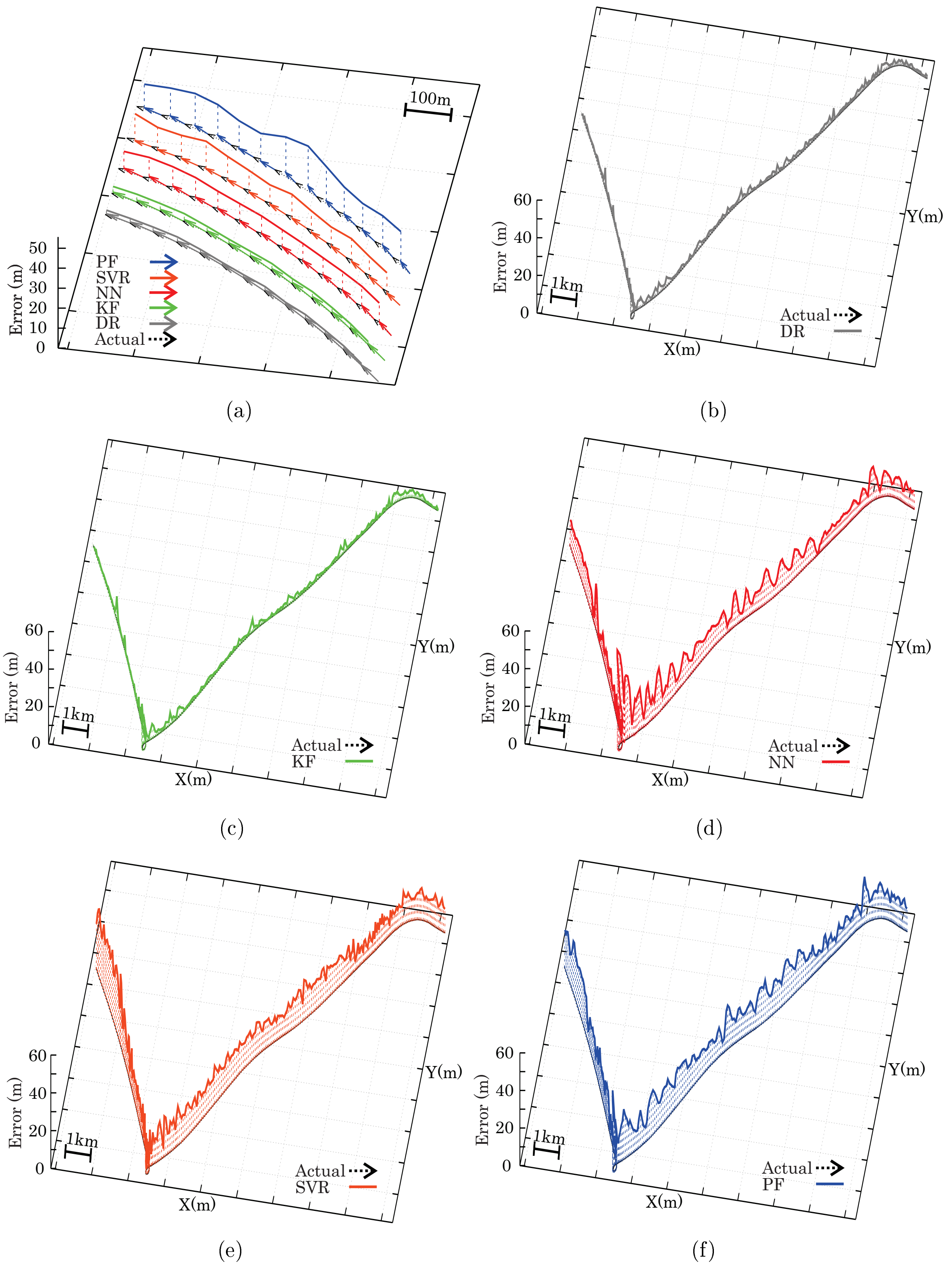


Figure 2.15. Impact of the measurements' interval on the computed predictions.

2.6.6 The Impact of the Localization Errors

To evaluate the impact of the localization error, we increase this parameter from 1 m to 5 m. Thus, when increasing the errors in the computed positions, we analyses how this introduced errors interfere on the predictions' accuracy. Since we keep the same time step for the predictions, the trajectories of the vehicles in this scenario will be the same, thus the total distance traveled by the vehicles keeps constant with an average of 10275 m. In this context, an interesting result can be seen in Figures 2.16(a) and 2.16(b): the prediction accuracy of the PF and NN is almost constant while increasing the errors in the current computed positions of the vehicles. In this result, these algorithms are not highly affected by the Gaussian noise introduced on the location of the vehicles while the KF, DR and specially the SVR are highly affected by the introduced localization errors. Also, we can notice that, in terms of the predicted distance error and the MSE the KF filter leads to more accurate results followed by the DR. However, when the localization errors are greater than 4 m the DR algorithm has an accuracy closer to the PF and NN. Also, we can notice that, for localization errors greater than 5 m, the KF tends to be less accurate than the PF and NN. Regarding these results, for high localization errors, the non-linear nature algorithm like PF and NN seem to be more suitable for high levels of localization errors. The main reason for that relies on the fact that, despite the Gaussian nature of the localization errors, for high values of errors, the linear trajectories start becoming non-linear due to the interference of the noise on the computed positions.

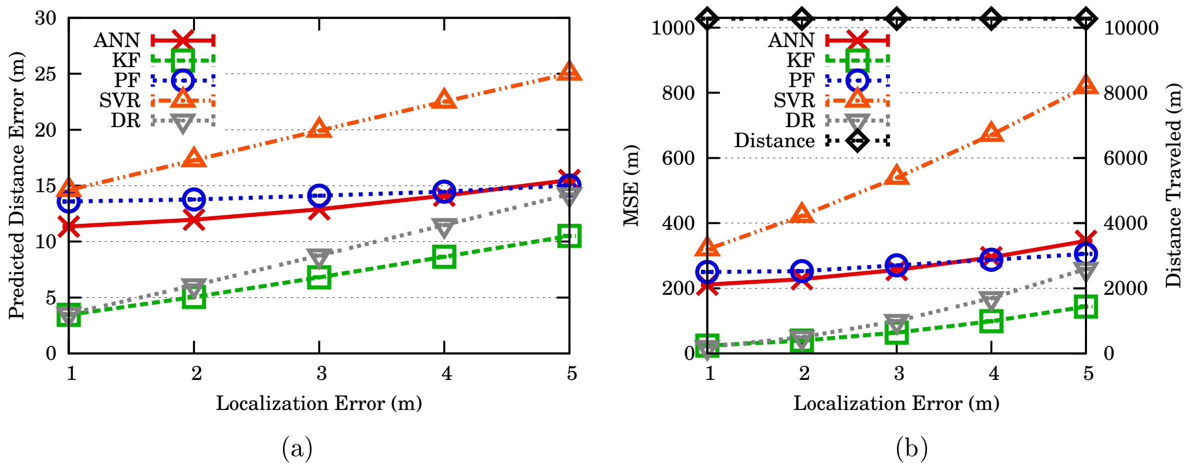
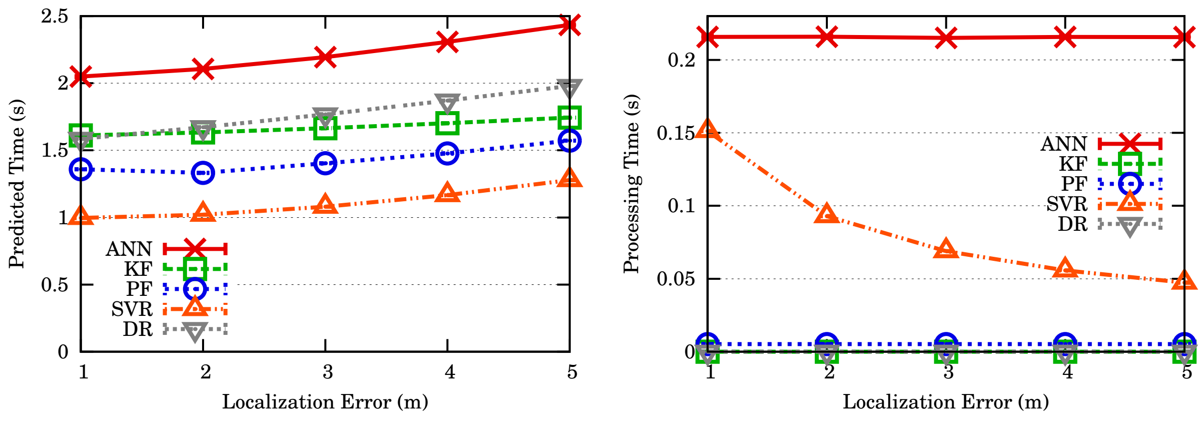


Figure 2.16. Impact of the localization errors in terms of distance.

As shown in Figure 2.17(a), we can notice that the DR, KF and PF give results closer to the ideal expect time values. Also, we can notice a disadvantage for the DR when the localization errors are greater than 3 m. In terms of the predicted distance



(a) (b)

Figure 2.17. Impact of the localization errors in terms of time.

errors and the MSE, this result indicates that the size of the predictions regarding distance are closer to the real distance. Also, in Figure 2.17(a), we can notice that the SVR presents results lower than the expected values. In this case, the localization errors make the SVR algorithm perform predictions in the average lower than the expected distance to the real location and, consequently resulting in a prediction time shorter than the expected values. On the other hand, the NN present average results of the predicted time greater than the expected values in the average.

Regarding the computational time required to perform the predictions, we can notice on Figure 2.17(b) that the DR, KF and PF are more efficient as well when increasing the localization errors. Also, this result was expected since these algorithms do not require training and the DR and KF have a fast initial convergence to accurate predictions. Regarding the Machine Learning algorithms, the computational time required for the predictions is also higher when increasing the localization errors, due to the time required to perform training on the samples set. Another interesting result can be seen in Figure 2.15(d): the computational time decreases in the SVR while in the NN it remains almost the same. For the NN, the main reason for this behavior is the fact that the training of the MLP algorithm is performed in constant 400 epochs before each prediction. The decrease in the SVR computational time required is explained by the fixed ϵ parameter introduced to measure the deviation of training samples outside ϵ-insensitive zone. In this case, the introduced localization error makes the localization samples become more sparse and the SVR algorithm is able to find faster the support vectors. However, the accuracy of the selected support vectors is highly compromised by such localization errors.

2.6.7 Localization Prediction Behavior for the Localization Errors

We also show some snapshots to illustrate the behavior of the localization prediction algorithms for the introduced localization errors. In figure 3.8(c) we show a snapshot of 10 predictions performed by all the algorithm side by side to compare them. In this result, we can notice that besides affecting the computed position of vehicles, such errors also affect the accuracy of all the prediction algorithms. Therefore, we can notice that the introduced localization errors affect the nature of vehicle's trajectories since it start becoming more non-linear. In this result, we can notice that the NN, KF, and the PF are able to give better accuracy predictions in terms of direction and distance to the target location. We also can notice that booth DR and SVR are more affected by such localization errors when compared to lower levels of errors. In this scenario, the KF is affected by the localization errors with more predictions in the wrong direction. However, the KF is still able to give the best results.

With a snapshot of 300 predictions, in Figure 2.18(c), we can notice that the best accuracy is achieved by the KF for linear and non-linear trajectories. We can also notice that the KF has the best estimations in terms of distance and direction of the predictions for the localization errors introduced. As can we see in Figure 3.8(d), the accuracy of DR is highly affected by such localization errors, since it only uses the last known position to compute the predictions. In these algorithms, we can notice that the localization errors affect the accuracy in all aspects since the linear trajectories start becoming non-linear, even with a Gaussian nature localization errors. However, in this scenario these algorithms still present the best accuracy in terms of non-linear trajectories and they have the advantage of a fast recovery when the trajectory turns back to a linear trajectory for small localization errors.

In Figures 2.18(d) and 2.18(e), we can notice that the NN and PF have a good accuracy for linear and non-linear cases being able to overcome the localization errors in almost the same level of DR and KF. For higher localization errors the NN and the PF will overcome the DR and KF with the best accuracy in the localization prediction. However, the NN is still affected by sudden and abrupt changes in the vehicle's trajectories which leads to high errors in the direction of the predictions. Finally, as shown in Figure 2.18(f), the SVR is more affected by the localization errors, especially when the trajectories are linear. In this case, the localization errors introduced turn the localization samples more sparse and the SVR algorithm is able to find faster the support vectors, due to the fixed ϵ parameter to measure the deviation of training samples outside ϵ-insensitive zone. However, these support vectors are not the best options, thus resulting in a low accuracy. For the Machine Learning algorithm, the

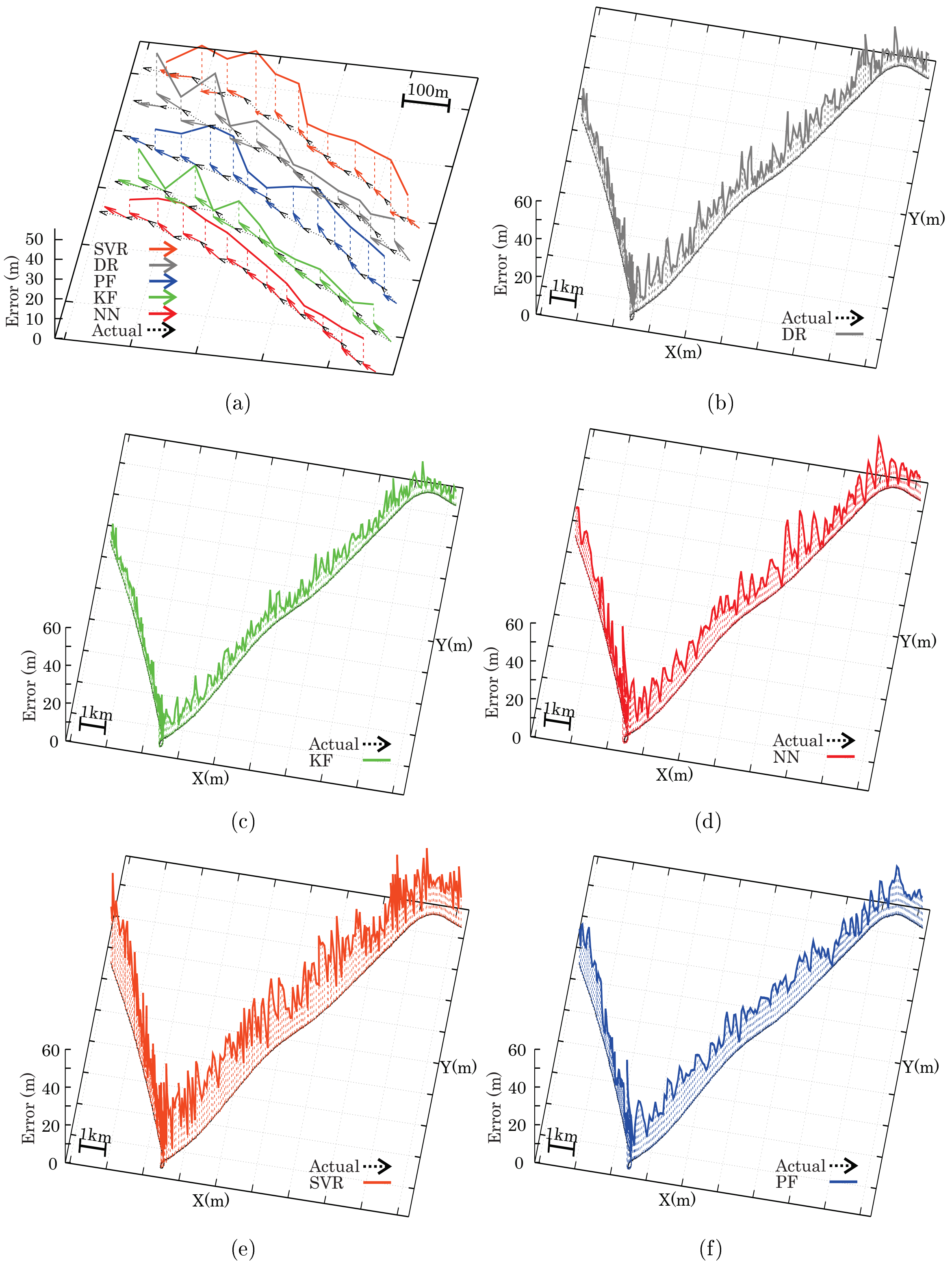


Figure 2.18. Impact of the localization errors on the computed predictions.

interference of localization errors indicates that, regarding the behavior of the results, the best alternative to overcome such limitations is the introduction of methods for dynamic parameter selection.

2.7 Sumary

In this Chapter, localization prediction were studied from the viewpoint of VANets. We discussed how these localization predictions methods can improve most VANet applications, especially critical ones. We surveyed proposed approaches for localization, target tracking and time series prediction techniques that can be used to estimate the future position of a vehicle. A number of localization prediction methods are available to be used by vehicles to estimate future positions: Dead Reckoning, Neural Networks, Support Vector Regression, Kalman Filter and Particle Filter. All of these techniques have their pros and cons. In this work we argue that localization prediction for VANets as an extension of a Data Fusion localization system is a feasible approach to circumvent the problem of *disseminating outdated localization information* in vehicular networks. We then show how localization prediction techniques can be used to compute accurate predicted positions based on a number of relatively inaccurate sample position estimations.

Chapter 3

A Prediction-based Routing Algorithm for Vehicular Ad Hoc Networks

3.1 Introduction

A major challenge in VANets is to provide reliable information exchange between vehicles with strict delay constraints. For emergency applications in VANets, people's safety is a key factor and must be considered. In certain scenarios, in cases of collisions and accidents, alert messages must be delivered in time to prevent further hazards. The high speed of vehicles is another specific challenge in VANets, which motivates the research for new data communication algorithms, since traditional protocols for Ad Hoc and MANet do not have satisfactory performance when applied to vehicular networks, due to their highly dynamic topology (Li and Wang, 2007).

In this Chapter, we consider a vehicle predicted location as its direction and speed at a given future time step (vectorial trajectory). We them propose a new Routing algorithm for data communication in VANets: the LPRV (Localization Prediction-based Routing for VANets) algorithm. The main idea of the proposed LPRV algorithm is to exploit the knowledge of vehicles predicted locations and a digital map as metrics to forward data packets, without the need for exchanging additional control message. We evaluate the performance of the proposed algorithm using the NS-2 simulator in comparison to both classic Flooding and SIFT (Labiod et al., 2010) (Simple Forwarding over Trajectory) algorithms. We also present an extensive set of experiments that clearly demonstrate the efficiency of our proposed solution in different scenarios, especially in terms of delivery rate, number of hops and delay, while maintaining a reduced number message transmissions.

The remaining of this Chapter is organized as follows. Section 3.2 describes the related work regarding position-based and geocast routing. Section 3.3 presents our LPRV algorithm, whereas Section 3.4 describes its performance evaluation. Finally, Section 3.5 presents our conclusions.

3.2 Related Work

Position-based routing has been identified as one of the most promising routing paradigms for VANets (Li and Wang, 2007). In this approach, packets are forwarded using the vehicles geographic location, which can be obtained through the use of on-board navigation systems (GPS), maps, mobility and traffic models. The GPSR (Karp and Kung, 2000) (Greedy Perimeter Stateless Routing) is one of the most well-known position-based routing protocols. It combines greedy forwarding with face routing to reach destinations where greedy forwarding fails. One main drawback of the GPSR protocol is the interference caused by buildings and other obstacles in urban scenarios, generating failures in greedy forwarding process, since direct communications between nodes may not exist. The A-STAR (Liu et al., 2004) (Anchor-based Street and Traffic Aware Routing) position-based routing protocol was proposed to overcome such interferences in city environments. A-STAR uses digital maps to compute a sequence of crossing points (anchors), through which a packet must visit to reach its destination. This algorithm also explores traffic awareness to ensure a higher probability of delivery success. Results indicate that A-STAR has the best performance when compared to GPSR, since it can select paths with higher connectivity for packet delivery. However, the A-STAR algorithm needs to keep streets' traffic information updated to compute the anchors. This means an additional overhead since cities can change the buses fleet according to its demands. The TBF (Niculescu and Nath, 2003) (Trajectory-Based Forwarding) is a position-based routing protocol that first introduced the idea of using a predefined map trajectory (path) to guide routing decisions and forward packets along a predefined path. The source specifies the trajectory in a packet and based on the neighborhood location information, a forwarding node makes a greedy decision to determine the next hop that is the closest to the trajectory.

An approach based on the position-based routing, the Geocast (Maihofer, 2004), has attracted interest in several VANet applications, mainly related to safety (accident alerts and prevention). In geocast, messages are also forwarded via location information, however, packets are delivered to all nodes in a given geographical region. Classic Flooding is a well-known routing protocol that can also be used to deliver packets to a geocast region. In this approach, all network nodes propagate a particular packet until it is received at its final destination. The classic Flooding was not originally proposed as a geocast routing protocol, however, it is useful for comparison with other geocast protocols and it is a building block for many of them (Maihofer, 2004). The SIFT (Labiod et al., 2010) (Simple Forwarding over Trajectory) protocol uses trajectory-based routing in order to achieve scalability. SIFT computes the shortest path between source and destination (geographical region) through a digital map to forward packets and limit broadcasting at the computed path, without exchanging any control messages among network nodes. This protocol uses the nodes' distance to the shortest path to guide data forwarding and, also as a contention mechanism to avoid unnecessary transmissions.

As in our proposed solution, position-based and geocast routing protocols in the literature are mainly based on the knowledge of vehicles location and digital maps to forward data packets. However, these studies do not consider vehicles predicted future locations as metric for data communication in VANets, which is the main motivation of this work.

3.3 Localization Prediction-based Routing for VANets – LPRV

In this section, we propose a new routing algorithm for data communication in VANets: the LPRV (Localization Prediction-based Routing for VANets) algorithm. The main idea of the proposed LPRV algorithm is to exploit the knowledge of vehicles predicted future locations as a metric to forward data packets, without the need for exchanging any extra control message, since trajectories are sent along with the packets. To avoid the broadcast storm problem, the LPRV algorithm also takes advantage of a digital map to limit the scope of message exchanges in the shortest path for vehicles between source and destination, thus avoiding unnecessary transmissions. Since predicted locations are encoded in the packets, only nodes with future computed positions closer to the destination are chosen as next hop of a forwarding a packet. To ensure the election of the best trajectories in the packet forwarding, the proposed algorithm computes a contention time in which the farthest nodes in a given path segment wait more time than the closer nodes, thus ensuring better greedy forwarding decisions. This will be further discussed and defined below.

3.3.1 Preliminary Definitions

In this section, we formally present the concepts used in this work.

Definition 1 (Vehicular Ad Hoc Network): we define a VANet as an Euclidean graph $G = (V, E, r)$, where $|V| = n$ is the number of nodes and r is the communication range; $V = \{v_0, v_1, v_2, \dots, v_{n-1}\}$, where $\{v_0, v_1, \dots, v_{n-1}\}$ is the set of vehicles; $\langle i, j \rangle \in E$ iff v_i reaches v_j, in other words, v_i is inside the communication range r of a node v_j; and $\forall v_i \in V$, $(Xc_i, Yc_i, Zc_i) \in \mathbb{R}^3$ is the computed position of nodes v_i (i.e., using a localization system).

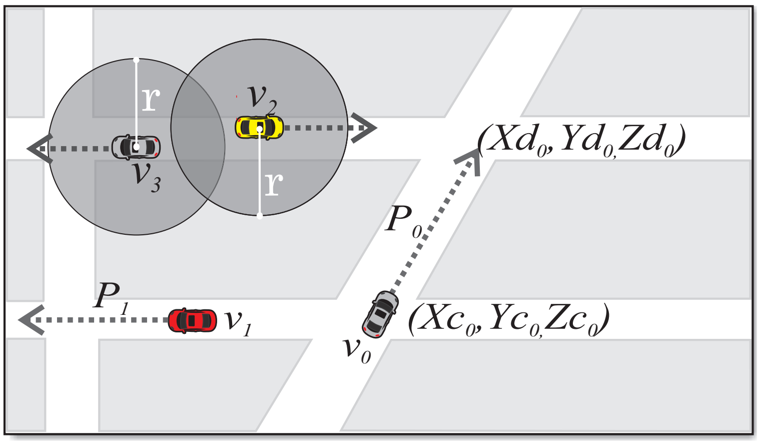


Figure 3.1. LPRV Forwarding: network nodes definition, location and trajectory.

Definition 2 (Vehicle Future Location Prediction - P_i): Is the prediction of a vehicle i future position. It can be defined as a time series regression forecasting problem and also can be formulated as a target tracking problem. In this work we consider a vector that represents the movement of a vehicle i from its current position to a future computed position. This trajectory can be a line, a curve or any other trajectory that can be mathematically expressed. For the sake of simplification, we consider that a vehicle will maintain the trajectory of a straight line. This line is defined as $P_i = ((Xc_i, Yc_i, Zc_i), (Xd_i, Yd_i, Zd_i))$ (as depicted in Figure 1), where (Xc_i, Yc_i, Zc_i) is the current vehicle's position, (Xd_i, Yd_i, Zd_i) is the next estimated position direction and s_i its displacement speed. We also define a function $P_i.distance(D_k)$ which computes the shortest distance from the line P_i (trajectory) to the packet destination point D_k, based on the Euclidean geometry distance computation from a line to a point. As we showed in Chapter 2, the prediction of a vehicle future position can be accurately computed using target tracking and time series approaches. In addition, according to Barrios and Motai (2011) a future location of an automobile can also be accurately predicted using a combination of Global Positioning System (GPS), Geographic Information System (GIS) and Kalman Filters (KFs). Also related, as shown in (Gning and Bonnifait, 2004), such trajectory information can be obtained by the combination of dead reckoning and in-vehicle sensors.

Definition 3 (Digital Map): digital road map is defined as a directed graph $M =$

(A, S), where $A = \{a_0, a_1, a_2, \ldots, a_m\}$ is the set of vertices (e.g. crossings in a urban area) and $\forall a_i \in A$, $(Xa_i, Ya_i, Za_i) \in \mathbb{R}^3$ is a vertex location, $\langle i, j \rangle \in S$ iff exists a path from a_i to a_j, in other words, exists a street on the map where a vehicle $v_i \in V$, starting passing by a_i can reach a_j. We define the function $M.shortestPath(a_s, a_d)$ which returns a set of vertices $A' \cup A$ corresponding to the shortest path between the starting vertex a_s and the destination vertex a_d (as shown in Figure 2). We also define the function $M.lastVertex(P_i)$, which returns the last visited vertex and the next vertex to be visited by the trajectory P_i.

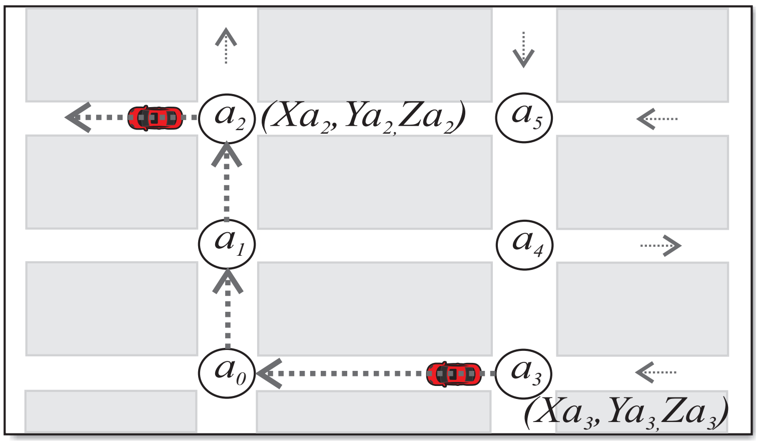


Figure 3.2. Digital map and the shortest path function example: $M.shortestPath(a_3, a_2)$.

3.3.2 The LPRV Algorithm

Our proposed LPRV algorithm, shown and explained in Algorithm 2, is divided into three operating phases: send, reception, and forward. The first phase starts when the application generates a data packet to be forwarded to a destination region (Lines 1–6). The destination D_i is chosen by the application and it refers to a monitoring station or an area to report an event, such as when a vehicle collision occurs. This packet contains, among other information, the node future position and displacement speed (P_i and s_i, Line 4), the initial vertex on map where the packet was generated, and it is sent by broadcast to all nodes in the one-hop neighborhood (Line 6).

The next two phases start when a node receives a packet (Lines 7–8). First, it is verified if the position of the current node is within the packet's destination area (Lines 9–12). If so, the packet is received (reception phase). Otherwise, the forward phase is started by checking if the current node has never forwarded this packet (Line 13). If this condition holds, the node updates its forward table, trajectory and speed information, the last visited vertex, and the next vertex to be visited in the

Algorithm 2 The LPRV Algorithm

```
▷ Input:
 1: Node $v_i$ sends a $packet$ with id $pktid_i$ to $D_i$.
 Action:
 2: $pktid_i \leftarrow nextPktID()$;                                   {Packet id}
 3: $src_i \leftarrow v_i; D_i \leftarrow (X_d, Y_d, Z_d, Range)$;      {source node and destination}
 4: $P_i \leftarrow predicLocation(); s_i \leftarrow speed()$;{Node future position and speed}
 5: $startM_i \leftarrow Map.lastVertex(P_i)$;                          {Course starting vertex}
 6: Broadcast $packet(pktid_i, src_i, D_i, P_i, s_i, startM_i, data)$;
▷ Input:
 7: Node $v_j$ receives a $packet$ with id $pktid_f$ addressed to $D_k$ area.
 8: $msg_k \leftarrow packet(pktid_k, src_k, D_k, P_k, s_k, startM_k, data)$;
 Action:
 9: $Lp_j \leftarrow (Xc_j, Yc_j, Zc_j)$;                              {Position of the current node}
10: if $Lp_j.distance(D_k) \leq D_k.Range$ then                       {Node within dest area}
11:     $receive(msg_k)$;                                             {Current node receives the packet}
12: end if
13: if $(src_k, pktid_k) \notin Fwd_j$ then                           {IF: the node never forwarded the packet}
14:     $Fwd_j \leftarrow Fwd_j \cup (src_k, pktid_k)$;                {Update the forward table}
15:     $P_j \leftarrow predicLocation(); s_j \leftarrow speed()$;     {Node trajectory and speed}
16:     $lastM_j \leftarrow Map.lastVertex(P_j)$;                     {Last visited vertex}
17:     $nextM_j \leftarrow Map.nextVertex(P_j)$;                     {Next vertex to visit}
18:     if $(lastM_j \in Map.shortestPath(startM_k, D_k)) \quad \vee(nextM_j \in Map.shortestPath(startM_k, D_k))$
        then                                                          {IF: node trajectory is in the shortest path}
19:         if $P_j.distance(D_k) < P_k.distance(D_k)$ then           {IF: node trajectory is more closer to
            destination}
20:             $time_j \leftarrow \frac{distance(Lp_j, nextM)}{(s_j \times \alpha)}$;    {Time to next vertex}
21:             Broadcast $packet(pktid_k, src_k, D_k, P_j, s_j, startM_k, data)$ in $time_j$;   {Forwards the packet}
22:         end if
23:     end if
24: end if
```

digital map based on its trajectory (Lines 14–17). These two vertices are used to verify if the current vehicle's trajectory is in the computed shortest path on the digital map (Line 18). The current node's trajectory is also compared with the trajectory of the last node that forwarded the packet in order to verify if the current node's trajectory is closer to destination area (Line 19). This comparison basically consists of computing the shortest distance from each line to a point based on the Euclidean geometry (Definition 2). If the current node has a better trajectory, a time for this node to the next vertex in the digital map is computed (Line 20). This time is used as a broadcast storm contention mechanism. This strategy forces the nodes closer to the destination area to forward their packets first, preventing nodes with farther trajectories from forwarding unnecessary packets (as depicted in Figure 3.3.2). The parameter α is an environment adaptation parameter to adjust the contention time value in the same scale of the network packet delay. It can be dynamically computed using the difference of base 10 logarithms of both values (e.g. $time_j \leftarrow 10^{\log time_i - \log pktDelay_i}$). Finally, when the contention time expires, the packet is forwarded via broadcast to all nodes in the one-hop neighborhood (Line 21). This process is repeated until the packet reaches its destination.

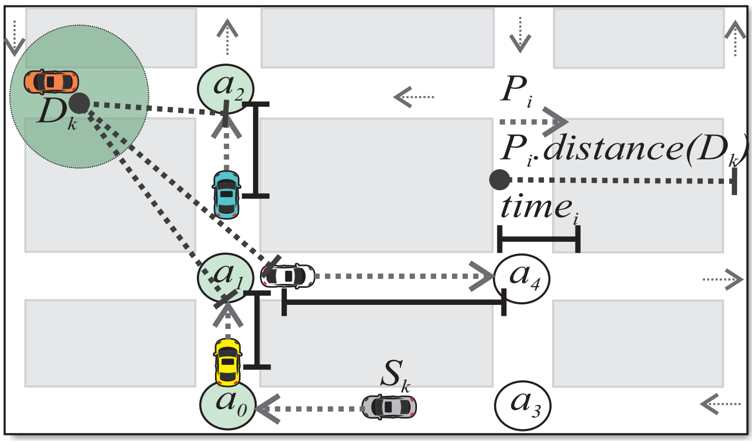


Figure 3.3. LPRV Forwarding: choice of the trajectory closer to destination and contention time computing.

It is important to note that this work focuses mainly on simple packet forwarding. We consider this approach more suitable for VANets scenarios since the LPRV algorithm does not use any control messages to find destination nodes, maintain routing paths, and report errors. Traditional end-to-end routing schemes that try to maintain routes between source and destination nodes are not very efficient in VANets due to the high mobility of nodes. Finally, the LPRV algorithm can be easily adapted to reply to data queries (Boukerche and Nikoletseas, 2004), likewise in Wireless Sensor Networks where the sink node sends a query (which is disseminated by flooding) to the sensor network, as if it was a distributed database system (i.e., sensor databases (Hong and Madden, 2004)).

3.4 Performance Evaluation

3.4.1 Methodology

The performance evaluation is performed through simulations using the NS-2. We evaluate the performance results of our proposed algorithm, in comparison to both classic Flooding and SIFT algorithms, in terms of network scale, delivery ratio, delay, number of hops, and packet traveled distance. The default values for our simulation parameters are shown in Table 1. In all the results, curves represent average values, while error bars represent confidence intervals for 95% of confidence from 33 independent instances (seeds).

Parameter	*Value*
Simulation area	1000 m×1000 m
Number of nodes	700
Communication range	100 m
One hop delay	0.1 s
Non-determ. errors	30 µs
Vehicles' speed	7-40 km/h

Table 3.1. LPRV simulation parameters.

As in Labiod et al. (2010), the simulation field map represents a simple grid-shaped urban scenario. Network nodes are distributed on a 1000 m×1000 m rectangular simulation area. The road map is a 10×10 one-way streets grid, where two parallel streets always have opposite direction of traffic between each other. We assume that each vehicle travels these streets with speeds from 7 km/h to 40 km/h. Thus, vehicles are allowed to overtake each other. We also defined four source/destination targets where packets need to be forwarded from: $S_1 = (0, 0, 0)$ to $D_1 = (800, 800, 0)$, from $S_2 = (900, 100, 0)$ to $D_2 = (100, 800, 0)$, from $S_3 = (900, 900, 0)$ to $D_3 = (100, 100, 0)$ and from $S_4 = (100, 900, 0)$ to $D_4 = (800, 100, 0)$ according to the Cartesian coordinate system of the simulation area (as shown in Figure 4). It is important to notice that, although the choice of random source/destination targets can improve the average results, we choose to analyze the worst case for data delivery points in terms of distance. In addition, we consider that the application generates a data packet when reporting information to a monitoring station, such as when a vehicle collision occurs. Also, these points are not stationary and they may vary depending on the algorithm's application.

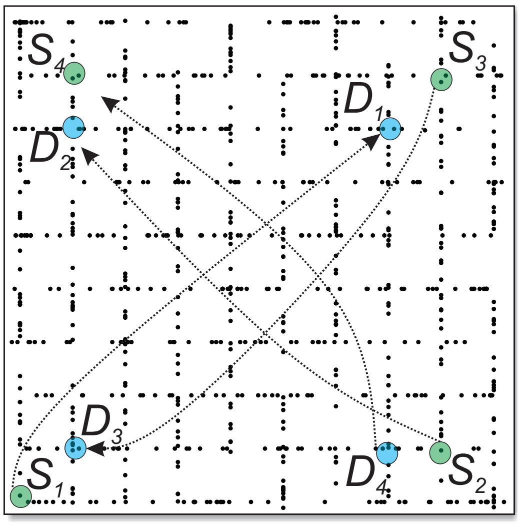


Figure 3.4. Simulation scenario: vehicles nodes, packets' source and destination

Regarding the network topology, we assume that node location initially obeys a disturbed grid, in which the location of each node in the streets node is disturbed by a random zero-mean Gaussian error. Therefore, nodes will tend to uniformly occupy each street without forming a regular line. Finally, to simulate delay measurement inaccuracies we disturbed the mean delay by a standard deviation of 30 µs (Maróti et al., 2004).

3.4.2 The Impact of Network Scale

Scalability is evaluated by increasing the network size from 350 to 1000 vehicles. As shown in Figure 3.5(a), we can notice that the Flooding algorithm is able to deliver almost all packets after 450 nodes. This result is explained by the characteristic of this algorithm of always delivering packets if there is connectivity between source and destination. Our LPRV algorithm has a higher data delivery rate, being higher than the SIFT algorithm. In this case, with small-scale network, these two Algorithms are affected by the existence of intermittent connectivity at computed delivery paths, while the Flooding algorithm can deliver packets by bypassing these areas through detour paths. However, a disadvantage of Flooding is highlighted in the number of transmitted packets when increasing the network scale (as shown in Figure 3.5(b)). In this result, we can see the reduced number of transmitted packets achieved by both LPRV and SIFT algorithms where the SIFT algorithm has a small advantage. This result is mainly due to the limited packet broadcast proposed in these solutions without the need for exchanging additional control message.

Figure 3.5(c) shows that our LPRV algorithm outperforms the other algorithms by using fewer hops to deliver packets. Since predicted trajectories closer to the destination are selected to forward packets, LPRV can deliver packets via the shortest paths, as well as by greedy forwarding algorithms. Additionally, the lower average number of hops in the delivered packets achieved by LPRV is also due to its characteristic of choosing only predicted future positions located on the shortest path for vehicles between source and destination. As we can see in Figure 3.5(d), the choice of best trajectories performed by the LPRV algorithm also leads to a smaller distance traveled by the packets. These results show the benefits of using vehicles predicted future locations as a metric for data communication, demonstrating that our LPRV algorithm is scalable.

3.4.3 The Impact of Vehicle Speed

To evaluate the impact of vehicle speed, we increase this parameter from 20 km/h to 80 km/h. As shown in Figure 3.6(a), we can see that the Flooding algorithm is not

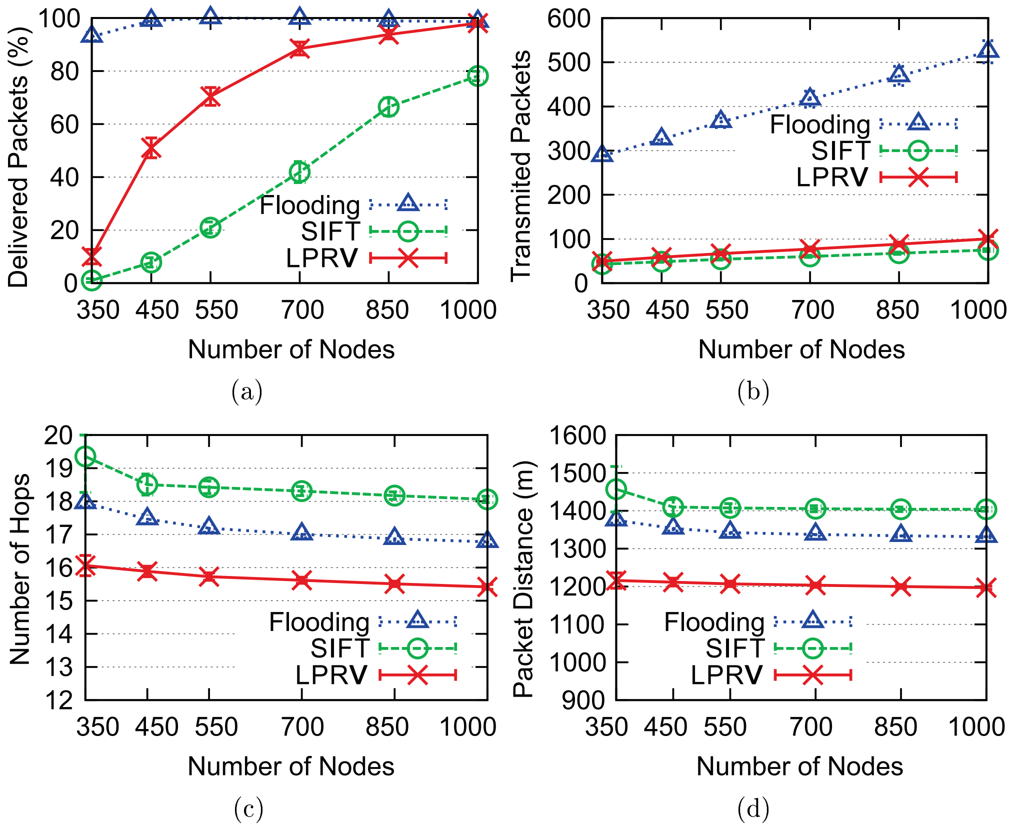


Figure 3.5. Impact of the network scale.

affected by the vehicles speed. However, the SIFT algorithm is highly affected by this increase while the LPRV algorithm can achieve a higher delivery rate, being less affected by the increase in the vehicles speed, since it actually uses the knowledge of the vehicles speed to forward packets. The delivery rate decrease in these two algorithms is explained by the increase in the vehicles speed, which generates a higher number of regions with intermittent connection. In Figure 3.6(b), we can highlight again the high number of packets transmitted by the Flooding algorithm, while the SIFT algorithm and LPRV perform less packet transmissions due to their limited broadcasts.

Figure 3.6(c) shows that increasing the vehicle's speed does not affect the number of hops traversed by packets in all analyzed algorithms. LPRV presents better performance in terms of the number of hops required to deliver packets, since it uses predicted trajectories closer to the destination to forward packets considering the speed of the nodes, and also by choosing only predicted future positions located on the shortest path for vehicles between source and destination. As shown in Figure 3.6(d), the LPRV algorithm also has a lower delay in delivered packets due to the increase of the nodes speed, which reduces the waiting time for the contention of the packets, thus resulting in a shorter data delivery delay. The SIFT algorithm has a higher packet

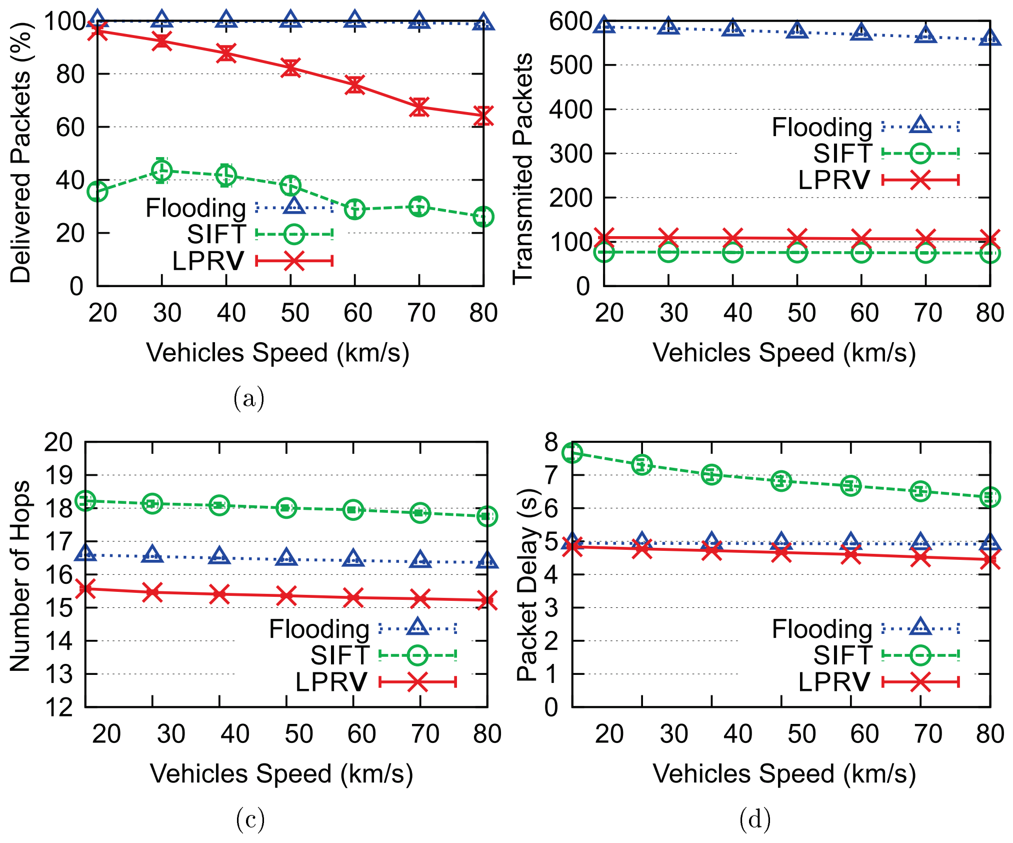


Figure 3.6. Impact of the vehicles speed.

delay since its contention mechanism does not consider the vehicle's speed.

3.4.4 The Impact of the Communication Range

To evaluate the impact of the communication range, we increase this parameter from 90 m to 160 m. After a communication range of 120 m, almost 100% of the packets are delivered in all algorithms (as shown in Figure 3.7(a)). This is due to the fact that a greater coverage area is achieved when increasing the communication range, which results in fewer areas with intermittent connectivity. For a communication range bellow 120 m, we can notice that the SIFT algorithm is the most affected while the LPRV algorithm can deliver more packets. As the LPRV algorithm uses predicted locations to verify if a vehicle is in the shortest path area, a larger coverage area is achieved when compared to the use of node's location, making the LPRV algorithm more robust to areas with intermittent connectivity. As shown in Figure 3.7(b), the Flooding algorithm has a higher decrease in the packet delay with the increase in the communication range. The SIFT and LPRV algorithms have a constant packet delivery delay when increasing the communication range due to the contention time introduced.

However, the LPRV algorithm is able to deliver packets with lower delay.

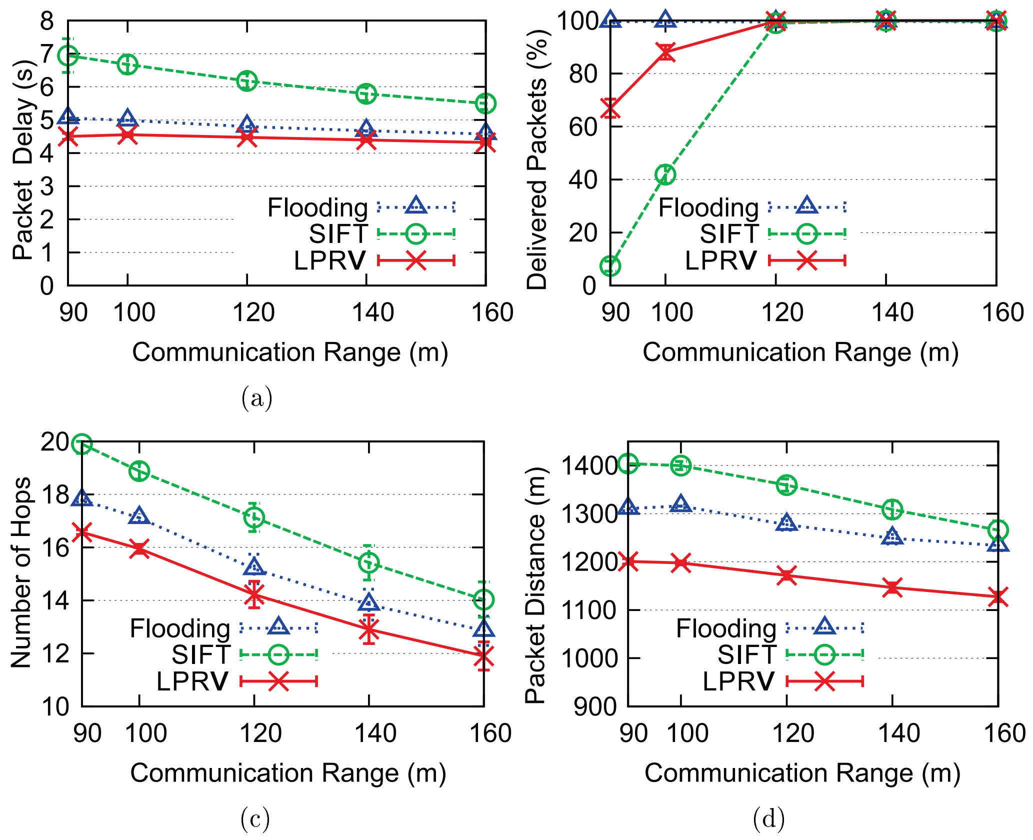
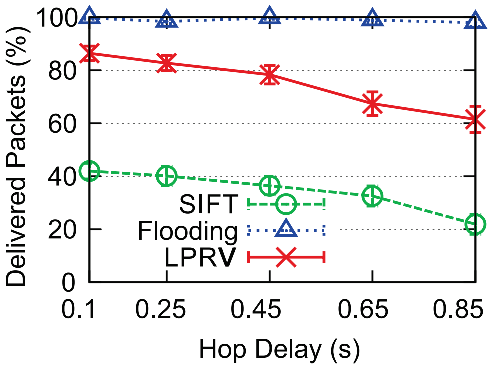


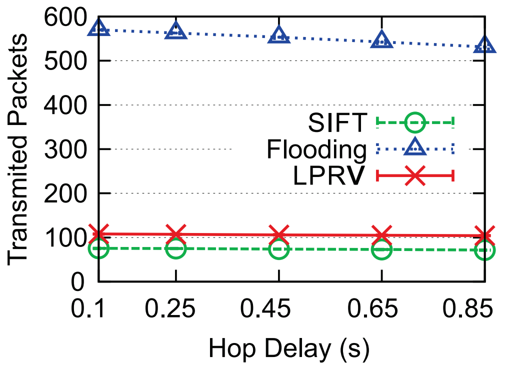
Figure 3.7. Impact of the communication range.

Figure 3.7(c) shows that the number of hops used to deliver packets decreases in all analyzed algorithms, with a slight advantage to LPRV. This decrease was expected since with the increase of the communication range, a greater coverage area is achieved, thus resulting in fewer hops required to deliver packets. As we can see in Figure 3.7(d), the increase in the communication range also decreases the distance traveled by the packets in all algorithms with an advantage to the LPRV algorithm. In these results, the better results obtained by the LPRV algorithm show the use of vehicles predicted future localization an efficient strategy in terms of communication range, since the benefits of the increased coverage area are combined with the advantages of forwarding packets through predicted trajectories closer to the destination.

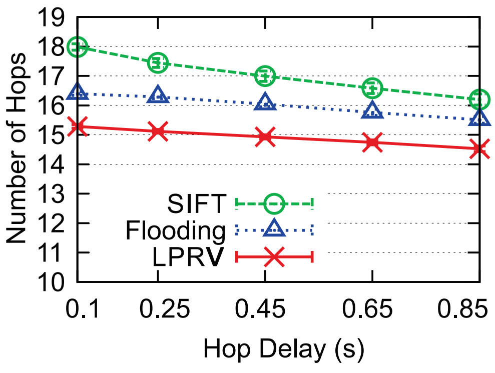
3.4.5 The Impact of Hop Delay

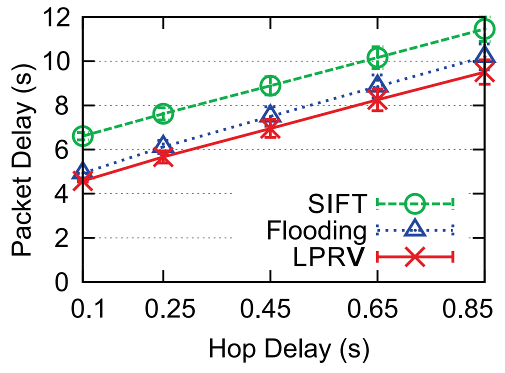
Hop delay refers to the processing time of the node before forwarding a packet (i.e., to compute its location and predict the trajectory, and to access the digital map). To evaluate the impact of this delay, we vary this parameter from 0.1 s to 0.8 s. As depicted in Figure 3.8(a), this delay affects the number of transmitted packets in the LPRV algorithm and especially in the SIFT algorithm, since the increase in packet delay leads to more frequent network topology changes. The LPRV algorithm has better results compared to the SIFT algorithm due to the use of the vehicles predicted locations, resulting in less impact on network topology changes. As can be seen in Figure 3.8(b), the increase of the packet delay does not affect the algorithms analyzed in this work in terms of transmitted packets. In this result, we can see the reduced number of transmitted packets achieved by the LPRV and SIFT algorithms, in which the SIFT algorithm has a small advantage. Also, this result shows that the LPRV algorithm and the SIFT algorithm have a reduced number of transmitted packets for the different VANet scenarios analyzed.





(a)





(c) (d)

Figure 3.8. Impact of the hop delay.

Figure 3.8(c) depicts that the number of hops to deliver packets has a small decrease in all analyzed algorithms, with an advantage to LPRV. For all analyzed scenarios in this work, the use of vehicles predicted future locations and by choosing only trajectories located on the shortest path for vehicles between source and destination leads to a few hops used in packets delivery. Finally, as we can see in Figure 3.8(d), the LPRV algorithm has a small delay in the delivered packet when increasing the one hop delay. This result shows that the LPRV algorithm introduces a lower packet delay in the different VANet scenarios analyzed.

3.5 Sumary

In this Chapter, we proposed a new VANet routing algorithm that uses the knowledge of the vehicles predicted locations to improve the routing performance in several aspects. In our algorithm, called LPRV (Trajectory-based Routing for VANets), we exploit the knowledge of vehicles predicted future locations and a digital map as metrics to forward data packets, without the need for exchanging any control message. We presented an extensive set of simulation experiments comparing our proposed solution to both classic Flooding and SIFT algorithms. The obtained results demonstrated the efficiency of the proposed solution for different VANet scenarios and the benefits of using vehicles predicted locations as a metric for data communication, especially in terms of delivery rate, number of hops and delay, with a reduced number transmitted packets.

Chapter 4

Conclusions

This Chapter summarizes the thesis proposal conclusions and future research directions. We first present the thesis conclusions in Section 4.1. Then, in Section 4.2, we present the future directions of this work, and we finish the document by presenting, in Section 4.3, the list of produced works and publications we achieved during the conception of this thesis.

4.1 Final Remarks and Summary of Contributions

In this work, localization prediction were studied from the viewpoint of Vehicular Ad Hoc Networks (VANets). We discussed how these localization predictions methods can improve most VANet applications, especially critical ones. We surveyed proposed approaches for localization, target tracking and time series prediction techniques that can be used to estimate the future position of a vehicle. A number of localization prediction methods are available to be used by vehicles to estimate future positions: Dead Reckoning, Neural Networks, Support Vector Regression, Kalman Filter and Particle Filter. All of these techniques have their pros and cons. In this work we argue that localization prediction for VANets as an extension of a Data Fusion localization system is a feasible approach to circumvent the problem of disseminating outdated localization information in vehicular networks. We then show how localization prediction techniques can be used to compute accurate predicted positions based on a number of relatively inaccurate sample position estimations.

As a general conclusion, the Dead Reckoning, Kalman Filter, and Particle Filter have shown best computational performance in terms of response time. The Machine Learning methods also showed a viable computational effort for computing the predictions. For lower localization errors, the Dead Reckoning and the Kalman Filter achieved the best accuracy in the predictions due to the fact that the trajectories in the realistic VANet scenario analyzed are strongly linear. However, when introducing high levels of localization noise, Particle Filters and Neural Networks successfully filtered the errors associated to the target prediction estimation. Consequently, the Particle Filter and Neural Networks tends to outperform the Kalman Filter as the localization error increases since such Gaussian errors can affect the linear aspect of the vehicles' trajectories.

We also proposed a new VANet routing algorithm that uses the knowledge of the vehicles predicted locations to improve the routing performance in several aspects. In our proposed LPRV algorithm, we exploit the knowledge of vehicles predicted future locations and a digital map as metrics to forward data packets, without the need for exchanging any control message. We presented an extensive set of simulation experiments comparing our proposed solution to both classic Flooding and SIFT algorithms. The obtained results demonstrated the efficiency of the proposed solution for different VANet scenarios and the benefits of using vehicles predicted locations as a metric for data communication, especially in terms of delivery rate, number of hops and delay, with a reduced number transmitted packets.

4.2 Directions for Future Research

This work leads to some particularly interesting directions. The first is to properly characterize the trajectories nature in terms of linear/non-linear samples, so that we can understand the expected magnitude, direction, and orientation of the error resulting from localization prediction algorithms. Such knowledge allows us to improve the localization prediction methods that use such information to compensate and reduce the impact of prediction errors. The second is related to the localization errors on the computed position of the vehicles, which extremely affects the performance of a predictor algorithm. Therefore, even though these approaches tackles different processes and measurement models, an interesting alternative to improve VANets services through localization prediction can be combining, in a single solution, two or more solutions to deal with Gaussian/non-Gaussian noise sources and linear/non-linear trajectory models.

Regarding our LPRV algorithm, the results are very promising, but some limitations still need to be further exploited as future work. First, we will evaluate our solution using real world vehicular mobility data and the interference of buildings on the wireless link. Then, we will evaluate the performance and the computational cost of the proposed solution using several methods for predicting vehicle future location and evaluate the impact of errors introduced by such algorithms for linear/non-linear models subjected to Gaussian/non-Gaussian noise sources.

4.3 List of Publications

Publications done by the author during the doctorate.

- **Awards:**

  Balico, L. N., Oliveira,H. A., Barreto, R. S., Loureiro, A.A. F., and Pazzi, R. W. (2015). **A prediction-based routing algorithm for vehicular ad hoc networks. Best International Paper Award**. In *20th IEEE Symposium on Computers and Communications (ISCC'15).*

- **Conferences papers:**

  Balico, L. N., Oliveira, H. A., Barreto, R. S., Loureiro, A. A., and Pazzi, R. W. (2015). A prediction-based routing algorithm for vehicular ad hoc networks. In *Proceedings of the 20th IEEE Symposium on Computers and Communications (ISCC'15)*, page to appear, Larnaca, Cyprus.

  Balico, L. N., Oliveira, H. A., Souza, E. L., Pazzi, R. W., and Nakamura, E. F. (2015b). On the performance of localization prediction methods for vehicular ad hoc networks. In *Proceedings of the 20th IEEE Symposium on Computers and Communications (ISCC'15)*, page to appear, Larnaca, Cyprus.

  Balico, L. N., Oliveira, H. A., Nakamura, E. F., Barreto, R. S., and Loureiro, A. A. (2015b). Routing and data aggregation toward a high speed sink in wireless sensor networks. In *Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS'15)*, pages 260–265, Fortaleza, Brazil.

  Balico, L. N., Oliveira, H. A., Nakamura, E. F., Barreto, R. S., and Loureiro, A. A. (2013a). Roteamento e agregação de dados usando sinks em alta velocidade em redes de sensores sem fio. In *Proceedings of the 31th Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*pages 542-554, Brasília, Brazil.

  Lima, M., Oliveira, H., Nakamura, E., Balico, L., and Loureiro, A. (2013b). Greedy routing and data aggregation in wireless sensor networks. In *Proceedings of the 18th IEEE Symposium on Computers and Communications (ISCC'13)*, pages 342–347, Split, Croatia.

  Neto, J., Lima, M., Mota, E., Cerqueira, E., and Balico, L. (2013c). Adaptive contact volume prediction in delay tolerant networks. In *Proceedings of the 18th IEEE Symposium on Computers and Communications (ISCC'13)*, pages 372–376, Split, Croatia.

- **Periodical papers under authors evaluation**.

  Balico, L. N., Souza, E. L., Oliveira, H. A., Pazzi, R. W., Nakamura, E. F., Barreto, R. S., and Loureiro, A. A. (2015d). Data aggregation toward a high speed sink using actual localization systems in sensor networks. *Springer Wireless Networks.* **(Submitted)**.

  Balico, L. N., Souza, E. L., Oliveira, H. A., Pazzi, R. W., Nakamura, E. F., Barreto, R. S., and Loureiro, A. A. (2015e). A survey on localization prediction in vehicular ad hoc networks. *IEEE Communications Surveys and Tutorials* . **(Submitted)**.

Appendix A

Glossary of Terms

A-STAR	*Anchor-based Street and Traffic Aware Routing.*
BP	*Back Propagation algorithm.*
DR	*Dead Reckoning.*
EKF	*Extended Kalman Filter.*
GNSS	*Global Navigation Satellite System.*
GPS	*Global Positioning System.*
GPSR	*Greedy Perimeter Stateless Routing.*
ITS	*Intelligent Transportation Systems.*
KF	*Kalman Filter.*
LPRV	*Localization Prediction-based Routing for VANets.*
MANet	*Mobile Ad hoc Network.*
MLNN	*Multilayer Feed Forward Neural Network.*
MLP	*Multilayer Perceptron.*
MSE	*Minimum Squared Error.*
NN	*Neural Network.*
NS-2	*The Network Simulator - ns-2 v2.34.*
PF	*Particle Filter.*
SIFT	*Simple Forwarding over Trajectory.*
SVM	*Support Vector Machines.*
SVR	*Support Vector Regression.*

TBF	*Trajectory-Based Forwarding.*
UKF	*Unscented Kalman Filter.*
V2I	*vehicle-to-vehicle.*
V2V	*vehicle-to-vehicle.*
VANet	*Vehicular Ad Hoc Network.*
VC	*Vapnik-Chervonenkis theory.*
WSN	*Wireless Sensor Network.*

Appendix B

Acknowledgments

The authors acknowledge the support granted by FAPEAM through process number 01135/2011 (PRONEX), 582/2014 and 579/2014 (PRO-TI-PESQUISA). FAPESP through process 08/57870-9 and by CNPq through process 573963/2008-8.

Bibliography

Agarwal, A. and Das, S. (2003). Dead reckoning in mobile ad hoc networks. In *Proceedings of IEEE Wireless Communications and Networking (WCNC'03)*, pages 1838–1843, New Orleans, Louisiana, USA.

Al-Sultan, S., Al-Doori, M. M., Al-Bayatti, A. H., and Zedan, H. (2014). A comprehensive survey on vehicular ad hoc network. *Journal of Network and Computer Applications*, 37:380–392.

Alam, N. and Dempster, A. (2013). Cooperative positioning for vehicular networks: Facts and future. *IEEE Transactions on Intelligent Transportation Systems*, 14:1708–1717.

Alam, N., Tabatabaei Balaei, A., and Dempster, A. (2013). Relative positioning enhancement in vanets: A tight integration approach. *IEEE Transactions on Intelligent Transportation Systems*, 14:47–55.

Ammoun, S., Nashashibi, F., and Laurgeau, C. (2007). Crossroads risk assessment using gps and inter-vehicle communications. *IET Intelligent Transport Systems*, 1:95–101.

Armaghan, M., Fathy, M., and Yousefi, S. (2009). Improving the performance of beacon safety message dissemination in vehicular networks using kalman filter estimation. In *Communication and Networking*. Springer.

Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188.

Balico, L. N., Oliveira, H. A., Barreto, R. S., Loureiro, A. A., and Pazzi, R. W. (2015). A prediction-based routing algorithm for vehicular ad hoc networks. In *Proceedings* *of the 20th IEEE Symposium on Computers and Communications (ISCC'15)*, page to appear, Larnaca, Cyprus.

Barrios, C. and Motai, Y. (2011). Improving estimation of vehicle's trajectory using the latest global positioning system with kalman filtering. *IEEE Transactions on Instrumentation and Measurement*, 60(12).

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition.* Oxford University Press.

Bonissone, P. P. (1997). Soft computing: the convergence of emerging reasoning technologies. *Springer Soft Computing*, 1:6–18.

Boukerche, A. and Nikoletseas, S. (2004). Wireless communications systems and networks. chapter Protocols for Data Propagation in Wireless Sensor Networks, pages 23–51. Plenum Press.

Boukerche, A., Oliveira, H. A. B. F., Nakamura, E. F., and Loureiro, A. A. F. (2008). Vehicular ad hoc networks: A new challenge for localization-based systems. *Elsevier Computer Communications*, 31(12):2838–2849.

Boukerche, A., Rezende, C., and Pazzi, R. (2009). Improving neighbor localization in vehicular ad hoc networks to avoid overhead from periodic messages. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'09)*, pages 1–6, Honolulu, Hawaii, USA.

Capka, J. and Boutaba, R. (2004). Mobility prediction in wireless networks using neural networks. In *Management of Multimedia Networks and Services*, pages 320–333. Springer.

Chausse, F., Laneurit, J., and Chapuis, R. (2005). Vehicle localization on a digital map using particles filtering. In *Proceedings of the IEEE Intelligent Vehicles Symposium.*, pages 243–248, Las Vegas, Nevada, USA.

Daum, F. (2005). Nonlinear filters: beyond the kalman filter. *IEEE Aerospace and Electronic Systems Magazine*, 20:57–69.

Doucet, A., de Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice.* Information Science and Statistics. Springer.

E. Najjar, M. and Bonnifait, P. (2005). A road-matching method for precise vehicle localization using belief theory and kalman filtering. *Springer Autonomous Robots*, 19(2):173–191.

Fernandez-Madrigal, J. A., Cruz-Martin, E., Gonzalez, J., Galindo, C., and Blanco, J. L. (2007). Application of uwb and gps technologies for vehicle localization in combined indoor-outdoor environments. In *Proceedings of the 9th IEEE International Symposium on Signal Processing and Its Applications (ISSPA'07)*, pages 1–4, Sharjah, United Arab Emirates.

Gning, A. and Bonnifait, P. (2004). Guaranteed dynamic localization using constraints propagation techniques on real intervals. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA'04)*, volume 2, New Orleans, LA, USA.

Golestan, K., Seifzadeh, S., Kamel, M., Karray, F., and Sattar, F. (2012). Vehicle localization in vanets using data fusion and v2v communication. In *Proceedings of the Second ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet'12)*, pages 123–130, Paphos, Cyprus Island.

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings-F*, 140:107–113.

Griffin, R. and Sage, A. (1969). Sensitivity analysis of discrete filtering and smoothing algorithms. *Journal of the AIAA*, 7:1890–1897.

H. Nguyen, A. Bhawiyuga, H. J. (2012). A comprehensive analysis of beacon dissemination in vehicular networks. In *Proceedings of the IEEE 75th Vehicular Technology Conference (VTC'12-Spring)*, pages 1–5, Yokohama, Japan.

Hartenstein, H. and Laberteaux, K. (2008). A tutorial survey on vehicular ad hoc networks. *IEEE Communications Magazine*, 46:164–171.

Haykin, S. (1998). *Neural Networks: A Comprehensive Foundation.* Prentice Hall PTR, 2nd edition.

Hong, W. and Madden, S. (2004). Implementation and research issues in query processing for wireless sensor networks. In *Proceedings of the 20th IEEE International Conference on Data Engineering (ICDE'04)*, page 876, Washington, DC, USA. IEEE.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Elsevier Neural Networks*, 2:359–366.

Hossain, E., Niyato, D., and Han, Z. (2009). *Dynamic Spectrum Access and Management in Cognitive Radio Networks.* Cambridge University Press, 1st edition.

Hrizi, F., Härri, J., and Bonnet, C. (2012). Can mobility predictions be compatible with cooperative active safety for vanet? In *Proceedings of the Ninth ACM International Workshop on Vehicular Inter-networking, Systems, and Applications (VANET'12)*, pages 111–114, Low Wood Bay, Lake District, United Kingdom.

Huang, C., Chuang, Y., Yang, D., Chen, I., Chen, Y., and Hu, K. (2008). A mobility-aware link enhancement mechanism for vehicular ad hoc networks. *EURASIP Journal on Wireless Communications and Networking*, 2008:24:1–24:10.

Ibnkahla, M. (2000). Applications of neural networks to digital communications - a survey. *Elsevier Signal Processing*, 80(7):1185–1215.

Irie, B. and Miyake, S. (1988). Capabilities of three-layered perceptrons. In *IEEE International Conference on Neural Networks*, pages 641–648, San Diego, California, USA.

Jiang, B. and Ravindran, B. (2011). Completely distributed particle filters for target tracking in sensor networks. In *Proceedings of the 25th IEEE International Parallel Distributed Processing Symposium (IPDPS'11)*, pages 334–344, Anchorage, Alaska, USA.

Julier, S. J. and Uhlmann, J. K. (1997). A new extension of the kalman filter to nonlinear systems. In *Proceedings of the International AeroSense Symposium (SPIE'97)*, pages 182–193, Orlando, Florida, USA.

Kaaniche, H. and Kamoun, F. (2010). Mobility prediction in wireless ad hoc networks using neural networks. *Journal of Telecommunications*, 2:95–101.

Kailath, T. (1998). *Linear Systems.* Prentice-Hall information and system sciences series. Prentice Hall International.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *ASME Journal of Basic Engineering.*

Karlsson, R., Schon, T., and Gustafsson, F. (2005). Complexity analysis of the marginalized particle filter. *IEEE Transactions on Signal Processing*, 53:4408–4411.

Karp, B. and Kung, H. T. (2000). Gpsr: greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th ACM Annual International Conference on Mobile Computing and Networking (MobiCom'00)*, pages 243–254, Boston, MA, USA.

Kim, W., Park, J., Yoo, J., Kim, H., and Park, C. G. (2013). Target localization using ensemble support vector regression in wireless sensor networks. *IEEE Transactions on Cybernetics*, 43:1189–1198.

Kim, W., Yoo, J. H., and Kim, H. (2012). Multi-target tracking using distributed svm training over wireless sensor networks. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA'12)*, pages 2439–2444, Saint Paul, Minnesota, USA.

King, T., Füßler, H., Transier, M., and Effelsberg, W. (2006). Dead-reckoning for position-based forwarding on highways. In *Proceedings of the 3rd International Workshop on Intelligent Transportation (WIT'06)*, pages 199– 204, Hamburg, Germany.

Krakiwsky, E. J., Harris, C. B., and Wong, R. V. C. (1988). A kalman filter for integrating dead reckoning, map matching and gps positioning. In *IEEE Symposium in Position Location and Navigation Symposium (PLANS'88)*, pages 39–46.

Labiod, H., Ababneh, N., and García de la Fuente, M. (2010). An efficient scalable trajectory based forwarding scheme for vanets. In *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA'10)*, Perth, Australia.

Langendoen, K. and Reijers, N. (2003). Distributed localization in wireless sensor networks: A quantitative comparison. *Elsevier Computer Networks*, 43:499–518.

Lecun, Y. (1986). *Learning Processes in an Asymmetric Threshold Network*. Springer-Verlag.

Lee, W. and Krumm, J. (2011). Trajectory preprocessing. In Zheng, Y. and Zhou, X., editors, *Computing with Spatial Trajectories*, pages 3–33. Springer New York.

Li, F. and Wang, Y. (2007). Routing in vehicular ad hoc networks: A survey. *IEEE Vehicular Technology Magazine*, 2(2):12–22.

Li, T., Ekpenyong, A., and Huang, Y. (2006). Source localization and tracking using distributed asynchronous sensor. *IEEE Transactions on Signal Processing*, 54:3991–4003.

Li, X. R. and Jilkov, V. P. (2003). Survey of maneuvering target tracking. part i. dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 39:1333–1364.

Li, Y., Ren, W., Jin, D., Hui, P., Zeng, L., and Wu, D. (2014). Potential predictability of vehicular staying time for large-scale urban environment. *IEEE Transactions on Vehicular Technology*, 63:322–333.

Liou, S. and Huang, Y. (2005). Trajectory predictions in mobile networks. *International Journal of Information Technology*, 11:109–122.

Lippmann, R. P. (1987). An introduction to computing with neural nets. *IEEE ASSP Magazine*, 4:4–22.

Liu, G., Lee, B.-S., Seet, B.-C., Foh, C. H., Wong, K. J., and Lee, K.-K. (2004). A routing strategy for metropolis vehicular communications. In *Proceedings of the International Conference on Information Networking (ICOIN'04)*, pages 134–143, Busan, Korea.

Lytrivis, P., Thomaidis, G., Tsogas, M., and Amditis, A. (2011). An advanced cooperative path prediction algorithm for safety applications in vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, 12:669–679.

Maihofer, C. (2004). A survey of geocast routing protocols. *IEEE Communications Surveys Tutorials*, 6(2):32–42.

Maróti, M., Kusy, B., Simon, G., and Lédeczi, Á. (2004). The flooding time synchronization protocol. In *Proceedings of the 2nd ACM international conference on Embedded networked sensor systems (SenSys'04)*, pages 39–49, Baltimore, MD, USA. ACM.

Mo, Z., Zhu, H., Makki, K., and Pissinou, N. (2008). Mobility-assisted location management for vehicular ad hoc networks. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC'08)*, pages 2224–2228, Las Vegas, Nevada, USA.

Müller, K.-R., Smola, A., Rätsch, G., Schölkopf, B., Kohlmorgen, J., and Vapnik, V. (1997). Predicting time series with support vector machines. In *Proceedings of the 7th International Conference on Artificial Neural Networks (ICANN'97)*, pages 999–1004. Springer, Lausanne, SWI.

Nakamura, E. F., Loureiro, A. A. F., and Orgambide, A. F. (2007). Information fusion for wireless sensor networks: Methods, models, and classifications. *ACM Computing Surveys*, 39:1–55.

Niculescu, D. and Nath, B. (2003). Trajectory based forwarding and its applications. In *Proceedings of the 9th ACM Annual International Conference on Mobile Computing and Networking (MobiCom'03)*, pages 260–272, San Diego, CA, USA.

Obradovic, D., Lenz, H., and Schupfner, M. (2006). Fusion of map and sensor data in a modern car navigation system. *Journal of VLSI signal processing systems for signal, image and video technology*, 45(1-2):111–122.

Olfati-Saber, R. (2005). Distributed kalman filter with embedded consensus filters. In *Proceedings of the 44th Conference on Decision and Control - European Control Conference (CDC-ECC'05)*, pages 8179–8184, Seville, Spain.

Papadimitratos, P., De La Fortelle, A., Evenssen, K., Brignolo, R., and Cosenza, S. (2009). Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation. *IEEE Communications Magazine*, 47(11):84–95.

Parker, R. and Valaee, S. (2006). Vehicle localization in vehicular networks. In *Proceedings of the IEEE 63rd Vehicular Technology Conference (VTC'06)*, pages 1–5, Melbourne, Australia.

Peker, A. U., Tosun, O., and Acarman, T. (2011). Particle filter vehicle localization and map-matching using map topology. In *IEEE Intelligent Vehicles Symposium (IV'11)*, pages 248–253, Baden-Baden, Germany.

Ramos, H. S., Boukerche, A., Pazzi, R. W., Frery, A. C., and Loureiro, A. A. F. (2012). Cooperative target tracking in vehicular sensor networks. *IEEE Wireless Communications*, 19:66–73.

Rezende, C. G., Pazzi, R. W., and Boukerche, A. (2009). An efficient neighborhood prediction protocol to estimate link availability in vanets. In *Proceedings of the 7th ACM International Symposium on Mobility Management and Wireless Access (MobiWAC'09)*, pages 83–90, Tenerife, Canary Islands, Spain.

Rosencrantz, M., Gordon, G., and Thrun, S. (2003). Decentralized sensor fusion with distributed particle filters. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI'03)*, Acapulco, Mexico.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. pages 318–362. MIT Press.

S., E. L., Nakamura, E. F., and Oliveira, H. A. B. F. (2009). On the performance of target tracking algorithms using actual localization systems for wireless sensor networks. In *Proceedings of the 12th International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'09)*, pages 418–423, Tenerife, Canary Islands, Spain.

Sapankevych, N. I. and Sankar, R. (2009). Time series prediction using support vector machines: A survey. *IEEE Computational Intelligence Magazine*, 4(2):24–38.

Schön, T., Gustafsson, F., and Nordlund, P. (2005). Marginalized particle filters for mixed linear nonlinear state-space models. *IEEE Transactions on Signal Processing*, 53:2279–2289.

Schubert, R., Richter, E., and Wanielik, G. (2008). Comparison and evaluation of advanced motion models for vehicle tracking. In *Proceedings of the 11th IEEE International Conference on Information Fusion (FUSION'08)*, pages 1–6, Cologne, Germany.

Simic, S. N. (2003). A learning-theory approach to sensor networks. *IEEE Pervasive Computing*, 2:44–49.

Simon, D. (2006). *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley-Interscience.

Skog, I. and Handel, P. (2009). In-car positioning and navigation technologies - a survey. *IEEE Transactions on Intelligent Transportation Systems*, 10:4–21.

Smola, A. J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, 14:199–222.

Souza, É. L., Nakamura, E. F., Oliveira, H. A. B. F., and Figueiredo, C. M. S. (2013). Reducing the impact of location errors for target tracking in wireless sensor networks. *Journal of the Brazilian Computer Society*, 19(1):89–104.

TAPASCologne (2014). Tapascologne project. `http://sourceforge.net/apps/mediawiki/sumo/`.

Tomar, R. S., Verma, S., and Tomar, G. S. (2010). Prediction of lane change trajectories through neural network. In *International Conference on Computational Intelligence and Communication Networks (CICN'10)*, pages 249–253, Bhopal, India.

Tran, D. A. and Nguyen, T. (2008). Localization in wireless sensor networks based on support vector machines. *IEEE Transactions on Parallel and Distributed Systems*, 19:981–994.

Uppoor, S. and Fiore, M. (2011). Large-scale Urban Vehicular Mobility for Networking Research. In *Proceedings of the IEEE Vehicular Networking Conference (VNC'11)*, Amsterdam, Netherlands.

van der Merwe, R., Wan, E. A., and Julier, S. I. (2004). Sigma-Point kalman filters for nonlinear estimation and Sensor-Fusion: Applications to integrated navigation. In *Proceedings of the AIAA Guidance, Navigation & Control Conference*, pages 16–19, Providence, Rhode Island, USA.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory.* Springer-Verlag New York, Inc.

Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999.

Vercauteren, T., Guo, D., and Wang, X. (2005). Joint multiple target tracking and classification in collaborative sensor networks. *IEEE Journal on Selected Areas in Communications*, 23:714–723.

Viani, F., Rocca, P., Benedetti, M., Oliveri, G., and Massa, A. (2010). Electromagnetic passive localization and tracking of moving targets in a wsn-infrastructured environment. *Inverse Problems*, 26.

Wahab, A. A., Khattab, A., and Fahmy, Y. A. (2013). Two-way toa with limited dead reckoning for gps-free vehicle localization using single rsu. In *Proceedings of the IEEE 13th International Conference on ITS Telecommunications (ITST'13)*, pages 244–249, Tampere, Finland.

Welch, G. and Bishop, G. (2001). An introduction to the kalman filter. In *The 28th International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'01)*, Los Angeles, California, USA.

Yousefi, S., Mousavi, M. S., and Fathy, M. (2006). Vehicular ad hoc networks (vanets): Challenges and perspectives. In *Proceedings of the 6th IEEE International Conference on ITS Telecommunications*, pages 761–766, Chengdu, China.