Politecnico di Torino

Master Degree in Communications and Computer Networks Engineering

Department of Electronics and Telecommunications

Collaborative positioning in C-V2X networks

Fabrizio Cuofano

December 2018

Abstract

In the arising development of Intelligent Transportation Systems (ITS), the need for a precise and accurate vehicular positioning system is needed. Traditional Global Navigation Satellite Systems (GNSS) such as GPS do not meet the requirements of many safety ITS applications like collision avoidance or positioning at lane level, which need to rely on a very accurate positioning system, due to their limited availability, continuity and accuracy in challenging, harsh urban environments with high buildings and limited sky visibility.

Among the numerous Cooperative Positioning (CP) solutions proposed in literature to address this problem, a novel approach has been proposed based just on raw GNSS pseudorange measurements in order to estimate the Inter Agent Range (IAR) between two peers in Non-Line-Of-Sight (NLOS) conditions.

Performances of this method will be assessed with a simulation-based approach focusing on the delay between the aid request and the actual receipt of the information from the aiding peer, exploiting the 4G LTE cellular network infrastructure. SimuLTE is used as a simulator for the network, based on the simulation framework provided by OMNeT++, and coupled with SUMO (Simulation of Urban MObility).

TABLE OF CONTENTS

List of Figures

Figure **Page**

CHAPTER 1

INTRODUCTION

Positioning methods relying on Global Navigation Satellite Systems (GNSS) are largely exploited as a technology to estimate the position in mass-market commercial devices like our smartphones or in vehicular applications.
Several systems exist and the most important are:

- the USA NAVSTAR GPS
- the European project Galileo
- the Russian Federation GLONASS
- the Chinese BeiDou system

Even if with slight differences, these traditional positioning systems are based on similar working principles and all of them suffer in their performances due to limited availability of GNSS signals, low accuracy,lack of continuity and limited sky visibility in dense urban environments or undercover and indoor areas.

The main issue in harsh urban environments is the presence of lots of tall obstacles, such as buildings or skyscrapers, which are attenuating the GNSS signal or reflecting it causing a multipath effect and that do not allow a straight Line-of-Sight (LOS) path between the user receiver and the satellites.
This typically results in a loss of reliability for the position estimate or even in a complete fail for a GPS receiver, as it requires the pseudorange measurements -that is an approximation, due to the user clock bias, of the distance between the receiver and

the GNSS satellite- from at least 4 different satellites in LOS.
As a matter of fact, this results in having a system of four linearly independent equations and by solving it the outcome is the spatial position of the user (x, y, z) and the bias of the user clock with respect to the GNSS reference time scale. Consequently, if a user misses even just one satellite, it cannot solve the standard Position, Velocity and Time (PVT) algorithm and thus it doesn't have an estimate for its own position.

1.1 Differential GNSS

The idea of a differential GNSS system is not new. Considering GPS as an example, the first proposals for such a system date back to 1980 [21].
Augmentation systems were firstly designed in order to meet the strict requirements of maritime and aeronautical applications, that were not satisfied by the standard GNSS services in terms of:

- **Accuracy**
- **Availability**, i.e. service coverage
- **Continuity**, i.e. availabilty of the service over time
- **Reliability**, strictly related to the concept of **integrity** that is the probability that the information will not be hazardously misleading for the user

Nowadays these systems are exploited also for other applications such as autonomous driving, terrestrial surveillance of tectonics plate movements and any other safety-related application.
The basic principle is to remove the errors contributions that are common to the user and to a Reference Station (RS) or to a network of RS, that are receiving signals transmitted by satellites and measure the corresponding pseudoranges.
The actual location of a RS is known with geodetic precision, so that the estimated position can be compared with the known one. At this point, the differential correction can be computed and it is typically broadcast to the final users, that may apply it to their pseudoranges measurements, improving the accuracy of the estimated position.

1.1.1 Ground Based Augmentation Systems

A Ground Based Augmentation system is intended primarily to enhance aircraft related operations such as precise approach, departure or terminal movements in an airport area, and it is thus mainly focused on addressing integrity and safety issues, although it also increases the accuracy.
Since the functioning of the augmentation relies on the assumption of error sources correlation between the user and the RS, such as ionospheric and tropospheric delay, it is clear that the efficiency of the correction terms decreases as the distance between the two increases, because those common errors become increasingly de-correlated.
Note that other uncorrelated error contributions such as multipath propagation and the noise introduced by the hardware equipment of the user receiver cannot be cancelled applying corrections and still remain a relevant issue. A common example of a GBAS implementation, is shown in figure 1.1:

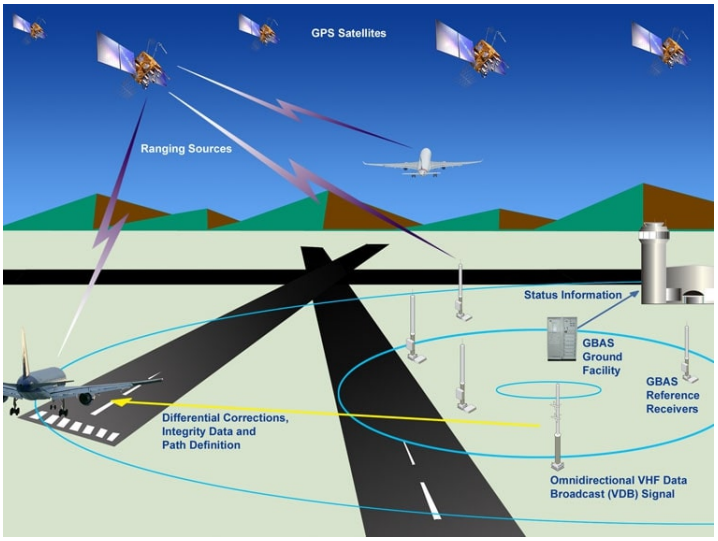


Figure 1.1: Common architecture of a GBAS [4]

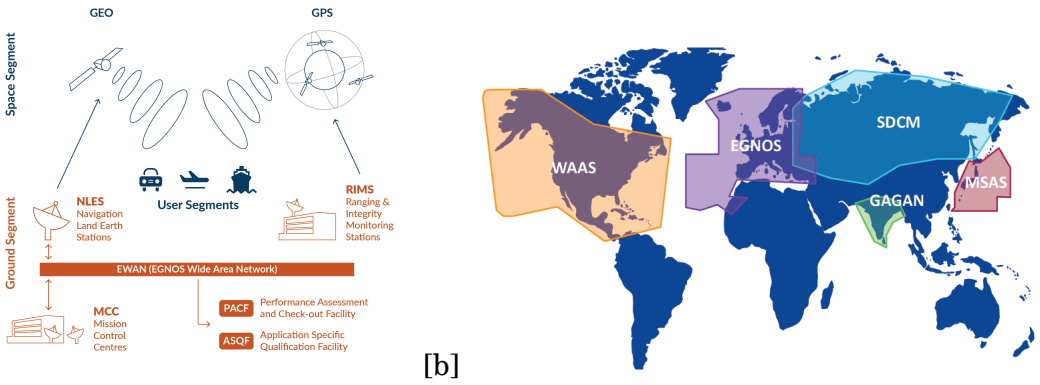
Considering today's GBAS standards, the maximum range is moreover limited by the VHF radio data link transmission to about 55km.

1.1.2 Satellite Based Augmentation Systems

In SBAS, instead of sending the overall error corrections (as in GBAS), the total pseudo-range error is split into its main components and each one is estimated for the entire target region to be covered, that is typically wide. The error components are:

- Clock errors, per satellite
- Ephemeris errors, per satellite
- Ionospheric and tropospheric errors

Due to the large extension of the areas to be covered, in a SBAS is necessarily required a number of geostationary satellites and dedicated network of reference station plus several master stations. As an example it is here reported a snapshot of the European Geostationary Navigation Overlay System (EGNOS):



[a] [b]

Figure 1.2: EGNOS logical topology (a); Existing and planned worldwide SBAS (b)

1.2 A-GNSS

Assisted Global Satellite Navigation System is an augmentation system that relies on the provision of assistance data, based on the existence of a communication link between the user and the source of the data, typically within a conventional cellular network. All of the different techniques have been mainly designed in order to enhance the performances of the so called Time-To-First-Fix (TTFF), i.e. the time required for a receiver to acquire the signals of the satellites in view and to compute the first estimate of the user position, and to increase the sensitivity of the GNSS receiver in a harsh environment.

The basic concept in A-GNSS is to provide, with the assistance of a dedicated server, an external aiding to the Mobile Station (MS) through a wireless network.

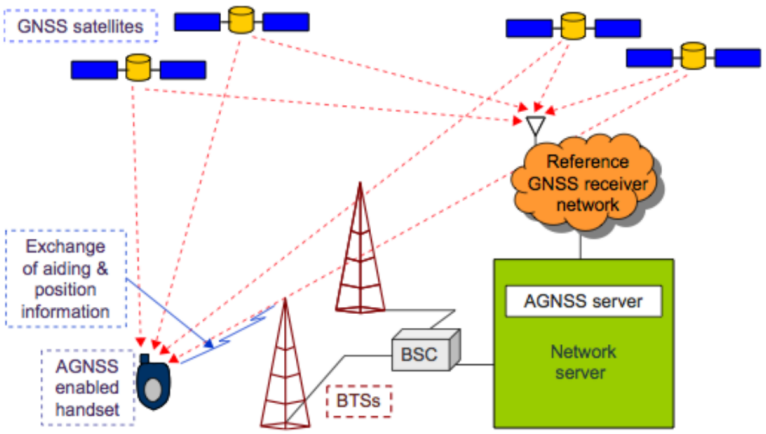


Figure 1.3: Simplified architecture of A-GNSS

Note that a stand-alone GNSS receiver generally needs the orbital information of the satellites in view to compute the current position. These orbital information are stored in the navigation message composed by the ephemeris, that contain satellite-specific orbital information required for real-time coordinate computation and the almanac, that is a simplified version of the ephemeris containing more general and less accurate orbital parameters about all satellites of the entire constellation.

Conisdering A-GPS, navigation data shown in figure 1.4 are transmitted with a data rate of 50 bps, so the bit duration is 20 ms. One word is composed by 30 bits and 10 words create a subframe, that consequently has a duration of 6 s. Every subframe starts with a telemetry word (TLM), used for synchronization purposes, and a handover word (HOW) that contains the 17-bit time of week (ToW) needed for the receiver to calculate the transmission time and the pseudorange, starting from the current GPS week. One frame (also called page) is composed by 5 subframes, thus it is 30 s long. In particular, subframes 1, 2 and 3 contain health status, clock data and ephemeris data. Subframe 4 contain ionospheric model data, offset with UTC time and almanca data, also stored in subframe 5. One superframe is made of 25 frames and thus it is 12.5 minutes long.

For a receiver to download the complete alamanc it takes up to 12.5 minutes, whereas for the ephemeris the time required is up to 30s, which contribute to a large TTFF value, especially in case of a cold start.

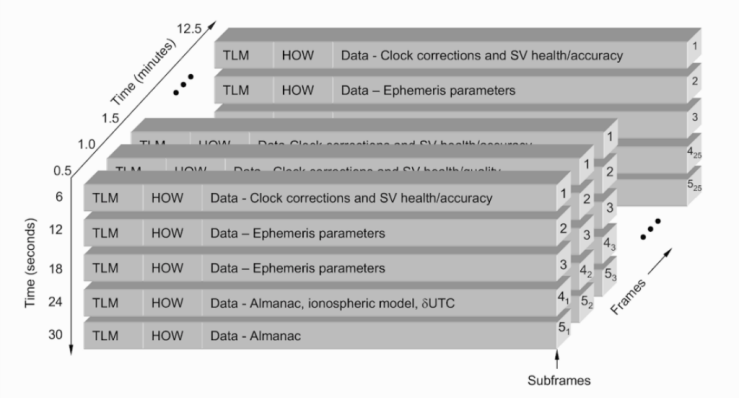


Figure 1.4: Navigation message organization

The assistance data provided through A-GPS typically include:

- Precise ephemeris
- Constellation almanac
- Iono/tropospheric corrections
- Reference position of the MS and reference time
- Acquisition parameters such as Doppler shift and code delay

A-GNSS can work in two different operation modes:

- MS-Assisted mode, that requires only less expensive GPS modules consisting of an antenna, a receiver, and a digital processor for making measurements. The position is calculated by the network that processes the signal samples acquired by the MS receiver and by using the additional data in the server
- MS-Based mode, that requires a fully functional GNSS receiver inside the MS capable to compute position fixes from the measurements, with the help of the assistance information data received from the A-GNSS server

A-GPS could be additionally classified, depending on the type of assistance data provided, frequency or time (either coarse or fine).

In both cases, the TTFF is decreased since the assistance data help the GPS receiver in the acquisition phase, when it has to search for the presence of a given satellite over a 2-D grid called Serach Space (SS). The SS is composed by code delay bins (1023 chips for standard GPS C/A code) and by frequency bins. The Doppler frequency space

is not fixed, but is dependent from the satellite motion, the receiver motion and the receiver local oscillator offset. If the receiver is provided with acquisition parameters (or at least a rough estimation of them), it can significantly reduce the time required for the acquisition as the whole SS to span is reduced.

1.3 Positioning techniques in Cellular networks

Radiolocation of the user and positioning methods were implemented also for mobile networks, historically due in a large extent to the FCC E911 mandate that forced USA telecommunication operators to provide location for emergency situations.
The different proposed methods are typically network-based positioning, i.e. they rely on computations performed by the cell nodes of the cellular network itself, even if some of them may also require handset-based measurements.

1.3.1 Cell-ID

Cell identification, also known as Cell Global Identity (CGI) in GSM networks, is a form of proximity sensing technique, which has the advantage of requiring no additional hardware or upgrade both for the network and the MS. On the other hand, the main drawback of this method is the lack of accuracy and the poor level of performance achievable.

1.3.2 Angle of Arrival

The Angle of Arrival technique is a method used for measuring the angle of arrival, thus the direction of propagation of signals in uplink. Antenna arrays should be used in the network Base Transceiver Stations (BTS) and the communication among them is required, although time synchronization is not needed. One of the main drawbacks is the dependency on a direct LOS path between User Equipment (UE) and BTSs, and the large degradation of accuracy performances with multipath propagation.

1.3.3 Time of Arrival

Another multilateration method is based on the measurement in uplink of the time of arrival, performed by BTSs. In this technique, a tight time synchronization of all

the BTSs and UE is required, plus the need of additional receivers known as Location Measurement Units (LMU), whose deployment in the network might be very expensive.

1.3.4 Time Difference of Arrival

TDoA is the most popular and widespread used positioning technique implemented in cellular networks and in its downlink form it measures the observed time of arrival difference at the user from a number of different BTS. Also this method requires an accurate network synchronization and the employment of additional LMU to calculate clock differences between BTS.
A big drawback common to ToA and TDoA techniques is the so-called herability problem, i.e. the ability for the user to hear and receive signals transmitted from a sufficient number of BTS. As a matter of fact, cellular networks are typically designed in order to minimize the number of BTS required to provide service and to reduce the overlapping among different cells, thus is consequently difficult for a user to receive signals from different base stations.

1.4 Indoor Positioning

Traditional GNSS receivers do not work properly in indoor environments, as the satellite signals are too attenuated to be detected and discriminated from background noise, whereas cellular positioning techniques in general do not provide a sufficient level of accuracy. To overcome the problem of fixing positions inside buildings, a number of different indoor positioning solutions have been proposed, exploiting Wi-Fi networks, RFID, WPAN like Bluetooth or even non radiolocation methods that rely on infrared and ultrasounds.

1.4.1 WLAN Positioning

Wireless Local Area Networks (WLAN) have been standardized by the Instite of Electrical and Electronics Engineers (IEEE) in the 802.11 family of protocols, with several variants like 802.11a/b/g/n. Wi-Fi access points have become massively widespread over the past few years, both in residential areas and in public buildings, making WLAN a viable solution for indoor positioning. Almost every WLAN positioning system that have been developed so far, relies on the measurement of the Received Signal Strength (RSS), the Signal to Noise Ratio (SNR) or the proximity sensing. RSS and SNR measurements both

rely on beacon messages, that can be transmitted either from the mobile terminals in uplink or broadcasted from the WLAN access point in downlink.

Regarding the downlink, mobile Wi-Fi terminals typically perform a passive scanning to detect the available networks and select the best access point, based on the beacons that are periodically transmitted by APs, containing several parameters like a timestamp, supported data rates and versions, and the AP cell identifier, known as Basic Service Set Identifier (BSSI). During passive scanning, terminals are continuously listening to beacons from nearby access points and store their parameters and measurements of RSS and SNR, consequently selecting the best AP available to which connect.
Concering measurements in the uplink, mobile terminals are also able to perform an active scanning and this occurs whenever a terminal does not receive an access point beacon in the passive scanning time interval. In this situation, mobile terminals have to generate explicit probe requests to which all the nearby access points answer with a beacon and then the proper AP is selected.

Since measurements can be observed either in downlink or uplink, with passive and active scanning, it is possible to implement both terminal-based and terminal-assisted positioning. In any case all the WLAN localization techniques use proximity sensing, trilateration or fingerprinting positioning methods.

1.4.1.1 WLAN proximity sensing

In this technique, the mobile terminal position is assessed from the position of the access point, similarly to what happens for cellular identification of 1.3.1. The proximity sensing is the easiest method to be implemented, as it only requires a mapping between AP's BSSI and rooms in the indoor environment, but on the other side it has low accuracy levels, especially in buildings with different floors.

1.4.1.2 WLAN trilateration

The position is estimated from lateration by using the distances between the mobile terminal and access points, that are determined by the beacon transmission path loss. Indoor trilateration require an accurate alignment in the positions of the access points in the building, with respect to a given coordinate reference system, possibly local. This method typically suffers from multipath progapation, that is caused by the beacon's reflection and scattering on walls, ceilings and objects inside an indoor environment. The resulted attenuation is very hard to be predicted and it corrupts the path loss measurement, thus corrupting also the derived estimated position. As an additional

drawback, the path loss computation requires not only the RSS, but also the transmitted signal power, that requires more signalling overhead.

1.4.1.3 WLAN fingerprinting

Fingerprinting is a radiolocation method based on pattern matching, that consists in determining the terminal position starting from the propagation characteristic of emitted radio signals in a given area to be mapped. The technique consists of two different phases, one off-line and the other one online. The first off-line phase is the radiomapping, also known as calibration or training, where the location of interest is divided into a grid (regular or not) and for every point inside it, the observer registers the RSS from multiple base stations, resulting in an array of values for each reference position. During the second online phase, the measurements of the RSS experienced by the terminal from the access points in radio range are compared with all the entries stored in the reference database and the position is chosen according to the best match. The fingerprinting method can be applied both in terminal-assisted or terminal-based mode, depending on whether the off-line calibration is performed from RSS measurements in uplink or downlink, respectively.
A significant drawback of the fingerprinting method is the overhead to create radiomaps in the off-line calibration phase, since carrying measurements in a close grid to completely cover a whole building is a large time consuming process. Moreover, the training process has to be repeated every time that the position in the configuration of just one of the access points in the indoor environment is changed. To overcome this problem, several WLAN fingerprinting solutions have been proposed, where radiomaps are derived from mathematical models or created from measurements with a statistical approach, achieving quite good accuracy performances.

1.4.2 RFID Positioning

Radio Frequency Identification is a quite common technology used in many different applications such as retail, access control, advertising, transportation and logistics, passports, toll collection, industry automation and many other. This technology is essentially based on the exchange of electromagnetic signals between a reader and a tag.
An RFID tag can either be active or passive, depending if it is equipped with a battery or simply passively stimulated by radio signals. Active tags typically also have more memory, computational power and a larger radio range, but they are more expensive

then the cheap passive tags.
The number of solutions proposed in literature for indoor RFID positioning mainly exploit proximity sensing or alternatively rely on RSS measurements or fingerprinting and some of them can achieve quite a fine-grained degree of accuracy, but the main disadvantage of those systems is the large number of tags typically required, resulting in high deployment costs for large-scale areas. Solutions like [26] have also been proposed for RFID positioning in vehicular environments, for lane level and road segment positioning, but still the main issue is the huge number of tags needed to cover highways or entire city roads, that is not economically feasible on a wide coverage area.

1.5 Integrated and hybrid positioning

The A-GPS system itself depicted in 1.2 can be considered as an integration solution to overcome limitations of the standard GPS service. In order to address the problem of positioning in harsh environments, a number of hybrid solutions with the integration of different positioning technologies were proposed over the past years.

In [10] it is proposed a system combining GPS, Wi-Fi positioning based on RSS fingerprinting and vehicle kinematics. In [18] standard GNSS is tightly coupled with visual odometry data from an on-board vehicle camera. Other solutions like the one proposed in [11] are based on the integration of Inertial Navigation Systems (INS). INS are navigation aid systems that typically exploits rotation and motion sensors, e.g. gyroscopes and accelerometers, in order to continuously track the position, the orientation and velocity of a dynamic target object. Those systems are typically combined with GNSS with different degrees of integration, from loose to tight and ultra-tight and can significantly improve positioning accuracy and continuity [6]. Another system that has been integrated for positioning purposes [5] with GNSS/INS is the Ligth Detection and Ranging (LiDAR), an optical range estimation technology similar to a radar system, that uses light instead of radio signals, typically by means of a laser emitting pulses at a given wavelength. A fusion technique with a different terrestrial ranging measurements provided by Ultra Wide Band (UWB) units has been presented in [25]. The approach of [17] is instead based on the tight fusion of radio signals of opportunity (SOP). This navigation technique exploits a number of different signals, such as GNSS, cellular, television and Wi-Fi signals -depending from the ones available in the environment- considering each one as potentially useful for extracting positioning or timing data information.

1.6 Considerations

In 1 different collaborative positioning techniques already existing have been presented. Differential GNSS systems, although generally improving reliability and accuracy compared to the standard GNSS service, are not well suited for vehicular applications and urban environments with NLOS conditions and multipath propagation.
The SBAS work by broadcasting corrections to the users through dedicated geostationary satellites and the absence of an adequate sky view in urban canyons limits the beneficial usage of this augmentation system. Similar issues arise for GBAS, that are mainly designed to properly work in airports for aircrafts in close airfield range, not for vehicles in city roads.

The A-GPS is enhancing the TTFF, reducing the time required for a position fix, but even this system suffers from the absence of a direct LOS propagation link in dense urban canyons and consequently the ultimate accuracy is still not sufficient.

Concerning positioning methods implemented in cellular networks, they typically do not achieve a level of accuracy suitable for vehicular application in urban areas, not to mention the fact that they are quite costly to be implemented.

WLAN and RFID positioning methods have been designed to address the problem of locating a terminal in an indoor environment and even though solutions for lane level positioning exist, these are not really a feasible solution due to the large number of RFID tags required.

Integrated systems are based on the coupling and data fusion of different technologies like GNSS, INS, Wi-Fi, LIDAR, visual odometry and SOP. Although these solutions might provide an enhancement compared to the standard GNSS positioning performances, they always require additional measurements, sensors and dedicated hardware, implying that general low-cost mass-market GNSS receivers (like in our personal devices) are not sufficient to implement such techniques.

CHAPTER 2

COLLABORATIVE POSITIONING BASED ON GNSS INTER VEHICLE RANGING

This chapter deals with the presentation of a novel collaborative position method, particularly a range estimation technique that is relying only on pseudorange measurements provided by a standard GNSS receiver, in order to estimate the Inter Agent Range (IAR) between two peers in a scenario of Non-Line-Of-Sight (NLOS).
Additionally, other significant related ranging techniques based on GNSS measurements, that might be integrated into Cooperative Positioning algorithms will be here presented.

2.1 Preliminary overview

The interest for positioning-based applications in the framework of Intelligent Transportation Systems (ITS) has increasingly risen over the past few years, especially concerning vehicular applications. As shown in 1, the standard GPS is not suitable for harsh urban environments due the limitations in availability, continuity and accuracy of the positioning service. Although a number of solutions have been proposed in literature to address the problem, these typically require the integration of additional specialized hardware architectures in order to implement collaborative positioning.

The collaborative method proposed in [12] is based on a hybrid algorithm that integrates standard GNSS pseudorange measurements together with the estimation of the NLOS range between couples of peers, having simultaneously the view of a satellite

in common.

In urban canyons and more in general in challenging scenarios where signal blockage, NLOS with satellites and multipath propagation are frequent, the effect on GNSS receivers tracking satellites results in a temporary outage of the positioning service, since the number of visible satellite is not sufficient to solve the standard Position, Velocity and Time (PVT) algorithm. As a matter of fact, it is well known from GNSS fundamentals that a receiver needs at least four satellites in order to estimate its position, whose quality is moreover affected by a number of error sources and by the geometrical distribution of the satellites acquired. When a receiver does not have a sufficient number of satellites in view, it is possible to exploit the IAR additional measurement in a properly designed algorithm to estimate a position fix.

2.2 Fundamentals of GNSS position estimation

In a GNSS receiver, the estimation of the user position is based on the measurement of the satellite to user range. This geometric distance, can be exactly known only if both the receiver and the all the satellites are perfectly synchronized with the GNSS time reference scale. However reaching such an accurate timing synchronization at a low cost and reasonable complexity for mass-market commercial GNSS receivers would be impossible, as such class of devices are typically equipped with cheap oscillators. The most widespread and less expensive receivers are based on code phase measurements, i.e. rely on the estimation of the propagation time Δt of the signal transmitted from the satellite to the user, that is done comparing the received signal with a replica locally generated by the receiver. If all the oscillators were perfectly synchronized, the geometric satellite to user range would be computed as:

$$r = c\Delta t \quad \text{where } c \text{ is the speed of light} \tag{2.1}$$

Due to the aforementioned timing impairments, the actual measurement is not really a range and it is thus referred to as pseudorange:

$$\rho = r + c(\delta t_u - \delta t_s) \tag{2.2}$$

where δt_u and δt_s represent the time offset with respect to the actual GNSS time scale of the user and the satellite clocks, respectively. The clock mounted on-board of a satellite

is based on a expensive and highly accurate atomic clock, so that its bias is typically low. Additionally, the GNSS control segment determines its offset w.r.t. the actual time scale and the transmit corrections to satellites, which then broadcast the information to the users in order to compensate for it, so that δt_s can be considered negligible and the previous equation can be written as:

$$\rho = r + c\delta t_u \tag{2.3}$$

2.2.1 PVT linearization solution

In order to estimate the user position vector $\mathbf{x}_u = (x_u, y_u, z_u)$ in a three dimensional reference system and the receiver clock offset δt_u it is required to have at least four pseudorange measurements, whose generic expression is:

$$\begin{aligned} \rho_j = r_j + c\delta t_u &= \sqrt{(x_j - x_u)^2 + (y_j - y_u)^2 + (z_j - z_u)^2} + c\delta t_u \quad \text{with} j \in [1,4] \\ &= f(x_u, y_u, z_u, \delta t_u) = f(\mathbf{x}_u, \delta t_u) \end{aligned} \tag{2.4}$$

where j denotes the jth satellite in view with its three components position vector and $\mathbf{x}_u$ the positioning state vector of the user.

This system of equations is a non-linear problem that can be solved by means of iterative techniques based on linearization [7]. The generic pseudorange can be approximated by expanding it in a Taylor series around a given known location, that is taken as a linearization point with coordinates $\hat{\mathbf{x}}_u = (\hat{x}_u, \hat{y}_u, \hat{z}_u)$. With the approximate position, the approximate pseudorange can be expressed from 2.4 as follows:

$$\begin{aligned} \hat{\rho}_j &= \sqrt{(x_j - \hat{x}_u)^2 + (y_j - \hat{y}_u)^2 + (z_j - \hat{z}_u)^2} + c\hat{\delta} t_u \\ &= f(\hat{x}_u, \hat{y}_u, \hat{z}_u, \hat{\delta} t_u) = f(\hat{\mathbf{x}}_u, \hat{\delta} t_u) \end{aligned} \tag{2.5}$$

The offset of the unknown position of the user can be expressed as a displacement with respect to the known location taken as an approximation:

$$\begin{aligned} \Delta x_u &= x_u - \hat{x}_u \\ \Delta y_u &= y_u - \hat{y}_u \\ \Delta z_u &= z_u - \hat{z}_u \\ \Delta t_u &= \delta t_u - \hat{\delta} t_u \end{aligned} \tag{2.6}$$

The generic pseudorange can be written as a function of the user position and its clock bias or it can be equivalently expressed in terms of the offset with the known location:

$$\rho_j = f(x_u, y_u, z_u, \delta t_u) = f(\hat{x}_u + \Delta x_u, \hat{y}_u) + \Delta y_u, \hat{z}_u + \Delta z_u, \hat{\delta t}_u + \Delta t_u) \tag{2.7}$$

The last term of the expression can be approximated through a Tayolor series expansion, truncated after the first order partial derivative to conserve linearity, as follows:

$$\begin{aligned} \rho_j = f(\hat{\mathbf{x}}_u, \hat{\delta t}_u) + \frac{\partial f(\hat{\mathbf{x}}_u, \hat{\delta t}_u)}{\partial \hat{x}_u} \Delta x_u \quad + \frac{\partial f(\hat{\mathbf{x}}_u, \hat{\delta t}_u)}{\partial \hat{y}_u} \Delta y_u + \\ + \frac{\partial f(\hat{\mathbf{x}}_u, \hat{\delta t}_u)}{\partial \hat{z}_u} \Delta z_u \quad + \frac{\partial f(\hat{\mathbf{x}}_u, \hat{\delta t}_u)}{\partial \hat{\delta t}_u} \Delta t_u + \dots \end{aligned} \tag{2.8}$$

Denoting the geometrical range between satellite j and the known linearization point as:

$$\hat{r}_j = \sqrt{(x_j - \hat{x}_u)^2 + (y_j - \hat{y}_u)^2 + (z_j - \hat{z}_u)^2} \tag{2.9}$$

The first order partial terms can be calculated explicitly:

$$\begin{aligned} \frac{\partial f(\hat{\mathbf{x}}_u, \hat{\delta t}_u)}{\partial \hat{x}_u} &= \frac{\partial \hat{\rho}_j}{\partial \hat{x}_u} = -\frac{x_j - \hat{x}_u}{\hat{r}_j} \\ \frac{\partial f(\hat{\mathbf{x}}_u, \hat{\delta t}_u)}{\partial \hat{y}_u} &= \frac{\partial \hat{\rho}_j}{\partial \hat{y}_u} = -\frac{y_j - \hat{y}_u}{\hat{r}_j} \\ \frac{\partial f(\hat{\mathbf{x}}_u, \hat{\delta t}_u)}{\partial \hat{z}_u} &= \frac{\partial \hat{\rho}_j}{\partial \hat{z}_u} = -\frac{z_j - \hat{z}_u}{\hat{r}_j} \\ \frac{\partial f(\hat{\mathbf{x}}_u, \hat{\delta t}_u)}{\partial \hat{\delta t}_u} &= \frac{\partial \hat{\rho}_j}{\partial \hat{\delta t}_u} = c \end{aligned} \tag{2.10}$$

At this point it is possible to rewrite (2.8) substituting the partial terms:

$$\rho_j = \hat{\rho}_j - \frac{x_j - \hat{x}_u}{\hat{r}_j} \Delta x_u - \frac{y_j - \hat{y}_u}{\hat{r}_j} \Delta y_u - \frac{z_j - \hat{z}_u}{\hat{r}_j} \Delta z_u + c\Delta t_u \tag{2.11}$$

That can be further reordered as:

$$\Delta\rho_j = \hat{\rho}_j - \rho_j = h_{j,x} \Delta x_u + h_{j,y} \Delta y_u + h_{j,z} \Delta z_u - c\Delta t_u \tag{2.12}$$

where the partial derivative terms in (2.10) were simplified by introducing the new variables:

$$
\begin{aligned}
h_{j,x} &= \frac{x_j - \hat{x}_u}{\hat{r}_j} \\
h_{j,y} &= \frac{y_j - \hat{y}_u}{\hat{r}_j} \\
h_{j,z} &= \frac{z_j - \hat{z}_u}{\hat{r}_j}
\end{aligned}
\tag{2.13}
$$

The three components above actually represent the direction components of the unitary steering vector departing from the linearization point and directed towards the *jth* satellite, defined as:

$$\mathbf{h}_{j,\hat{u}} = (h_{j,x}, h_{j,y}, h_{j,z})$$

Considering that there are four unknowns $(\Delta x_u \Delta y_u \Delta z_u \Delta t_u)$ the minimum number of satellites required for the user to estimate its position is four, with the resulting linearized system composed of the pseudoranges:

$$
\begin{cases}
\Delta\rho_1 = h_{1,x}\Delta x_u + h_{1,y}\Delta y_u + h_{1,z}\Delta z_u - c\Delta t_u \\
\Delta\rho_2 = h_{2,x}\Delta x_u + h_{2,y}\Delta y_u + h_{2,z}\Delta z_u - c\Delta t_u \\
\Delta\rho_3 = h_{3,x}\Delta x_u + h_{3,y}\Delta y_u + h_{3,z}\Delta z_u - c\Delta t_u \\
\Delta\rho_4 = h_{4,x}\Delta x_u + h_{4,y}\Delta y_u + h_{4,z}\Delta z_u - c\Delta t_u
\end{cases}
\tag{2.14}
$$

Such system can be rearranged in a more compact matrix form defining:

$$
\boldsymbol{\Delta\rho} = \begin{bmatrix} \Delta\rho_1 \\ \Delta\rho_2 \\ \Delta\rho_3 \\ \Delta\rho_4 \end{bmatrix}
\qquad
\mathbf{H} = \begin{bmatrix} h_{1,x} & h_{1,y} & h_{1,z} & 1 \\ h_{2,x} & h_{2,y} & h_{2,z} & 1 \\ h_{3,x} & h_{3,y} & h_{3,z} & 1 \\ h_{4,x} & h_{4,y} & h_{4,z} & 1 \end{bmatrix}
\qquad
\boldsymbol{\Delta x} \begin{bmatrix} \Delta x_u \\ \Delta y_u \\ \Delta z_u \\ -c\Delta t_u \end{bmatrix}
$$

That yelds to the compact notation:

$$\boldsymbol{\Delta\rho} = \mathbf{H}\boldsymbol{\Delta x} \tag{2.15}$$

Whose solution can be obtained by inverting the matrix $\mathbf{H}$, given that it is non-singular:

$$\mathbf{\Delta x} = \mathbf{H}^{-1}\,\mathbf{\Delta \rho} \tag{2.16}$$

As soon as the vector $\mathbf{\Delta x}$ is computed, the user can retrieve its own position coordinates and its time offset exploiting the relationship (2.6) with the linearization point. The degree of accuracy of the position estimate is strictly related with the choice of the linearization location and the consequent coordinate offset with the user.
If the displacement results above a give threshold, which is determined by the accuracy requirements of the user, the receiver will iteratively repeat the process by taking the last position estimate as linearization point, until the displacement is within an acceptable value.
If the receiver has more than four visible satellites, namely n, the linear system of (2.13) will be overdetermined and the matrix $\mathbf{H}$ will be in the form:

$$\mathbf{H} = \begin{bmatrix} h_{1,x} & h_{1,y} & h_{1,z} & 1 \\ h_{2,x} & h_{2,y} & h_{2,z} & 1 \\ h_{3,x} & h_{3,y} & h_{3,z} & 1 \\ h_{4,x} & h_{4,y} & h_{4,z} & 1 \\ \vdots & \vdots & \vdots & 1 \\ h_{n,x} & h_{n,y} & h_{n,z} & 1 \end{bmatrix} \tag{2.17}$$

In such a case, it is necessary to use a least square solution in order to obtain the unknown variables. In particular a solution is given by the expression:

$$\mathbf{\Delta x} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\,\mathbf{\Delta \rho} \tag{2.18}$$

Provided that $\left(\mathbf{H}^T\mathbf{H}\right)$ is square and invertible.

2.3 Inter Agent Range estimation

In order to present the working principle of the collaborative positioning method proposed in [12], it is here considered the basic geometric scenario depicted in the following figure:

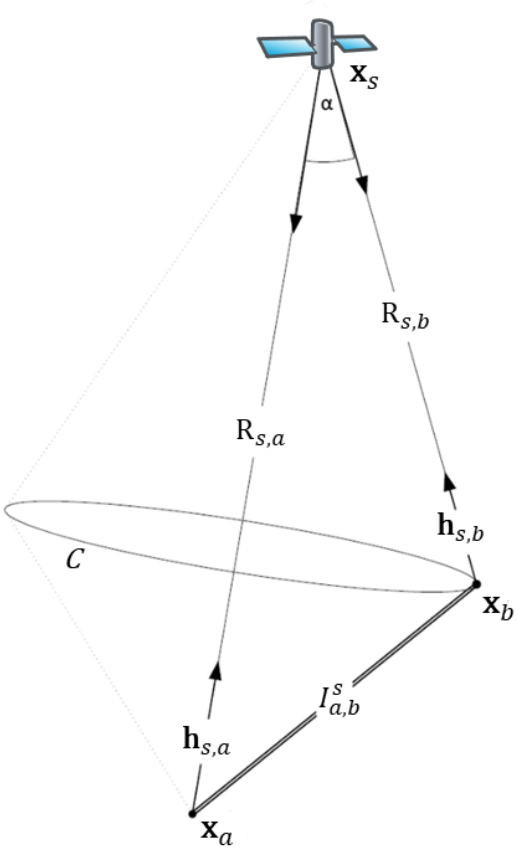


Figure 2.1: Geometrical reference scenario for the Inter Agent Range estimation

In this static scenario there are two ground peers equipped with mass-market GNSS receivers. Both the receivers have one satellite in common in Line-of-Sight, s.
One of the two, named the aiding peer $\boldsymbol{a}$, has a sufficient number of satellites to perform the PVT computation and to estimate its own position. The other receiver, the aided peer $\boldsymbol{b}$, has instead an underdetermined set of equations due to an insufficient number of visible satellites and cannot obtain a position estimate.

The last estimated position fix for both the peers is the positioning state vector within a

given reference system:

$$\mathbf{x}_i = (x_i, y_i, z_i), \quad \text{with } i \in [a, b] \tag{2.19}$$

Similarly the position of the satellite in common s in a Earth Centered Earth Fixed (ECEF) reference frame is:

$$\mathbf{x}_s = (x_s, y_s, z_s) \tag{2.20}$$

The satellite-to-receiver range is computed as defined in Section 2.2.1 between the satellite and both the receivers:

$$\mathbf{r}_{s,i} = \mathbf{x}_s - \mathbf{x}_i \qquad \text{with } i \in [a, b]$$

$$r_{s,i} = \|\mathbf{x}_s - \mathbf{x}_i\| = \sqrt{(x_s - x_i)^2 + (y_s - y_i)^2 + (z_s - z_i)^2} \qquad \text{with } i \in [a, b]$$

where $r_{s,i}$ denotes the magnitude of the range vector.
The unitary steering vectors, which directed from the receiver towards the satellite s are then defined as:

$$\mathbf{h}_{s,i} = \frac{\mathbf{x}_s - \mathbf{x}_i}{\|\mathbf{x}_s - \mathbf{x}_i\|} = \frac{\mathbf{r}_{s,i}}{r_{s,i}} \quad \text{with } i \in [a, b] \tag{2.21}$$

Note that as shown in Section 2.2.1 the PVT computation typically requires linearization techniques to obtain a valid solution. The unitary steering vector components are thus computed with respect to the chosen linearization point similarly to equation (2.13):

$$\begin{aligned} h_{s,x_i} &= \frac{x_s - \hat{x}_i}{\hat{r}_{s,i}} \\ h_{s,y_i} &= \frac{y_s - \hat{y}_i}{\hat{r}_{s,i}} \\ h_{s,z_i} &= \frac{z_s - \hat{z}_i}{\hat{r}_{s,i}} \end{aligned} \tag{2.22}$$

Where the linearization point for the receiver i is:

$$\hat{\mathbf{x}}_i = (\hat{x}_i, \hat{y}_i, \hat{z}_i) \tag{2.23}$$

And the satellite to user range expressed w.r.t. the linearization point is:

$$\hat{r}_{s,i} = \sqrt{(x_s - \hat{x}_i)^2 + (y_s - \hat{y}_i)^2 + (z_s - \hat{z}_i)^2} \tag{2.24}$$

So that the user i, having j satellites in view, can build the matrix for the PVT computation:

$$\mathbf{H}_i = \begin{bmatrix} h_{1,x_i} & h_{1,y_i} & h_{1,z_i} & 1 \\ h_{2,x_i} & h_{2,y_i} & h_{2,z_i} & 1 \\ \vdots & \vdots & \vdots & \\ h_{j,x_i} & h_{j,y_i} & h_{4,z_i} & 1 \end{bmatrix} \tag{2.25}$$

The angle α subtended between the two receivers with respect to the shared satellite s can thus be computed as:

$$\alpha = \arccos\left(\frac{\mathbf{h}_{s,a} \cdot \mathbf{h}_{s,b}}{|\mathbf{h}_{s,a}|\,|\mathbf{h}_{s,b}|}\right) \tag{2.26}$$

At this point, the Inter Agent Range between the two receivers can be calculated from the angle α by means of the trigonometric law of cosines, that allows to compute the unknown side of the triangle with vertexes the shared satellite and the two agents:

$$I^s_{a,b} = \sqrt{r^2_{s,a} + r^2_{s,b} - 2 r_{s,a} r_{s,b} \cdot \cos(\alpha)} \tag{2.27}$$

A very important remark is the fact that the IAR estimation is just relying on geometrical assumptions rather than traditional range estimation where an actual measurement of the range has to be performed, e.g. by means of UWB ranging modules, LiDAR, radar or other terrestrial ranging sensors. As a consequence, the traditional constraint of having a direct LOS propagation path between the two collaborating GNSS receivers is overcome.
Notice also that the privacy of the aiding peer b is preserved. As a matter of fact, if the aided peer receives the included angle α and the current satellite to receiver range

magnitude $r_{s,b}$ it is impossible for him to know the exact vector position $\mathbf{x}_b$.
The position of user b is known just with ambiguity to user a, as it can be noticed from figure 2.1, where it is shown the circumference C, that is the locus of all the points at distance $r_{s,b}$ from the shared satellite and $I^s_{a,b}$ from b. This property holds since the dot product of (2.26) is performed by b then sent back to receiver a, which cannot retrieve the components -i.e. the orientation- of the unitary steering vector $\mathbf{h}_{s,b}$ as it can be proofed that the operation is not invertible.
As a conceptual idea, considering a generic vector $\mathbf{h} = (h_x, h_y, h_z)$ there is an infinite number of triplets satisfying the condition over the unitary magnitude:

$$\sqrt{x^2 + y^2 + z^2} = 1 \tag{2.28}$$

2.3.1 The hybrid positioning method

As pointed out in section ERROR SOURCES, in real environments the pseudoranges measurements performed by a GNSS receiver are corrupted by several error sources that propagate in the PVT algorithm, affecting the final accuracy of the user position estimate. These errors can be statistically modelled as random variables independent and identically distributed, according to a Gaussian distribution with zero mean and variance σ^2_{UERE}. The IAR estimation can be extended to a more generic, dynamic scenario, by taking into account the effects of real noisy measurements. From the heuristic analysis conducted in [12], also the estimates of the IAR can be modelled as random variables by:

$$\hat{I}_{a,b} = I_{a,b} + \varepsilon_I$$

where ε_I is the IAR error, characterized as a zero mean Gaussian random variable with variance $\kappa \cdot \sigma_{UERE}$:

$$\varepsilon_I \sim \mathcal{N}(0, \kappa \cdot \sigma_{UERE})$$

Similarly to what done for the pseudoranges in the PVT linearization solution, it is possible to define the displacement for IAR estimates as:

$$\Delta I_{a,b} = \hat{I}_{a,b} - I_{a,b} \tag{2.29}$$

Recall from previous section 2.2.1 that generally the PVT algorithm provides a

solution by solving equation (2.16):

$$\mathbf{\Delta\rho} = \mathbf{H}\,\mathbf{\Delta x}$$

The hybrid solution proposed in [12] in based on the integration of the standard pseudorange measurements with the additional data of IAR estimates. In particular, considering that the aided agent a has j visible satellites, the integration relies on the the usage of matrix composition in order to merge the available assistance data, with the standard computations. The expression above is thus modified as follows:

$$\left[\frac{\Delta\rho_{j,a}}{\Delta I_{a,b}}\right] = \mathbf{H}^{*}_{\mathbf{a}}\,\mathbf{\Delta x} \qquad \text{with} \quad \mathbf{H}^{*}_{\mathbf{a}} = \left[\frac{\mathbf{H}_a}{\mathbf{H}_{IAR}}\right] \tag{2.30}$$

where $\mathbf{H}_a$ is the matrix of (2.25), computed by the aided receiver a, whereas $\mathbf{H}_{IAR}$ is is the matrix computed from the linearization, through the same linearization point exploited for the computation of $\mathbf{H}_a$, of IAR additional data as:

$$\widetilde{\mathbf{h}}_{b,\hat{\mathbf{x}}_a} = \left(h_{b,\hat{x}_0}, h_{b,\hat{y}_0}, h_{b,\hat{z}_0}\right) \tag{2.31}$$

where $\hat{\mathbf{x}}_a = (\hat{x}_0, \hat{y}_0, \hat{z}_0)$ is the linearization point used in the standard PVT algorithm by the agent a.

The IAR estimates provided by all the receiver that are under GNSS service are collected altogether and integrated into the matrix $\mathbf{H}^{*}_{\mathbf{a}}$. It is possible however that close aiding neighbours provide IAR estimates which are strongly correlated, resulting in an ill-conditioned matrix and a consequent low accuracy of the estimated position fix. In order to overcome this issue and to improve the convergence performance of the solution, a self-adaptive iterative algorithm following the approach shown in [cite SAIA] is here employed.

Summarizing, the complete step by step procedure exploited in the presented collaborative algorithm based on the IAR estimation is reported:

- the aided receiver experiencing a GNSS service outage sends to the aiding receiver

b its last known steering vector $\mathbf{h}_{s,a}$ towards the shared satellite s;

- the agent b computes the included angle α exploiting equation (2.26);
- b sends back to the aided agent a the computed angle $alpha$ plus its current satellite-to-receiver range $r_{s,b}$;
- a computes the IAR estimate $I^s_{a,b}$ applying equation (2.27);
- a computes the hole matrix $\mathbf{H}^*_\mathbf{a}$ integrating its $\mathbf{H}_a$ with IAR linearized data in $\mathbf{H}_{IAR}$
- a finally estimates its own position by solving equation (2.30) with a weighted least mean square self-adaptive iteration algorithm for convergence.

2.4 Other GNSS based ranging techniques

In the increasing interest on the domain of Intelligent Transportation Systems (ITS) and vehicular applications, a fundamental role is the ability of providing to the connected cars a precise and almost real-time positioning. Conventional GNSS systems like GPS generally do not meet the requirements of applications such as collision warning, lane level positioning or safety critical applications, especially in urban environments. One of the proposed paradigms to overcome standard GNSS service limitations is the Collaborative Positioning approach, where two peers or more actively share relevant information. Considering inter vehicle ranging techniques that are based just on the exploitation of GNSS code pseudorange measurements [20], other methods different from the IAR algortihm have been proposed, such as:

- Absolute Position Differencing
- Raw Pseudoranges
- Single Differencing
- Double Differencing

2.4.1 Absolute Position Differencing

In the APD technique, both the GNSS receivers are allowed to exchange their absolute position in terms of the positioning state vector, obtained through the Least Square

solution presented in 2.2.1. Having the absolute position of the other receiver, the inter-vechicle range can be simply calculated as:

$$I_{a,b} = \|\hat{\mathbf{x}}_a - \hat{\mathbf{x}}_b\| = \sqrt{(\hat{x}_a - \hat{x}_b)^2 + (\hat{y}_a - \hat{y}_b)^2 + (\hat{z}_a - \hat{z}_b)^2} \tag{2.32}$$

where $\hat{\mathbf{x}}_{a,b}$ is the last estimated position for both the receivers.

2.4.2 Raw Pseudoranges

In RR technique, raw pseudorange measurements are used instead of the absolute position, without any data differencing. Considering two generic receivers, the pseudorange measurement to satellite j of one user can be expressed in terms of a displacement from the position of the other one:

$$\mathbf{x}_b = \mathbf{x}_a + \Delta\mathbf{x} \tag{2.33}$$

where the components of $\Delta\mathbf{x} = (\Delta x, \Delta y, \Delta z)$ represent the offset of user b with respect to user a in the three coordinates.

Exploiting this displacement relationship, the generic pseudorange from satellite j for both the receivers can be written as:

$$\begin{aligned} \rho_{j,a} &= \left\|\mathbf{x}_j - \mathbf{x}_a\right\| + c\delta t_a + \varepsilon j, a \\ \rho_{j,b} &= \left\|\mathbf{x}_j - \mathbf{x}_b\right\| + c\delta t_b + \varepsilon j, b = \\ &= \left\|\mathbf{x}_j - \mathbf{x}_a - \Delta\mathbf{x}\right\| + c\delta t_b + \varepsilon j, b \end{aligned} \tag{2.34}$$

Those pseudoranges can be approximated around a given linearization point $\hat{\mathbf{x}}_u$ through a Taylor series expansion and can be rewritten by using the definition of unitary steering vector. Assuming to have N satellites in view, shared by both the GNSS receivers, the aided agent a will have its set of pseudranges plus the measurements provided by the other receiver, for a total of $2N$ equations. Following the approach detailed in [20], those equations can be properly combined and solved through the usual least square solution, that eventually yelds to the estimation of the displacement vector $\Delta\hat{\mathbf{x}}$. The inter agent range is then calculated as:

$$I_{a,b} = \|\hat{\mathbf{x}}\| = \sqrt{(\Delta\hat{x})^2 + (\Delta\hat{y})^2 + (\Delta\hat{z})^2)} \tag{2.35}$$

Note that following this approach, the number of equations in the PVT is perfectly doubled, so that also the number of unknowns is doubled, increasing from 4 to 8. This implies having at least eight linearly independent equations and four satellites simultaneously in view for both the GNSS receivers.
This condition is typically not possible in harsh environments such as urban canyons or dense urban areas, where the PVT set of equations results usually undetermined.

2.4.3 Single Differencing of Pseudoranges

The SD approach is similar to the one used in AD, but instead of differentiating the absolute positions, the aided peer a receives the pseudorange measurement of user b and it computes a new estimate as a difference with respect to its one as follows:

$$\begin{aligned} SD^{j}_{ab} &= \rho_{j,a} - \rho_{j,b} = \\ &= r_{j,a} + c\delta t_a + \varepsilon_{j,a} - r_{j,b} - c\delta t_b - \varepsilon_{j,b} = \\ &= \Delta r_{j,ab} + c\Delta t_{ab} + \varepsilon_{r_j,ab} \end{aligned} \tag{2.36}$$

where the following substitutions have been used:

$$\begin{aligned} \Delta r_{j,ab} &= \rho_{j,a} - \rho_{j,b} \\ c\Delta t_{ab} &= c\delta t_a - c\delta t_b \\ \varepsilon r_j, ab &= \varepsilon j, a - \varepsilon j, b \end{aligned} \tag{2.37}$$

Note that by differencing the errors on the two pseudoranges, the common related error components such as ionospheric and tropospheric error, satellite clock, ephemeris predictions etc. can be cancelled, assuming a relatively small distance between the two vehicles.
On the other hand, the uncorrelated error component given by sources such as multipath, receiver noise and minor residual components are actually increased.

Using the same definition used in previous section, of position displacement vector among the two GNSS receivers, such an offset can be expressed in terms of the satellite to user range difference.
As a matter of fact, the satellite ranges -in the order of thousands of km- are much greater than the inter vehicle distance -in the order of m-, so that as a first order approximation the difference $\Delta rj, ab$ can be written as the inner product between the displacement

vector and the unitary vector steering form satellite j to receiver a:

$$\Delta r_{j,ab} = \mathbf{h}_{j,a} \Delta \mathbf{x} \tag{2.38}$$

Supposing to have N available measurements, user a will compute all the pseudorange differences for the Nth satellite, composing the usual $\mathbf{H}$ matrix defined in (2.16), and solving iteratively for an estimate of the position displacement $\Delta\mathbf{x}$, from which the inter vehicle range can be calculated as in (2.35).

2.4.4 Double Differencing of Pseudoranges

In the DD of the pseudoranges, the user a follows the same approach as in the SD techniques and further differentiate data using single difference estimates from (2.36) taken from different satellites as follows:

$$\begin{aligned} DD^{ij} &= SD^{i}_{ab} - SD^{j}_{ab} = \\ &= \Delta r_{i,ab} + c\Delta t_{ab} + \varepsilon_{r_i,ab} - \Delta r_{j,ab} - c\Delta t_{ab} - \varepsilon_{r_j,ab} = \\ &= \Delta R^{ij} + \varepsilon_{r_{ij}} \end{aligned} \tag{2.39}$$

Where ΔR^{ij} is the difference between the satellite to user ranges and $\varepsilon_{r_{ij}}$ is the difference among the uncorrelated errors of the two satellites i,j.

Similarly to what happens with the SD technique, here the bias due to the user clock offset is completely removed, but there is a trade-off with the corresponding increase of the error component due to uncorrelated error sources.

Following the same reasoning used in the single differencing, the difference in the satellites ranges can be approximated as a function of the two unitary steering vectors directed towards the two satellites and the displacement of the position vector of (2.33) as follows:

$$\Delta R^{ij} = [\mathbf{h}_{i,a} - \mathbf{h}_{j,a}] \Delta \mathbf{x} \tag{2.40}$$

Subsequently, in order to construct the matrix for the least square solution, one of the visible satellites is chosen as a reference, namely $j = ref$, so that the double difference estimates of all the other N available satellites can be expressed with respect to it:

$$\mathbf{h}_{i,ref} = [\mathbf{h}_{i,a} - \mathbf{h}_{ref,a}] \qquad \text{with} \quad i \in [N] \tag{2.41}$$

All the possible double differences are then collected into a matrix form, solved for $\mathbf{\Delta x}$:

$$\mathbf{\Delta DD} = \begin{bmatrix} DD^{1,ref} \\ DD^{2,ref} \\ DD^{3,ref} \\ \vdots \\ DD^{N,ref} \end{bmatrix} \approx \mathbf{H}^{ref} = \begin{bmatrix} h^x_{1,ref} & h^y_{1,ref} & h^z_{1,ref} \\ h^x_{2,ref} & h^y_{2,ref} & h^z_{2,ref} \\ h^x_{3,ref} & h^y_{3,ref} & h^z_{3,ref} \\ \vdots & \vdots & \vdots \\ h^x_{N,ref} & h^y_{N,ref} & h^z_{N,ref} \end{bmatrix} \cdot \mathbf{\Delta x} = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}$$

CHAPTER 3

SIMULATION FRAMEWORK OVERVIEW

In Chapter 3 it is presented the background simulation environment. It will be provided a general overview of all the simulating components that are used in order to carry out simulations of the IAR collaborative positioning algorithm in a urban environment, relying on C-V2X (Cellular Vehicle To Everything) communication channels.

3.1 OMNeT++

A number of different solutions for the simulation of networks currently exist, each one with its particular flavor and characteristics. The main reasons for choosing OMNeT++ (Objective Modular Network Testbed in C++) are its development support, modularity, plus the fact that it is open source and well documented. OMNeT++ was initially developed at the Technical University of Budapest, Department of Telecommunications by András Varga [22].
OMNeT++ is not really a simulator by itself, but it is rather a simulation library and framework, modular and extensible, that is based on C++ components. It is a discrete event simulator based on a message-passing paradigm and although being a general purpose platform, it is primarily used as a basis to build network simulations.

In OMNeT++ simulation models are assembled together by means of basic components called modules, hierarchically nested. The communication among different interconnected modules happens through message passing, using message definitions (`.msg` files) that are automatically translated into C++ classes. The output and input

interfaces of a module are called gates and they are typically linked by a connection to allow the message exchange from one module to another.

There are two module types, which are simple and compound.
Simple modules are the active components in the simulation model and their behaviour is implemented in a corresponding C++ file, typically created by subclassing in order to easily allow code reuse for different use cases.
A compound module is instead defined by grouping together different simple modules, which then become its submodules, with an unlimited number of hierachy levels allowed. The entire simulation model itself, called network, is indeed a compound module.
Every module has an associated NED language topology description (`.ned` files) that describe the module structure with its parameters, gates, submodules and connection. A generic model organization is shown in the following figure:

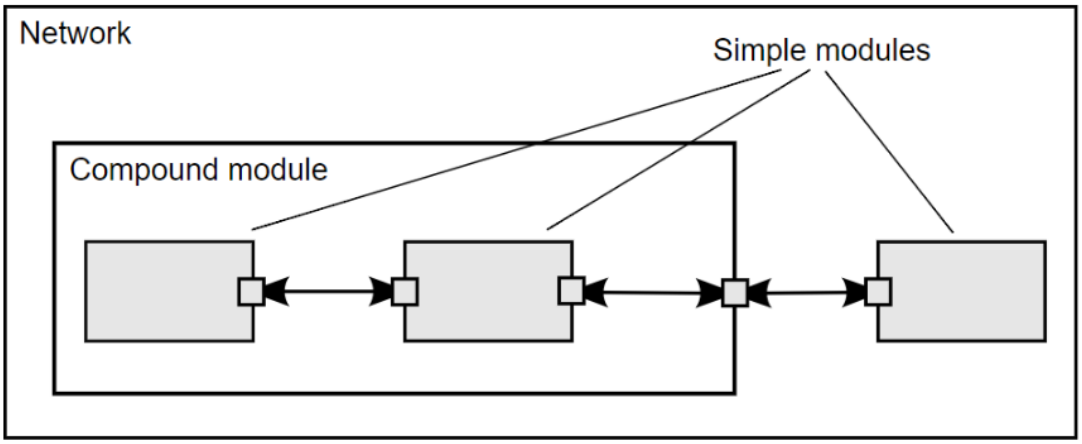


Figure 3.1: General OMNeT++ modules hierarchical architecture [16]

As already stated above, one of the main advandages of OMNeT++ is its full modularity and the possibility to integrate different tools or model libraries. Among the main simulation models already available in OMNeT++ the INET Framework is considered as the standard protocol model library, containing among the other models for TCP, UDP, IPv4, IPv6, Ethernet, PPP, 802.11 as well as support for mobility, applications and many additional protocols and components.

A lot of different INET-based frameworks are currently implemented and mantained by different research groups, such as Veins and SimuLTE.

3.2 Veins simulation framework

Veins (Vehicles in Network Simulations), is an open source vehicular network simulation framework [19]. The available simulation models are executed within the network simulator OMNeT++ and simultaneously interact with a road traffic simulator, i.e. SUMO (Simulation of Urban MObility).

Veins is a simulation framework as it carries out the proper modelling of lower protocol layers and nodes mobility and deals with the setting and running of the simulation correct execution.
Furthermore Veins comes with a large number of already implemented simulation models that can be applied within the environment of a vehicular network, for example a TraCI (Traffic Controlo Interface) module used for the proper interaction with SUMO, a two-ray interference model for path loss computation, obstacle shadowing and signal attenuation by building and a Veins_INET subproject, which allows the seamless integration and usage of the standard OMNeT++ INET framework modules.

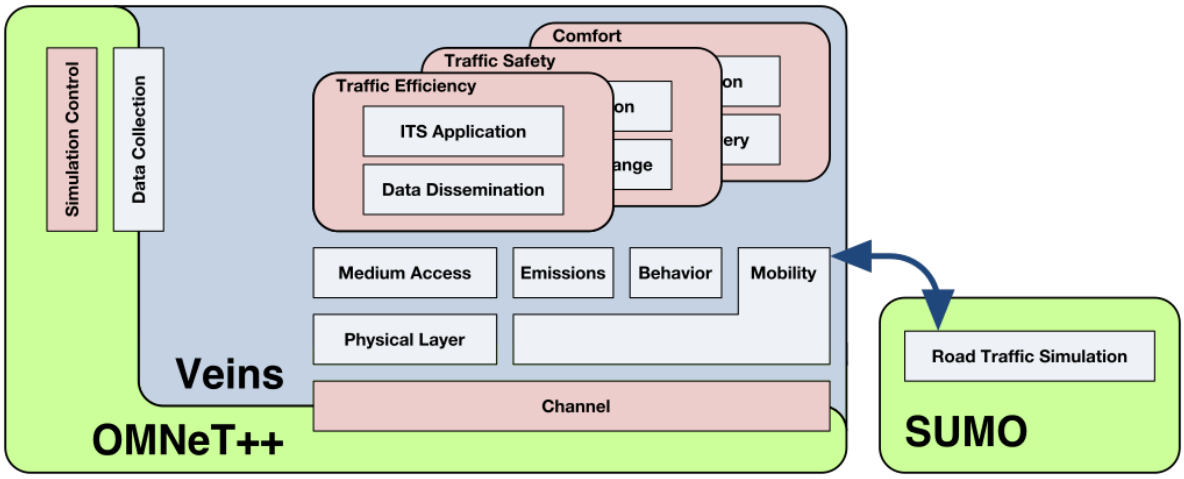


Figure 3.2: Veins simulation framework

3.2.1 SUMO

As already mentioned in the previous section, Veins is a simulation framework designed for the coupling of a network simulator like OMNeT++ and a road traffic simulator.
SUMO is a traffic simulation platform developed by the German Aerospace Center (DLR) and it is released as an open source software since 2002 [9]. It is a microscopic and continuous traffic simulator, designed to manage large road networks, that allows

simulation not only of vehicles, but also includes pedestrians and public transportation, modelling for example single vehicles with models for car-following and lane changing behavior.

The road network is described in a proper `.net.xml` file, that specifies all the roads or *edges* with position, shape and speed limits, lanes, intersections or *junctions*, connections between lanes at junctions and possibly traffic light logics that are to be modelled in the network topology.

After the creation of a network, SUMO requires also a so-called traffic demand, i.e. the description about vehicles or pedestrian behavior in the network. Modelling of the traffic demand is done in a proper `.rou.xml` file which contains all the information about vehicle type and related physical properties, such as speed, acceleration, departure time etc. and a route defining the complete path that the vehicle will follow during the simulation or just a trip, which specifies only the starting and the arriving edge.

SUMO has been continuously developed since the first release and it provides the user with a large number of tools, typically written in Python, such as automatic network generation, import or conversion from other traffic simulators, applications for traffic demand generation and randomization over large simulation scenarios and a TraCI interface to couple the road traffic simulation with an external simulation model for V2X communications [8].

3.3 SimuLTE

SimuLTE is a communication simulator for designed to simulate LTE and LTE-Adanced FDD (Frequency Division Duplex) cellular networks, starting from 3GPP Release 8 and further releases. The tool has been developed within an open source project of the Computer Networking Group of The University of Pisa and it is described in detail in [24].

SimuLTE is written in C++ and it is based on the simulation framework of OMNeT++ and INET, gaining all the aformentioned advantages of full modularity, customization, inter-operability, parametrization and simulation automation.
SimuLTE provides a simulation model for both the data plane of LTE E-UTRAN (Evolved-Universal Terrestrial Radio Access Network) and the EPC (Evolved Packet Core). The generic structure of the model is composed by three main nodes which are the LTE Binder, the UE (User Equipment) and the eNodeB (Evolved Node B) modules.

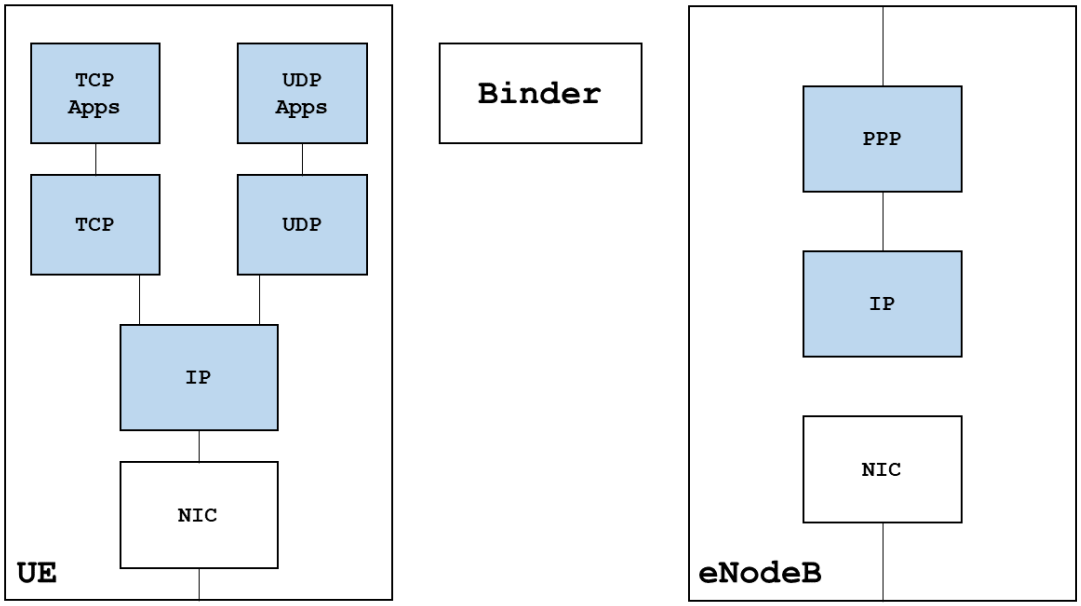


Figure 3.3: Main modules of SimuLTE, where the light blue modules are imported from alreay implemented INET modules

Regarding the Binder module, it acts as a kind of "glue" in the simulation model, since it is transparent to all the LTE nodes in the simulation and it is responsible for gathering relevant information about them, such as tracking of the allocation of radio resources each TTI (Transmission Time Interval) of 1ms.
All nodes in the simulation network can moreover access those information invoking the Binder themselves, for example during the inter-cell interference computation experienced by a given user.

Both the UE (User Equipment) and the eNodeB (Evolved Node B) are implemented as compound modules, which can be interconnected one to the other and also to different modules in order to create mixed network scenarios.
As a matter of fact both the compound modules are created collecting newly written, dedicated modules like LTE NIC (Network Interface Card) together with modules taken from the INET framework, such as IP, TCP and UDP for the UE or PPP (Point-to-Point Protocol) for the connection of eNB with other IP nodes.

The LTE protocol stack is modelled within the NIC module, that is built upon the standard IWirelessNic interface already defined in the INET library, allowing to create mixed hybrid scenarios.
The general architecture for the NIC module is shown in Figure 3.4, where the LTE

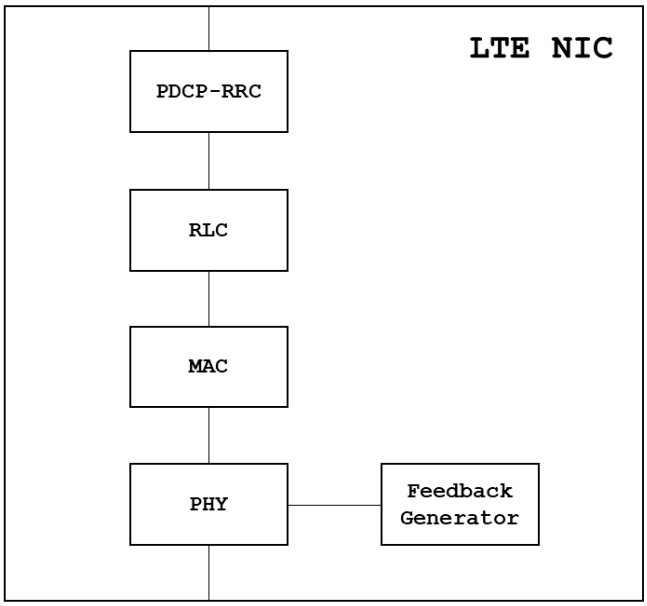


Figure 3.4: LTE NIC module architecture

layers PDPC-RRC, RRC, MAC and PHY are implemented for both UE and eNB, whereas the Feedback Generator module is implemented just in the users, in order to generate channel feedback, i.e. CQI (Channel Quality Indicator) on the UE nodes.

Among the numerous implemented features, SimuLTE provide the user with a X2 interface [13] for the interconnection among several eNodeBs in order to better assess the performances of algorithms such as CoMP (Coordinate Multi Point) Scheduling and Network Assisted Handover.

Another important functionality implemented in SimuLTE is the network-assisted D2D (device to device) communication [23], which allows a direct communication link among peer UEs without the need of using the eNB as a relay. In this approach the network node is just exchanging control data with the connected UEs, but the exchange of data happens directly among users. Both unicast and multicast LTE D2D communications are currently supported by SimuLTE [14].

Furthermore, the constant developement by the researchers has recently introduced an enhancement to SimuLTE, allowing a tight integration with the vehicular network simulator Veins, within the same, common simulation framework provided by OMNeT++ and the INET library [15].

CHAPTER 4

REFERENCE SCENARIO

This chapter deals with the presentation of the reference scenario used for the simulations, starting from the definition of a communication protocol to exchange pieces of collaborative information and detailing the software resources implemented for the simulation.

All the collaborative positioning methods based on GNSS measurements without the integration of other additional sensor or dedicated hardware for ranging estimation, require the exchange of information between two or more vehicles.

In order to follow the standardization approach envisioned by the European Telecommunication Standards Institute (ETSI), the proposed communication protocol for the exchange of relevant data will be based on CAM and DENM messages.

4.1 CAM

The advances in electronics, cellular communications and network application of the recent years, are setting the way for a world where billions of devices will be interconnected together. In this scenario, the focus on vehicles is becoming increasingly popular, so that topics such as autonomous driving and connected cars have become of large interest, also thanks to the automotive industry boost.

The European Union has struggled in standardizing new technologies and communication systems, in order to allow an harmonized interconnection among vehicles. The

Cooperative Awareness Message (CAM) detailed in [1] is one of the main parts of this standardization effort pursued by ETSI in the framework of ITS.

The main idea behind cooperative awareness is to provide road users and roadside infrastructure equipment with relevant information such as heading, speed, position and other dynamic attributes about all the other nodes which are present in the road network.

Reciprocal awareness is the basis for a number of different road safety and road traffic management applications, that have been described in a Basic Set of Applications [2] [3], such as emergency vehicle warning, collision risk warning, traffic efficiency and others.

This kind of awareness can be achieved by exchanging useful information, that can be of different types:

- V2V: vehicle to vehicle communication
- V2I: vehicle to infrastructure communication
- I2V: infrastructure to vehicle communication
- V2P: vehicle to pedestrian communication

Wireless networks comprising different types of communication technologies are also referred to as V2X (vehicle to everything), meaning that the exchange of information can happen among different types of nodes.

The dissemination, management and processing of the CAM messages is performed by the Cooperative Awareness basic service, which has to be mandatory implemented in all types of nodes or ITS-Stations that are present in the road.

4.1.1 CAM PDU format

According to ETSI specifications, the CAM is in general composed by a common header plus a number of multiple containers, some of which are mandatory.
The general structure of a CAM PDU is shown in figure 4.1: The red outlined boxes identify all the data structures that must be present in the CAM PDU, whereas the black boxes are optionally included.

Mandatory fields are the ITS PDU header, the Basic container and the HF container. The Basic container contains information such as the type of ITS-S and the last known position at the moment of the CAM generation.

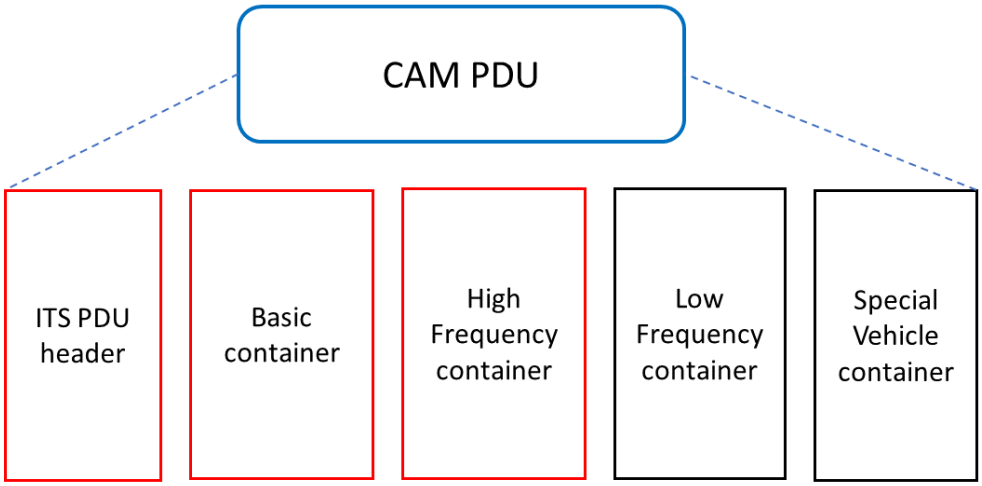


Figure 4.1: CAM PDU format

The High Frequency container is used to store highly dynamic information such as heading, speed, drive direction, acceleration and vehicle dimensions.

The Low Frequency container might be present or not and contains low dynamic or static data, such as vehicle role, exterior lights and path history.

The conditional Special Vehicle container might be used for special kind of vehicles and information about them, for example a Public Transport container or a Special Transport container.

The general structure of a CAM PDU is highly flexible and it can therefore be modified, by defining additional optional containers for dedicated applications.

4.1.2 CAM frequency management

The CA basic service should be activated within ITS-S activation, whatever type it might be (e.g. vehicle, road side unit, personal). It is responsible for the CAM proper generation and management.

Considering vehicle ITS-S, the ETSI standard identifies specific bounds for the CAM generation frequency, that is the inter-generation time interval between two different CAMs.

In particular the limits for the transmission are recommended as:

- The lower bound for the CAM generation time should not be less than the parameter `t_gen_cam_min` = 100 ms, which in terms of generation frequency corresponds to a value of `f_gen_cam_min` = 10 Hz

- The upper bound for the CAM generation time should not be more than the parameter `t_gen_cam_max` = 1 ms, which in terms of generation frequency corresponds to a value of `f_gen_cam_max` = 1 Hz

The generation of a CAM is subordinate to a specific set of conditions, which should be repeatedly checked every `t_check_cam_gen` and its value should be set less or equal than `t_gen_cam_min`.

In order to be compliant to the channel usage specified in the Decentralized Congestion Control (DCC), the parameter `t_gen_cam_dcc` should be chosen with the minimum and maximum range limits of the CAM generation time. If the parameter is not specified, it should be set `t_gen_cam_dcc = t_gen_cam_min`.
Regarding the conditions which trigger a CAM generation, these specifically are:

- The time interval with respect to the last CAM generation is equal or greater than `t_gen_cam_dcc` and one of the following three conditions on the ITS dynamic parameters is satisfied:

  1. the heading absolute difference is equal or greater than 4 °;
  2. the position difference is equal or greater than 4 m;
  3. the speed absolute difference is equal or greater than 0.5 m/s;

- The time interval with respect to the last CAM generation is equal or greater than `t_gen_cam` or `t_gen_cam_dcc`

Where the parameter `t_gen_cam` is the actual valid upper bound to the CAM inter-generation time at a given time instant. It should be set by default at `t_gen_cam_max`. In case the first set of conditions is satisfied, its value should be set set to the time elapsed since the last generated CAM. In case the CAM is generated due to the second conditions, after `n_gen_cam` consecutive generations, its value should be set again as default. Regarding the maximum number of consecutive CAM generations, the parameter `n_gen_cam` should be set by default at 3.

4.2 DENM

Another important piece that contributes forming the complete framework in the ITS applications and funding components is the Decentralized Environmental Notification (DEN) basic service that is designed specifically to address all the BSA applications falling within the range of Road Hazard Warning (RHW).

This service is responsible for collecting, creating, handling, disseminate and managing the Decentralized Environmental Notification Message (DENM), whose transmission is not stricly periodic as for the CAM, but instead triggered in response to specific events related to a road traffic event.

Whenever a station determines the detection of a road hazard warning warning, it starts broadcasting the DENM to other ITS-S in the relevance area. RHW applications should provide the DEN service with all the information required to construct and manage a DENM. The transmission in then typically repeated within a certain frequency and the ITS-S continues sending it as long as the triggering event is not finished or until the expiry of a time interval.

The dissemination of DENM is regulated by some parameters, which are detailed for the given specific application. Such parameters are:

- Transmission frequency
- Required transmission latency
- Priority
- Destination area

The termination of the might happen in two ways, depending on whether the event termination is detected by the originating ITS, called cancellation, or or by one ore more authorized third party ITS-S, in which case is called negation.

The general structure of a DENM PDU is a common ITS PDU header and the actual DENM. The header is the same as for the CAM structure and contains information about the protocol version, message type and a timestamp. The remaining DENM is composed by three main fixed and ordered parts, that are management, situation and location container. Every component is further made of a sequence of data elements or structures, as shown in the general DENM scheme:

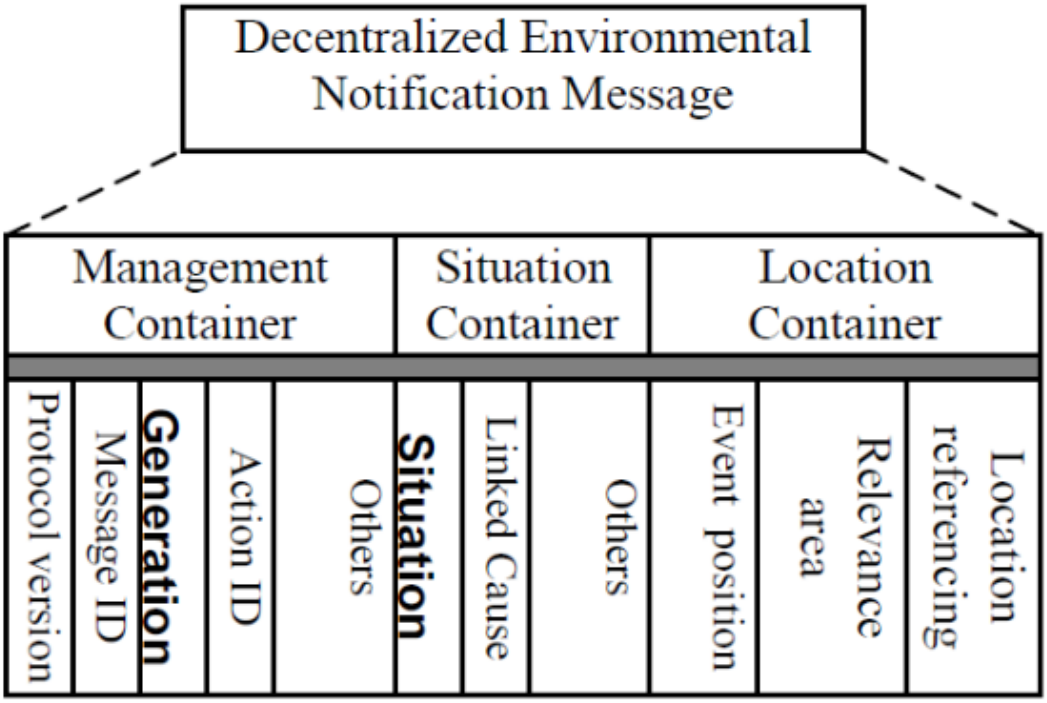


Figure 4.2: DENM payload general format

4.3 C-V2X technology

4.4 Application protocol

In order to simulate the exchange of information depicted in the IAR Collaborative Positioning algorithm, it was decided to exploit C-V2X as a wireless communication technology and to rely upon the transmission of custom CAM and DENM packets.

The simulation environment is based on OMNeT++ version 5.1.1, acting as the network simulator sort of "glue" among all the software components. On top of it, it is installed the tool SimuLTE to simulate the cellular network, coupled with Veins 4.6 for the implementation of the vehicles. The last resource is Sumo 0.30, used as a road traffic simulator.

The application has been designed based on an already implemented collision detection application, from Politecnico di Torino. Recall from Chapter 2 that in the IAR technique, the aided vehicle sends to the aiding peer just its last known unitary steering vector. The aiding requests are modelled as conventional CAM packets, with some

dedicated additional parameters such as:

- a flag to discriminate between standard CAMs and positioning aiding requests;
- a request number that coupled with the vehicle ID allows to recognize the aided peer and the corresponding response;
- the unitary steering vector for the IAR algorithm.

In this way it is possible to simulate both standard vehicles that are just moving sending CAMs and CP vehicles, using the same unified application.

Regarding the responses containing the actual collaborative information, those have been differentiated from the already implemented DENM messages. The justification for this choice is the fact that DENM have been envisioned primarily for road hazard warnings and not for positioning or navigation purposes, thus the need for a dedicated message.

As an addition, although the focus is primarily on the evaluation of the IAR technique, other different ranging estimation techniques based on GNSS estimates exist, as already presented in Section 2.4. By using a dedicated packet response, it would be possible to further modify the structure of data, adding the extra information that is needed for such techniques.

In the IAR estimation for example, the aid response has to contain just the information about the included angle *alpha* plus one satellite to receiver range, whose order of magnitude is below 32000 km, so that it could be stored in just a couple of bytes or 16 bits.

Considering instead ranging techniques such as Double Differencing of pseudoranges, they require much more information such as the full pseudorange measurement vector for all the satellites with the corresponding error information.

On the other hand, considering the request of positioning aid, it does not need data structures of variable size, so that a CAM with slight modifications can be used.

The fundamental modules are the *Client*, associated to every car created in the simulation and the *Server*, whose behavior is implemented in a corresponding `C++` file. The two modules have also dedicated `.ned` files, in order to allow a parmetrization of their behavior. Both are simple modules based on the reference of `IUDPApp` that is a standard UDP application module taken form the INET simulation library.

[a]

```
package lte.apps.collision_detection;

import inet.applications.contract.IUDPApp;


simple ClientCP like IUDPApp
{
    parameters:
        int localPort = default(9090);
        int destPort = default(3000);
        string destAddress;
        int PacketSize = default(240);

        // PacketSize in bytes for Collaborative Positioning Response
        int PacketSizeCP = default(300);
        int flagCP = default(0);

        double startTime @unit("s") = default(0s);
        string position = default("metro");

        // Max CAM frequency for cars
        double frequency_car_min = default(0.1);// minimum cam interval
        double frequency_car_max = default(1.0); // maximum cam interval
        // CAM frequency for peds
        double frequency_ped = default(1);
        // Delays in the communication
        double elaboration_time_car @unit("s") = default(0.2s);   //200ms
        double elaboration_time_ped @unit("s") = default(0.05s);  //50ms
        // Delays due to encryption/decription
        double decryption_ped @unit("s") = default(0s);
        double decryption_car @unit("s") = default(0s);
        double encryption_server @unit("s") = default(0s);


        @display("i=block/source");
    gates:
        output udpOut;
        input udpIn;
}
```

[b]

```
package lte.apps.collision_detection;

import inet.applications.contract.IUDPApp;


simple ServerCP like IUDPApp
{
    parameters:
        int localPort = default(3000);
        int destPort = default(9090);
        double delay = default(0);
        int PacketSize = default(263);
        // PacketSize in bytes for Collaborative Positioning Response
        int PacketSizeCP = default(300);
        string position = default("metro");

        // Delays due to encryption/decription
        double encryption_ped @unit("s") = default(0s);
        double encryption_car @unit("s") = default(0s);
        double decryption_server @unit("s") = default(0s);

    gates:
        output udpOut;
        input udpIn;
}
```

Figure 4.3: ClientCP.ned code snippet (a); ServerCP.ned code snippet (b)

4.5 LTE infrastructure

The communication technology implemented in all the simulation scenarios is a cellular vehicle-to-infrastructure based on LTE. The network has a simple topology with a single eNodeB, whose module is provided in SimuLTE, that is positioned at the center of the road topology created with SUMO.
Since OMNeT++ and SUMO are using different coordinate reference system, the position of the cell has to be carefully changed in the network file called `Grid.ned`, in order to actually match the correct corresponding point in the road network being simulated.

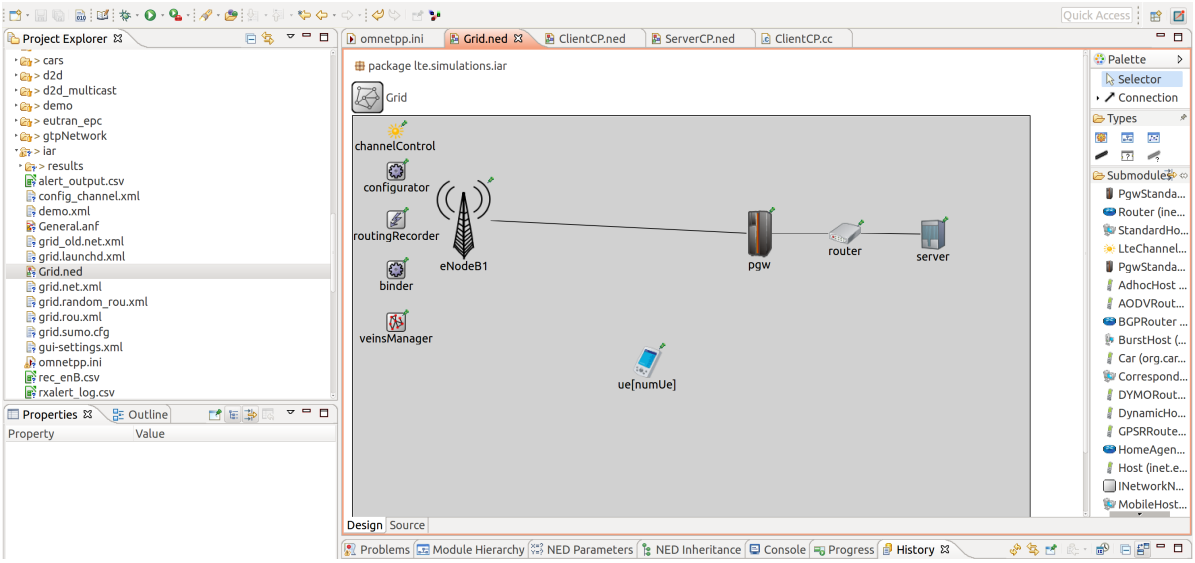


Figure 4.4: Network Grid.ned

The `binder` module is taken directly from SimuLTE libraries and it is needed for the correct simulation of the network, as it is needed for tasks like feedback computation.

The `channelControl` contains parameters for the modelling of the shared channel, whereas the `configurator` is a submodule imported from the INET library and it is used for the the automatic assignement of IP addresses to the netowrk nodes such as the vehicles.

The `server` submodule is derived from the INET standard host, and it is located at the edge of the mobile cellular network, close to the eNB. This module is the one responsible for running the `ServerCP` application and it is in charge for managing and answering to the requests of collaborative positioning coming from the nodes.

A fundamental module is the `veinsManager`, that is the one responsible for the creation of the vehicle nodes in the simulation and provides the interface for the correct

coupling of the vehicles in both the network and the road traffic simulation.

Additionally, in the network is defined an array of `ue`, which are the standard LTE User Equipment nodes, whose number is created according to the parameter `numUe` set by default at zero.

4.6 SUMO road network

The road network have been created in SUMO, relying onto the provided tools NETGENERATE, NETCONVERT and NETEDIT. In particular NETGENERATE is a command line application, that allows to create a network with given characteristics specified as instructions in the terminal used.
The modelled road network reported in Figure 4.5, has been created from scratch with a command line instruction in order to generate a grid topology:

```
netgenerate -grid -grid.number=6 -grid.length=100 -output-file=grid.net.xml
```

which creates a grid network with 6 junctions both in the x-axis and the y-axis, spaced by 100 m, resulting overall in a square of side 500 m.
By using the `netconvert` command it is possible to modify the newly created network, for example to change the number of lanes, which has been set to 2 lanes for each bi-directional edge. The network has then been checked in the GUI provided by `netedit`, in order to verify the correctness of the shape, junctions, edges and to modify or change existing connections. All the crossings in the network are connected in the same fashion, that is no turnarounds allowed and for the internal nodes, the allowed turn directions are straight or right for the right lane, left for the left lane. For the crossing on the outer edges of the network, a vehicle is allowed to just turn right or left.

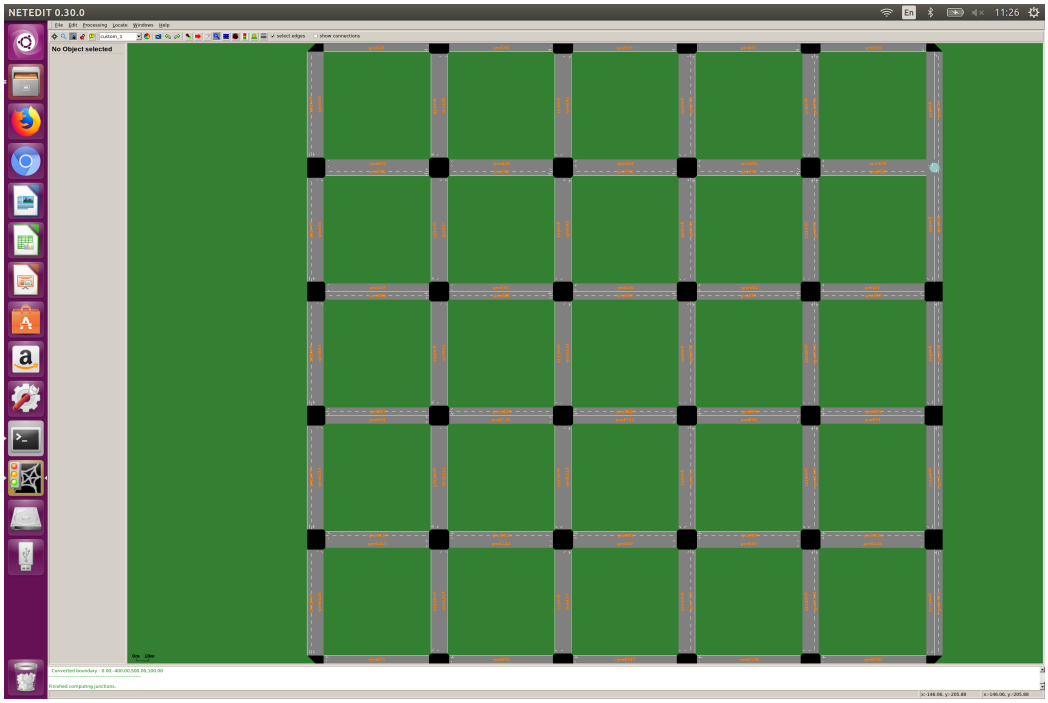


Figure 4.5: Manhattan like grid network created in SUMO

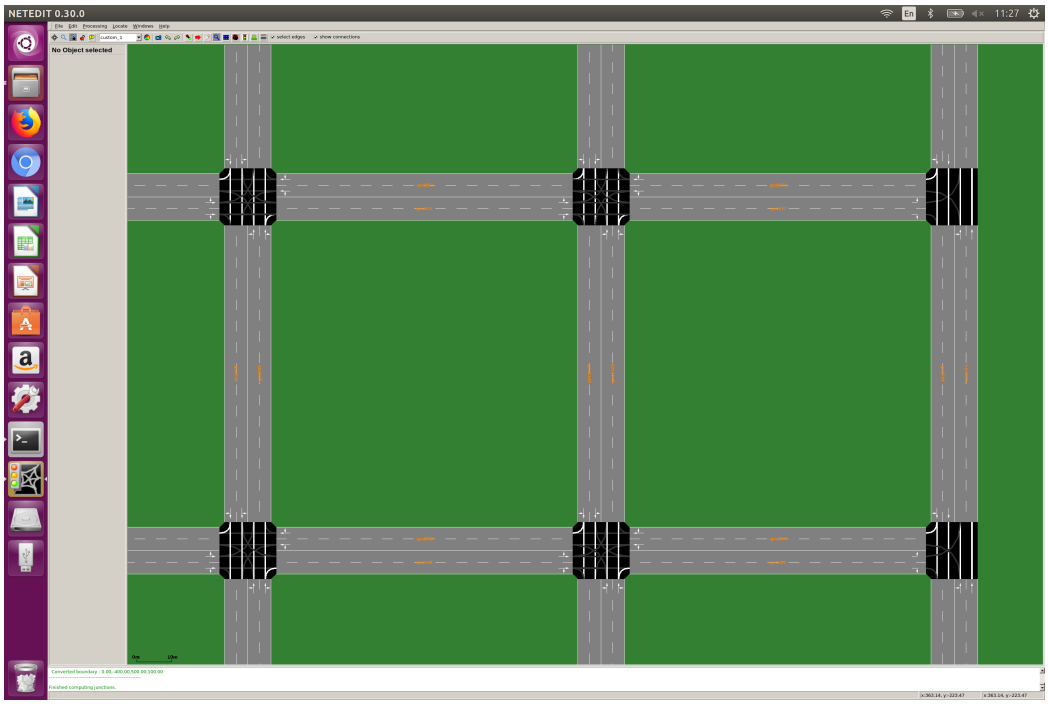


Figure 4.6: Zoomed view of an internal node crossing, with allowed connections

4.7 SUMO traffic demand

After the creation of the network, the next step is to create mobility traces in order to actually simulate traffic in the road network. The traffic demand is a description about the vehicles, their characteristic, parameters and the path to be followed on the road topology. As a matter of fact, In SUMO a vehicle is specified by three parts:

- the actual vehicle;
- the path on the network that the vehicle should take;
- the vehicle type with the physical parameters of the vehicle;

The path might be a route, that is a complete description of all the nodes that a car has to cross, or it might be given as a trip, that is specifying just the starting and the end edge. Routes and trips can have an associate id, in order to be assigned to multiple vehicles at the same time.

Among the physical parameters that define a vehicle in the simulation, there are the id, acceleration, departure speed, departure time, width, length, departure lane etc.. All the relevant physical properties might be grouped together into global vehicle types `vType` and assigned simultaneously to a number of vehicles.

In order to obtain such a description, several different methods can be exploited, using trip or flows definitions, OD-matrices, by hand or by randomization.

In particular, the mobility traces have been generated using the Python script `randomTrips.py`, that allows to easily generate random traffic models, by choosing starting and destination edge according to a uniform distribution or with a custom distribution specified by the user. It is also possible to define the time interval for generating the trips and to optionally generate complete routes. The output generated file is stored in a `.rou.xml` file, that can be used in the simulation for the mobility of the vehicles.

CHAPTER 5

SIMULATION RESULTS

All the presented scenario are referring to the road network topology depicted in Figure 4.5, whereas the reference LTE network is the one presented in Figure 4.4. The differentiation on different types of simulations has been achieved by properly setting and modifying relevant parameters such as the number of vehicles, the running application and the presence or not of additional users connected to the network infrastructure.

5.1 Scenario A

In the first simulated scenario, namely Scenario A, all the vehicles in the network are just transmitting CAMs towards the eNodeB, without doing anything else. This has been accomplished by disabling the collaborative positioning part in the `ClientCP`, leaving just the modelled behaviour for the generation and sending of CAM messages, according to ETSI specifications for vehicle ITS-S.

The simulation have been performed with a duration of 200 s, repeated 10 times using different seeds for randomization and with 100 vehicles in the road. The results have been then collected altogether and averaged over the number of runs, for each vehicle.

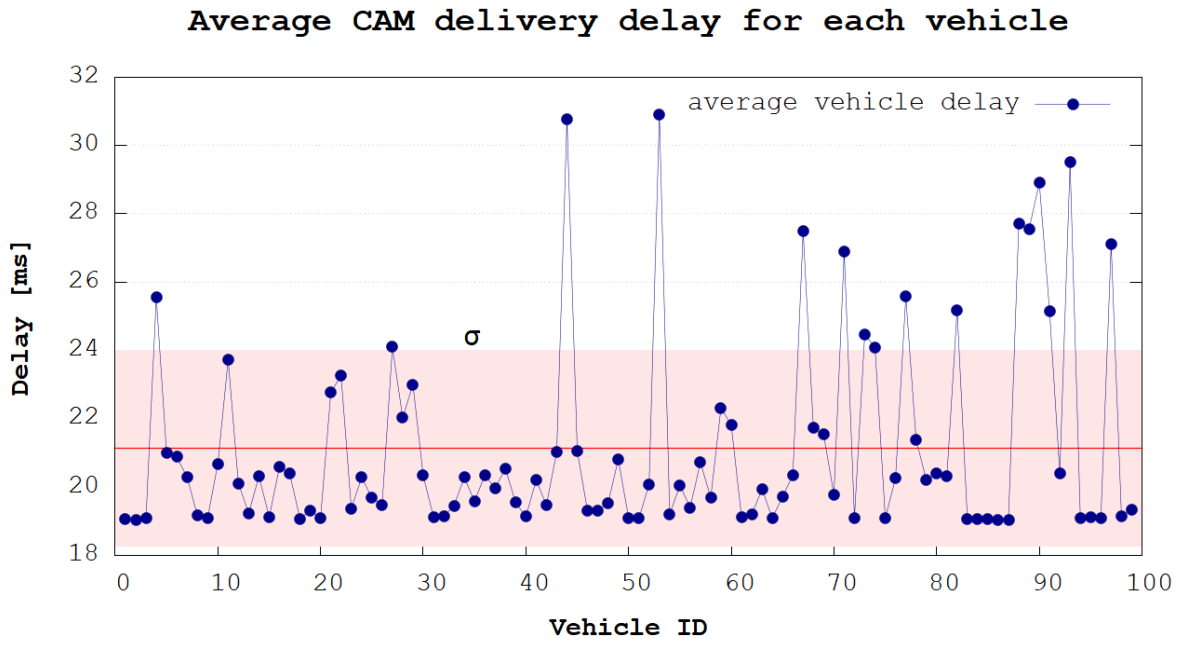


Figure 5.1: CAM average delay

5.2 Scenario B

In the second simulated scenario, namely Scenario B, some of the initialized vehicles are chosen according to a uniform distribution. Such vehicles will transmit the CP requests and receive the responses from the server, while all the remaining vehicles occupy the channel by sending conventional CAMs.

As already mentioned, the CP behavior is implemented in the `ClientCP.cc`. In particular it is based on the parameter `flagCP`: if its value is equal to 0, the client behaves like a conventional vehicle transmitting CAM, otherwise if it is set to 1, it sends a CP request.

The generation rate of such requests is managed in the `handleMessage` method, where the CP vehicle schedules a self message to generate a new request according to a negative exponential distribution with parameter $\lambda = 1s$. This is achieved by using the OMNeT++ function exponential:
`t_check_gen_cp = exponential(mean)` Given those preliminary considerations, the scenario have been simulated in two cases:

1. with 10 cars;

2. with 100 cars;

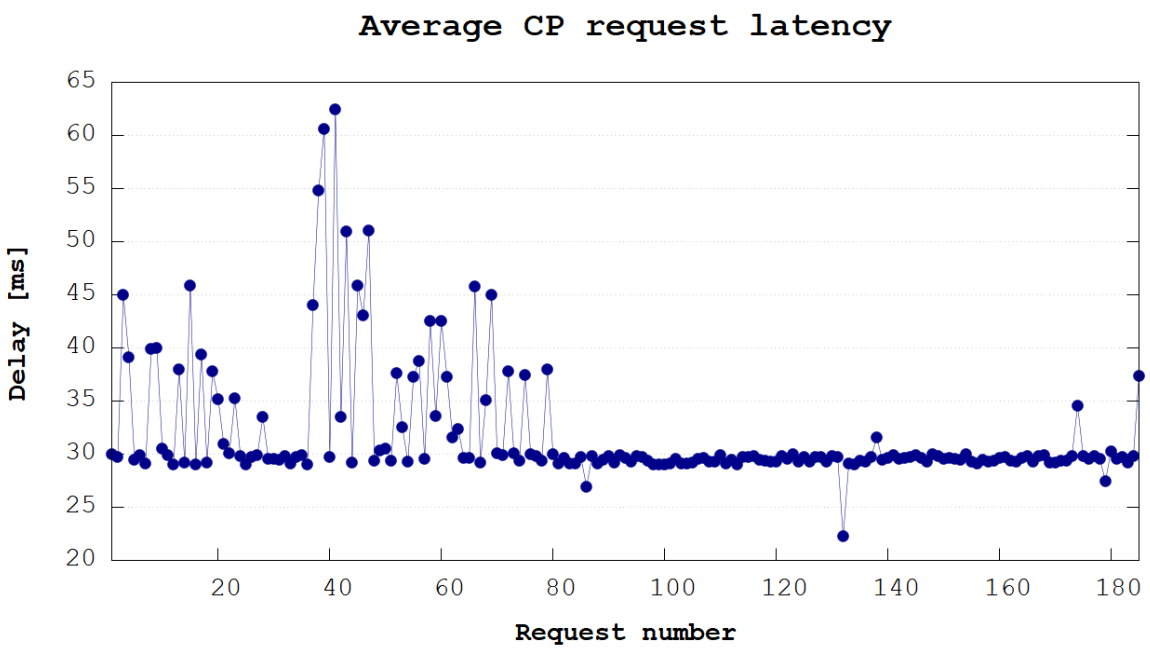


Figure 5.2: Case 1 with 10 cars

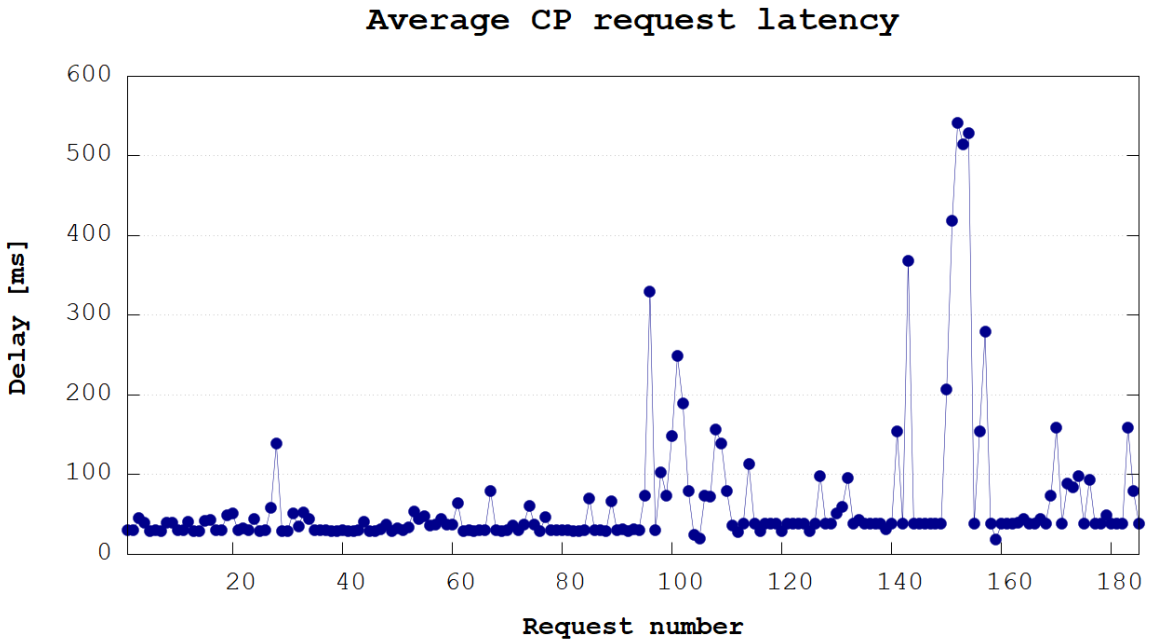


Figure 5.3: Case 2 with 100 cars

5.3 Scenario C

In the third simulated scenario, namely Scenario C, the situation is identical with respect to Scenario B, with the addition of LTE users. The generic UE is taken from the SimuLTE library and it implements the INET mobility. The UEs are placed in the network in order to simulate a more realistic scenario regarding traffic load on the eNodeB and channel utilization. The number of vehicles and UEs have been changed, in order to obtain different simulation scenarios, as shown in Table 5.1

Table 5.1: Table simulation configurations

Run	Vehicles	UEs
1	10	25
2	10	50
3	50	25
4	50	50

The number of user has been changed from the `.ini` file, where UE has two different `UDPApps`, one for a VoIP in Uplink towards the eNodeB, and the other dedicated for the generation of VoIP Downlink traffic. The server has a number of applications twice as the number of UEs, plus another one dedicated to CP purposes.

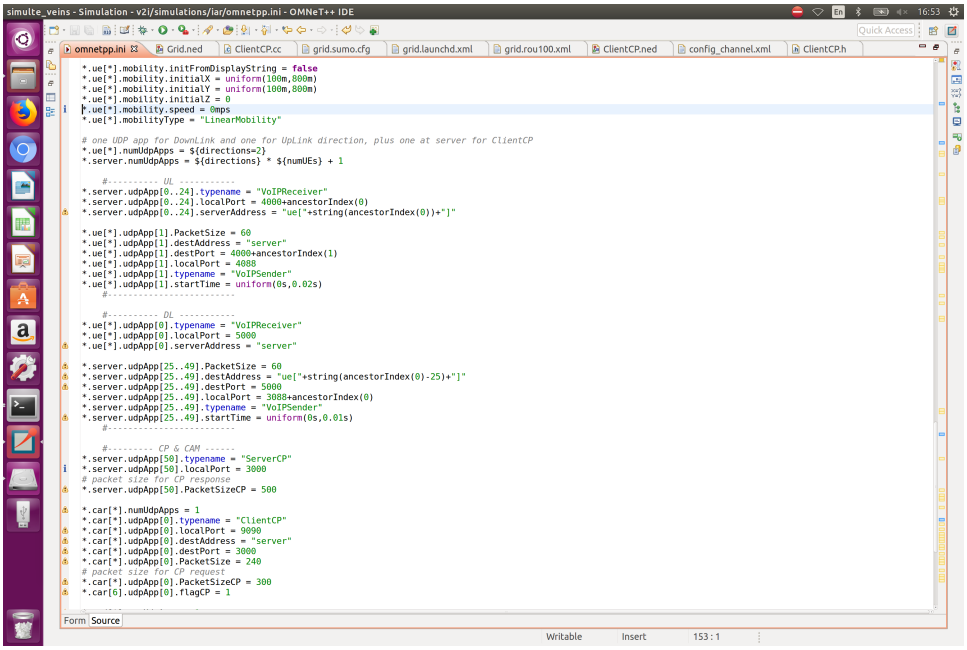


Figure 5.4: Configuration file snippet with 25 UEs

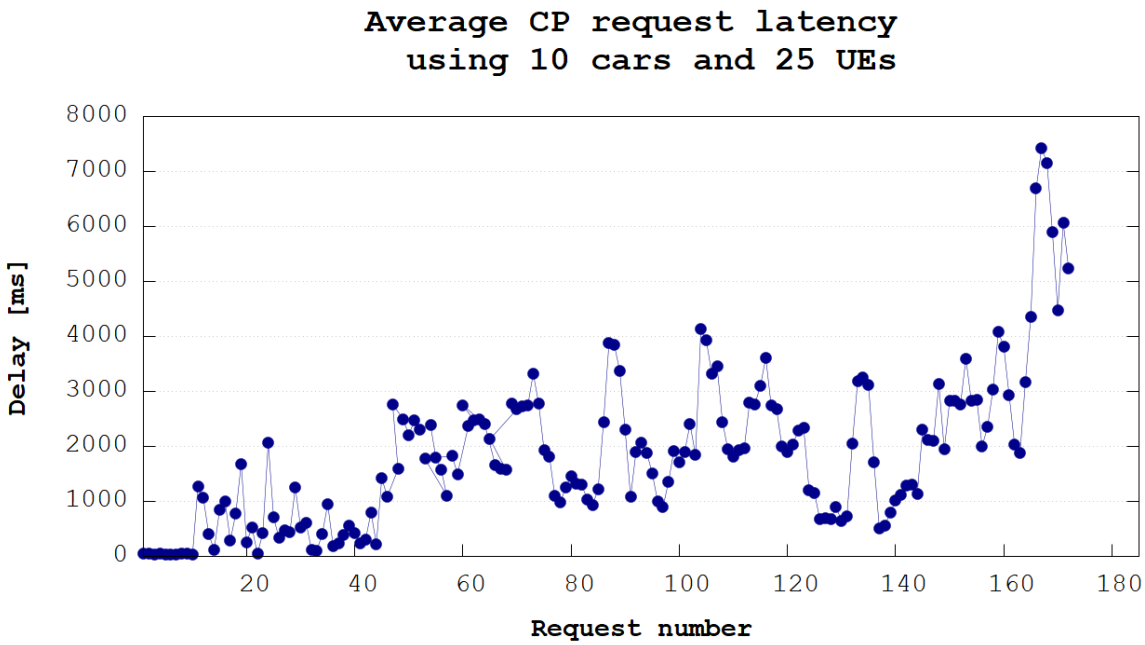


Figure 5.5: Run 1

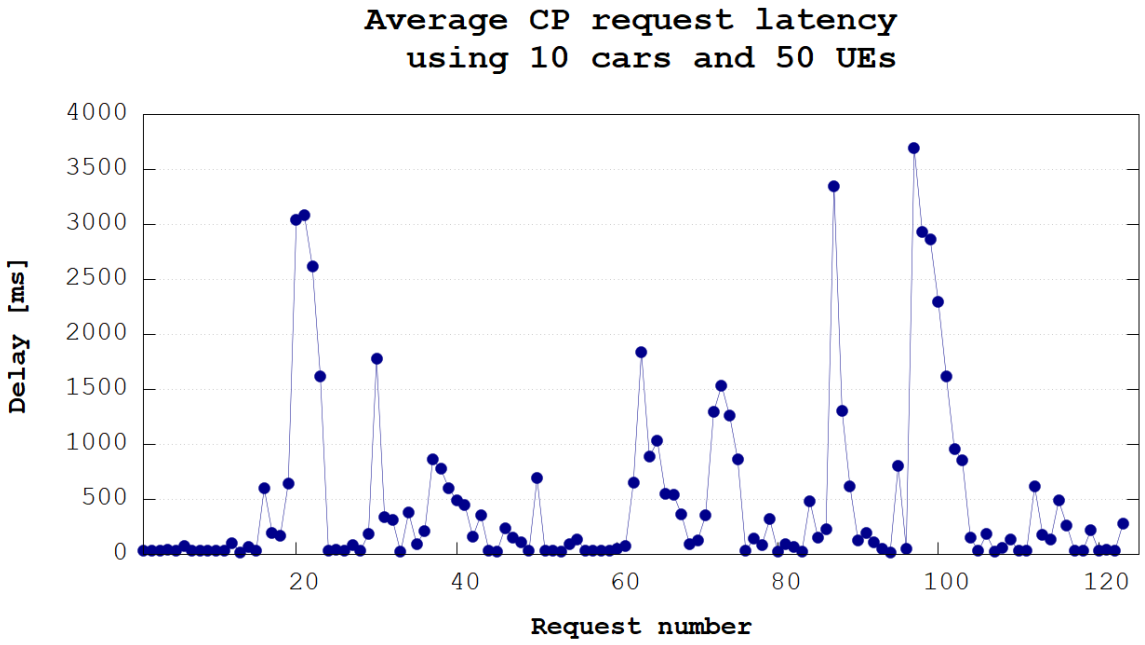


Figure 5.6: Run 2

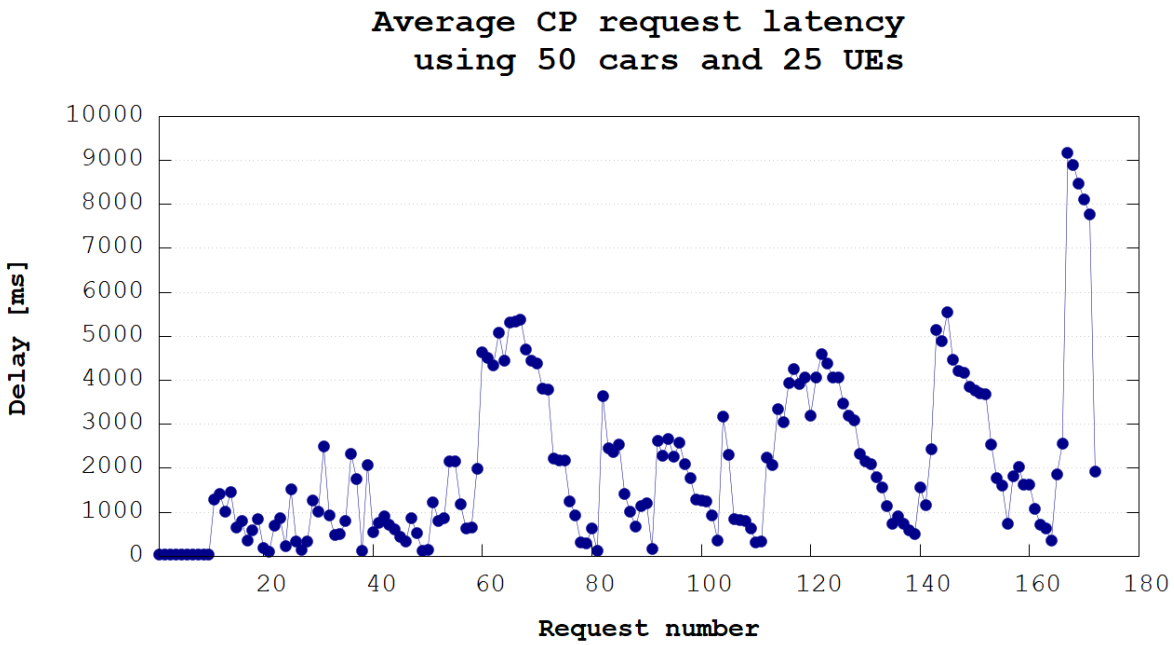


Figure 5.7: Run 3

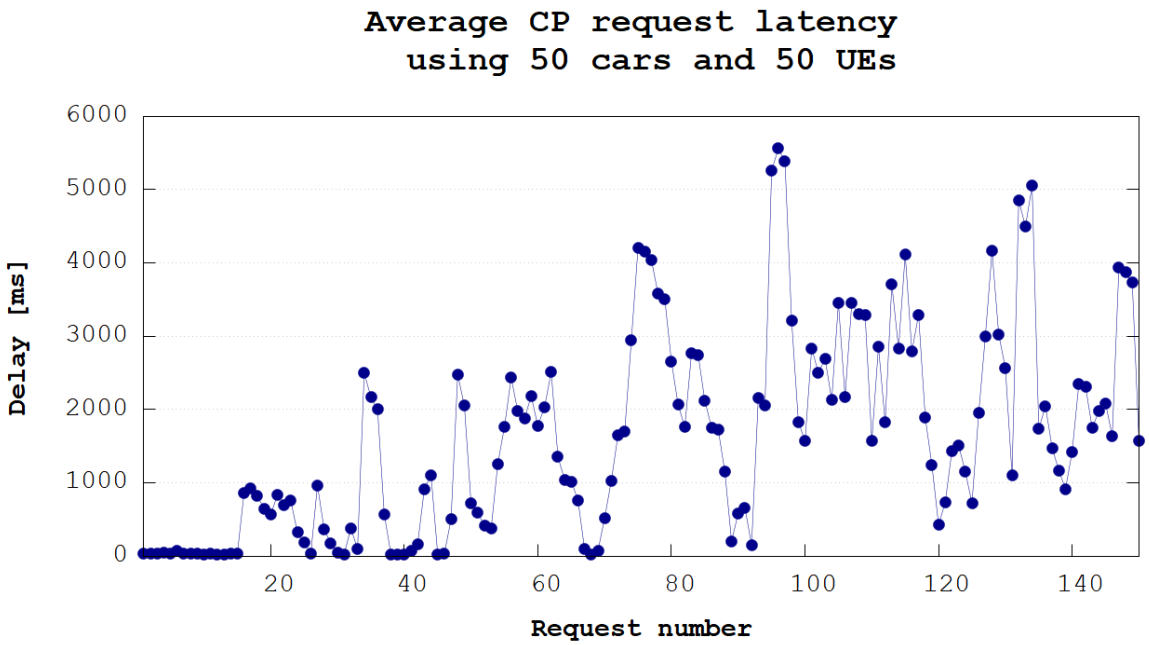


Figure 5.8: Run 4

CHAPTER 6

CONCLUSION

This last chapter deals with a general summary and some conclusive remarks about the obtained results.

The conducted simulations showed that in ideal networking conditions, the latency due to the V2X infrastructure is quite small and in the order of tens of milliseconds, making C-V2X a suitable candidate for applications like Collaborative Positioning algorithms.

Nevertheless, in the second case study of Scenario B, the overall trend of the delay starts to become less predictable and has larger oscillation peaks, even though mantaining an acceptable mean value.

In the third case, with the introduction of additional sources of traffic on the network and a more realistic channel model, the pattern of the latency becomes highly unstable, with large fluctuations and variations, reaching completely unacceptable values in the order of seconds.

An important role for those results is the dimensioning of the road network, which has a side of 500 m. In such a topology, the vehicles towards outer edges will be in a great distance with respect to the eNodeB, thus resulting in large delays, possibly due also to retransmissions.

BIBLIOGRAPHY

[1] ETSI EN 302 637-2 V1.3.1 (2014-09), *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service*, 2014.

[2] ETSI TR 102 638 V1.1.1 (2009-06, *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Definitions*, tech. rep., 2009.

[3] ETSI TS 102 637-1 V1.1.1 (2010-09), *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 1: Functional Requirements*, tech. rep., 2010.

[4] FEDERAL AVIATION ADMINISTRATION, *Satellite navigation - ground based augmentation system (gbas)*, Apr 2018.

[5] A. FERNANDEZ, M. WIS, P. F. SILVA, I. COLOMINA, E. PARES, F. DOVIS, K. ALI, P. FRIESS, AND J. LINDENBERGER, *GNSS/INS/LiDAR integration in urban environment: Algorithm description and results from ATENEA test campaign*, in 2012 6th ESA Workshop on Satellite Navigation Technologies (Navitec 2012) & European Workshop on GNSS Signals and Signal Processing, IEEE, dec 2012.

[6] M. S. GREWAL, L. R. WEILL, AND A. P. ANDREWS, *Global Positioning Systems, Inertial Navigation, and Integration*, Wiley-Interscience, 2nd ed ed., 2007.

[7] E. D. KAPLAN AND C. J. HEGARTY, *Understanding GPS/GNSS. Principles and Applications*, Artech, 3rd ed., 2017.

[8] D. KRAJZEWICZ, J. ERDMANN, M. BEHRISCH, AND L. BIEKER, *Recent Development and Applications of SUMO - Simulation of Urban MObility*, International Journal On Advances in Systems and Measurements, 5 (2012), pp. 128–138.

[9] D. KRAJZEWICZ, G. HERTKORN, C. RÖSSEL, AND P. WAGNER, *Sumo (simulation of urban mobility)*, in Proc. of the 4th middle east symposium on simulation and modelling, 2002, pp. 183–187.

[10] N. LU, N. CHENG, N. ZHANG, X. SHEN, AND J. W. MARK, *Connected vehicles: Solutions and challenges*, IEEE Internet of Things Journal, 1 (2014), pp. 289–299.

[11] I. MILLER, B. SCHIMPF, M. CAMPBELL, AND J. LEYSSENS, *Tightly-coupled GPS / INS system design for autonomous urban navigation*, in 2008 IEEE/ION Position, Location and Navigation Symposium, IEEE, 2008.

[12] A. MINETTO, C. CRISTODARO, AND F. DOVIS, *A Collaborative Method for Positioning based on GNSS Inter Agent Range Estimation*, in 2017 25th European Signal Processing Conference (EUSIPCO), 2017.

[13] G. NARDINI, A. VIRDIS, AND G. STEA, *Modeling x2 backhauling for LTE-advanced and assessing its effect on CoMP coordinated scheduling*, in 2016 1st International Workshop on Link- and System Level Simulations (IWSLS), IEEE, jul 2016.

[14] ——, *Simulating device-to-device communications in omnet++ with simulte: scenarios and configurations*, CoRR, abs/1609.05173 (2016).

[15] G. NARDINI, A. VIRDIS, AND G. STEA, *Simulating cellular communications in vehicular networks: making simulte interoperable with veins*, arXiv preprint arXiv:1709.02208, (2017).

[16] OMNET++, *www.omnetpp.org*, Nov 2018.

[17] K. M. PESYNA, JR, Z. M. KASSAS, J. A. BHATTI, AND T. E. HUMPHREYS, *Tightly-Coupled Opportunistic Navigation for Deep Urban and Indoor Positioning*, 2011.

[18] M. SCHREIBER, H. KONIGSHOF, A.-M. HELLMUND, AND C. STILLER, *Vehicle localization with tightly coupled GNSS and visual odometry*, in 2016 IEEE Intelligent Vehicles Symposium (IV), IEEE, jun 2016.

[19] C. SOMMER, R. GERMAN, AND F. DRESSLER, *Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis*, IEEE Transactions on Mobile Computing, 10 (2011), pp. 3–15.

[20] M. TAHIR, S. S. AFZAL, M. S. CHUGHTAI, AND K. ALI, *On the accuracy of inter-vehicular range measurements using GNSS observables in a cooperative framework*, IEEE Transactions on Intelligent Transportation Systems, (2018), pp. 1–10.

[21] S. P. TEASLEY, W. M. HOOVER, AND C. R. JOHNSON, *Differential GPS navigation*, in PLANS '80 - Position Location and Navigation Symposium, 1980, pp. 9–16.

[22] A. VARGA, *The Omnet++ Discrete Event Simulation System*, in European Simulation Multiconference, 2001.

[23] A. VIRDIS, G. NARDINI, AND G. STEA, *Modeling unicast device-to-device communications with simuLTE*, in 2016 1st International Workshop on Link- and System Level Simulations (IWSLS), IEEE, jul 2016.

[24] A. VIRDIS, G. STEA, AND G. NARDINI, *Simulating LTE/LTE-Advanced networks with SimuLTE*, (2016).

[25] Z. XIONG, F. SOTTILE, M. A. SPIRITO, AND R. GARELLO, *Analysis of Hybrid and Cooperative Positioning Algorithms in Urban Canyon Scenarios*, in 2013 International Conference on Localization and GNSS (ICL-GNSS), IEEE, jun 2013.

[26] E. ZHANG, Y. KUANG, W. JIANG, AND M. A. UMER, *Active RFID positioning of vehicles in road traffic*, (2011).