UNIVERSITÀ DELLA CALABRIA

UNIVERSITA' DELLA CALABRIA

Dipartimento di Ingegneria Informatica, Modellistica, Elettronica e Sistemistica

Dottorato di Ricerca in

INFORMATION AND COMMUNICATION TECHNOLOGY

CICLO

XXXI

TITOLO TESI

Settore Scientifico Disciplinare ING INF/03

Coordinatore: Ch.mo Prof. Felice Crupi

Firma _____

Supervisore/Tutor: Prof. Floriano De Rango

Firma _____

Supervisore/Tutor: Ing. A.F. Santamaria

Firma _____

Dottorando: Dott. Pierfrancesco Raimondo

Firma _____

UNIVERSITY OF CALABRIA

DOCTORAL THESIS

Enhancements of Autonomous Vehicles Environment based on smart communication systems

Author:
Pierfrancesco RAIMONDO

Supervisor:
Prof. Floriano DE RANGO,
Ing. Amilcare F. SANTAMARIA

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Information and Communication technologies
DIMES

June 4, 2019

Declaration of Authorship

I, Pierfrancesco RAIMONDO, declare that this thesis titled, "Enhancements of Autonomous Vehicles Environment based on smart communication systems" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"The courage to walk into the Darkness, but strength to return to the Light."

Parables of the Allspring

UNIVERSITY OF CALABRIA

Abstract

Computer Science Engineering
DIMES

Doctor of Philosophy

Enhancements of Autonomous Vehicles Environment based on smart communication systems

by Pierfrancesco RAIMONDO

The first part of this thesis is focused on the vehicular networks and on the different protocols that are used in vehicular environment to exchange information and data. A description of the most used protocols of Vehicular Ad-hoc NETwork (VANET) is presented along with their positive and negative aspects, their issues and some improvements. The capability of exchange data and information is one of the most important features to improve everything making objects smart. Obviously this is valid also for vehicles that can be improved using connectivity and networks. New paradigms and architecture are growing in interest thanks also to new technologies such as lightway protocols or 5G giving birth to new Internet of Vehicles (IoV). Autonomous vehicles are a special kind of vehicles able to move in an urban environment without any human interaction thanks to several sensors equipped onboard. These sensors can detect obstacles and possible dangerous situations on the road during a journey. To ensure a high reliability it is necessary a cooperation between all equipped sensors like cameras, Lidar, Radar and GPS. Image and video elaboration is made through computer vision algorithms that are able to identify and track objects moving inside sensors line of sight while lidar and radar create a three-dimensional model of the surrounding environment. These gathered information have only a local scope. In order to solve this problem a "collaborative" approach could be used to establish a communication network among these vehicles. In this way every autonomous vehicle could be able to handle warning situations and traffic jam in a better way exploiting information exchange with others. It is important to build a robust network infrastructure in order to achieve a reliable message exchange. Vehicles , equipped with on board units, are starting to be considered part of the IoT domain, where instead of classical smart objects we consider smart vehicles. These smart vehicles can exchange messages among themselves to improve their useful environment knowledge. New routing protocols and optimization algorithms are necessary in order to make the network infrastructure more flexible and able to adapt to changes. Furthermore vehicles move at a very high speed in comparison to other common devices and the link low stability makes the network high dynamic and unstable. To resolve these issues it will be necessary to investigate actual VANET protocols and improve them to better manage its high dynamism and link quality using also technologies that belong to Internet of Things domain. Integrating these heterogeneous technologies together it will be possible the creation of a middleware between the IoT layer and On Board Units. These solutions, based on evolutive algorithms and adaptive protocols have to be verified through simulations and mathematical models. To do so, a deep study on mobility models and simulation tools has been necessary. The future of mobility is, taking into consideration the recent trends, electric. Fuel propelled vehicles will some day disappear from our city to leave plate to this new kind of vehicles with electric engine and powered by batteries. Electrical mobility will be a very important part of the future and also autonomous vehicles will be electric. The consumption of on-board vehicle sensors must be taken into account and must be modeled by simulation tools that are used

to describe future mobility scenarios. In the end a possible application of all studied research fields is proposed, summarized inside the proposition of a platooning application for autonomous vehicles. A platoon consists of a set of vehicles that moves at the same speed along the road with a low inter-vehicle gap. In this the members of the platoon will benefit of several advantages that will be treated in the body of this thesis.

Acknowledgements

I would like to thank all the lab staff that helped me to write this thesis and especially my supervisors Floriano De Rango and Amilcare Francesco Santamaria. Thanks also to my collegue Vincenzo Inzillo that suffered with me during these three hard work years. A great thank goes to Dalila for letting me eat a lot of pizza and chocolate (not together) and for sharing with me important moments of our lives. I also want to mention here my friends Tommaso, Fabiola and Pepo that endured the pain of a stressed, cynical and nichilist me. This holds true mostly for Tommaso and Fabiola because Pepo is like an anthropomorphous cat that observes and judges. Thanks also to Ivonne (the final "e" is important) for my bad English checks and for all the coffees that offered me in these years, they are quite a lot.

One day at time three years have passed

Contents

List of Figures

List of Tables

Acronyms

AC Autonomous Car. 3, 69, 72

ACC Adaptive Cruise Control. 77, 90, 93

AI Artificial Intelligence. 25, 26, 28, 46, 48

AODV Ad-hoc On-demand Distance Vector. 14

AT Autonomous Things. 3

AV Autonomous Vehicle. 1, 50, 126

BMS Battery Management System. 54

BSS Basic Service Set. 12

CACC Cooperatve Adaptive Cruise Control. 86, 90, 91, 93, 135, 136, 144

CC Cruise Control. 93

CoAP COnstrained Application Protocol. 21, 22, 25

CoV Cloud of Vehicles. 22, 29, 30

D-MAC Directional MAC. 12

DES discrete-event simulation. 55

DSRC Dedicated Short Range Communication. 11, 12

EG-RAODV Evolving Graph Reliable Ad Hoc On-Demand Distance Vector Routing Protocol. 14

ETSISmartM2M European Telecommunications Standards Institute Smart Machine2Machine. 25

EV Electric Vehicle. 53–55, 68, 70–72, 123, 133

FIB Forward Information Base. 14

FL Fuzzy Logic. 25

FPV First Person View. 1

FSCTP Fast and Scalable Transmission Control Protocol. 18, 109, 112, 114, 115

FSM Finite State Machine. 50

GNSS Global Navigation Satellite System. 45

GPL General Public License. 88

GPRG Grid Based Predictive Geographical Routing. 13

GPS Global Positioning System. 4, 16, 26, 28, 45, 46, 93, 95

GWO Grey Wolf Optimizer. 101, 105, 137–139

HBEFA Handbook Emission Factors for Road Transport. 66

HLAR Hybrid Location based Ad-hoc Routing. 14

HTTP Hyper Text Transfer Protocol. 22

iCAR InterseCtion based traffic Aware Routing. 14

ICT Information and Communication Technology. 10

IEEE Institute of Electrical and Electronics Engineers. 12, 33

IMU Inertial Measurement Unit. 43

IoT Internet of Things. 21, 22, 24, 25, 29, 30, 116, 122, 143, 144

IoV Internet of Vehicles. viii, 21, 25, 143

ITS Intelligent Transportation System. 10, 14, 29, 143

KERS Kinetic Energy Recovery System. 68–70, 123

LAS Local Area Service. 25–31, 116, 120

Lidar Light Detection and Ranging. 35, 36, 41–44

LTE Long Term Evolution. 32, 33

LwM2M Lightweight Machine2Machine. 25

M2M Machine2Machine. 116

MANET Mobile Ad-hoc NETwork. 10, 11

MAPE-K Monitor, Analyze, Plan, Execute and Knowledge. 48

MD Membership Degree. 32

MEMS Micro Elctromechanical Mirrors. 43

MQTT Message Queueing Telemetry and Transport. 21, 25, 27, 29, 116

NDN Name Data Networking. 14

NED NEtwork Description. 87

NEMO NEtwork MObility. 14

NS Network Simulator. 86

OBD On-Board Diagnostic. 25, 32, 116

OBU On Board Unit. 15, 16, 19, 110

OFDM Orthogonal Frequency Division Multiplexing. 12

OSM Open Street Map. 64–66, 71, 76, 121, 123, 126

PDDL Planning Domani Definition Language. 48

PDF Probability Density Function. 32

PIT Pendint Interest Table. 14

PLEXE PLatooning EXtEnsion. 85, 90–93, 98, 134, 144

PPP Precise Point Positioning. 46

ProSe Proximity Service. 33

QoS Quality of Service. 29, 32

Radar Radio Detection and Ranging. 36, 39

ReST Representational State Transfer. 22

RSU Road Side Unit. 12, 14–17, 19, 113

RTK Real Time Kinematic. 46

SAE Society of Automotive Engineers. 1, 5, 6

SAS Self Adaptive System. 48

SPAN Synchronous Position Attitude and Navigation. 45

STRIPS Standford Research Institute Problem Solver. 48

SUMO Simulation of Urban MObility. 58–60, 62, 63, 66, 68, 77–80, 85, 88–93, 109, 110, 113, 120, 125, 126, 131, 143

TCP Transmission Control Protocol. 13, 18, 19, 60, 89, 92, 112, 113

TraCI Traffic Control Interface. 60, 61, 89, 92

UAV Unmanned Aerial Vehicle. xiii, 2, 3

UDP User Datagram Protocol. 13, 18, 19, 112, 113

UGV Unmanned Ground Vehicle. xiii, 2, 3

US Unmanned Spacecraft. xiii, 2, 3

USGS United States Geological Survey. 65

USV Unmanned Surface Vehicle. xiii, 2, 3

UUV Unmanned Underwater Vehicle. xiii, 2, 3

V2I Vehicle-to-Infrastructure. 10, 15

V2V Vehicle-to-Vehicle. 10, 12, 15, 113

VANET Vehicular Ad-hoc NETwork. viii, xiii, 10, 11, 13, 14, 17–19, 23, 25, 85, 88, 92, 109, 112, 123, 143, 144

VC-MAC Vehicular Cooperative MAC. 12, 13

VEINS VEhicles In Network Simulator. 85, 88–93, 110, 112, 120, 134, 143

VSU Vehicle Status Update. 26–28, 30

WAVE Wireless Access in Vehicular Environment. 12, 99

WPA Weapon Target Assignment. 137

XML eXtensible Markup Language. 63, 66, 69

To my family always supporting me

To my niece that has started her journey called life since a few months

To all the people that cherished and helped me in all possible ways. I probably would not have been able to finish this without your help

Chapter 1

Introduction

This chapter will briefly introduce the concept of autonomous vehicle, highlighting the difference between unmanned vehicles called drones and autonomous car with passengers. In the first section a small classification of drones and autonomous drones is presented. In the second part the focus is on the autonomous cars and on the classification of autonomy levels of Society of Automotive Engineers (SAE).

1.1 AV and UV

An Autonomous Vehicle (AV) is a vehicle that is capable of moving inside an environment without any human interaction. Nowadays these kind of vehicles are becoming a concrete reality and all big car manufacturers have a Research&Development department working on this kind of technology. Autonomous vehicles can be divided in two main categories :

- Unmanned vehicles
- Vehicles that carry passengers

An unmanned or uncrewed vehicle is commonly known as drone and it's a vehicle without a human pilot aboard. Drones can either be remote controlled or autonomous. These kind of vehicles are used for many applications and tasks that would be dangerous or lethal for human beings or to reach places that are normally unreachable by normal means. They have been developed and deployed for both civilian and military for achieving a set of dull, dirty and dangerous activities. In fact, the motivation behind the implementation of an unmanned vehicle is the safeguard of human life in very dangerous activities. The first drones that have been developed are remotely controlled drones in order to allow the completion of very dangerous tasks from a safe distance for the human operator. To manage complex task a First Person View (FPV) video streaming is necessary. So, a video camera mounted on the drone transmits a live video to the operator enabling a more precise control. Latest radio transmitting technology allows to realize a video broadcast at a distance of several kilometers. However, remote radio controlling is not possible in some hostile environments i.e. under the sea. For this reason , also thanks to technological progress, in the last years drones have been equipped with an on-board

intelligence to perform tasks autonomously with minimal or even without human interaction. There exists different types of these drones :

- UGV
- UAV
- USV
- UUV
- US

1.1.1 UGV

An UGV is a ground vehicle that operates while in contact with the ground. The UGV is the land-based counterpart of UAVs and UUVs. Some use cases for these vehicles are handling explosives by disarming or detonating bombs, military applications like surveillance reconnaissance and target acquisition, agriculture applications like short window harvesting or spraying and thinning tasks. The ground movement is assured by a variety of locomotion mechanism that depends on the type of ground on which the vehicle has to move. Some ground drones move on wheel while others that have to face a more difficult environment have tracks. In the last years new prototypes able to face any type of ground have been developed. These prototypes use an adaptive locomotion similar to humans and animals through the use of legs [1].

FIGURE 1.1: An UGV with legs locomotion.

1.1.2 UAV

UAVs, commonly known as drones, are aircraft without a human pilot aboard. While they were originally developed for military applications, in the last years their use is rapidly expanding to commercial, scientific and recreational environment. Nowadays it is possible to buy an UAV at a relatively low price, depending on the equipment and materials. Civilian UAVs now vastly outnumber military ones, with estimates of over a million sold in 2015. They can be seen as an early commercial application for Autonomous Things (AT) , directly followed by Autonomous Car (AC) and home robots. Like the major part of technologies they were designed to carry out 3d missions *dull*, *dirty* or *dangerous*. One of the key aspects for these vehicles is the energy consumption because of the energy needed by rotors to lift the drone body. For this reason very light and resistant materials are used and a good trade off between battery capacity and battery weight must be found in order to maximize flight time [2].

1.1.3 USV&UUV

These kind of drones are able to move respectively on the surface of the water and in deep waters. While the former hasn't to face many problems and are substantially similar to the UGV, the latter has to face more issues. Unlike other forms of unmanned vehicles they must move inside an insidious medium for remote communications. The vehicle ability to communicate in real time is hindered by the distortions caused by water as well as the multitude of obstacles that the environment presents. For these reasons along with new transmission technologies also the study for autonomous task accomplishment is evolving in a really fast way. Main applications for these kinds of drones are the disposition of underwater mines, ship hull inspection, nuclear reactor decontamination , exploration, mining and drilling [3].

1.1.4 US

USs are spacecrafts with varying level of autonomy that are capable of performing a pre-programmed list of operations on their own. Alike UUV these vehicles were designed to perform complex tasks without any human interaction. Due to the extreme distance from the earth it would be impossible to maneuver them in a real time way. For this reason only essential messages are sent to these vehicles and all the routines to perform tasks are pre-loaded.

1.2 Autonomous Cars

An autonomous car is a car in which human drivers are never required to take control to safely operate the vehicle. Also known as *driverless* cars, they combine sensors and software control to navigate and drive the vehicle. Currently there are no legally

fully-autonomous operating vehicles but only partially-autonomous cars and trucks with a varying percentage of autonomy. There are various technologies that have been adopted to enhance drivers quality of life and have been slowly adopted in the new produced cars. Some of these technologies are :

- brake assistance
- lane assistance
- collision avoidance
- park assistance

Brake assistance technology helps to generate the maximum braking power available even if the pressure on the pedal is not enough. The lane assistance technology warns drivers when the vehicle begins to move out of its lane in order to minimize accidents due to others lane invasion. A collision avoidance system is a safety system designed to prevent or reduce the severity of collisions. It can be implemented through the use of different sensors that can evaluate in real time the distance from the preceding car. Sensors are also used to evaluate if there is enough room to fit the vehicle for parking. Park assistance technology can help the driver to perform the parking task by notifying the space left in the gap or by managing the entire procedure fully autonomously. The major differences between autonomous cars and autonomous drones are that cars have passengers and their safety must be ensured in any situation. Moreover cars needs to move inside an urban environment in which also vehicles that are driven by humans are present. To do so they must be able to recognize road signs and horizontal signage as well as having a knowledge of road laws. In order to do so these vehicles equip large set of sensors to perceive the environment in which they are moving. Sensors equipped on these vehicles will be discussed in the next chapter but can be mainly divided in three groups [4].

- Video sensors
- Radar and Lidar Sensors
- Field Sensors

Video sensor are capable of capturing a video stream of the surrounding environment through cameras. The video stream then is computed by the computational unit. Radar and Lidar are sensors based on the wave refraction. Field sensors analyze the changes, i.e. electromagnetic field to detect metal obstacles in the way. The range and precision of these sensors depend on the used technology and on their working principle. Some sensors can detect an obstacle at a hundred meters of distance while others have a range of few centimeters. To determine exact position of the vehicle on the streets also Global Positioning System (GPS) technology is necessary equipped on-board of these vehicles. GPS alone is not enough accurate in computing the exact position of the vehicle on the earth surface, so position

data gathered from satellites signals is combined with other sensors. Thanks to the information given by accelerometers, gyroscopes and other devices it is possible to have an accuracy of few centimeters even if the satellite signal is not accurate or it is lost for few seconds. The data gathered from all these sensors are then elaborated by a computational unit to create a 3-dimensional model of the surroundings. This model is then used to reach the destination avoiding obstacles. The on-board intelligence has also to forecast the movement of other cars or pedestrian. This routine must be always running and must respond in a real time way to avoid accidents. An autonomous car must be able to manage every situation in a real time way, in order to ensure the safety of passengers and other humans inside the urban environment [5].

FIGURE 1.2: Autonomous car prototype made by Google.

1.3 Autonomous Driving Classification

The SAE provided in 2016 a taxonomy for motor vehicle driving automation systems [6]. It provided a detailed definition for 6 levels of automation describing the features equipped on-board for each level and also addressing the three primary actors in driving: human driver, driving automation system and other vehicles. In this classification active safety systems such as electronic stability or braking assistance, collision avoidance and so on are not considered because they do not perform part

or all of the driving task. These safety systems only activate during emergency situation in order to ensure the safety of passengers and do not change or eliminate the role of the driver in performing driving tasks. However, often this kind of systems are also equipped on autonomous vehicles at any level of autonomy. In Fig. 1.3 the different types of autonomous levels in the SAE classification are shown. It is important to state that the level of autonomous driving that is currently available for commercialized cars is 3. In the next section a brief description of each level will be provided in order to understand the current state of development of these vehicles.

SAE level	Name	Narrative Definition	Execution of Steering and Acceleration/ Deceleration	*Monitoring* of Driving Environment	Fallback Performance of *Dynamic Driving Task*	System Capability *(Driving Modes)*
***Human driver* monitors the driving environment**						
0	No Automation	the full-time performance by the *human driver* of all aspects of the *dynamic driving task*, even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a
1	Driver Assistance	the *driving mode*-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	Human driver and system	Human driver	Human driver	Some driving modes
2	Partial Automation	the *driving mode*-specific execution by one or more driver assistance systems of both steering and acceleration/ deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	**System**	Human driver	Human driver	Some driving modes
***Automated driving system* ("system") monitors the driving environment**						
3	Conditional Automation	the *driving mode*-specific performance by an *automated driving system* of all aspects of the dynamic driving task with the expectation that the *human driver* will respond appropriately to a *request to intervene*	System	**System**	Human driver	Some driving modes
4	High Automation	the *driving mode*-specific performance by an automated driving system of all aspects of the *dynamic driving task*, even if a *human driver* does not respond appropriately to a *request to intervene*	System	System	**System**	Some driving modes
5	Full Automation	the full-time performance by an *automated driving system* of all aspects of the *dynamic driving task* under all roadway and environmental conditions that can be managed by a *human driver*	System	System	System	**All driving modes**

FIGURE 1.3: The SAE classification for autonomous driving.

1.3.1 Level 0

The first level of the SAE classification describes a vehicle with no automation at all. The driver must perform all the driving task with eventually some assistance from the on-board systems.

1.3.2 Level 1 "hands on"

In this level the driver must take care of the driving task but the control of some functions can be shared with the automated system. When the automated systems are in control the driver must be always ready to intervene if something unexpected happens. Some examples of Level 1 automations are:

- the cruise control in which the driver controls the vehicle steering while the autonomous system decides the speed of the vehicle.

- Park assistance in which the vehicle takes care of the parking procedure but the driver has to be ready to retake control at any moment.
- A lane keeping Assistance that notifies the driver and eventually takes control in the case of crossing unawares the lanes dividing line.

1.3.3 Level 2 "hands off"

This level is known as hands off because the automated system takes care of each driving task performed in ordinary and specific scenarios. The driver is not required to follow each task because the automated system is able to handle them but can be notified if something was wrong to take control. The contact between driver's hand and wheel is mandatory.

1.3.4 Level 3 "eyes off"

The driver can turn his attention away without any fear while the automated system takes full control of the vehicle. The vehicle is capable of handling all possible events and situations that need an immediate response but the system can notify the driver for any intervention that is demanded. An example of a specific scenario in which the vehicle is capable of handling all task is the *traffic jam scenario*. In this scenario the vehicle is moving slowly due to the traffic jam and, for the recently commercialized cars, a physical between the lanes must be present.

1.3.5 Level 4 "mind off"

Level 4 is the same of level 3 but the driver intervention is never demanded because the vehicle is capable of handling any situation that may occur. The driver, in this way, is free to leave its seat or to sleep. This level of autonomous driving can be supported, for now, only in restricted areas or under specific circumstances. Outside these circumstances the vehicle must be capable of aborting the trip and find a safe parking slot in order to allow the driver to retake control of the vehicle.

1.3.6 Level 5 "steering wheel optional"

In this level the steering wheel is an optional and all the driving task can be performed autonomously by the vehicle. An example would be an automated taxi capable of moving inside an urban environment with no human interaction at all except for the destination setting. Vehicles that are compliant to this level are currently only prototypes.

Chapter 2

Vehicular Communication

The use of communication inside the vehicular environment leads to several advantages in terms of users quality of life and safety. A lot of effort has been put in vehicular communication in order to develop protocols and applications to enhance safety in urban environment that allows also positive outcomes for the urban environment. Through communication it is possible to achieve also a reduction of dangerous emissions inside the urban environment by managing in a smart way traffic flows to avoid jams. In this way it is possible to reduce the vehicle stay time inside the city and prevent these problems before they become significant. A lot of researches have been done in this field of interest with important results that show how a simple message exchange among vehicles can enhance overall safety. In further sections networking structures that allow vehicles communication will be presented. An autonomous vehicle should not rely, at least for now, on communication to accomplish the basic task of driving. At the actual stage it is impossible to provide a constant and failure-less access to the network and an autonomous vehicle has as its prime directive to ensure the safety of passenger and other human beings inside urban environment. However, communication can be used to drastically enhance some functions and services to improve overall quality of life. It is also very important to highlight the behavior difference between a human driver and an autonomous driving system when receiving information. First of all, an autonomous vehicle is able to process and react thousand times faster than a human driver. The reaction time is crucial when dealing with dangerous situations that need a fast response in order to ensure the safety of passengers. Secondly we have to consider the possibility that an important information could be ignored or missed by a human driver. I.e. in a scenario in which a message suggests a route change that slightly increases the travelling time but improves the overall quality of life by decreasing the average travelling time of all vehicles, a human driver could decide to be selfish and ignore the information. An autonomous vehicle, instead, would choose to perform a rerouting to improve overall quality of life [7].

2.1 VANET

Information and Communication Technology (ICT)s are one of the key factors that have pushed the development of innovating solutions in the automotive industry and society. In the last years technology made important steps forward that allowed the birth of new paradigms based on devices connected every-time and everywhere. Sharing information among vehicles and infrastructure is enabling a lot of applications for safety and traffic efficiency but also for a better quality of life. This kind of applications are good examples of what is called Intelligent Transportation System (ITS). The main goals of an ITS are to improve safety, efficiency and overall quality of life inside the urban environment thanks to the adoption of new ICT technologies. One of the key factors for the development of an ITS is the possibility to exchange information between vehicles [8]. One of the hottest topics in the networking research field are precisely VANETs. a VANET can be considered as a special kind of Mobile Ad-hoc NETwork (MANET) in which vehicles that equip special kind of devices can create spontaneous networks while moving inside the city [9]. The information exchange between vehicles is possible even if there is no infrastructure support thanks to direct communication. This kind of networks are considered one of the most suitable technologies for the realization of an ITS end for this reason protocols and transmitting technologies are constantly evolving. For the implementation of this vision communication is not enough so vehicles have to equip on-board sensors and computing capability to elaborate them. All future vehicles will have a digital heartbeat able to compute information coming from on-board sensors and from other vehicles. The purpose of VANET architecture is to enable communication among nearby vehicles and between vehicles and infrastructure equipment on the roadside. For this reason we can classify networks in three categories [10].

- Vehicle-to-Vehicle (V2V) networks
- Vehicle-to-Infrastructure (V2I) networks
- Hybrid networks

In V2V networks vehicles exchange messages only among them without any infrastructure support. This kind of communication is mainly used for safety applications because of the short range that the communication allows. V2I networks allow a communication between vehicles and roadside equipment. These kinds of connection are mainly used to gather information and data. Thanks to the these connections it is also possible to allow internet connection to vehicles. Hybrid connections enable message exchange single and multi-hop. In this way each vehicle is able to reach an infrastructure node through the routing of messages among vehicles. This is possible only if a forwarding path exists between the source vehicle and the destination roadside unit using as relay nodes the vehicles in the path. These kinds of network configurations are shown in Fig. 2.1. There are several aspects in common

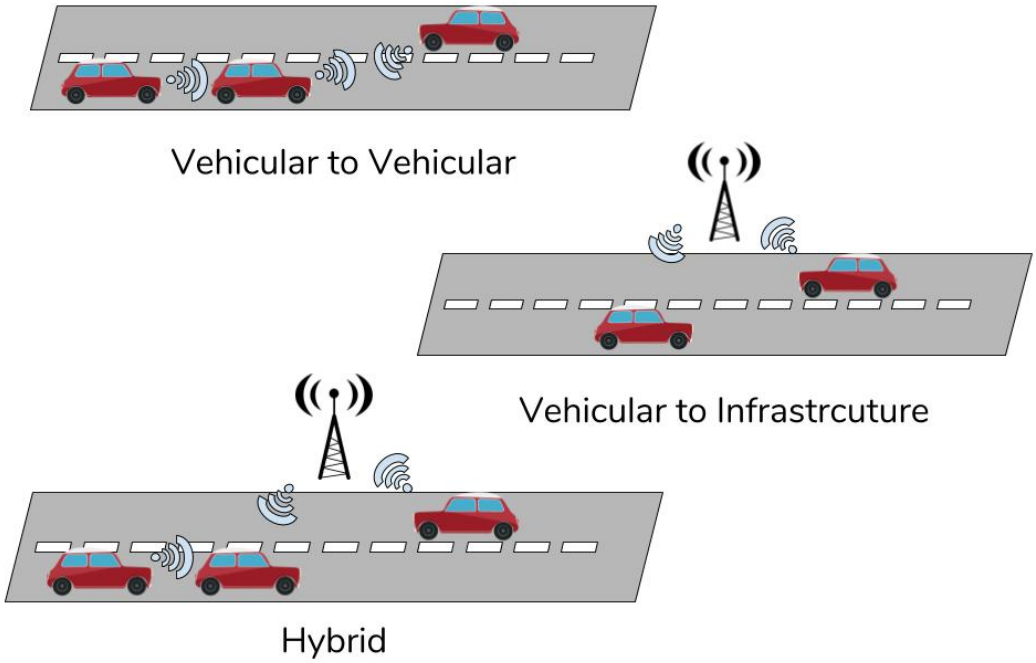

FIGURE 2.1: VANET networks types

between MANET and VANET such as low bandwidth and short transmission range but VANET networks have a higher dynamic degree due also to the higher movement speed of nodes. The link between two nodes can quickly disappear while new ones with other nodes are created. Moreover, in vehicular environment nodes are static , in case of a infrastructural node, or move constrained to specific pathways represented from streets. Also the propagation model is very different because vehicles move in three types of environment: highway, rural and city. While on highways the propagation model is similar to the free space model, in cities and rural areas the communication is more complex due to the presence of buildings, trees and other object that can be considered as obstacles [11].

VANET protocol Stack To deal with all the peculiar characteristics that have been previously introduced a protocol stack for vehicular networks has been developed [12]. The modified protocol stack will be briefly discussed:

1. Physical Layer
   The physical layer is in charge of transmitting through a mean data coming from upper layers. Inside an urban environment wireless transmissions are deeply influenced from many factors. The frequency is influenced by Doppler effect and a high number of obstacles cause multi-path fading. Even if some infrared and millimeter waves solution have been tested for vehicular communication the community preferred a medium-range technology called Dedicated Short Range Communication (DSRC) that operates in the 5.9 GHz band.

In this way connections can reach a 1 Km range in ad-hoc fashion and each connected node can move freely. Moreover, the available bit-rate is relatively high from 6 to 54 Mbps if two service channels are used contemporary. In order to reduce neighbors interference more channel are used. DSRC is very suitable for the canonical network infrastructure in which the edge router are called Road Side Unit (RSU). Each RSU can work as a router between to vehicle nodes on the road or as an access to external networks e.g. internet. In this way data exchange for safety or infotainment applications is available for both roadside-to-vehicle and V2V. Each channel of DSRC has a 10 Mhz bandwith and the whole band is divided in six service channels and one control channel as shown in Fig. 2.2. DSRC is part of the Institute of Electrical and Electronics

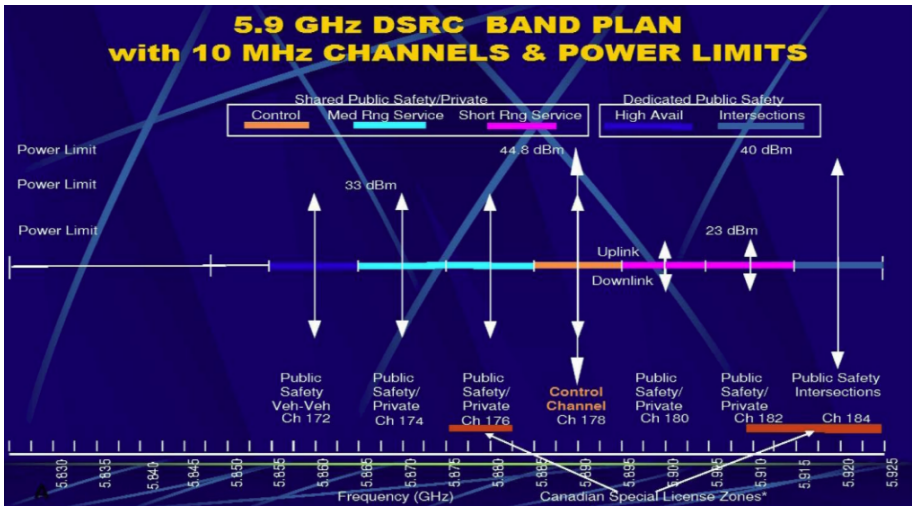

FIGURE 2.2: The DSRC channel division.

Engineers (IEEE) 802.11 p Wireless Access in Vehicular Environment (WAVE) [13]

2. MAC layer
   MAC layer have to provide a reliable access to the transmission channel. In vehicular environment these kinds of protocol have to deal with the hidden station problem and also with an optimal band-with share among nodes. The hidden station problem is remarkable in scenarios where a high number of nodes are connected causing a decrease of the data transfer. In order to tackle these issues a Orthogonal Frequency Division Multiplexing (OFDM) technique to avoid collisions is preferred. To allow a safe and fast communication among nodes the simplification of the Basic Service Set (BSS) has been adopted. In this way the creation and disruption of an ad-hoc connection among nodes is lighter and simpler. Some mac protocols that have been developer for vehicular environment are i.e. Directional MAC (D-MAC) and Vehicular Cooperative MAC (VC-MAC). The D-MAC focuses on the exploitation of several antenna in order to back the connection on the remaining antennas if some of them has

a malfunction. The VC-MAC instead focuses on the transmission of the information among group of vehicles electing some vehicles inside a groups as relay nodes to minimize interference ensuring the reception of the data.

3. Network Layer
   The network layer is responsible to provide a reliable communication between two nodes of the network. In vehicular environment different communication paradigm must be supported like *unicast*, *multicast* and *broadcast*. The strategies that can be used in multi-hop networks like VANET can be divided into two main categories. Topology based and position based. Topology based strategies imply that every node have to construct a table structure to manage the forwarding of packets. These data structures can be built when are needed in a real time way (proactive protocols) or can be maintained during the whole time using protocol messages. Position based protocols exploit the knowledge about the position of the nodes inside the networks to better manage the packet forwarding. Obviously this division is not always so defined and we can acknowledge also an hybrid category between these two.

4. Transport and Application Layer
   Some complex and evolved applications inside the vehicular environment need similar service provided by Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) protocols. Unfortunately protocols designed in this way are not applicable to VANET scenarios with high mobility and dynamicity. Vehicles in urban environment move at a very high relative speed and connection link can last only few seconds. Moreover, these kind of protocols should minimize the end-to-end delay to provide emergency information in safety applications. Probably these two layers are more challenging than the others and are the most interesting research area in the last years. A lot of protocols have been proposed to comply these requirements and are still a hot topic even now.

2.1.1 VANET Routing Protocols

Routing protocols in vehicular networks are still improving and this area of research is improving rapidly. These networks use a multi-hop scheme in which data packets are passed through each relay node until they reach their destination [14]. Usually the destination node is a vehicle moving along the roads so the choice of relay nodes for routing should be guided by their geographical position. The Grid Based Predictive Geographical Routing (GPRG) use a two dimensional grid map to predict all possible future position of a destination node in order to select a set of possible relay node to route data packets. VANET routing protocols can be divided in two main categories : topology-based protocols and geographic/position based protocols. Topology based protocols perform the forwarding of packets through relay node using only the information about the link state while position based protocols

are not table driven and do not share link state information. The latter category only use information about the spatial position of nodes achieving in this way a good scalability. The Hybrid Location based Ad-hoc Routing (HLAR) routing protocol tries to combine the strength of these two routing categories by switching to topology based reactive protocol when the geographic information is degraded or the density of vehicles is not good enough. Applications in the area of ITS usually perform only a single hop transfer in order to deliver important information to the near nodes. Another issue to face when performing multi-hop routing in VANET is the congestion in crowded areas. The InterseCtion based traffic Aware Routing (iCAR) protocols tries to cope with these problems using the information of an offline map of the area and real time traffic advertisement [15]. In this way the protocol is capable of balancing the load of the networks distributing the data packet paths among alternative routes. RSU can also be used to give internet access to the vehicles in its coverage area. These vehicles move fast and for this reason an improved management of the handoff in ip-based network is necessary to have an efficient re-assignment of ips. NEtwork MObility (NEMO) protocols tries to ease the hand-off procedure by involving other vehicles. Other vehicles act as relay node to allow a longer/farther communication with the RSU to successfully accomplish the task. Another way to cope with the high dynamic network topology issue is the analysis of the network topology in order to predict its evolution in the near future. Evolving Graph Reliable Ad Hoc On-Demand Distance Vector Routing Protocol (EG-RAODV) protocol applies a standard Ad-hoc On-demand Distance Vector (AODV) routing protocol but taking into account the possible topology evolution of the network. [16]

2.1.2 Named Data Networking in VANET

One of the most troubling characteristic of VANET is the intermittent connectivity due to the high speed of the vehicles. One possible solution to this problem is to switch to Name Data Networking (NDN) network paradigm [17]. With this paradigm the information are forwarded not on the basis of the position of the recipient but only on the content. In this important information can be spread in a very fast and efficient way. One of the main problems for NDN with a wireless interface is the impossibility to do not forward the message through the input interface. This problem is faced through the implementation of a smart forwarding of packets on the top of the blind forwarding. A way to create a path for content dissemination is proposed in [18]. In this proposal a content request message is broadcasted until it reach a node able to provide the information needed. Each relay node stores in a table the id (MAC address) of the node from which has received the message. When the node with the requested information receive the packet switch the origin address with the destination address and send towards the already formed path the data. All these tasks are performed through the use of two tables : the Forward Information Base (FIB) and the Pendint Interest Table (PIT). The PIT contains the information about already forwarded but not satisfied interests request while the FIB contains

information about how to forward messages depending on the content name. In order to improve performance of the protocol also a caching message table is added named Content Store in which are stored previously received data path that may satisfy future requests.

2.2 VANET applications to enhance Quality of Life

The communication in vehicular environment can improve overall quality of life in cities because it provides the basics for a set of applications that cannot work on classic vehicles. Using IEEE802.11p standard it is possible to design protocols able to increase passive and active safety systems [19]. An example of safety enhancement in vehicular environment through communication can be found in [20]. In this proposal vehicles share with with each neighbor and with equipment installed on the road infrastructure called RSU useful information. These information are continuously gathered to detect possible problems with traffic flows and congestion. Once one of these conditions is detected all fathered information are spread into the network allowing all informed vehicle to react in time to avoid congestion or dangerous situations. The proposed architecture is composed of a global city road manager that exchange information with the RSU in a periodic manner. The information about the status of roads and flows is sent from the manager towards the vehicle through V2I communication . Each RSU is responsible of a certain localized geographical area and send the status message only to the interested vehicles. The update is periodic and RSU communicate with the traffic manager every T_{update} seconds asking for network topology changes. Obviously each RSU only asks about the covered lanes of interest. A topology change can be considered, for example, a weight on a lane that is increased meaning that the lane in question is suffering congestion. The reference architecture can be seen in Fig. 2.3.

One of the main issues for the traffic flows inside the city is the delay with which a car can take countermeasures to avoid jams. These blocks are commonly recursive and the traffic congestion increase in an exponential way due the fact that vehicles can't find feasible exit gate in time. This represents a key factor for traffic management because several surveys have already proven that the higher the traffic is the higher the CO_2 level is spread into the air. Thanks to a continuous monitoring of the urban environment it is possible to avoid the formation of jams reducing in this way the concentration of CO_2 in the air. On each vehicle is equipped an On Board Unit (OBU) that is slightly different from the classic one used in IEEE802.11p. In this architecture the OBU represents the core of the vehicle system and is able to communicate with the external world exploiting V2V and V2I communications. Each vehicle consider another vehicle as a neighbor if the distance between them is lesser than a threshold

$$\begin{gathered} \text{rdistance} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \\ \text{rdistance} < \text{NeighThreshold} \end{gathered} \tag{2.1}$$

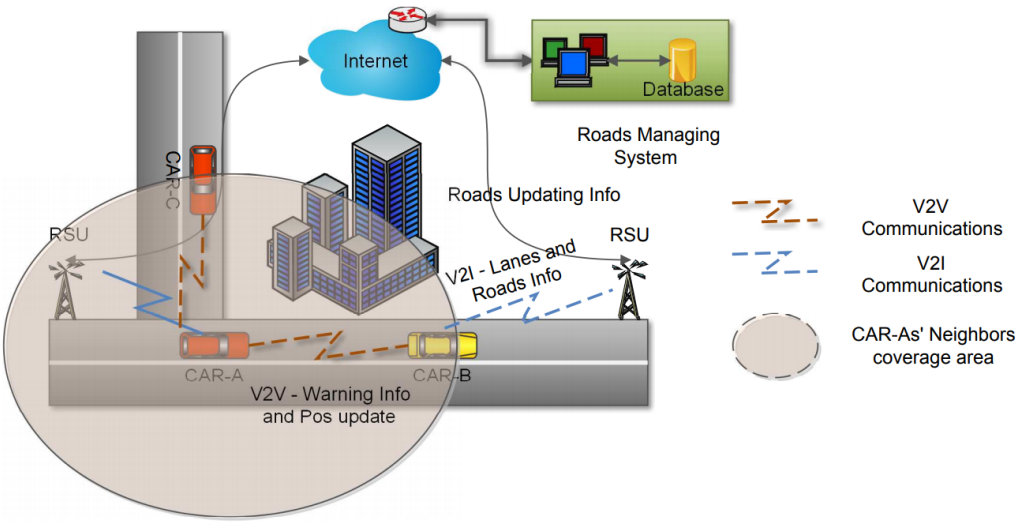

FIGURE 2.3: Reference Architecture

In this case we consider the first node position as the centre of the Cartesian plane $p_1 = (x_1, y_1)$ while the position of the second node is $p_2 = (x_2, y_2)$. The third dimension is not considered because we suppose that city areas are distributed on a plane or quite plane surfaces. The neighbor list is built upon the exchange of the *PosUpdate* message. This message is periodic and its propagated only if the recipient is at distance lesser than the threshold. In this way it is possible to dramatically reduce the protocol overhead. The most important entities that work in VANET and networks layer are :

- OBU
- RSU
- the road traffic manager

As said befor the OBU is slightly enhanced respect to the classic version. In this version the core can query the sensors data and the position from GPS, summarize them and then send them to other vehicles using GeoWave protocol. This OBU can receive both network layer message exploiting multihop commnuication and vehicle direct messages. The message that the OBU can send and receive are :

- *PosUpdate* - This message is generated by each vehicle in a periodic way and sent in broadcast. It can be received by both vehicles and RSUs and is used to inform them of the current position of vehicles. This message is also used to build each node neighbor list.
- Warning Message (*WRGM*) and Accident Data Message (*AccDataMsg*) - These are messages used to warn about dangerous situations . These messages are sent in geocast and broadcast respectively.

- Congestion Update (*CongUdate*) - This message is received from the network layer and it is generated by the traffic manager to alert bout road congestion symptoms or blocks. It is sent in a geocast manner.

The RSU main task is to work as a connector between vehicles and Control and Management Center. It also spread messages along the network and monitor the covered roads. When a vehicle enter the covered area of an RSU it becomes one information supplier helping increasing real time knowledge of the status of the roads. Each RSU is equipped is tha GeoWave Data Base to avoid multiple duplicate message send. The Road Traffic Manager is in charge of managing the whole city traffic exploiting the communication on the VANET layer. It receives the information about traffic from each RSU responsible of a coverage area and balance the traffic load of different flows to avoid congestion and jams. When a congestion is detected it sends warning messages to the interested RSU areas [21]. The important thing to avoid is the creation of a new jam just because all the vehicle are redirected in the same area. For this reason a smart re-routing an weighting policy is necessary. It is necessary to consider different parameters such as travelling time and number of vehicles on the roads.

$$tt_j = \frac{\left|n_j\right| \cdot l_j}{\sum_{i=1}^{i=\left|n_j\right|} S_{i,j}} : j \in R \tag{2.2}$$

$$v_{\text{space}} = \frac{1}{n_j} \cdot \left(\sum_{i=1}^{n_j} length\left(vehicle_i\right) + \frac{\sum_{i=1}^{n_j} speed\left(vehicle_i\right)}{k_{speed}} \right) \tag{2.3}$$

$$h\left(n_j\right) = \left\{ n_j - \left[\frac{1}{2} \cdot \frac{l_j}{v_{space}} + \left(\frac{\sum_{i=1}^{n_j} length\left(vehicle_i\right)}{n_j} \cdot \delta_{space} \right) \right] \right\} \tag{2.4}$$

$$K_{cong,j} = \begin{cases} \alpha & \text{if } h\left(n_j\right) \geq Th_{\text{cong}} \\ \beta & \text{otherwise} \end{cases} \tag{2.5}$$

The parameters used in the equation are explained in Tab. 2.1. The cost of the edge is calculated as

$$w_j = tt_j + \left\{ K_{\text{cong}} \cdot \left(\frac{1}{UpBound_j} - \frac{1}{\text{InRate}_j} \right) \cdot \left[n_j \cdot v_{\text{space}} \right] \right\} \tag{2.6}$$

$$\text{cost}_j = \begin{cases} tt_j & \text{if InRate }_j = 0 \\ w_j & \text{otherwise} \end{cases} \tag{2.7}$$

In case of $k_{cong} > 0$ this means that there is congestion on the considered lane, therefore it is important to give a different weight on the considered lane advising other vehicles about this situation. This message is exchanged by protocol among neighbors only in order to avoid increasing overhead.

Parameter	description
l_j	Length of the j-th lane of the considered Road (R);
n_j	Number of vehicles on the j-th lane of the road R;
tt_j	Traveling Time along the j-th lane of the considered Road (R);
$SS_{i,j}$	the speed of the i-th vehicle on the j-th lane of the Road(R);
k_{cong}	The congestion factor related to j-th lane of the considered Road (R)
k_{speed}	Coefficient for safety distance computation
δ_{space}	e Coefficient to manage congestion level
α, β	Congestion parameters that are configurable as system parameters
Th_{cong}	is the congestion threshold configurable as system parameters
$h(n_j)$	This terms is function of the number of the vehicles that are on the j-th lane of the Road (R)

TABLE 2.1: Parameter description

2.3 Techniques to transfer contents in vehicular environment

Communication in vehicular environment are very difficult to manage because of the high level of dinamicity and the high speed and mobility of the vehicles. Due to unplanned disconnection events it is very important to transfer information and data as fast as possible taking also into account the consistence of the exchanged data and reliability of the connection. An example of protocol that tries to achieve these objectives is the Fast and Scalable Transmission Control Protocol (FSCTP) [22]. This protocol allows data transfer by using a bidirectional UDP and uses TCP to start and manage the connection. To obtain a faster data transfer more parallel sessions are exploited managing redundancy and resources wastes. In this way it is possible to obtain a cooperative multi-session layer managed in a VANET environment. FSCTP protocol is an high level protocol and for this reason can be considered independent by the lower levels. It is particularly suitable for the VANET environment because it has a small protocol overhead and it ensures that sent packets reach the destination node by making a light control based on the TCP protocol. The UDP protocol instead is fully exploited to transfer all data. This high level protocol is capable of tuning several parameters on the run such as transmission rate using a time windows control mechanism and ensuring the reconstruction of the right sequence using sequence numbers. The security is ensured thanks to encryption of data. All the control operation are performed by the TCP protocol while the transfer of the data is performed by the UDP protocol. The control during the connection is necessary to adapt the transfer to the network condition making it reliable. The

packets are characterized by a header of 96 bits and a 496 bits payload. The header is composed of :

- a 64 bit timestamp;
- a 32 bit sequence index;

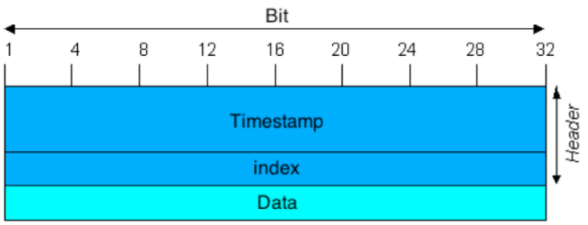

FIGURE 2.4: Protocol Packet

All the data inside the packet are encrypted with Advanced Encryption Standard cryptography the only information that is not encrypted is the time-stamp. At the beginning of a connection between client and server, the client requests the server public key and generate an AES key also called session key. The session key is then encoded with the server's public key by the client and then sent to the server. The server decrypt it with its own private key and answer the client with a message of key provision exchange success. After this phase the transmission can start with an AES encryption. It is important to remember that the first 64bit are not encrypted because they do not contain sensible information. The security of the transmission inside the VANET environment is very important due to the sensible nature of application. One of the application context that can be suitable for this protocol is a client, represented by an OBU request some content that is hosted on a server on a RSU. The client starts the connection with the synchronization phase in which the AES session key is created and shared. After the key exchange, the client selects the service or data of interest from the server that will answer it with the information about streaming size, packet size, number of packet and sending rate on the control TCP layer. The data exchange is then started on the data UDP layer and, during this operation, the monitoring is always running in background through a window mechanism. The time windows here is used to send periodically lost packet list towards the server exploiting the TCP connection, ensuring in this way packet retransmission and avoiding the rebuilding of the packet order. The round trip time is instead used to guess the status of the network and to tune the incoming data rate. The transmission is considered complete when the server receive the completion message that is sent from the client only after all packets are received and checked. An example of the protocol is shown in Fig. 2.5.

During the data transmission through the TCP connection messages are periodically exchanged to tune the transmission rate. When a generic host receives a packet then:

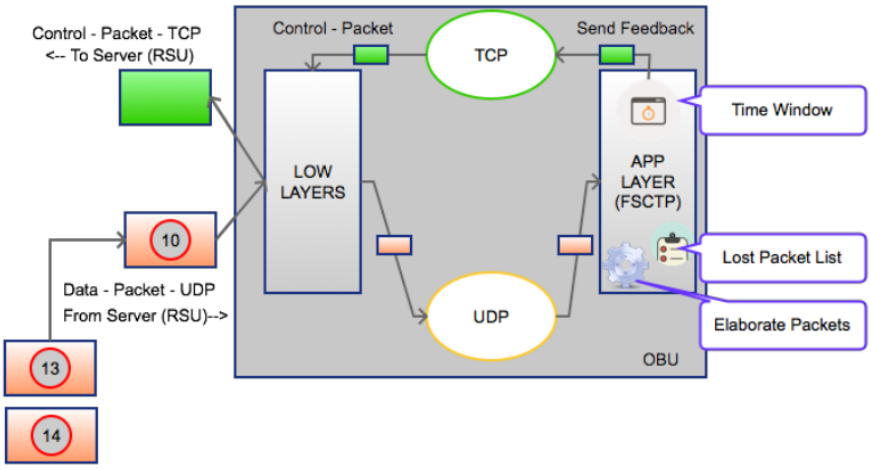

FIGURE 2.5: Protocol Application Scenario

- If the sequence number is greater than the last arrived the packet is marked as the last received and a control is performed to see how many packets are out of sequence. If there are any, the list of lost packed is updated as well as the number of lost packet.
- If the sequence number is lesser than the last arrived the packet is marked as received and the list of missing packet is updated by removing it.

The size of window is fixed and it is chosen at the beginning of transfer.

$$n = \left\lceil \frac{S}{N} \right\rceil \tag{2.8}$$

n is the total number of packet that will be sent during the session and S is the total size in bytes of the data that need to be transferred. The total amount sent bytes are

$$BW_S = N * pckt\_\text{size} \tag{2.9}$$

The window mechanism manages the request of lost packets retransmission. The size of the window is initially set to $MIN\_T + GT$ seconds . MIN_T is the lower bound on window size while GT is the guard time and it is assumed to be $\delta * T$ where $\delta \in]0, 1]$. These contributions are calculated at runtime during the transmission session in this way:

$$T_j = N_j * \frac{\sum_{i=1}^{M_j} \Delta_i}{M_j} \tag{2.10}$$

- M_j is the amount of received packets during the $j-th$ transmission window
- N_j represents the number of packet sent in the $j-th$ window
- δ_i is the end to end delay of the $i-th$ packet of the $j-th$ transmission window.

This mechanism ensures reduction of wasting time and reduce the time needed by the receiver to reorder the packet sequence. When the total amount of lost packets surpass a defined threshold (Maximum Lost Threshold) per window, the transmission rate is update by the client that send it towards the server. It is calculated as:

$$RS_j = (1 - \alpha) \text{ last Rate}_j + \alpha * \text{ newRate }_j \tag{2.11}$$

As said before knowing the number of packets for each time window it is possible to evaluate the new rate as

$$\text{newRate}_j = \frac{N_j^* M_j}{T_j} \tag{2.12}$$

When the amount of lost packets is greater then 10% of the total packets the receiver sends the receiving rate to the sender in order to change it. In this way it is possible to avoid network overloading thanks to a dynamic tune of the rate transmission. When the lost packets percentage becomes too high the transmission rate is reduced to adapt to the network status also because if kept regular many of the packets sent would not be delivered.

2.4 Internet of Vehicles

Until now cars have been seen only like a transport mean to move comfortably from one location to another. With the development of new technologies and the progress of the existing one the way cars are conceived changed dramatically. Mostly thanks to the researches in autonomous driving cars are now equipped with a high number of sensors and and have a remarkable computational power. In the last years we have seen the rise and a very fast development of the Internet of Things (IoT) paradigm. In the IoT paradigm every object is equipped with sensors and is able to communicate thanks to an internet access. In this way data measured from the objects can be shared in the cloud or directly with other smart object. The number of connected objects is very huge, so new protocols have been designed in order to inter-connect a massive number of smart objects. With this perspective vehicles that are equipped with numerous sensors can be seen as *smart vehicles* with a always on connection to internet that gather and share data among them [23]. Transposing the IoT paradigm in vehicular environment a new similar paradigm was created: IoV [24] 2.7. Actually there is no standard protocol that has been adopted from the community for IoT but some of them are quickly becoming a standard *de-facto* thanks to their features. Two of these protocols that are widely used to develop IoT applications and that could be used in vehicular environment are Message Queueing Telemetry and Transport (MQTT) and COnstrained Application Protocol (CoAP). MQTT is a protocol based on the publish subscribe paradigm. Each entity that wants to share information is called a publisher and performs a publish of data on a certain topic. Each entity that wants to receive information about a certain topic is called

subscriber and have to express its interest by subscribing to the topic. In this way subscribers receive only information in which they are interested and a message is sent only when a publisher has data that wants to share with its subscribers. In order to allow this mechanism to work a third entity is necessary to collect messages and forward only to interested subjects. This entity is called broker and keep track of the subscriptions to distribute all the updates that are sent to it. Another interesting feature of this protocol is the possibility to organize the topics in a hierarchical tree that can be addressed by subscriber thanks to special characters [25]. CoAP instead is based on the Hyper Text Transfer Protocol (HTTP) Representational State Transfer (ReST) model but implemented for small devices with very few computational capabilities. A server make resources available under an URL and clients can access these resources with methods analogue to HTTP GET, POST, PUT and DELETE. This protocol have been designed keeping in mind the very poor computational capabilities of IoT devices and can work with 10 KiB of RAM and 100 KiB of code space [26]. While CoAP does not need explicitly a third entity to forward the request to objects it is necessary a method to address directly the device inside the url. The basic paradigms of both protocol are showed in Fig. 2.6.

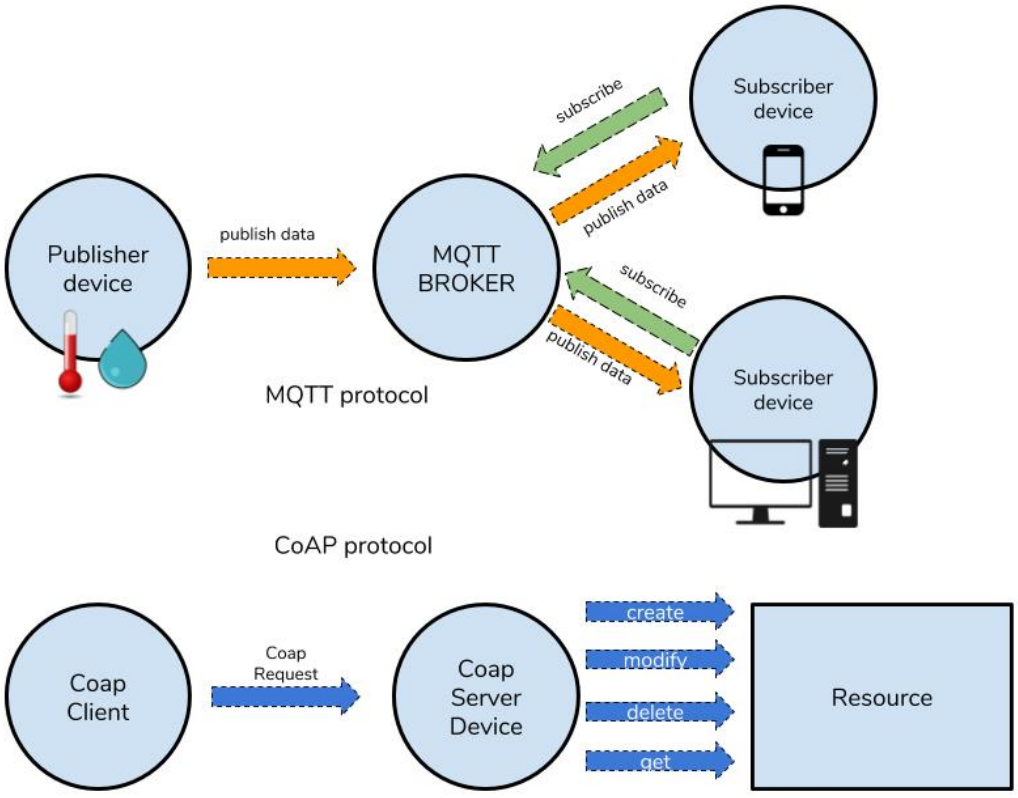

FIGURE 2.6: Behavior of MQTT and CoAP protocols .

With the computation capability of every vehicle and a new network of smart vehicles it is not wrong to think this architecture as a Cloud of Vehicles (CoV). The application range that can be developed on this architecture is similar to the one

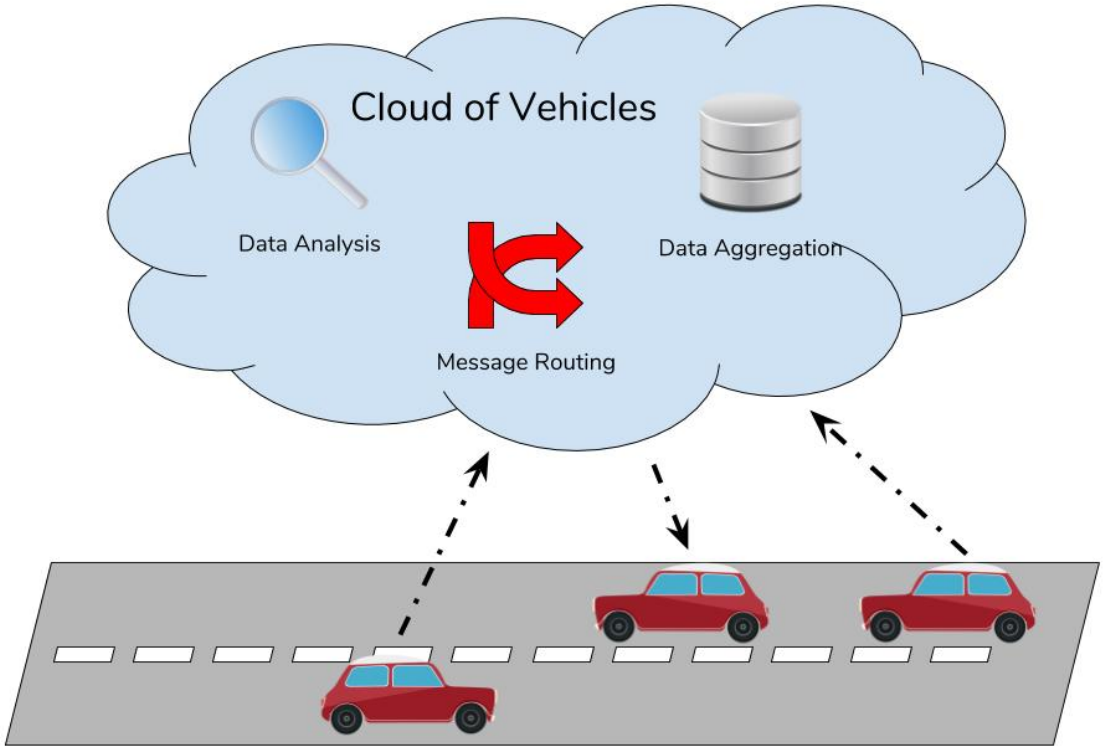

FIGURE 2.7: Vehicles connected to the cloud .

described in the VANET section and range from safety and comfort to entertainment and commercial services.In these applications some notable characteristics and trends can be observed and will be briefly discussed ahead. In vehicular networks some information exchanged among vehicles can have a spatial interest limits. E.g. in safety applications information are shared only in a limited geographic area because only the vehicle that are moving in the same area are interested in their content. When a car perform a strong braking want to share the information about its sudden deceleration only with oncoming vehicles in a 100m radius. This property is also important to maintain the scalability of a global vehicular network in order to keep the information only in the interested area avoiding the flood of not interesting information. Moreover, these networks are *content-centric*. Vehicles do not care which is the source of the information but focus only in the information content. This feature is the key for managing fleet of autonomous car with the platooning technique. In platooning a group of car moves fast along a route with very little distance between vehicles. To maintain this particular formation a constant exchange of messages is necessary in order to support high levels of coordination. An example of platooning application is shown in Fig. 2.8.

Another very strong trend is the exchange among vehicles of the information gathered thanks to equipped on-board sensors. This is a very strong asset for the

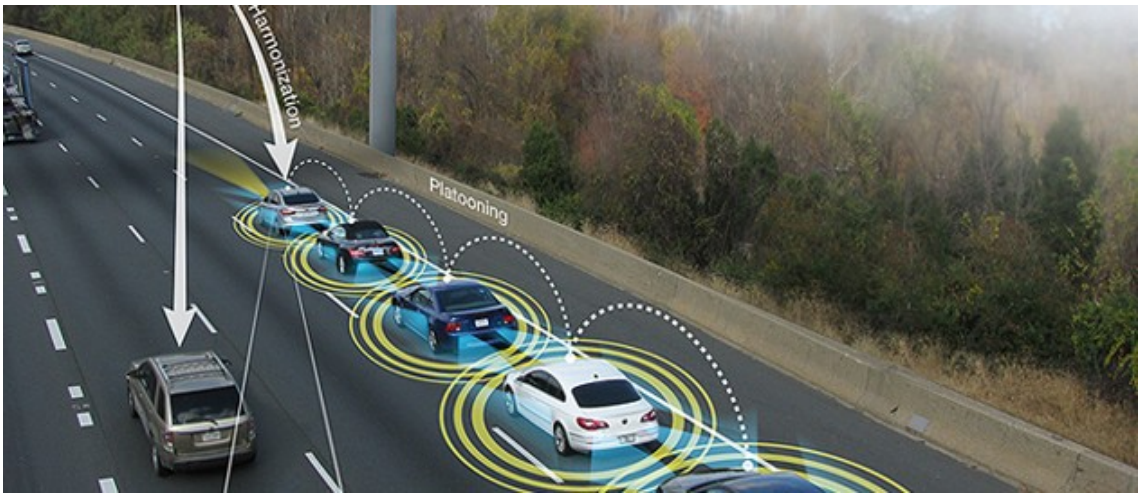

FIGURE 2.8: Cars that travel in formation thanks to platooning.

evolution and concretization of autonomous driving. Until now the research focused on the implementation of a fully autonomous car able to move freely inside an urban environment. While it is necessary that an autonomous vehicle has to move safely in a urban environment even without any connection, it is important to understand that communication among vehicles can improve dramatically the quality of the results already achieved. I.E. the crowdsourcing of information about street status, pavimentation and jam condition could drastically improve the vehicle flow management inside the city. In this way the overall quality of life would raise thanks to the decreasing of the time spent inside the traffic and, consequentially, the decreasing of the emitted CO_2. In the end can be stated that the urban fleet of vehicles is slowly evolving into a collection of sensors platforms that can be used to gather strategic information about the urban status.

2.4.1 Proposal of an IoT communication protocol to enhance energy consumption and safety

Car manufactures are developing vehicles with on-board high-tech solutions to offer assisted traveling features such as for example smart parking and advanced systems for improving onboard safety. Additionally, advanced networking and embedded devices are also used to provide on board services such as social networking, advanced navigation, multimedia contents distribution, assisted driving etc [27]. Gathering information from the environment it is crucial to build an infrastructure that is light, robust, flexible and easy to put into practice and deploy. These features are generally recognized as the main element of IoT devices. These use unique communication protocols because, right now, it does not exist a standard protocol. Nowadays exists several de-facto standards that have become very popular in IoT applications. Some of these protocol have been mentioned before and are:

- CoAP
- European Telecommunications Standards Institute Smart Machine2Machine (ETSISmartM2M)
- MQTT
- Lightweight Machine2Machine (LwM2M)

Connected vehicles are among the possible applications of IoT where every car has the capability to share data with other vehicles and even submit them to the cloud for more technical analysis. The ability of sharing essential data through a web connection morphed the traditional VANET paradigm in to the brand-new IoV paradigm. While in VANET paradigm vehicles usually have the ability to exchange data between themselves with single or multi-hop technique the downside is that it cannot ensure a stable Internet connection. The IoV paradigm rather focuses on the smart integration between human beings and vehicles exploiting an always-on connection with different kinds of technologies. The profilation of drivers could represent a very important opportunity to find new solutions for increasing the quality of life of connected vehicles and several studies have already been done on this argument[28].

2.4.2 Communication Enhancement

Knowing in advance the behavior of a vehicle that travels along the road could allow a better management of the resources inside the city. An example is shown in [29] in which the issue to recognize driver behavior interfacing the On-Board Diagnostic (OBD) system and integrating these data with others coming from an embedded device equipped on the vehicle is investigated. A classifier based on the Fuzzy Logic (FL) has been proposed to realize an Artificial Intelligence (AI) module that recognizes the driver driving style. This recognition does not end in itself but its results are then used to suggest corrective action and to better manage flows inside the city improving road conditions. The corrective actions where simply alert messages send to the vehicle noticing problems during the travel or suggestion for a more safe driving. To allow these useful services communication with a platform called Local Area Services (LASs) has been introduced. This is clearly an example of how is possible integrating IoT protocol inside the vehicular environment and, in other words, an example of IoV application. The designed architecture is shown in Fig. 2.9.

Obviously the management of this huge amount of data must be handled in order to optimize network resources. The IoV layer, the one that is nearer to the vehicle, is composed of the proposed on-board device. This device continuously gather data from the vehicle and the surrounding environment. It is also able to exploit modern car diagnostic through the use of a device able to exchange messages with CAN-Bus protocol. The external connection is obtained through a 4G data connection in order to reach LAS instances. LAS nodes can be considered fog nodes that performs initial computational task in order to manage in a soft real time way

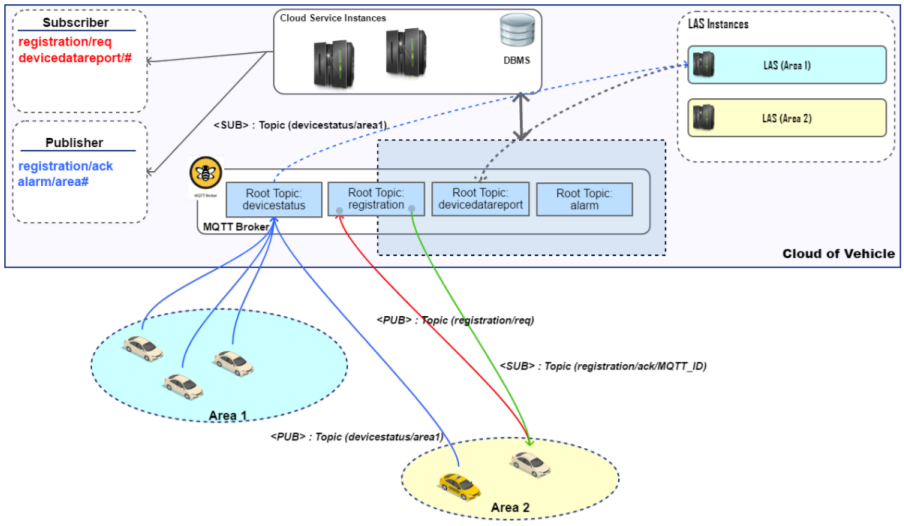

FIGURE 2.9: Distributed Reference Architecture

congestion and flow control in the interested area. Data are also sent from the LAS to the cloud for further analysis. The cloud analysis is responsible of extracting the information about drivers to better fit AI classifier to the driver. If congestion is noticed in a LAS network infrastructure some countermeasures are taken in order to adapt data generation and send rate. The routine installed on the on-board equipped device is simple and can be summarized in the next points:

- get GPS data (latitude, longitude and speed)
- get last MAX values of Acceleration along the axes occurred in the time period
- create an MQTT message and formats the data
- publish the MQTT message
- reset timer

This timed procedure is also triggered if an anomaly in the acceleration along the axes is recognized. This happens because this kind of anomaly can be referred to a sudden brake during travel, to an accident or a sudden change of the vehicle direction. Each vehicle communicate with the LAS service thanks to a 4G connection. The services are accessed thanks to transparent gateway for MQTT-SN [30]. The purpose of these services is to aggregate and pre-process data coming from the device. With the aggregation of these data it is possible to achieve a better training-phase for the AI module speeding up the training and avoiding the waste of computational resources. With different LAS spread on the territory it is possible to obtain a better load balance because each device send data to the nearest LAS. As aforementioned the main task of a LAS is to assist the training of the classifier in the cloud and to take under control area congestion in terms of networking. In order to perform this task the LAS receive a status update from each vehicle an thanks to this Vehicle Status

Update (VSU) also flow and congestion control is performed. To fully exploit the potential of the MQTT protocol a specific organization or the topic hierarchy has been studied. The organization of topics in a topic tree allows a better management of the resources and avoid the introduction of supplementary protocol messages that will only add weight to the protocol overhead. With a tree hierarchy is in fact possible to emulate with a simple trick a unicast connection having a better computational resources management and avoiding the handling of undesired message. The topic tree is shown in Fig. 2.10.

Topic	Pub	Sub
registration		
├─ request	IoT Dev	Cloud
└─ ack		
└─ device_id	Cloud	IoT Dev
devicestatus		
├─ area_n	IoT Dev	LAS
├─ handover		
│ └─ dev_id	LAS	IoT Dev
└─ feedback	LAS	IoT Dev
devicedatareport		Cloud
├─ area_n	LAS	
alarm	Cloud	
└─ area_n		LAS

FIGURE 2.10: Topic tree used in LAS communication

The vehicle registration message can be used by each vehicle to join up the network. When the registration procedure is ended vehicle starts to send VSU messages to the related LAS entity. This improve the stability and the leverage of the LAS computational load. Once cloud elaborates this message it evaluates LAS position and status. Furthermore, it uses position information to evaluate which LAS entity can serve as supervisor. This message is pushed with the registration/request topic to the broker that notify the cloud related service which is currently a subscriber of that topic. The device id is a key field of the registration demand message and it is employed to send out the ack right to the device utilizing the MQTT topic tree. Both LAS and cars receive this message through MQTT services. This analysis is made considering the status of every LAS and vehicle placement. Each device, obviously, is normally a subscriber of the topic registration/ack/own_id where the personal id field is the identification code of these devices. Once elaboration is performed a new message is pushed to the broker with the topic registration/ack/device_id. Vehicle on-board device then begins with vehicle status update routine. As aforementioned before the VSU message is periodically sent from a car to the referenced LAS. the structure of the message is shown in Fig. 2.11.

In the VSU message the following fields are stored:

- MQTT_ID that is the unique identifier of the vehicle;
- TIMESTAMP that is the timestamp of the MQTT message;

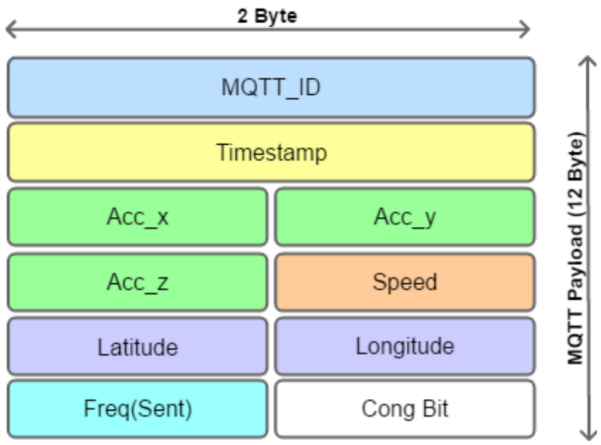

FIGURE 2.11: Vehicle status update message

- USR_ACC_XYZ that is the value of acceleration along the three axes gathered from the accelerometer and gyroscope equipment;
- USR_SPEED is the last speed value retrieved;
- USR_POS is the GPS position in latitude and longitude of the car;
- USR_FREQ is a parameter used for managing the sending rate of data and consequently the congestion of the network;
- USR_CONG_BYTE is a field used to inform LAS of possible congestion;

Thanks to the data acquired from the VSU message the AI module based on fuzzy logic is capable of notifying the vehicle if the current driving style is not adequate or potentially dangerous. To do so a message containing these fields is sent:

- MQTT_ID that is the vehicle unique identifier;
- USR_AVG_SPEED that represents the average vehicle speed in the area covered by the LAS;
- AVG_CONS is the average consumption of the vehicle in the area;
- USR_AVG_BHVR is a field used for noticing the driving behavior adopter by other user in teh area;

The LAS function is also to detect possible dangerous situation in a soft real time way without using cloud computational capabilities which would add a considerable delay. When a problem is detected the LAS can notify it with a alarm notification message. This message is used to notify info about dangerous situation and it contains:

- MQTT_ID;

- TIMESTAMP that is the time signature of the message sent;
- ALERT_TYPE is a field that represent the warning type for example car accidents, queues or anomaly in other vehicle behavior;
- USR_POS the last position detected;

2.4.3 Network Congestion Management

The number of vehicles moving along the street is really huge so some tweaks should be adopted to maintain the network in a good condition avoiding congestion. If the network becomes unreliable due to a wrong usage of the network resources connectivity problems can arise. If the data are not up-to-date and the number of samples lowers there could be classifier errors because of loss of information. When vehicles are involved in communication the data traffic generated is really huge. To deal with this issue a congestion control must be applied. All the message sent through MQTT protocol are set on Quality of Service (QoS) level 1.

In eq.2.13 LAS_j is the $j-th$ LAS and N_l is the number of LAS instances in the system. Let L be the total number of LAS instances in the ITS:

$$L = \cup_{j=1}^{N_l} LAS_j \tag{2.13}$$

In eq.2.14 v_k is the $k-th$ vehicle while N_v is the total number of vehicle.

$$V = \cup_{k=1}^{N_v} v_k \tag{2.14}$$

From the eq.2.13 and eq.2.14 it is possible to define LV_j as the set of vehicles managed by the $j-th$ LAS instance.

$$LV_j \subseteq V \quad s.t. \quad V = \cup_{j=1}^{N_l} LV_j \tag{2.15}$$

Let LV_j and LV_m be two vehicle sets belonging to the $j-th$ and $m-th$ LAS instance respectively. This means that $LV_j \cap LV_m = \emptyset$ with $j \neq m$. Moreover, $f_0^{i,j}$ can be determined as the initial frequency of the $i-th$ vehicle managed by the LAS_j. During the data forwarding of the $i-th$ node, because of the congestion, data may not be received by the CoV.The MQTT-SN version has an associated buffer for each IoT device and starts to collect packets consuming memory resource. When the buffer is full all the further received packets are dropped. Because of this a congestion detection module is present on the device. Let us to define q_k as the data queue related to the $k-th$ IoT device, and its maximum queue size with the q_k^{max} term. The congestion threshold related to the each buffer is defined as α_{th}. It is possible to describe the buffer status ratio as the ratio between q_k and q_k^{max} as shown in eq.2.16:

$$\alpha_k = \frac{q_k}{q_k^{max}} \tag{2.16}$$

If α_k is greater than α_{th} then there is a serious risk of system congestion and the IoT device sends a message with a congestion flag to the LAS which is responsible of the area that covers neighbour vehicles. The Congestion Risk (C_R) is saved in a precise field of the VSU packet. This flag is called congestion bit and it is set to 1 if the congestion risk is detected. Nevertheless, a single C_R received from the LAS might not represent a congestion inside the area because the IoT device could be in a severe status because of various external causes such as channel condition, lack of signal, obstacles and so on. For this reason, LAS_j waits to collect more C_R messages by diverse devices before raising a congestion event management. If this occurs, LAS_j will send to registered devices a new frequency for reducing their reports frequency. This action can reduce congestion level. However, the evaluation of congestion level carried out by a generic LAS is thus described. Let Tw be the observation time window of the LAS_j in the CoV. Let define $r_{i,j}(Tw)$ as a variable counting the number of VSU messages with congestion notification bit equal to 1. This information is coming from the vehicle $i-th$ in the observation window Tw under the coverage area of $j-th$ LAS. Let $H_j(T_w)$ be the amount of vehicle handled by the LAS_j in the observation Tw.

$$CI_j(Tw) = \frac{\sum_{i=1}^{H_j(T_w)} 1|_{r_{i,j}(Tw)\geq 1}}{H_j(T_w)} \tag{2.17}$$

The term CI_j should be $0 \leq CI_j(Tw) \leq 1$; moreover, the higher is the value of $CI_j(Tw)$ and the higher is the congestion level of the LAS_j managed area during the observation window Tw.

Let us define φ as the threshold that advises the congestion level; from this two situation can be stated:

- $CI_j(Tw) < \varphi$, and no congestion is verified in the coverage area;
- $CI_j(Tw) \geq \varphi$, congestion is present in the coverage area;

When congestion is verified, LAS_j predispose a new report frequency to each registered device. The new frequency is computed in accordance with eq.2.18, otherwise the update rate remain unchanged as shown in eq.2.19.

$$f_{i,j}(Tw+1) = \frac{1}{2} \cdot \frac{\sum_{h=1}^{H} f_{i,j}(Tw)}{H} \quad if \quad CI_j(Tw) \geq \varphi \tag{2.18}$$

$$f_{i,j}(Tw+1) = f_{i,j}(Tw) \quad if \quad CI_j(Tw) < \varphi \tag{2.19}$$

All the symbols used into the previous equations are described in Tab. 2.2.

2.4.4 Behavior Classifier

Fuzzy logic can be used to recognize driver behaviors and classify them. The complete process is completed through two sub-procedures that we refer as a two step

TABLE 2.2: Table of Symbols

Symbol	Description
$f_0^{i,j}$	Starting frequency of the vehicle $i-th$ under the LAS $j-th$
L	LAS instances set
V	Vehicles set
N_t	Number of LAS instances
N_v	Number of active vehicles in the system
LV_j	Vehicles served by the $j-th$ LAS instance in the observation window (Tw)
$H_j(Tw)$	Is the number of vehicle served by the LAS $j-th$ in the observation window (Tw)
q_k	The current queue size of the $k-th$ vehicle
q_k^{max}	The maximum queue size of the $k-th$
α_k	Buffer status of the vehicle $k-th$. It is evaluated as the queue size Ratio
α_{th}	queue congestion threshold
$r_{i,j}$	is the congestion report related to the vehicle $i-th$ of the LAS_j
$CI_j(Tw)$	Congestion index measured by the LAS $j-th$ in the observation window Tw

classifier. The initial one is in charge of understanding the context where driver is; The second stage has the primary task to comprehend the behavior of the driver starting with the information of the context received as result of the very first step. In order to collect needed data we need to use several sources. The most important is definitely the vehicle that is capable of supplying several data including speed, acceleration, consumption and so forth. These data can provide us important details about driver practices and harmful behaviors in an indirect way. An Environment Classification represents an essential step since it is strictly related to the driving style. To be able to put into action the classifier we utilized an approach using a FL model. It applies the inference model in fuzzyfication stage and centroid based algorithm in the de-fuzzification process. The Membership Degree (MD) of an object referred to a fuzzy set can assume any value in the range [0,1], unlike a normal set, which is fixed to the values 0 and 1 (false and true): in FL, the MD is usually to be meant as indicating "simply how much" a property holds true. Through some input-output relationships it is possible to approximate any function or system to depict or control [31]. To be able to characterize the various driving styles it is necessary to identify in-advance the surroundings where the driver is travelling. A human being driver instantaneously recognizes the context that surrounds him but, to automate the detection of a likely aggressive driving behavior, it is important to identify the features of the existing environment. To do so the as the primary characteristic parameters to identify an environment the average velocity and accelerations have been chosen. Many observations have already been made

considering data produced by several users and vehicles. In order to collect data an OBD module has been directly interfaced obtaining vehicle dynamics such as for example fuel usage, acceleration/deceleration, torque, etc. Using these info a Probability Density Function (PDF) for urban, suburban and extra-urban environments has been built to be able to automatically recognize the Membership Degree (MD) in each case. The model has been realized through the use of Matlab software program on preacquired data samples. The MD matrix is evaluated to obtain an array of $nx2$ dimensions, where n is the value known as row-count. It is related to the amount of the samples in the observation. In this matrix sampled points meet environments class, which is determined by utilizing a centroid algorithm where the distance between a cluster center and a point is minimized. The relationship between points and clusters center is computed by using an iterative procedure that calculate each point distance from the current centroid. The relationship between these features is shown in Fig. 2.12.

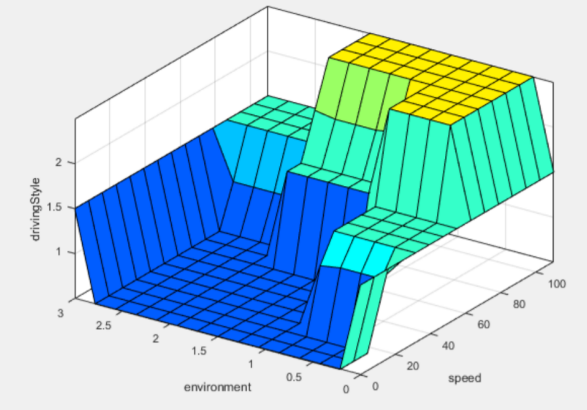

FIGURE 2.12: 3d image representation of driver behavior classifier based on FL approach

2.5 5G

5G is an emerging platform that aims to unify and integrate various connection technologies in order to make possible the developing of applications that need ultra-reliable low-latency connections. This radio access technology purpose is to create an heterogeneous architecture to exploit the advantages all previous communication technologies like 2G, 3G, 4G, WiFi and even satellites. This kind of technology is very interesting because it promises to grant an ubiquitous access to connections with high QoS standards. Moreover, it will not replace existing technologies in which a considerable amount of time and effort have been invested. In this way it will not nullify the return of investments made in the current infrastructure like Long Term Evolution (LTE)[32]. The introduction of 5G in vehicular environment would provide a network platform able to tackle the major issues that the actual vehicular

standards are not able to address significantly. Some of these issues are originated by system requirements of safety and non-safety applications. Some of the requirements involves the communications loads, the congestion control, the resource access and reliability [33]. In vehicular environment safety application rely on the regular and frequent transmission of information beacons to each near vehicle. In order to achieve a better neighbor awareness it would be necessary to maintain an high frequency of transmission. Unfortunately a high number of transmissions increases the number of erroneous transmission end errors. To manage this assure some adaptive beaconing approaches have been proposed but the current status of IEEE802.11p cannot guarantee the quality of the neighbor awareness due to the shared wired spectrum. The major portion of the wireless spectrum is used by periodic beacons and this causes a quickly saturation of the channel capacity. In vehicular networks several techniques to cope with channel congestion have been developed but none of these have proven to be very effective in tackling the problem. Unfortunately in this network environment a fair access to resources cannot be guaranteed. Within these dynamic networks topology changes continuously and it is almost impossible to guarantee a fair access policy to the shared resources. Due to all the motivations previously discussed it is really difficult to manage a vehicular network ensuring a minimum level of reliability. This scenario cannot be tolerated in safety application in which a missed beacon receipt or even a delay can make the difference in dangerous situations. One of the most interesting features that could face this problem is the Proximity Service (ProSe) [34]. ProSe allows devices within communication range to exchange data directly without the necessity to access cellular networks. This technology add to the preexisting well known up-link and down-link of cellular network the concept of side-link. It uses the same transmission schema of the LTE technology but allows spontaneous and simplified beaconing exchange in a predetermined range. Side-link communication doesn't need any network infrastructure. In this way it is possible to avoid congestion by using side-link communication when in range and up-link/down-link communication otherwise. In this way it is possible to better manage the spectrum by distributing beacon transmission between these two working modes lowering the congestion caused by the neighbor communication. This technology is very relevant especially for safety application that have a proximity nature [35]. An example of a 5G architecture is shown in Fig. 2.13

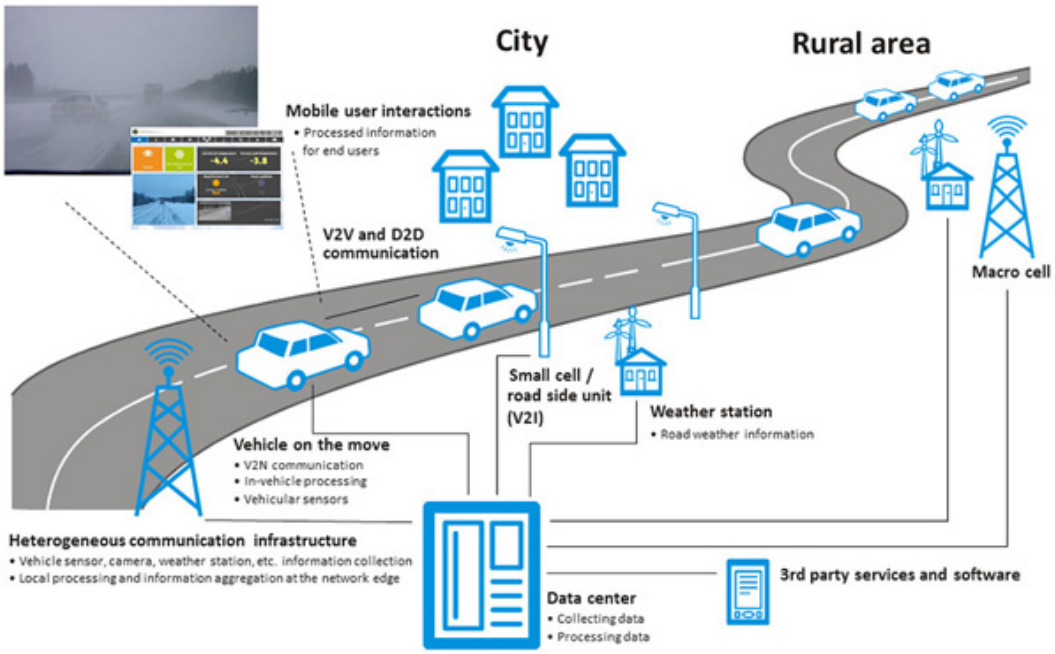

FIGURE 2.13: An example of 5g architecture.

Chapter 3

Autonomous Vehicles Architecture

3.1 Introduction

In this chapter we will analyze the architecture of an autonomous vehicles with the focus on equipped on-board sensors and on the computing unit. All data received from the sensors need to be analyzed in a real time way. All the information received from the sensors are used to build a model of the environment in which the vehicle is moving. Thanks to this model a planning phase starts in which the vehicle have to choose the path to reach its destination avoiding incoming obstacles. Path planning is a continuous routine that update the behavior of the vehicle on the basis of the information received by sensors in real time. This routine is very expensive in terms of computation capabilities and for this reason a powerful computational core is needed to perform this task. Relevant computation capabilities are also necessary to process data from the equipped on-board sensors. One of the most computational expensive task is to process data from a stream video of a camera. Computer vision analysis techniques and algorithms are really computational-demanding due to multimedia nature of the data that have to be analyzed. Computer vision algorithm have to detect and track objects in a real time way continuously analyzing the incoming video stream. Another high computing-demanding task is the analysis of information coming from Light Detection and Ranging (Lidar). Lidar technology deals with light as a mean to scan the surrounding area. While the use of light allow a really fast detection of obstacles it is really difficult to cope with its speed. For this reason very expensive technology is necessary to receive these information and a high computational power to analyze and compute the data [36], [37].

In this section all the sensors equipped on an autonomous vehicles will be deeply described along with the technology used. As previously stated data gathered from on-board equipped sensors need to be integrated in order to build a model of the surrounding environment. Each sensors have its pros and cons and that is why multiple type of sensors are needed. The sensors types that will be discussed are:

- Cameras
- Radar
- Lidar

- Proximity sensors
- GPS

3.2 Cameras

Cameras are one of the most important and necessary device that are needed on autonomous vehicles. While other sensors can detect obstacles and pedestrians on the road only through camera is possible, at least for now, to identify road signals and horizontal signage. This is mandatory because the first autonomous vehicles will move inside an urban environment designed for human driving and have to understand signals made to be interpreted through human sight. The number of equipped camera on-board can vary in relation to the manufacturer policy and to the presence of other kind of sensors. I.e. Tesla cars are equipped with 8 cameras in order to have a 360 degree view of the surrounding environment. Cameras are able to recognize lane markings and road signs and have the ability to see colors. This feature is very important during the driving task because colors have different meanings related to the sign or light. Moreover, in relation to the cost of other sensors the very small fabrication cost is considered one of the key factors for the wide spreading adoption of autonomous vehicles. One of the most challenging questions about camera sensors is their capability of standing alone without the support of more complex and pricey sensors. Cameras are well known for the poor performance when the visibility conditions are not good just like the human eyes. The major part of the autonomous vehicles manufacturers decide to support cameras with Lidar or Radio Detection and Ranging (Radar). Tesla manufacturers opted for the less expensive radar on the front while almost all the others think that , at least for the moment, is not possible to remove the Lidar technology on these vehicles. An israeli company called MobileEye that works on autonomous vehicles states that if a human is capable of driving only using vision also a computer can. They firlmy believe that cameras can be the sole source of vision and that is enough to perform the driving task [38].

3.2.1 Computer vision

Computer vision is an interdisciplinary field of study that deals with techniques and algorithms to make computer detect, track and recognize objects in static images or videos [39], [40]. Its purpose is to extract information from images similar to the information extraction performed by humans through sight. In vehicular environment these techniques are mostly used to detect obstacles and other vehicles, to recognize them, track them if they're moving and then predict their behavior. In this way an autonomous vehicles can modify its movement plan according to the rapidly changes inside the environment. In Fig. 3.1 it is shown the ascending levels of vision for semantic interpretation. At the lowest levels there is vehicle detection

that is based on appearance , motion and shape. The middle level concern the tracking of moving vehicles and operations of position analysis, possible dynamics and filtering are performed. The top level performs the semantic interpretation of the lower data levels in order to predict the behavior and the movement of the objects, which are their goals and the maneuver to achieve them.

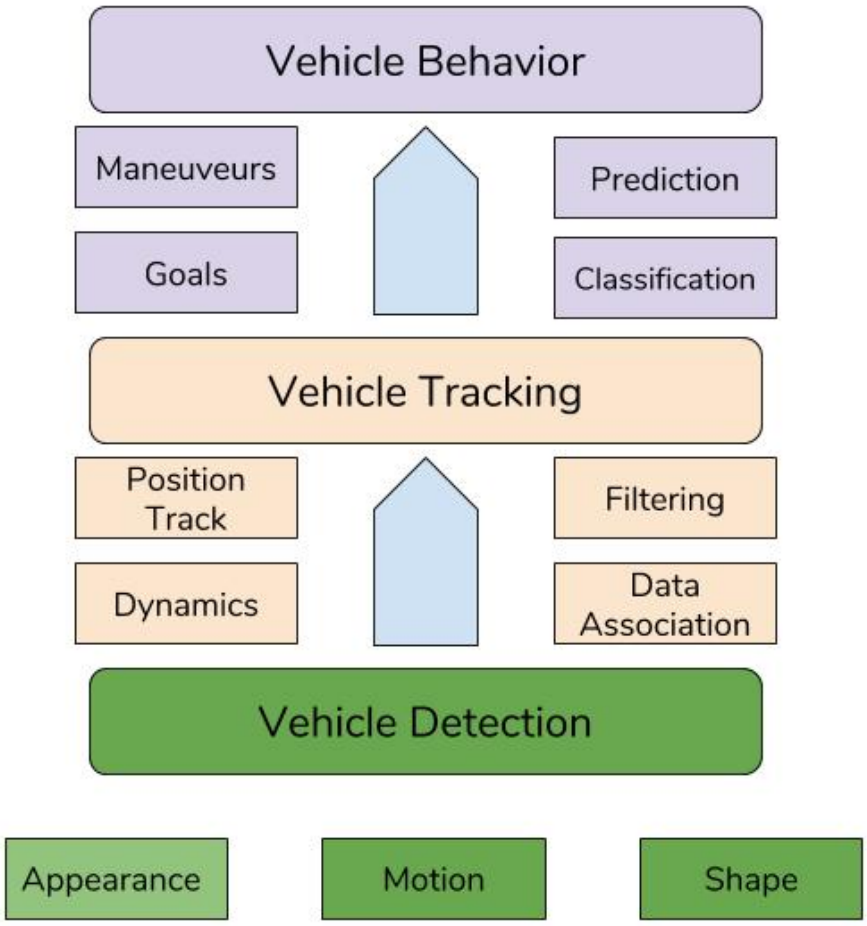

FIGURE 3.1: Computer vision task to detect and recognize vehicles

The task of vehicle detection unfortunately presents many difficult problems. The road scenario is semi-structured and gives a low number of landmarks allowing only weak assumptions. Moreover, also the movements at relatively high speed introduce effects of ego and relative motion. Each object encountered on the road have, from a sight perspective, shape, colours and size very variable. We also have to consider that other object are moving inside the urban environment and the presence of large obstacle that can partially or totally clutter the vision area. There exists two architectural approaches that can be used for performing these tasks:

- Monocular vehicle detection
- Stereo Vision vehicle detection

When using monocular cameras it is furthermore possible to divide detection of vehicle in two broad categories: appearance-based and motion-based. In appearance-base methods the detection of vehicles is performed directly on images thanks to pixel recognition. On the contrast in motion based approach a sequence of images is necessary. This sequence is necessary because this task is performed through a sequential scene comparison in which moving pixels are highlighted. Monocular cameras are not very good in perceiving the depth of images. Even if exist some techniques that are able to approximately calculate the distance of objects from the camera point of view for this task a stereo vision architecture is preferred.On the contrary in stereo-vision motion based approaches are preferred. A stereo architecture can easily detect depth and 3D features of objects. In this way motion of objects is perceived by the camera in a more natural way enabling a painless motion-based approach to detect vehicles. A multiview geometry ease the direct measurement of 3-D information. A more difficult task is the analysis of detected vehicles movement for a semantic interpretation. The semantic interpretation consists in characterizing the behavior [41] of other vehicles moving along roads. A robust vehicle detection and tracking are fundamental prerequisites to perform a good behavior analysis. The key parameters to perform the behavior recognition are vehicle's position and vehicle's velocity as time dependant parameters. Using this spatial and temporal information derived from vehicle's detection and tracking a precise analysis is possible to identify probable maneuvers that other vehicles are going to perform or are undergoing. The semantic evaluation of these data is not valid only to recognize the maneuvers but also the human driver behavior. Several studies have been done to categorize human drive style using various selection feature [42]. Some of them divide the behaviors only in two categories : normal or abnormal. Others instead offers a more precise categorization like below normal, normal , aggressive and very aggressive according to information like speed and direction change. Knowing the behavior of a vehicle it is necessary to decide the short term movement of the autonomous vehicle. This classification is done by considering four main criteria . The first one is the road context in which the vehicle is moving. To give a semantic interpretation to a vehicle behavior is important to know the road context. Velocity and trajectory evaluations vary depending on the kind of street. I.e. a speed of 90 km/h can be considered aggressive in a urban environment but normal on a highway. The second aspect that have to be taken in consideration is the maneuver that the detected and tracked vehicle are going to perform. The most forecasted maneuvers are turning, lane change , overtaking and undertaking. Third the analysis of the trajectories as long-term sequence of position and velocities is performed in order to depict future trajectories and eventual collisions. At last the real classification is computed and the movement planning phase is done.Obviously the more the prediction is in the very next future the more is accurate. A summary schema for behavior classification is shown in Fig. 3.2.

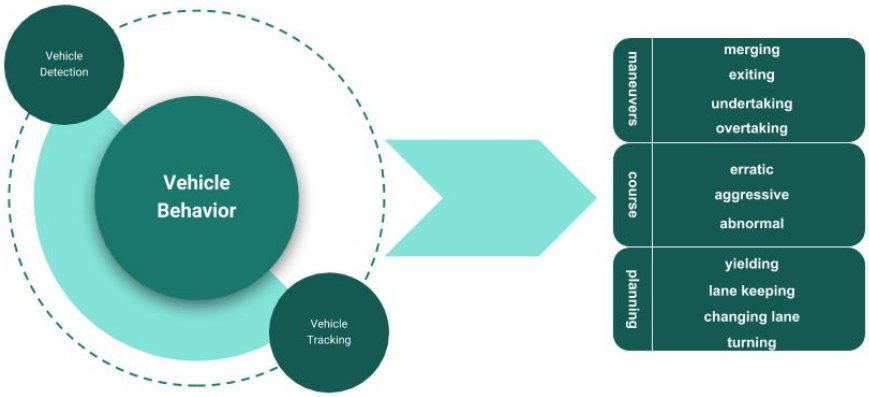

FIGURE 3.2: vehicle behavior classification

3.3 Radar

The working principle of radar is really easy to understand. Some electromagnetic waves are emitted from an antenna. These waves are reflected when hit obstacles with an electrically leading surface. If the emitted waves are then received again they reveal an obstacle in the emission propagation direction. Electromagnetic waves travel at constant speed through air that is approximately the speed of light. Thanks to this characteristic it is very simple to calculate the distance from the obstacle only knowing the round trip time of the waves. Normally waves travel in straight line all around the emitting point but by using special radar antennas it's possible to focus their emission in a specific direction. In this way also equipment precision is enhanced and it is possible to calculate even direction and height of the detected object. There are many advantages in using Radar technologies. This technology is able to operate day or night because is not influenced by the amount of light in the environment and is able to operate in all weather conditions penetrating even walls or layers of snow. Its coverage is really broad and can detects and tracks moving object [43]. Most expensive models can even detect the shape of objects with enough precision to allow a obstacle classification and recognition. Automotive radar works on the same basis principle trying to maintain a good trade off between costs and precision. It's use in current automotive scenario is very spread and is equipped on passenger cars, trucks , buses and even the smallest sedan. Highly automated driving pushed further the adoption of this technology that started a very fast evolution in terms of equipment dimensions and cost. DAIMLER with other car manufacturer started to integrate the radar technology to enhance existing safety mechanism or to offer new solutions. Some of the offered solutions involve blind spot detection, cruise control with stop and go and 360 degree pre-crash sensing with pre-triggering of air-bags. The introduction of autonomous emergency-braking and pre-crash system has been possible thanks to the adoption of multimodality. Multimodal radars are capable of varying their range modifying the wavelength and form emitted. In

this way it is possible to detect an obstacle with a large range extent and then progressively switch to a narrow range extent to extract obstacle features like shape and dimensions in a really accurate manner. The first field of use of this technology was the emergency braking system. Mercedes-Benz adopted in 2013 the PRE-SAFE®[44]. With this technology the car was able to activate braking 600ms before the unavoidable collision with the front car. Some time later the system was extended with pedestrian detection. Dislocating radar sensors not only on the front but also on the back and on the side granted the possibility to develop other kind of application that not necessarily are related to safety. The use of radar is widely spread in parking application. There are several application in which the vehicle helps the driver during the search of a suitable parking and even start an automated parking routine at the press of a button. The use of an improved version of these technology is currently used in the development of fully autonomous vehicles that, in the near future, will be able to move not only in highly structured environment like highways but also in semi-structured of low-structured areas of urban environment. The main issue to face is the impossibility to segment the driving task in a sequence of finite maneuvers as for simple task like parking. The driving task is, as stated before, a very complex and dynamic task that needs to be handled in a real time way. Another very important issue to face in the near future is that with the adoption of radar technology on large scale will give birth to several problems regarding interference between these devices. In Fig. 3.3 the position of radar sensors is shown with corresponding possible applications.

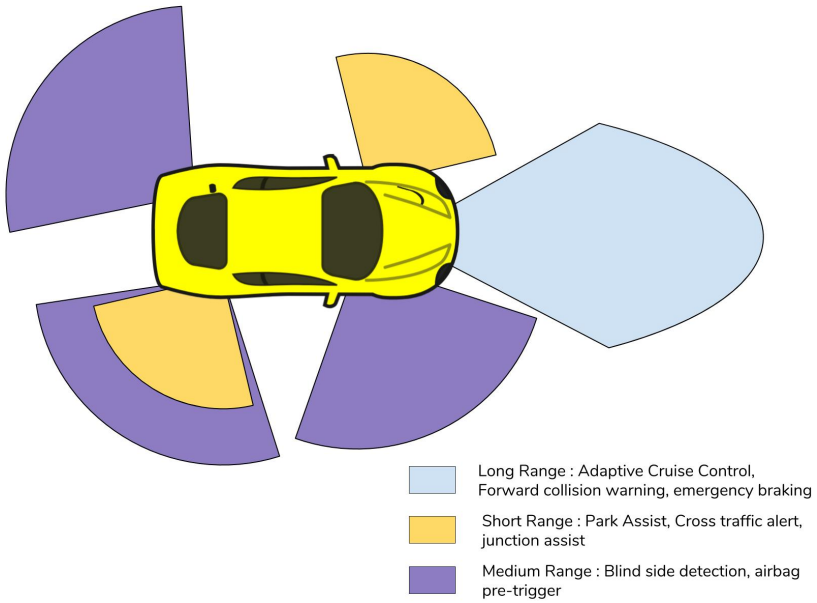

FIGURE 3.3: Position of radars on autonomous vehicles and applications

3.3.1 Radar Perception Techniques and Tweaks

Thanks to multimodal functionality it is possible to generate multiple detection for each target. All these generated data are very difficult to be interpreted. For this reason the semantic interpretation of the data is done with deep learning techniques like neural networks. In this way data detected from multiple scans can be clustered and analyzed to extract shape, dimension and other important features. To have a better description of the surrounding environment canned area are structured in radar-grids. With these grids space is divided in cells that can be classified as occupied or free to provide a more suitable schema for vehicle path planning. When the current planning phase is done the vehicle has to move according to the planned path using data received from radar and radar-grids to estimate if the ego-movement is going as planned and eventually take countermeasures for an incorrect followed trajectory. During the movement actuation data gathered from radars are continuously analyzed because of the dynamic environment. Signals from an azimuthal Doppler sensors can even detect if an obstacle is a vehicle or not. It has been discovered that a praticular doppler signal signature is associate to a moving wheel. Knowing the position of the wheels and their direction it is possible to calculate the possible trajectory of the scanned vehicle. It should be remembered that an autonomous vehicle receive data also from other sensors. All the technique described before can be greatly improved with a sensor-fusion approach in which data gathered from radars are integrated with others coming from other sensors [45], [46].

3.3.2 Sonar

Sonar technology is not taken in high consideration when talking about autonomous vehicles. Sonar working principle is the same as radar but use sonic waves instead of electromagnetic waves. sonic waves are very slower compared to other kind of waves and for this reason components are very cheap. Sonic sensors are still used in automotive environment for park assistance applications. They are positioned on bumpers and can detect only if an obstacle is really close to the maneuvering car. They don't start to activate until the car is only few meters away so there's no interference at all. Usually they have two or three discrete levels of closeness and warn the driver with the frequency beeping.

3.4 Lidar

The working principle of Lidars is the same of radars and sonars but instead of using electromagnetic waves or sonic waves it uses light. This technology was firstly developed in the early 1960s to detect submarines from aircraft. A Lidar generate a laser pulse train that hit obstacles in its path and reflect a certain amount of it to the source. The difference in laser return times and wavelengths is computed and is used to create a 3-dimensional model of the hit object. One of the characteristics that

can be considered a great improvement respect to radar technology is the fact that lidar can detect even non metallic objects. Lidar technology is also much more accurate because the train pulse can vary from hundred to hundred of thousand pulse giving in this way a detailed representation of even very small objects [47]. Obviously the equipment needed that can manage multiple detection at the speed of light must be very complex and only in the last years we achieved a technology level to give this device an enough small size to fit cars. Generally there are two types of detection methods for Lidar technology : coherent and incoherent. Coherent detection use Optical heterodyne detection reducing in this way the energy needed to operate but with the expense of much more complex transreceivers. They both use two main pulse models : micropulse and high-energy systems. Micropulse pulse model has been developed thanks to the new computing capabilities of modern computer architectures and is used in low powered lasers that do not hurt human eyes. High energy systems instead demand more energy to work and are used usually for the measurement of atmospheric parameters such as height, layering and density of clouds. With an enough accurate sensor equipment it is also possible to acquire pressure, temperature , wind, humidity and gas concentration [48].

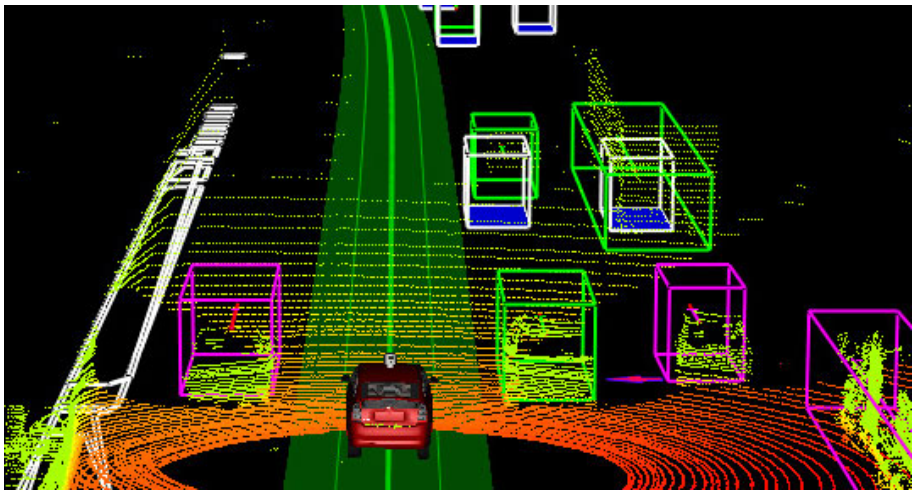

FIGURE 3.4: Obstacle detected insite the urban environment through lidar

3.4.1 Components

Now the components that are necessary to Lidar to work will be described

- Laser
  For most common non specific application laser of 600-1000 are used to ensure sight-safety for humans. Laser frequency is heavily influenced by the application scope. I.e. in airborne sans lasers of 1064 nm are usually used while for undersea scanning 532 nm are preferred because penetrates water with less attenuation. Laser characteristics include also laser repetition rate, pulse length,

pass through gain material and q-switch. In general the better the resolution is the shorter the pulse must be. Flash lidars use a single source of light that illuminate the surroundings. However the range of detection is deeply influenced by the light burst provided. To do not harm human eyes the wavelengths must be in the eye-safe spectrum and this entail very expensive imagers. A technique that allow a 360 degree representation of the surrounding environment without using Flash lidars is to attach to the laser Micro Elctromechanical Mirrors (MEMS). In this way the laser beam is directed to a single mirror that can be reoriented to any area of the target field. On autonomous vehicles MEMS are always spinning on the top to have a complete view of the environment. However MEMS operate on a single plane and to add a dimension another mirror is required that rotates along the interested direction. Unfortunately these mirrors are very sensible to shocks and vibrations and for this reason a frequent calibration is required.

- Scanner and optics
  The speed necessary to build an image of the surrounding area depends on the scanning speed of the laser beam. Usually for the scanning procedure are used different configuration of mirrors for example: dual oscillating plane mirrors or polygon mirrors with a dual axis scanner. The optic chosen deeply influence the resolution of the create image and the range objects can be detected. The returning signal is collected thanks to a hole mirror or a beam splitter.

- Receiver Electronics
  The returning beam receiver is composed of light or other electromagnetic waves sensors made in silicon avalanche photodiodes or photomultipliers. Their function is to convert light photons into current allowing a digitalization of the objects hit by the laser beam. The sensitivity of these sensors have to be tuned with other components of the Lidar.

- Position Systems
  Lidars mounted on moving devices or vehicles need to know in a very precise way their absolute position. This is necessary for the orientation of the sensors and to combine the data received on the surrounding environment with their relative position in relation to the Lidar equipment. To improve the information about position also an Inertial Measurement Unit (IMU) is needed. Thanks to data received from gyroscopes, accelerometers and electromagnetic field information on the absolute position can improve to reach a precision of few centimeters.

The use or Lidar technology in combination with all other sensors really helps improving the knowledge about the surrounding environment of an autonomous vehicle. One of the problems for the adoption of these technology is currently the very high cost of the equipment. Currently a Lidar sensor can reach the price of several

tens of thousands of euros. To avoid an unaffordable price for autonomous vehicles on the early market some companies are trying to develop very low cost Lidar sensors under 1k euros. The major part of the manufacturer believes that Lidar technology is an irreplaceable key feature but some believe that a further progression in computer vision could lead to a safe autonomous driving without the use of Lidars [49].

3.4.2 Road Recognition

For autonomous vehicle the detection of road limits and curbs is a fundamental task to allow a safe trip in urban environment. In [50] a curb detection technique is proposed using a 3D-Lidar. Usually, in a urban scenario, road boundary are delimited by the position of curbs on both sides of the road. While detecting the curb is not so complicated, problems arise in presence of intersection where the curbs are discontinued. Several solution have been proposed in literature based also on computer vision techniques applied on the video streams of multiple cameras. A possible solution is to use the HDL-32E *Velodine* Lidar sensor that features up 32 lasers aligned from +10 degrees to -30 degrees and is capable of generating a cloud points of 700.000 samples with an accuracy of 2cm. This kind of sensors is usually mounted on the top of the car so a transformation of the coordinate from polar to car is necessary as a pre-elaboration for the road recognition. When the vehicle is moving it slightly change its angle compared to the ground so a inertial system is necessary to recalculate the correct angle for the transform. The road area contains curbs, sidewalks and the road surface while the off-road area contains trees, building and other tall objects. Through filtering is possible to distinguish between road and off-road area in a very simple manner. The street baseplane is defined by a linear equation and a least-squared method is applied to plane fitting. All the points with negative angle are selected and random sample-consensus method is applied. The extraction of the on-road area it is not accurate because the latter phase consists in the application of curb detection. Curbs are detected through a sliding beam method with a search based method in each frame for the detection inside the points cloud. It consists in a sequence of beams originated from a source point evenly spaced to detect intersection discontinuity. The issue with this technique is that is really difficult finding a fixed distance that is robust for various kind of intersection with different gaps. The proposed sliding beam model overcome the problems by not relying on a fixed distance and consists in two beam layer. The bottom layer beam model determine the road segmentation and the angles represents the road direction compared to the vehicle current position. The top beam layer instead takes care of segmenting the road ahead of the vehicle. Thanks to this scanned layer is possible to recognize the road shape ahead and to detect intersections. An example of the application of this method can be seen in Fig. 3.5.

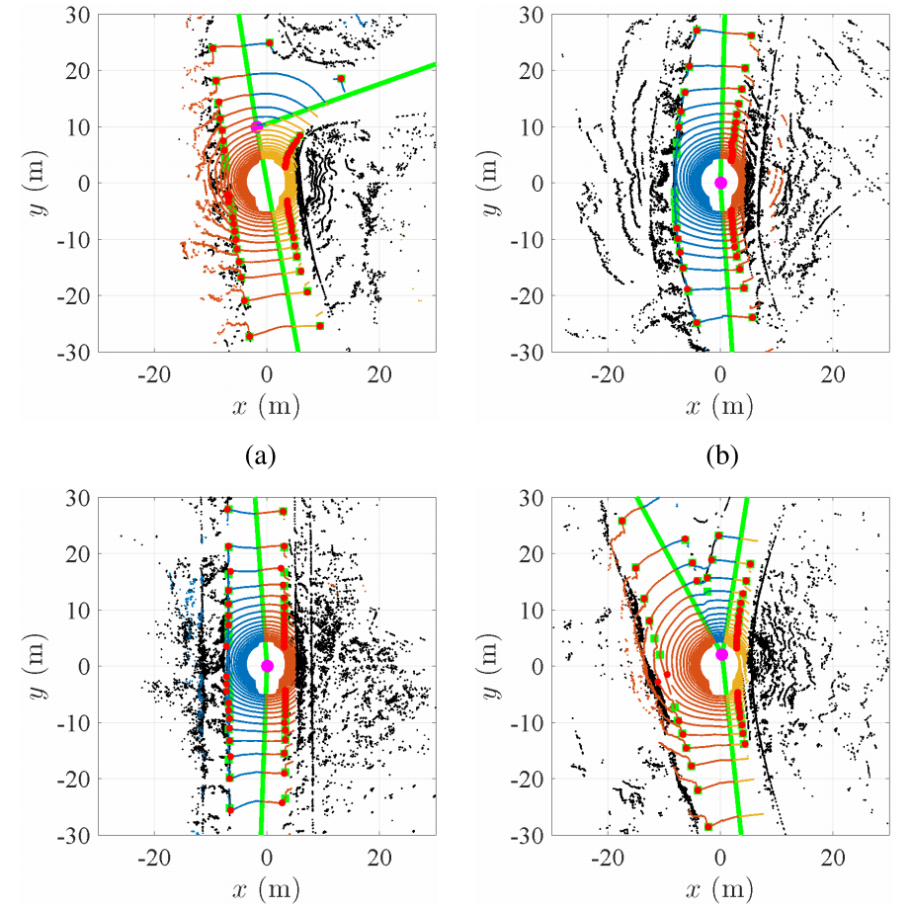

FIGURE 3.5: Curb Detection with sliding beam technique

3.5 Global Positioning System

Autonomous vehicles needs a very accurate knowledge of their absolute position inside the street [51]. For this reason a High Precision Global Navigation Satellite System (GNSS) technology is needed to ensure a vehicle stays in lane. With this technology an accuracy a decimetre-level precision is ensured. For a such high level of accuracy GNSS use special receivers that use multiple frequencies with Synchronous Position Attitude and Navigation (SPAN) and GPS anti-jam technology . GPS is a satellite navigation system that can give to users the location and time information in all weathers conditions. It provides with a 3-dimensional positioning and navigation with timing to all user around the globe. The GPS system consists of three segments :

- GPS satellites
- the earth control system
- GPS user equipment

The space segment is composed of the constellation of satellites circling the earth every 12 hours. These satellites receive the signal from the earth control system , store

it and then retransmit the route/navigation sent from earth. The number and disposition of the satellites ensure that almost four satellite are available simultaneously from any point on the earth surface. The earth control segment is composed of five monitor stations with atomic clocks spread around the globe. These monitor stations monitor the satellite status and, when abnormalities in the signal are detected, signal is revised and sent back to the GPS through ground antennas. In the user equipment segment we find all the GPS receiver

3.5.1 Technology

The technology behind this level of precision uses two or three different frequencies that are broadcasted by different position satellite costellations i.e. GPS, GLONASS, Beidou and Galileo. Thanks to multi-frequency receivers it is possible the mitigation of errors caused by atmospheric conditions and signal delays. Obviously the more constellations are used the more the system is able to compute a precise position specially in urban environment. Classic GPS equipment without correction has an accuracy of five to 10 meters. Using multi frequency antennas and combining the information received from different constellations it is possible to improve the position quality. Many algorithms can compare data received from more constellations and make a good estimation of the error to correct the position to an accuracy of few centimeters. To enable extreme precision systems with on-earth devices have been developed. With Real Time Kinematic (RTK) the on board receiver equipped on vehicles listen also for on earth signals. The on ground antennas have a range of 40 kilometers. In this way errors in satellite clocks and satellite orbits are removed and atmospheric effects on signal propagation are mitigated. Another technique that use on earth transmitter is Precise Point Positioning (PPP). With PPP a global workd-wide network of references are available to all vehicles and are transmitted via satellite or cellular networks. The third segment is the user devices segment. Each user equipment receive the GPS satellite signal to determine the position on the earth surface. The working of Global positioning system is based on the trilateration mathematical principle. The position is determined from the distance measurement to satellites.Three satellites are used to determine the exact position on earth surface while a forth signal is necessary to validate the data. With one satellite a large number of points equidistant from the satellite are the candidate for possible position of the receiver. With two satellites the candidate positions for user equipment becomes only two. A third satellite signal is necessary to determine only one point. This process is shown in Fig. 3.6.

3.6 Planning and Decision Algorithms

The planning problems in AI is about making the right decision when trying to achieve some goal. The act of pursuing this goal can be easily translated in a sequence of action and each one bring closer the achievement of the final goal. Each

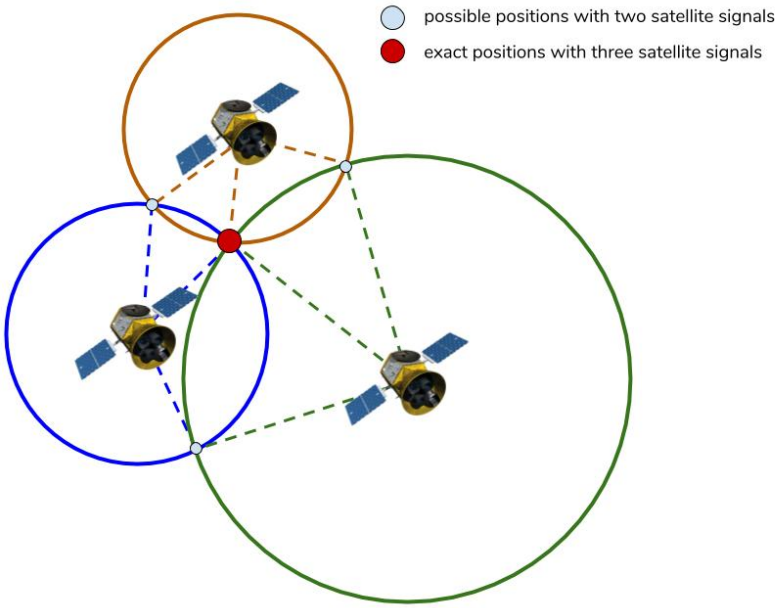

FIGURE 3.6: Trilateration process to determine user GPS receiver position

step transform the state of the world so it will satisfy the predefined goal. The world is considered as composed of atomic facts or state variables and each action can make some facts true and other false. The simplest scenario can be considered the one in which a single agent in a deterministic environment can be fully observed. In these conditions a plan is only a sequence of primitive actions that can be considered discrete. The evolution of the world is searched by the planner that evaluate each possible evolution of the world navigating a graph representation of states of the world. The difficulty of the planning task is strongly dependent on the simplifying assumptions employed. The assumptions that determine the difficulty of the problem involves:

- Deterministic or nondeterministic actions
- state variables discrete or continuous
- full observable or partial observable states
- number of possible initial state
- duration of actions

The Classical Planning Problem that is the simplest possible planning problem is determined by a unique known initial state, durationless actions, deterministic actions on at time, single agent performing the task. If the planner has as input the set of possible action which can be performed in the domain as well as the initial state the planning task is considered domain dependant. Otherwise the planning task

is called domain independent and can be applied to different domains. The most used programming languages to describe planning domains and specific problems are based on state variables and are Standford Research Institute Problem Solver (STRIPS) and Planning Domani Definition Language (PDDL). STRIPS is composed of:

- An initial state
- specification of the goals
- a set of action.

Each action has preconditions and post-conditions which represent what must be established before and after the action. PDDL is an attempt to standardize AI planning languages. There are several de-facto version of PDDL but the community developed also other versions that are not officially recognized 3.7.

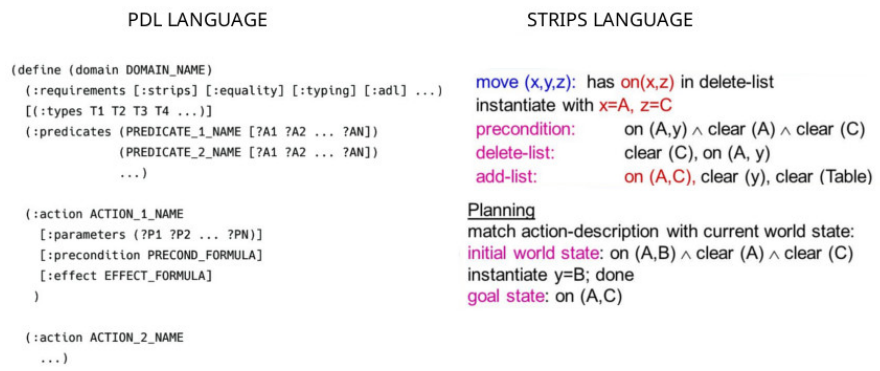

FIGURE 3.7: Languages used for planning world and problems

The task for autonomous vehicles of reaching a destination from a starting point can be considered a very difficult planning problem in which the state of the world is uncertain and always evolving due to external causes. This kind of planning problem cannot be treated with classic approaches and must be decomposed in subproblems that are treated in a very small amount of time (also called reaction time).

An autonomous vehicle must sense and interact the surrounding environment during all the duration of the driving task. To do so it has to perform a routine that is very similar to Self Adaptive System (SAS). A SAS monitor and analyze the surrounding environment and then make a decision to achieve its goals. This system has a self-knowledge allowing a reconfiguration even real time to better satisfy the requirements. To better explain this routine we can refer to the Monitor, Analyze, Plan, Execute and Knowledge (MAPE-K) routine introduced by IBM in 2006 [52]. This whitepaper introduced the IBM vision of autonomic computing as the ability to manage itself and dinamically adapt to change in accordance with business policies

and objectives. It consists of 4 sequential phases and and a one that contains the previous four and it is shown in 3.8.

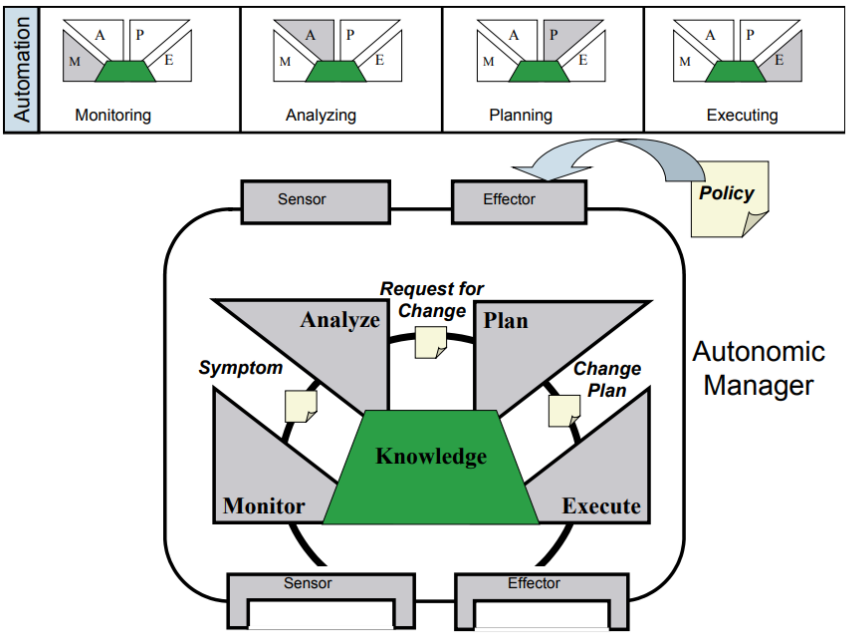

FIGURE 3.8: MAPE-k automatic Manager

The architecture loop is divided into four parts that share the one that contains all of them : the knowledge.

- The monitor function provides all the mechanisms that are necessary to gather, aggregate and filter data and information from the managed resources
- The analyze part provides all the tools necessary to model complex situations. Taking as input the data elaborated from the monitor function it is capable of discovery patterns to predict future situations
- After having all the possible information about future trends it is possible to plan future actions to achieve goals and objectives. The predictive models that are generated from the second phase are strictly necessary for the planning stage because, even if short-termed and not certain it greatly improve the results.
- After the planning phase there is the actuation phase called execute. The execute function provide the mechanisms that control the execution of a plan elaborated from the previous phase. Obviously the execution of the plan must take into consideration dynamic updates.

Each phase or function is connected with the knowledge source. A knowledge source is an implementation of a registry, dictionary or database that provide access to the data through specific and well known interfaces. This architecture loop

applied to AV is slightly different but it is composed of the same key items and is shown in Fig. 3.9.

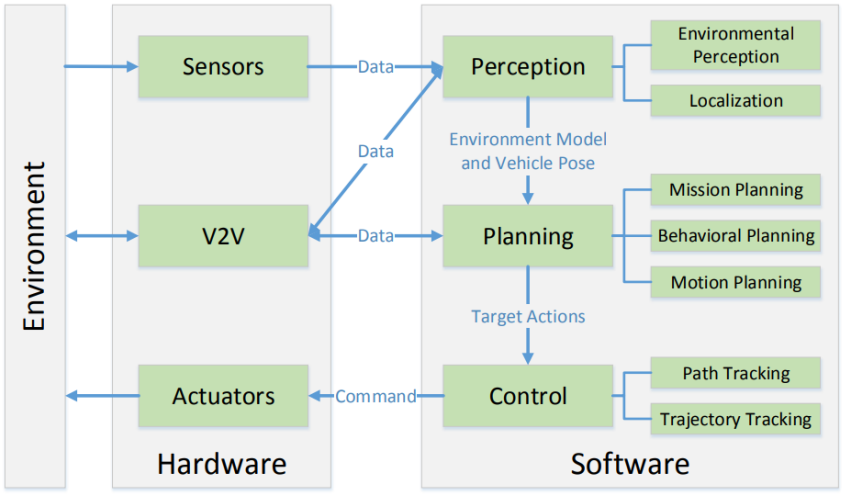

FIGURE 3.9: Autonomous Vehicles decision architecture

On AV the route planning or also called mission planning is generally performed as a graph search over a directed network that represent the the path connectivity. The planning phase can be further divided into two subsections [53]. We can distinguish :

- Behavioral Planning
- Motion Planning

The behavioral planner task is to ensure that the vehicle follows the road rules and interacts with other actors in a safe manner while trying to achieve its goal. Its goal is to reach the desired destination. The behavioral planning is usually realized through the achievement of consecutive local goal settings placing virtual obstacles on the path and adjusting the drivable region bounds on the run. The first planner of this kind were modeled as Finite State Machine (FSM) of varying complexity to cover all the possible scenario that could be encountered performing autonomous driving. Obviously with a FSM structure a vehicle may perform unexpectedly if it encounter a situation that is not previously accounted. Livelock or deadlock between the states are always a possibility when using this kind of approach and for this reason, to manage a larger ruler set, new decision structure have been developed. The motion planning is a much broader research field than the behavior planning and range in various applications such as manufacturing, medical and emergency response , agriculture and transportation. Usually these kind of planners are evaluated against the computational efficiency and completeness. The motion planning problem has been proven to be very computational expensive. For this reason many approaches to the resolution of this specific problem suffer from the curse of dimensionality. The more

is wide and complex is the space of motion the more the computational capability to solve it must be high. One of the approach to lower this intrinsic complexity is to transform the continuous space in a discrete set of possible positions [54]. The algorithm can be referred as complete only if it terminates in a finite time returning a solution if it exists or indicates if no solution is available for the particular configuration given as input. Even guarantee the completeness of an algorithm for these kind of problems is really difficult because some configuration may demand an exhaustive research in the space of solution that is really computational expensive. The most common approaches to the discretization of the solution space are :

- combinatorial planning
- sampling-based planning

The combinatorial planning approach consists in building a discrete representation which exactly represent the original problem. In this way it is possible to find a complete solution to the motion planning problem but the necessary computational resources rise with the increment of the size of the movement space and the number of obstacles in it. For this reason this approach cannot be applied to instances of the problem that are too big in terms of motion space. Sampling-Based approaches instead rely on a random sampling of the continuous space and on the construction of a feasible path graph. The feasibility check is performed by ensuring that no collision will happen using the trajectory with obstacles or edges. This kind of approach guarantee a probabilistic completeness. It consists in the prove that given sufficient time to check an infinite number of samples the probability to find a solution, if it exists, converges to one. However it is not only important to find a solution but also the quality in terms of goodness expressed with an evaluation function. Some approaches used also in heuristic methods select an acceptable base solution and try to improve it during the execution of several computational step. The number of steps can be chosen accordingly to the maximum response time of the algorithm. Another complexity factor that must be taken into account is that usually environment are not static or known a priori. If the motion planning is going to be performed in an dynamic environment it has to be active during all the length of the trip to destination. It is necessary because it has to adapt and eventually modify the planned trajectory according to possible movement of obstacles or the appearance of a new one. The planning phase just like the others that have been described before must be performed in a cyclic way until the achievement of the goal.

Chapter 4

Mobility Models for the next vehicle generation

In this chapter a deep study of the mobility models used inside simulation are presented. New mobility models for the future vehicle must take into account several factors that classic models do not consider because are considered negligible. With the advancement of technology new vehicles will use new form of propulsion. Nowadays the research community is investing a lot of effort studying electric engines for vehicles. These new models are then used into simulations to verify the goodness of new protocols and technical innovations. For this reason this chapter will provide the basic knowledge about simulations and simulators and will describe the reference mobility models and simulators used in literature. Also an implementation of a mobility model for electric vehicle is presented along with an ad-hoc simulator developed in java. The simulators that are herein described are well known simulators that are used by the scientific community for a long time and have been proven sound and solid. In the future of vehicle environment there are for sure new form of locomotion powers. New mobility models must be implemented for several reasons. Already in our city is possible to see charging station for electric cars and they are spreading in a really fast way thanks to several advantages. Moreover the price of the technology behind electric cars and vehicles is steadily lowering through time. The adoption of new propulsion system is almost necessary. The IEA(2012) estimates that 20% of global primary energy use and 25% of the energy-related carbon dioxide (CO_2) emissions are attributable to the vehicle sector alone. If the current trends persist, global energy demand for transport and energy related CO_2 emissions are expected do double by 2050. In the end we have almost reached the end of the cheap-oil era. For these reasons electric and hybrid vehicles are the current most probable option for the future of vehicle mobility and a lot of researches are currently in progress [55]. Electric vehicles have already been around since the late 1800s. They were very popular and a number of Electric Vehicle (EV)s were sold until about 1918. For example in the year 1900, about 4200 automobiles were on the road, out of which 38% were electric, 22% gasoline powered, and 40% steam. With the advancement of gasoline motors, low-cost gasoline, and the invention of electric powered starter for the inner combustion engines, the curiosity in EVs

totally declined. Regardless of it, some motor vehicle companies continued to focus on study and advancement of EV systems by experimenting with various kinds of propulsion motors, energy storage space systems, and in addition incorporating advanced power transformation technologies. The used technology tendencies are also equally relevant for hybrid and plug-in hybrid vehicles. The effort to improve overall quality of life in cities is certainly shared between different sectors, and the road transport sector is expected to reduce its emissions by 95%. For this reason it is considered essential to develop low carbon and oil independent transport solutions. However with the forecasted constantly increasing number of passengers, total independence on essential oil and zero tailpipe emissions technology will probably be needed in the long term. Moreover, for now this sector is highly dependent on oil, which raises source depletion and security of supply concerns. Improvements in efficiency of current vehicles, biofuels and electrical powertrains are three solutions being considered to tackle this issue. Road based transportation accounts for a large share of Europe CO_2 emissions, 22% in the UK. Lastly, urban pollution due to vehicles use causes health problems. A growing concern about climate change triggered agreements between EU countries to cut their emissions by 80% by 2050 to stabilize atmospheric CO_2 at 450 ppm in order to keep global warming under 2° C.To store electric energy on a vehicle a battery pack is necessary. These battery arrays can be implemented with several technologies that have their pros and cons. It is particularly important for Lithium-ion technology batteries a very good and careful management since, despite their goodness, they can be damaged and present a risk of fire or explosion in particular circumstances, and their high cost makes even more crucial an increase in their cycle and duration lives. Battery cell characteristics can differ slightly, it is necessary to balance the charge between each cell to prevent damage and improve the lifetime of the stack. It involves voltage and current monitoring in each cell, and temperature monitoring in multiple points to ensure that none of the cell is functioning outside its operational conditions. The benefits offered are a longer duration and routine life, increased safety and a higher power capability for a relatively small cost boost. These parameters are crucial for users as well as to optimize the charge and discharge processes and must be communicated to on-board systems (safety system, communication with the driver, engine management). Monitoring can improve both range and life cycle price of batteries which could mitigate the current social acceptance difficulties. In order to meet these goals, mathematical models of the behaviour of each battery technology have been elaborated. Moreover it is important to operate the battery in a safe, efficient and nondamaging way. Battery blocks are composed of cells arranged in parallel and series to meet the needs of the engine. Electric Battery Management System (BMS) have several main roles: one of these is to monitor the electric battery to determine information such as its State of Charge, Condition of Health (the ability of the battery to deliver its specified output) and Remaining Useful Life. It has been demonstrated that advanced BMS can significantly improve the efficiency of EV and extend the life of electric batteries

. Different modelling methods have already been proposed in literature. Some models have been proposed that take into account the change in capacity and impedance of batteries during their lifetime (decreasing the available power), the model for the monitoring and prediction of degradation, and the development of advanced charging algorithms to maximize the battery existence . Passive balancing strategies have been used, during charge, using dissipation through resistors, but it is not an efficient solution. [56] The battery is the most common energy storage device and the battery charger is an integral part of the EV system. The propulsion system in an EV consists of a power storage system, the energy converter, the propulsion engine and connected controllers .One of the most important subsystems that's needed is within an EV is the propulsion system, which gives the tractive power to propel a car [57].

4.1 Simulation Basics

A simulation is just a reproduction of the behavior of a system. To reproduce this behavior two kind of models can be used:

- A Concrete Model;
- Abstract Model;

An example of a concrete model is a scale reproduction of a boat in a particular tank while an abstract model involve the use of instruments like computers or mathematical models. The behavior of a simulated system is made through probability distribution to generate casual events and gather statistics and performances. Abstract models are used when the systems are too complex or the scenarios cannot be directly observed. Sometimes in order to prove the veracity of a thesis simulators are needed. They are specifically needed when the test for a thesis are too expensive to be done or when the proposal is so innovative that is not supported by the current technology/devices/protocols [58]. The most common simulators use discrete-event simulation (DES) models. These models describe the operation of a system with a discrete sequence of time events. Each event occurs at a particular instant in time and can trigger a state change in the system. Between consecutive events, nothing changes in the system because no event occurred; thus the simulation can directly jump in time from one event to the next. On the other hand continuous simulation continuously tracks the system dynamics over time. Instead of being event-based it is activity-based simulation; time is decomposed into small time slices and the system state is updated according to what happened in each time slice. Because discrete-event simulations do not have to simulate every time slice, they can typically run much faster than the corresponding continuous simulation thanks to the fact that the simulation time jumps from an event occurrence time directly to the next[59]. The model used to represent the behavior of the system must

be complex enough to depict the system but have to remain as simple as possible. The main elements that usually are used in a simulation model are :

- state variables;
- events;
- entities and attributes;
- resources;
- activities and delays;

A system can be described in each instant by a set of variables that for this reason are called state variables. An example of state variable is the number of users inside a queue or the number of packets in a buffer.In a discrete model the variables can change only in correspondence of defined time instant while in continuous model can change with continuity compared to time. An event is an occurrence that changes the value of a state variable. Two kind of events can occurs: exogenous and endogenous. The first are events that are external to the system while the latter are internal. Entities are system elements that need to be defined. A good example of entity is a buffer and one of the attributes that define it is the number of packets in the buffer. Resources are elements of the system that can provide a service to entities. An entity can request one or more resources and if one of these resources are not available in that moment the entity must wait in queue to use it. Instead if the necessary resources are all available they are *captured*, *withheld* for the necessary time and then *released*. An activity is an operation for which the duration is known at the beginning of the simulation. The time can be constant or the result of an aleatory variable. A delay is a period of time that is not known in advance because it depends on the condition of the system.

4.2 Electric Vehicles Overview

The electric drive systems consists, in the simplest case, of very few components shown in 4.1:

- A battery (1)
- A Inverter/Converter (2)
- A motor (3)
- Changeable or Fixed Gears (4)
- Differential Gear (5)

From the outside it is really difficult to distinguish and electric car from a fuel propelled car because, in the most cases, electric cars are created by converting a gasoline-powered car. Often the only thing that clues the nature of a electric car is the fact that even in motion it is nearly silent. During the conversion the gasoline engine is replaced by and electric motor. Near the electric motor a controller is installed to use the right power from an array of rechargeable batteries.

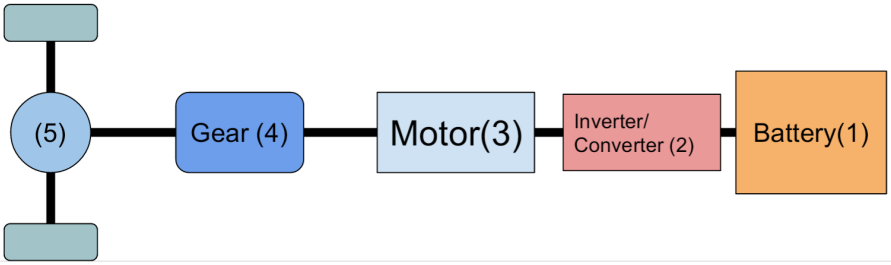

FIGURE 4.1: Base schema of an electrical vehicle

It starts with the battery in the car that is connected to the engine. The spinning rotor is what creates the mechanical energy turning the gears of the car, which, consequentially, rotate the tires. Therefore, when the electrical energy from the car battery is supplied to the engine, the coils create rotating magnetic fields that pull the conducting rods externally of the rotor along behind it. The coils within the stator made of conducting wire are arranged on opposite sides of the stator core and act as magnets . Electrical energy comes to the stator via the car's battery. The rotation of the wheels is what then powers the alternator in the car and the alternator recharges the battery.The battery powers the engine, which powers the gears and wheels. In an electrical car usually there is no alternator. As referenced above, the electric battery starts the motor, spinning the gears, which rotates the wheels. This process happens whenever the accelerator is pressed. The rotor gets pulled along by the rotating magnetic field, requiring even more torque. While there is absolutely no separate alternator, the engine within an electric car acts as both the electric motor and an alternator. That's why electric cars are therefore unique [60]. The batteries technology is still too pricey for the mass adoption of electrical vehicle. The cheapest and most common type of batteries are the lead-acid due to the mature and well known technology and availability. Nickel metal hybrid batteries are less efficient in charging and discharging but have and higher energy density. Their life is exceptional long but as a downside they have very poor efficiency and high self discharge rate. Sodium Zebra technology batteries have poor density but good resistance. The only problem is that the chemical reaction that produce energy is started by heat so this kind of batteries must be heated before use. For this reason they are not suitable to be used in rigid climate. One of the most promising battery technology is Lithium.ion. It is already a mature technology and is dominating in the Electrical vehicle development because it has an impressive energy density and a good charge/ discharge efficiency [56]. The inverter and the converter are often put together in

single block and are necessary to recharge the battery pack and to provide energy to the motor. Because of the heating an adequate cooling and ventilation are strongly necessary to keep the components operational. The motor converts electrical energy into mechanical energy. These kind of motors have an high power-to-weight ratio and are composed of a stator and a rotor. The rotor slides into the stator and is enclosed in two bells that are placed on either sides [61].

4.3 Simulators

To better understand how a mobility model works the Simulation of Urban MObility (SUMO) simulator is here presented along with its module to manage the mobility and the position update of vehicle inside the urban map. Also an implementation of the mobility model inside a java ad-hoc simulator is presented with a comparison with the respective modules.

4.3.1 SUMO

SUMO is an open source microscopic multi-modal traffic simulation framework that allows to simulate a traffic demand. A traffic demand can be defined as the quantity of vehicles that must pass through a particular street or a set of chosen roads. It is microscopic because it allows to model each single vehicle specifying its route and behavior. Its simulations are deterministic but random number generators with different seeds can be used to differentiate each simulation to the other. The set of tool that the SUMO package offer is wide and contains all the programs that are needed do setup, run and analyze a simulation scenario. It also offers an OpenGL graphic interface useful to debug and test the correct behavior of the traffic. One of the most important features of sumo is the interoperability with other applications at runtime. Also this simulator is full written in c++ using only standard libraries [62]. Since 2001, after being recognized as a valid tool from the research community, it has been used in several national and international research project such as:

- traffic ligths evaluation
- route choice and re-routing
- evaluation of traffic suveillance method
- simulation of vehicular communication
- traffic forecast

The two major goals in the design of this software were portability and high execution speed. For this reason the first versions were developed to be run from the command line only without any graphical interface and configuration parameters were input as arguments from terminal. Without the load of a graphic visualization the computational time was really short. Each software had a different purpose

and was designed to run individually from the others. This function fragmentation allows an easier extension of each of the applications within the package due to a smaller amount of code that needs to be known and modified. Furthermore, it enables the usage of swifter data structures, each tweaked for the current objective, rather than using complicated and ballast-loaded ones. For this reason SUMO more uncomfortable in comparison to other simulation packages. The representation of road networks and traffic demand are needed to simulate in a particular format. SUMO road networks can be either created using an application named *netgen* or imported from a digital road map. The road network import program is called *netconvert* and allows to read networks from other famous traffic simulators. It also reads other common formats that are used for describing maps as the OSM format. SUMO is most used as microscopic traffic simulation. Each vehicle can be defined explicitly and must possess at least an identifier, a departure time, and the vehicle's path through the network. Optionally each vehicle can be described in a more detailed way adding departure and arrival properties, such as the lane to use, the velocity, or the position. It is possible to associate to each vehicle a type which describes the vehicle's characteristics and the parameters needed for the movement model. It is also possible to associate to each vehicle a pollutant or noise emission class. Additional parameters allow the description of the vehicle's appearance within the user interface. The description of vehicles can be specified from different sources. In case of large-scale scenarios usually it is necessary to use "origin/destination matrices" (O/D matrices). These matrices can specify the transition between map areas in terms of vehicle number in a certain amount of time. To describe large-scale traffic scenarios, they are the most appropriate choice. The SUMO software collection includes a tool called *od2trips* that is capable of converting O/D matrices to single vehicle trips. *od2trips* also assigns an edge as a depart/arrival position. The generated paths are composed of a begin and finish road, a starting time but no explicit route information. paths are usually calculated by executing a traffic assignment using usually a shortest path algorithm that take in consideration different weight parameters [63]. The simulation is time-discrete with a time simulation step duration of 1s. It is possible to configure a different simulation step but it could cause problems for other modules that consider only the default step length. For example with a simulation step lesser than 1 second the major part of the mobility models cannot prevent events of car collisions because the constraints of safe distance are compliant only with the 1 second time step. The simulation is space-continuous and vehicle's position is defined by the lane on which the vehicle is travelling and the distance from the starting point of the lane. Each vehicle's speed is computed using a so-called car-following model during each simulation step. SUMO provides the generation of various output results for each simulation run. These cover anything from simulated induction loops to single position of vehicle in each time step for all vehicles. It can also provide complex information about each vehicle's path or aggregated measures along a street or lane. SUMO can also provide a noise emission

and a pollutant emission / fuel consumption model. One of the most important contribution to SUMO is Traffic Control Interface (TraCI) that is a module that allows a remote control interface to adapt the simulation during runs. With TraCI any external framework can query sumo to gain information about the simulation mobility environment and to modify at run-time its state [64].

TraCI

After starting SUMO, clients connect to SUMO by setting up a TCP connection to the appointed SUMO port. Through this connection it is possible to retrieve data and modify on the run the parameters of the simulation. The connection and closing phase to TraCI are shown in Fig. 4.2.

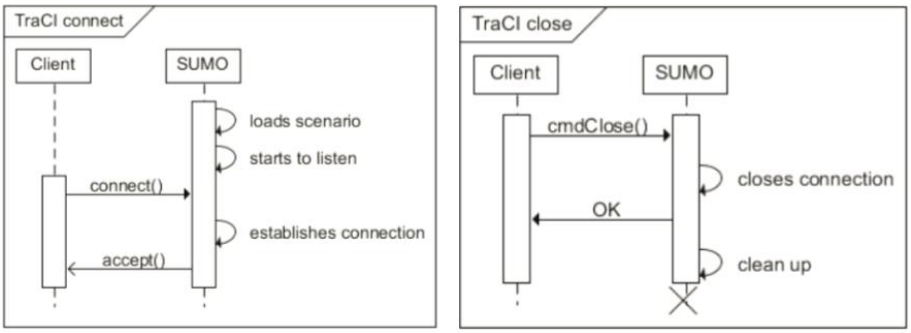

FIGURE 4.2: Traci connection open and close phase

The client sends instructions to SUMO to manage the simulation run, to determine single vehicle's behavior or to request information about the simulation scenario. SUMO answers with a Status-response to each request and optional information results that depend on the supplied command. The client must trigger each simulation step in SUMO using the TraCI/Control-related command *Command 0x02*: Simulation Step command. If any subscriptions have been performed, the subscribed information are returned before the time step advancement. In case of multiple clients connected to sumo the simulation will progress to the next only when all clients have successfully sent the simulation step advance command [65]. At the moment all clients get all subscription. The client is in charge for closing the connection making use of the close command. When all clients send the close command, the simulation will end, freeing all resources.Each TraCI message is encapsulated inside a simple TCP message that is composed of a small header that contains the overall message size and then a a set of consecutive commands. For each command the length and the identifier of the command are put first. The structure of a generic TraCI message is shown in Fig. 4.3.

Each command has an ack with a status response. In the response the identifier refers to the command that needs to be acknowledged and also includes a result and a description. The result is an ubyte that can assume the following values:

- 0x00 in case of success;

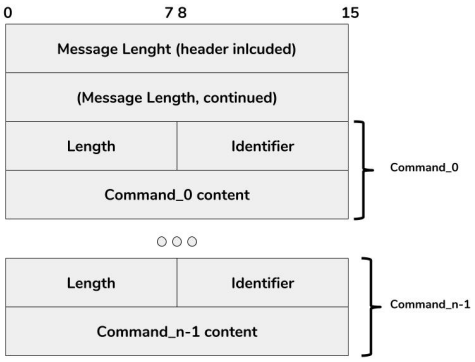

FIGURE 4.3: TraCI generic message

Data Type	Size	Description	Id
ubyte	8 bit	integer numbers (0 to 255)	0x07
byte	8 bit	integer numbers (-128 to 127)	0x08
integer	32 bit	integer numbers (-231 to 231-1) includes bitsets with bit0 meaning the least significant bit	0x09
double	64 bit	IEEE754 floating point numbers	0x0B
string	variable	32 bit string length, followed by text coded as 8 bit ASCII	0x0C
stringList	variable	32 bit string count n, followed by n strings	0x0E
compound object	variable	Compound object, internal structure depends on concrete object. The compound object identifier is always followed by an 32bit-int denoting the number of component types. Then the components are given in sequence.	0x0F

TABLE 4.1: TraCI datatypes

- 0xFF in the command has failed due to some errors;
- 0x01 if the command is not yet implemented in SUMO;

Values inside the TraCI message can be represented by atomic or compound data type. The supported datatypes are shown in Tab. 4.1.

Theory of Traffic Simulations

When talking about traffic simulation there can be a distinction related to the level of detail of the simulation. The three major detail levels are :

- macroscopic;
- microscopic;
- mesoscopic;

- submicroscopic;

In a macroscopic model the basic entity that is analyzed and characterized with parameters are the traffic flows. Microscopic models focus on the vehicle entity describing it in terms of movement, position, speed and so on. In these models every single vehicle is simulated on the street assuming that the behavior of each vehicle depends on the vehicle characteristics and on the driver controlling behavior. Mesoscopic models are located between the boundaries of microscopic and macroscopic models. In these models usually each vehicle is simulated using a queue approach. The movements of the vehicles are reduced to the advancement in a specific queue or in queue change. A sub microscopic model possess a very high level of simulation detail. They can describe the torque of the engine, the rotation speed, preferred gear switching action and so on. This level detail has an important weakness because to support this huge amount of information and calculation the computation time is far more high than other models. Obviously for this reason large scale road networks cannot be used because the simulation time will be too much high [66].

4.3.2 Mobility Model in SUMO

One of the most famous mobility simulator among the research community is SUMO. The SUMO project stared around the 2002 as another yet traffic simulator but, unlike other possible tools, as a open-source program. In this way during these years a lot of contributions channeled into this project improving it and adding new functions and modules. For this reason this simulator has been used as a reference testbed of new traffic models [67]. SUMO is not only a traffic simulator, but rather a collection of programs which help to prepare and to compute traffic simulations.

SUMO mobility model

The default-car following model in SUMO is a modified version of the Krauss model, introduced by Stefan Krauss in his PhD thesis in 1998 [68]. This mobility model is a collision-free model. Any collision is avoided because the following vehicle modulate its speed according to the ahead vehicle. The parameters that can be configured in Krauss SUMO model are depicted in Tab. 4.2. The Krauss model is a microscopic, space-continuous, car following model that is based on the safe speed. The concept of safe speed is pretty simple. The safe speed is the speed that a vehicle must have in order to have enough time to stop its movement before getting in touch with an ahead vehicle that suddenly slow down. The safe speed is calculated as:

$$v_{safe} = v_l(t) + \frac{g(t) - v_l(t)t_r}{\frac{v_l(t)+v_f(t)}{2b} + t_r} \tag{4.1}$$

Where

- $v_l(t)$ is the speed of the leading vehicle at time t

Parameter	Description
accel	The acceleration ability of vehicles of this type (in m/s^2)
decel	The deceleration ability of vehicles of this type (in m/s^2)
emergencyDecel	The maximum deceleration ability of vehicles of this type in case of emergency (in m/s^2)
sigma	The driver imperfection (between 0 and 1)
tau	The driver's desired (minimum) time headway. This is based on the net space between leader back and follower front)
minGap	minimum allowed distance between vehicles

TABLE 4.2: Parameter Configuration for the Krauss model in SUMO

- $g(t)$ is the distance (gap) between the two vehicle at time t
- t_r is the driver reaction time that is usually around 1s
- b is the maximum deceleration of the vehicle expressed in m/s^2

Often in a urban environment V_{safe} may be greater than the maximum speed allowed from the road limits. Moreover the maximum allowed speed could also be greater, during a time step, of the possible speed that can be reached due to the vehicle maximum acceleration. for this reason the computed resulting speed which is called *desired* or *wished* speed is

$$v_{des} = min[v_{max}, v + at, v_{safe}] \tag{4.2}$$

Where t is the duration of a time step simulation [69].

SUMO Map module

SUMO uses its very own representation for the simulation map in which the vehicles travel. Every information is inside an eXtensible Markup Language (XML) SUMO network files. This file describes the road and intersection where the vehicles run along or across. A SUMO network can be considered as a directed graph in which the junctions are the nodes of the graph while streets and roads are the edges. Each street is composed of a collection of lanes and is characterized by parameters that describe their position, shape and speed limit. In each junction (node) there is a description of the right way of regulation and information about the traffic lights logic. Also the information about the connection between lanes are stored in junctions. Even if is human readable due to the XML format this file is not meant to be edited by hand. The proper way to create this file si by using the tool *netedit*. This is a graphical tool that allows the design and description of vehicle networks. Another way of generating net. XML file is by converting another map format in the sumo net description. With this tool is possible to import maps of real locations into sumo to use them in the simulation environment. One of the most common source for

Name	Type	Description
id	id(string)	The id assigned to the edge
from	id(string)	The id of edge starting node
to	id(string)	The id of the edge ending node
priority	integer	The importance of the road (optional)
function	enum	an abstract edge purpose

TABLE 4.3: Edge Attributes

maps is Open Street Map (OSM). All the information that are shared from the users on this free wiki world map are successfully imported in sumo. Due to the lack of information about altitude on OSM to take in consideration the street gradient into simulation environment these information have to be retrieved from other source [70].

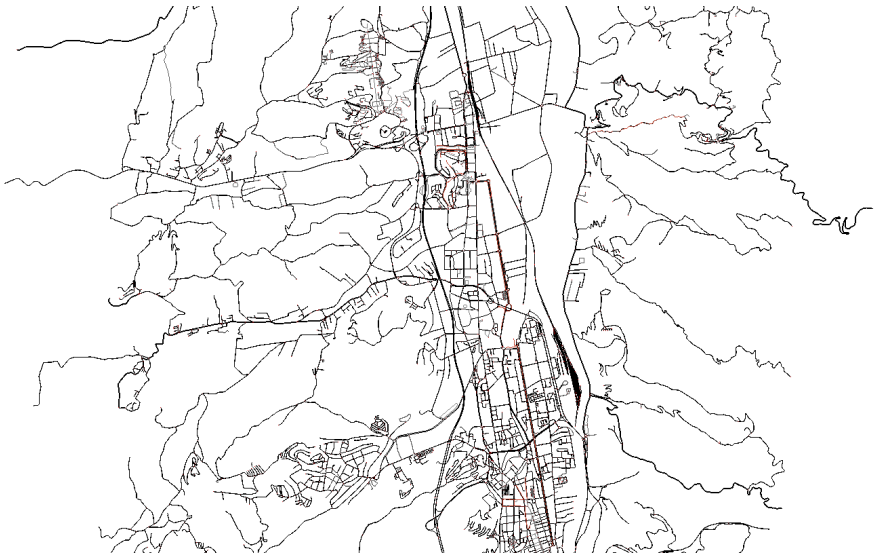

FIGURE 4.4: An example of a converted map from sumo

In 4.4 a .net.xml map converted from OSM file is shown. Each edge can be defined with several parameters that can be placed as attributes of the element *edge*. The possible attribute and the value that they can assume are summarized in 4.3

The *function* attribute describes how an edge is used and if has a correspondence in the real world or it is just a fictitious edge to help for assignment. Possible value of the *enum* are:

- *normal*: this is the default value if not specified and denote that the edge is just a plain part of the road network.
- *connector*: This value denotes that the edge is a macroscopic connector. There is no difference between connector and roads for simulation purpose simply connectors do not have a correspondence in the real world.
- *internal*: The edge is a part of a intersection

name	type	description
id	id(string)	The lane id
index	running number	the number of the lane in the edge
speed	float	The maximum speed allowed on the lane (m/s)
length	float	the length of the lain in m
shape	position vector	the geometry of the lane in polyline points

TABLE 4.4: Lane Attributes

name	type	dexcription
id	id(string)	The id of the junction
x	x-position(real)	The x-coordinate of the junction
y	y-position(real)	The y-coordinate of the junction
z	z-position(real)	The z-coordinate of the junction (optional)
incLanes	id list	The ids of the lanes that end at the intersection
intLanes	id list	the ids of the lanes within the intersection
shape	position list	A polygon describing the boundaries of the intersection
customShape	bool	flag that indicates if the shape was customized

TABLE 4.5: Junction Attributes

Each lane can be composed of one or multiple lane. For example a one way street is represented with an edge with only one lane while a two-way street with an edge with two lanes. Inside the .net.xml file a lane is a nested element inside the edge element. Also the lane element owns several attributes that help define the behavior of the vehicles that move through it.

The junction element represents a node in a graph that is the interconnection between edges. It owns the information about the screams cross including also the right-of-way rules vehicle have to follow when crossing. The junction is described by following attributes

The attribute *z-position* of a junction is optional but is very important because it is used from some consumption models to compute the energy consumption at each timestep. It is really difficult to find maps that, after the conversion with *netconvert*, has also this kind of information for this reason some tweaks are needed. Very few areas on OSM possess the information about altitude but it is possible to integrate these information in other ways. One of the possibilities is to use the *osmosis-srtm-plugin* which can process the OSM input file together with SRTM data in order to add a height *ele* tag to each node. The srtm file are provided by the United States

Geological Survey (USGS) that is a US department of the Interior [71]. From the selector of the site it is possible to select an area and retrieve the information about digital elevation as .hgt.zip files. Once the OSM file is integrated with information about the digital elevation it is possible to convert it in the sumo network configuration. Some advanced modules use elevation information to better simulate real case scenarios.

SUMO Consumption module

Sumo doesn't have any standard consumption module for fuel propelled vehicles. It implements instead the emissions model. The current default model is based don the HBEFA3/PC_G_EU4 that si a gasoline powered Euro norm 4 passenger car modeled using the HBEFA3 model.Handbook Emission Factors for Road Transport (HBEFA) contains information in a database [72] about each pollutant that can be generated by vehicles but SUMO implemented only the most important:

- CO_2
- CO
- HC
- NO_x
- PM_x
- fuel consumption

In this way it is possible to assign to each vehicle and emission class and log as output consumption an pollutant emissions. Electric mobility model was implemented by Tamás Kurczveil and Pablo Álvarez López from the TU-Braunschweig. The core model is implemented in the vehicle device *device.battery* and additional features are charging stations and an XML output option to log during each timestamp the consumption of each vehicle. The implemented electric vehicle consumption model considers mechanic and electric vehicle parameters and computes the variation of the level of discharge between discrete time steps. If the engine needs a higher torque than its defined maximum, recalculations of the vehicle speed and acceleration occur in order to adapt to the constraints of its elements. The level of discharge is subsequently determined using an electrical traction model [73]. The vehicle battery energy depletion is computed by summing its kinetic, potential, and rotational energy gain components from one discrete time step to the next, and then subtracting the losses caused by energy wasted from various resistances. The vehicle's energy $E_{veh}(t)$ at the discrete time step t can thus be calculated as

$$E_{veh}(t) = E_{kin}(t) + E_{pot}(t) + E_{rot,int}(t) \tag{4.3}$$

that expliciting the various terms it becomes

$$E_{veh} = \frac{m}{2} * v^2(t) + m * g * h(t) + \frac{J_{int}}{2} * v^2(t) \tag{4.4}$$

Where

- $E_{veh}(t)$ is the vehicle's energy at the discrete time step t
- m is the vehicle mass
- $v(t)$ is the vehicle speed at time t
- g is the gravity acceleration
- $h(t)$ is the vehicle altitude at time t
- J_{int} is the moment of inertia of internal rotating elements

Regarding the energy losses ΔE_{loss} are generated by air drag, rolling friction and curve resistance. Another term of the loss is constant and represent user consumption as for example air conditioning. So the energy gain between two simulation steps can be calculated as

$$\Delta E_{gain}(t) = E_{veh}(t+1) - E_{veh}(t) - \Delta E_{loss}(t) \tag{4.5}$$

The energy loss term is made by four different components

$$\Delta E_{loss}(t) = \Delta E_{air}(t) + \Delta E_{roll}(t) + \Delta E_{curve}(t) + \Delta E_{const}(t) \tag{4.6}$$

Each of these components can be further defined

$$\Delta E_{air}(t) = \frac{1}{2}\rho_{air} * A_{veh} * c_w + v^2(t) * |\Delta s(t)| \tag{4.7}$$

where ρ_{air} is the air density, A_{veh} is the front surface area of the vehicle, c_w is the air drag coefficient and $s(t)$ is the covered distance.

$$\Delta E_{roll}(t) = c_{roll} * m * g * |\Delta s(t)| \tag{4.8}$$

In this equation c_{roll} is the rolling resistance coefficient and

$$\Delta E_{curve}(t) = c_{rad} * \frac{m * v^2(t)}{r(t)} * |\Delta s(t)| \tag{4.9}$$

Here c_{rad} is the curve resistance coefficient

$$\Delta E_{const}(t) = P_{const} * \Delta t \tag{4.10}$$

P_{const} is the power absorbed by constant consumption. To empower the model also some constant efficiency and propulsion factors can be added η_{recup} η_{prop}

$$E_{bat}(t+1) = E_{bat}(t) + \Delta E_{gain}(t) * \eta_{recup} \tag{4.11}$$

$$E_{bat}(t+1) = E_{bat}(t) + \Delta E_{gain}(t) * \eta_{prop}^{-1} \tag{4.12}$$

In the implemented SUMO model there is also the vehicle charging model. The location of the charging stations and their charging power efficiency must be specified with parameters from the user. This model works by considering that vehicles are charging their batteries if they stop in the proximity of a charging station. The charging step is ruled by equation:

$$E_{Bat}(t+1) = E_{Bat}(t) + P_{chrg} * \eta_{chrg} * \Delta t \tag{4.13}$$

taking into consideration that

$$0 \leq E_{Bat} \leq E_{Bat,max} \tag{4.14}$$

4.4 Implementation of the Mobility Model in Java

The quantity and position of electric vehicles in the city environment can deeply influence the traffic dynamics and really should be taken into consideration. To analyze these kinds of scenario an ad-hoc simulator has been created in Java. This allows us better understand distribution of traffic flows and their dynamics inside urban environment. Common vehicular simulators have already been improved and refined for several years but they usually do not consider some qualities that highly influence EVs and the usage of their electric battery packs to move. Therefore, in [74] a proposal to improve the simulation model to better depict new scenarios with an important number of vehicles that use electric power to move in the urban and suburban areas. Among the factors that deeply influence battery duration of vehicles one of the most important is the street gradient. This factor will not just influence EVs but also gas engine vehicles and really should be taken into consideration when simulating flows. Also some improvement to extend EVs battery lifetime has been developed through years and have been adopted lately by automotive companies. In this simulator some innovative principles have been developed to have more accurate results. One big improvement is definitely the use of regenerative braking that is capable of transforming some of the kinetic energy that's dissipated as temperature in reusable energy that can replenish vehicle battery. The improvement to the simulation models are the influence of slope on energy consumptions and a Kinetic Energy Recovery System (KERS) module for regenerative braking. The slope module considers street slope calculating a slope coefficient that influences vehicles acceleration and consumption and also driver behavior. Every time an automobile

with KERS performs braking a few of the kinetic energy of the deceleration can be recovered and reused to improve battery life span. The KERS module is used to improve simulations of vehicles battery. So in this work inside the simulation two innovative modules have been introduced. A Map Module manages the creation of the map inside which vehicles move. Information regarding streets slope comes from the Google Maps API. The map can be represented by an oriented graph that derives straight from the representations of true maps which can be found on Open Street Map (OSM) [75]. For this reason an XML parser has been created that parse the structure of the file and build the graph representation utilized inside the simulations. For every node a request to the Google Maps API is performed in order to gather the altitude of every node. The slope is evaluated and kept as attribute of the link connecting two adjacent nodes. This coefficient measures the slope gradient of a segment thanks to the altitude information of the two nodes to which the edge is connected. In addition to the node height also an AC has been computed. TheAC of an area is calculated as

$$AC = \sum_{i=1}^{n} (segment\_length_i / total\_length) * slope_i \tag{4.15}$$

In this way it is possible to classify each area with a AC in three different classes

- low class: $0 > AC < 2$
- medium class: $2 < AC < 5$
- high class: $AC > 5$

Due to the limitations of the Google Maps Api we had to minimize the number of requests through altimetry data caching. In order to gather information about street elevation a software interface with Google Maps API web services has been developed. To collect these information it is necessary to make RESTful requests to the server specifying latitude and longitude coordinates of the point of interest. In order to explain how the regenerative braking module works it is necessary to show how the consumption is computed.

- vehicle instantaneous speed
- vehicle instantaneous acceleration (a)
- street gradient angle coming from AC
- battery energy load expressed in kWh

Regarding vehicles' movement we take into account three forms of energy depletion:

- W_v is the work necessary to move the vehicle
- W_t is the work necessary to overcome tires friction

- W_a is the work necessary to overcome air friction

Taking in consideration

$$m * a = F - m * g * sin\varphi \tag{4.16}$$

The necessary force to move the vehicle taking into account the street gradient is

$$F = m * (a + g * sin\varphi) \tag{4.17}$$

and its work is

$$W_v = \frac{1}{Efficiency_v} * \Delta t * F * \frac{V + V_{old}}{2} \tag{4.18}$$

The mass vehicle m is known as well as its efficiency. The work necessary to withstand tires friction is:

$$W_t = \frac{1}{Efficiency_v} * \Delta t * (m * g * cos\varphi * cRoll) * \frac{V + V_{old}}{2} \tag{4.19}$$

Where $cRoll$ is the friction coefficient of tires. The work necessary to withstand air friction with the assumption that the air speed is equal to 0, therefore the work is evaluated as herein shown :

$$W_a = \frac{1}{Efficiency_v} * \Delta t * \frac{\rho * cDrag * Across}{2} * (\frac{V + V_{old}}{2})^3 \tag{4.20}$$

The cubic component of speed is composed of the squared component necessary to calculate the force and a component necessary for the distance that is multiplied with Δt.And so energy depletion model can be expressed like:

$$E_d = \frac{W_v + W_t + W_a}{3600000} \tag{4.21}$$

Part of the energy that is dissipated through air heating below the brakes can be recovered through a particular device that very often is used in EV environment called KERS. To add this device effect to the model we considered the kinetic energy produced by the negative acceleration expressed in Joule (J). This energy is defined as :

$$E_{kin} = \frac{1}{2} * m * (V^2 - V_{old}^2) * \eta_{reg} \tag{4.22}$$

where η_{reg} is the recoverable energy percentage. This coefficient depends on the brake device efficiency that has a theoretical limit of 30% To better understand the components of the equations a table is provided The mobility model used in simulations is a car following model based on the Gipps model in which each vehicle travels along a road just following a previous vehicle. Four kind of roads Urban, Secondary extra-urban, Principal extra-urban and Highway have been taken into account. In this way a vehicle target velocity is chosen based on the kind of street that the car is travelling on and the driver behavior. Each automobile is usually modeled with a particular drive style that impact on accelerations and decelerations. Gipps

TABLE 4.6: Table of Symbols

Symbol Name	Description
W_v	Work for moving vehicle (J)
W_t	Work to withstand tires friction (J)
W_a	Work to withstand air friction (J)
$Efficiency_v$	Vehicle (v) Efficiency
V	Current vehicle speed (m/s)
V_{old}	Last measured speed (m/s)
φ	Slope angle (deg)
m	vehicle mass (Kg)
F	Force to move vehicle (N)
a	Vehicle acceleration (m/s^2)
$cRoll$	Friction Coefficient
$cDrag$	air Friction Coefficient
$Across$	Cross Sectional Area (m^2)
Δ_t	Observation Time Window (s)
ρ	Density air coefficient (Kg/m^3)
η_{reg}	Recoverable energy in percentage
E_d	Total power necessary (J/s)

model is one of the the most utilized microscopic model in vehicular environment simulation. Furthermore, we considered also that different users may possess different drive style. The constraints that need to be respected in the Gipps model are few and assure that no car accidents will happen:

- Vehicle n will not exceed its driver's desired speed;
- Its free acceleration must be adequate to the speed change;
- Consider the reaction time of a typical driver. If the preceding vehicle brakes the following one must have enough space and time to slow down in safety;

This type of model is used primarily to analyze the consequences on the traffic stream when there are changes in the network. The Gipps model is a behavioral car-following model which describes how one car reacts to the behavior of the preceding automobile and has been enhanced considering driving styles such as BN (Below Regular), N (Normal), A (Aggressive) and VA (Very Intense). Prior to the Gipps model the most part of approaches were based on the acceleration variation[76]. Through several simulation campaigns involving different kind of vehicles and areas we highlighted the influence of street slope in cars consumption. This factor becomes really important taking into account new forms of propulsion like EV propulsion with electric engines. The consumption / harvesting of energy due to the street gradient must be taken into account to better depict real case scenarios. A main schema of the modules that compose the proposed ad-hoc simulator is shown in 4.5. Besides classic simulation modules like the scheduler and the mobility module we added as aforementioned some module responsible of parsing the OSM

map file, a slope module that modulate consumption related to the gradient of the street. The slope module also deals with the energy harvested from the regenerative braking.

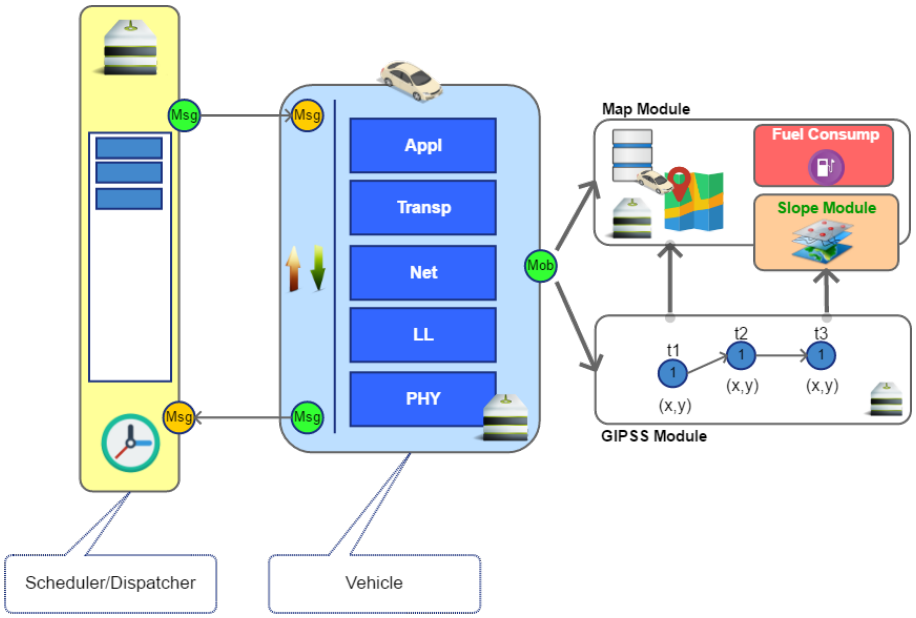

FIGURE 4.5: Modules of the simulator.

Also in [77] the authors propose an electric vehicle simulator based on the well known SUMO to better investigate the issue of energy consumption of EV. They expanded the classical two-dimensional simulator modeling the altitude of the streets and transforming in this way SUMO into a three-dimensional simulator that takes into account street gradient in energy consumption model. As shown inf Figure 4.6 the probability distribution function is different considering or not the additional consumption/regeneration due to the gradient influence. It is also possible to see in a area with a sever AC how consumption changes in relation to the The energy consumption of an electric vehicle is deeply influenced also by the weight and the mass of the bodywork of the car. This happens because the consumption model takes into account also the work necessary to win against the air drag that depends on the exposed across surface. The weight deeply influence the work needed to move the mass of the vehicle on the inclined plane. In Figure 4.7 the average energy consumption related to the vehicle mass is shown with and without street gradient taken into account.

4.4.1 Java ad-hoc simulator modules

The developed java simulator for electric mobility is composed of several modules that will be shown in the next sections. To configure a simulation scenario it is possible to use an xml configuration file in which will be specified all the parameters

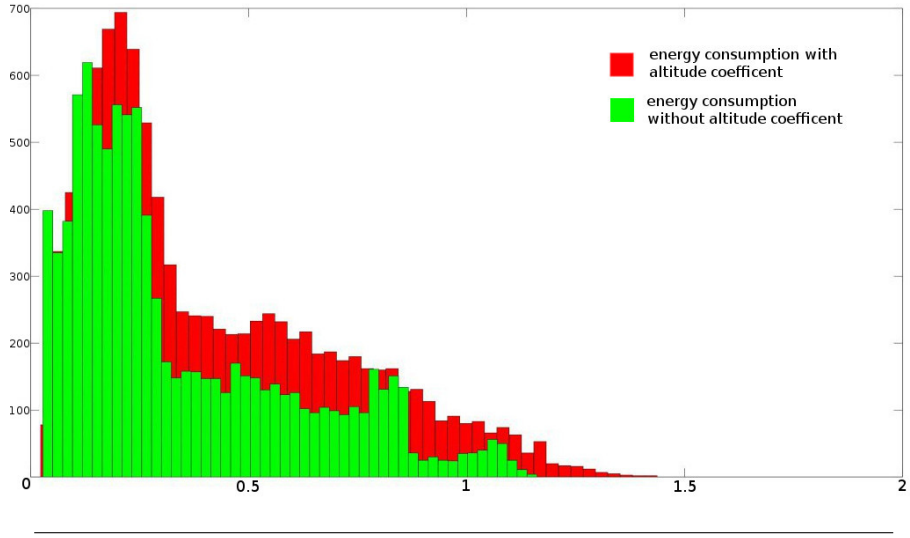

FIGURE 4.6: Comparison of pdf considering map area with altitude coefficient and not

necessary for the simulation. This file is then parsed to build the simulation run by a parser. the possible parameters that can be configured are:

- Name of the project with tag <*description*<
- the type of channels thanks to a <*channel*> section containing as attributes
  - the *id* of the channel
  - the *type* for example 802.11p
  - the *capacity*
  - the *packet_dimension*
  - *propagation_time*
- the source and destination of traffic flows in elements named flow with the next attribute
  - *source* the node id source of vehicle
  - *destination* the node id to reach
  - *exitAt* the time at which the vehicle flow must start
  - *generation_rate* the rate of the vehicles to be generated expressed in vehicle per minute
  - *color* the color the vehicle of the flows will assume
  - *max_vehicles* the number of vehicles generated for the flow
  - *behaviour* the behavior the vehicle will adopt
- the type of vehicle that will be generated are specified with two element $<$ *vehicleC* $>$ and $<$ *vehicleE* $>$ the attributes to characterize them are

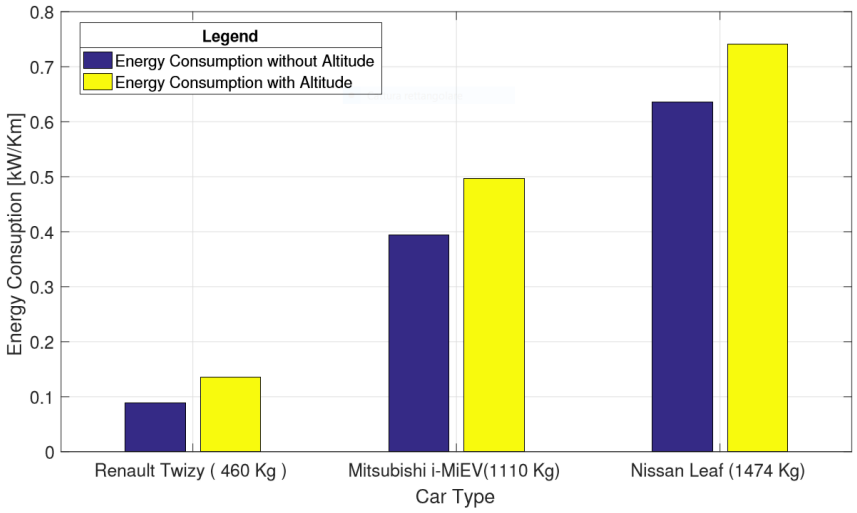

FIGURE 4.7: Average energy consumption related to vehicle class.

- *id* the id of the type of vehicle
- *brand* the brand of the vehicle
- *model* the model of the vehicle
- *mass* the mass of the vehicle in Kg
- *engine_efficiency* it's a coefficient that describes the efficiency of the engine between 0 and 1
- *CdA* is the drag coefficient already multiplied for the across area
- *max_speed* the max speed of the vehicle
- *max_acceleration* the max acceleration of the vehicle
- *max_deceleration* the max deceleration of the vehicle

for electrical vehicle we have further attributes that can be configured:

- *battery_capacity* the battery capacity in Kwh
- *charging_time* the time that is necessary to an electric vehicle to fully replenish the batteries

Mobility Gipps Module

As aforementioned the mobility model used is based on the Gipps car following model for micro-simulations [76]. The gipps model is a behavioural car-following model which describes how one car reacts to the behaviour of the preceding vehicle. This kind of model is used mainly to analyse the effects on the traffic flow when changes occur on the road network. Before the Gipps model the most part of approaches were based on the acceleration variation described in :

$$a_n(t+\tau) = l_n * \frac{[v_{n-1}(t) - v_n(t)]^k}{[x_{n-1}(t) - x_n(t)]^m} \tag{4.23}$$

In 4.23 $n-1$ is the following vehicle, n is the followed vehicle, is the reaction time , $x_n(t)$ is the location of vehicle at time t, $v_n(t)$ is the speed of n at time t and l_n, k and m are parameters to be estimated. Gipps model instead is derived by setting limits that permit to calculate safe cruise speed according to the the preceding vehicle. The main assumption of the model is that the driver of the following vehicle adapt his speed in order to maintain a safe distance from the preceding vehicle and to have enough time to stop it without causing an accident. The constraints that need to be respected in the Gipps model are :

- Vehicle n will not exceed its driver's desired speed
- Its free acceleration must be adequate to the speed change
- Considering the reaction time of the driver if the preceding vehicle brakes the following one must have enough space and time to slow down in safety

The first two constraints can be expressed together inside the inequality :

$$v_n(t+\tau) \leq v_n(t) + 2.5a_n\tau(1 - \frac{v_n(t)}{V_n})(0.0025 + \frac{v_n(t)}{V_n})^{1/2} \tag{4.24}$$

In 4.24 a_n is the maximum acceleration of the vehicle n, V_n is the desired speed of vehicle n and τ is the apparent reaction time considered as a constant. Regarding the constraint about braking if vehicle $n-1$ starts braking as hard as desirable at time t then the vehicle will stop at some point x^*_{n-1} derived from

$$x^*_{n-1} = x_{n-1}(t) - \frac{v_{n-1}(t)^2}{2b_n} \tag{4.25}$$

here b_{n-1} is the most severe braking that $n-1$ vehicle can undertake and it is expressed as a negative factor. The following vehicle will not react until time $t+\tau$ and it will not stop before reaching point x^*_n expressed as

$$x^*_n = x_n(t) + [v_n(t) + v_n(t+\tau)] * \frac{\tau}{2} - \frac{v_n(t+\tau)^2}{2b_n} \tag{4.26}$$

To ensure safety also the size of the vehicle must be taken into account $S_n - 1$. Expressed this way the driver of vehicle n would have no margin of error and so a possible additional delay θ so the time considered to brake is the sum of the true reaction time τ and the safety reaction time θ. This space limitation on braking can be expressed as

$$x_{n-1}(t) - \frac{v_{n-1}(t)^2}{2b_{n-1} - s_{n-1}} \geq x_n(t) + [v_n(t) + v_n(t+\tau)]\frac{\tau}{2} + v_n(t+\tau)\theta - \frac{v_n(t+\tau)^2}{2b_n} \tag{4.27}$$

In this inequality all parameters can be derived from direct observation except for the b_{n-1} that should be replaced by some estimate b. this last inequality can be

rewritten as

$$v_n(t+\tau) \leq b_n\tau + \sqrt{(b^2\tau^2 - b_n(2[x_{n-1}(t) - s_{n-1} - x_n(t)] - v_n(t)\tau - \frac{v_{n-1}(t)^2}{b}} \tag{4.28}$$

The speed of n vehicle is limited by the minimum of two terms : the first one is bounded to the user desired speed according to limited acceleration of the vehicle , the second one , as mentioned before, express the speed to maintain to do avoid accident if the preceding vehicle brakes with this maximum capability. We can therefore express the speed of the n vehicle like

$$\begin{aligned} &v_n(t+\tau) = min\{v_n(t) + 2.5a_n\tau(1 - \tfrac{v_n(t)}{V_n})(0.025 + \tfrac{v_n(t)}{V_n})^{1/2} \\ &, b_n\tau + \sqrt{(b_n^2\tau^2 - b_n[2[x_{n-1}(t) - s_{n-1} - x_n(t)] - v_n(t)\tau - \tfrac{v_{n-1}(t)^2}{b}]} \end{aligned} \tag{4.29}$$

In this way the transition between two states of V_n occurs smoothly. The only occurrence in which the transition is not smooth is when a new vehicle moves into the gap between two vehicles or when the leading vehicle stops in a rough way. Obviously the vehicle position will be calculated as :

$$x_n(t+\tau) = x_n(t) + v_n(t+\tau) * \tau \tag{4.30}$$

In the end the mobility module is responsible of computing the new position of a vehicle in terms of speed, acceleration and travelled distance.

Map module

The map module is responsible of parsing the OSM file representing a map to convert it in a graph representation that is used to run the simulation. Unfortunately the open street map service does not provide information about the altitude so in order to have also information about the altitude of points an interface to the Google Maps Api have been implemented. Due to the policies of the google service to gather the necessary information request caching was necessary. After gathering altitude information the slope coefficient of each street/edge is computed and then saved as an attribute of the edge in the graph representation. In this way it possible to integrate the information on the OSM map database with altitude with less effort in an automated way. At each simulation step calculate the position of a vehicle inside the map graph thanks to the information received by the mobility module

Consumption module

The consumption module purpose is to compute at each time step of the simulation the consumption derived from the movements of vehicle inside the city. To do so the module need to know the street in which the vehicle is moving querying the position of the vehicle to the map module. the relative consumption due to the slope inside the equations 4.18 and 4.19. The work necessary to win the air drag is not influenced by the street gradient as can be seen in 4.20. In each time step the consumption is

computed and in the case of electric vehicle it is subtracted from the residual battery capacity of the vehicle. If the acceleration is negative and the vehicle is slowing down also a term of regenerated energy is added to the battery.

Slope module

The slope module is a simple module that return the street gradient needed to compute the consumption given the position of the map. It access the internal graph representation of the map and returns the slope attribute of the edge that is representing the street on which the vehicle is moving. The sequence call of each module is shown in 4.8

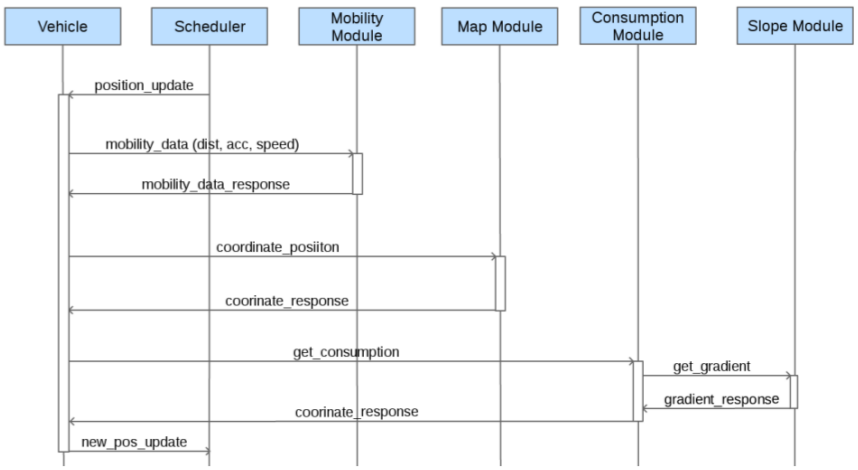

FIGURE 4.8: Sequence call to each module

4.5 SUMO enhancements

Current vehicle generation has already many sensors on board to ease or automate some maneuvers. Some of the applications that are already available are for example assisted parking and Adaptive Cruise Control (ACC). The more the applications become autonomous the more the perception of the environment from the vehicle must improve. For these reasons in the next vehicle generation the number of sensors will increase as well as the computational capability of the on-board computer. Sensors and computational units have an energy consumption that depends on the application but for sure it cannot be ignored anymore. In an autonomous vehicle scenario these considerations will be more valid especially if the considered vehicles are not fuel propelled but use electric energy stored in batteries. The SUMO simulator already provide a model of electric cars but it doesn't take into account the consumption of possible equipped sensors. To have a way more realistic simulation about consumption of electric vehicles with equipped sensors a specific device of SUMO has been implemented to add in simulations the consumption of sensors and

computational unit. This device called *device.sensors* is interfaced with the already existing *device.battery* subtracting the energy consumed by sensors to the residual energy capacity of the vehicle battery.

4.5.1 Sensors Device Model

Following the directives of the SUMO wiki a device can be considered as a container of data which resides in individual vehicles. Each device is notified about all the movements of vehicle and can interact with the vehicle or other device equipped on board. Some devices like the sensors one need to write output. SUMO provide a basic class called *MSDevice* to extend and a empty class ready to be used out of the box called *MSDevice_Example*. This base class shown in 4.9 is structured as a set of methods that need to be implemented while in 4.10 the class reference to other SUMO classes is shown.

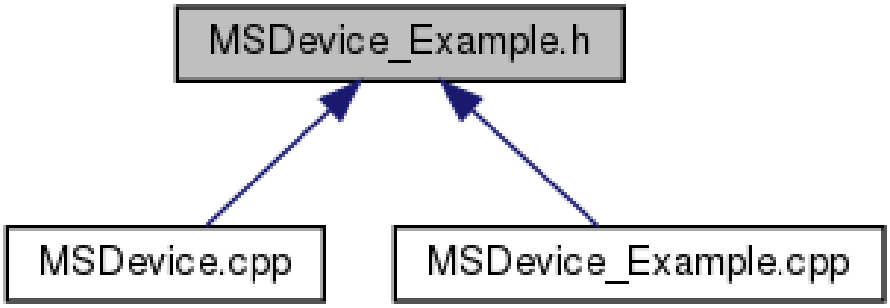

FIGURE 4.9: MS Device class

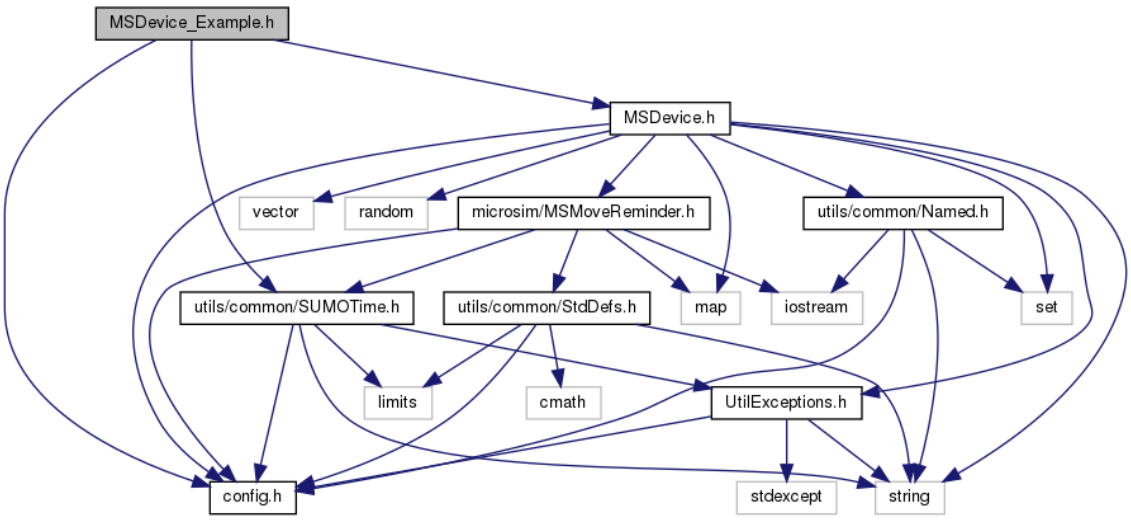

FIGURE 4.10: MS Device class reference

From the file reference it's clear that each device needs the *MSMoveReminder* class to be notified of each movement of the vehicle with a callback method called *notifyMove*. Other important methods that needs to be implemente in order to work within the simulator are the setup and configuration method that are used by SUMO when needs to instantiate and configure a device within a vehicle. The *buildVehicleDevices* method is used by sumo to create the device parsing the options that have been configured inside the configuration file of the vehicle. After the parsing phase it instantiate the device through the constructor and assign it to the vehicle. The *notifyEnter* method is a callback that is called when the vehicle with this equipped device enters the simulation scenario while the *notifyLeave* is called when the vehicle leaves the simulation scenario. The two method *getParameter* and *setParameter* are used to retrieve or modify at runtime a parameter of a device. An example of the signatures of the methods is shown in the header file of *MSDeviceExample*.

```
class MSDevice_Example : public MSVehicleDevice {
public:
    static void insertOptions(OptionsCont& oc);
    static void buildVehicleDevices(SUMOVehicle& v,
                                    std::vector<MSVehicleDevice*>& into);
public:
    ~MSDevice_Example();
    bool notifyMove(SUMOVehicle& veh, double oldPos,
                    double newPos, double newSpeed);
    bool notifyEnter(SUMOVehicle& veh, MSMoveReminder::Notification reason,
                     const MSLane* enteredLane = 0);
    bool notifyLeave(SUMOVehicle& veh, double lastPos,
                     MSMoveReminder::Notification reason,
                     const MSLane* enteredLane = 0);
    const std::string deviceName() const {
        return "example";
    }
    std::string getParameter(const std::string& key) const;
    void setParameter(const std::string& key, const std::string& value);
    void generateOutput() const;
private:
    MSDevice_Example(SUMOVehicle& holder, const std::string& id,
                     double customValue1, double customValue2,
                     double customValue3);
private:
    // private state members of the Example device
    double myCustomValue1;
    double myCustomValue2;
    double myCustomValue3;
private:
    MSDevice_Example(const MSDevice_Example&);
    MSDevice_Example& operator=(const MSDevice_Example&);
};
```

Following the directives of implementing a new device in SUMO a new class called *MSDevice_Sensors* has been implemented to add the consumption of sensors at each

time-step during the movement of the vehicle. The configuration of the device it is performed during the setup step and the parameters are read from the configuration file of vehicles as recommended. The sensors that can be equipped on a vehicle and of which the energy consumpiont contribute will be considered are :

- Cameras
- Lidar/Radar
- GPS

The data generated from these sensors to perceive the surrounding vehicle environment must be elaborated in a real time way from the vehicle in order to manage different application. The energy consumption of the computing unit is also considered and can be configured in the same way of the sensors. The overall configuration of this device can be done through a vehicle type definition in SUMO or directly inside a specific vehicle. An example of vehicle type configuration is shown just below.

```
<vType id="tipo1" accel="2.6" decel="4.5" sigma="0.5" length="2.5" minGap="2.5" maxSpeed="1
    <param key="maximumBatteryCapacity" value="30000"/>
      <param key="maximumPower" value="30000"/>
      <param key="vehicleMass" value="10000"/>
      <param key="frontSurfaceArea" value="5"/>
      <param key="airDragCoefficient" value="0.6"/>
      <param key="internalMomentOfInertia" value="0.01"/>
      <param key="radialDragCoefficient" value="0.5"/>
      <param key="rollDragCoefficient" value="0.01"/>
      <param key="constantPowerIntake" value="100"/>
      <param key="propulsionEfficiency" value="0.8"/>
      <param key="recuperationEfficiency" value="0.4"/>
      <param key="stoppingTreshold" value="0.1"/>
      <!--Sensors Parameters -->
      <param key="has.sensors.device" value="true"/>
      <param key="cameraNumber" value="2"/>
      <param key="cameraConsumption" value="2.5"/>
      <param key="lidarNumber" value="1"/>
      <param key="lidarConsumption" value="60.0"/>
      <param key="cpuNumber" value="2"/>
      <param key="cpuConsumption" value="350.0"/>
      <param key ="gpsNumber" value ="1"/>
      <param key ="gpsConsumption" value = "20.0"/>
  </vType>
```

In the first part of the configuration all the parameters of the battery device are defined. The battery device parameters define both the characteristics of the vehicle and of the electric engine besides the characteristics of the battery itself. Since the sensors device is strictly engaged with the battery device int REFERENCE the meaning of both device parameters are explained. At each timestep the sensors device compute the consumption from the vehicle equipped sensors, log it in a output file and subtract it to the battery remaining autonomy.

TABLE 4.7: Table of Battery and Sensors Parameters

Battery

Parameter	**Description**
maximumBatteryCapacity	Maximum battery capacity E_{max}
maximumPower	Maximum power which the vehicle can reach
vehicleMass	Vehicle mass m_{veh}
frontSurfaceArea	Front surface area A_{veh}
airDragCoefficient	Air drag coefficient c_w
internalMomentOfInertia	Mom. of inertia of int. rot. elements J_{int}
radialDragCoefficient	Radial drag coefficient c_{rad}
rollDragCoefficient	Rolling resistance coefficient c_{roll}
constantPowerIntake	Avg. (constant) power of consumers P_{const}
propulsionEfficiency	Drive efficiency η_{prop}
recuperationEfficiency	Recuperation efficiency η_{recup}
stoppingTreshold	Minimum velocity to start charging

Sensors

Parameter	**Description**
cameraNumber	number of cameras equipped
cameraConsumption	the consumption of one camera in Wh
lidarNumber	the number of equipped lidars
lidarConsumption	the energy consumption of one lidar Wh
gpsNumber	the number of equipped gps
gpsConsumption	the consumption of a gps in Wh
cpuNumber	the number of computing units equipped
cpuConsumption	the consumption of a singlle cpu

The sensors class extends the *MSDevice* class implementing and overriding all necessary methods to work 4.11

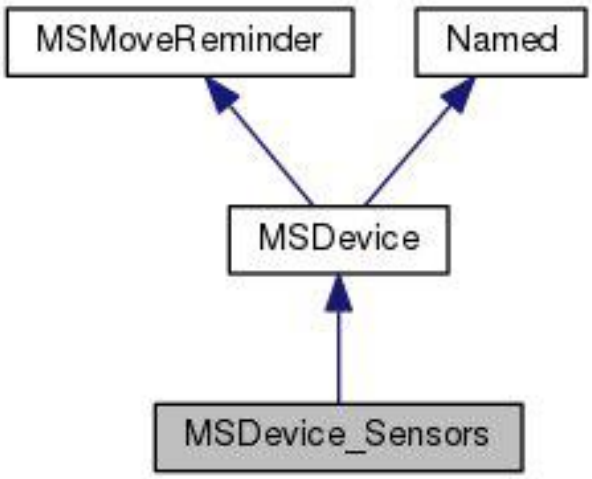

FIGURE 4.11: sensors class hierarchy

The sensors class file reference is shown in 4.12

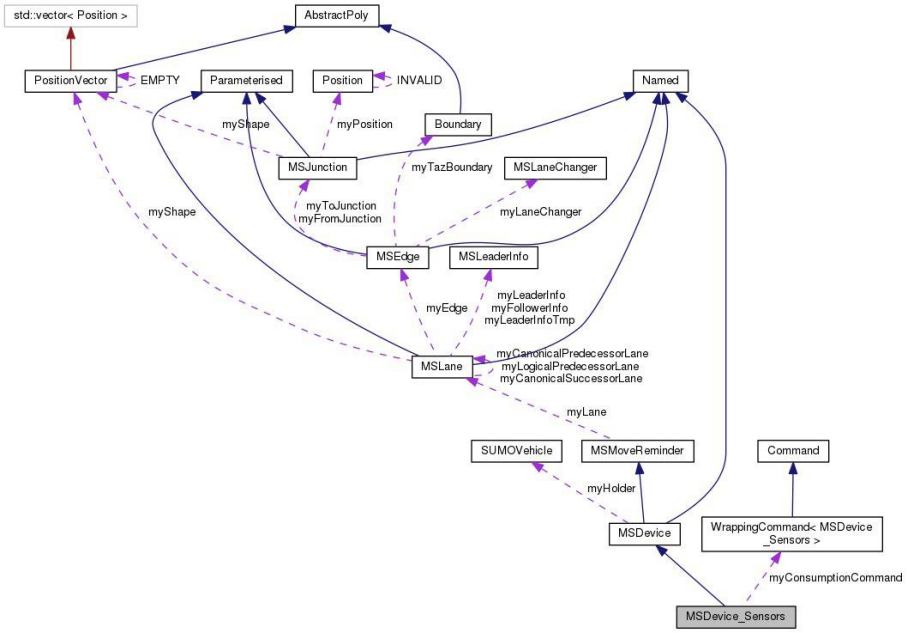

FIGURE 4.12: sensors class references

To log the consumption of sensors at each time step an export class has been implemente following the example of the *MSDevice_Battery*. The export class purpose si to retrieve the information that have to be logged and write them in an xml format in order to be analyzed in a second moment. In this class the only method that needs to be implemented is the write method . It takes as input the output device, the timestep and the precision that will be used to collect data. the output device is

an object that facilitate the write in xml format with very easy functions like open tag, close tag and write attribute. Each export class can acces the vehicle from the vehicle interface of MSNet. After controlling the presence of the interested device it is possible to append to the xml file the data retrieving directly from the device of interest. This is done at each timestep thanks to the invocation of the write method inside the *MSNet* class. During each time step a method calle *writeOutput* is called and inside this function all the device that needs output are used. The code necessary for the output is

```
// sensors dumps
    if (OptionsCont::getOptions().isSet("sensors-output")) {
            MSSensorsExport::write(
               OutputDevice::getDeviceByOption("sensors-output"),
               myStep,oc.getInt("sensors-output.precision"));
  }
```

The consumption of camera, lidars and GPS is calculated as the instant nominal power nominal power while the cpu consumption strongly depends on utilization of the computational unit. For this reason to compute it the model takes into account different factors like the speed at which the vehicle is moving, the number of obstacles that surround the vehicle and so on. An example of an output xml file of a simulation that involves battery and sensors equipped on vehicles is shown below.

```
<sensors-export>
    <timestep time="1.01">
        <vehicle id="vtypeauto.0" istantEnergyConsumed="0.279401"
        totalEnergyConsumed="0.279401" remaining_energy="14999.639093"/>
        <vehicle id="vtypeauto.1" istantEnergyConsumed="0.146074"
        totalEnergyConsumed="0.146074" remaining_energy="14999.772412"/>
        <vehicle id="vtypeauto.2" istantEnergyConsumed="0.144899"
        totalEnergyConsumed="0.144899" remaining_energy="14999.773588"/>
        <vehicle id="vtypeauto.3" istantEnergyConsumed="0.280844"
        totalEnergyConsumed="0.280844" remaining_energy="14999.637643"/>
    </timestep>
    <timestep time="2.02">
        <vehicle id="vtypeauto.0" istantEnergyConsumed="0.280824"
        totalEnergyConsumed="0.560225" remaining_energy="14991.143936"/>
        <vehicle id="vtypeauto.1" istantEnergyConsumed="0.145634"
        totalEnergyConsumed="0.291708" remaining_energy="14991.409002"/>
        <vehicle id="vtypeauto.2" istantEnergyConsumed="0.145518"
        totalEnergyConsumed="0.290417" remaining_energy="14991.409478"/>
        <vehicle id="vtypeauto.3" istantEnergyConsumed="0.281873"
        totalEnergyConsumed="0.562717" remaining_energy="14991.136660"/>
    </timestep>
    <timestep time="3.01">
        <vehicle id="vtypeauto.4" istantEnergyConsumed="0.271612"
        totalEnergyConsumed="0.271612" remaining_energy="14999.646881"/>
    </timestep>
    <timestep time="3.02">
        <vehicle id="vtypeauto.0" istantEnergyConsumed="0.279747"
        totalEnergyConsumed="0.839972" remaining_energy="14982.729196"/>
```

```
        <vehicle id="vtypeauto.1" istantEnergyConsumed="0.145910"
        totalEnergyConsumed="0.437618" remaining_energy="14983.129610"/>
        <vehicle id="vtypeauto.2" istantEnergyConsumed="0.146179"
        totalEnergyConsumed="0.436595" remaining_energy="14983.130225"/>
        <vehicle id="vtypeauto.3" istantEnergyConsumed="0.280516"
        totalEnergyConsumed="0.843233" remaining_energy="14982.723266"/>
    </timestep>
    <timestep time="3.16">
        <vehicle id="vtypeauto.5" istantEnergyConsumed="0.273776"
        totalEnergyConsumed="0.273776" remaining_energy="14999.659666"/>
    </timestep>
    <timestep time="3.31">
        <vehicle id="vtypeauto.6" istantEnergyConsumed="0.276573"
        totalEnergyConsumed="0.276573" remaining_energy="14999.659977"/>
    </timestep>
    <timestep time="4.01">
        <vehicle id="vtypeauto.4" istantEnergyConsumed="0.276072"
        totalEnergyConsumed="0.547684" remaining_energy="14991.237763"/>
    </timestep>
    <timestep time="4.03">
        <vehicle id="vtypeauto.0" istantEnergyConsumed="0.280000"
        totalEnergyConsumed="1.119972" remaining_energy="14974.220903"/>
        <vehicle id="vtypeauto.1" istantEnergyConsumed="0.147193"
        totalEnergyConsumed="0.584811" remaining_energy="14974.755492"/>
        <vehicle id="vtypeauto.2" istantEnergyConsumed="0.144608"
        totalEnergyConsumed="0.581204" remaining_energy="14974.758926"/>
        <vehicle id="vtypeauto.3" istantEnergyConsumed="0.281453"
        totalEnergyConsumed="1.124686" remaining_energy="14974.215302"/>
    </timestep>
</sensors-export>
```

It is possible to convert this file in CSV format using the python sumo tool *XML2CSV.py*.

Chapter 5

Platooning

5.1 Introduction

A platooning application allows a set of vehicle to travel along a road maintaining a small inter-vehicle gap at a constant speed. This set of cars is called platoon and is usually managed by a special vehicle, usually the first of the queue that is also called leader. There are several advantages of being a part of a platoon. For example maintaining a constant speed without sudden accelerations or decelerations allow vehicles to reduce their consumption improving autonomy and, in case of fuel propelled vehicles, lowering the CO_2 emissions. The platoon formation is also useful in terms of aerodynamics because, except for the leader, each member of the platoon face a lesser air resistance when travelling thanks to the previous vehicle. Platooning applications are very useful when considering large vehicles used to transport goods and for this reason many manufacturer companies are investing in this field. This chapter will provide a description of the tools used to simulate a platooning application together with a proposition of a protocol to enhance the already known good features of the platooning improving energy saving of equipped sensors. This study has been possible thanks to the results shown in the previous chapter and the enhancement implemented in SUMO to take into consideration the consumption of equipped sensors [78]–[80].

5.2 Simulation Tools

A platooning application can be fully considered a cooperative control scenario in which it is strictly necessary the exchange of data and information among vehicle. For this reason is necessary to exploit VANET protocols and messages to share necessary information for the coordination of all platoon members. To fully simulate the protocol stack and network dynamics the simulator presented in the previous chapter is not enough. In the next sections OMNeT++, and VEhicles In Network Simulator (VEINS) will be described. OMNeT is a simulation framework used in telecommunication to depict all the network dynamics while Veins can be considered a bridge extension between OMNeT and sumo. In the and also PLatooning EXtEnsion (PLEXE) , an extension of veins with new mobility models for implementing

Cooperatve Adaptive Cruise Control (CACC) protocol is briefly described.

5.2.1 OMNET++

The OMNeT++ discrete event simulation environment is openly available since 1997. It has been developed for the simulation of communication networks, multiprocessors and other distributed systems , but rather than creating a specialized simulator, OMNeT++ was designed to be as general as possible. Ever since then, this concept has proven right, and OMNeT++ is currently being adopted in several different domains such as :

- queuing network simulations;
- wireless and ad-hoc network simulations;
- business process simulation;
- peer-to-peer network simulations;
- optical switch and storage area network simulations;

This simulator is a C++ based discrete event simulator and, as aforementioned, is public-source. It can be used with an Academic Public License that makes it free when used in a non-profit way. It can be placed within the gap between academic, open-source tools such as Network Simulator (NS) [81] and commercial, expensive alternatives like OPNET[82]. The OMNet++ approach is a framework approach because it provides the basic tools and module to build and write very complex simulations. It integrates many external modules such as the mobility module of the INET framework [83]. its main design purposes are to enable large-scale modular and customizable simulations with visualization and easy debugging. Output and input data are open in order to generate input and process outputs with commonly available software tools. It also provide some integrated functionalities to analyze output data. Almost every object in this framework is amenable to *simple module*. These modules can be grouped together to create a *compound module* and the number of hierarchy is not limited. Modules communicate through messages. Messages can be sent both via connection between modules or directly to their destination 5.1.

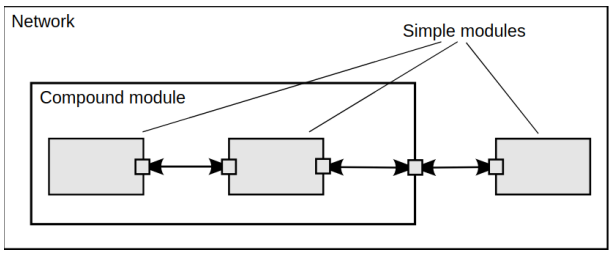

FIGURE 5.1: Example of Modulse in OMNet++

Gates are input and output interfaces of modules that can receive or send out messages. An input and output gate can be linked through a connection. Connections are created within a single level of module hierarchy. For this reason messages travel inside a chain of connection and modules. The configuration of a module can be expressed with parameters. To ease the definition of module a new description language has been introduced called NEtwork Description (NED) language. Typically in NED language there are the declaration of simple modules and the definition of compound modules and networks. The declaration consists of the enumeration and description of interfaces (gates) and the declaration on module inner parameters. A compound module definition instead consists of the declaration of external interfaces, the list of the modules of which it is composed and their interconnection. The network is a compound module that contains all the other modules. The core functionality is implemented inside simple modules. The OMNet++ framework does not distinguish between messages and events because events are implemented as messages. It is possible to implement a simple module by extending the *cSimpleModule* class and it is possible to define its behavior in two different approaches. The first approach is the *coroutine-based* approach in which the module code runs on a separated thread which receive control from the schedule when a message is received. The second one is the *event-processing approach* in which the simulator calls the given function of the module with the message as argument. The OMNeT simulator architecture is shown in Fig 5.2.

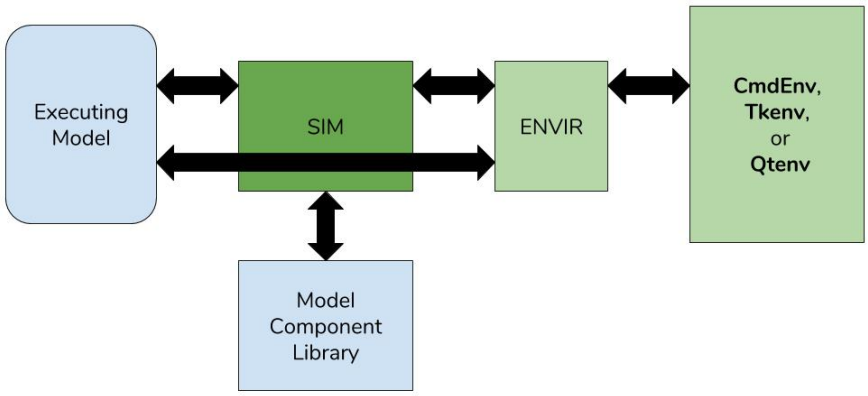

FIGURE 5.2: OMNet++ architecture

Each of these blocks represent an OMNet++ component. The *Sim* block represents the simulation kernel and class library that is linked to simulation programs. The envir block is another library that contains all the code that is shared among all user interface elements. It is a facade class that needs to be extended for embedding OMNet++ in third party application. The *Cmdendv*, *Tkenv* and *Qtenv* blocks are the actual implementation for user interface in different environments. A simulation can be linked with only one or more of these envs. The Model Component Library

block represents the entire collection of simple module definitions and their C++ implementations, compound module types, channels, networks, message types, and everything belonging to models used in the executing model. The executing model block is the model that has been set up for the simulation. This module contains all the instances of components that are needed during the simulation run. The black arrows represent the interaction between components. The simulation kernel is in a central position within the architecture because needs to communicate with several modules. It exchanges messages with the Execution model to notify the future events and to activeta module. On the contrary the Executing model calls functions in the simulation kernel and uses classes of its library. The sim block interfaces with the model component library to instantiate simple modules and other component at the beginning of the simulation run. The envir is a facade also to the executing model and it is necessary to grant access to information such as simulation result or logs. The Envir is the main actor during simulations because is the place where is locate the main loop of the simulation. It determines which models are set up and manage the Sim. The Cmdenv/Tkenv/Qtenv are concrete implementation of user interfaces and communicate with the Envir to show the simulation status and updates [84]–[86].

5.2.2 VEINS

VEINS is a simulation framework developed to couple a network and a traffic simulator. In this way it is possible to use two well-known and reliable simulator already acknowledged by the research community to model realistic communication patterns of VANET nodes. The traffic simulator is the aforementioned microscopic road traffic simulator SUMO while the network simulation is managed by the OMNet++ framework. The coupling of these two famous framework was achievable thanks to the terms of a General Public License (GPL) that allows the download and modification of the source code. In this way is is possible to achieve the best of the two framework providing useful interfaces and methods to ease the interaction between them. With a bidirectional coupling it is possible, for the network simulator, to directly control the road traffic and to simulate the influence of communication protocols and application to the urban scenario. OMNet++ is an event-based simulator that schedule events and mobility movement at regular intervals. This aspect aspects fits perfectly with the sumo simulation approach which advance simulation time in discrete steps. Both of these simulators use a command buffer to provide synchronous execution at defined time intervals. At each time step it is OMNet++ that inject all the necessary commands inside the SUMO buffer to trigger changes in the scenario. After the completion of the scheduled activities in sumo for a single step all the positions and information about vehicles in the scenario are sent back to OMNet++. Thanks to these information the network simulator is able to move, create or delete the network nodes accordingly to the position of vehicles. Then OMNet++ complete its schedule of events and send the command for advance the time

step also in SUMO. This twin chain loop is shown in Fig. 5.3. Thanks to the TraCI

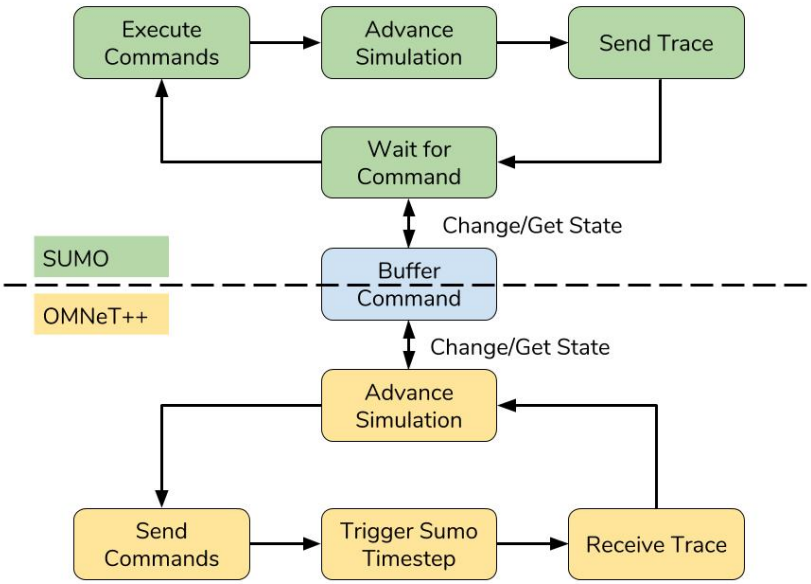

FIGURE 5.3: Veins architecture

interface from the OMNeT++ environment is possible to control the advancement in the simulation environment. The TraCI TCP server is always listening for incoming connection and to receive message in the TraCI format 4.3. Veins offers an easy and accessible implementation to the sumo function hiding the difficulty to manually compose a TraCI message thanks to the implementation of an easy interface. The message sequence of the whole VEINS environment is shown in Fig. 5.4.

This advanced coupling is possible only because the developers of sumo fostered the coupling thanks to the design of the TraCI interface. To receive information in a more detailed way it is also possible to subscribe to certain events that will be notified during their occurrence[87]. Veins main function is to ease the writing of application-specific simulation code. It can also be used out of the box but only a few parameters can be specified for a single use case. Its real purpose is to serve as an execution environment for user written code. The framework takes care modeling lower protocol layers and node mobility and of ensuring its user application execution. After the simulation run results are collected in output files. Veins contains a large number of already validated simulation models for vehicular environment [88]. Not all of them are necessary depending on the simulated application and, in fact, usually only few of therm are instantiated in one simulation. Veins simulation models can be considered as a toolbox. This toolbox contains almost what is needed to build a complex simulation scenario for vehicular networking. Obviously an end user needs to know the characteristic of every used model to tweak and customize it for a specific scenario. Veins is an Open Source project and this means

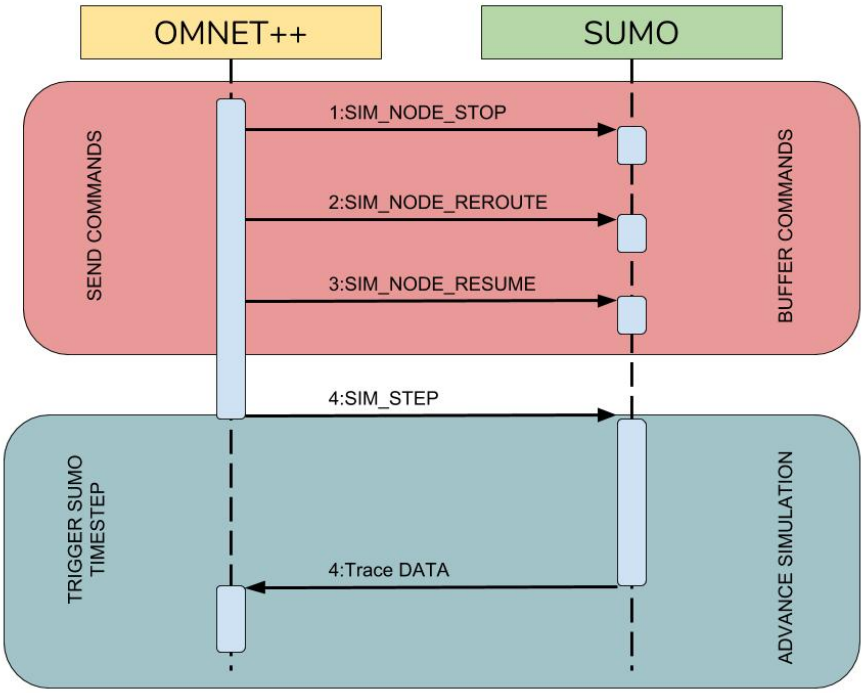

FIGURE 5.4: Veins message sequence

that the environment and all its simulation models are freely available for download modification and use. Any simulation performed with Veins can be shared with the research community because the complete tool chain required is freely shareable for any interested user. In this way is also possible to verify the obtained results and ease the debug and validation stage for each application scenario [89], [90].

5.2.3 PLEXE

glsplexe is an extension of veins to help create and simulate platooning scenarios. Platooning is an application in which autonomous vehicle cooperate using wireless communication exchanging important data about position, speed and acceleration. In this way these vehicles are capable of reducing the gap between them forming groups that are called platoons. This group organization of vehicles can improve the usage of the city infrastructure reducing pollution and increasing safety to enhance overall quality of life in urban environments. The platooning application is a multi-discipline application because it involves several research fields such as communication, control, traffic engineering and so on. To better depict and analyze these complex scenarios a tool like PLEXE has been developed using as starting point mature and well known simulators like SUMO , OMNet++ and VEINS [91]–[93]. The core of the platooning application is represented by the Cruise Control. The cruise control used in autonomous vehicles for platooning is the CACC that is an extension of ACC that exploit wireless communication among vehicles to reduce the time for

the formation assessment. The way the CACC works is really simple: it controls longitudinally each vehicle by computing a desired acceleration u that then is applied to maintain a specific inter-vehicle gap. This decision is not only based on the data gathered from internal sensors but also on the data exchanged with other vehicles. The acceleration computed from the CACC is sent to the lower modules that manage the mechanical parts of the vehicle. It is important to note that the engine and the braking system requires some time to apply the control . The cruise control can be modeled as a function $u = C(x, r)$. In this function u is the desired acceleration while x is a set of variables representing the state of the vehicle and r a set of reference variables. The easiest type of cruise control is the maintaining of a goal speed during the travel. This objective can be achieved with a very simple control system

$$u = C\left(\boldsymbol{x} = \{\dot{x}\}, \boldsymbol{r} = \{\dot{x}_{\text{des}}\}\right) = k_p\left(\dot{x}_{\text{des}} - \dot{x}\right) \tag{5.1}$$

With this equation we just express de desired acceleration u as the difference between the desired speed $\dot{x}_{\text{des}}$ and the actual speed $\dot{x}$. The difference is multiplied by a gain k_p that determines the speed of the pursued goal. The CACC is based on the same principle but the variables that are used for tweaking the speed and acceleration pursue are more and are gathered through a message exchange among vehicles. In control theory the equation that is used to represent the real system is called the plant. The plant here is the function $\ddot{x} = P(u, x)$. A possible implementation of the plant can be

$$\ddot{x}_k = P\left(u, x = \{\ddot{x}_{k-1}\}\right) = \alpha u + (1 - \alpha)\ddot{x}_{k-1} \tag{5.2}$$

$$\alpha = \frac{\Delta_t}{\tau + \Delta_t} \tag{5.3}$$

Equation 5.2 is used to compute the acceleration at each time step k using the desired acceleration computed by the controller at the previous time step $k-1$. Equation 5.3 regulates the sampling time Δ_t, τ is the engine time constant in seconds. The PLEXE framework is built on top fo the VEINS framework by extending both SUMO and OMNet++ as shown in 5.5

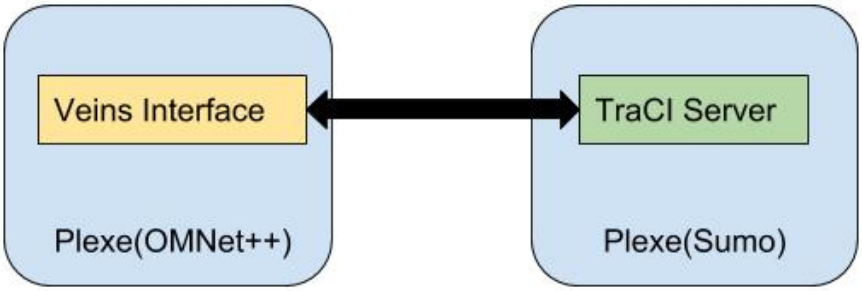

FIGURE 5.5: Plexe Architecture

Obviously OMNet++ is used to simulate the communication among vehicles while sumo takes care of providing a realistic mobility simulation. The car-following

models provided by sumo are not suitable for the simulation of platooning applications and for this reason a new model has been added. To do so important information have been added to the *VehicleVariables* class that are used in the $CC_VehicleVariables$ class that stores the controller , vehicle state and other important information. The true car following model is implemented inside the $MSCFModel_CC$ class. The model provided contains also the default Krauss car following models that is used if no platoon is available [94]. The main problem with the sumo simulator is that SUMO is oriented to microscopic simulations. This means that the level of details described for each vehicle is not high enough to have a good representation of the dynamics of engines. To do so a sub-microscopic simulation is necessary and PLEXE implemented some tweaks over sumo to tackle this issue. For this reason a realistic engine model is provided by the PLEXE extension with the class *GenericEngineModel*. This class has a method called *getRealAcceleration* that compute given some parameter such as current speed, current acceleration and desired acceleration the value for the acceleration in the next time step. To allow a full compliance with the SUMO TraCI server model some modification were necessary in both side of VEINS. This means that both TraCI command interface class and the TraCI TCP request have been implemented. To avoid an excessive modification inside the core of SUMO two generic purpose call have been widely used :

- $setGenericInformation$
- $getGenericInformation$

In addition to this new car following model also a lane change model have been introduced to override the default behavior. In this way it is possible to specify with a parameter the desired lane for the platooning. The logic of the lane change model is implemented inside the *MSCACCLaneChanger* which extends the basic lane change class *MSLaneChanger*. When the network is setting up each edge instantiate the custom lane changer instead of the default. In this way more parameters and methods are available to manage the lane of the platoon. With the $setFixed - Lane$ method it is possible to specify the target lane of the platoon that will be occupied as soon as possible always respecting the safety feasibility rules. The *setLaneChangeAction* is a callback to specify the behavior of a vehicle when a change lane event occurs. With constants like $DRIVER\_CHOICE$ or $STAY\_IN\_CURRENT\_LANE$ it is possible to modify the behavior of the platoon car at runtime [95]. The coupling between SUMO and OMNet is achieved by the creation of a network node for each vehicle in the mobility model. Each network node is equipped with a full protocol stack for VANET including IEEE 802.11p wireless interface and a beaconing protocol. Each time a vehicle move in OMNet the position update is replicated. The coupling is implemented through the aforementioned TraCI interface that allows a bi-directional communication. In order to develop a working platooning application some modification and the addition of new elements have been performed in both SUMO and OMNet. In SUMO the main change can be considered the implementation of a new

car-following model managed by cruise control. These car controls manage both longitudinal control based on open or closed loop control for parameters like acceleration and speed . It can also manage transversal control to appropriately change lanes if necessary in platoon dynamics. The module in SUMO that manages the driving action is called Cruise Control (CC) and it is based on the Krauss mobility model. PLEXE to enable a platooning application add two other models: the ACC model and the CACC model [96]. The CACC model to work needs to be fed with information about each vehicle inside the platoon and for this reason must be coupled with a protocol that provides these information through message exchange. For this reason also new modules and features have been added on the OMNet side in VEINS. The main addition is the implementation of a basic protocol for message dissemination, and the addition of an application directly on top of the message distribution. The PLEXE framework provide a general base protocol and a general application that can be customized or extended depending on the necessities. The general architecture of the PLEXE framework is shown in Fig. 5.6.

Platooning Control

The goal for the control over a platoon of vehicles is to regulate speed and relative distance between each vehicle and its predecessor [97], [98]. A platoon can be considered as a set of vehicles plus a leading one that is considered as a reference for the ensemble. Usually each vehicle is equipped with on-board sensors and GPS in order to obtain information about absolute position, speed and distance. The vehicles that compose the platoon are also equipped with IEEE 802.11p radio interfaces to allow a message exchange among vehicles. Each vehicle can be described with the inertial agent [99].

$$\begin{aligned} \dot{r}_i(t) &= v_i(t) \\ \dot{v}_i(t) &= \tfrac{1}{M_i} u_i(t) \end{aligned} \tag{5.4}$$

in 5.4 r_i[m] and v_i[m/s] are the absolute position and speed, M_i [kg] is the mass and u_i is the propelling force of the $i-th$ vehicle. This modeling approach is not new because there have been other approach in interconnected vehicle literature similar to this one. The goal speed of the entire platoon is decided by the leader so at steady state the equation 5.5 must be full filled.

$$\begin{aligned} \dot{r}_0(t) &= v_0 \\ \dot{v}_0 &= 0 \end{aligned} \tag{5.5}$$

In these terms the problem of maintaining a desired speed and an acceptable gap between vehicles can be rewritten as

$$\begin{aligned} r_i(t) &\to \tfrac{1}{\Delta_i} \left\{ \textstyle\sum_{j=0}^{N} a_{ij} \cdot \left(r_j(t) + d_{ij} \right) \right\} \\ v_i(t) &\to v_0 \end{aligned} \tag{5.6}$$

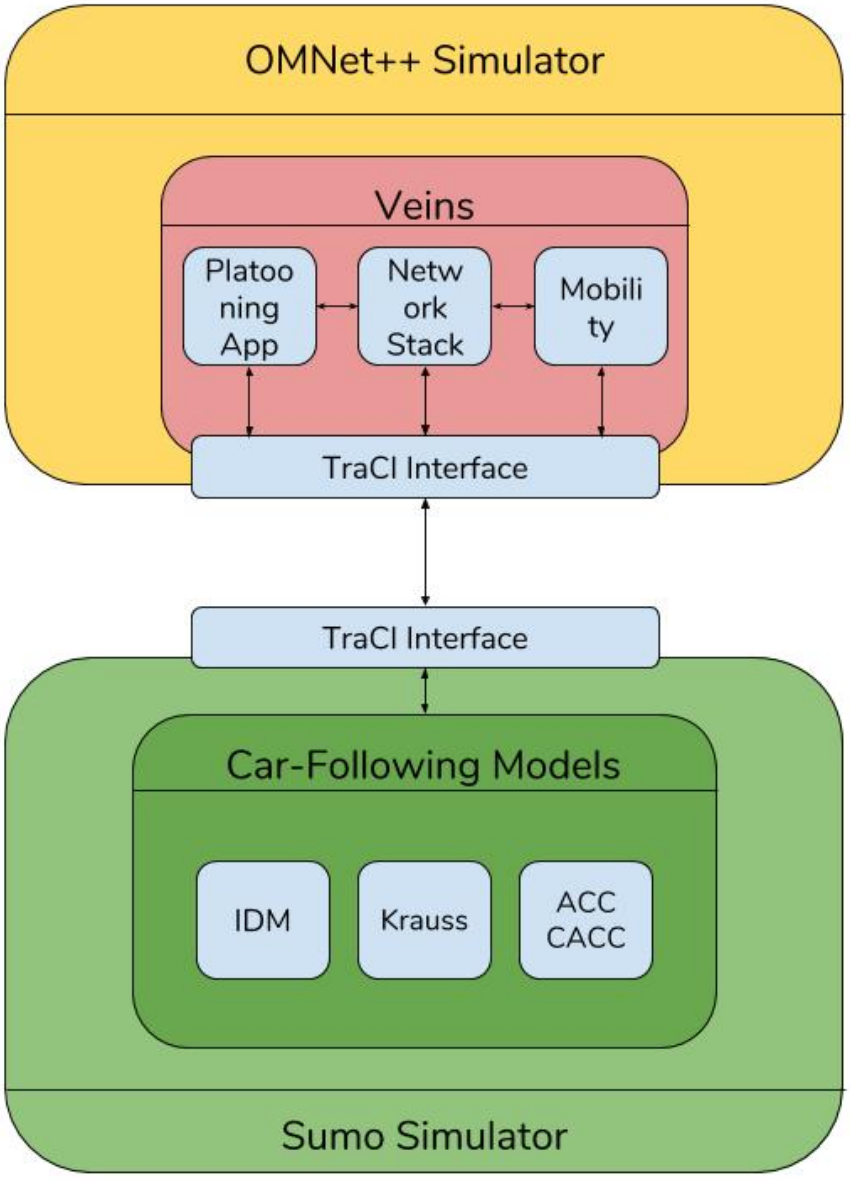

FIGURE 5.6: Intercation between modules in Plexe

Herein d_{ij} is the desired distance between vehicles i and j; a is a term needed to module the topology of the vehicular network representing the presence or absence of a link between two vehicles and the term Δ_i is $\Delta_i = \sum_{j=0}^{N} a_{ij}$ and represent the degree of connectivity of a vehicle. The distance d_{ij} between two vehicles can be expressed as $d_{ij} = h_{ij}v_0 + d_{ij}^{st}$ where h_{ij} is the term that represents the constant time headway. This time is the time necessary from a vehicle to travel the distance to reach its predecessor while d_{ij}^{st} is the distance between vehicles at standstill. The consensus problem can be managed only taking into consideration also the communication delays between vehicles.

$$u_i = -b\left[v_i(t) - v_0\right] - \frac{1}{\Delta_i}\sum_{j=0}^{N} k_{ij}a_{ij}[r_i(t) - r_j\left(t - \tau_{ij}(t)\right) - \tau_{ij}(t)v_0 - h_{ij}v_0 - d_{ij}^{st}] \quad (5.7)$$

In 5.7 k_{ij} and b are both control gains to regulate the behavior and must be both major than zero. The time variant communication delays are represented by the terms $\tau_{ij}(t)$ and $\tau_{i0}(t)$. They are correspondingly referred to the i-th vehicle when the information is generated and transmitter by vehicle j and the time delay when is the leader the source of the transmission. The synchronization of the whole platoon is ensured by timestamp included inside every message and generated from the GPS [100]. When talking about platoon applications the referral environment is often highways or multi-lane streets that do not have a very accentuated curvature. For this reason only longitudinal control is taken into account. According to newtown laws

$$m\ddot{x} = F + F_g + F_{\text{aero}} + F_{\text{drag}} \quad (5.8)$$

$$m\ddot{x} = F - mg\sin(\theta) - \frac{\rho A C_d}{2}\dot{x}^2\,\text{sgn}(\dot{x}) - d_m \quad (5.9)$$

In a one way street we can safely assume that all the vehicles are going towards the same direction so $\text{sgn}(\dot{x}) = 1$ The engine is modeled with the first degree equation:

$$\tau\dot{F} = -F + u \quad (5.10)$$

The model of the entire vehicle can be represented in Fig. 5.7.

Each used parameter is explained in 5.1

5.3 Platooning Protocol

The proposal of this platooning protocol aims to ease the formation of platoons of vehicles in order to improve the condition of the members that belongs to the platoon. Usually a platoon formation is composed of one leader and a set of members that are managed from the leader. In this proposal it is possible to find 4 different roles:

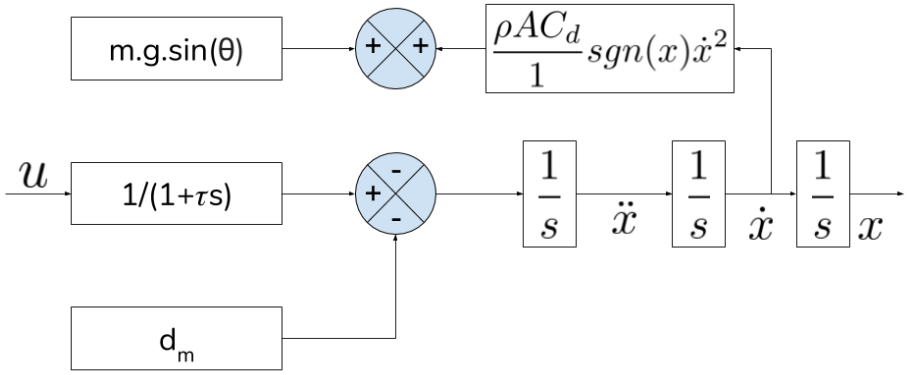

FIGURE 5.7: Dynamic system prepresentation

TABLE 5.1: Table of Symbols

Symbol Name	Description
m	Mass of the vehicle
x	Position of hte vehicle along the X axis
F	Force produced by the vehicle engine
F_g	Gravitational Force
F_{aero}	Aero-dynamical drag force
F_{drag}	Mechanical Drag Force
g	Gravity acceleration
θ	angle between the road surface and horizontal plane
ρ	Mass of Air
A	Cross Sectional Area
C_d	drag coefficient of the vehicle
d_m	Amplitude of the mechanical force Coefficient
τ	Time constant of the engine of the vehicle
u	Control input of the engine

- The leader - It is the vehicle responsible of guiding the platoon, manages the request of join of external vehicle and also the leave request of a platoon member. it also :
  - Send periodically a recruiting beacon. The recruiting beacon is a message that contains the basic information about the platoon. When a vehicle receive a recruiting beacon it evaluate the information packed inside the beacon to decide if it is suitable to join the platoon
  - Relay all recruiting beacon of other near leaders. Obviously the flooding of this messages is limited with a small time to live.
  - it has to accept and analyze all joining requests. If the member is suitable and all the conditions are met the leader will take care of all the joining phase sending the joining response to the requesting member. A set of position messages will be sent to the joining member until it reaches its platoon position

  - it has to accept and manage the leave request of a platoon member. It also have to verify if the condition for the leaving of a member are met.
  - It will supervise the election of a new leader. If a join or a leave maneuver is in progress the leader cannot leave its role and will postpone the role change through a nack message.

- Manager - The manager is the vehicle that has the responsibility to manage the internal dynamics of the platoon checking periodically the status of each platoon member end setting up the elections in a periodical manner. Each election will be independent from the previous and will be based only on the current state of each member. The election for changing a vehicle role has been introduced for a fairness policy that will be shown in the next sections.

- Member: It is a member of the platoon and for this reason it does not possess active roles in the platoon dynamics. The only task that it has to accomplish is to send periodically its state to the manager of the platoon. After an election a member cannot refuse a new role. to accept the role it will:

  - Send message to the leader or manager. To accept it will send a positive ack and it will became instantly the new leader or manager while if it cannot become for some reason the new leader or manager a nack will be sent and the member will have to wait for another election result message.

- Lone member - the lone member is a vehicle that does not belong to a platoon and it cannot join a platoon without requesting the authorization. When receiving a recruiting beacon it can ask the leader to join the platoon if the condition are favorable. The conditions that a lone member considers are :

  - Feasibility of the join maneuver: This factor takes into consideration the minimum distance between the platoon and the speed variation necessary to reach it. The joining vehicle will have to compute a joining point that belongs to the intersection of the two paths. An example of join maneuver is shown in 5.8
  - Common interest: the reach of a destination area that will be shared between the platoon and the lone member
  - Energy saving: the lone member can save some energy from the equipped sensors if it will act as some specific roles.

In this particular case the only the leader and the manager keep the sensors fully active while the rest of the platoon members

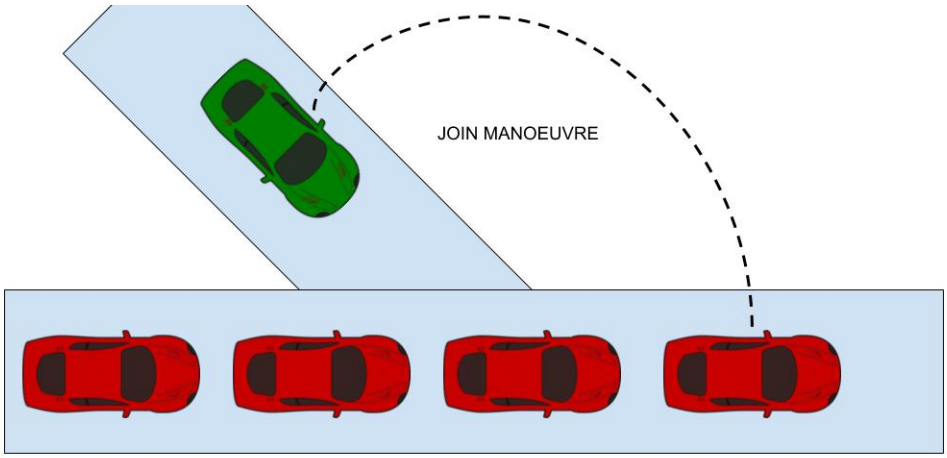

FIGURE 5.8: Example of join meuveur

5.4 Implementation

Making advantage of the PLEXE framework the protocol and the testing scenarios have been implemented. There are several classes that can be used to ease the implementation of a platooning application:

- maneuver
- scenario
- Application
- protocol

As is suggested from the name the maneuver class is a class that can be extended for implementing a platoon maneuver such as the join of the platoon or the leave. The maneuver class can be considered just as a container with all the pointer to the modules that are necessary to orchestrate the maneuver. this class also contains the *onManeuverMessage* callback that is called when a maneuveur message is received by the vehicle. The pointer to the modules are :

```
protected:
    GeneralPlatooningApp* app;
    BasePositionHelper* positionHelper;
    Veins::TraCIMobility* mobility;
    Veins::TraCICommandInterface* traci;
    Veins::TraCICommandInterface::Vehicle* traciVehicle;
```

The *GeneralPlatooningApp* is the platooning application at the top of the layers and manages everything. The *GeneralPlatooningApp* extends the base app and contains the method to begin the various implemented maneuver, a callback for manage a maneuver message handling and for the platoon beacons used to manage the platoon formation. The *BasePositionHelper* is a fundamental class that is necessary to coordinate the members of the platooning or the vehicles that are joining it. Among all methods the data structure and methods to maintain the formation are used to detect the role of each vehicle and the relative position inside the platoon. The other three components of veins are the access interface to TraCI for mobility , command interface and vehicle command interface. The scenario it's a useful utility class that allows the orchestration of a simulation scenario that comprehend various maneuvers and events to depict in a realistic way real world situation. Using this class it is possible to test and debug single or sequential events to detect errors or critical configurations for the application. The protocol class can be used do model different protocols grouping methods and policies in a single class that can be assigned to a layer of the stack. The provided default protocol class is the *UnicastProtocol* that provides the behavior of the WAVE protocol with unicast and broadcast message sending. The classes hierarchy is shown in 5.9

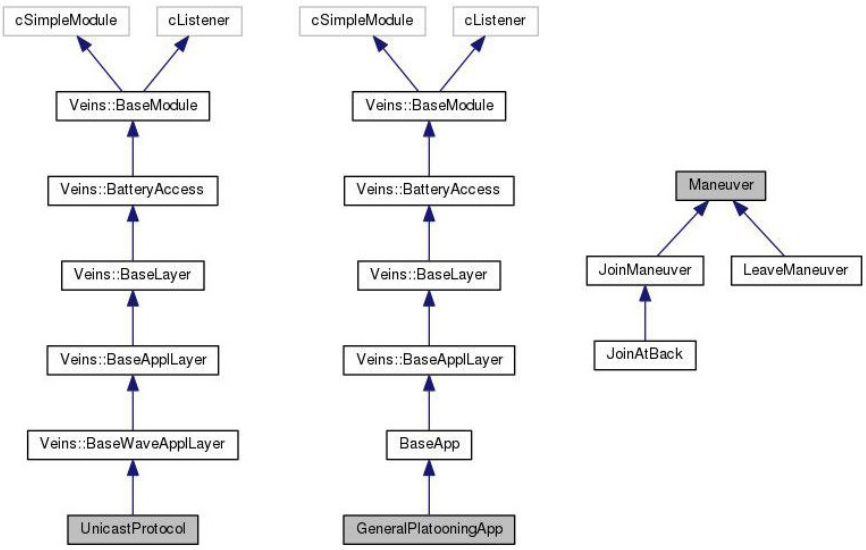

FIGURE 5.9: Plexe classes hierarchy

To send the recruiting beacon message from the leader it has been scheduled a periodic self message event to send the recruiting beacon. Beacon are received from every vehicle that is in transmission range but are processed only by lone members that are searching for the platoon. If a lone member decide to take part of the platoon it send the join request message. To the test the protocol some scenario have been

implemented to evaluate the successful end of maneuvers. The two basic maneuvers that we will firstly explain are

- Join Maneuver
- Leave Maneuver

5.4.1 Join Maneuver

The join maneuver starts when a member that is not part of the platooning formation wants to join it after receiving a recruiting beacon from the leader. In order to do so it sends a JoinRequest message to the leader and wait for a JoinResponse. The leader accept the request only if it is not engaged in other maneuvers otherwise it will answer with a negative response. When the current maneuver in which the leader engaged ends and another request is received it will answer with a positive join response. During all the joining maneuver the joining vehicle is managed by the leader that will send information about the position continuosly in order to ease the platooning approaching maneuver. When the joining vehicle is finally part of the platoon formation sends a joinFormation message to the leader that will answer with a joinFormationAck. After this message exchange the vehicle is formally part of the platoon formation and its speed and acceleration is handled by the leader through the platoonbeacon messages. A schema of the joining maneuver is shown in 5.11

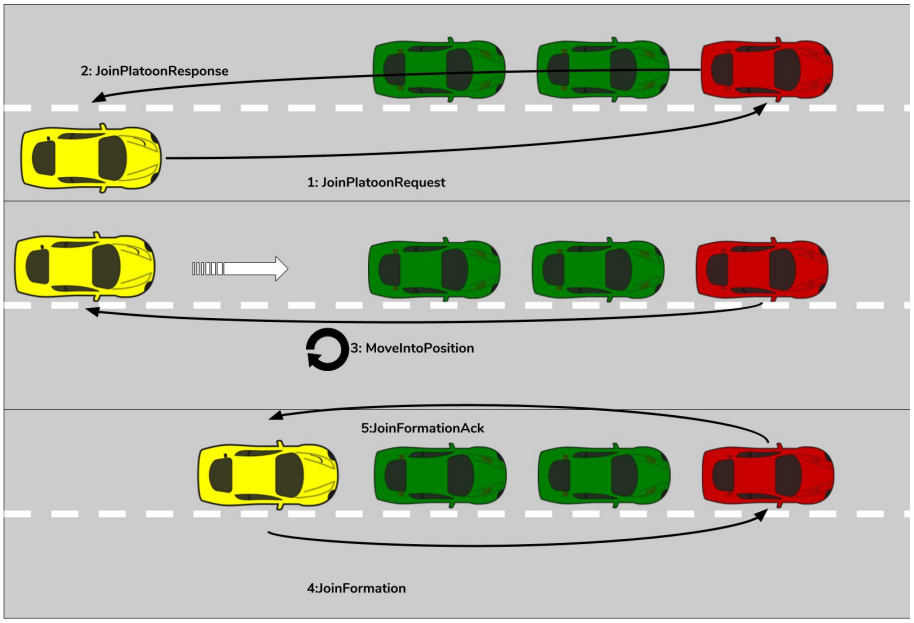

FIGURE 5.10: Join maneuver schema

5.4.2 Leave Maneuver

The leave maneuver of a platooning member is simpler than the joining one. The member express is will to leave the platoon formation through a leave request that is directly send to the platoon leader. If the platoon leader is not involved in another maneuver and the safety leave conditions are met the leader answer with a positive join response. Inside the positive leave response there is also specified the new lane that the leaving vehicle have to occupy to move away from the platoon formation. After moving onto the leaving lane the leaver will send a leaveFormation to the leader and disengage the formation. The maneuver schema is shown in

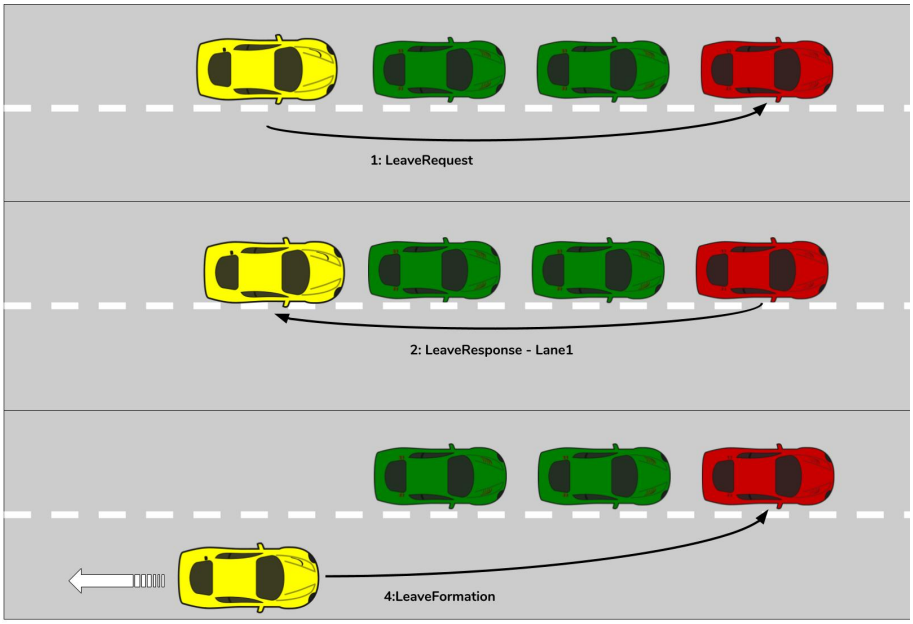

FIGURE 5.11: Leave maneuver schema

5.5 Platoon Selection

The task of distributing the vehicles in platoons trying to find the best possible assignment si a very difficult problem. Knowing the parameters of each car and each platoon it is necessary to decide if and which platoon a lone vehicle has to join on the base of several considerations. To describe this problem we decided to express it as an optimization problem. The approach used to solve it is an algorithm bio-inspired that takes inspiration from the grey wolf hunting procedure. This algorithm is herein described.

5.5.1 Grey Wolf inspired algorithm

The Grey Wolf Optimizer (GWO) algorithm is a population based algorithm inspired from the hunting behavior of the wolf pack [101]. Even if is competitive in relation

to other evolutionary algorithms its simple formulation and easy implementation made it a popular choice to resolve optimization problems. One of the fault of this algorithm is easy trapped in local optimum. This problem is usually generated from an imbalance between the exploitation and exploration phase. A brief description of the algorithm is here provided with a mathematical description of the hunting behavior. The leader of the pack is called the α wolf and is in charge of making decision about the hunting, sleeping and movement of the pack. The second in charge of the group is the β wolf and its role is to help the α leader in taking decisions. When the leader of the pack becomes too old and tired the new α leader is chosen among the β candidates. The lower level of the wolves hierarchy is represented by the ω type. This kind of wolves is the scapegoat of the pack and is able to satisfy the whole pack. The δ wolf must submit to the α and β but dominate over the ω . This kind of wolves are the scout that protect the entire pack. During hunting the δ wolves are the one that pursue and seize the pray while waiting for the whole pack for the final attack. The hierarchy of the wolves pack is shown in 5.12

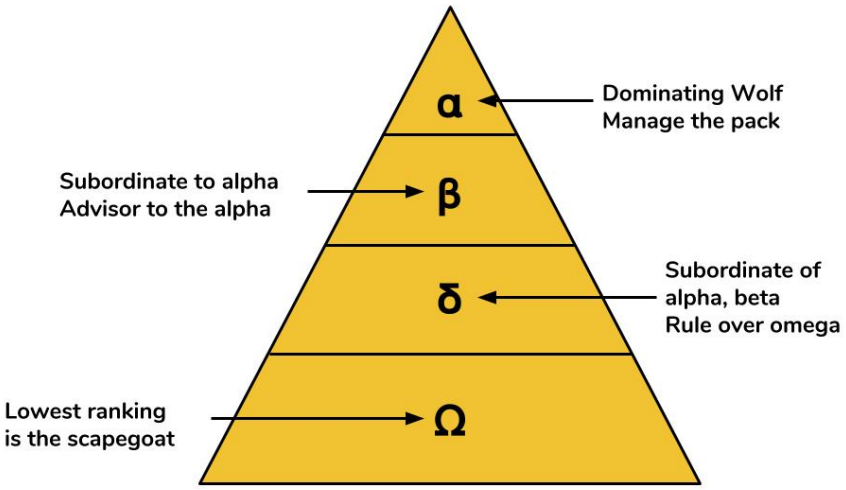

FIGURE 5.12: Wolves Hierarchy

The algorithm is composed by three steps and the surrounding behavior is modeled as

$$X_i^d(t+1) = X_p^d(t) - A_i^d \left| C_i^d X_p^d(t) - X_i^d(t) \right| \tag{5.11}$$

t is the number of iterations , X_i is the vector that represent the position of a gray wolf , X_p is the vector that represents the position of the prey, $A_i^d = 2a \cdot r_1 - a$, $C_i^d = 2 \cdot r_2$, r_1 and r_2 are random vectors in $[0, 1]$, $\vec{a}$ is decreased in a linear manner

from 2 to 0. Each ω wolf must update its position regarding the α , β and δ. The update relation is represented as

$$\begin{cases} X^d_{i,\alpha}(t+1) = X^d_\alpha(t) - A^d_{i,1}\left|C^d_{i,1}X^d_\alpha(t) - X^d_i(t)\right| \\ X'_{i,\alpha}(t+1) = X^d_\beta(t) - A^d_{i,2}\left|C^d_{i,2}X^d_\beta(t) - X^d_i(t)\right| \\ X^d_{i,\delta}(t+1) = X^d_\delta(t) - A^d_{i,3}\left|C^d_{i,3}X^d_\delta(t) - X^d_i(t)\right| \end{cases} \tag{5.12}$$

$$X^d_i(t+1) = \frac{X^d_{i,\alpha}(t+1) + X^d_{i,\beta}(t+1) + X^d_{i,\delta}(t+1)}{3} \tag{5.13}$$

The initialization of the algorithm is really important because it has a great impact on the convergence speed and the final convergence precision. The position of the optimal solution inside the solution space is unknown so the positioning of the candidate must cover the space uniformly. For this reason a Good-point-set-technique is usually used to generate a population of candidate uniformly distributed inside the solution space. The exploration is the ability to move into the solutions space for searching new and better solutions while the exploiting is the ability to find the optimal near a good solution.

Each wolf is a search agent and the movement of each ω wolf or other is deeply influenced by the movement of the α, β and δ. Their role is chosen at the beginning of the initialization phase and among the whole pack. In this way the final motion of the pack is calculated by the sum of the position vectors of the three most influencing wolf kind. To better understand the movements in the solution space of the pack the 5.13 is here represented.

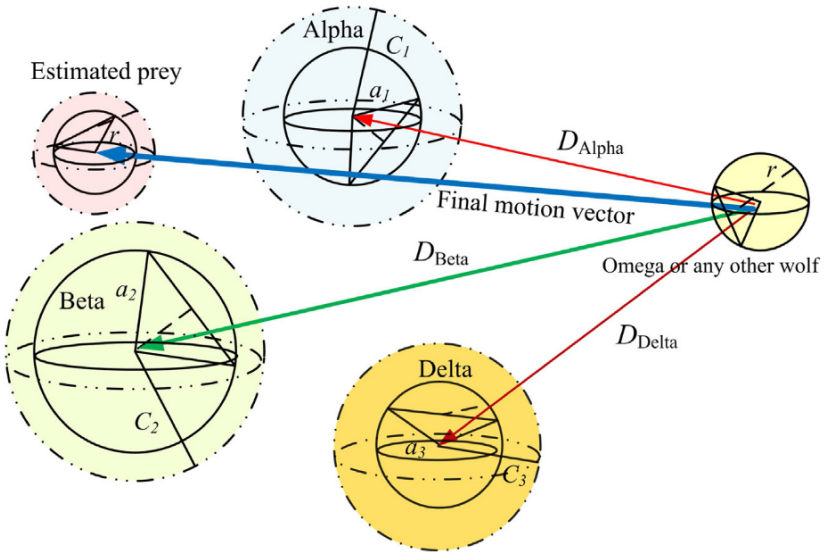

FIGURE 5.13: Hunting strategy movement

The position of the pack is based on the results of the three most important wolves and is calculated using 5.13.

As said the algorithm is very simple and consist of few simple steps. The algorithm is very simple and after the initialization steps a simple cycle is used to update the position of the pack through the update of A, a and C. Each wolf can search inside and hyper sphere with the wolf as its center and can enlarge or diminish its searching range in relation to the absolute value of A. At the end of the procedure the α agent is returned as the solution [102].

Algorithm 1: Wolf Hunting Schema

```
initialize array of wolves X_i;
initialize A,a and C;
generate random population of wolves;
calculate the objective of population;
Set X_α as the leader;
Set X_β as the second best;
Set X_δ as the third best;
while Verified or Time ended do
    Update the position of wolves;
    Update A, a and C;
    Calculate the Objective of population;
    Update X_α, X_β and X_δ;
end
report X_α;
```

5.5.2 Problem formulation

The problem of assigning a vehicle to a platoon can be described as an optimization problem. The variable x_{ij} is 1 if the vehicle j belongs to the platoon i otherwise its value is 0; The objective function that needs to be maximize is :

$$\sum_{i=1}^{M}\sum_{j=1}^{N} inter_{ij} * [1 - \prod_{k=1}^{M}(1 - \eta_{ij} * e_{ij})^{x_{ij}}] \tag{5.14}$$

subject to the constraints

$$\begin{cases} \sum_{j=1}^{N} x_{ij} \leq q_i \quad i = 1, ..., M \\ \sum_{i=1}^{M} x_{ij} = 1 \quad j = 1, ..., N \\ x_{ij} \in \mathbb{N} \\ 0 \leq x_{ij} \leq 1 \\ x_{ij} * (1 - e_{ij}) = 0 \end{cases} \tag{5.15}$$

In the 5.14 objective function $inter_{ij}$ is the size of the intersection between the set of edges of the vehicle path and the set of edges of the platoon path.

$$interr_{ij} = |(V_j(e_1, e_2, e_3, ..., e_l) \cap P_i(e_1, e_2, e_3, ..., e_k))| \tag{5.16}$$

η_{ij} is a energy factor to model the advantage that a car j receive joining the i platoon. η can be calculated as

$$\eta_{ij} = \frac{En_j - En_i}{En_j} \tag{5.17}$$

remembering that j are the vehicles and i are the platoons. q_i is the max size of the platoon and depends on how many vehicles a leader can manage. The first constraint express that the number of vehicles inside the platoon cannot excess max limit of vehicles. The second constraint is needed to allow a vehicles to belong to exact only one platoon. Third and fourth constraints express the binary nature of the variable x_{ij} while the last contains the e term necessary to elegibility. The elegibility is needed to express if a vehicle can belong to a determined platoon or not. The proposed GWO algorithms uses the the well known hunting wokf pack inspired technique in a hybrid discrete scenario. How this assignment algorithm works is shown below. The algorithm can be divided into two main routines:

- The initialization routine;
- The pack position update;

This pseudo-code show how a first starting solution is built. This solution will be used as a starting point in the pack position update routine to move inside the solution space.

The duration of the research is determined by the parameter a and a max number of iterations that can be set. In each iteration of the research routine the a parameter is updated as:

$$a = 1 - \frac{t}{T_{max}} \tag{5.18}$$

where t is the counter of the current iteration while T_{max} is the threshold time in which the research must be finished. The constraints of the optimization problem are embedded within the code of the update and initial solution.

Algorithm 2: Pseudo code of the initialization method

Data: The platoon scenario parameters, including M,N, $Q = [q_i]_{1xM}$, $INTERR = [Inter_j]_{1xN}$, $H = [\eta_{ij}]_{MxN}$ and $E = [e_{ij}]_{MxN}$, the population size Ws

Result: The Initial Solution

```
while p<size(N_p) do
    n_p = N_p(p);
    Set initial value of the n_pth solution to a zero-vector [0]_{1xMN};
    while i<size(M) do
        Sort all platoon in descending order according to H(i,:);
        Set q_max = q_i;
        while j<size(N) do
            if q_max <= 0 then
                break;
            end
            if E(i,IN(j))==1 then
                Generate random number n_rand between 0 and 1 ;
                if n_rand > 0.5 then
                    Assign the vehicle;
                    Update q_max=q_max − 1 ;
                end
            end
            j++;
        end
        i++;
    end
    p++;
end
```

Algorithm 3: Pseudo-code of the modular position update method

Data: The platoon scenario parameters, including M,N, $Q = [q_i]_{1xM}$, $INTERR = [Inter_j]_{1xN}$, $H = [\eta_{ij}]_{MxN}$ and $E = [e_{ij}]_{MxN}$, The first three best solutions in the previous iteration $\vec{S}^{\alpha}(t)$, $\vec{S}^{\beta}(t)$, $\vec{S}^{\delta}(t)$, The control parameter a

Result: The new assignment scheme in the $(t+1)th$ iteration $\vec{s}_i(t+1)$

Generate a random number between (0,1) called *rand*;
if $rand \leq a$ **then**
    Calculate the fitness value of $\vec{S}^{\alpha}(t)$, $\vec{S}^{\beta}(t)$ and $\vec{S}^{\delta}(t)$ denoted by f_{α}, f_{β} and f_{δ} ;
    Calculate $\omega_l = f_l/(f_{\alpha} + f_{\beta} + f_{\delta}), (l = \alpha, \beta, \delta)$;
    Let *elite* = [α β δ];
    Set cum =0;
    Generate a random number between (0,1) denoted by r_0;
    while $z \leq 3$ **do**
        temp = elite(z);
        cum = cum + ω_l ;
        if $r_0 \leq cum$ **then**
            $\vec{s}_{temp}(t+1) = \vec{s^l_{temp}}(t)$;
            break;
        end
        z++ ;
    end
else
    Set $q_{max} = q_i$;
    while *j<size(N)* **do**
        if $q_{max} \leq 0$ **then**
            break ;
        end
        if *E(i,j)==1* **then**
            Generate random n_{rand} between 0 and 1 ;
            if $n_{rand} > 0.5$ **then**
                Assign the vehicle ;
                Update q_{max}=$q_{max} - 1$;
            end
        end
        j++;
    end
end

Chapter 6

Results

In this chapter some results of my works during my academic career will be shown. During the first part of the studies the focus of the research was on the connectivity among vehicles with interfaces and protocols proposition to tackle new issues relative to the high dynamicity of the environment. Vehicular networks are very unstable due to the continuous change of the topology and for this reason classic protocols and applications cannot be used. The results of some solutions proposed in this area will be described. The focus of the research then has moved on the mobility models and the simulation of vehicles inside the urban environment with particular interest on the electric mobility. During this period a new java simulator for electric vehicle has been implemented. Also an extension of the well known SUMO simulator has been made to support the energy consumption of sensors equipped on vehicles. Complex sensors are increasingly common on board of vehicles thanks to the diffusion of automated maneuver such as smart parking or collision avoidance. The future of mobility on wheels is, looking at the current situation, electric and for this reason is also important to start some consideration on the chance of fully autonomous vehicle powered by electric energy and with electric engine. In the last part of the PhD the focus has been on autonomous vehicle and in particular on the platooning application with electric powered cars.

6.1 VANET enhancement

The VANET research topic is really interesting for the research community for several years and even today. In this area several protocols and architectures are proposed to manage in a smart way the issues caused by the high dynamic topology of the network. Herein will be presented the results and conclusion of two proposed protocols. The first one is called GeoCasting Wave and its purpose was to enhance active and passive safety in urban environment, while the second one is called FSCTP and it is a protocol to exchange data in VANET environment in a fast and secure way.

	Flow 1	Flow2	Flow3	Flow4	Total
Low	0.10	0.07	0.08	0.10	0.35
Mid-Low	0.14	0.10	0.13	0.14	0.51
Mid-High	0.20	0.14	0.17	0.20	0.71
High	0.33	0.20	0.25	0.33	1.12

TABLE 6.1: Input Rate for traffic flows

6.1.1 GeoCasting Wave

The details about the GeoWave protocol have been already described in 2.2 section. Along the protocol also a network architecture was proposed, that is able to continuously gather information about the traffic flows inside the city environment. With the help of a slightly enhanced OBU and a clustering system based on the position of vehicle inside the city it is possible to send localized information only to interested vehicles. To test this protocol several simulations were performed. The used simulator is OMNet++ connected through TraCI interface to the SUMO mobility simulator. The protocol implementation was made entirely in VEINS that can be considered as a bridge framework between the previous two. The protocol was evaluated in terms of traffic management, safety goals reached and CO_2 emissions. The safety index that was used is based on the number of accidents in the city. A real map was used during all the simulation campaigns and the starting and ending points of traffic flows are showed in 6.1.

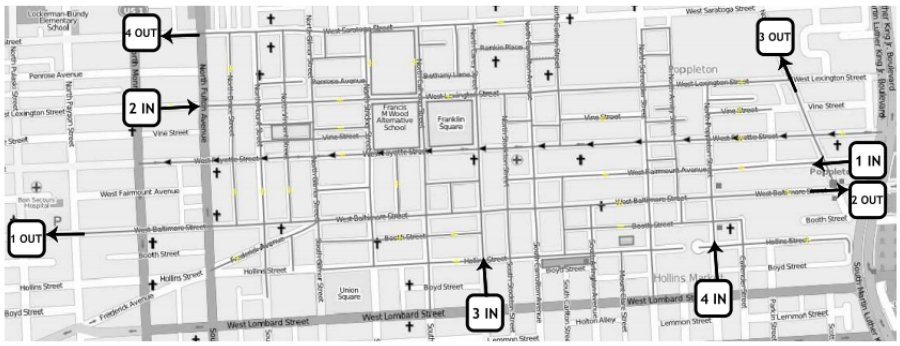

FIGURE 6.1: Flows starting and end points

The flow parameters used into simulations are described in table 6.1

To evaluate the protocol overhead we have considered the flooding reduction of protocol messages in case of position update messages. A smart flooding policy reduces the wasted resources. The test scenario is composed of several vehicles travelling along the maps with and without neighbor management. The number of packets spread into the network are shown in 6.2 Along with the number of packets received, also the effects on air pollution of a smart cooperation between vehicles has been demonstrated. Thanks to a cooperative approach it is possible to reduce the total amount of CO_2 emissions and also the average travelling avoiding sudden

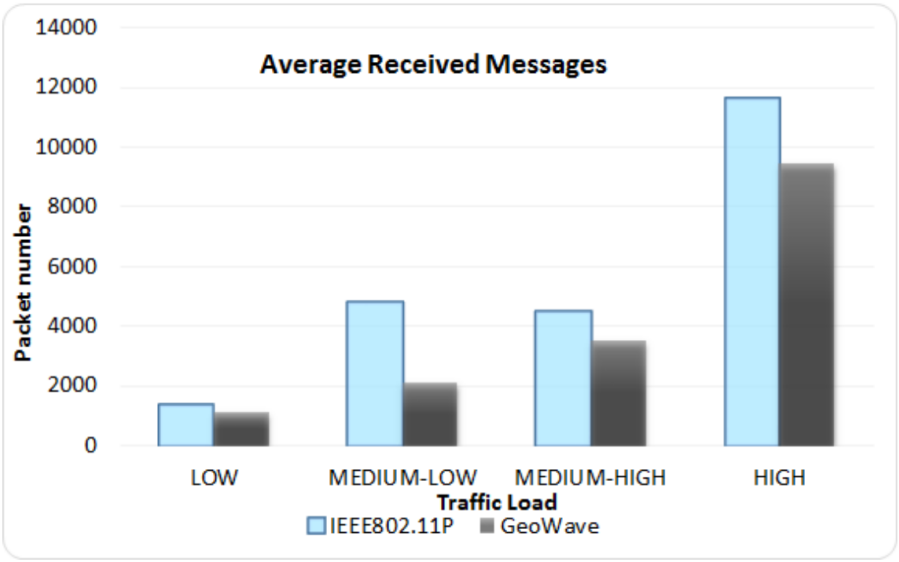

FIGURE 6.2: Average number of received messages per node

accelerations and decelerations. In 6.3 is possible to note how the average CO_2 emissions are reduced, while in 6.4 the average travelling time of vehicles is depicted.

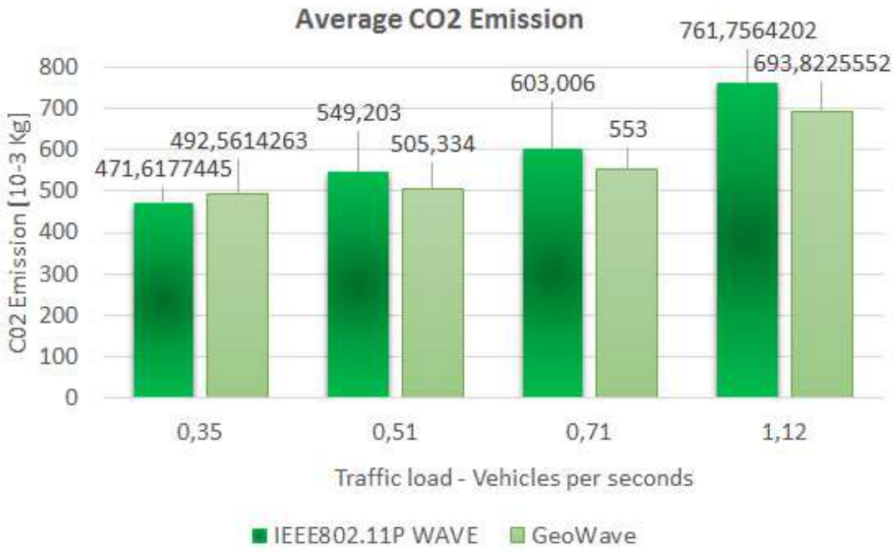

FIGURE 6.3: Average CO2 emissions

The last simulation campaign has been designed to show the benefits of the protocol in terms of traffic management especially during accident events. Using the proposed protocol when an accident message is spread nearby vehicles react in a shorter time avoiding other collisions and diminishing in this way the probability of the occurrence of new accidents. These results are pointed out in 6.5 6.6

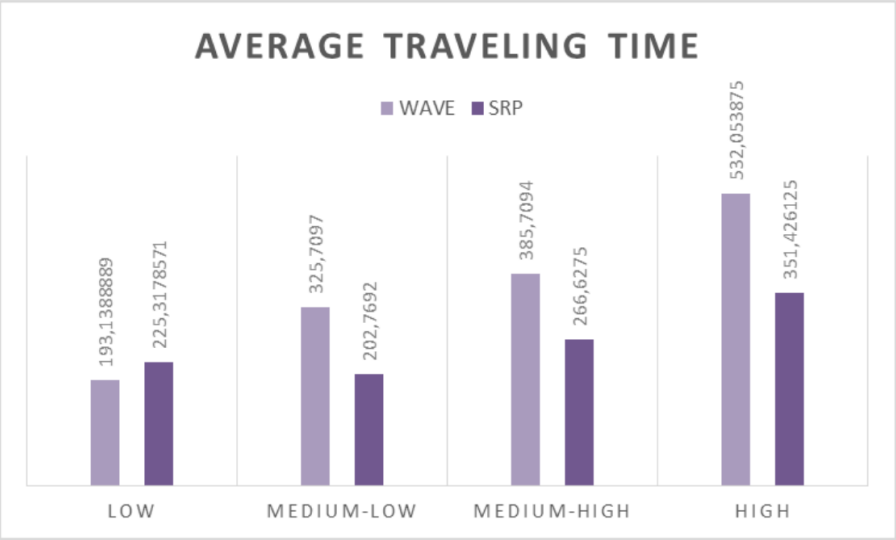

FIGURE 6.4: Average Travelling Time

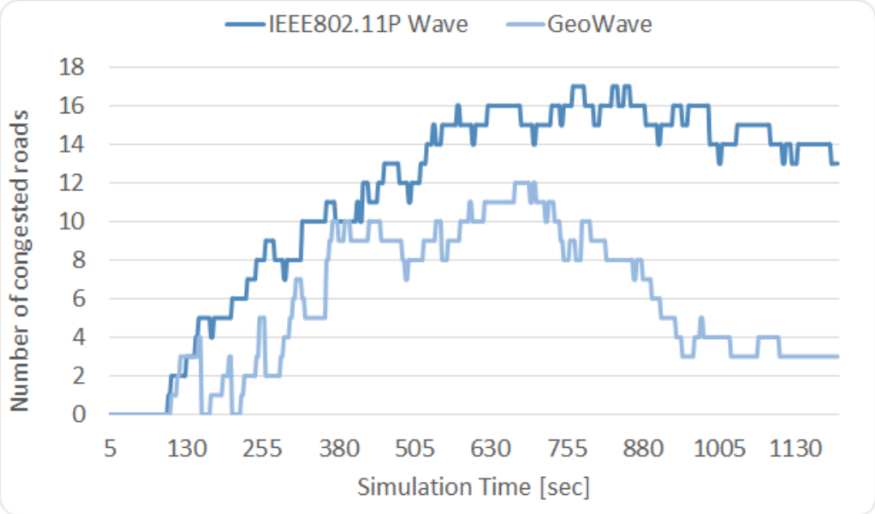

FIGURE 6.5: Congestion Avoidance Behavior

6.1.2 FSCTP transfer protocol in VANET networks

Transfer content in VANET environment is a very difficult issue. The sudden changes of topology cause some of the paths using to distribute contents are really unstable and, for this reason, it is necessary to send and receive data as fast as possible. To face this issue a new transfer protocol has been developed called FSCTP. This protocol allows a data transfer by using both TCP and UDP protocols. The data transfer will be achieved using the UDP protocol while the TCP is used to start and manage the connections. To improve the transmission rate multiple parallel connection are exploited. To prove the goodness of the protocol a set of simulations has been prepared. The tools used are the OMNet++ simulator with the VEINS framework to

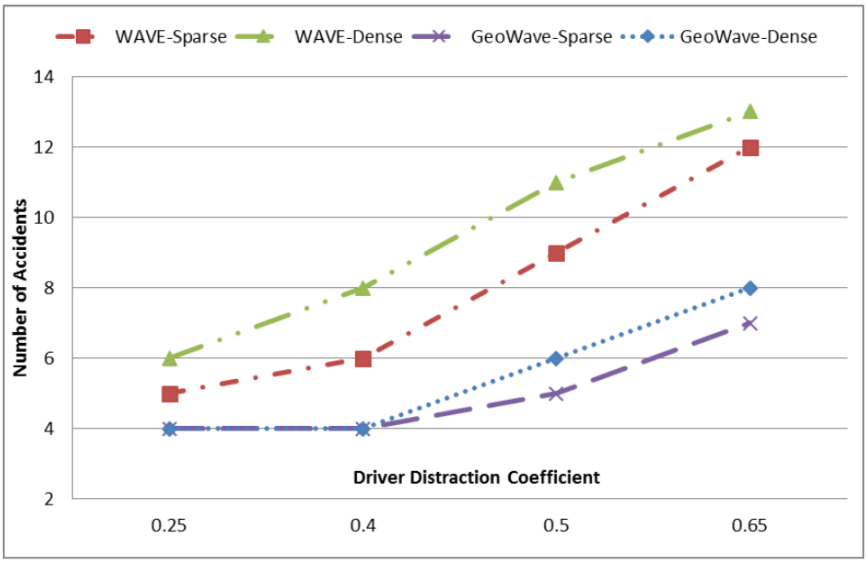

FIGURE 6.6: Total number of accidents VS Driver Distraction Coefficient decreasing

Parameter	Value
Size of city area	$2Km^2$
Rate of inbound vehicles	4 - 5 vehicles per minute
Vehicle's average speed	30 – 40 [Km/h]
# of road cross	7
# of lanes per road	1-2
Fading factor	2 – 2.5
Antenna Gain TX	2
Antenna Gain RX	2
Power TX	20 dBm
Power RX threshold	-95 dBm
Frequency	5.9 GHz

TABLE 6.2: Parameters configuration

exploit the already existing protocol implementation such as IEEE802.11p, TCP and UDP layers. The mobility was managed by the well known SUMO simulator. The campaigns aim was to prove that the proposed protocol performs better than the standard FTP protocol thanks to the UDP transfer 6.7. The considered network for simulation doesn't possess an high load and has a good coverage thanks to a high number of V2V links. The configuration used in simulation is shown in 6.2

The trend of FSCTP protocol when guard time is changed is shown in 6.8. Same configuration is kept to show the protocol behavior when decreasing the guard time. Vehicles ask for a file transfer from the server connected to the RSU. It is possible to note that the transfer time increases at the guard time augmentation. This happens because the vehicle takes a while to communicate the lost packets list to the server.

In Figure 6.9 the average number of change rate request sent is shown. The

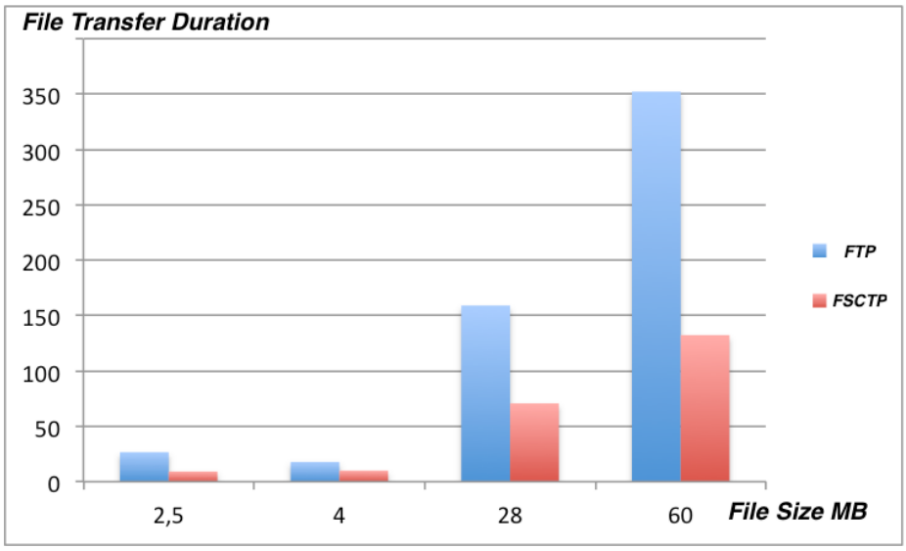

FIGURE 6.7: FSCTP and FTP vs. file size increasing

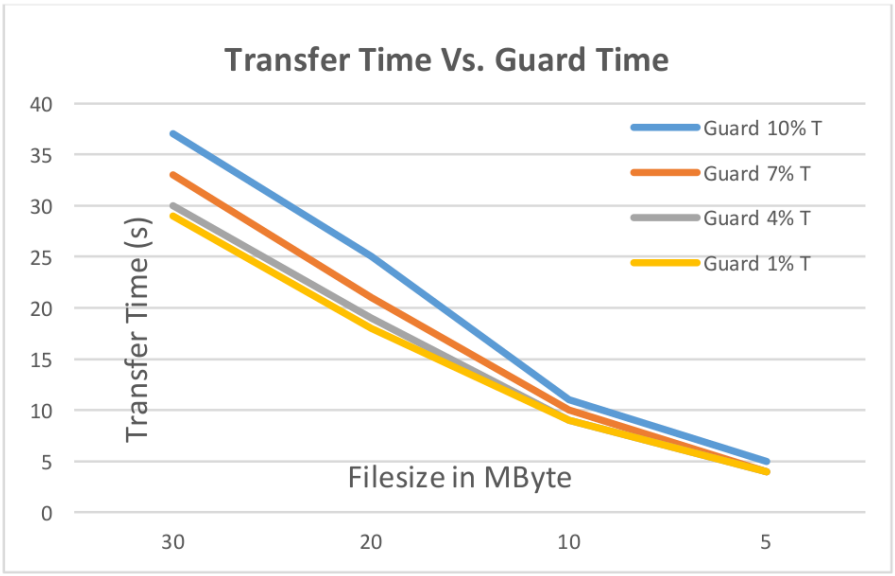

FIGURE 6.8: FSCTP behavior when Guard Time is decreased

higher is the guard time, the lower is the number of requests. But a certain delay is observed that cause the worsening of the network performances, increasing the number of lost packets due to congestion or missing of available relay nodes. This happens when a good coverage is not present.

In Figure 7 a comparison between FSCTP and UDT is shown [103]. As it is possible to note the FSCTP outperforms the UDT. The proposed portocol is better in terms of scalability and resources allocation but with the same of reliability of UDT protocol. The FSCTP at the increasing of fading factor sends a lower number of packets along the network. In this way the network can support a higher number of sessions than the UDT 6.11. The behavior when the channel noise level increases is also good.

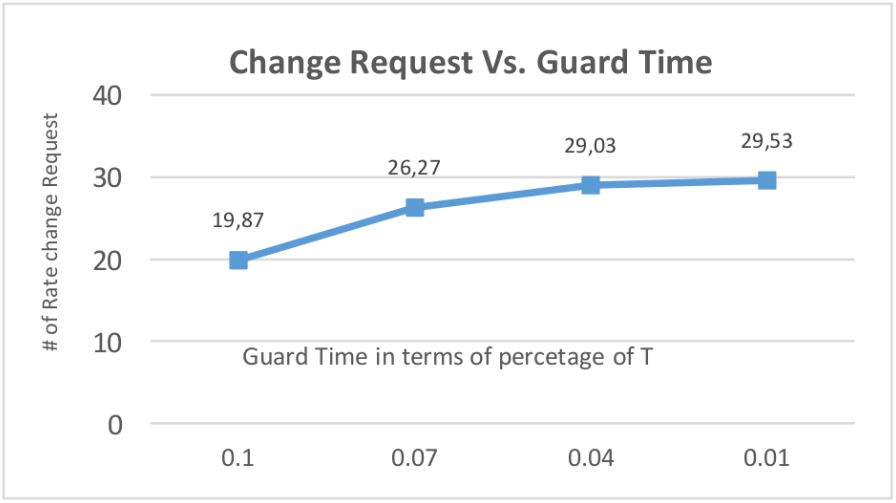

FIGURE 6.9: FSCTP behavior when Guard Time is decreased.

When noise level is high FSCTP is able to tune the transmission to do not waste resources fitting network conditions. This is shown in Figure 6.10 where channel noise levels and the trends of Mbit sent between server and client are considered .

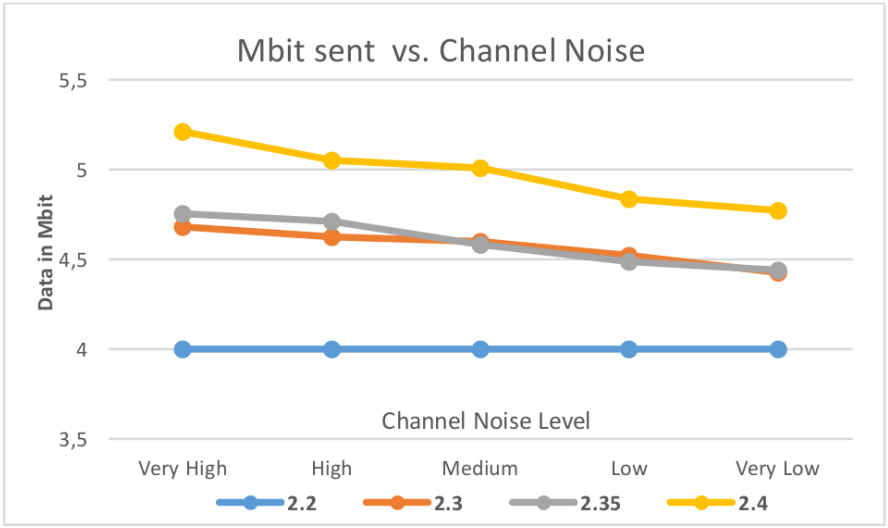

FIGURE 6.10: FSCTP behavior vs. channel noise.

In the end this protocol outperforms a TCP/UDP based protocol called UDT, ensuring high reliability content transfer and lower protocol overhead and speed. Increasing network loads and channel noise it shows a good behavior. Increasing the fading factor has been also considered to reduce the received power to test the performances. In this condition, FSCTP outperforms the UDT while keeping retransmission level low.

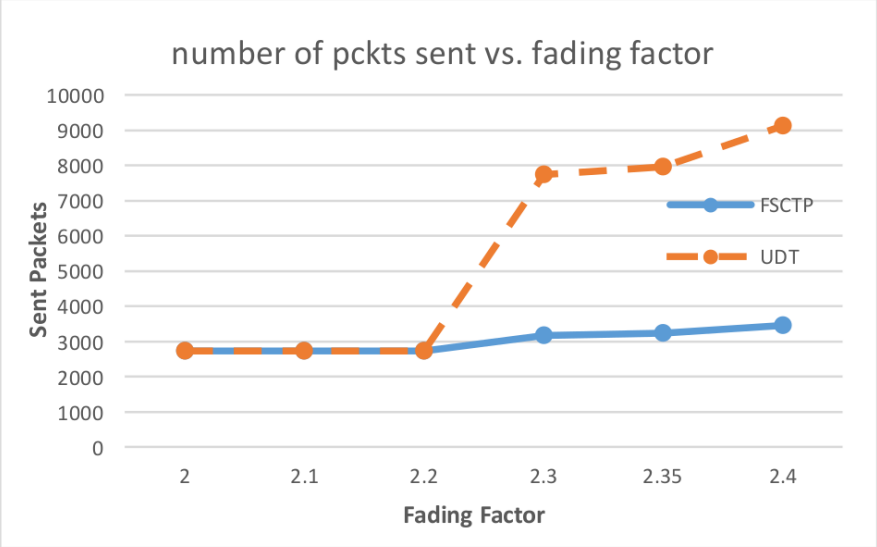

FIGURE 6.11: FSCTP Vs UDT campaigns.

6.2 IoT protocols applied to Vehicular Environment

With the advent of the IoT paradigm my research focused on the possibility to apply protocols and techniques of the IoT to the vehicular environment. The IoT environment and the vehicular environment share have features in common. In both cases we have to consider a very huge number of nodes that needs to share information to enhance their provided service. For these reasons we have proposed a communication protocol in vehicular environment to enhance the overall quality of live by improving safety and energy consumption of vehicles. Regarding this work a Machine2Machine (M2M) [104] platform to recognize wrong driver habits has been designed. When a hazardous behavior is recognized some corrective actions are suggested to the driver. The whole architecture is composed of an embedded device equipped on each vehicle that sends data toward a LAS eploiting MQTT protocol. The whole system has been evaluated in terms of classifier errors, congestion rate of LAS and effect of corrective message on driving style and CO_2 emissions. The testbed for the architecture was composed of several vehicles equipped wit a smart device connected with CAN bus through OBD. The connection between the OBD and the embedded device is made through the bluetooth interface. All the data gathered are then sampled and sent towards the server for a further analyisis. The procedure used to gather and send the data are shown in 4 while a representation of the whole architecture is shown in 6.12

The on road test have been performed three tracks that are shown in 6.13 The (a) path is a urban track of 4 km while (b) is an extra-urban track of 3 km. The last (c) path is another urban track of 3 km. All journeys were made around University of Calabria (Italy) campus that is an environment very close to an urban and sub-urban area. The first simulation campaign aims to point out the goodness of the

Algorithm 4: Publish Data Procedure

```
initialize timer;
while True do
    set value status to not sent;
    if timerTick then
        get last GPS position;
        get last SPEED;
        getMaxAccXYZ;
        create and format MQTT message;
        publish message;
        reset timer;
        set value status to sent;
    end
    if ModAccXYZ >Threshold and value is not sent then
        get last GPS position;
        get last SPEED;
        getMaxAccXYZ;
        create and format MQTT message;
        publish message;
        reset timer;
    end
end
```

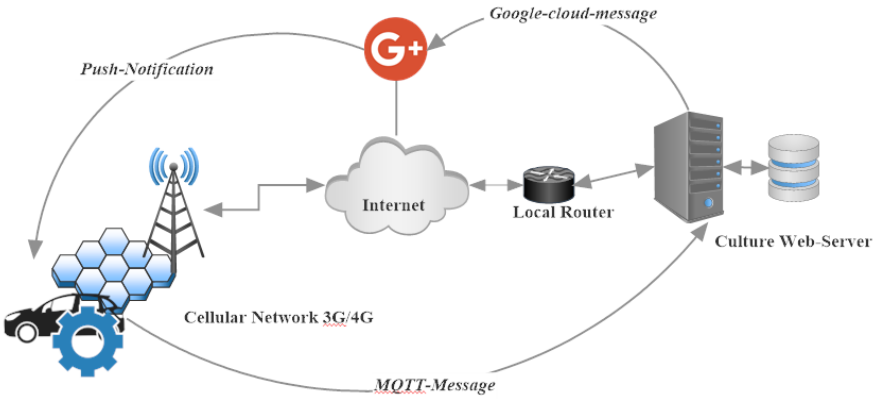

FIGURE 6.12: Testbed architecture

implemented classifier. When a single driver assume the same behavior it is possible to compare samples because their distribution is quite similar 6.14.

Studying the number of occurrences of the acceleration norm it is possibile to point out that increasing the vehicle speed, acceleration are distributed in a different way. Changing the environment from an urban to a faster one with an higher average speed also acceleration occurrences increase thanks also to the higher number of side acceleration. The between two environment is shown in 6.15

Then using the classifier described in 2.4.4 the goodness of the classifier in terms of error is shown in 6.3.

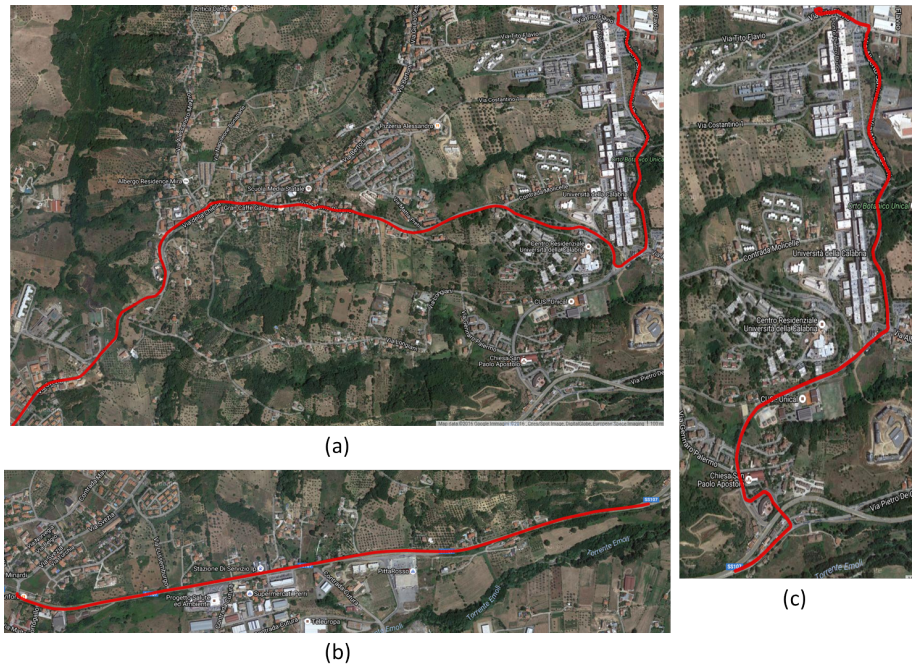

FIGURE 6.13: Example of recognition tracks used to evaluate performances of system.

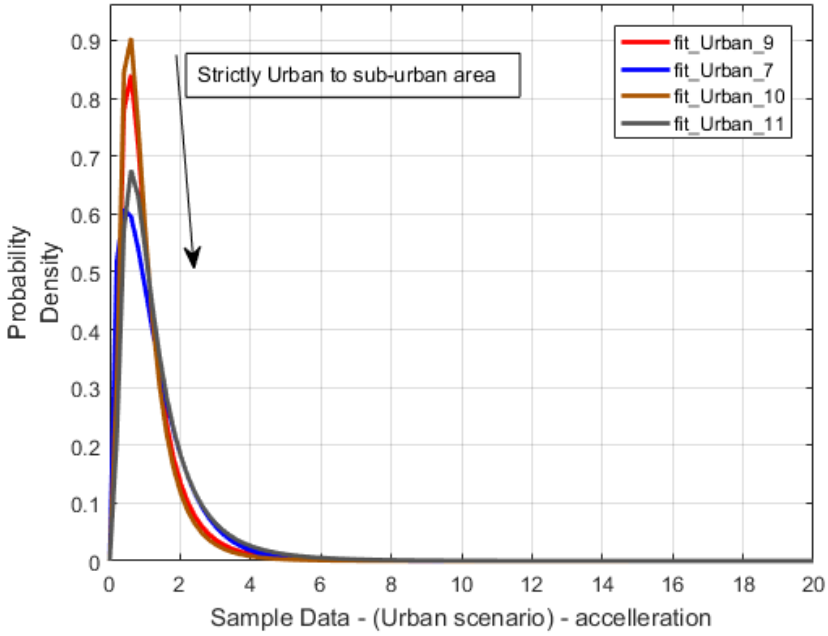

FIGURE 6.14: PDF comparison achieved from data sampling of the same driver that assumes the same behavior (Normal.) on different tracks.

Higher errors levels occurs during the classification of urban environments, here

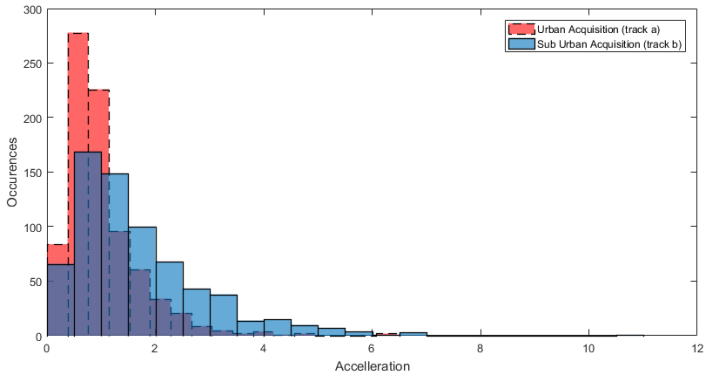

FIGURE 6.15: Comparison between urban and sub-urban accelerations in terms of occurrences.

TABLE 6.3: Measured errors in different environments

	Error			
Environment	Urban	Sub-Urban	Extra-Urban	#Samples
Urban	**0.1607**	0.8393	1	1525
Sub-Urban	0.9463	**0.0557**	0.9979	1435
Extra-Urban	1	0.9031	**0.0969**	1876

the error reaches the value of 16%. Regarding other environments error is lower and range from 5% to 10%. The higher error percentage on urban environment classification is due to the high similarity between urban and sub-urban. The most evident difference is the value of the average speed. The fuzzy clustering method used to detect the three environment classes produces three environment classes with two edges of uncertainty. The first version of the classifier 6.16 considered only speed to detect the environment in which the vehicle was moving. It had unfortunately an high error rate and for this reason we decided to use also information about accelerations6.17.

The second simulation campaign aim was to verify the influence of the corrective message on the driver behavior and, consequently, on fuel consumption and emissions. The tracks used for this campaign are shown in 6.18 and 6.19 that are extra-urban, urban and sub-urban roads.

Thanks to the information acquired from the devices equipped on vehicles we compared the consumption trends of some divers classes. We compared aggressive vs normal drivers. The consumption trend in a observation time period of one month can be seen in 6.20

The influence of a notification of an aggressive driving style on the driver behavior is fundamental for these results. In 6.20 how alerting the user about an aggressive driving style can foster a more energy-safe behavior reducing aggressive driving occurrences and consequently vehicle consumption is proved. The last simulation campaign was focused on network in particular on the mechanism of congestion and

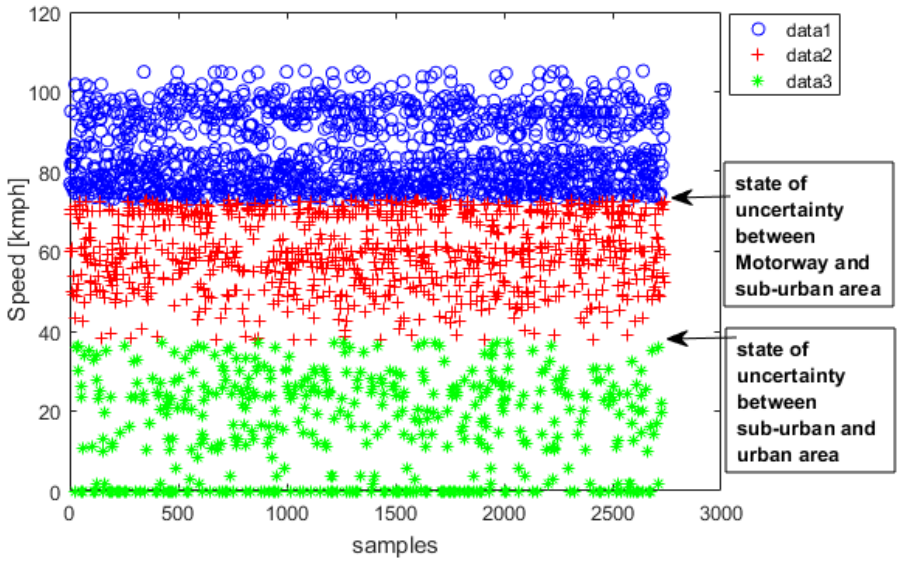

FIGURE 6.16: Fuzzy Centroid for environment classification

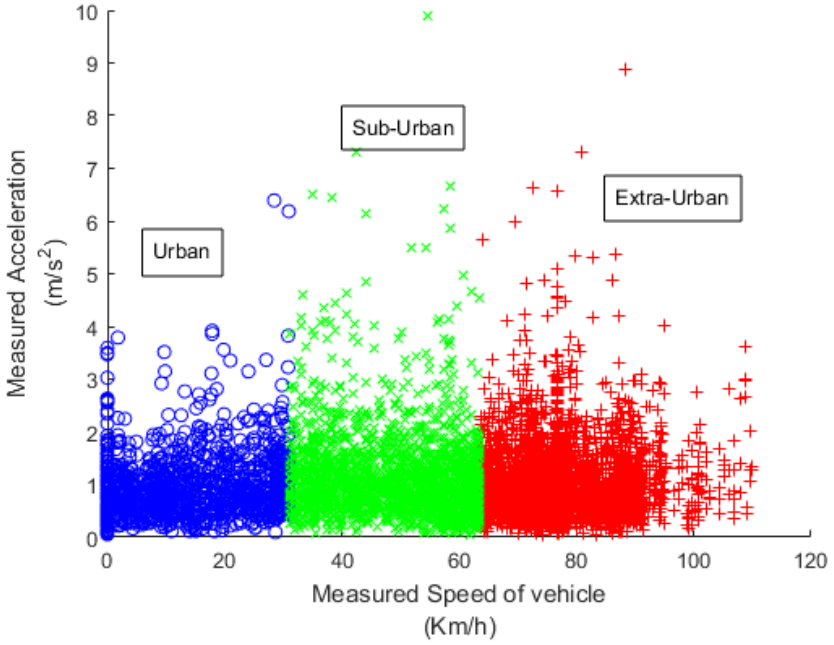

FIGURE 6.17: Environment classifier achieved by using speeds and accelerations and centroid in Mandami inference model.

flow control management. These simulations were performed on OMNeT++ simulator and VEINS framework with SUMO. The *TraciApp* module was customized to comprehend the necessary communication mechanisms. LAS instances were instead implemented in the *TraciRsuApp*. The map used for this campaign was extracted by

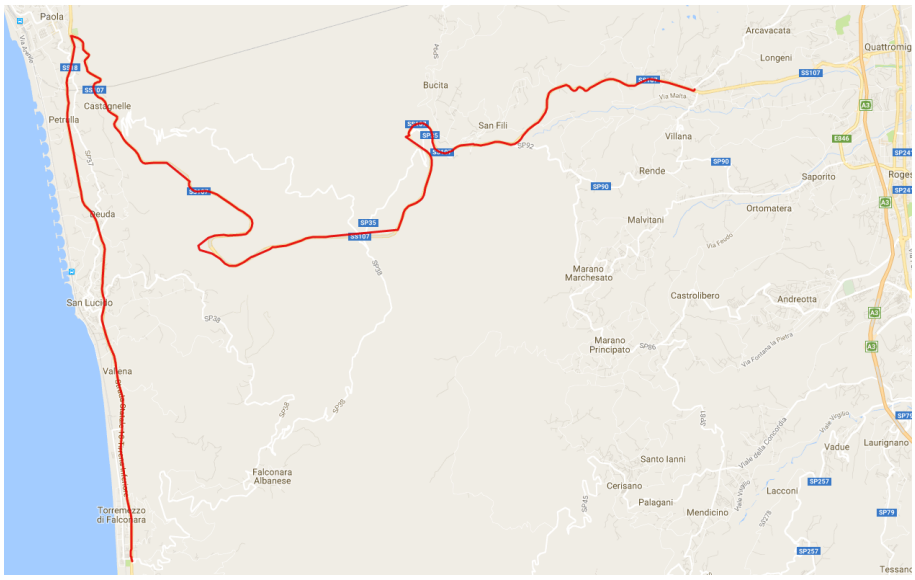

FIGURE 6.18: Used Path for extra-urban environment.

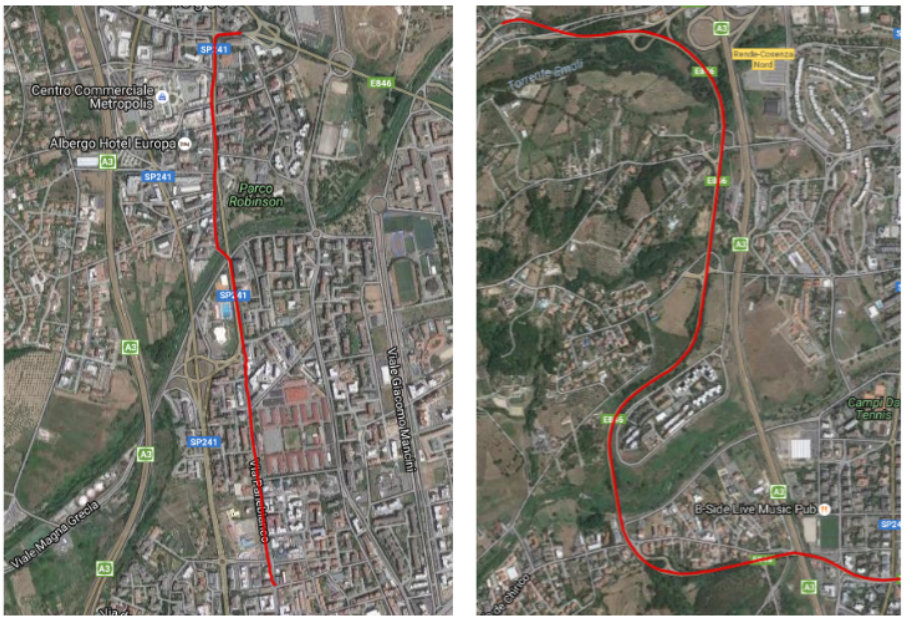

FIGURE 6.19: Urban (a) and Sub-Urban (b) tracks used to evaluate system performances.

a real map using OSM. The behavior of the congestion and flow management mechanism effects are shown in 6.21 and . In particular, we can observe that the dropping rate of the vehicles that are travelling through a congested area drastically decreases if congestion and flow control is on. This result is influenced by classifier performance. In fact, if data shared by vehicles are lost due to the congestion then the classifier is not able to recognize with the same accurancy the environments and

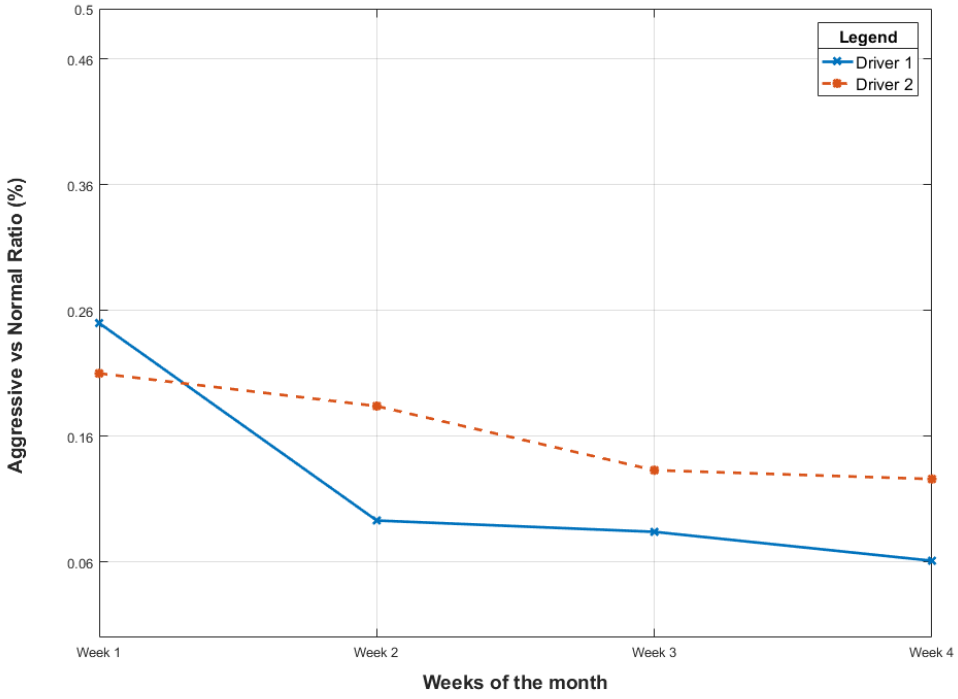

FIGURE 6.20: **Aggressive occurrences trend along one month of observations.**

behaviors.

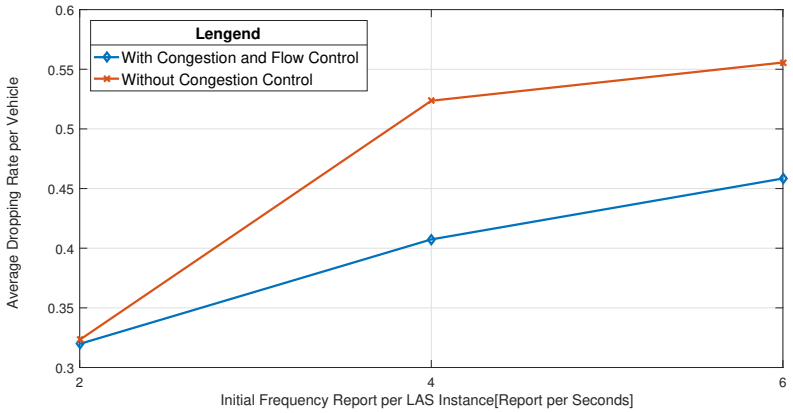

FIGURE 6.21: **Average Dropping rate per vehicle evaluated in a congested scenario.**

In the end through several simulation campaign we showed the goodness of the proposal. Our proposal consisted in a IoT platform to to recognize driver behavior taking into account the environment in which the user is moving. In this way it is possible to promote a more safely driving style and better manage flows inside the city.

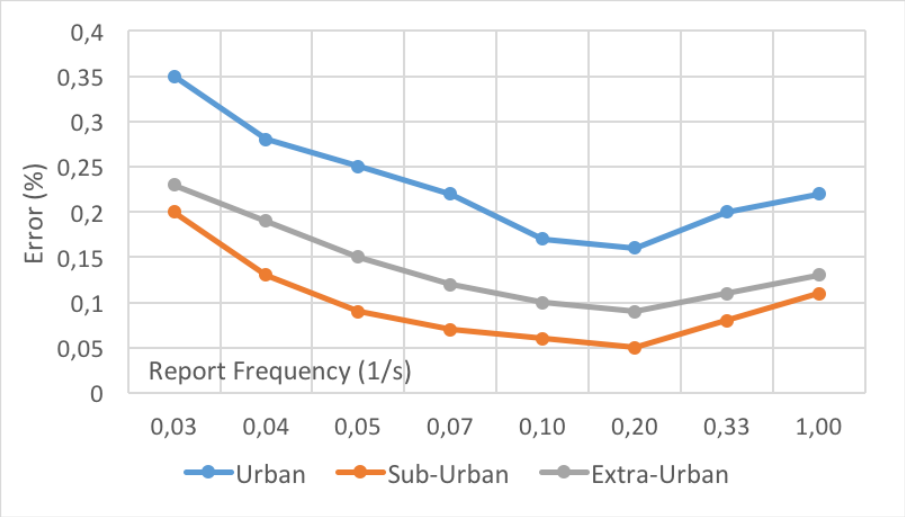

FIGURE 6.22: **Classifier Errors vs. Frequency Report**

6.3 A vehicular traffic simulator model for electric mobility

In 4.4 we presented an implementation of a java simulator with a mobility model for electric vehicles. This model to compute the consumption of vehicle taking into account also information about the altitude and the slope coefficient of the streets. to be more accurate also a KERS emulation mechanism has been introduced in the model. This simulator with this model was designed to be easily extended and to support also VANET protocol integration. Moreover to allow more realistic simulation an integration to the OSM service have been implemented. Since OSM maps do not possess enough information about streets altitude a module to query the Google Maps Api has been added do retrieve in a painless and automatic way all missing information. This study purpos was to investigate of the slope coefficient and street altitudecan affect overall results in different scenarios. Moreover,to depict a simulation scenario near the actual status, also hybrid traffic composed by EV and fuel propelled vehicles have been considered. During the first campaign it has been evaluated the difference in consumption distribution. A urban area has been considered and to better understand the difference in altitude that a vehicle meet during a journey the altitude variation has been depicted in 6.23

The differences between the distribution of vehicles consumption taking into account the altitude coefficient and the driver behavior is presented in 6.24. These data are obtained by changing the considered map area. Each map is characterized by a different average slope coefficient which are summarized in table 6.4 while the driving style considered in our simulation are described in table 6.5.

Each point of the figure represent the average consumption per Km of a vehicle in a map with a specific altitude coefficient and with a specific behavior. When the altitude coefficient of the map raises also the consumption increases. The driving behavior influence the consummations in the same way. Another simulation campaign

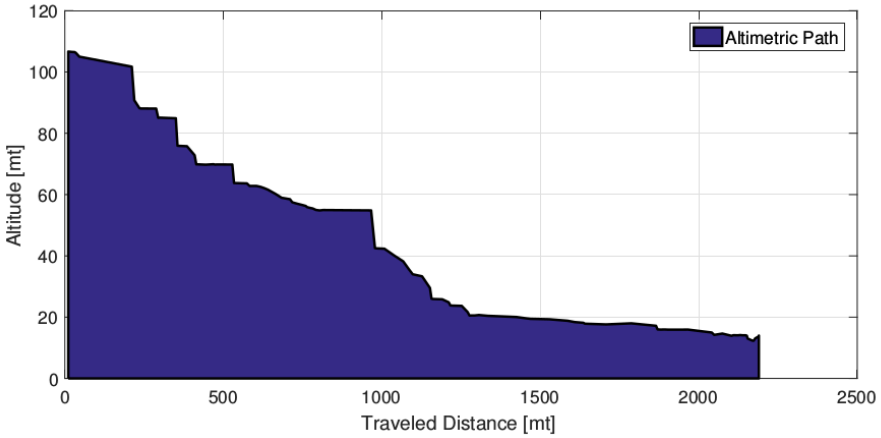

FIGURE 6.23: Altimetric Path along the map area

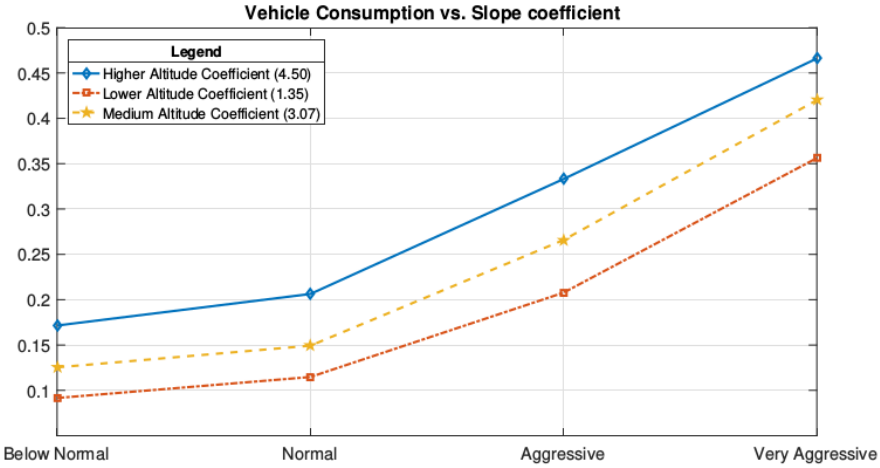

FIGURE 6.24: Average Energy consumption per Km of vehicles versus drivers' behaviors

Map Name	Avg. Slope Coefficient	Area Size
Map Area 1	4.50	15.6 Km^2
Map Area 2	3.07	9.5 Km^2
Map Area 3	1.35	10.6 Km^2

TABLE 6.4: Slope Coefficients

Road Type	BN	N	A	VA
Urban	15.0	25.0	45.0	60.0
Extra Urban sec	20.0	50.0	80.0	100.0
Extra Urban pr	20.0	60.0	100.0	120.0
Highway 50.0	80.0	120.0	140.0	

TABLE 6.5: Table of speeds and Behavior

was made to show the impact of slope coefficient on the consumption of vehicles. The difference in consumption is also determined by the vehicle mass. Using the vehicles with different parameters shown in table 6.6 We obtained the energy con-

Model	Brand	Mass	CdA	Efficiency
Twizy	Renault	460	0.40	0.8
i-MiEV	Mitsubishi	1110	0.75	0.8
Leaf	Nissan	1474	0.57	0.8

TABLE 6.6: Vehicle Parameters

sumption vs vehicle mass in the two cases in which the altitude is considered or not and the results are shown in 6.25.

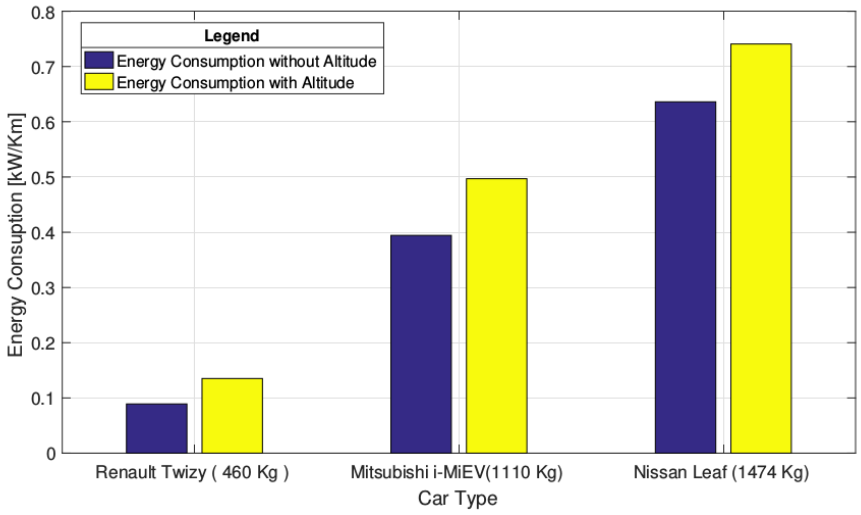

FIGURE 6.25: Average Energy consumption related to vehicle class

This work aimed to propose a new simulation framework in which simulation environment is much realistic. This is achieved by taking into consideration also altitude and related roads' slope coefficient. Moreover, through several simulation campaigns, it has been proved that the altitude heavily influences the results and in particular the Energy Consumption. Some considerations about the behavior and vehicle characteristics have been carried out.

6.4 Improvement of the electric mobility model in SUMO

SUMO already provides a model for electric mobility that takes into account several parameters that can be seen in 6.7.

This model takes into consideration also road altitude to calculate the energy consumption contribute to move the vehicle on a inclined plane. Multiple simulation campaign have been done to check the impact of the slope coefficient on consumption that are shown in this section. Moreover an extension of the electric model

key	Value Type	Default	Description
maximumBatteryCapacity	float	0 (Wh)	Maximum battery capacity E_{max}
maximumPower	float	0 (W)	Maximum power of vehicle
vehicleMass	float	0 (kg)	Vehicle mass mveh
frontSurfaceArea	float	0 (m^2)	Front surface area A_{veh}
airDragCoefficient	float	0	Air drag coefficient cw
internalMomentOfInertia	float	0 (kg·m^2)	Mom. of inertia of int. rot. elements J_{int}
radialDragCoefficient	float	0	Radial drag coefficient c_{rad}
rollDragCoefficient	float	0	Rolling resistance coefficient c_{roll}
constantPowerIntake	float	0 (W)	Avg. (constant) power of consumers P_{const}
propulsionEfficiency	float	0	Drive efficiency η_{prop}
recuperationEfficiency	float	0	Recuperation efficiency η_{recup}
stoppingTreshold	float	0.1 (km/h)	Minimum velocity to start charging

TABLE 6.7: Electricl Model Parameters in SUMO

has been implemented to consider also other source of consumption. Nowadays vehicle are equipped with a wide set of sensors to help the driver in some maneuver like parking assistance or cruise control. Moreover one of the most interesting topic in the research community are autonomous vehicles. AV are self driving car that are able to move autonomously inside and urban environment. To do so some advanced sensors are necessary to perceive the surrounding environment and take decision about the trajectory. These sensors, depending on the specifications, consume energy to work. Energy consumption of sensors , for both fully autonomous vehicle or advanced vehicles, is not so relevant when the vehicle is fuel-propelled but could affect in a relevant way a vehicle powered by electric batteries. One of the simulation campaigns aimed to depict the difference between the consumption of vehicles with and without taking into consideration altitude. The urban area used to run our simulation is the city near the University of Calabria campus in an area of approximately 270 km^2 using the map exported on OSM shown in 6.26

This map has been converted in sumo map format using the tool *netconverter* shown in6.27 . Also the obstacle and building have been added thanks to the SUMO tool *polyconvert* 6.28.

The flows that have been simulated are specified inside the routing xml file. The traffic flow defined is an average city traffic lasting 12 hours divided in three periods characterized by the numbe of active vehicles inside the city. The flows were divided in three periods:

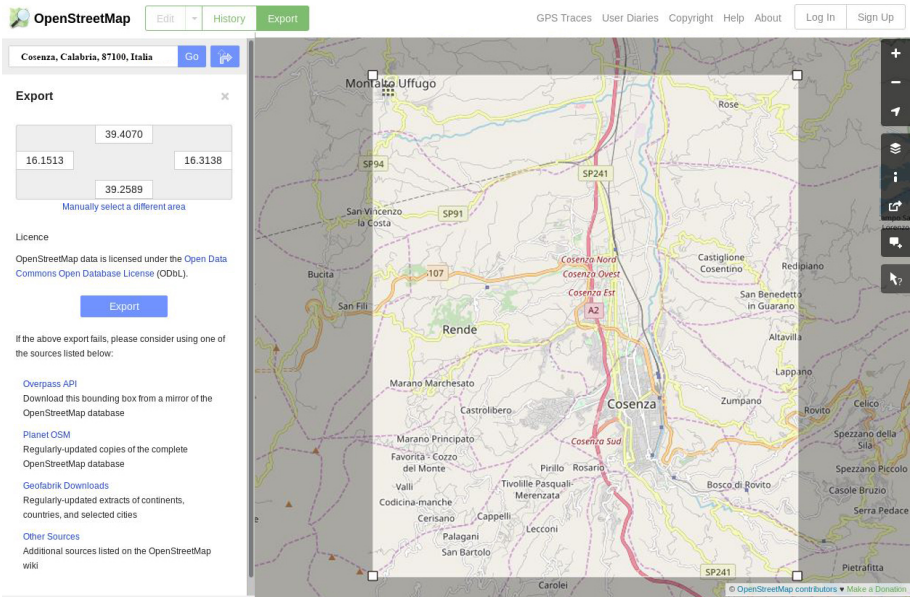

FIGURE 6.26: Map exported from OSM

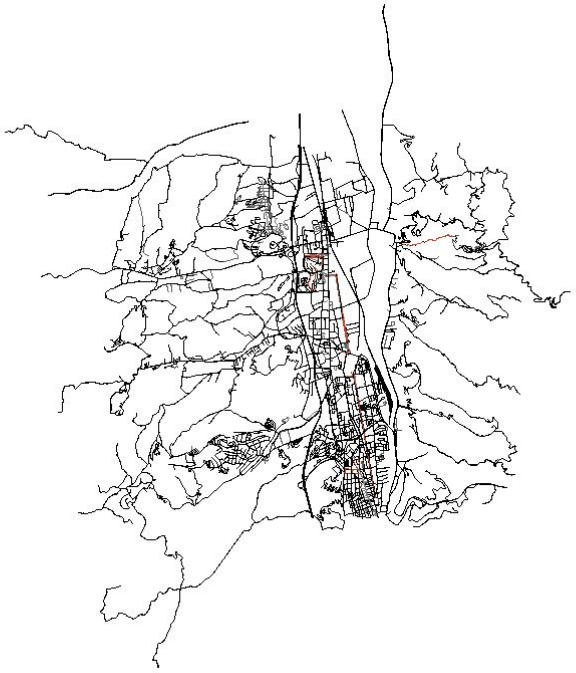

FIGURE 6.27: Sumo map format of the area

- First Period: rate of approximately 1690 vehicles per hour for a total of 5.450 vehicles inside the city in 3.23 hours divided in 108 flows.
- Second Period: rate of approximately 1000 vehicles per hour for a total of 6650 vehicles inside the city in 6.65 hours divided in 132 flows.

FIGURE 6.28: Obstacles in the map

- Third Period: rate of approximately 1200 vehicles per hour for a total of 2550 vehicles inside the city in 2.18 hours divided in 50 flows.

The vehicle type used in the simulations is electric and is powered by batteries and all parameters used can be found in 6.8

Even if a recuperation energy mechanism is present the slope coefficient has relevant impact on the battery autonomy. The analysis shows that the average energy consumption of vehicles using altitude is relevantly higher than the simulation made without considering it. The results are shown in 6.9. A graphical representation of the results is also shown in 6.29

In 6.29 flows composed of 50 vehicles are shown. For each flow the average consumption with and without altitude is represented in the histogram. Depending on the length of the trip and the slope coefficient of the travelled roads.

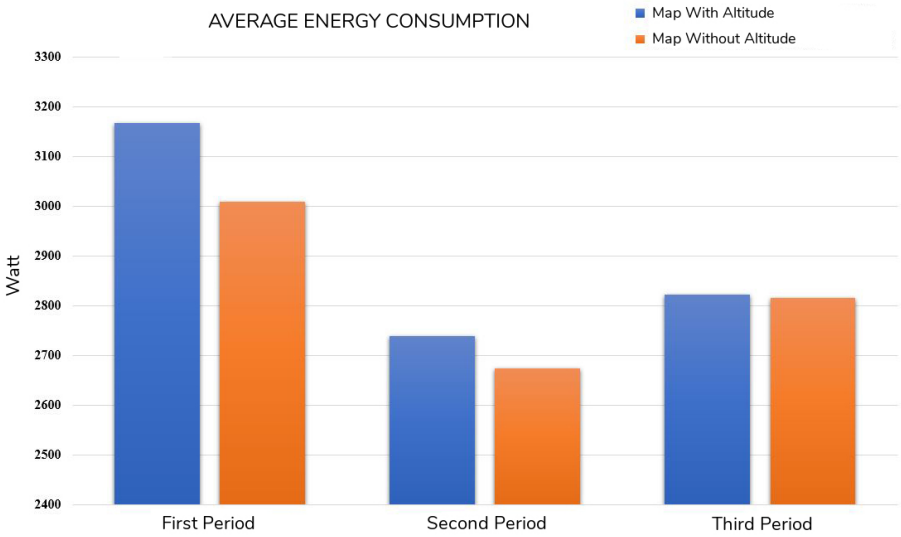

FIGURE 6.29: Average Energy consumption for each period

Parameter	Value	Description
id	Electric Vehicle	Name and type of vehicle
accel	2.6	maximum acceleration in m/s^2
decel	4.5	maximum deceleration in m/s^2
length	4.0	vehicle length in m
maxSpeed	40	maximum speed in m/s
minGap	2.5	minimum distance between vehicle in m
sigma	0.5	*krauss* model value
maximumBatteryCapacity	40000	maximum battery capacity in W
maximunPower	20000	maximum power intake of the vehicle W
vehicleMass	1500	vehicle mass in Kg
frontSurfaceArea	2.0	Front surface area in m^2
airDragCoefficient	0.3	areaodinamci coefficient of the vehicle
internalMomentOfInertia	0.1	Inertia coefficient
radialDragCoefficient	0.5	radial drag coefficient
rollDragCoefficient	0.01	roll drag coefficient
constantPowerIntake	100	constant power intake in Wh
propulsionEfficiency	0.9	propulsion efficiency factor
recuperationEfficiency	0.5	coefficent of energy recuperation
stoppingTreshold	0.1	minimum speed to start the recharging

TABLE 6.8: Simulation Parameters

	Map with Altitude	**Map without altitude**
Period 1	3166.883 Watt	3008.995 Watt
Period 2	2739.450 Watt	2673.622 Watt
Period 3	2822.484 Watt	2816.523 Watt

TABLE 6.9: Average consumption of vehicle with and without altitude

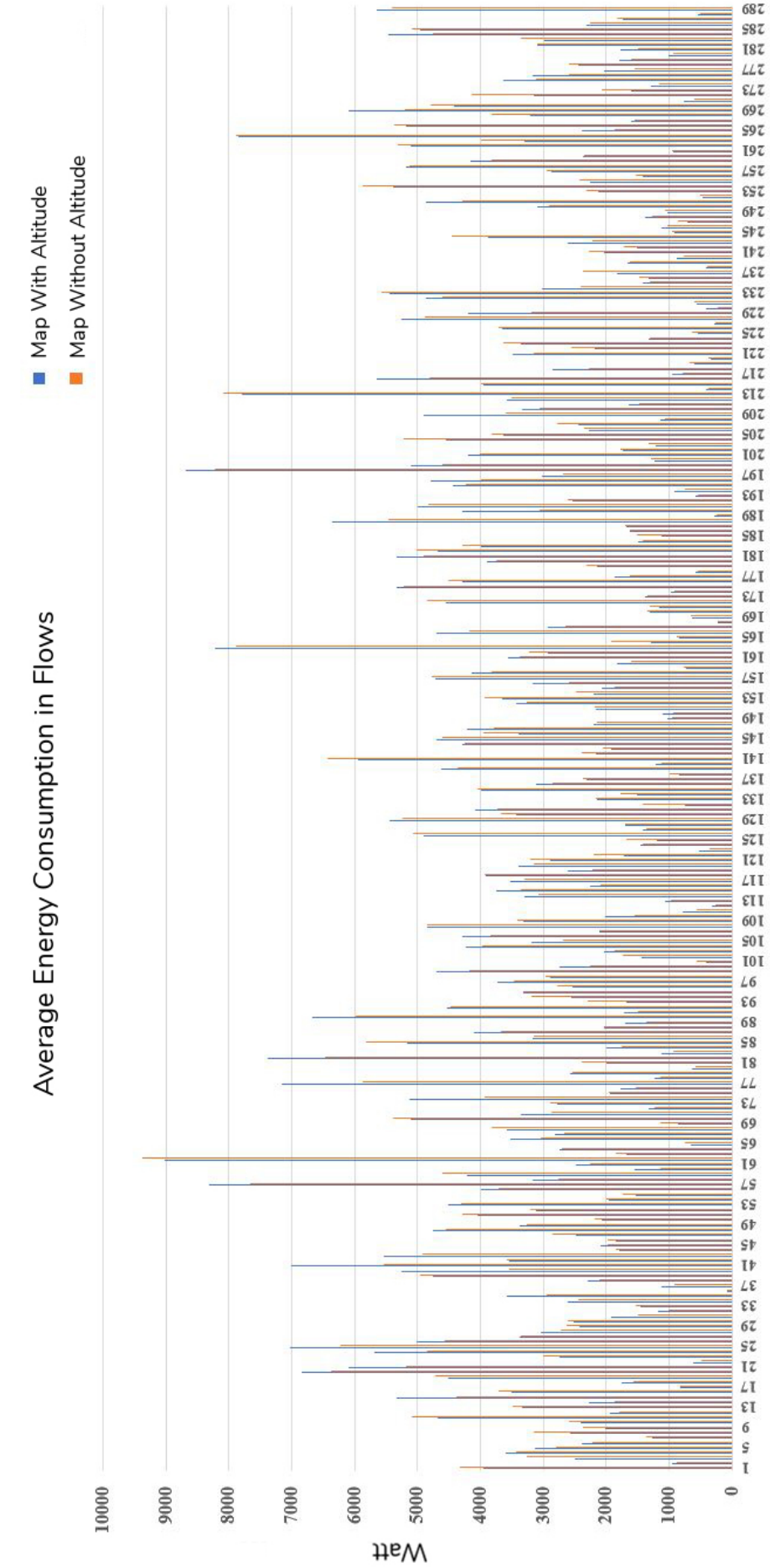

FIGURE 6.30: Comparison between average consumption in flows with and without altitude

To better model the energy consumption in electric vehicles an extension of the well known SUMO simulator has been developed. This extension, as shown in 4.5.1, is composed of a sensors device equipped on each vehicle that compute the consumption of different sensors configuration and subtract it to the remaining energy inside the battery device. The next simulation campaign aims to show the impact of different configuration of on-board sensors on the autonomy of the vehicle. The configurations taken into account represent a vehicle with limited autonomous applications and a fully autonomous vehicle. The two vehicle configuration used during simulations are showed in 6.10 and 6.11

Sensor	**Number**	**Nominal Consumption**	**Total Consumption**
Lidar	1	9.8 Watt	9.8 Watt
Camera	3	2.5 Watt	7.5 Watt
GPS	1	20 Watt	20 Watt
CPU	1	350 Watt	350 Watt

TABLE 6.10: Configuration of Type 1 vehicles

Sensor	**Number**	**Nominal Consumption**	**Total Consumption**
Lidar	6	9.8 Watt	58 Watt
Camera	6	2.5 Watt	36 Watt
GPS	1	30 Watt	30 Watt
CPU	1	3200 Watt	3200 Watt

TABLE 6.11: Configuration of Type 2 vehicles

The map used for this campaign is the same of the previous one and also the slope coefficient of the roads has been taken into account. Using the SUMO tool *duarouter* a traffic of 14650 vehicles with progressive starting time has been setup. The simulation duration is of 12 hours divided in three periods :

- First Period: rate of approximately 1690 vehicles per hour for a total of 5.450 vehicles inside the city in 3.23 hours divided in 109 flows.
- Second Period: rate of approximately 1000 vehicles per hour for a total of 6650 vehicles inside the city in 6.65 hours divided in 133 flows.
- Third Period: rate of approximately 1200 vehicles per hour for a total of 2550 vehicles inside the city in 2.18 hours divided in 51 flows.

The two sensors configuration have been tested both with the aforementioned flow configuration. The simulation results that have been carried out shows that a heavy sensor configuration on a electric vehicle has a really big impact on the autonomy. The difference between the two sensors configuration can be seen in 6.31

Regarding the impact of sensors on energy consumption of a electric vehicle a configuration with an high number of sensors can represent a percentage of battery consumption of 13% - 29% while a low number sensors configuration range between

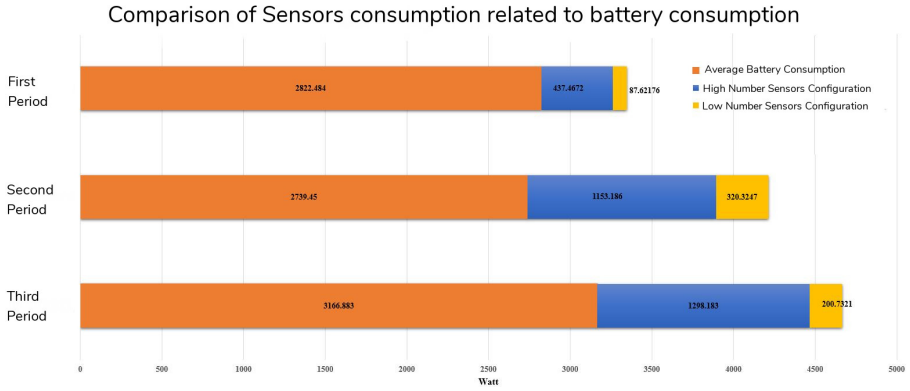

FIGURE 6.31: Comparison of sensor consumption related to battery

0.62% and 10.47%. It is, for this reason, necessary to tailor the sensor configuration of a vehicle taking into consideration the application type. These results are shown in 6.32

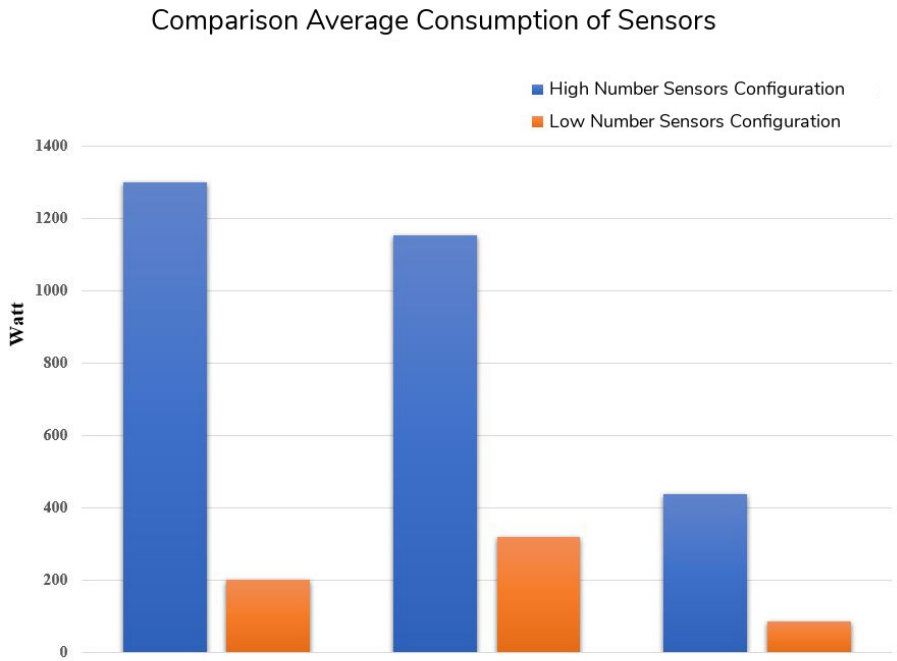

FIGURE 6.32: Comparison of average sensors consumption

In 6.33 flows composed of 50 vehicles are shown. For each flow the average consumption with a low sensors number and high sensors number configurations is represented.

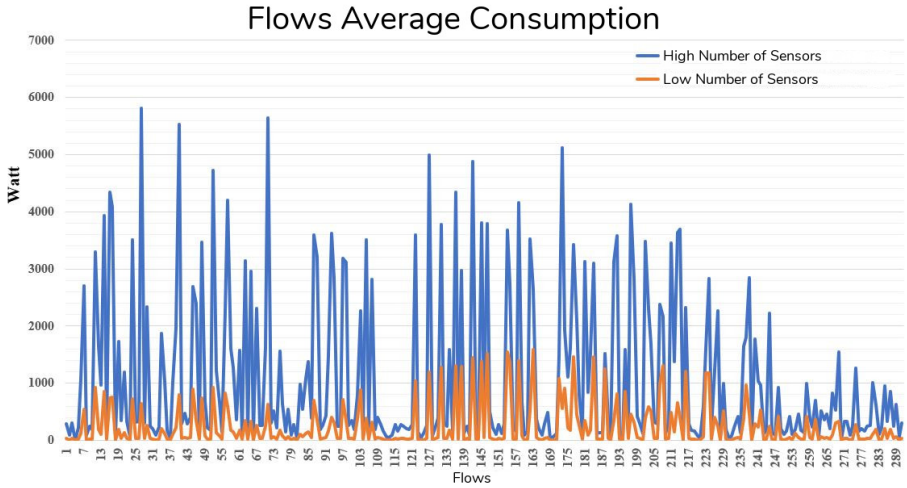

FIGURE 6.33: Comparison of average sensors consumption for each flow

6.5 The impact of driving style in consumption

The heavy influence of the driving style on the consumption and consequentially on an electrical vehicle autonomy or CO_2 emissions in the case of fuel propelled vehicles is clear. Results are very impressive in terms of average speed and accelerations of the aggressive drivers. A safer and carefully driving style is reported by the observations that were conducted showing an impressive reduction of fuel consumption with a normal behavior. When talking about EV not only the behavior must be taken into account regarding consumption but also other factors that deeply impact on the consumption of an electric engine. One of the most underrated of this factor is the street gradient that influence directly the work needed to move the mass of the vehicle and the work necessary to win the friction of the tires. In order to oppose to these influence in electrical mobility some kind of tweaks have been developed like the energy recovery system from the brakes and from the engine rotor. These aforementioned influences are shown respectively in figures 6.35 and 6.34.

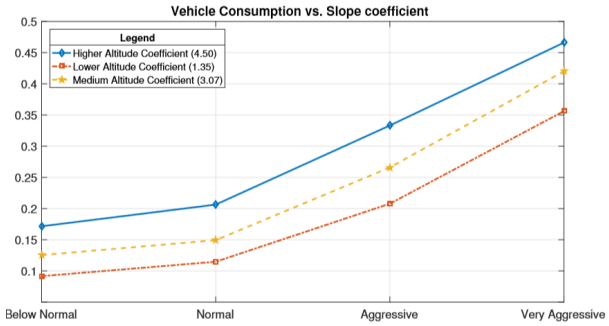

FIGURE 6.34: Consumption vs Behavior

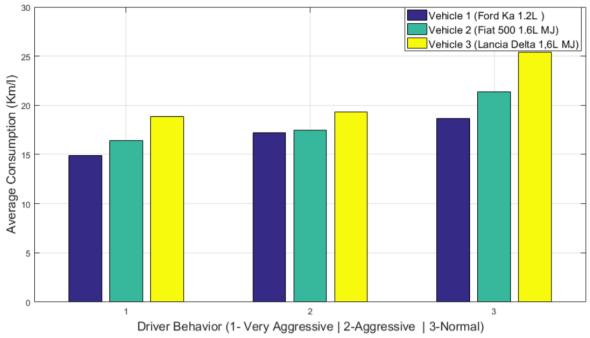

FIGURE 6.35: Behavior vs Class

In figure 6.34 it is possible to note how both slope coefficient and driving style have a great influence on the consumption of a vehicle. With less aggressive driving style and a minor street gradient the consumption are far lesser than the other categories. Increasing the aggressiveness of the driving style and the street gradient the consumption raise significantly. Figure 6.35 depict how the driving style has a greater impact on the consumption for class of vehicles with a considerable mass. The reason is that the work needed to move a vehicles with a considerable mass is more than the work to move vehicle with lesser mass. This is accentuated by the sudden accelerations and deceleration that characterize an aggressive driving style.

6.6 Platooning

The platooning application consists in several vehicles that, becoming a group called platoon, travel along the path with a small inter-vehicle gap. To maintain control of each vehicles and to avoid collisions it is strictly necessary a coordination among all vehicles that are inside the formation. This coordination is ensured thanks to a message exchange that allows a vehicle called leader to send information and commands to the other platoon members. When taking into consideration a platoon of electric autonomous vehicle the platoon formation offers several more advantages. When in a platoon an autonomous vehicle can deactivate some of the on-board equipped sensors to save energy during the travel while inside the platoon formation. To verify this energy saving from platoon members some simulation campaigns have been made thanks to the PLEXE framework. To take into account energy consumption from sensors beside the normal energy used by the vehicle to move an extension of PLEXE similar to the extension proposed in 4.5.1 has been done. The PLEXE framework use the VEINS framework but introduce new ad hoc function and parameters to model platoons. For this reason some adjustments to the code were necessary mostly regarding the simulation time step. As seen in 5.2.3 one of the major difficulties introducing the platooning inside VEINS was transforming a microscopic

simulation model into a sub-microscopic simulation model. The configuration used for the vehicles is herein reported

```
<vType id="vtypeauto" accel="2.5" decel="9.0" sigma="0.5" length="4" minGap="0"
maxSpeed="1000" color="1,0,0" probability="1"
speedFactor="1000" carFollowModel="CC" tauEngine="0.5"
omegaN="0.2" xi="1" c1="0.5" lanesCount="4"
ccAccel="1.5" ploegKp="0.2" ploegKd="0.7" ploegH="0.5" >
        <param key="maximumBatteryCapacity" value="30000"/>
      <param key="maximumPower" value="30000"/>
      <param key="vehicleMass" value="1500"/>
      <param key="frontSurfaceArea" value="1"/>
      <param key="airDragCoefficient" value="0.5"/>
      <param key="internalMomentOfInertia" value="0.01"/>
      <param key="radialDragCoefficient" value="0.5"/>
      <param key="rollDragCoefficient" value="0.01"/>
      <param key="constantPowerIntake" value="100"/>
      <param key="propulsionEfficiency" value="0.7"/>
      <param key="recuperationEfficiency" value="0.0"/>
      <param key="stoppingTreshold" value="0.1"/>
      <!--Sensors consumption parameters -->
      <param key="has.sensors.device" value="true"/>
      <param key="cameraNumber" value="2"/>
      <param key="cameraConsumption" value="2.5"/>
      <param key="lidarNumber" value="1"/>
      <param key="lidarConsumption" value="60.0"/>
      <param key="cpuNumber" value="2"/>
      <param key="cpuConsumption" value="350.0"/>
      <param key ="gpsNumber" value ="1"/>
      <param key ="gpsConsumption" value = "20.0"/>
    </vType>
```

When a vehicles besides the leader and the manager join a platoon it can put in standby some of the sensors that are active for the navigation and use the directives of the leader to maintain the position inside the platoon avoiding collisions. In this way it is possible to save energy consumed by sensors. Obviously the greater is the number of vehicle inside the platoon the more is the energy saving but the harder is the job of the leader to maintain and guide the formation. Because of the sub-microscopic nature of the simulation model it is impossible to simulate , in a reasonable amount of time, complex dynamics of the platoon and for this reason two of the main possible maneuvers are here reported.

6.6.1 Join Maneuver

The join maneuver consists in a lone vehicle that, after receiving the recruiting beacon from a leader, decide to join a platoon, move to the joining position and switch its cruise control to CACC. When the CACC cruise control overtake the driving task some of the on-board sensors are switched off. The trend of the consumption of the various platoon member is shown in 6.36

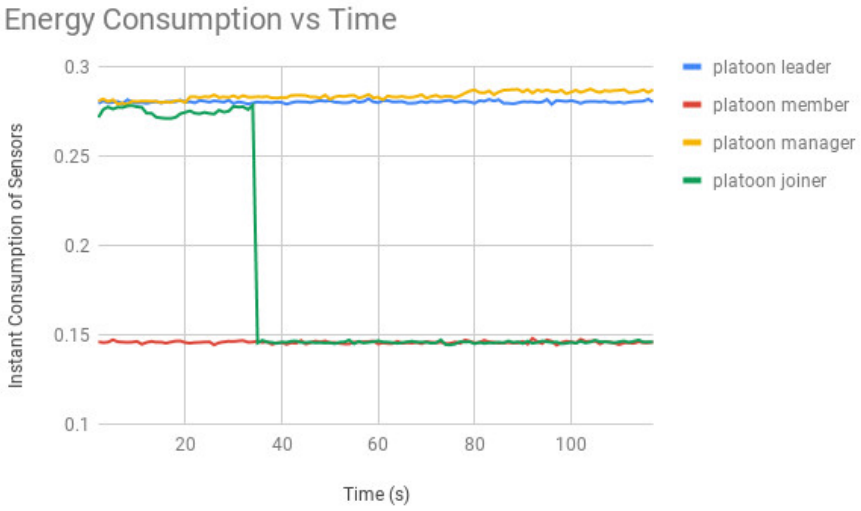

FIGURE 6.36: Energy consumption over time

The configuration of the maneuver is pretty simple. A formation platoon of three vehicles is spawned at the beginning of the simulation while a vehicle outside the platoon is spawned a few moments later. When the vehicle outside the platoon receive the recruiting beacon it communicate his will to join the platoon sending to the leader a unicast message *joinPlatoonRequest*. If the leader is not involved in another maneuvers it answers with a positive *joinPlatoonResponse*. The vehicle is guided by the leader through a set of *moveToPosition* messages. When the joining vehicle is finally in position and is part of the platoon switch its cruise control to CACC and send a *joinFormation* to the leader. The leader answer with a *joinResponse*.

6.6.2 Leave Maneuver

The leave maneuver is the opposite of the join maneuver but it is simpler in terms of messages exchange. When a vehicle want to leave the platoon formation sends the leader a *leavePlatoonRequest*. If the Leader is not involved in other maneuver it answers with a positive *leavePlatoonResponse*. After receiving a positive ack for its request the leaver. The trend of the instant consumption of the platoon vehicles and the leaver is shown in 6.37

The Energy consumption of the leaver member is less than the other because after leaving the platoon it slow down to leave the platoon consuming less energy for the locomotion. In the end each vehicle that is capable of joining a platoon should do it. This can be also seen in 6.38 where the average instantaneous consumption is depicted for each kind of vehicle.

Even if is always advisable for a lone vehicle to join the platoon the problem of joining the *best* one still presist.

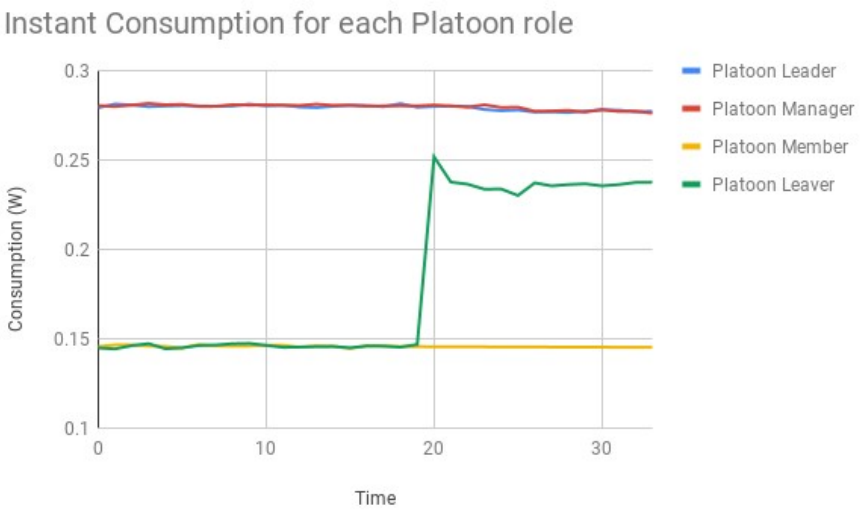

FIGURE 6.37: Energy consumption over time in leaving maneuver

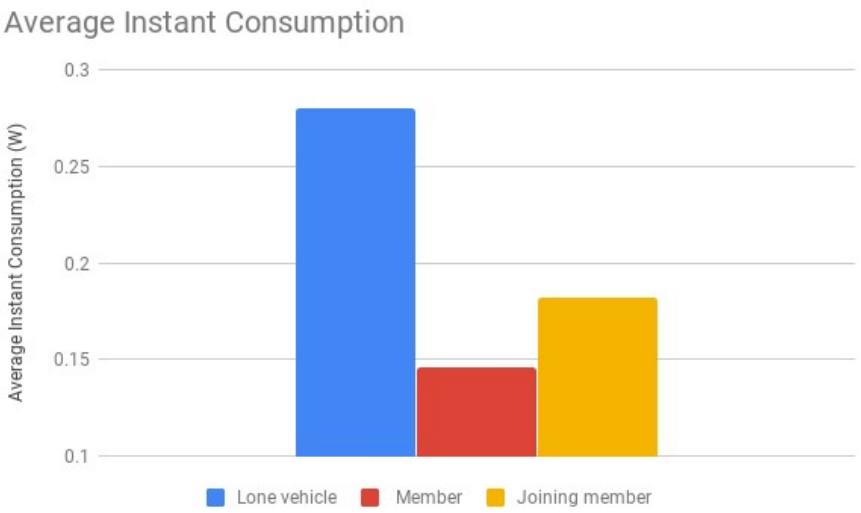

FIGURE 6.38: Average Instant Consumption of vehicles

6.7 Gray Wolf Optimizer for Platoon selection

The problem to assign a vehicle to the best platoon is really difficult that is really similar to the Weapon Target Assignment (WPA) problem. To resolve this problem a meta-heuristic approach has been used base on the GWO algorithm. This technique is based on the way gray wolves hunts by searching for the prey and surrounding it before the final attack. This resolution scheme produce good results in terms of quality of the solution and computational resources to find it. Referring to the algorithm shown in 5.5 some tests have been made. The algorithm parameter are shown in Tab. 6.12.

Parameter	Description
EnV	An array describing the average energy consumption of each vehicle
EnP	An array describing the average consumption of vehicles inside the platoon
q0	An array describing the max size of each platoon
Interr	A Matrix describing for each vehicle ad each platoon the number of consecutive edges shared along the path
X0	A Matrix representing the starting solution of the GWO algorithm. Usually a zero matrix
ws	An integer representing the size of the wolf pack searching for the prey
t	An integer representing the max number of iteration that the algorithm should last

TABLE 6.12: Parameters Used for the algorithm

TABLE 6.13: Problem Size

Name	Description	Value
Cars Number	Number of Cars to assign	20
Platoons Number	Number of Platoons	5
Iterations Number	Number of iterations to find a solution	5 10 15 and 20
Sarting Solution	Initial Solution which the algorithm use at startup	Empty (no cars assigned to platoons)
Platoon Size	Max member allowed inside a platoon	5

The size of the problem used to test the algorithm is shown in :

To test the convergence speed and the quality of of the found solution we only change the number of wolves composing the pack. The parameter that has been changed during each trial is the wolf pack population size that started from 5 wolves and then has been incremented by 5 until reach a population of 20. An initial solution is passed as a parameter because it should be possible to start the algorithm by a configuration where some vehicles are already inside a platoon. The elegibility matrix is then computed on the solution provided as input.The results obtained from the runs are shown in 6.39.

A large population of wolves allows a better and wider search in the solution space because the movement direction of the pack is less influenced by few important wolves. In this way also the probability of local optimum trap is lesser. In Fig. 6.40 and Fig. 6.41 are shown the trend of the fitness value during a run of the algorithm with respectively 5 and 20 wolves as a pack population size.

Using a larger wolf pack size the solution found is better and the time needed to surround the prey is lesser.

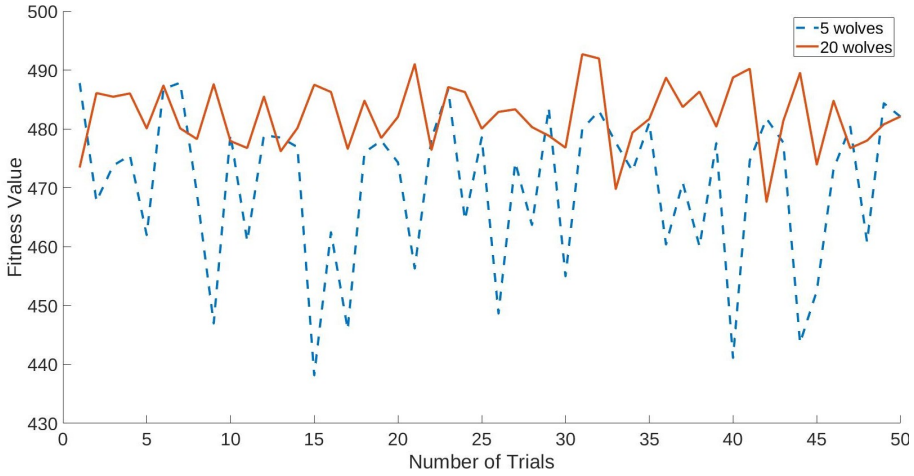

FIGURE 6.39: The trend of the fitness value in 100 trials with a population of 5 and 20 wolves

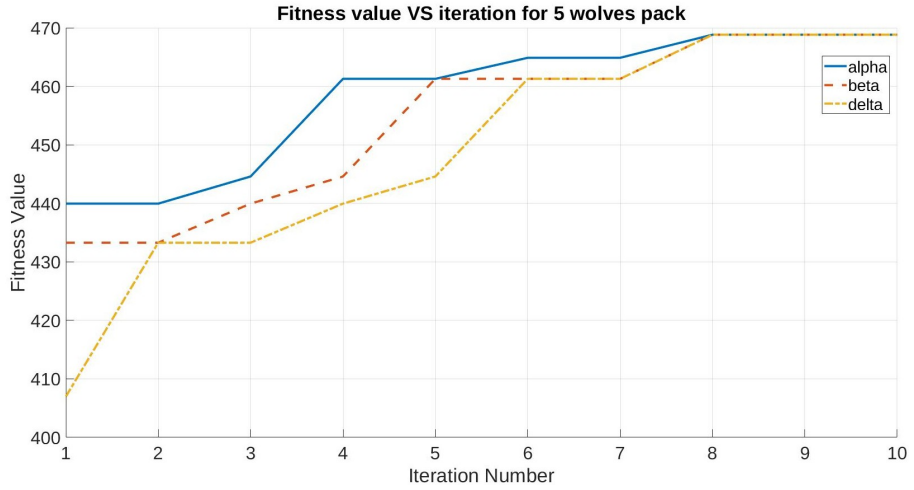

FIGURE 6.40: Fitness value of the alpha beta and delta wolves with a wolf pack size of 5

In 6.42 it is shown the average fitness value of 50 trial of the algorithm with different wolves size pack. Increasing the number of wolves that moves inside the solution space the algorithm is able to find better solution for the same configuration problem. To evaluate the algorithm performances, we assessed the solutions by comparing its results with the results achieved with an exhaustive search. In Fig.6.43, the trend of the proposed bio-inspired algorithm versus the optimum solution considering the same traffic pattern is shown. In particular, studying the simulations we did, we carried out the fitness values versus the maximum available platoon formation size. The size of platoon represents a constraint for our research algorithm, and an admissible solution cannot present a platoon with a higher dimension. But it is also possible to note that with the same configuration, increasing the number of available place in the platoon GWO needs more wolves in the packs to achieve better results.

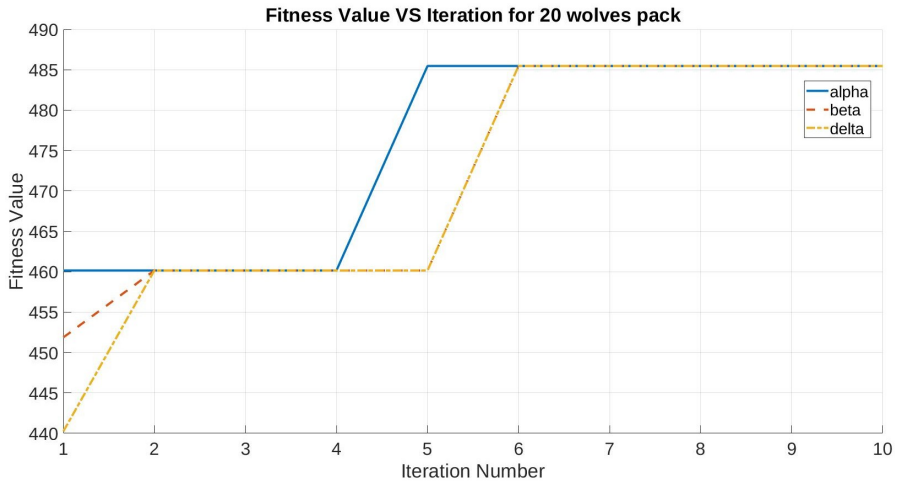

FIGURE 6.41: Fitness value of the alpha beta and delta wolves with a wolf pack size of 20

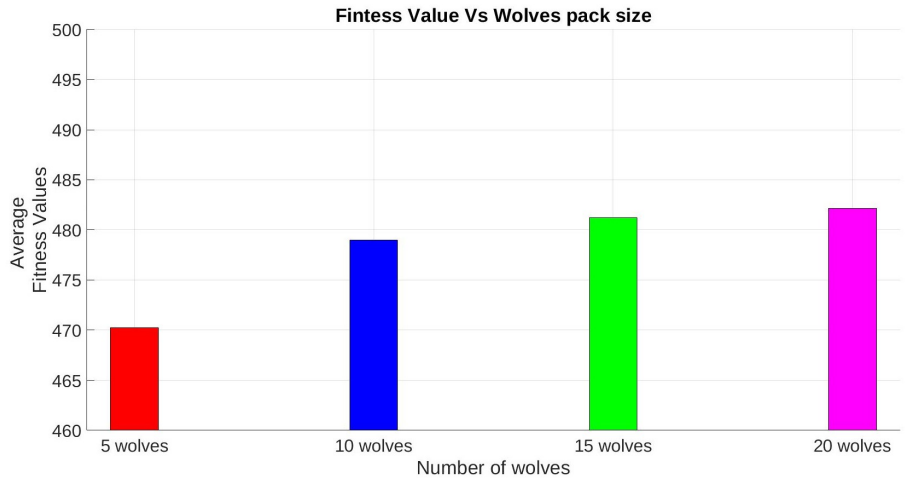

FIGURE 6.42: Average fitness value of 50 trials vs wolf pack size

However, the final solution is closer to the optimal one in both cases.

In the simulation campaigns, we performed we measure that to take advantages of platooning the time spent in the platoon by a vehicle should be higher than the 60% of the overall travelling time. Considering this value, we reported in the Fig.6.44 the value obtained in case of local search versus the global search, which is the mechanism we use in this proposal, is shown. It is possible to note how the global search performs better because the primary node adopts an optimisation algorithm for finding the better platoon than a local one.

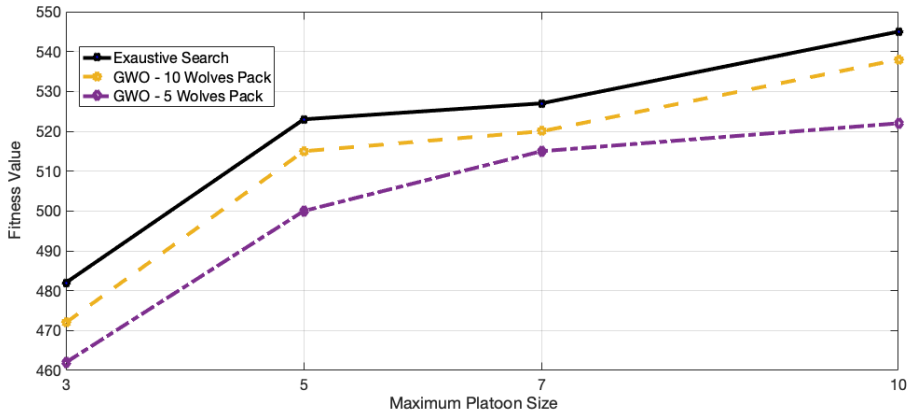

FIGURE 6.43: Fitness value obtained by the exhaustive search and GWO instances. This measure how far is the sub-optimal solution from the optimal one.

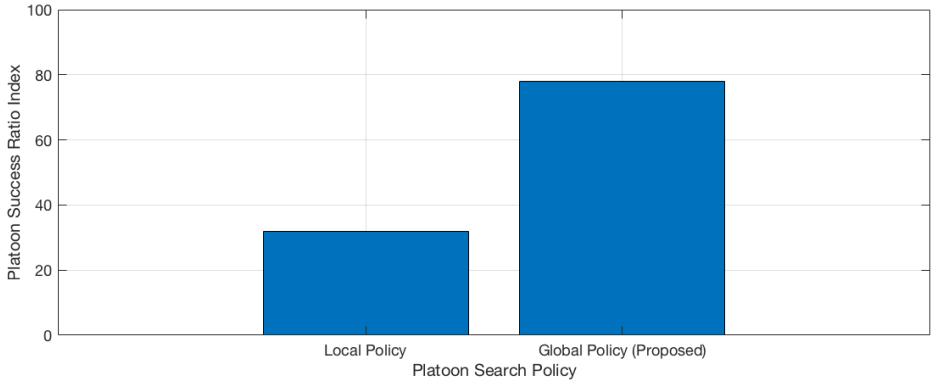

FIGURE 6.44: Platoon Success Ratio Index versus local or global policy adopted in platoon formation by the system.

Chapter 7

Conclusions

During these three years of PhD I have focused my researches on studying and improving communications protocols and techniques in vehicular environment. I have started with the study of classic VANET protocols trying to improve the transfer content between vehicles and ITS management of traffic flows inside the city. Communication is one of the key factors that can really improve quality of life in various fields of research and especially inside the vehicular environment. The rise of the smart objects and of the IoT thanks to the advancement in the technology of communication field is happening also inside other areas. Since a few years the interest about IoV is relevantly rising and the research community is trying to find new solutions to enhance safety and quality of life through communication. In order to deal with the autonomous vehicle I had to study in deep the various network protocols and architectures used in vehicular environment. Another very interesting research area that will deeply influence the future of vehicles is the investigation of new propulsion engines and energy sources. For this reason I have invested a considerable amount of time studying electric models in current state of the art simulators and implementing this model in a java ad-hoc simulator, embedding out of the box the integration with real maps and taking into consideration the altitude and the slope coefficient. I have started by creating my electric consumption model on a java simulator written ad-hoc to make practice with the simulation environment. After, getting enough experience, I have decided to move on tools and simulators that have been already proven worthy and solid by the academic community. Battery life and electric vehicles autonomy are deeply influenced by various aspects. The use of on-board sensors is spreading really fast because they are needed to realize several applications to enhance safety and quality of life, by automating or monitoring several activities. The presence of a large set of sensors is mandatory when considering autonomous vehicles that will not be excluded by the evolution of the propulsion vehicle system and, for this reason, will be battery powered too. To better depict a future scenario is necessary to take into account also the energy consumption of various sensors that will be equipped on autonomous vehicles and influence the battery autonomy. The sensors consumption model has been firstly implemented on SUMO / VEINS as a device attached to the vehicle interfacing the battery device

to update its status and then transported on PLEXE to apply it in a autonomous vehicle platoon scenario. The results have shown that an heavy sensors configuration has a deep impact on vehicle autonomy and that some countermeasures need to be developed in order to extend battery life. A platoon application has been proposed to mitigate the sensors consumption thanks to the standby of some of the sensors, when inside a platoon formation, using a CACC system. Also the selection of the best platoon for each vehicle is very difficult to make. From a centralized point of view the problem can be described as an assignment problem, in which each vehicle must be assigned to a platoon. For this reason, an algorithm inspired by the grey wolf pack hunting routine has been proposed to find a good enough assignment for each vehicle. This algorithm tries to emulate the way that wolves moves when hunting in packs. The movements of the entire pack is influenced by each wolf, but a great influence is exerted by α, β and δ specimens that are used to estimate the direction of the movement vector inside the solutions space. I'm inclined to extend this study in the near future to improve CACC protocols and to tackle other issues that can be related to autonomous vehicles.

7.1 Future Works

In the near future I am going to investigate further the topic of autonomous vehicles and how to simulate an urban environment in which autonomous and normal vehicles move in. I will expect interesting results about the interaction of autonomous vehicles which always respect driving rules and normal vehicles that are driven by humans that sometimes don't respect them. I want also to deeply investigate the topic of platoons using, and comparing, different approaches and algorithms to manage the platoon creation and formation. I think that the comparison between the gray wolf optimization technique and other heuristic or meta-heuristic algorithms, such as simulated annealing and genetic algorithms. I'm also interested in the new simulation tools that allow a very rich description of a scenario, in terms of very fine details. These kinds of simulation tools are very difficult to use and model because have to take into account a very huge number parameters. For example, the platooning simulation tool that I've used to describe the platooning creation and formation management is a very complex tool, in which even the attrition of internal components is taken into account to compute each time step speed and consumed energy. These tools have also a different time scale because the time between two consecutive events is really small. In the end, I think I will also try to integrate all the different protocols from IoT and VANET inside a comprehensive architecture to exploit their strengths and mitigate their weaknesses. I've waited so long before starting this idea because I've thought I needed a very deep comprehension of all these kinds of technology and protocols.

Chapter 8

Publications

- **Cooperative Video-Surveillance Framework in Internet of Things (IoT) Domain**
  AF Santamaria, P Raimondo, N Palmieri, M Tropea, F De Rango
  The Internet of Things for Smart Urban Ecosystems, 305-331

- **The Impact of the Roads' Slope Coefficient in a Vehicular Energy Model**
  AF Santamaria, P Raimondo, G Cotugno, F De Rango
  2018 IEEE/ACM 22nd International Symposium on Distributed Simulation and Real Time Applications (DS-RT)

- **MDMC: A WSN cooperative protocol for Minimizing the Data Distortion**
  F De Rango, AF Santamaria, P Fazio, P Raimondo
  2018 11th IFIP Wireless and Mobile Networking Conference (WMNC), 1-7

- **A decentralized ITS architecture for efficient distribution of traffic task management**
  AF Santamaria, M Tropea, P Fazio, P Raimondo, F De Rango, M Voznak
  2018 11th IFIP Wireless and Mobile Networking Conference (WMNC), 1-5

- **A real IoT device deployment for e-Health applications under lightweight communication protocols, activity classifier and edge data filtering**
  AF Santamaria, F De Rango, A Serianni, P Raimondo
  Computer Communications 128, 60-73

- **USE OF PARTICLE SWARM OPTIMIZATION ALGORITHM FOR DIGITALIZED SINEWAVE SIGNAL PARAMETERS ESTIMATION**
  P Raimondo
  International Journal of Computing 16 (4), 203-209

- **A distributed architecture for eHealth applications based on Internet of Things (IoT)**
  AF Santamaria, F De Rango, A Serianni, P Raimondo
  Journal of Network and Computer Application

- **A two stages fuzzy logic approach for Internet of Things (IoT) wearable devices**

AF Santamaria, P Raimondo, F De Rango, A Serianni
Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016 IEEE

- **Smart wearable device for health monitoring in the internet of things (IoT) domain**
AF Santamaria, A Serianni, P Raimondo, F De Rango, M Froio
Proceedings of the summer computer simulation conference, 36

- **A fast and scalable content transfer protocol (FSCTP) for VANET based architecture**
AF Santamaria, F Scala, C Sottile, M Tropea, P Raimondo
Unmanned Systems Technology XVIII 9837, 98370T

- **An efficient traffic management protocol based on IEEE802. 11p standard**
AF Santamaria, C Sottile, A Lupia, P Raimondo
Performance Evaluation of Computer and Telecommunication Systems (SPECTS)

- **Smart sensing to drive real-time loads scheduling algorithm in a domotic architecture**
AF Santamaria, P Raimondo, F De Rango, A Vaccaro
Wireless Sensing, Localization, and Processing IX 9103, 91030K

Bibliography

[1] S. N. Shinde and S. Chorage, "Unmanned ground vehicle", *International Journal of Advanced Engineering, Management and Science*, vol. 2, no. 10, 2016.

[2] R. Collinson, "Unmanned air vehicles", in *Introduction to Avionics Systems*, Springer, 2011, pp. 489–498.

[3] T. Hardy and G. Barlow, "Unmanned underwater vehicle (uuv) deployment and retrieval considerations for submarines", in *International Naval Engineering Conference and Exhibition 2008*, 2008.

[4] T. Litman, *Autonomous vehicle implementation predictions*. Victoria Transport Policy Institute Victoria, Canada, 2017.

[5] J. M. Anderson, K. Nidhi, K. D. Stanley, P. Sorensen, C. Samaras, and O. A. Oluwatola, *Autonomous vehicle technology: A guide for policymakers*. Rand Corporation, 2014.

[6] *Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems*. DOI: 10.4271/j3016_201401. [Online]. Available: https://doi.org/10.4271/j3016_201401.

[7] A. Festag, "Standards for vehicular communication—from ieee 802.11p to 5g", *e & i Elektrotechnik und Informationstechnik*, vol. 132, no. 7, pp. 409–416, 2015, ISSN: 1613-7620. DOI: 10.1007/s00502-015-0343-0. [Online]. Available: https://doi.org/10.1007/s00502-015-0343-0.

[8] G. Dimitrakopoulos and P. Demestichas, "Intelligent transportation systems", *IEEE Vehicular Technology Magazine*, vol. 5, no. 1, pp. 77–84, 2010.

[9] P. Ranjan and K. K. Ahirwar, "Comparative study of vanet and manet routing protocols", in *Proc. of the International Conference on Advanced Computing and Communication Technologies (ACCT 2011)*, 2011, pp. 517–523.

[10] P. Papadimitratos, A. De La Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation", *IEEE communications magazine*, vol. 47, no. 11, 2009.

[11] K. C. Lee, U. Lee, and M. Gerla, "Survey of routing protocols in vehicular ad hoc networks", in *Advances in vehicular ad-hoc networks: Developments and challenges*, IGI Global, 2010, pp. 149–170.

[12] M. Benamar, N. Benamar, K. D. Singh, and D. El Ouadghiri, "Recent study of routing protocols in vanet: Survey and taxonomy", in *WVNT 1st International Workshop on Vehicular Networks and Telematics*, 2013.

[13] J. B. Kenney, "Dedicated short-range communications (dsrc) standards in the united states", *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[14] B. Paul, M. Ibrahim, M. Bikas, and A. Naser, "Vanet routing protocols: Pros and cons", *arXiv preprint arXiv:1204.1201*, 2012.

[15] S. Yousefi, M. S. Mousavi, and M. Fathy, "Vehicular ad hoc networks (vanets): Challenges and perspectives", in *ITS Telecommunications Proceedings, 2006 6th International Conference on*, IEEE, 2006, pp. 761–766.

[16] I. Wahid, A. A. Ikram, M. Ahmad, S. Ali, and A. Ali, "State of the art routing protocols in vanets: A review", *Procedia Computer Science*, vol. 130, pp. 689 – 694, 2018, The 9th International Conference on Ambient Systems, Networks and Technologies (ANT 2018) / The 8th International Conference on Sustainable Energy Information Technology (SEIT-2018) / Affiliated Workshops, ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2018.04.121. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050918304836.

[17] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos, *et al.*, "Named data networking (ndn) project", *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, vol. 157, p. 158, 2010.

[18] E. Kalogeiton, T. Kolonko, and T. Braun, "A multihop and multipath routing protocol using ndn for vanets", in *2017 16th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, 2017, pp. 1–8. DOI: 10.1109/MedHocNet.2017.8001640.

[19] A. Abdelgader and W. Lenan, "The physical layer of the ieee 802.11 p wave communication standard: The specifications and challenges", in *Proceedings of the world congress on engineering and computer science*, vol. 2, 2014, p. 71.

[20] A. F. Santamaria, C. Sottile, A. Lupia, and P. Raimondo, "An efficient traffic management protocol based on ieee802. 11p standard", in *Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2014), International Symposium on*, IEEE, 2014, pp. 634–641.

[21] W. Liang, Z. Li, H. Zhang, S. Wang, and R. Bie, "Vehicular ad hoc networks: Architectures, research issues, methodologies, challenges, and trends", *International Journal of Distributed Sensor Networks*, vol. 11, no. 8, p. 745 303, 2015.

[22] A. Santamaria, F Scala, C Sottile, M Tropea, and P Raimondo, "A fast and scalable content transfer protocol (fsctp) for vanet based architecture", in *Unmanned Systems Technology XVIII*, International Society for Optics and Photonics, vol. 9837, 2016, 98370T.

[23] M. Gerla, E. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds", in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, 2014, pp. 241–246. DOI: 10.1109/WF-IoT.2014.6803166.

[24] M. Gerla, E.-K. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds", in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, IEEE, 2014, pp. 241–246.

[25] A. Banks and R. Gupta, "Mqtt version 3.1. 1", *OASIS standard*, vol. 29, 2014.

[26] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (coap)", Tech. Rep., 2014.

[27] C. Barberis and G. Malnati, "Design and evaluation of a collaborative system for content diffusion and retrieval in vehicular networks", *IEEE Transactions on Consumer Electronics*, vol. 57, no. 1, pp. 105–112, 2011, ISSN: 0098-3063. DOI: 10.1109/TCE.2011.5735489.

[28] D. Paluszczyszyn, M. Al-Doori, W. Manning, D. Elizondo, and E. Goodyer, "Range extended engine management system for electric vehicles: Control design process", Sep. 2014.

[29] A. Santamaria, F De Rango, P Raimondo, and A Serianni, *M2m iot communication system in iov domain to promote a safer and eco-friendly driving style*, 2018.

[30] N. D. Caro, W. Colitti, K. Steenhaut, G. Mangino, and G. Reali, "Comparison of two lightweight protocols for smartphone-based sensing", in *2013 IEEE 20th Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*, 2013, pp. 1–6. DOI: 10.1109/SCVT.2013.6735994.

[31] Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis", *IEEE Transactions on Computers*, vol. C-26, no. 12, pp. 1182–1191, 1977, ISSN: 0018-9340. DOI: 10.1109/TC.1977.1674779.

[32] M. Shafi, A. F. Molisch, P. J. Smith, T. Haustein, P. Zhu, P. Silva, F. Tufvesson, A. Benjebbour, and G. Wunder, "5g: A tutorial overview of standards, trials, challenges, deployment, and practice", *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 6, pp. 1201–1221, 2017.

[33] A. Gupta and R. K. Jha, "A survey of 5g network: Architecture and emerging technologies", *IEEE access*, vol. 3, pp. 1206–1232, 2015.

[34] S.-Y. Lien, C.-C. Chien, G. S.-T. Liu, H.-L. Tsai, R. Li, and Y. J. Wang, "Enhanced lte device-to-device proximity services", *IEEE Communications Magazine*, vol. 54, no. 12, pp. 174–182, 2016.

[35] S. A. A. Shah, E. Ahmed, M. Imran, and S. Zeadally, "5g for vehicular communications", *IEEE Communications Magazine*, vol. 56, no. 1, pp. 111–117, 2018, ISSN: 0163-6804. DOI: 10.1109/MCOM.2018.1700467.

[36] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset", *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[37] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo, "Development of autonomous car—part i: Distributed system architecture and development process", *IEEE Transactions on Industrial Electronics*, vol. 61, no. 12, pp. 7131–7140, 2014.

[38] D. Pomerleau, "Visibility estimation from a moving vehicle using the ralph vision system", in *Intelligent Transportation System, 1997. ITSC'97., IEEE Conference on*, IEEE, 1997, pp. 906–911.

[39] A. Santamaria, P Raimondo, N Palmieri, M Tropea, and F De Rango, "Cooperative video-surveillance framework in internet of things (iot) domain", in *The Internet of Things for Smart Urban Ecosystems*, Springer, 2019, pp. 305–331.

[40] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[41] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis", *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1773–1795, 2013, ISSN: 1524-9050. DOI: 10.1109/TITS.2013.2266661.

[42] J. D. Alonso, E. R. Vidal, A. Rotter, and M. Muhlenberg, "Lane-change decision aid system based on motion-driven vehicle tracking", *IEEE Transactions on Vehicular Technology*, vol. 57, no. 5, pp. 2736–2746, 2008.

[43] J. Dickmann, J. Klappstein, M. Hahn, N. Appenrodt, H. Bloecher, K. Werber, and A. Sailer, ""automotive radar the key technology for autonomous driving: From detection and ranging to environmental understanding"", in *2016 IEEE Radar Conference (RadarConf)*, 2016, pp. 1–6. DOI: 10.1109/RADAR.2016.7485214.

[44] R. Schoeneburg and T. Breitling, "Enhancement of active and passiv e safety by future pre-safe® systems", in *Proceedings: International Technical Conference on the Enhanced Safety of Vehicles*, National Highway Traffic Safety Administration, vol. 2005, 2005, 9p–9p.

[45] H. Liu, Y. Liu, X. Gu, Y. Wu, F. Qu, and L. Huang, "A deep-learning based multi-modality sensor calibration method for usv", in *2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM)*, IEEE, 2018, pp. 1–5.

[46] W.-l. ZHANG, B. Wu, W.-f. LI, and X.-k. LAI, "Discussion on development trend of battery electric vehicles in china and its energy supply mode [j]", *Power System Technology*, vol. 33, no. 4, pp. 1–5, 2009.

[47] B. Schwarz, "Lidar: Mapping the world in 3d", *Nature Photonics*, vol. 4, no. 7, p. 429, 2010.

[48] G. Heritage and A. Large, *Laser scanning for the environmental sciences*. John Wiley & Sons, 2009.

[49] I. I.H.M. A. Zermas Dimitris, "Lidar object detection system for automated vehicles", pat. 10 031 231, 2018. [Online]. Available: http://www.freepatentsonline.com/10031231.html.

[50] Y. Zhang, J. Wang, X. Wang, and J. M. Dolan, "Road-segmentation-based curb detection method for self-driving via a 3d-lidar sensor", *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2018, ISSN: 1524-9050. DOI: 10.1109/TITS.2018.2789462.

[51] A. El-Rabbany, *Introduction to GPS: the global positioning system*. Artech house, 2002.

[52] "An architectural blueprint for autonomic computing.", *Autonomic Computing White Paper*, 2006.

[53] S. D. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, D. Rus, and M. H. Ang, "Perception, planning, control, and coordination for autonomous vehicles", *Machines*, vol. 5, no. 1, p. 6, 2017.

[54] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[55] H. Huo, M. Wang, L. Johnson, and D. He, "Projection of chinese motor vehicle growth, oil demand, and co2 emissions through 2050", *Transportation Research Record: Journal of the Transportation Research Board*, no. 2038, pp. 69–77, 2007.

[56] A. M. Andwari, A. Pesiridis, S. Rajoo, R. Martinez-Botas, and V. Esfahanian, "A review of battery electric vehicle technology and readiness levels", *Renewable and Sustainable Energy Reviews*, vol. 78, pp. 414 –430, 2017, ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2017.03.138. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1364032117306251.

[57] K. Rajashekara, "Present status and future trends in electric vehicle propulsion technologies", *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 1, no. 1, pp. 3–10, 2013, ISSN: 2168-6777. DOI: 10.1109/JESTPE.2013.2259614.

[58] P. Sharma, "Discrete-event simulation", *Int. J. Sci. Technol. Res*, vol. 4, no. 04, pp. 136–140, 2015.

[59] A. Beck, *Simulation: The practice of model development and use*, 2008.

[60] Y. Hori, "Future vehicle driven by electricity and control-research on four-wheel-motored" uot electric march ii"", *IEEE Transactions on Industrial Electronics*, vol. 51, no. 5, pp. 954–962, 2004.

[61] G. Maggetto and J. V. Mierlo, "Electric and electric hybrid vehicle technology: A survey", in *IEE Seminar Electric, Hybrid and Fuel Cell Vehicles (Ref. No. 2000/050)*, 2000, pp. 1/1–111. DOI: 10.1049/ic:20000261.

[62] D. Krajzewicz, "Traffic simulation with sumo–simulation of urban mobility", in *Fundamentals of traffic simulation*, Springer, 2010, pp. 269–293.

[63] F. K. Karnadi, Z. H. Mo, and K.-c. Lan, "Rapid generation of realistic mobility models for vanet", in *Wireless communications and networking conference, 2007. WCNC 2007. IEEE*, IEEE, 2007, pp. 2506–2511.

[64] 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*, vol. 5, no. 3&4, 2012.

[65] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, "Traci: An interface for coupling road traffic and network simulators", in *Proceedings of the 11th communications and networking simulation symposium*, ACM, 2008, pp. 155–163.

[66] M. Pursula, "Simulation of traffic systems-an overview", *Journal of geographic information and decision analysis*, vol. 3, no. 1, pp. 1–8, 1999.

[67] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility", in *The Third International Conference on Advances in System Simulation (SIMUL 2011), Barcelona, Spain*, vol. 42, 2011.

[68] S. Krauß, "Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics", PhD thesis, Universitat zu Koln., 1998.

[69] J. Song, Y. Wu, Z. Xu, and X. Lin, "Research on car-following model based on sumo", in *Advanced Infocomm Technology (ICAIT), 2014 IEEE 7th International Conference on*, IEEE, 2014, pp. 47–55.

[70] SUMO. (). Networks/sumo road networks - sumo. (Accessed on 09/26/2018).

[71] USGS. (2018). Earthexplorer - home, [Online]. Available: https://earthexplorer.usgs.gov/ (visited on 09/27/2018).

[72] HBEFA. (2018). Handbook emission factors for road transport, [Online]. Available: http://www.hbefa.net/e/index.html/ (visited on 09/27/2018).

[73] T. Kurczveil, P. Á. López, and E. Schnieder, "Implementation of an energy model and a charging infrastructure in sumo", in *Simulation of Urban MObility User Conference*, Springer, 2013, pp. 33–43.

[74] P. Raimondo, A. F. Santamaria, F. D. Rango, and A. Bosco, "A vehicular traffic simulator model for evaluating electrical vehicle performances in a configurable mobility scenario", in *Proceedings of 8th International Conference on Simulation and Modeling Methodologies, Technologies and Applications - Volume 1: SIMULTECH,*, INSTICC, SciTePress, 2018, pp. 198–205, ISBN: 978-989-758-323-0. DOI: 10.5220/0006919301980205.

[75] OpenStreetMap. (2018). Open source maps, [Online]. Available: https://www.openstreetmap.org/ (visited on 09/18/2018).

[76] P. Gipps, "A behavioural car-following model for computer simulation", *Transportation Research Part B: Methodological*, vol. 15, no. 2, 105–111, 1981. DOI: 10.1016/0191-2615(81)90037-0.

[77] R. Maia, M. Silva, R. Araújo, and U. Nunes, "Electric vehicle simulator for energy consumption studies in electric mobility systems", in *2011 IEEE Forum on Integrated and Sustainable Transportation Systems*, 2011, pp. 227–232. DOI: 10.1109/FISTS.2011.5973655.

[78] P. Kavathekar and Y. Chen, "Vehicle platooning: A brief survey and categorization", in *ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, American Society of Mechanical Engineers, 2011, pp. 829–845.

[79] J. Luo and J.-P. Hubaux, "A survey of inter-vehicle communication", Tech. Rep., 2004.

[80] R. Bishop, "A survey of intelligent vehicle application‖", in *Proceedings of IEEE intelligent Vehicles Symposium*, 2000.

[81] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, "Network simulations with the ns-3 simulator", *SIGCOMM demonstration*, vol. 14, no. 14, p. 527, 2008.

[82] opnet. (). Opnet technologies – network simulator | riverbed, [Online]. Available: https://www.riverbed.com/gb/products/steelcentral/opnet.html?redirect=opnet (visited on 10/08/2018).

[83] R. Nagel and S. Eichler, "Efficient and realistic mobility and channel modeling for vanet scenarios using omnet++ and inet-framework", in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, p. 89.

[84] A. Varga, "Using the omnet++ discrete event simulation system in education", *IEEE Transactions on Education*, vol. 42, no. 4, 11–pp, 1999.

[85] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment", in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, p. 60.

[86] A. Varga, "Omnet++", in *Modeling and tools for network simulation*, Springer, 2010, pp. 35–59.

[87] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved ivc analysis", *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 3–15, 2011.

[88] S. Gupte and M. Younis, "Vehicular Networking for Intelligent and Autonomous Traffic Management", in *IEEE International Conference on Communications (ICC 2012)*, Ottawa, Canada: IEEE, 2012, pp. 5306–5310. DOI: 10.1109/ICC.2012.6364617.

[89] C. Sommer and F. Dressler, "Progressing Toward Realistic Mobility Models in VANET Simulations", *IEEE Communications Magazine*, vol. 46, no. 11, pp. 132–137, 2008. DOI: 10.1109/MCOM.2008.4689256.

[90] S. Schellenberg, R. Berndt, R. German, and D. Eckhoff, "Evaluating the Electrification of Vehicle Fleets Using the Veins Framework", arXiv, Hamburg, Germany, Tech. Rep. 1409.1003, 2014. [Online]. Available: http://arxiv.org/pdf/1409.1003.pdf.

[91] S. E. Shladover, "Path at 20—history and major milestones", *IEEE Transactions on intelligent transportation systems*, vol. 8, no. 4, pp. 584–592, 2007.

[92] C. Bergenhem, Q. Huang, A. Benmimoun, and T. Robinson, "Challenges of platooning on public motorways", in *17th world congress on intelligent transport systems*, 2010, pp. 1–12.

[93] *Plexe*, http://plexe.car2x.org/, (Accessed on 11/15/2018).

[94] M. Brackstone and M. McDonald, "Car-following: A historical review", *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 2, no. 4, pp. 181–196, 1999.

[95] M. Segata, "Platooning in sumo: An open source implementation",

[96] B. Van Arem, C. J. Van Driel, and R. Visser, "The impact of cooperative adaptive cruise control on traffic-flow characteristics", *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 4, pp. 429–436, 2006.

[97] D Swaroop and J. K. Hedrick, "String stability of interconnected systems", *IEEE transactions on automatic control*, vol. 41, no. 3, pp. 349–357, 1996.

[98] C. Bergenhem, S. Shladover, E. Coelingh, C. Englund, and S. Tsugawa, "Overview of platooning systems", in *Proceedings of the 19th ITS World Congress, Oct 22-26, Vienna, Austria (2012)*, 2012.

[99] Y. J. Li, "An overview of the dsrc/wave technology", in *International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, Springer, 2010, pp. 544–558.

[100] S. Santini, A. Salvi, A. S. Valente, A. Pescapé, M. Segata, and R. L. Cigno, "A consensus-based approach for platooning with intervehicular communications and its validation in realistic scenarios", *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 1985–1999, 2017.

[101] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer", *Advances in engineering software*, vol. 69, pp. 46–61, 2014.

[102] A. A. Heidari and P. Pahlavani, "An efficient modified grey wolf optimizer with lévy flight for optimization tasks", *Applied Soft Computing*, vol. 60, pp. 115–134, 2017.

[103] Y. Gu and R. L. Grossman, "Udt: Udp-based data transfer for high-speed wide area networks", *Computer Networks*, vol. 51, no. 7, pp. 1777–1799, 2007.

[104] A. Aijaz and A. H. Aghvami, "Cognitive machine-to-machine communications for internet-of-things: A protocol stack perspective", *IEEE Internet of Things Journal*, vol. 2, no. 2, pp. 103–112, 2015.